

UNITED STATES PATENT AND TRADEMARK OFFICE

BEFORE THE PATENT TRIAL AND APPEAL BOARD

AVIDBOTS CORP. and AVIDBOTS USA CORP.,

Petitioners,

v.

BRAIN CORPORATION,
Patent Owner.

Case No. IPR2025-01603
U.S. Patent No. 10,823,576

**DECLARATION OF DR. MARY K. BOLIN
IN SUPPORT OF PETITION FOR *INTER PARTES* REVIEW
OF U.S. PATENT NO. 10,823,576**

I, Mary K. Bolin, declare as follows:

1. I am Emeritus Professor at the University of Nebraska, Lincoln Libraries. I have held the following positions: Professor and Catalog and Metadata Librarian, Digital Initiatives and Special Collections Department, University of Nebraska—Lincoln Libraries (2015–2020); Professor and Chair, Discovery and Resource Management and Coordinator of Collection Development, University of Nebraska, Lincoln Libraries (2004–2015); San Jose State University School of Information: Lecturer (2008–present); Head of the Technical Services Department, University of Idaho Library (1993–2003; promoted to Professor in 1999); Head of the Cataloging Department, University of Idaho Library (1986–1993; promoted to Associate Professor and tenured in 1991); and Head of Records Maintenance Section, Cataloging Department, University of Georgia Libraries (1981–1986).

2. Over the course of my work as a librarian, professor, researcher, student research supervisor, and author of numerous scholarly papers and other publications, I have had extensive experience with cataloging records and online library management systems built around Machine-Readable Cataloging (MARC) standards. I also have substantial experience in authenticating printed documents and establishing the date when they were accessible to ordinarily skilled researchers.

3. Attachment A is my full resume.

4. I have been retained by counsel for Avidbots Corp. (“Petitioner”) to evaluate, authenticate, and establish the date of public availability of documents relevant to the above-captioned Petition for *Inter Partes* Review. For my time performing this service, I am being paid my usual hourly fee. I have no financial interest in this action, or affiliation with Petitioner. My compensation is not dependent upon the outcome of, or my testimony in, this proceeding or any other proceeding, including before the U.S. Patent Office.

A. Documents of Interest

5. The documents that are the subject of this Declaration are the following:

- **Document 1.** Olson, E. B. (2008). Robust and efficient robotic mapping. Massachusetts Institute of Technology.
- **Document 2.** Springer Handbook of Robotics” (2d ed.) (ISBN 3319325507) chapter 46: “Simultaneous Localization and Mapping.” Cyrill Stachniss, John J. Leonard & Sebastian Thrun.
- **Document 3.** Springer Handbook of Robotics” (2d ed.) (ISBN 3319325507) chapter 65: “Domestic Robots.” Erwin Prassler, Mario E. Munich, Paolo Prijanian, and Kazuhiro Kosuge.

B. Methodology

6. I searched for these documents in the OCLC database, Google Scholar, and several subscription databases such as IEEE Xplore and ACM Digital Library. I also searched the library catalogs of institutions such as the University of Nebraska—Lincoln and other large research libraries. My goal was to find a record in one of these databases that contained a date indicating when these documents were available to the public.

7. *Use of MARC Records.* The “Machine Readable Cataloging” standard, also known as “MARC,” is an industry-wide standard method of organizing library catalog information. The Library of Congress first developed MARC in the 1960s. A MARC-compatible library is one that has a catalog consisting of individual MARC records for each of its works. Today, MARC is the primary communication protocol for the transfer and storage of bibliographic metadata in libraries.

8. OCLC is a database of MARC records used by libraries worldwide. When an OCLC-participating institution acquires a work, it creates a MARC record for this work in its computer catalog system, and OCLC automatically supplies the date of creation. The MARC record creation date reflects the date on which (or shortly after) the item was first acquired and cataloged. Once the MARC record is created by a cataloger at an OCLC-participating member institution, it is

instantaneously available to any OCLC-participating members and, therefore, available to the public.

9. A MARC record comprises several fields, each of which contains specific data about the work. Each field is identified by a standardized, unique, three-digit code corresponding to the type of data that follows. For example, a work's title is recorded in field 245, the primary author of the work is recorded in field 100, an item's International Standard Book Number ("ISBN") is recorded in field 020, an item's Library of Congress call number is recorded in field 050, and the publication date is recorded in field 260 or 264 under the subfield "c." If a work is a periodical, then its publication frequency is recorded in field 310, and the publication dates (*e.g.*, the first and last publication) are recorded in field 362, which is also referred to as the enumeration/chronology field. The library that created the record is recorded in field 040 in subfield "a" with a unique library code. When viewing the MARC record online via OCLC's Worldshare Record Manager database, hovering over this code with the mouse reveals the name of the library. I used this method of "mousing over" the library codes in the OCLC database to identify the originating library for all of the MARC records in this Declaration, as well as the "display holdings" command to see a list of libraries that own the item. In the MARC records in the OCLC Worldshare Record Manager database, MARC

field 040 contains the OCLC symbols of institutions transcribing, creating, and modifying the record.

10. OCLC was created “to establish, maintain and operate a computerized library network and to promote the evolution of library use, of libraries themselves, and of librarianship, and to provide processes and products for the benefit of library users and libraries, including such objectives as increasing availability of library resources to individual library patrons and reducing the rate of rise of library per-unit costs, all for the fundamental public purpose of furthering ease of access to and use of the ever-expanding body of worldwide scientific, literary and educational knowledge and information.”¹ Among other services, OCLC and its members are responsible for maintaining the WorldCat database (<http://www.worldcat.org/>), which is used by independent and institutional libraries throughout the world. OCLC is the international cooperative to which libraries catalog records and others pay to access MARC records. The MARC records referenced in this Declaration, extracted from the OCLC database, were generated in the ordinary course of

¹ Third Article, Amended Articles of Incorporation of OCLC Online Computer Library Center, Incorporated (available at <http://www.oclc.org/en-US/membership/councils.html>).

business. Further, I personally located and downloaded each of these records associated with the document on which this Declaration focuses.

11. OCLC provides its members online access to MARC records through its OCLC Worldshare Record Manager database. When an OCLC member institution acquires a work, it creates a MARC record for this work in its computer catalog system in the ordinary course of its business. MARC records created at the Library of Congress are batch-loaded into the OCLC database through a subscription to MARC Distribution Services daily or weekly. Once the MARC record is created by a cataloger at an OCLC member institution or is tape-loaded from the Library of Congress, it is then made available to any other OCLC members online, and, therefore, made available to the public.

12. When an OCLC member institution creates a new record, OCLC automatically supplies the date of creation for that record. The MARC record creation date reflects the date on which, or shortly after which, the item was first acquired or cataloged.

13. The date of creation for the MARC record appears in the fixed field (008), characters 00 through 05. Field 005 is a 16 digit field that records the date characters `yyyymmdd` including 4 for the year, 2 for the month, and 2 for the day. The date entered on file never changes. These dates are automatically supplied by the OCLC system and cannot be changed by the user. When the MARC record is

created in OCLC, it is available throughout the world to other libraries and in WorldCat, the public version of OCLC Worldshare Record Manager. Thus, the date of creation of the MARC record by a cataloger at an OCLC member institution reflects when the underlying item is accessible to the public.

14. MARC records also include several fields that include subject matter classification information. An overview of MARC record fields is available through the Library of Congress.² For example, 6XX fields are termed “Subject Access Fields.”³ Among these, for example, is the 650 field, which is the “Subject Added Entry – Topical Term” field.⁴ The 650 field is a “[s]ubject added entry in which the entry element is a topical term.” *Id.* The 650 field entries “are assigned to a bibliographic record to provide access according to generally accepted thesaurus-building rules (e.g., *Library of Congress Subject Headings* (LCSH), *Medical Subject Headings* (MeSH)).” *Id.* Thus, a researcher can easily discover material relevant a topic of interest with a search using the terms employed in the MARC Fields 6XX.

15. For Documents 1-3, I performed a search of OCLC (www.oclc.org), which is a large shared database used by libraries around the world to create and

² See <http://www.loc.gov/marc/bibliographic/>.

³ See <http://www.loc.gov/marc/bibliographic/bd6xx.html>.

⁴ See <http://www.loc.gov/marc/bibliographic/bd650.html>.

exchange cataloging data, using MARC format, which is a universal communications format for that data.

16. Using the records from OCLC, I determined when each record was created, and which libraries own the items. I then visited the website of the library or libraries that own the items, searched their library catalog, and determined when the items were added to the library's collection.

17. I am familiar with many online catalogs, databases, and search engines. In preparing for this Declaration, I used authoritative information systems such as the above-mentioned WorldCat (<https://www.worldcat.org>), the online catalog of the Library of Congress (<https://catalog.loc.gov>), shared research databases such as ResearchGate and CiteSeerX (formerly known as CiteSeer), the online catalog of a number of large research libraries such as the University of Nebraska—Lincoln, IEEEExplore, and Google Scholar search for records. These records are identified and discussed in this Declaration.

18. The IEEE Xplore⁵ research database is a vital resource for undergraduate and graduate students, researchers, scientists, and the public, offering access to high-quality, peer-reviewed publications. IEEE Xplore serves as a repository for IEEE's ebooks, standards, conference proceedings, technical reports,

⁵ See <https://ieeexplore.ieee.org/Xplore/home.jsp>.

and journal articles, with content dating back to 1988 and selectively covering materials from 1913 to 1987. The database currently includes over 5.5 million full-text documents and is recognized for its rigorous curation and accurate bibliographic records. IEEE Xplore ensures precise reporting of publication dates and event locations in alignment with industry standards for major scholarly and technical journals. Approximately 20,000 new documents are added each month. While abstracts are freely available, full-text access requires a subscription or institutional login.

19. I am familiar with the Internet Archive, a digital library formally certified by the State of California as a public library. Among other services that the Internet Archive makes available to the general public, is the Wayback Machine, an online archive. The Internet Archive's Wayback Machine service archives webpages as of a certain capture date to track changes in the web over time. The Internet Archive has been in operation as a nonprofit library since 1996 and has hosted the Wayback Machine service since its inception in 2001. I consider the Internet Archive's records keeping to be as rigorous and detailed as other formal library records keeping practices such as MARC records and OCLC records.

C. Conclusions Regarding Document 1

20. The following are my conclusions with respect to the authenticity and public availability of Document 1— Olson, E. B. (2008). *Robust and efficient robotic mapping*. Massachusetts Institute of Technology.

21. Authentication of Document 1:

A. Attachment B is a PDF of the MARC record for Document 1 from OCLC, showing a creation date of January 9, 2009.

B. Attachment C is a true and correct copy of Document 1, downloaded from the D-Space Repository at the Massachusetts Institute of Technology, showing a publication date of 2008.

C. Attachment D is a screenshot of the PDF properties of Document 1, showing a creation date of February 8, 2009.

D. Attachment E is a PDF of the metadata record for Document 1 in DSpace, showing an accession and availability date of March 19, 2009.

E. Attachment F is a PDF of a Google Scholar search for articles citing Document 1 from 2008 to 2015.

F. Attachment G is an article published in 2009 entitled “On Measuring the Accuracy of SLAM Algorithms,” by R. Kümmerle and colleagues that cites Document 1.

G. Attachment H is a PhD thesis submitted to the University of Oxford in 2009 entitled “Probabilistic Localization and Mapping in Appearance Space,” by Mark Cummins that cites Document 1.

1. Public Availability of Document 1

22. Based on the fact that Document 1 is found in the OCLC database with a creation date of January 9, 2009, and in the MIT D-Space repository with PDF creation date of February 9, 2009, and that the D-Space metadata record shows an availability date of March 29, 2009, Document 1 was available in January 2009, and no later than March 2009, and could have been found by a skilled researcher.

2. Locating Document 1

23. In my professional experience as a librarian, academic libraries possessed some form of digital catalog by 2005. A skilled researcher could have accessed the MIT digital catalog or WorldCat.org, and found Document 1 by searching for keywords related to the relevant field.

24. For example, Document 1 could have been located by searching the above libraries for any words in some portion of the DSpace metadata. This would include, for example, “robotic mapping,” and “mobile robots.” Thus, it is my conclusion that someone searching for the title or subject matter of Document 1 would have been able to locate Document 1 between January and March 2009.

25. Further, Google Scholar indicates that Document 1 was cited 126 times from 2008 through 2015, including by authors without any apparent institutional connection or private access to Document 1 (Attachment F). For example, Document 1 was cited in, “On Measuring the Accuracy of SLAM Algorithms,” an article published in 2009 by R. Kümmerle and colleagues, (Attachment G), each of whom was affiliated with University of Freiburg (Germany) at the time of publication. As another example, Document 1 was cited in, “Probabilistic Localization and Mapping in Appearance Space,” a PhD thesis submitted to the University of Oxford by Mark Cummins in 2009 (Attachment H). These citations confirm that a skilled researcher in the relevant field exercising reasonable diligence would have been able to locate Document 1 by 2009.

D. Conclusions Regarding Document 2

26. The following are my conclusions with respect to the authenticity and public availability of Document 2— Springer Handbook of Robotics” (2d ed.) (ISBN 3319325507) chapter 46: “Simultaneous Localization and Mapping.” Cyrill Stachniss, John J. Leonard & Sebastian Thrun.

27. Authentication of Document 2:

A. Attachment I is a true and correct copy of Document 2,
downloaded from <https://link-springer->

B. Attachment J is a PDF of the results screen from the SpringerLink database, showing an online availability of January 1, 2016.

C. Attachment K is a screenshot of the PDF properties of Document 2, showing a creation date of July 2016.

D. Attachment L is a PDF of the MARC record for *Springer Handbook of Robotics*, second edition, 2016, showing a creation date of March 31, 2016.

1. Public Availability of Document 2

28. Based on the fact that Document 2 is found in the SpringerLink database with an availability date of January 1, 2016, with PDF properties showing a creation date of July 2016, and that a MARC record for the *Springer Handbook of Robotics*, second edition, 2016, is found in OCLC, showing a creation date of March 31, 2016, Document 2 was available in January 2016, and could have been found by a skilled researcher.

2. Locating Document 2

29. In my professional experience as a librarian, academic libraries possessed some form of digital catalog by 2005. A skilled researcher could have accessed the digital catalogs of libraries such as those at Texas A&M University, the

University of Nebraska at Omaha, or Harvard University, which are some of the libraries who own the book that contains Document 2.

30. For example, Document 2 could have been located by searching the above libraries for any words in the book or chapter title, author names, subject headings (from Field 650 of the MARC records), or the abstract, including “Robotics,” “Artificial Intelligence,” or “Automation.” Thus, it is my opinion that someone searching for the title or subject matter of Document 2 would have been able to locate Document 2 by January 2016.

E. Conclusions Regarding Document 3

31. The following are my conclusions with respect to the authenticity and public availability of Document 3— Springer Handbook of Robotics” (2d ed.) (ISBN 3319325507) chapter 65: “Domestic Robots,” by Erwin Prassler, Mario E. Munich, Paolo Prijanian, and Kazuhiro Kosuge.

32. Authentication of Document 3:

A. Attachment M is a true and correct copy of Document 3, downloaded from https://link-springer-com.libproxy.unl.edu/chapter/10.1007/978-3-319-32552-1_65

B. Attachment N is a PDF of the results screen from the SpringerLink database, showing an online availability of January 1, 2016.

C. Attachment O is a screenshot of the PDF properties of Document 3, showing a creation date of July 21, 2016.

D. Attachment P is a PDF of the MARC record for *Springer Handbook of Robotics*, second edition, 2016, showing a creation date of March 31, 2016.

1. Public Availability of Document 3

33. Based on the fact that Document 3 is found in the SpringerLink database with an availability date of January 1, 2016 with PDF properties showing a creation date of July 21, 2016 and that a MARC record for the *Springer Handbook of Robotics*, second edition, 2016, is found in OCLC, showing a creation date of March 31, 2016, Document 3 was available in January 2016 and could have been found by a skilled researcher.

2. Locating Document 3

34. In my professional experience as a librarian, academic libraries possessed some form of digital catalog by 2005. A skilled researcher could have accessed the digital catalogs of libraries such as those at Texas A&M University, the University of Nebraska at Omaha, or Harvard University, which are some of the libraries who own the book that contains Document 3.

35. For example, Document 3 could have been located by searching the above libraries for any words in the book or chapter title, author names, subject

headings (from Field 650 of the MARC records), or the abstract, including “Robotics,” “Artificial Intelligence,” or “Automation.” Thus, it is my opinion that someone searching for the title or subject matter of Document 3 would have been able to locate Document 3 by January 2016.

I hereby declare that all statements made herein of my own knowledge are true and that all statements made on information and belief are believed to be true; and further that these statements are made with the knowledge that willful false statements and the like are punishable by fine, imprisonment, or both, under Section 1001 of Title 18 of the U.S. Code. I state under penalty of perjury under the laws of the United States of America that the foregoing is true and correct.

Executed this 5th day of November, 2025, in Minneapolis, Minnesota.

A handwritten signature in cursive script, appearing to read "Mary K. Bolin", written in black ink on a light background.

Mary K. Bolin

Attachment A

Mary K. Bolin, PhD
2433 U Street
Lincoln, Nebraska 68503
mary@marykbolin.com

Mary K. Bolin, PhD

Professor Emeritus
University Libraries
University of Nebraska—Lincoln
mbolin2@unl.edu

Lecturer,
School of Information
San José State University
mary.bolin@sjsu.edu

EDUCATION

PhD, Educational Studies (Educational Leadership and Higher Education), 2007, University of Nebraska

Dissertation: *A Typology and Discourse Analysis of the Status and Appointments of Librarians at Land Grant Universities* (available at digitalcommons.unl.edu)

MA, English, 1999, University of Idaho

Thesis: *Grace: A Contrastive Analysis of a Biblical Semantic Field* (available at digitalcommons.unl.edu)

MSLS, 1981, University of Kentucky

BA, Linguistics, 1976, University of Nebraska—Lincoln

WORK EXPERIENCE

Center for Digital Research in the Humanities. Fellow, 2019- Emeritus, 2020-

University of Nebraska--Lincoln Libraries: Professor and Catalog and Metadata Librarian, Digital Initiatives and Special Collections, 2015-2020. Emeritus, 2020-

University of Nebraska—Lincoln Libraries: Professor and Chair, Discovery and Resource Management, 2004-2015; Coordinator of Collection Development, 2012-2015

University of Idaho Library: Head, Technical Services Department, 1993-2003. Promoted to Professor in 1999

University of Idaho Library: Head, Cataloging Department, 1986-1993. Promoted to Associate Professor and tenured, 1991

University of Georgia Libraries: Head, Records Maintenance Section, Cataloging Department, 1981-1986

Teaching:

San José State University School of Information:

LIBR 281, Metadata. 2009-

LIBR 248, Introduction to Cataloging and Classification. 2013-

LIBR 202, Information Retrieval. 2008-2013

Associate Coordinator, Gateway PhD program, San Jose State University, 2021-2022

American Library Association:

Cataloging for Non-Catalogers, Online Continuing Education Course, 2017-

Intermediate Cataloging for Non-Catalogers, Online Continuing Education Course, 2018-

Lyrasis:

Webinar on BibFrame, taught twice a year, 2020-

Quick and Clean Cataloging, Parts 1, 2, and 3. 2022-

Independent Study in Idaho:

Library Science C415, Technical Services for Small Libraries, 2004-2013

Library Science C416, Library Services to Rural Libraries, 2004-2013

Library Science C422, Use of the School Library, 1999-2013

Library Science C425/525, Organization and Management of Small Libraries,
1988-2013

Library Science C427, Library or Media Center Practicum, 1995-2013

Gateway PhD Program (San Jose State University and Queensland University of Technology).
SJSU supervisor for two PhD students, 2015-

Gateway PhD Program (San Jose State University and Manchester Metropolitan University).
Associate Coordinator, 2021-2022

Consulting

Expert witness for multiple attorneys employed by Wolf, Greenfield & Sacks, P.C.,
Hughes Hubbard & Reed LLP, and other firms. 2017-

Editing

Language and academic editing for [EditSprings](#), a service for international scholars, 2020-

SCHOLARSHIP

Refereed Publications:

(Publications available from digitalcommons.unl.edu)

Pytlik Zillig, Brian, and Mary K. Bolin. (Forthcoming 2024). “Visualization of Semantic Primes in Texts from Early Modern English Drama.” *Journal of the Japanese Association for the Digital Humanities*.

Bolin, Mary K. (2022). “Academic Libraries and Writing Centers: Collaborations at US Public Research Universities.” *School of Information Student Research Journal*, 11(2). Retrieved from <https://scholarworks.sjsu.edu/ischoolsrj/vol11/iss2/2>

Bolin, Mary K. (2022). *Refocusing Academic Libraries through Learning and Discourse: The Idea of a Library*. Oxford, UK: Chandos/Elsevier.

Bolin, Mary K. (2022). “Metadata, Catalog Systems, Linked Data, and the Evolving ILS.” In: Sandra Hirsh (Ed.), *Information Services Today: An Introduction*. 3rd ed. New York: Rowman and Littlefield.

Wintermute, Harriet E., and Bolin, Mary K. (2018). "Metadata Generators for Backlog Reduction: Metadata Maker Streamlines Cataloging and Facilitates Transition. *Technical Services Quarterly* 35(2) doi: <https://doi.org/10.1080/07317131.2018.1422879>

Bolin, Mary K. (2018). "Metadata, Catalog Systems, Linked Data, and the Evolving ILS." In: Sandra Hirsh (Ed.), *Information Services Today: An Introduction*. 2nd ed. New York: Rowman and Littlefield.

Bolin Mary K. (2017). *The 21-century Academic Library: Global Patterns of Organization and Discourse*. Oxford: Chandos Publishing. (Chandos Information Professional Series)

Bolin, Mary K. (2017). Linguistics and LIS: A Research Agenda." *SLIS Student Research Journal*, 7:1.

Retrieved from <http://scholarworks.sjsu.edu/slissrj/vol7/iss1/2>

Bolin, Mary K. (2014). "Open Access Web Resources for Library Continuing Education and Training." In S. Hines (Ed.), *Revolutionizing the Development of Library and Information Professionals: Planning for the Future* (pp. 107- 124). Hershey, PA: . doi:10.4018/978-1-4666-4675-9.ch008.

Bolin, Mary K. (2014). "The Language of Academic Librarianship: The Discourse of Promotion and Tenure. " *Advances in Library Administration and Organization* 32.

Bolin, Mary K. "Librarian Status at US Research Universities: Extending the Typology." *Journal of Academic Librarianship* 34:4 (September 2008)

Bolin, Mary K. "A Typology of Librarian Status at Land Grant Universities." *Journal of Academic Librarianship*, 34: 3 (May 2008)220:230.

Bolin, Mary K. "The Library and the Computer Center: Organizational Patterns at Land Grant Universities." *Journal of Academic Librarianship*, 31:1 (2005)

Eckwright, Gail Z. and Mary K. Bolin. "The Hybrid Librarian: The Affinity of Collection Management with Technical Services and the Organizational Benefits of an Individualized Assignment." *Journal of Academic Librarianship*, 27:6 (2001)452:457.

Bolin, Mary K. "Catalog Design, Catalog Maintenance, Catalog Governance," *Library Collections, Acquisitions, and Technical Services*. 24: (2000)53-63.

Bolin, Mary K. "The Collegial Environment and the Functional Organization," *Journal of Library Administration*. 29:2(1999):49-61.

Bolin, Mary K. "Cataloging Environmental Impact Statements and Related Documents," *Resource Sharing and Library Networks*. 14:1(1999): 21-40.

Bolin, Mary K. "Waiting for the Electrician or Someone like Him: Client-Centered Technical Services" Proceedings of the Association of College and Research Libraries National Conference, Nashville, April 1997.

< http://www.ala.org/Content/ContentGroups/ACRL1/Nashville_1997_Papers/Bolin.htm >

Grants:

Electronic Theses and Dissertations. Seed Grant from University of Idaho Office of Research, 2002. Co PI with Gail Z. Eckwright. \$6500.

Interoperability of Metadata for Thematic Research Collections. Grant to the Center for Digital Research in the Humanities from the Institute of Museum and Library Services 2005-2008. Project Team Member.

http://unlcms.unl.edu/cas/center-for-digital-research-in-the-humanities/projects/pages/iom_files/IOM_Final_Narrative_Report_July_2008.pdf

Other Publications:

Library Philosophy & Practice. Peer-reviewed electronic library science journal. Mary K. Bolin and Gail Z. Eckwright, founders and editors. <http://digitalcommons.unl.edu/libphilprac>

Library Science C415, Technical Services for Small Libraries, through Independent Study in Idaho. 2004.

Library Science C416, Library Services to Rural Libraries, through Independent Study in Idaho. 2004.

Library Science C422, Use of the School Library, through Independent Study in Idaho. Revised 1999.

Library Science C425/525, Organization and Management of Small Libraries through Independent Study in Idaho. Revised 2005.

Library Science C427, Library or Media Center Practicum, Independent Study in Idaho. Revised 1999.

Presentations:

Bolin, Mary K. "BibFrame and the MARC Cataloging Environment: an Introduction." Lyrisis Webinar, June 16, 2020, August 17, 2021.. Recording available <http://lyrisis.adobeconnect.com/paifo9e0dm30/>

Pytlik Zillig, Brian, and Mary K. Bolin. "A Little More than Kin, a Little Less than Kind": the Semantic Field Grace in Early Modern English" Poster Session, Text Encoding Initiative Meeting, Graz, Austria, September 18, 2019

Pytlik Zillig, Brian, and Mary K. Bolin. "Shakespeare and the Enumeration of Semantic Primes." Research Paper Presentation, Text Encoding Initiative Meeting, Tokyo, Japan, September 13, 2018

"Library Philosophy and Practice: Creating and Maintaining an Open-Access E-Journal," Presentation to UNCL meeting (libraries at campuses of the University of Nebraska), Lincoln, Nebraska, March 20, 2008.

Presentation to "Thinking outside the Borders" participants on the open-access electronic journal Library Philosophy and Practice and its role in the international library community. University of Nebraska—Lincoln Libraries, October 24, 2007

Panel discussion on open access electronic journals. GWLA directors' meeting. Linda K. Hall Library, Kansas City, Missouri. April 2007.

"Waiting for the Electrician or Someone Like Him: Client-Centered Technical Services." Association of College and Research Libraries National Conference, Nashville, April 5, 1997.

" 'Some Problems of an Editor': the Idaho Librarian from Manuscript to Published Issue," Joint Annual Conference of the Idaho Library Association and the Washington Library Association, Spokane, Washington, April 6, 1995.

"Automating Idaho's Libraries: a Historical Perspective." Idaho Library Association Annual Conference, Idaho Falls, October 4, 1990.

"Small Library Management." Idaho Library Association Spring Conference, Lewiston, Idaho, May 13, 1989.

"No More 'Good Workers': Excellence and Academic Library Classified Staff." Association of College and Research Libraries 5th Annual Conference, Cincinnati, April 7, 1989.

"Communication to and from Cataloging Departments." Heads of Cataloging Discussion Group, American Library Association Midwinter Meeting, San Antonio, January 1988.

Discussion Leader at the LAMA (Library Administration and Management) Management Exchange. American Library Association Annual Meeting, San Francisco, June 1987.

"Staff Turnover." Heads of Cataloging Discussion Group, American Library Association Midwinter Meeting, Chicago, January 1987.

SELECTED PROFESSIONAL SERVICE

Reviewer. *Resources for College Libraries*. Linguistics bibliography, Spring 2020.

Peer reviewer for ALISE National Conference presentations, Spring 2020.

Peer reviewer for *Journal of Academic Librarianship*, 2012-

Peer reviewer for refereed journal, *Western Nigerian Journal of Medical Sciences* (<http://www.wnjms.org/>), a publication of Ladoke Akintola University of Technology (LAUTECH) Teaching Hospital, 2008-

PhD Reviewer for students from Aligarh Muslim University, India, 2010-

Outside reviewer for numerous library faculty promotion and tenure cases in US universities.

University of Nebraska--Lincoln

Chair, University Libraries Faculty, 2018

Faculty Senate Academic Rights and Responsibilities Committee, 2016- Chair, 2018-19

Faculty Senate – 2004-2008; 2008-2011

Faculty Senate Executive Committee 2004-2008

Academic Planning Committee (APC), 2005-2008; Vice Chair, 2006-2007; Chair, 2007-2008

Aesthetic Review Committee 2006-2007

APC monitor for academic program reviews: Department of Political Science, 2005;
Department of Educational Psychology 2006
Pacific Northwest Library Association:

Editor, *PNLA Quarterly*, December 2002-December 2013

Idaho Library Association:

Editor, *The Idaho Librarian*, September 1990-January 1996

Attachment B

Bibliographic Record - OCLC Number: 297537764

LDR	Record Status	'c' - Corrected or revised
	Type of Record	't' - Manuscript language material
	Bibliographic Level	'm' - Monograph/Item
	Type of Control	' ' - No specified type of control
	Encoding Level	'7' - Minimal level
	Descriptive Cataloging Form	'a' - AACR2
001	297537764	
005	20230922064919.1	
008	Date Entered on File	'090109'
	Type of Date/Publication Status	's' - Single date
	Date1/Beginning Date of Publication	'2008'
	Date2/Ending Date of Publication	' '
	Place of Publication, Production, or Execution Code	'xx' - No place, unknown, or undetermined
	Illustrations	'ab'
	Target Audience	' ' - Unknown or unspecified
	Form of Item	' ' - None of the following
	Nature of Contents Code	'bm'
	Government Publication	' ' - Not a government publication
	Conference Publication	'0' - Not a conference publication
	Festschrift	'0' - Not a festschrift
	Index	'0' - No index
	Literary Form	'0' - Not fiction
	Biography	' ' - No biographical material
	Language Code	'eng' - English
	Modified Record	' ' - Record is not modified
	Cataloging Source Code	'd' - Other
040	MYG \$b eng \$c MYG \$d OCL \$d OCLCQ \$d OCLCO \$d OCLCF \$d OCLCQ \$d OCLCO \$d OCLCQ \$d OCLCO	
090	\$b	
049	LDLL	
100	1 Olson, Edwin B. \$q (Edwin Brock), \$d 1977-	
245	1 0 Robust and efficient robotic mapping / \$c by Edwin B. Olson.	
260	\$c ©2008.	
300	129 pages : \$b illustrations (chiefly color), maps (chiefly color) ; \$c 29 cm	
336	text \$b txt \$2 rdacontent	
337	unmediated \$b n \$2 rdamedia	
338	volume \$b nc \$2 rdacarrier	
520	3 Mobile robots are dependent upon a model of the environment for many of their basic functions. Local ly accurate maps are critical to collision avoidance, while large-scale maps (accurate both metrically a nd topologically) are necessary for efficient route planning. Solutions to these problems have immediat e and important applications to autonomous vehicles, precision surveying, and domestic robots. Buildi ng accurate maps can be cast as an optimization problem: find the map that is most probable given the set of observations of the environment. However, the problem rapidly becomes difficult when dealing with large maps or large numbers of observations. Sensor noise and non-linearities make the problem e	

ven more difficult especially when using inexpensive (and therefore preferable) sensors. This thesis describes an optimization algorithm that can rapidly estimate the maximum likelihood map given a set of observations. The algorithm, which iteratively reduces map error by considering a single observation at a time, scales well to large environments with many observations. The approach is particularly robust to noise and non-linearities, quickly escaping local minima that trap current methods. Both batch and online versions of the algorithm are described. In order to build a map, however, a robot must first be able to recognize places that it has previously seen. Limitations in sensor processing algorithms, coupled with environmental ambiguity, make this difficult. Incorrect place recognitions can rapidly lead to divergence of the map. This thesis describes a place recognition algorithm that can robustly handle ambiguous data. We evaluate these algorithms on a number of challenging datasets and provide quantitative comparisons to other state-of-the-art methods, illustrating the advantages of our methods.

- 502 **\$b** Ph. D. **\$c** Massachusetts Institute of Technology, Dept. of Electrical Engineering and Computer Science **\$d** 2008
- 504 Includes bibliographical references (pages 123-129).
- 655 7 dissertations. **\$2** aat
- 655 7 Academic theses **\$2** fast
- 655 7 Academic theses. **\$2** lcgft
- 655 7 Thèses et écrits académiques. **\$2** rvmgf

Attachment C

Robust and Efficient Robotic Mapping

by

Edwin B. Olson

Submitted to the Department of
Electrical Engineering and Computer Science
in partial fulfillment of the requirements for the degree of
Doctor of Philosophy in Computer Science and Engineering
at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

June 2008

© Edwin B. Olson, MMVIII. All rights reserved.

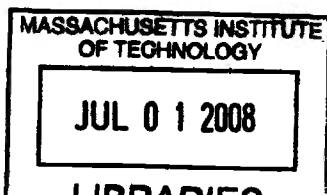
The author hereby grants to MIT permission to reproduce and
distribute publicly paper and electronic copies of this thesis document
in whole or in part.

Author
Department of Electrical Engineering and Computer Science
May 16, 2008

Certified by
Seth Teller
Professor of Computer Science and Engineering
Thesis Supervisor

Certified by
John Leonard
Professor of Mechanical and Ocean Engineering
Thesis Supervisor

Accepted by
Terry P. Orlando
Chairman, Department Committee on Graduate Students



ARCHIVES

Robust and Efficient Robotic Mapping

by

Edwin B. Olson

Submitted to the Department of Electrical Engineering and Computer Science
on May 16, 2008, in partial fulfillment of the
requirements for the degree of
Doctor of Philosophy in Computer Science and Engineering

Abstract

Mobile robots are dependent upon a model of the environment for many of their basic functions. Locally accurate maps are critical to collision avoidance, while large-scale maps (accurate both metrically and topologically) are necessary for efficient route planning. Solutions to these problems have immediate and important applications to autonomous vehicles, precision surveying, and domestic robots.

Building accurate maps can be cast as an optimization problem: find the map that is most probable given the set of observations of the environment. However, the problem rapidly becomes difficult when dealing with large maps or large numbers of observations. Sensor noise and non-linearities make the problem even more difficult—especially when using inexpensive (and therefore preferable) sensors.

This thesis describes an optimization algorithm that can rapidly estimate the maximum likelihood map given a set of observations. The algorithm, which iteratively reduces map error by considering a single observation at a time, scales well to large environments with many observations. The approach is particularly robust to noise and non-linearities, quickly escaping local minima that trap current methods. Both batch and online versions of the algorithm are described.

In order to build a map, however, a robot must first be able to recognize places that it has previously seen. Limitations in sensor processing algorithms, coupled with environmental ambiguity, make this difficult. Incorrect place recognitions can rapidly lead to divergence of the map. This thesis describes a place recognition algorithm that can robustly handle ambiguous data.

We evaluate these algorithms on a number of challenging datasets and provide quantitative comparisons to other state-of-the-art methods, illustrating the advantages of our methods.

Thesis Supervisor: Seth Teller

Title: Professor of Computer Science and Engineering

Thesis Supervisor: John Leonard

Title: Professor of Mechanical and Ocean Engineering

Contents

1	Introduction	9
1.1	Motivation	9
1.2	Probabilistic Problem Formulation	12
1.3	Loop Closing	13
1.4	Pose/Feature Graphs	15
1.5	Map Optimization	16
1.6	Contributions	16
2	Preliminaries	19
2.1	Pose/Feature Graphs	19
2.2	Graphical Models	19
2.3	Geometrical Conventions	20
2.4	Constraints	22
2.4.1	Common Constraint Forms	23
2.4.2	Linearization	23
2.4.3	Interpretation of the Chi-Squared Error	25
2.4.4	State Squared Error	25
3	Loop Closing	29
3.1	Introduction	29
3.2	Method Overview	32
3.3	Hypothesis Generation	32
3.3.1	Determining when poses overlap	32
3.3.2	Searching for a match	34
3.4	Hypothesis Sets	37
3.4.1	Pairwise Consistency	37
3.4.2	Finding the optimal subset	38
3.4.3	The Eigenvectors of the Consistency Matrix	40
3.4.4	Discretization of the Indicator Vector	40
3.4.5	Improving Hypothesis Set Construction	41
3.4.6	Analysis of Robustness to Outliers	42
3.5	Results	43
3.5.1	“Area C” Dataset	44
3.5.2	CSAIL Dataset	46
3.5.3	Killian Court Dataset	48

3.5.4	Stanford Gates Building	49
3.5.5	Intel Research Center	49
3.6	Summary	49
4	Batch Optimization of Pose Graphs	55
4.1	Prior Work	56
4.1.1	Recursive Linear Solutions	56
4.1.2	Graphical Models	58
4.1.3	Particle Filtering	58
4.1.4	Nonlinear Optimization	58
4.1.5	Hybrids	59
4.1.6	State Space	59
4.2	Overview and Intuition	60
4.3	Method	61
4.3.1	Incremental State Representation	61
4.3.2	Iterative optimization	63
4.3.3	Rigid-Body Constraints	66
4.3.4	The Error-Distributing Tree	68
4.3.5	Initial Learning Rate	71
4.3.6	Algorithm Summary	71
4.4	Results	72
4.4.1	CSW Dataset	73
4.4.2	High Noise Dataset	78
4.4.3	P5K20K Dataset	78
4.4.4	Intel Research Center Dataset	83
4.4.5	Stanford Gates Dataset	84
4.4.6	Dog Leg Configuration	85
4.4.7	Robustness	86
4.5	Summary	88
5	Incremental Optimization of Pose Graphs	89
5.1	Method	90
5.1.1	Adding a new constraint	91
5.1.2	Processing existing constraints	92
5.1.3	Algorithm Summary	93
5.2	Partial Iterations	93
5.3	Learning Rate Data Structure	95
5.4	Analysis	97
5.5	Results	98
5.6	Summary	103
6	Conclusion	105
6.1	Contributions	105
6.2	Future Work	106

A	Sensor Processing	109
A.1	Overview	109
A.2	Vision Processing	109
A.2.1	Data Collection Platform	109
A.2.2	Feature Detection	110
A.2.3	Feature Tracking and Landmark Initialization	110
A.2.4	Landmark matching	112
A.3	Laser Scan Processing	113
A.3.1	Challenges	114
A.3.2	Scan Matching	116
A.3.3	Loop Closing Hypothesis Generation	120
A.4	Summary	122

CONTENTS

CHAPTER 1

Introduction

1.1 Motivation

Intelligent robots must answer a fundamental question: given a number of noisy sensor observations, what does the world actually look like? The quality of the world model is dependent on the quality of the sensors, of course, but as the environments that robots operate in get larger and more complicated, the limiting factor becomes *algorithmic*: the amount of computational effort required to process sensor observations grows quickly with the number of observations.

While robots like the Roomba vacuum can get by with limited world models (i.e., detecting obstacles by bouncing off of them), an autonomous vehicle travelling at highway speeds needs a more sophisticated obstacle avoidance strategy. High-quality maps are critical.

Many basic robot capabilities, such as collision avoidance and path planning, depend upon a map of the environment. Consequently, map building is a critical component of almost every autonomous robot, particularly if they must operate in unfamiliar environments. Many applications fall into this category, including cars in urban driving scenarios, military convoy vehicles, indoor delivery robots, and assistive robots for the elderly and infirm. Maps are also useful in non-autonomous applications: a human can become disoriented when teleoperating a search-and-rescue robot amidst rubble; a real-time map can make the operators more productive. In these cases, robots use maps in order to complete their missions. The accuracy and quality of the map directly affects their ability to effectively carry out their missions. If an obstacle is poorly localized, a robot may collide with it. If the topological structure of an environment is not properly modeled, a robot may take a needlessly circuitous route to a destination, or fail to find a route entirely.

The map itself is the goal of an increasing number of systems. These include capturing data for Geographic Information Systems (GIS) (both indoors and outdoors), along with other survey applications (such as measuring the sag of a bridge by driving a robot beneath it). The value of these systems is directly related to the accuracy of the map and the cost of constructing it.

Map building is hard. The real world is a complex, dynamic, and cluttered place. A sensor's view of this world is noisy and often ambiguous (different things often look



Figure 1-1: Robot Applications. Map building arises in many contexts, including autonomous cars [37], indoor surveying [68], underwater robots for ship-hull inspection [75], and human-centric assistive robotics [57].

about the same) and there is an economic incentive to use less expensive sensors; the limitations of these sensors often magnify the challenges. In short, the aim of mapping research is to accurately model large, complex, and cluttered environments using cheap sensors that produce poor data.

Mapping has been extensively studied by many researchers, serving as a central research topic for roboticists for decades. Early approaches were largely derived from the Extended Kalman Filter (EKF) [65] and it continues to be used in many applications. The EKF, however, has computational and memory costs that grow quadratically with the map size. The Extended Information Filter (EIF) [58] improves these costs, as do the many variants of Sparse Extended Information Filters (SEIF) [69, 70, 71, 13, 14, 74, 76, 76]. New variants, offering not only computational cost improvements but also interesting cost-versus-quality trade-offs, continue to be developed.

The mapping problem has also been approached by means of Rao-Blackwellized particle filters [44, 66, 22] and non-linear optimization [41, 12, 34, 72, 20, 16, 10, 24, 48, 50]. These methods offer additional cost/quality trade-offs, and have incorporated ideas and techniques from the machine learning and numerical computing community.

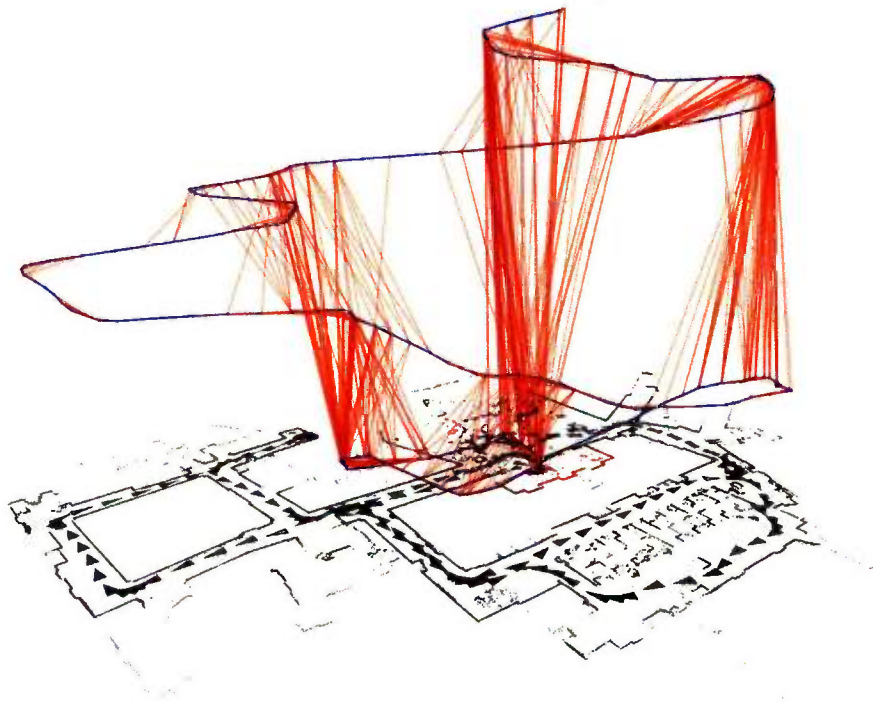


Figure 1-2: An automatically built map of an indoor office environment. Above the map, the trajectory (blue) is plotted with height corresponding to time. The large number of detected loop closures (shown in red) allow a high-quality posterior map to be computed.

However, the limited penetration of robots into our daily lives is a testament to the shortcomings of the state-of-the-art; at best, we see very expensive robots operating in rather limited environments. The challenge is to extend the state of the art by making these systems *faster* (so they can handle larger environments and greater numbers of sensor observations) and *more robust* to noise and ambiguity. Computational efficiency and robustness are the central themes of this thesis.

This thesis describes several new algorithms that extend the reach of mapping algorithms. We begin with a review of basic concepts in Chapter 2. The problem of identifying when a robot has revisited a particular place, loop closing, is a critical component of map building. Recognizing a place can be difficult because today's sensor processing algorithms can not always discriminate between similar (but distinct) places. We describe an algorithm that can automatically reject incorrect place recognitions that could otherwise result in poor maps in Chapter 3.

Given a set of correct place recognitions, the map building problem can be formally represented as a non-linear optimization: compute the map that is most consistent with all of the disambiguated sensor data. We describe a new algorithm for performing this optimization in Chapter 4, which is characterized by high speed and an unrivaled ability to escape local minima. As a result, our method can compute correct maps in cases where existing algorithms fail. In Chapter 5, we describe a variant of this algorithm that can be used in online settings— i.e., where the optimization problem is

constantly changing as new sensor data continuously arrives. Together, the methods of this thesis allow maps to be constructed even when sensor data is highly ambiguous, with computational costs that scale well with increasing environment size.

The remainder of this chapter provides a brief introduction to robot mapping and a high-level overview of this thesis's contributions.

1.2 Probabilistic Problem Formulation

In general terms, the map building problem can be described as finding the posterior probability distribution function of a map x given a set of sensor data z and a set of commanded robot motions u . We can divide the map into two parts: the trajectory of the robot s sampled at discrete times, and the location of other features f in the environment.

Unfortunately, most sensors do not unambiguously identify the features that they sense: they detect the presence of *some* feature, not the presence of a *particular* feature. The process of assigning feature identities to sensor observations is known as data association. Let d_i represent the identity of the i^{th} sensor observation z_i . The posterior probability distribution that we wish to discover is:

$$p(x \mid u, z, d) = p(s, f \mid u, z, d) \quad (1.1)$$

In the common case, sensors are mounted on the robot. Sensors observe the environment relative to the robot's current position, and the noise contaminating these robot-relative observations is independent. Projecting these observations into a global coordinate frame requires only the knowledge of the robot's position. In other words, sensor observations (in the global coordinate frame) are conditionally independent given the robot trajectory. This is the key insight behind FastSLAM[44], as it allows the posterior distribution (Eqn. 1.1) to be factored into terms corresponding to the robot trajectory and the individual position of each feature in the environment:

$$p(s, f \mid u, z, d) = p(s \mid u, z, d) \prod_i p(f_i \mid s, z, d) \quad (1.2)$$

The advantage of this factorization is that the conditional posterior distribution for a single feature $p(f_i \mid s, z, d)$ is easy to compute (e.g., with a Kalman filter). The posterior distribution for the trajectory is much harder: FastSLAM samples over this trajectory with a particle filter, whereas the approach in this thesis is to use non-linear optimization methods to compute it.

In short, once the trajectory of the map has been determined, a map can be easily computed. The critical role of the robot trajectory in map building has led to trajectory-centric approaches like the one presented in this thesis.

In principle, the data associations d are unknown and are correlated through the motion of the robot. Ideally, we would like to know the joint distribution:

$$p(d \mid u, z) \quad (1.3)$$

In this thesis, we make the common (but sub-optimal) approximation of computing d_i at about the time we observe z_i . Additionally, we do not revisit data associations based on subsequent sensor observations. Consequently, each data association d_i is a function of z_i and earlier sensor observations and their data associations.

$$p(d \mid u, z) \approx \prod_i p(d_i \mid u_{0\dots i}, z_{0\dots i}, d_{0\dots i-1}) \quad (1.4)$$

The data-associations d assign particular observations to particular features. This is a discrete-valued assignment, in contrast to the continuously-valued variables comprising the map x . Since it has both discrete and continuous aspects, map building is a hybrid estimation problem. The discrete nature of the data associations is especially vulnerable to errors: assigning an observation to the incorrect feature can have a profound impact on the quality of the resulting map. Chapter 3 will illustrate a robust means of computing data-associations d , using an approximation similar to Eqn. 1.4.

Maintaining posterior *distributions* over the entire map is both computationally expensive and often unnecessary. The maximum likelihood map is generally the single most useful quantity for path planning, obstacle avoidance, and other practical applications. Some marginal distributions, such as the probability distribution of the robot's current position and some nearby feature ($p(s_{last}, f_i, \mid u, z, d)$), can be easily estimated without maintaining the full posterior distribution. Chapter 3 will show how to estimate marginal distributions of this form.

Consequently, while this thesis explores mapping from a probabilistic perspective, the primary goal is not to compute distributions, but instead to estimate the maximum likelihood map. Chapters 4 and 5 will then estimate the maximum-likelihood map x using non-linear optimization methods.

1.3 Loop Closing

As a robot drives around, its dead-reckoning error grows without bound. Suppose that a robot revisits a previously observed place, collecting similar sensor data each time. How can the robot determine that it is in fact in the same place it was before, versus some other place that looks about the same? How can the robot determine its position and orientation within that place? Loop closing algorithms seek to answer these questions.

The term “loop closing” arises from the common case where a robot drives in a large loop, returning to its starting point. If, after travelling around the loop, the robot can recognize that it is at its starting position, it “closes” the loop, creating a geometric relationship between the first and last position of the robot. This geometric relationship has two major effects:

- It establishes a *topological* relationship between temporally distant parts of the robot's trajectory. The robot can use this new topological link as a “short-cut” when planning routes.

- It provides a *metrical* relationship between the beginning and end of the robot's trajectory that can be used to refine the estimate of the robot's trajectory, improving upon the estimate provided by dead-reckoning.

In general, the larger the loop, the more difficult it is to recognize a closure. This is due to accumulating dead-reckoning error as the robot traverses the loop: when the robot returns to its starting point, it must determine that it is back at its starting point— not merely somewhere that *looks* like its starting place. Large loop closures may be more difficult, but they also are much more valuable in terms of refining the trajectory of the robot. Consequently, it is critical that a loop closing algorithm be able to close very large loops.

Loop closing is related to place recognition, which has applications beyond path planning and map building. For example, the same algorithms could be used by a robot to find a recharging station and properly align itself to it.

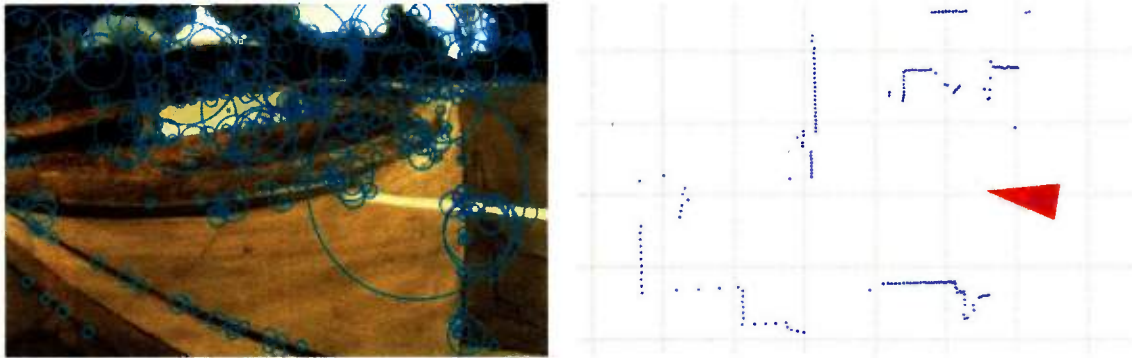


Figure 1-3: Sample sensor data. Different sensor modalities present different challenges when trying to “close the loop”. Left: Camera-derived visual features can be richly descriptive, but are also cluttered and not consistently repeatable under changing lighting and viewpoint. Right: Laser-scan data from an indoor environment is composed principally of walls at 90 degree angles. While the scan is very precise, many indoor environments produce similar looking scans.

Loop closing is typically performed via explicit data association: recently observed landmarks are associated with previously mapped landmarks. Optimal data association is NP-hard, though a number of heuristics are often effective [46, 2]. Unfortunately, data association errors can lead to divergence of the map; this is combated by performing joint data association on large numbers of observations, which increases the computational costs.

An alternative approach relies on pose-to-pose matching, in which the global identities of locally-observed features are never explicitly computed. We consider two instances of this problem. The first is based on laser scan matching, in which case the match is the result of a numerical optimization that attempts to align two scans so that they overlap as much as possible. The second instance is based on visually tracked objects in the environment. Possible alignments between poses are recovered by attempting to align these locally tracked features.

In both of these cases, the computational burden of globally identifying all landmarks is eliminated. The trade-off is that these pose-to-pose alignments, while comparatively easy to compute, have a fairly high error rate. For example, failures can

be caused by matching two similar-looking (but physically distinct) features, or by getting stuck in a local minimum when attempting to align two laser scans.

This thesis describes a loop-closing verification algorithm that allows robust loop closing using these simple pose-to-pose correspondences. The algorithm does not attempt to globally register landmarks, and thus does not require their expensive-to-compute covariance matrix. The algorithm considers a set of pose-to-pose “hypotheses” and extracts the subset that is maximally self-consistent. The algorithm exploits the property that correct hypotheses must be mutually consistent (since there is only one true configuration), whereas incorrect hypotheses are often inconsistent.

In Chapter 3, we illustrate how this maximally self-consistent set can be computed from the dominant eigenvector of a carefully-constructed matrix of pair-wise consistencies. We evaluate our system using both laser and camera data, and briefly describe our method for obtaining pose-to-pose hypotheses.

1.4 Pose/Feature Graphs

Pose/Feature graphs are an intuitive representation of map optimization problems. The nodes of a pose/feature graph represent the positions whose exact location are of interest, while the edges represent information about the relative geometric position of those positions.

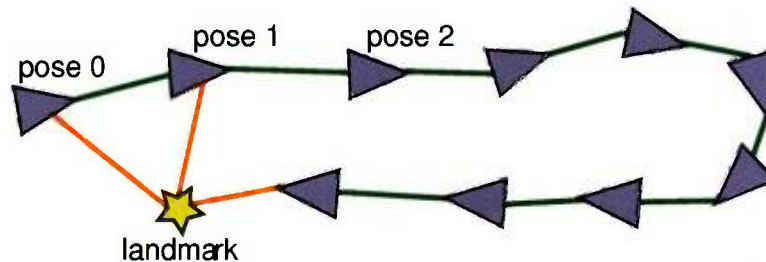


Figure 1-4: A Simple Pose/Feature graph. The robot’s trajectory is discretized into a series of poses (triangles). These poses, as well as landmarks (stars), are nodes in the pose graph. Edges in the pose graph represent constraints between the positions of nodes, such as those derived from odometry (green) and a landmark detector (orange).

For example, the position of the robot at a particular point in time can be represented as a node in the graph. These types of nodes are known as *poses*. Features (or landmarks) in the environment can also be represented as nodes in the graph: when sensors detect a landmark, the sensor observation is represented as an edge connecting the landmark to the pose from which the observation was made. Fig. 1-4 illustrates a simple pose/feature graph; see Fig. 1-2 for a real-world example.

A graph containing both poses and features is a “pose/feature graph”. A graph consisting of only poses and edges between poses is simply a “pose” graph.

1.5 Map Optimization

Loop closing adds edges to the graph, making it cyclic. Each edge relates the geometrical position of the two nodes it connects and has a cost function associated with deviations from that relationship. We now wish to find the maximum likelihood (minimum cost) position of each node. This is the map optimization step, and is what makes the map “line up” (see Fig. 1-5).



Figure 1-5: Map Building Example. Given the robot’s trajectory (left), we must first identify loop closures (middle). Finally, the maximum likelihood positions of all the nodes, given the loop closures and the original trajectory, are computed (right). The result is a map that is much better than the map resulting from dead-reckoning.

The pose graph encodes a non-linear optimization problem: the goal is to estimate the position of all the nodes (the state variables) such that the constraints have the minimum possible error. The large number of state variables, the non-linear nature of the constraint equations, and the poor initial estimate (typically derived from dead-reckoning) combine to make the optimization problem difficult. Its importance and difficulty have made optimization a heavily researched topic, but performance on large problems (those with many nodes and edges) remain computationally difficult.

Robustness to noise is an important performance metric: if an algorithm converges to the wrong answer (a local minimum), it hardly matters how quickly it managed to compute it. Sensor noise increases the likelihood of an algorithm getting stuck in a local minimum, and the desire to use less expensive sensors (e.g., wireless RF ranging to base stations, radars, cameras) often results in noisier measurements. The robustness of optimization methods has not been a focus of previous research, despite its importance.

We describe an optimization algorithm based on stochastic gradient descent that computes updates to the state vector by considering a single constraint at a time. This yields fast updates that rapidly explore the state space, allowing the algorithm to escape local minima that slow or trap existing algorithms. A batch version of this algorithm (for off-line use) is described in Chapter 4, followed by a version suitable for on-line use in Chapter 5.

1.6 Contributions

We have identified two major challenges in robotic mapping: loop closing and map optimization. This thesis describes several contributions that demonstrably advance

the state of the art for both of these challenges. These contributions include:

- A place recognition system based on pair-wise correspondences that can robustly identify loop closures, even when many hypotheses are incorrect.
- A map optimization algorithm whose iterations depend on only a single constraint at a time. The algorithm is robust to optimization hazards such as local minima, producing good maps when existing algorithms fail. The algorithm is generally competitive with, or faster than, existing methods.
- An incremental on-line map optimization algorithm, based on the batch algorithm, that is well-suited for real-time use on robots.

These algorithms can construct maps with greater speed and robustness than existing algorithms. As a result, robots can successfully operate in larger, more complex, and more ambiguous environments than was previously possible.

CHAPTER 2

Preliminaries

This thesis is about how to construct accurate maps. There are several ways of describing this problem; in this thesis, we use the *pose graph* metaphor. This chapter describes pose graphs and then moves on to more mechanical matters such as our geometrical and mathematical conventions.

2.1 Pose/Feature Graphs

The position and orientation of a robot at a particular point in time is a *pose*, graphically denoted in this thesis as a triangle (see Fig. 2-1). Over time, the continuous trajectory of the robot is discretized into a set of *poses*. A landmark in the environment is graphically represented as a star.

Information relating the position of two nodes, along with the uncertainty of that information, is known as a *constraint*¹. A constraint is represented as an edge in the pose graph. Constraints can express virtually any type of information, including a range constraint, bearing constraint, or a full rigid-body transformation. We model the uncertainty of these constraints as Gaussian, associating each with a covariance matrix.

This thesis focuses on two-dimensional pose graphs and rigid-body constraints, due to their ubiquity and applicability to many terrestrial mapping systems. We will also consider the case in which the robot is continuously moving and making new observations. This results in the addition of new nodes and edges to the graph. For simplicity, we assume that individual nodes do *not* move.

2.2 Graphical Models

Mapping problems can alternatively be described in terms of a graphical model of a Markov Random Field [30]. These graphical models are closely related to pose/feature

¹The term “constraint” is used in some contexts to mean an inviolable criterion that any solution must satisfy exactly. In the map optimization context, a constraint is uncertain: violation of a constraint incurs a cost that is a function of the magnitude of the violation. It is usually impossible to satisfy all constraints exactly— instead, we wish to minimize the total cost of all constraints.

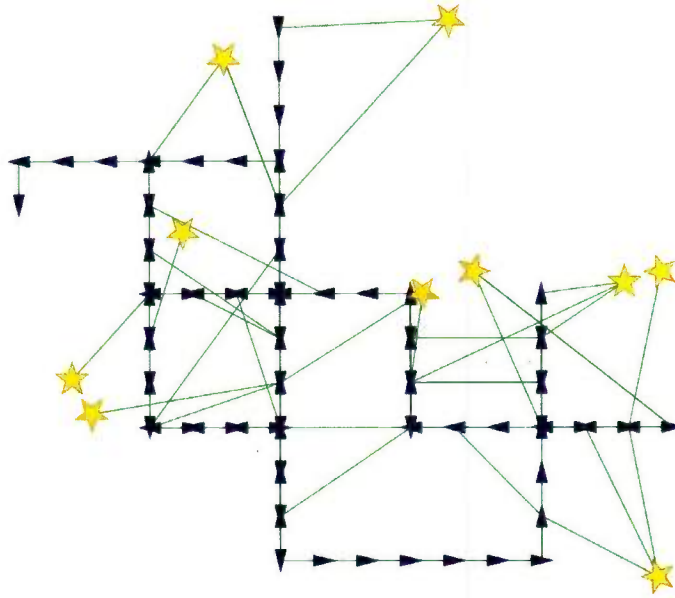


Figure 2-1: Sample Pose/Feature Graph. The robot trajectory is discretized into a series of poses (blue triangles) from which the robot observes landmarks (yellow stars). Both poses and landmarks are nodes in the graph. Graph edges represent information about the relative position and orientation of the two nodes.

graphs. The nodes of a pose/feature graph are the same as the random variables of a graphical model. A constraint between two nodes in a pose/feature graph implies a dependence between the same two nodes in a graphical model, which is also represented by an edge.

The difference between the two is a primarily an issue of connotation. A pose/feature graph describes the “experience” of a robot: its entire trajectory and its entire set of observations. It represents a complete statement of the structure of a particular mapping problem. For every such pose graph, there is an exactly equivalent graphical model.

In contrast, graphical models describe a joint probability distribution over the nodes in the graph. For example, the robot trajectory can be marginalized out. The resulting graphical model has no poses— only landmarks that are connected directly to each other. Naturally, landmarks cannot observe other landmarks, and so a marginalized graphical model does not correspond to a real set of observations. The use of a graphical model also implies a particular set of inference methods (e.g., belief propagation).

In this thesis, we use the term pose/feature graph to refer to a complete statement of the mapping problem (i.e., one in which no variables have been marginalized out).

2.3 Geometrical Conventions

A 2-D rigid body transformation is parameterized by three values, a translation in $\hat{x}$, a translation in $\hat{y}$, and a rotation θ . Subscripts will be used to identify particular sets

of parameters (e.g., x_A , y_A , and θ_A). These parameters can be used to compute a 3×3 transformation matrix, which we will identify by a capital letter. For example:

$$A = \begin{bmatrix} \cos(\theta_A) & -\sin(\theta_A) & x_A \\ \sin(\theta_A) & \cos(\theta_A) & y_A \\ 0 & 0 & 1 \end{bmatrix} \quad (2.1)$$

A point p , expressed in homogeneous coordinates [27] (e.g., $p = [x \ y \ 1]^T$) is transformed by A by simply $p' = Ap$. Transforming p first by B , then by A is computed by $p' = ABp$. When dealing with chains of rigid-body transformations, such as the path of a robot, we will write our transformations such that the rigid-body transformations project a point (whose coordinates are expressed relative to one pose) to a point whose coordinates are expressed relative to the *previous* pose. Consequently, the order of multiplication will be the same as the temporal order of the links. If the robot visits poses a and b (see Fig. 2-2), the matrix relating b to the global frame is $B = AT$ (not TA).

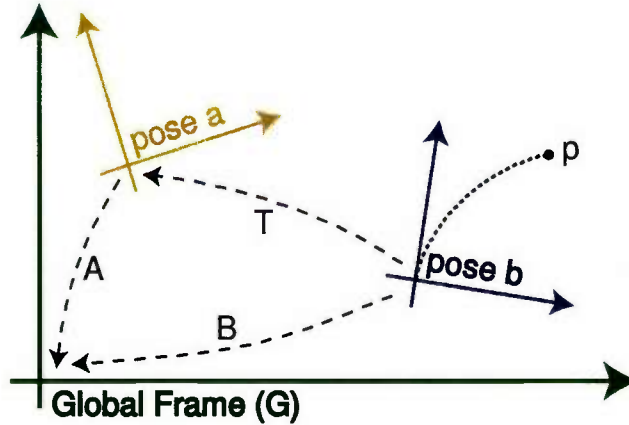


Figure 2-2: Coordinate transformations. Each robot pose represents a local coordinate system which can be related to the global coordinate by a rigid-body transform. Two local coordinate systems can also be related, e.g., $B = AT$. Point p , expressed with respect to coordinate frame B , can be projected to coordinate frame A by Ap and to the global coordinate frame by either Bp or ATp .

The product of two rigid-body transformations can be expressed directly in terms of their parameters. For example, consider $B = AT$:

$$\begin{aligned} x_B &= \cos(\theta_A)x_T - \sin(\theta_A)y_T + x_A \\ y_B &= \sin(\theta_A)x_T + \cos(\theta_A)y_T + y_A \\ \theta_B &= \theta_A + \theta_T \end{aligned} \quad (2.2)$$

It is also convenient to solve $B = AT$ for T ; this will arise in the prediction steps of the Extended Kalman Filter and in the residual computations for non-linear

optimization:

$$\begin{aligned} x_T &= \cos(\theta_A)(x_B - x_A) + \sin(\theta_A)(y_B - y_A) \\ y_T &= -\sin(\theta_A)(x_B - x_A) + \cos(\theta_A)(y_B - y_A) \\ \theta_T &= \theta_B - \theta_A \end{aligned} \quad (2.3)$$

Finally, it is also useful to invert a rigid-body transformation. This can also be done directly via the parameters. Letting $T = A^{-1}$, we have:

$$\begin{aligned} x_T &= -\cos(\theta_A)x_A - \sin(\theta_A)y_A \\ y_T &= \sin(\theta_A)x_A - \cos(\theta_A)y_A \\ \theta_T &= -\theta_A \end{aligned} \quad (2.4)$$

Constraints are uncertain quantities, having arisen from noisy sensor observations. This uncertainty can be represented as a probability distribution over the parameters of the rigid-body transformation. We assume that the probability distribution can be well-approximated as a multi-variate Gaussian distribution. This common approximation is motivated both by the quality of fit and by the computational conveniences that result. This distribution can be written in terms of the parameters of the rigid-body transformation:

$$\Sigma = \begin{bmatrix} \Sigma_{xx} & \Sigma_{xy} & \Sigma_{x\theta} \\ \Sigma_{yx} & \Sigma_{yy} & \Sigma_{y\theta} \\ \Sigma_{\theta x} & \Sigma_{\theta y} & \Sigma_{\theta\theta} \end{bmatrix} \quad (2.5)$$

When performing the operations on transformations, we must also compute the covariance of the new transformations. This is done by computing the Jacobians of the output as a function of the inputs, and summing the scaled covariances of the inputs. For example, consider the composition of two rigid-body transformations, $C = AB$. Let Σ_A be the covariance of the transformation A , and J_A be the Jacobian of the parameters of C with respect to the parameters of A . The covariance of C is then:

$$\Sigma_C = J_A \Sigma_A J_A^T + J_B \Sigma_B J_B^T \quad (2.6)$$

Σ_A is the covariance of transformation A , J_A is the Jacobian of the parameters of C with respect to the parameters of A , and J_B is the Jacobian of the parameters of C with respect to the parameters of B .

2.4 Constraints

A constraint (or edge) is a set of simultaneous equations f that relates the state variables x to some observed quantity z . For a particular constraint i , we write:

$$z_i = f_i(x) \quad (2.7)$$

The difference between the observed and predicted values of an observation is the *residual* r_i :

$$r_i = z_i - f_i(x) \quad (2.8)$$

When the residual is scaled by the constraint's confidence Σ_i^{-1} , the result is the χ^2 error. Note that the confidence is the inverse of the covariance matrix.

$$\chi_i^2 = (z_i - f_i(x))^T \Sigma_i^{-1} (z_i - f_i(x)) \quad (2.9)$$

As with virtually all previous work on robotic mapping, we assume that each edge is uncorrelated with other edges in the graph. In many cases, this is a reasonable assumption: errors in sensors are often well-modelled as additive independent noise. In reality, however, sensor observations are often correlated, at least to a minor degree. For example, a laser range scanner might be sensitive ambient temperature fluctuations, and a vision-based system is sensitive to intrinsic calibration errors. In both of these cases, multiple observations are affected by a single error source. This makes the measurements correlated. Since this effect is typically minor, we neglect it.

2.4.1 Common Constraint Forms

Different types of sensors produce different types of geometrical information. A lidar, for example, can measure the relative position and orientation of features with respect to the robot. Such an observation fully specifies the position of the feature with respect to the robot, and is called a rigid-body constraint. The observation equations for a rigid-body constraint are given in Eqn. 2.3.

Monocular cameras can only measure the bearing to a feature. If the bearing of node B is observed from node A (a “bearing-only” constraint), we have:

$$z_i = \arctan2(b_y - a_y, b_x - a_x) \quad (2.10)$$

Various radio-frequency and acoustic sensors are capable of measuring the range between a feature and the robot, but cannot determine the bearing. A range-only constraint is described by:

$$z_i = ((b_x - a_x)^2 + (b_y - a_y)^2)^{1/2} \quad (2.11)$$

2.4.2 Linearization

As seen above, the observation equations are not generally linear in the state variables. However, a serviceable approximation can usually be obtained by linearizing

the equations around the current state estimate x_0 :

$$F_i = f_i(x_0) \quad (2.12)$$

$$J_i = \left. \frac{\partial f_i}{\partial x} \right|_{x_0} \quad (2.13)$$

$$f_i(x) \approx F_i + J_i(x - x_0) \quad (2.14)$$

We can write the χ^2 error in terms of this linearization. First, we write the residual (error) of the constraint as $r_i = z_i - f_i(x_0) = z_i - F_i$. We also let $d = (x - x_0)$. These substitutions allow us to rewrite Eqn. 2.9 as:

$$\chi_i^2 \approx (J_i d - r_i)^T \Sigma_i^{-1} (J_i d - r_i) \quad (2.15)$$

The χ^2 error for the entire pose/feature graph is the sum of the χ^2 error of each edge, since each edge is assumed to be statistically independent:

$$\chi^2 = \sum_{i \in \text{edges}} \chi_i^2 \quad (2.16)$$

If we stack the J_i and r_i matrices, and create a block-diagonal matrix from the Σ_i^{-1} matrices, we can write:

$$\chi^2 \approx (Jd - r)^T \Sigma^{-1} (Jd - r) \quad (2.17)$$

where

$$J = \begin{bmatrix} \cdots & J_1 & \cdots \\ \cdots & J_2 & \cdots \\ & \vdots & \end{bmatrix} \quad r = \begin{bmatrix} r_1 \\ r_2 \\ \vdots \end{bmatrix} \quad \Sigma^{-1} = \begin{bmatrix} \Sigma_1^{-1} & & \\ & \Sigma_2^{-1} & \\ & & \ddots \end{bmatrix} \quad (2.18)$$

As a preview of the optimization methods to come, we can differentiate Eqn. 2.17 with respect to d to find the value of d that minimizes the χ^2 error:

$$\begin{aligned} \frac{\partial \chi^2}{\partial d} &= 2J^T \Sigma^{-1} Jd - 2J^T \Sigma^{-1} r = 0 \\ d &= (J^T \Sigma^{-1} J)^{-1} J^T \Sigma^{-1} r \end{aligned} \quad (2.19)$$

The value d represents a change in the state variable that will minimize the χ^2 error. However, inverting $(J^T \Sigma^{-1} J)$ is impractical in many cases. This has spurred the development of many algorithms, including those in Chapter 4 and 5.

Due to the non-linearity of the constraints, computing d will generally not reduce the error to the global minimum in a single iteration: several iterations are often required.

2.4.3 Interpretation of the Chi-Squared Error

The χ_i^2 error is the square of the Mahalanobis distance, which has a natural geometric interpretation: it is the number of standard-deviations of error between the predicted and observed values. Mahalanobis distances are used as plausibility tests: values above three occur with a probability of about 0.2% and thus strongly indicate either an error in z (e.g., a measurement outlier), the confidence Σ_i , or the state.

The number of degrees of freedom k of a graph is equal to the number of independent components (the number of observation equations) minus the size of the state vector [73]. For example, a 2D pose graph with N nodes and M edges has $3M - 3N$ degrees of freedom, assuming that each constraint is a rigid-body transformation.

The chi-square distribution gives the probability of the maximum likelihood map having a given χ^2 error as a function of the number of degrees of freedom. The chi-square distribution for k degrees of freedom has mean k and variance $2k$.

We can compare our χ^2 error to the value predicted by the chi-square distribution. If our χ^2 error is highly improbable, it indicates one of two potential problems. The first possibility is a mismatch between the sensor noise model and the actual noise of the sensor. For example, if the sensor noise model indicates a much lower noise level than is present in the actual observations, the χ^2 error will be larger than expected. The second possibility is that our current state estimate is not close to the maximum likelihood map. In this case, it is an indication that we should attempt to further optimize the map.

When comparing the performance of optimization algorithms across a number of different optimization problems, the dependence of the χ^2 error on k makes it difficult for a human to determine whether the χ^2 error is probable or not. Instead, we consider the probability density function for χ^2/k : its expectation is always 1.0 and its variance is $2/k$. Consequently, for large k , the distribution will be ever-more tightly centered around 1.0.

If we trust our sensor noise model, a χ^2/k value near 1.0 indicates that the state estimate is near the maximum likelihood solution. Larger values indicate that additional optimization is needed.

Conversely, if the optimization algorithm has converged to a value that is far from 1.0, it indicates a problem with the sensor noise model. If the observation covariance matrices are incorrectly scaled by a factor α , the normalized χ^2 will be $1/\alpha$.

Because of these useful properties, the χ^2 values reported in this thesis are always normalized by the number of degrees of freedom.

2.4.4 State Squared Error

Different optimization algorithms will produce different estimates of the global minimum x_{opt} over time. Evaluating these algorithms requires us to be able to determine which estimates are better than others.

The χ^2 error gives us a means to compare different estimates, and it is a good measure in that it is the quantity that the optimization algorithms are explicitly attempting to minimize. However, from a practical perspective, we wish to know how

similar the *maps* are to the optimal map: the χ^2 error tells only about the errors of the constraints, not the positions of the nodes.

As Fig. 2-3 illustrates, χ^2 error is not necessarily a good indicator of map quality. In this figure, a map with a χ^2 error of 3.6 is of much lower quality than a map with a χ^2 error of 33.2.

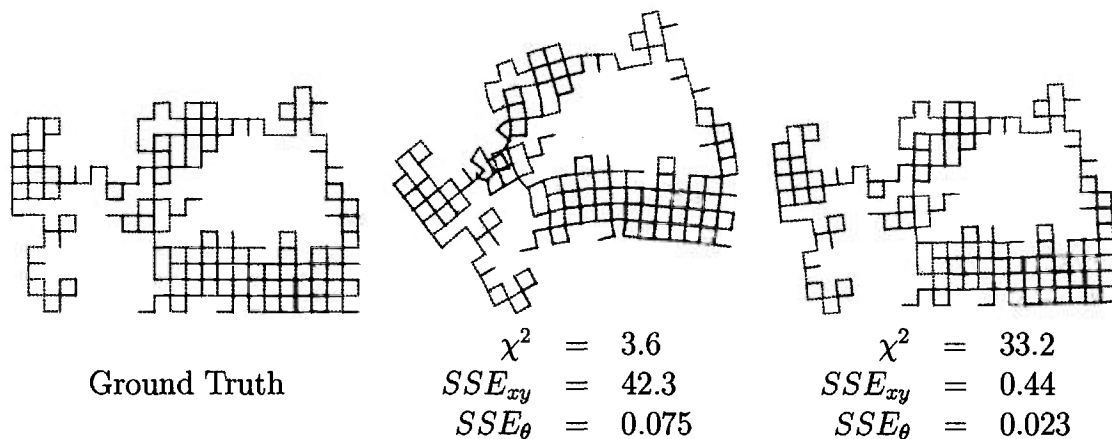


Figure 2-3: State Squared Error Example. χ^2 error is not always a good indicator of map quality. From a practical perspective, the right map is superior to the middle map, despite the fact that the middle map has one-tenth the χ^2 error. The SSE metrics, in contrast, clearly identify the map on the right as being better than the middle map.

The χ^2 error is not necessarily a good measure of map quality because different map distortions affect the χ^2 error to different degrees. In particular, the non-linear nature of the mapping problem results in local minima and long valleys in which the χ^2 error is both small and slowly changing. It is thus possible to have a small χ^2 error and yet have a map that is very different from the maximum likelihood map. As we will see in Chapter 4, this is one of the reasons that many map optimization algorithms either get stuck or converge very slowly. The opposite is also true: it is possible to have a high χ^2 error and be very close to the maximum likelihood map.

When x_{opt} is known, we can compute the distance of the current graph configuration x from the optimal value x_{opt} . The metric $\|x - x_{opt}\|^2$ is not a good choice, however, since it intermixes components of the state vector having different units (positions in meters and rotations in radians). Instead, we compute two different metrics so that positions and rotations are considered separately. In both cases, we use the mean squared error, but to avoid confusion with the χ^2 error (which is the squared error of the constraints), we will call these metrics the “state squared error”, reflecting that they are the squared error of the state variables.

We define SSE_{xy} to be the mean squared error of the (x, y) positions of each node in the pose graph relative to x_{opt} , and SSE_{θ} to be the mean squared error of the θ positions of each node in the pose graph.

Implementing this error metric presents a minor wrinkle, however: in most map building applications, the position of the map has no global “anchor”: translating and rotating the whole map has no effect on errors of node-to-node constraints, but has an obvious effect on the (x, y, θ) coordinates of those nodes. In other words,

projecting the whole map x by a rigid-body transform will not affect the χ^2 error but it will affect the SSE metrics as defined above.

Consequently, it is desirable to modify our definitions of the SSE metrics so that they are invariant to rigid-body transformations of the whole map. To do this, we introduce another rigid-body transformation T that we will apply to the map before computing the mean squared error of x with respect to x_{opt} . We will pick T to be the rigid-body transformation that minimizes SSE_{xy} . Minimizing SSE_{xy} has a tendency to minimize SSE_θ , but does not do so exactly. We have chosen to minimize SSE_{xy} rather than SSE_θ since all landmarks have a position, whereas some types of point features do not have an orientation.

The transform T can be computed in time proportional to the number of nodes using an algorithm developed by Horn [28]. Let a be the set of points corresponding to the positions of nodes in our current state estimate, and b be the positions of those points in the maximum-likelihood (or ground truth, if available) configuration. We write the position of the i^{th} point of a as (a_x^i, a_y^i) . Let $\bar{a}_x$ be the average x coordinate of the points in set a . We can compute the rigid-body transformation T in terms of intermediate variables M and N :

$$M = \sum_i (a_x^i - \bar{a}_x)(b_y^i - \bar{b}_y) - (a_y^i - \bar{a}_y)(b_x^i - \bar{b}_x) \quad (2.20)$$

$$N = \sum_i (a_x^i - \bar{a}_x)(b_x^i - \bar{b}_x) + (a_y^i - \bar{a}_y)(b_y^i - \bar{b}_y) \quad (2.21)$$

The parameters of the rigid-body transform T , which optimally projects points b to align with a (minimizing the sum of squared distances between a_i and Tb_i), are then given as:

$$T_\theta = \arctan2(M, N) \quad (2.22)$$

$$T_x = \bar{b}_x - \bar{a}_x \cos(T_\theta) + \bar{a}_y \sin(T_\theta) \quad (2.23)$$

$$T_y = \bar{b}_y - \bar{a}_x \sin(T_\theta) - \bar{a}_y \cos(T_\theta) \quad (2.24)$$

The SSE error metrics are complementary to χ^2 error. Map optimization is defined in terms of minimizing the χ^2 error, and so it remains an important consideration. However, different optimization algorithms will take different paths through the state space as they minimize the χ^2 error: the SSE measures give us insight into how rapidly these paths lead to x_{opt} , independently of the χ^2 error of those paths.

CHAPTER 3

Loop Closing

3.1 Introduction

Place recognition is a central problem in many robot applications. Whether a robot is delivering a package or guiding a tour, being able to recognize a location is critical. With a place recognition system, a robot could be trained to identify rooms or even particular areas inside rooms (e.g., the robot’s recharging station).

In mapping applications, place recognition is especially important. Without the ability to recognize previously-visited places, the position uncertainty of a robot increases without bound due to the ceaseless accumulation of dead-reckoning error. Place recognitions serve as constraints on the motion of the robot, allowing a correction of its dead-reckoning error. In the mapping context, place recognition is called “loop closing”: if the robot drives in a loop and back to its starting position (and recognizes it), the robot “closes” the loop. The quality and accuracy of the map is a function of the quantity and quality of the loop closures.



Figure 3-1: Three laser scans from the CSAIL dataset. Some environments present rich alignment cues (left: an elevator lobby). Incorrect matches arise from both spartan areas (middle: a corridor) and from repetitive/cluttered areas (right: office cubicles).

Loop closing is difficult for several reasons. Algorithms must be robust to modest changes in the environment caused by things like moving chairs or humans. Sensing limitations play a critical role. The data produced by laser scanners, for example, is metrically accurate (see Fig. 3-1), but it is often difficult to distinguish one room

from another: indoor environments tend to be composed of sets of straight walls and corners whose appearances are similar.

Camera-based systems, like those using the SIFT feature detector [38], provide a richer description of the environment (able to distinguish different posters on a wall, for example), but are still susceptible to ambiguity. Different offices, for example, may contain the same type of chair, and different outdoor environments contain the same types of street signs. The length of these feature vectors has a major impact on both total memory usage and the time required to index and search for similar features. Consequently, it is desirable to use shorter descriptors (such as PCA-SIFT [32]). Reducing the descriptor size generally increases the error-rate of matching: matching algorithms that are robust to higher error rates are thus very desirable.

Some authors have proposed using globally unique beacons, such as RFID tags [33], wireless networking base stations [25], or specialized sensor networks [54]. The resulting descriptors are concise and globally unique, but require the deployment of beacons and are not broadly applicable to all robotics applications.

In outdoor settings, GPS can provide a positional cue, however the accuracy of GPS (especially in consumer-grade devices) is insufficient for many applications. For example: while GPS can usually tell a car what *road* it is on, it can't reliably indicate which *lane* it is in. For many applications, greater accuracy is needed. This additional accuracy can be achieved by recognizing places: each recognition allows the map to be refined.

In a map-building context, place recognition is known as “loop closing”, reflecting the common case in which a robot travels around a large loop and returns to its original position (thus “closing” the loop). However, any place recognition adds an edge to the pose graph, creating a new cycle—the robot does not need to physically travel in a circle in order for a loop closure to occur.

Loop closing has been approached in a variety of ways. The discovery of topological loop closures has been performed with odometry alone using hidden Markov models [61]. Folkesson [17] uses negative information to validate loop closure hypotheses, while Kuipers uses bootstrap learning to identify places [35]. Bosse validated his loop closures using a loop-consistency test [5].

In robotic mapping applications, place recognition has generally been cast in terms of explicitly associating landmarks with previously observed landmarks. This process is called “data association”. Data association on one (or just a few) features at a time is highly susceptible to errors, and data-association errors can cause catastrophic divergence of the map.

In an attempt to decrease their error rate, data association algorithms typically perform association on groups of observations at once. In analogy, it is easier to recognize astronomical constellations than single stars. Given the mean and covariance of a set of tracked landmarks, the probability of a set of observations matching that set can be computed. However, as the number of landmarks grows, the number of possible assignments grows exponentially. Many authors have proposed solutions to this problem, providing reasonably efficient search heuristics that often perform well [46, 2]. Error rates can also be addressed by revisiting data associations previously made [26].

Even if the number of landmarks is kept small enough to ensure good performance, such methods still require that the position estimates and covariances of landmarks be maintained. This is a major imposition for robots that do not otherwise care about landmarks and would not otherwise maintain their covariance. It is also an imposition for mapping robots that use an optimization method in which landmark covariance data is not readily available, such as FastSLAM [44], information-form filters [70, 13, 76], non-linear optimization methods (Gauss-Seidel relaxation [12], Multi-Level Relaxation [20]), and the methods described in Chapters 3 and 4 and its derivatives [24, 23].

In this chapter, we describe an approach that attempts to close loops by performing a number of simple pose-to-pose matches. Given a number of naive matches (“hypotheses”), our goal is to identify those that are correct. A *set* of these hypotheses has an interesting property: the correct hypotheses all agree with each other (since there is only one true configuration), whereas the incorrect hypotheses tend to be incorrect in different ways (and thus do not agree with each other). Our algorithm exploits this property by computing the subset of hypotheses that is most self-consistent. In comparison to the exponential cost of performing data association amongst N features, the cost of our approach is $O(N^2)$ in the number of hypotheses. Our approach is similar to CCDA [2], except that our notion of consistency is not limited to discrete boolean quantities.

Our approach does not require landmarks to be tracked, and thus does not require covariance information about the landmarks or a data association algorithm. All that is required is the ability to compute possible rigid-body transformations between two sensor scans. Our approach can be viewed as a outlier-rejection step, extending previous pose-to-pose methods [41].

A major advantage of our proposed method is that it computes the “second-best” set of mutually-consistent hypotheses as well. When using RANSAC [15] or Joint Compatibility Branch and Bound (JCBB) [46], the second-best solution is generally a trivial variation on the best solution: it is thus not very informative. In contrast, the second-best solution reported by our new approach is *orthogonal* to the best solution—i.e., represents a substantially different interpretation of the data. Our method allows the quality of these two solutions to be quantitatively compared, allowing ambiguous solutions to be safely discarded. (If the data can be equally well explained in two different ways, it is not safe to trust either explanation.) In these ambiguous cases, our method will occasionally reject hypotheses that are correct. Rejecting correct hypotheses has an impact on the quality of the final map (since some information is lost), but it does not result in the sort of rapid divergence that accepting an incorrect hypothesis can cause.

This chapter presents our algorithm and results using both camera-derived features and laser-derived features, illustrating the flexibility of the method.

3.2 Method Overview

Our loop-closing algorithm operates in several stages. We first generate a number of hypotheses. Given two robot positions (at different points in time), we can estimate the probability that those two positions are close enough together for them to have observed the same environmental features. If so, we attempt to generate a hypotheses relating the position of those poses. The particular hypothesis-generation algorithm varies according to the types of sensors being used: cameras and lidars, for example, produce very different types of data that require different matching algorithms.

We group the resulting hypotheses into “hypothesis sets”. Our task is to identify those hypotheses that are correct. By considering pairs of these hypotheses at a time, we will identify which hypotheses are pairwise-compatible. Using an algorithm based on the spectral properties of the pair-wise compatibility matrix, we can identify the set of hypotheses that are most compatible with each other.

3.3 Hypothesis Generation

3.3.1 Determining when poses overlap

The first step in identifying loop closure hypotheses is to identify those poses that might contain overlapping sensor readings. Given a pose a , we wish to find all other poses in the graph whose sensor readings might overlap those from pose a . This could be determined from the posterior distribution (Eqn. 1.1) by marginalizing out pose a and any landmarks. Instead, we describe a conservative approximation that is much easier to compute. In particular, our method does not require knowledge of the posterior pose and landmark covariances.

Dijkstra Projection

We estimate the relative positional uncertainty between nodes a and b by searching for the minimum-uncertainty path through the pose/feature graph that connects them. This path can be found by using variant of Dijkstra’s Algorithm [55]:

In order to compare the uncertainty of two paths, we use the determinant of the paths’ covariance matrices. Geometrically, the determinant can be interpreted as the area of the covariance ellipse, and is thus proportional to the area of the region in which pose b is likely to be found given the position of pose a . Larger search areas generally result in more computational effort. Consequently, picking the path with the smaller determinant generally minimizes the amount of computation needed to find pose b .

The Dijkstra Projection is conservative in the sense that it will overestimate the uncertainty. For example, if a pose graph contains multiple independent paths between nodes a and b , the uncertainty of their relative positions will be less than that computed by the Dijkstra projection. Overestimating the uncertainty will result in larger search areas and thus greater computational effort when searching for loop closures.

Algorithm 1 Dijkstra Projection from node a

```

Initialize a set of paths  $P = \{\}$ 
for all edges  $e$  leaving node  $a$  do
   $P = P \cup e$ 
end for
visited( $a$ ) = true
while some nodes unvisited do
  find  $p$ , the minimum-uncertainty path in  $P$ 
  if !visited(destination( $p$ )) then
    optimal-path( $b$ ) =  $p$ 
    visited( $b$ ) = true
    for all edges  $e$  leaving node  $b$  do
       $P = P \cup \text{compose-transforms}(p, e)$ 
    end for
  end if
end while

```

However, even if the exact covariance could be easily computed, the search areas are often artificially enlarged in order to ensure that loop closures can be found whenever possible. Over-confident sensor-noise characterizations, for example, cause the “exact” covariance to be too small, and can result in missed loop closures. If loop closures are consistently missed, the quality of the map will suffer. In short, the conservatism of the Dijkstra algorithm does not typically cause any problems.

Sensor range

In addition to the relative positions (and uncertainties) of two nodes, we must know the sensor range in order to determine whether two sensor scans might overlap: large sensor ranges make it more likely that nodes will have overlapping fields of view. In indoor environments, the effective sensor range varies dramatically depending on the environment: a robot may only be able to sense a few meters when inside an office, but may be able to sense tens of meters outdoors.

We model the sensor range at a particular node a as a circle with center c_a (whose coordinates are expressed in terms of a) and radius r_a . We compute c_a as the centroid of the landmarks observed, with r_a being the average distance of the landmarks to the centroid. In order to test whether sensor observations at node b might overlap with those at node a , we similarly compute a circle for node b with center c_b and radius r_b , but project c_b into the same coordinate system as c_a using the Dijkstra projection between poses a and b . Let Σ_{ab} the uncertainty of the transformation from a to b as

given by the Dijkstra projection. We can write the Mahalanobis distance as:

$$d = (c_b - c_a) \quad (3.1)$$

$$s = \max(0, \|d\| - r_a - r_b) \frac{d}{\|d\|} \quad (3.2)$$

$$mahl = s^T \Sigma_{ab}^{-1} s \quad (3.3)$$

In the equations above, d is the Euclidean vector between the circle centers, and s is the separation vector between their sensor ranges (the vector d reduced in magnitude by the sum of the sensor ranges). If the resulting Mahalanobis distance is less than 3, it is reasonably likely that the two nodes have overlapping sensor data. In aggregate, our approach for computing the pair-wise “compatibility” is similar to the metric used by GraphSLAM [72], with the improvement that we explicitly consider sensor range. Our use of these pair-wise compatibilities differs, however.

3.3.2 Searching for a match

Given a prior on the relative position between a and b (as computed by the Dijkstra projection), and sensor observations from pose a and b , the task is now to compute a rigid-body transformation that is likely to correctly align a and b . The techniques for doing this are sensor-dependent; we have implemented both a vision-based method and a laser-based method. These are described in more detail in Appendix A, but are summarized here.

Vision-derived Local Landmark Matching

Our vision data sets were collected during the 2007 DARPA Grand Challenge National Qualifying Event by Team MIT’s vehicle, Talos [37]. Our vision data sets use SIFT features extracted from an array of calibrated cameras providing a nearly 360 degree field-of-view. Note that these cameras were essentially monocular: their fields of view did not significantly overlap.

In addition to cameras, Talos had a high-end inertial measurement unit (IMU)/GPS system giving it very good positioning capabilities. In our experiments, we evaluate the performance of our algorithm on both the high-quality IMU data as well as artificially degraded data that is representative of a more typical vehicle.

SIFT features are extracted from each camera frame and are matched against a set of active “tracks”. A track is comprised of a SIFT feature descriptor and a list of the observations of that feature (the position of the car and the bearing to the feature).

Periodically, the measurements for each track are examined to see if the position of the SIFT feature can be determined. Since each observation of a feature consists of only a bearing to the feature, the feature must be observed from several different positions in order to compute its position. Each bearing to the feature corresponds to a ray: when the rays from different observations intersect, we can estimate of the location of the feature. Once the feature is localized, the track is reset so that the



Figure 3-2: Overhead view of vision dataset area. The car made three counter-clockwise loops, with each loop approximately 610 m in length.

feature can be independently localized again and again. The result of this process is a sequence of independent rigid-body transformations relating the vehicle to the feature.

At this point, we have a list of SIFT landmark observations (rigid-body constraints) for every node in the graph. In order to find the rigid-body transformation between two nodes, we compute the matches between the SIFT landmarks of the two nodes. These SIFT correspondences are computed based on the SIFT descriptors. Given two sets of matching SIFT features, we compute a rigid-body transformation that best aligns those landmarks. We can score each rigid-body transformation by counting the total number of landmarks from node b that are projected near their counterpart in node a ; using this metric, we find the best rigid-body transform using RANSAC [15].

The rigid-body transform output by RANSAC is a hypothesis relating nodes a and b . The quality of this hypothesis is strongly dependent on the reliability of matching SIFT features based on their descriptors. When the whole 128-dimensional SIFT descriptor is used, these SIFT matches are often correct, leading to good rigid-body transformations. As the descriptors are reduced in size, however, the transforms become increasingly unreliable.

Another source of error results from correct but undesirable SIFT matches. For example, portions of the car are visible in some cameras: these produce SIFT observations that move with the car. While these obvious parts of the car could be

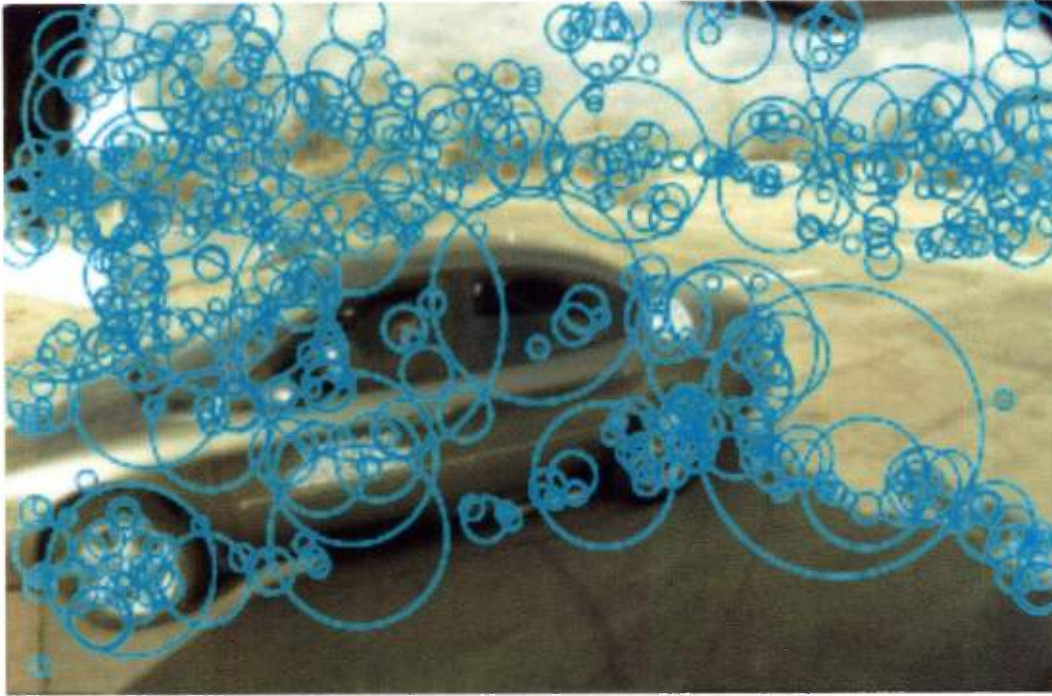


Figure 3-3: Sample SIFT detections. Cyan circles indicate SIFT features. Note how the other car, the shadow cast by our own car, and portions of our own car (black regions on left and top) cause undesirable SIFT detections.

masked out, other related effects are difficult to mask. For example, as the car drives around, it casts a shadow on the ground. This shadow generates many SIFT detections. Other moving cars can also cause undesirable SIFT matches. Since many of these sources of error are unavoidable, we did not attempt to mask any of them—instead, we let our proposed method detect and correct the resulting errors.

Laser Matching

We have processed a number of datasets with laser scanner data, including several common mapping benchmarks. These datasets were collected indoors using a single scanner with a 180-degree field-of-view. The complexity of these datasets varies significantly in terms of the noise of the range measurements, the rate at which observations are made, and the complexity and size of the environment.

Line features are extracted from individual point scans. Two correspondences between node a and b (two lines from each pose) are used to compute a rigid-body transformation. This rigid-body transformation is then scored by spatially correlating the points from b with a rasterized version of the points from a . The rigid-body transformation yielding the highest correlation is then subjected to a brute-force refinement step, in which the rigid-body transformation is numerically optimized.

The result is a rigid-body transformation T that causes the points from scan b to “line up” with the points from scan a . If the Mahalanobis distance of T is less than a threshold (typically 3) from the prior derived from the Dijkstra projection, T is output as a hypothesis.

In indoor environments (where there is often repeated structure), the probability of an incorrect match increases with the uncertainty of the prior. Failures can occur when the line correspondences are incorrect but happen to lead to a good correlation.

3.4 Hypothesis Sets

A single hypothesis by itself is impossible to validate. However, if two hypotheses are considered jointly, it is possible to test them against each other, even if those two hypotheses relate different pairs of nodes.

3.4.1 Pairwise Consistency

Given two hypotheses, we can construct a closed loop comprised of two hypotheses plus two additional rigid-body transformations obtained from Dijkstra projections (see Fig. 3-4).

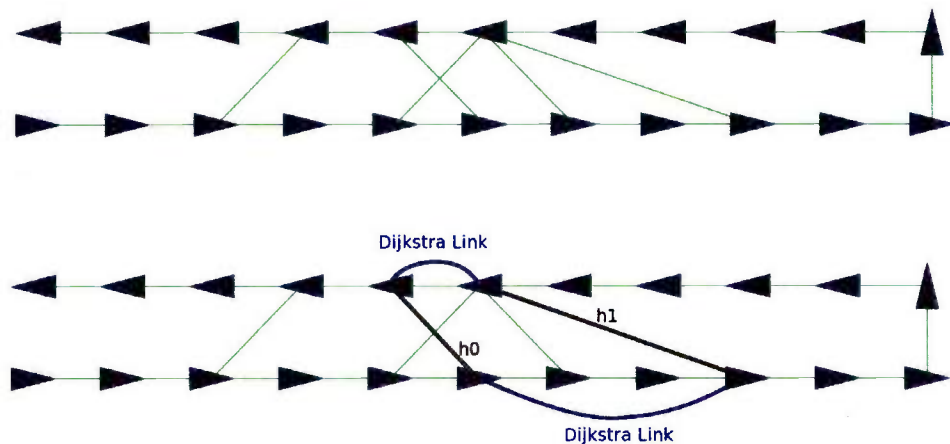


Figure 3-4: Hypothesis Set Loop. A hypothesis set contains a number of hypotheses (top). Given a pair h_0 and h_1 of them (bottom), a loop can be formed by adding a pair of Dijkstra links. The rigid-body transformation around this loop is compared to the identity transform. If either hypothesis is incorrect, it is unlikely that the resulting transformation will be close to the identity transform.

Naturally, the rigid-body transformation around any closed loop *should* be the identity matrix. Due to measurement errors (even for “good” hypotheses), it isn’t ever *quite* the identity matrix, but this uncertainty is captured by the uncertainties associated with the hypotheses. Thus we can compute the rigid-body transformation around the loop (including its net covariance), and compute the probability that the resulting rigid-body transformation is actually the identity matrix. This probability is the pair-wise consistency of the two hypotheses. If both hypotheses are correct, it is likely that the probability will be large. However, if either hypothesis is wrong, the rigid-body transformation of the loop will not be the identity matrix; this results in a low probability.

3.4.2 Finding the optimal subset

Given a set of N hypotheses, we compute the pair-wise consistency for each pair, yielding an $N \times N$ “consistency” matrix A . We now wish to find the subset of hypotheses that is maximally self-consistent. In other words, we wish to find a subset of hypotheses whose pair-wise consistency is, on average, the greatest.

A hypothesis set can be viewed as a graph, with each hypothesis represented as a node. The (weighted) adjacency matrix of this graph is the pair-wise consistency matrix A .

Our solution is based on our previous work, an algorithm called “Single Cluster Graph Partitioning” (SCGP) [51, 49, 4]. SCGP is a graph partitioning method that attempts to find a single cluster of well-connected nodes, rejecting other nodes.

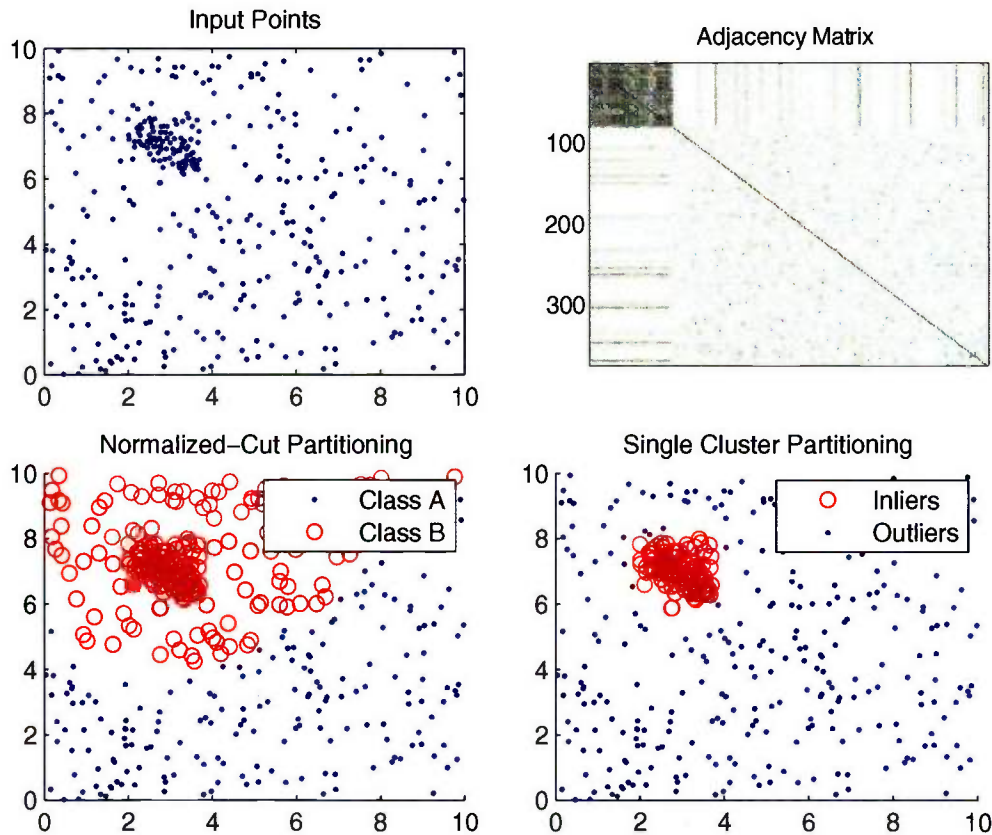


Figure 3-5: SCGP versus Normalized Cuts. Given a set of (x, y) points, each representing a hypothesis (top left), the goal is to identify the subset of hypotheses that forms the maximally self-consistent cluster. The adjacency matrix is plotted such that the desired points have small indices: their higher-than-average pair-wise consistency is clearly visible (top right). Our method, SCGP, produces the desired segmentation (lower right). Normalized-Cuts, also based on spectral clustering, uses a different clustering metric which results in a poor solutions on this type of problem.

Previous spectral clustering methods like Normalized-Cuts attempt to divide the graph into two parts such that clusters have both high intra-cluster consistency and low inter-cluster consistency. This metric does not work well on hypothesis filtering problems since incorrect hypotheses do not generally form a coherent cluster.

Consider a toy hypothesis filtering problem in which samples from a Gaussian distribution are mixed with uniform noise (see Fig. 3-5). Each point represents a hypothesis of the center of the Gaussian distribution: pair-wise consistency of hypotheses is simply a function of distance. As the figure illustrates, algorithms like Normalized-Cuts do not produce the desired output.

Let v be an $N \times 1$ *indicator vector* such that $v_i = 1$ if the i^{th} hypothesis should be accepted, and $v_i = 0$ otherwise. The sum of the compatibilities between the accepted hypotheses is $v^T A v$, and the number of accepted hypotheses is $v^T v$. Thus, the average pair-wise consistency $\lambda(v)$ for a subset of hypotheses v is simply:

$$\lambda(v) = \frac{v^T A v}{v^T v} \quad (3.4)$$

As an intuition-building exercise, suppose we have a set of accepted hypotheses v_0 , and we want to decide whether to add another hypothesis to it. Adding a new hypothesis will add N pair-wise consistencies (of differing weights) to the numerator, but will also increase the denominator by 1. If adding those pair-wise consistencies increases the average amount of pair-wise consistency per node, then we should add that hypothesis. Note that our actual algorithm, unlike this exercise, does not greedily select hypotheses one by one.

Now, our task is to find an indicator vector v that maximizes $\lambda(v)$. Unfortunately, this problem has an exponentially-large search space. In the tradition of spectral graph-cutting methods like MinMaxCuts [11] and NormalizedCuts [62], we relax v to allow continuous values. We will find an optimal answer for continuous-valued v , and then later discretize v back into a proper indicator vector.

With v now a continuous-valued vector, we can look for extrema of $\lambda(v)$ by differentiating $\lambda(v)$ with respect to v and setting the result to zero:

$$\frac{d\lambda(v)}{dv} = \frac{A v v^T v - v^T A v v}{(v^T v)^2} = \frac{A v - \lambda(v) v}{v^T v} = 0 \quad (3.5)$$

Note that we have exploited the fact that A is symmetric. Rearranging terms, we get:

$$A v = \lambda(v) v \quad (3.6)$$

By inspection, we see that the value of v that maximizes $\lambda(v)$ is the dominant eigenvector of A (and $\lambda(v)$ will be the dominant eigenvalue).

We note that the densest subgraph can be found in polynomial time by performing a number of max-flow computations on the hypothesis graph [21]. Our spectral approach is faster and (perhaps more importantly) leads to a confidence metric, as described below.

3.4.3 The Eigenvectors of the Consistency Matrix

As we have shown, the dominant eigenvector of the consistency matrix A yields the optimal continuous-valued indicator vector v . The second-most dominant eigenvector of A , however, is also useful.

Recall that A is symmetric and that the eigenvectors of a symmetric matrix are orthogonal [67]. Intuitively, this means that the hypothesis sets described by the first and second eigenvalues represent *different explanations of the data*. Thus, the ratio of the first and second eigenvalues conveys the relative confidence of the solution.

In algorithms like JCBB or RANSAC, it is also possible to keep track of the second-best answer, but the second-best answer is almost invariably a trivial variation on the best answer. SCGP's ability to evaluate the next-best *orthogonal* solution is unique, and extremely useful.

The first two dominant eigenvectors e_1 and e_2 of a symmetric matrix A can be efficiently computed using the Power Method as follows:

Algorithm 2 Power Method for extracting the two dominant Eigenvectors and Eigenvalues of a symmetric matrix A . A total of K power iterations are performed, with $K = 5$ being a reasonable value.

```

for  $i = 1; i \leq 2; i = i + 1$  do
   $e_i$  = a random  $N \times 1$  positive vector,  $\|e_i\| = 1$ 
  for  $iters = 0; iters < K; iters = iters + 1$  do
    for  $j = 1; j < i; j = j + 1$  do
       $\alpha = e_j \cdot e_i$ 
       $e_i = e_i - \alpha e_j$ 
    end for
     $e_i = e_i / \|e_i\|$ 
     $e_i = A e_i$ 
     $\lambda_i = \|e_i\|$ 
  end for
   $e_i = e_i / \|e_i\|$ 
end for

```

This algorithm performs five iterations of the power method per eigenvector, which is adequate in virtually all cases. (When the eigenvalues are closely spaced and convergence is slow, the confidence test based on the ratio of the eigenvalues would reject the eigenvectors anyway.) The inner loop in j removes any component of earlier eigenvectors from the current eigenvector.

We reject the whole hypothesis set when the confidence ratio λ_1/λ_2 is less than a threshold; we use a threshold of 2.

3.4.4 Discretization of the Indicator Vector

In an earlier step, we allowed v , an indicator vector nominally consisting of only zeros and ones, to take on arbitrary continuous values. We now need to discretize v back

into an indicator vector. We will denote the new discretized version of v as w . Given a threshold t , we write w as:

$$w_i(t) = \begin{cases} 1 & \text{if } v_i \geq t \\ 0 & \text{otherwise} \end{cases} \quad (3.7)$$

There are at least two reasonable ways of selecting t : picking t such that it maximizes the dot product $\hat{w} \cdot v$, or so that it maximizes Eqn. 3.4. Both work well, often producing the same answers. In our experiments, we have used the dot-product method.

The dot-product method can be implemented in $O(N \log(N))$ time. First, note that there are at most N different w vectors, since there are at most N distinct values in v . If the values of v are sorted, the dot product $w(t) \cdot v$ for increasing values of t can be incrementally computed in $O(1)$ time. Thus the total time complexity is $O(N + N \log(N)) = O(N \log(N))$.

We note that the discrete indicator vector w is not necessarily the globally optimal indicator vector. Instead, it is a discretized version of the optimal continuous indicator vector. While v is an approximation, its performance is very good.

Once the discrete-valued indicator vector has been computed, we add the hypotheses with $v_i = 1$ to the graph, discarding the others. (We assume that low-confidence hypothesis sets, as measured by λ_1/λ_2 , have already been discarded.) Adding edges to the graph allows an improved posterior to be computed; the computation of this posterior is the subject of Chapters 4 and 5.

3.4.5 Improving Hypothesis Set Construction

The computation of the pair-wise consistencies incorporates information about existing edges in the graph, since these edges are used to compute the Dijkstra Links shown in Fig. 3-4. As a reminder, the rigid-body transformations of the two hypotheses and the Dijkstra links are composed together, with the resulting rigid-body transformation compared to the identity matrix. However, these Dijkstra links are not known with certainty, and errors in those links can distort the resulting probability calculations.

In general, the error (and uncertainty) between two poses is related to the length of the trajectory between them: longer trajectories allow for greater accumulation of error. With this in mind, we can construct hypothesis sets so that the Dijkstra links will be derived from relatively short trajectory segments.

Each hypothesis relates a pair of nodes, which we denote as (a, b) . Our strategy is to restrict hypothesis sets to contain hypotheses of the form $(a + k_i, b + k_j)$ where each k is a relatively small number ($|k| \leq k_{max}$). Given two hypotheses $(a + k_1, b + k_2)$ and $(a + k_3, b + k_4)$, we can construct a loop using two Dijkstra links, $(a + k_1, a + k_3)$ and $(b + k_2, b + k_4)$. This strategy keeps the Dijkstra links short (at most $2k_{max}$ poses long), allowing us to limit the amount of error they introduce.

The value of parameter k_{max} is based on the local-navigation accuracy of the robot. In our systems, we set k_{max} to 8 poses, which typically translates to about 8 m of

trajectory for our indoor robots, or about 16 m of trajectory for our full-sized car. Larger values of k_{max} increase the number of opportunities to generate hypotheses for any given hypothesis set: this is important since we require a set to have a minimum number of hypotheses before it is processed (we used a minimum size of 4 in our experiments).

Large values of k_{max} can lead to inaccurate Dijkstra links. In general, this causes pair-wise consistency values to be smaller than ideal, not larger. (It is more likely that an error will result in a bad loop than a good loop.) Our method is robust to these types of errors: the algorithm continues to identify sets of hypotheses that are correct, but there is an increase in the number of falsely rejected hypotheses. In general, the performance of our method is good over a broad range of k_{max} values.

3.4.6 Analysis of Robustness to Outliers

At the heart of the SCGP is the pairwise consistency test which is used to construct the adjacency matrix. The behavior of the pairwise test can have a profound effect on the success or failure of the algorithm. SCGP can be applied to a large class of outlier rejection problems [51]; in the current context, correct hypotheses can be viewed as inliers, while incorrect hypotheses are outliers.

The most often encountered failure mode of SCGP is that SCGP selects most or all of graph, rather than just the subset of inliers. This happens when there are many outliers and those outliers have fairly high consistency with each other. In this section, we show how much outlier consistency SCGP can tolerate while still returning the correct answer.

Consider a scenario in which there are N_t true hypotheses and N_f false hypotheses. Assume that the pairwise consistency test reliably assigns edge weights of C_t to pairs of true hypotheses, and C_f to all other pairs. We assume that $0 \leq C_f < C_t$. The adjacency matrix (with inliers indexed first) has the following structure:

$$A = \left[\begin{array}{c|c} C_t & C_f \\ \hline C_f & C_f \end{array} \right] \quad (3.8)$$

By inspection, we see that an adjacency matrix of the form described above is rank 2, and has two non-zero eigenvalues. One non-trivial eigenvector corresponds to the subset of true hypotheses, while the other corresponds to the set consisting of the entire graph. The two non-zero eigenvalues are easily shown to be:

$$\lambda(v_t) = \frac{C_t N_t^2}{N_t} \quad (3.9)$$

$$\lambda(v_t + u_f) = \frac{C_t N_t^2 + C_f N_f^2 + 2C_f N_t N_f}{N_t + N_f} \quad (3.10)$$

We want SCGP to identify the set of true hypotheses, rather than the whole graph. This will happen so long as $\lambda(v_t) > \lambda(v_t + u_f)$. With some algebra, we see that this

constraint is satisfied when:

$$\frac{C_t}{C_f} > \frac{N_f}{N_t} + 2 \quad (3.11)$$

In other words, the ratio C_t/C_f determines how many outliers can be tolerated. Thus, a good consistency metric should try to make C_f as close to zero as possible, in order to maximize C_t/C_f . In the case of loop closure hypothesis filtering, C_f decreases exponentially as the rigid-body transformation of the loop deviates from the identity matrix. This helps ensure that the ratio C_t/C_f is large, allowing our algorithm to operate correctly even when the number of outliers greatly exceeds the number of inliers.

A different consideration for the pairwise consistency test is the “weight” of a hypothesis with itself. If we change the self-consistency of hypotheses, it is equivalent to adding a multiple of the identity matrix to the adjacency matrix.

$$\begin{aligned} (A + \alpha I)u &= \lambda(v)u \\ Au &= (\lambda(v) + \alpha)u \end{aligned} \quad (3.12)$$

We see immediately that the eigenvectors (and thus the SCGP solution) are unaffected. However, α shifts the *eigenvalues*, which can affect the convergence speed of Power Method-style algorithms for estimating the eigenvectors. Such a shift will also affect the confidence value λ_1/λ_2 . Both the confidence threshold and the self-consistency value are under the control of the user, however, so they can be chosen in a compatible manner.

3.5 Results

We evaluated our loop-closing method on both vision-based and laser-based datasets. Our vision data set is a 1.9km trajectory comprised of three complete loops in an outdoor urban environment. Our lidar-based data sets include standard map building benchmark datasets, as well as several new real and synthetic datasets.

Loop closure algorithms are difficult to evaluate quantitatively: there are no standardized datasets for this purpose. The benchmark datasets that we have processed are composed of raw sensor data, but because sensor processing methods vary between researchers, the resulting sets of hypotheses are not the same. Naturally, the quality and reliability of the sensor processing (and hypothesis-generating) systems has huge impact on the difficulty of the loop-closing problem.

We address this problem in two ways. First, we evaluate the performance of our system repeatedly on the same dataset, with the quality of hypotheses purposefully degraded to varying degrees. This allows us to determine the point (if any) at which our algorithm can no longer produce useful output. Second, our hypothesis datasets have been made available so that other researchers can perform a direct comparison to our method.

3.5.1 “Area C” Dataset

During the DARPA Urban Challenge, “Area C” was a testing area in which the ability of autonomous cars to queue at intersections (waiting for other cars) was tested. Team MIT’s vehicle logged a great deal of sensor data, including vision data, during its test. This logged data now serves as a vision-based mapping test¹

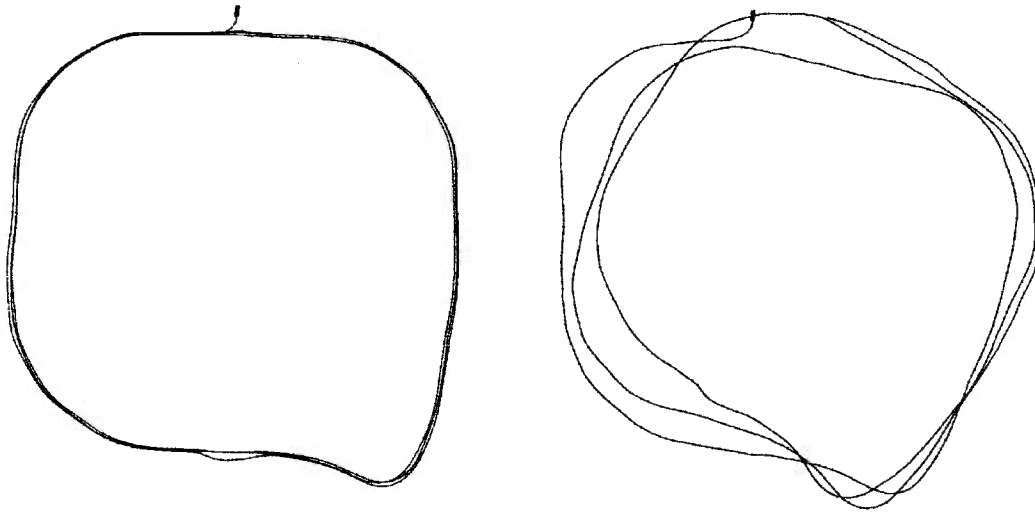


Figure 3-6: Area C dead-reckoning performance. Left: the dead-reckoning estimate, output by the high-end IMU, has an SSE of 1.02 versus the GPS-augmented ground truth. To simulate a more typical vehicle, additional noise was added (right): the initial configuration has an SSE of 487.

As described in Appendix A, we used SIFT features to initialize local landmarks. These local landmarks were then associated with each other in order to compute a rigid-body transformation relating two poses. This rigid-body transformation served as a hypothesis.

SIFT descriptors are 128-dimensional vectors. At their full size, they provide a fairly robust means of matching landmarks. However, hundreds of SIFT features can be detected in a single camera frame, making the storage of these descriptors problematic. Data structures to perform matching on these features also becomes more complicated. Other authors [32, 64] have described ways of reducing the dimensionality of SIFT descriptors, but the risk is that the discriminative power of the descriptor might be reduced. When designing new descriptors, it is important to know how much discriminative capability the descriptors must have in order to be able to close loops.

We answer this question by reducing the dimensionality of SIFT vectors in a simple way; this causes an increase in the hypothesis error rate. We then measure the ability of our approach to identify correct loop closures and reject incorrect hypotheses.

We reduce the dimensionality of SIFT vectors by dividing the descriptor into equally sized blocks and summing the elements in each block to form a shorter descriptor. While simplistic, our purpose is not to explore how to optimally reduce the

¹The method described here was not used by MIT during the DARPA Urban Challenge itself; the data here has been post-processed.

dimensionality of SIFT descriptors, but to explore what happens when descriptors become ambiguous. Our technique is a simple and easily repeatable means of doing this.

In Fig. 3-7, two sets of SIFT features observed at different times (but near the same place) are shown. The two sets of features are matched based on descriptor's nearest-neighbors; these correspondences are shown as lines. The RANSAC method described in Appendix A is used to identify a set of “good” correspondences from which a hypothesis is generated. As the descriptor is decimated, the matching performance drops as expected, producing increasingly poor hypotheses.

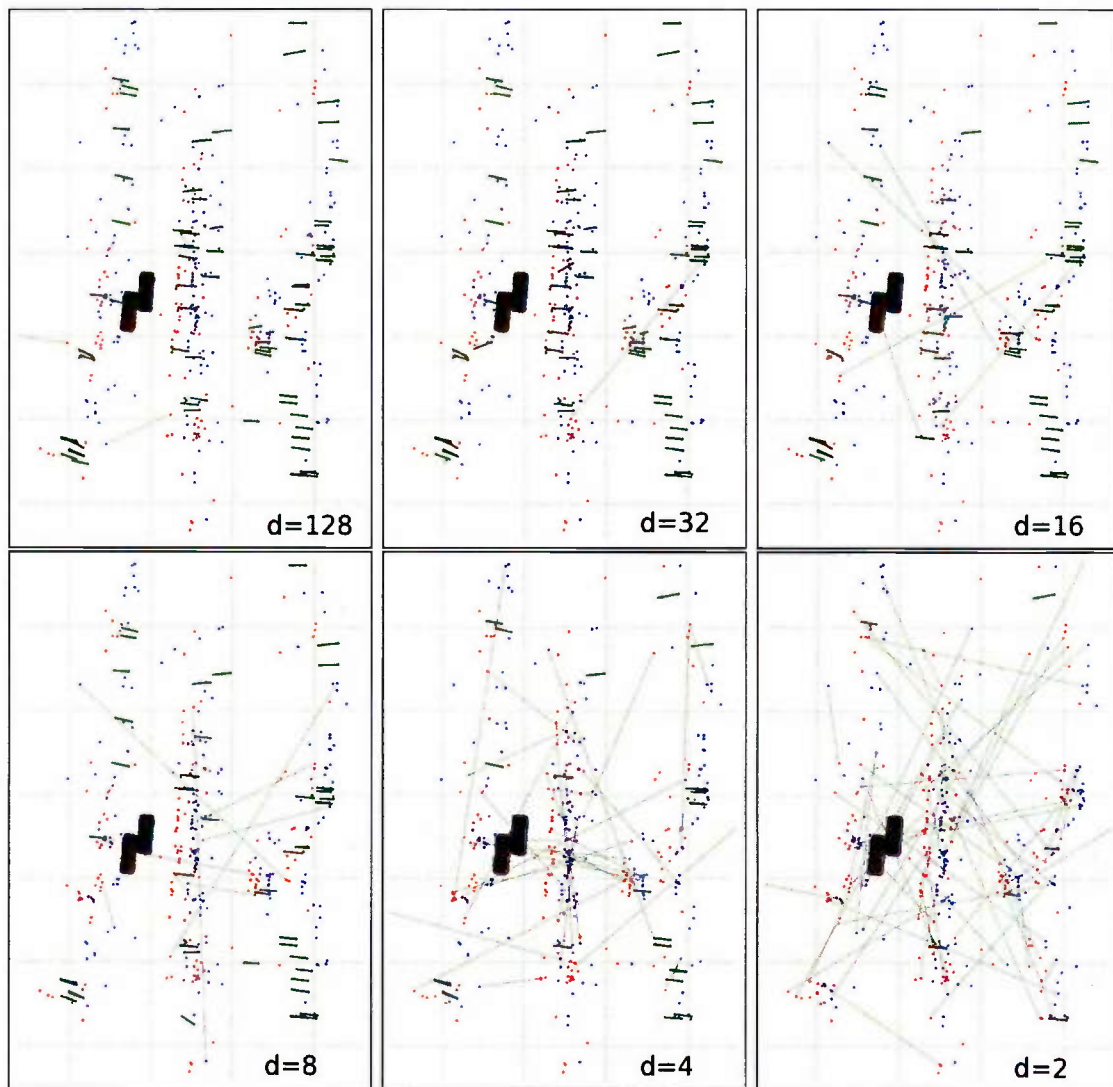


Figure 3-7: Matching decimated SIFT features. The SIFT features from two different visits to the same place are shown: dots correspond to features detected from the correspondingly-colored vehicle. Thick green lines indicate RANSAC-accepted correspondences, thin gray lines are rejected matches. As the size of the SIFT descriptor is reduced in size, the error rate increases. Our proposed loop closing method can effectively handle the increased error rate.

Significantly, our algorithm is able to identify correct hypotheses even when generated hypotheses are of very low quality (due to severe SIFT feature decimation).

In fact, our algorithm is still able to identify correct hypotheses even when those hypotheses are the result of scalar-valued SIFT descriptors.

As expected, our algorithm rejects a large fraction of the hypotheses when the SIFT descriptor is small (see Fig. 3-8); the acceptance rate rapidly increases with increasing SIFT descriptor size. Most importantly, the posterior maps that result from the filtered hypotheses are of high-quality (see Fig. 3-9).

The posterior map resulting from our vision-based approach, based on degraded odometry, was *better* (SSE=0.87) than the map resulting from the map derived from the expensive IMU alone (SSE=1.02). If we combine the high-quality IMU data with our vision method, the SSE drops to 0.07— even if using SIFT descriptors of size 3.

SIFT feature extraction and tracking were implemented using relatively slow methods and ran at about 10% of real time. Other authors have described ways of accelerating these steps [63]. Loop closing and the algorithms described in this chapter, in contrast, took between 7-9s (depending on the size of the SIFT feature descriptors)— about 62 times faster than real-time.

3.5.2 CSAIL Dataset

We collected the CSAIL dataset in the Computer Science and Artificial Intelligence Laboratory's 7th floor (see Fig. 3-10). Its relatively short length coupled with a significant amount of clutter and fairly long loops make it an interesting test case. Like all of the datasets that follow, its primary sensor is a 180-degree field-of-view laser scanner. The robot trajectory was estimated via incremental scan matching, and loop closure hypotheses generated using a combination of local feature matching and scan matching refinement. The particulars of these methods are described in greater detail in Appendix A.

We will use this dataset to examine an actual hypothesis set in detail. The hypothesis set occurred in the area shown by Fig. 3-11. This hypothesis set contained 46 hypotheses, 34 of which were accepted as correct. Six representative samples of both inliers and outliers are shown in Fig. 3-12. The outliers (on top) are subtly misaligned; in contrast, the inliers (bottom) are well-aligned.

The pair-wise consistency matrix of those 46 hypotheses is shown as a graph in Fig. 3-13. In this figure, each node represents a hypothesis (e.g., one of the panels from Fig. 3-12): the *length* of a line represents the consistency between two hypotheses, with shorter lines indicating greater consistency. The cluster of inliers is readily discernible on inspection, and indeed, SCGP identifies it automatically.

A total of 5.3s of CPU time was required to automatically generate loop-closure hypotheses and filter them using the mechanisms in this chapter. The graph had 201 poses, and a total of 1057 hypotheses were generated. Of these, 710 were accepted. (184 were rejected because they were members of hypothesis sets that had fewer than four hypotheses; 63 were rejected because they were members of a hypothesis set whose first two eigenvalues were too similar; and 100 were explicitly classified as outliers.)

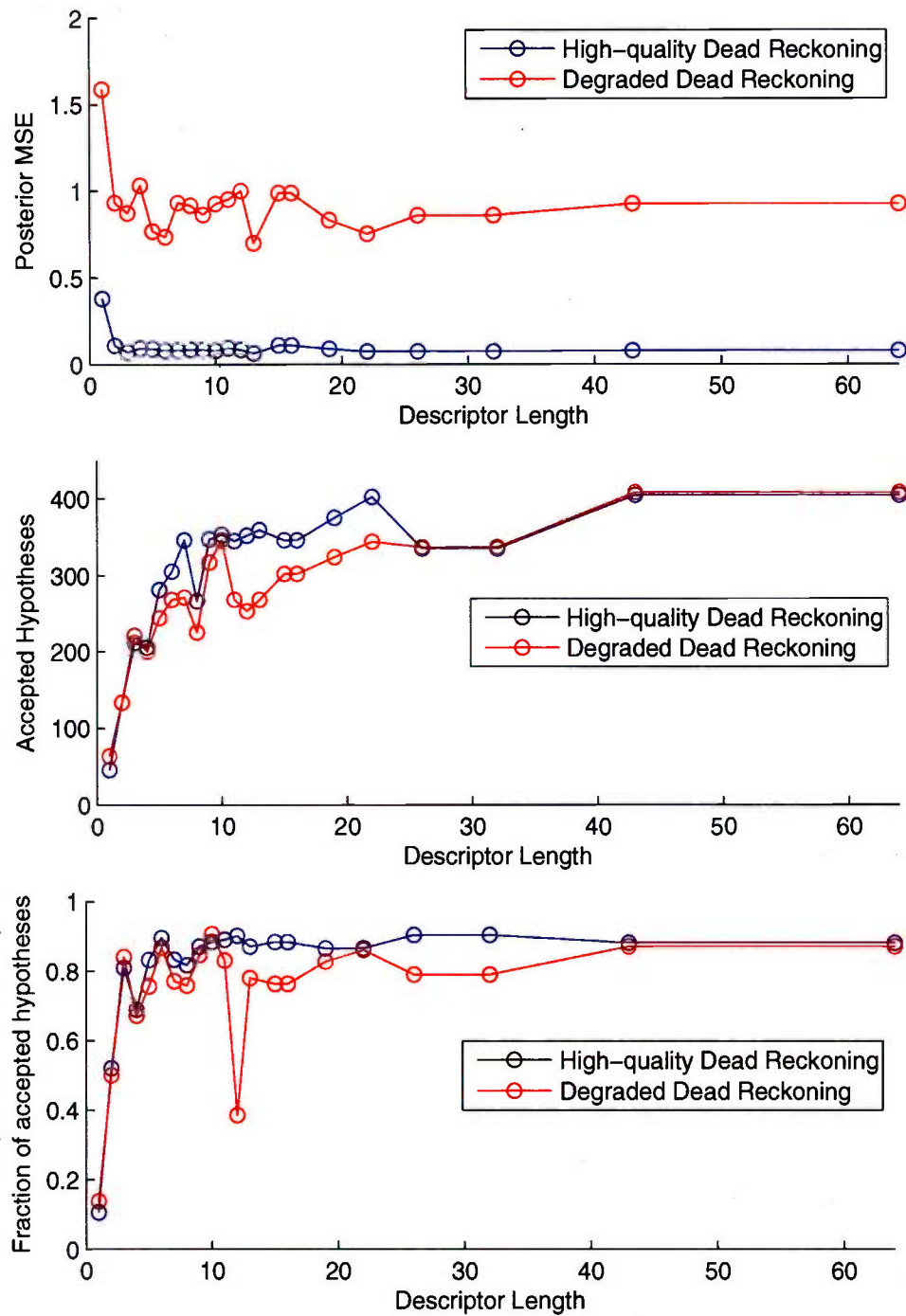


Figure 3-8: Area C Robustness. Shorter SIFT descriptors present greater perceptual ambiguity, and are more likely to produce incorrect matches. However, our method produces good maps in each case, and most surprisingly, can do almost as well with 3-dimensional vectors as with 128-dimensional vectors. The downward blip at $x = 12$ (bottom) was due to an unusually large number of ambiguous hypothesis sets, in which the ratio of the first to second eigenvalues was less than the threshold of 2.0.

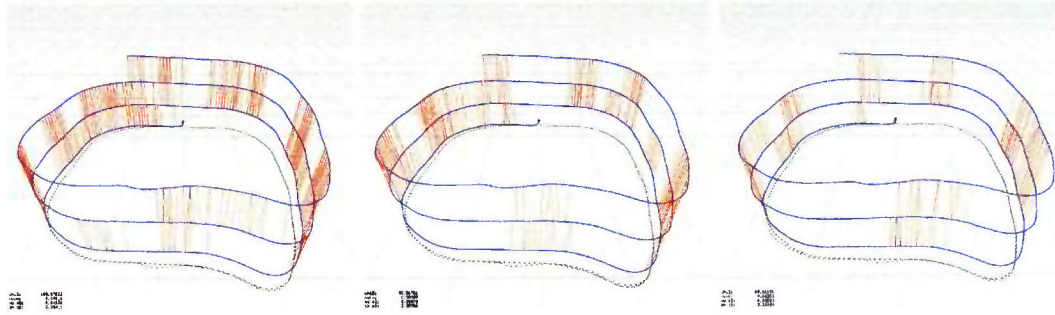


Figure 3-9: Area C Pose Graphs versus Descriptor Length. The pose/feature graph for the Area C dataset using 128-dimensional descriptors, 8-dimensional descriptors, and 1-dimensional descriptors. As the size of the descriptor goes down, ambiguity increases, resulting in fewer loop closures (red lines).

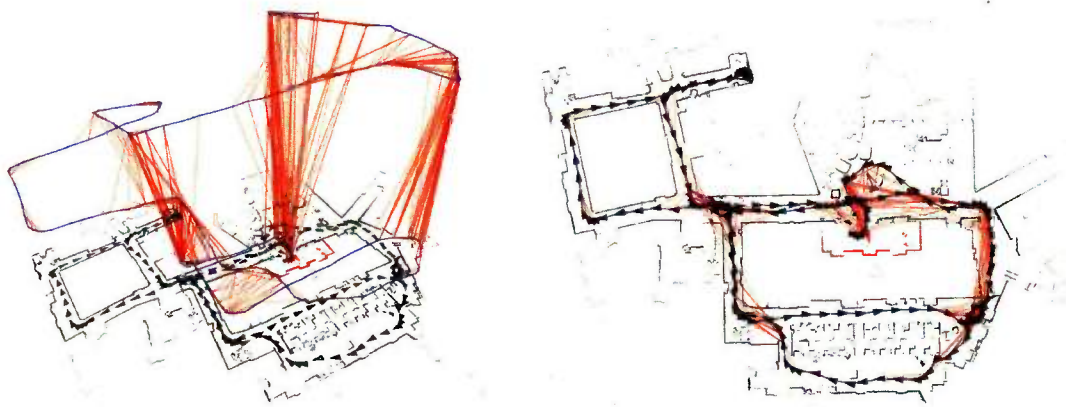


Figure 3-10: CSAIL Dataset Overview. Left: the pose graph is shown, with the trajectory in blue and loop closures in red. Right: the same pose graph, shown from above.

3.5.3 Killian Court Dataset

The Killian Court dataset, first collected and processed by Bosse [5], is challenging due to several long loops (see Fig. 3-15). Significant amounts of error can accumulate while the robot is traversing a long loop, which makes closing the loop harder. Another interesting characteristic of the Killian dataset is that it was recorded with a laser scanner with relatively high noise. This sensor noise makes it more difficult to recognize areas based on fine details. As a result, the hypothesis generator produced a greater number of incorrect hypotheses. The dataset also suffers from particularly poor wheel-derived odometry, as seen in Fig. 3-15.

As a partial compensation for this greater noise, loop closing was performed on larger sets of laser scans; i.e., instead of matching a single pose to another single pose, a small group of poses was matched at a time. This provided greater context for the hypothesis generator and reduced the rate of false positives.

In total, the Killian dataset consists of 1462 poses. These generated 6091 hypotheses, of which 4142 were accepted. 1187 were rejected because they were members of small hypothesis sets, 342 were rejected due to ambiguous eigenvalues, and 420 were

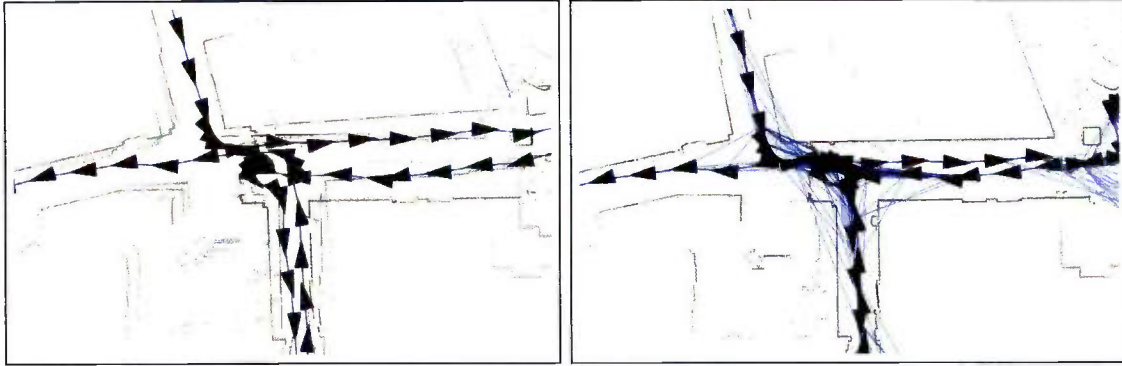


Figure 3-11: CSAIL loop closure close-up. Before loop-closing (left), laser scans show significant discrepancies. After recognition (right), not only are the discrepancies resolved, but the topological relationships are discovered.

classified as outliers. A total of 82s of CPU time was required to generate and filter the hypotheses. The dataset corresponds to 2 hours and 9 minutes of robot driving.

3.5.4 Stanford Gates Building

Like many of the other datasets, the Stanford Gates dataset (a standard benchmark) was recorded with a single 180-degree field-of-view lidar. The dataset is reasonably large, and contains several often-visited areas. These result in a large number of loop closures.

The Stanford dataset consists of 679 poses, which generated 7549 hypotheses. Of these, 6398 were accepted, with 470 rejected because their hypothesis set was too small, 253 rejected due to poor eigenvalue confidence, and 428 classified as outliers. Processing time was 64s; the dataset represents 34.5 minutes of data.

3.5.5 Intel Research Center

The Intel Research Center is a common benchmark dataset characterized by a large number of loop closures. The robot travels several times around the entire lab space, and visits each room. Of the datasets we processed, this dataset has the most loop closures.

The Intel dataset consists of 875 poses, spanning 44.9 minutes of robot driving. A whopping 15611 loop closure hypotheses were generated, and 12900 were accepted. Of those rejected, 758 were members of small sets, 715 were members of sets that failed the confidence test, and 1238 were classified as outliers. Total processing time was 180 s.

3.6 Summary

Recognizing places is a critical capability for many robot systems. Loop closing is a form of place recognition that is central to the task of map building: it prevents the

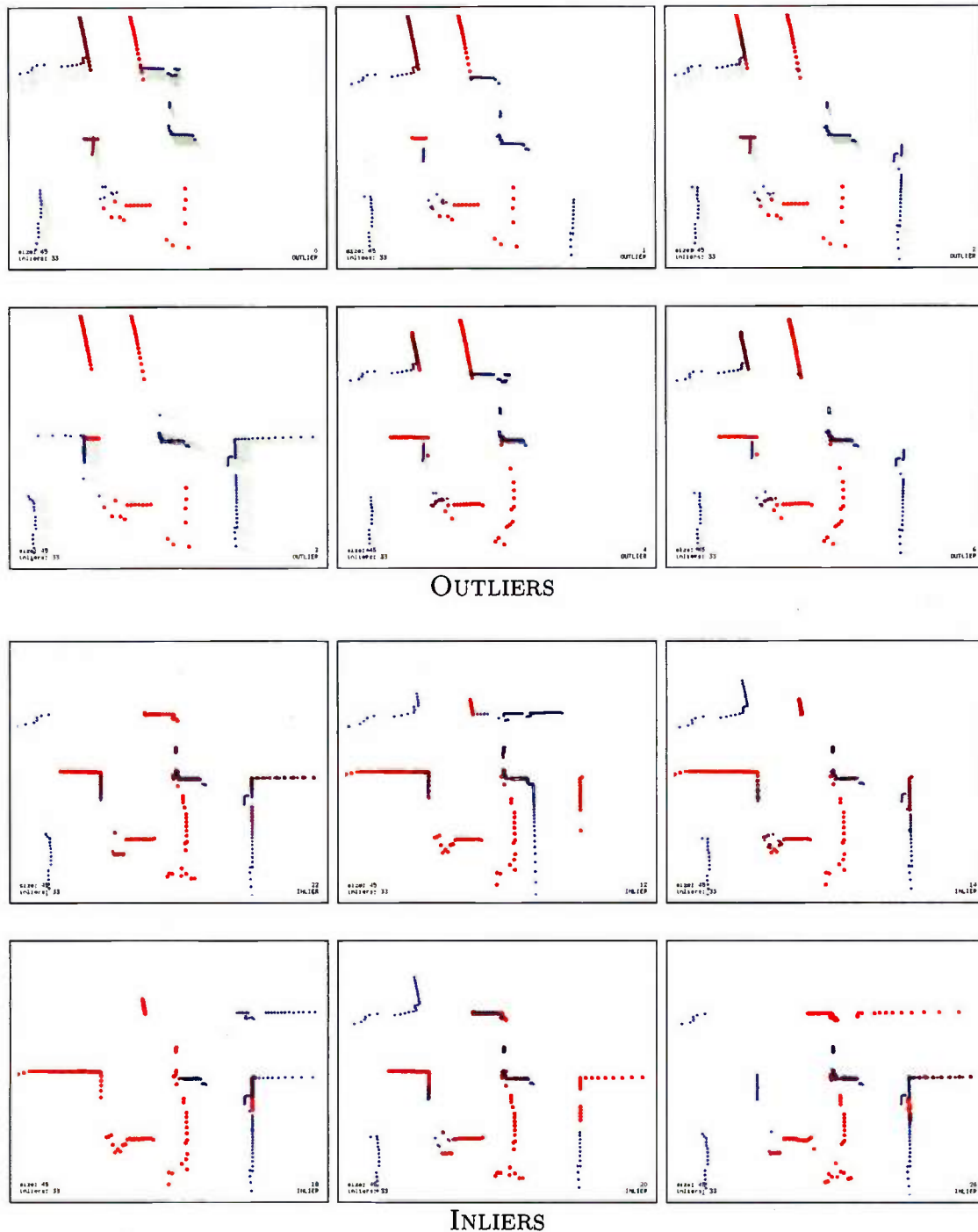


Figure 3-12: Filtered loop-closure hypotheses. Twelve representative hypotheses from a hypothesis set of size 45 are displayed. The outliers (top) and inliers (bottom) were automatically labelled using Single-Cluster Graph Partitioning. The alignment errors (derived from the posterior) are shown as gray lines: these errors are apparent in the outlier set.

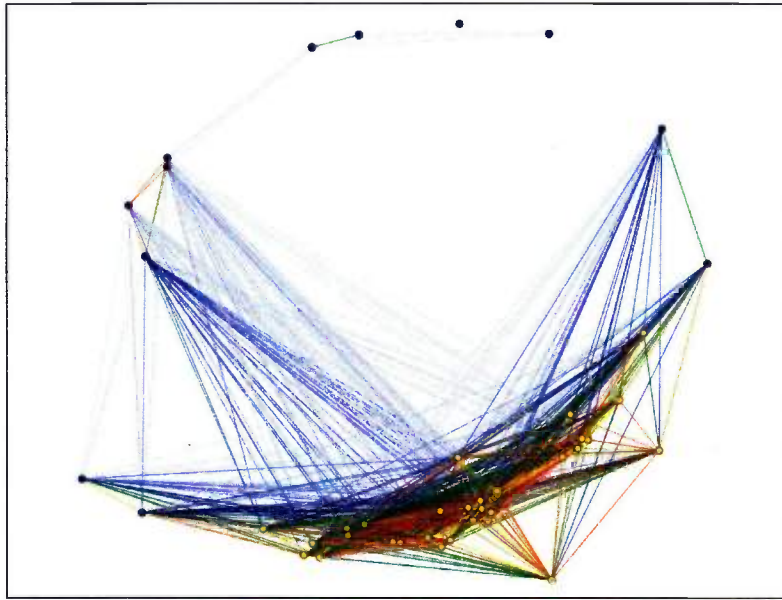


Figure 3-13: Adjacency graph for a set of 45 hypotheses. Each node represents a pose-to-pose hypothesis: the inlier set is indicated by yellow nodes (towards the bottom), with outliers in blue. Brightly-colored edges indicate greater pair-wise compatibility. The distance between two nodes is roughly proportional to the joint probability of the two hypotheses.

unbounded growth of dead-reckoning error. However, it is a difficult task due to both perceptual ambiguities and the large potential number of data associations.

In this chapter, we described an automatic loop closure system that can process a set of unreliable loop closure hypotheses and produce a set of correct hypotheses. It exploits the property that correct hypotheses generally agree with each other, whereas incorrect hypotheses tend to disagree with each other. The set of correct hypotheses is identified by examining the dominant eigenvectors of the pair-wise consistency matrix.

A critical capability of our method is the ability to determine the confidence of the solution. The second-best solution is not trivial variation on the best solution (as is the case in other algorithms), but is an orthogonal explanation of the data. The ratio of the eigenvalues of these two solutions serves as a useful confidence metric.

The method is also very fast, running much faster than real-time on all of the datasets we examined. It is also adaptable to a variety of sensing modalities: we demonstrated the system using local vision-based landmarks and laser-based scan matching.

The robustness of the system was demonstrated by decimating SIFT features. Originally 128-dimensional vectors, these were purposefully degraded to 1-dimensional scalars. Even though SIFT matching performance was severely affected, our system was able to identify correct loop closure hypotheses. The resulting loop closures allowed better posterior maps to be computed.

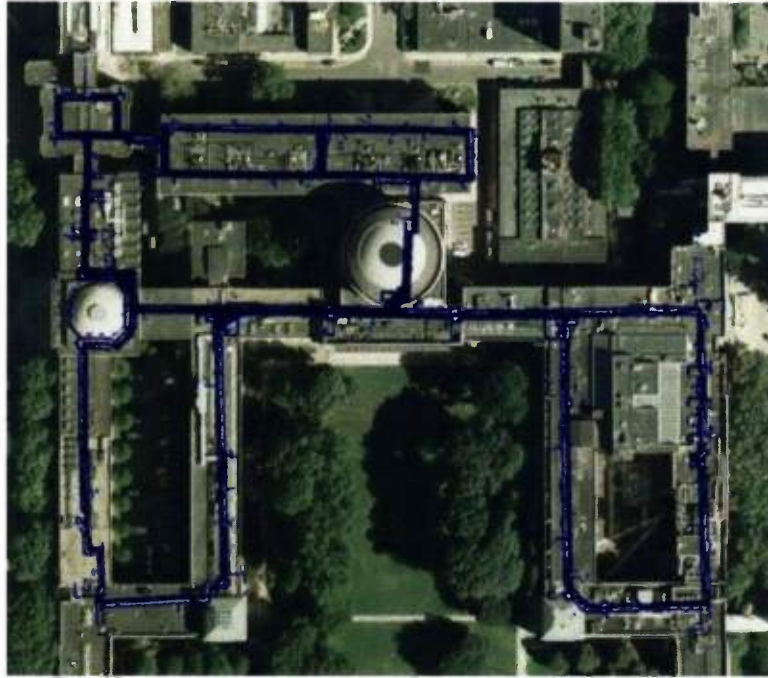


Figure 3-14: Killian Satellite Overlay. The long loops provide few opportunities for loop closing, leading to small distortions near the bottom of the figure. Macroscopically, however, the map aligns well with the satellite imagery.

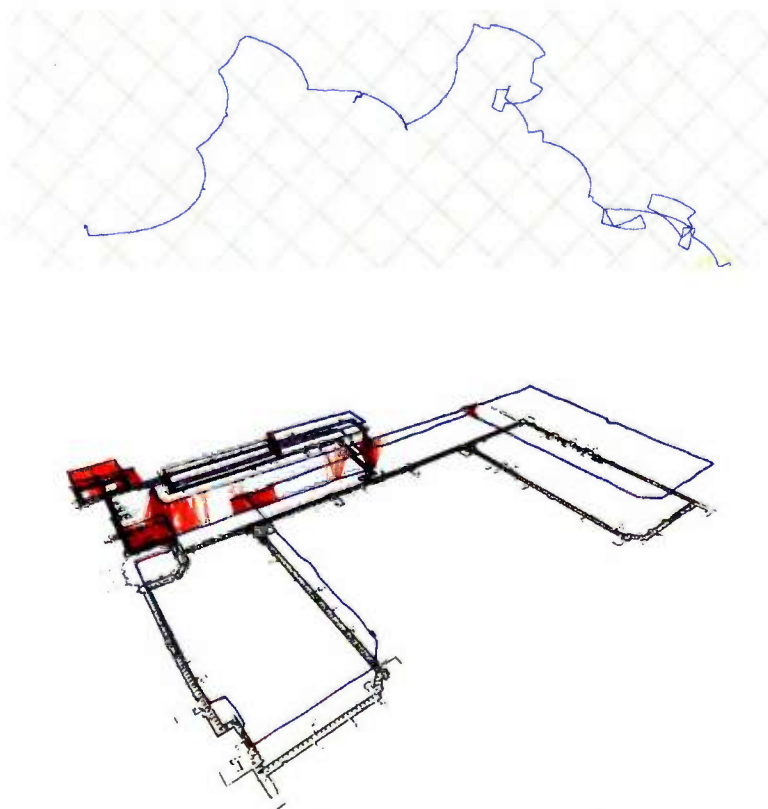


Figure 3-15: Killian Odometry and Pose Graph. Top: the raw wheel odometry (plotted on a 50 m grid). Bottom: The pose graph shows the relatively few areas in which loop closures are possible.

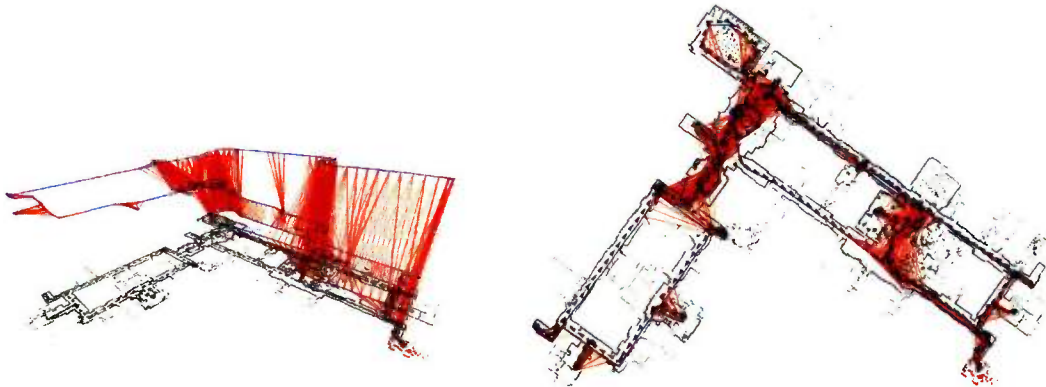


Figure 3-16: Stanford Gates Building. Left: the pose graph resulting from our algorithm. Right: the optimized map, viewed from above.

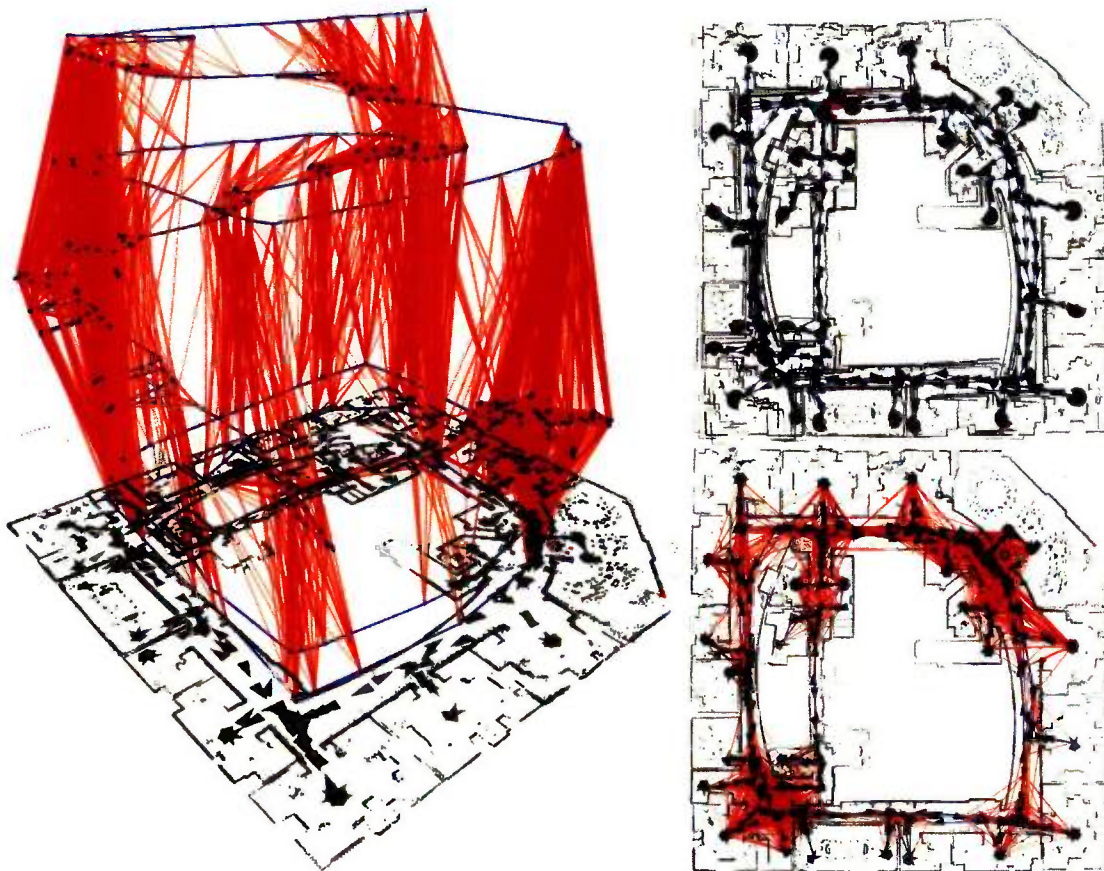


Figure 3-17: Intel research center. Left: the pose graph resulting from our algorithm. Top right: the uncorrected map. Bottom right: the optimized map, viewed from above.

CHAPTER 4

Batch Optimization of Pose Graphs

In this chapter, we describe our approach to optimizing pose graphs like those constructed in Chapter 3. The nodes of a pose graph represent particular places in the environment: these nodes can represent physical things (such as landmarks that have been sensed by the robot) or can represent the current or previous position of the robot. The edges of a pose graph relate the positions of two nodes and describe a cost as a function of the deviation from their preferred configuration. Pose graph optimization is the task of computing positions for each node such that the edges have minimum cost.

This optimization problem is critically important: the lowest-cost solution represents the best (maximum likelihood) map of the world. Whether a robot is avoiding obstacles, planning routes to some goal, or is explicitly trying to map an environment, the robot is dependent upon the fast and reliable optimization of the pose graph.

However, pose graph optimization is difficult for three central reasons:

- The state space is large. The size of the state vector is proportional to the number of nodes in the graph; a problem with thousands of state elements is *small*.
- The constraints are non-linear. As a result, the cost surface contains optimization hazards like local minima and valleys.
- The initial estimate is often poor. Most robots have meager dead-reckoning accuracy— as a result, the initial configuration of the pose graph can be far away from the best configuration.

This chapter describes a new optimization method that is both fast and robust. It can rapidly find the best configuration of nodes, and is robust to optimization hazards that trap existing methods. Our method also has low memory requirements and is relatively easy to implement.

Our method adapts Stochastic Gradient Descent (SGD) [56], commonly used in training artificial neural networks, to the problem of pose graph optimization. SGD has not previously been applied to map optimization, but has been used for localization [53].

We use a novel state-space representation, solving for the global-relative motions between poses, rather than solving for the absolute positions of the nodes. This representation allows faster reductions of error at each iteration, while a specialized data structure allows each iteration to be computed very quickly.

We demonstrate that our approach is not only very fast (competitive with or faster than other state-of-the-art methods), but is also significantly more robust: it finds correct solutions in cases that other methods get stuck in local minima.

4.1 Prior Work

4.1.1 Recursive Linear Solutions

If the constraints of a pose graph are approximated as linear and the noise is assumed to be Gaussian, pose graph optimization can be performed recursively using the Extended Kalman Filter (EKF) [65] or its dual, the Extended Information filter [58]¹. In these approaches, constraints are incorporated into the optimization one at a time. If the constraints are exactly linear, the resulting solution is optimal [43].

The Kalman filter maintains the covariance Σ and mean μ of the posterior distribution, while the Information filter maintains the information matrix Σ^{-1} and the information vector η . The information matrix and covariance are simply the inverses of each other, while the information vector is $\eta = \Sigma^{-1}\mu$.

These two representations provide an interesting trade-off in terms of the computational complexity of incorporating new information versus the computational complexity of extracting the posterior mean and covariance.

	Kalman Filter	Information Filter
Incorporate Constraint	$O(N^2)$	$O(1)$
Recover posterior mean/covariance	$O(1)$	$O(N^2)$

A large fraction of mapping applications, however, need to perform both operations. Even batch applications, in which the maximum likelihood solution is only needed at the very end, often need to recover the mean and covariance matrix at intermediate steps in order to perform data association. In other words, in order to compute the constraints from raw sensor data, the mean and covariance are often required. As such, in their naive forms, neither the Kalman filter nor Information filter is clearly superior.

However, the information matrix has a key property that has motivated a significant amount of research: sparsity. Whereas the covariance matrix rapidly becomes filled with significant non-zero values, the information matrix does not. The information matrix can be interpreted as the adjacency graph of a graphical model: its elements are non-zero only when variables are conditionally dependent (connected by an edge). If no variables have been marginalized out, the information matrix is

¹As opposed to the Kalman filter and the Information filter, the extended versions provide for non-linear constraints.

naturally sparse [13, 14]. This is because the observation range of a robot is limited, and thus a robot can observe only a small fraction of the environment from any given pose. A sparse information matrix can be stored in less memory than a covariance matrix (which is almost always dense), and the sparsity allows much faster recovery of the posterior mean and covariance.

In feature-based mapping methods, past robot positions are typically marginalized out. This marginalization creates new edges in the graphical model and makes the information matrix dense. However, the new edges tend to have small weights [18]. Thrun et al. described a method called Sparse Extended Information Filters (SEIF) in which these small weights are truncated. The result is a sparse approximation of the information matrix [69, 70, 71].

However, truncating elements of the information matrix can lead to over confidence [74]. An alternative method called Exactly Sparse Extended Information Filters (ESEIF) [75, 76] reimposes sparsity in a conservative manner by periodically relocalizing the robot. These relocalizations sever the conditional dependence on new robot positions from old positions. The resulting information matrix is truly sparse (hence “Exactly Sparse”), but the posterior does not exactly match that of the full information matrix: information about the previous position of the robot is discarded whenever the robot is relocalized.

Unfortunately, all recursive formulations suffer from linearization error. When a non-linear constraint is processed, it is linearized around the current state estimate—the non-linear aspects of the constraint are discarded. If the state estimate is of poor quality (i.e., far from the truth) when a constraint is linearized, the resulting linearization error is injected into the solution. Worse, the resulting uncertainty estimate is ignorant of this extra error. As a result, the uncertainty estimates are too low.

Some of the impact of this linearization error can be ameliorated by using the Unscented Kalman Filter (UKF). The Unscented Kalman Filter allows a more accurate projection of the covariance matrix after each constraint update by employing a small particle filter. Each particle is updated according to the non-linear form of the constraint; the covariance computed from these particles can be more accurate than the projection of the covariance based on only the first derivatives of the constraint.

The Iterated Kalman Filter [3] addresses linearization error in a different way. When a constraint is incorporated, that constraint might result in a large change in the state estimate. Since the constraint presumably brings the state estimate closer to the truth than it was before, it would be preferable to linearize around the new state estimate, rather than the previous state estimate. The Iterated Kalman filter allows this: the linearization point for each constraint is iteratively updated so that each constraint is linearized around a state estimate that reflects the contributions of that constraint. However, as new constraints continue to arrive and change the state estimate, older constraints are *not* relinearized.

4.1.2 Graphical Models

When a graphical model has a tree structure, the posterior distribution can be computed quite quickly [30]. Unfortunately, the graphical models corresponding to pose graphs are cyclic in virtually all interesting cases. Paskin’s Thin Junction Tree Filters (TJTF) [52] and Frese’s TreeMap [19] impose a tree structure on the graphical model by pruning edges. An advantage of this approach over SEIFs and ESEIFs (which similarly sacrifice edges in the graphical model for computational efficiency) is that constraints can be relinearized whenever inference is performed. Since edges are removed from the model, these methods compute an approximation of the correct posterior. These approximations can result in noticeable map artifacts.

4.1.3 Particle Filtering

Particle filters approximate the posterior distribution using a set of discrete samples. The best known method is Montemerlo’s FastSLAM [44]. FastSLAM particles are samples of just the robot trajectory: Montemerlo showed that the position of landmarks can be trivially recovered once the trajectory is known. (Landmarks are conditionally independent given the robot trajectory, since landmarks can only be observed from the robot trajectory).

The primary challenge inherent in particle filtering is accurately approximating the posterior: this can require large numbers of samples, especially when the uncertainty is large. Maintaining sufficient sample density and diversity has been an ongoing challenge [66, 22].

A unique advantage of particle filter methods is that data association is simplified: since each particle represents a trajectory, and the position of landmarks are independent given a trajectory, data association can be performed using a simple nearest-neighbor rule. Each particle can represent a different set of data association decisions.

4.1.4 Nonlinear Optimization

A growing family of iterative optimization methods can also be used to compute map posteriors. These approaches operate on the whole pose graph, and do not require any information to be discarded. Further, at each iteration, they can relinearize the constraints.

A brute-force nonlinear least squares implementation was suggested by Lu and Milios [41]. Their approach is similar in formulation to more modern approaches, but is generally too slow. Gradient Descent and Conjugate Gradient Descent [34, 72] have also been applied to map building.

Duckett *et al.* described an optimization method [12] using Gauss-Seidel relaxation. The basic approach is to repeatedly solve for the position of one node at a time, fixing the position of all the other nodes. They assumed absolute knowledge of the robot’s orientation, essentially making the problem linear. Fortunately, their approach can be extended in a straight-forward manner to account for unknown ro-

tation, and the resulting algorithm is quite good at fine-tuning a state estimate that is already close to the global minimum. Unfortunately, when the state estimate is far from the minimum, convergence can be impractically slow. Frese, Larsson, and Duckett attempted to improve convergence speed with the Multi-Level Relaxation (MLR) algorithm [20]. Folkesson’s Graphical SLAM method [16] is also based on Gauss-Seidel relaxation: it is accelerated by explicitly grouping nodes into “star nodes”, rather than automatically partitioning them as in MLR. Star nodes effectively reduce the size of the state space, accelerating convergence.

Konolige proposes a method [34] for accelerating convergence by reducing the graph to poses that have a loop constraint attached, solving for the other nodes separately. This can save considerable CPU time, but requires the graph to have low connectivity. This optimization is broadly applicable to most mapping algorithms.

Dellaert’s $\sqrt{SAM}$ method [10] performs a sparse Cholesky decomposition on the sparse information matrix. The factorization is greatly accelerated by selecting a good variable reordering (which is equivalent to picking an order to marginalize features in the corresponding graphical model). A key insight is that explicitly marginalizing out variables can only increase memory usage and decreases the effectiveness of the on-demand variable reordering algorithm. The method is generally one of the fastest methods available for full nonlinear optimization. Its performance, however, is very sensitive to the structure of the information matrix and the fill-in of the resulting Cholesky factors.

All non-linear optimization methods run the risk of getting stuck in local minima. Unfortunately, this risk is not hypothetical for map-building problems: non-linear approaches are susceptible to producing substantially different results depending on the initial conditions, as previously shown [24] and further demonstrated in this chapter.

The linearizations of recursive formulations like the EKF and EIF guarantee a quadratic and convex optimization surface with an easy-to-find minimum, but the resulting answer may not be a good solution to the actual problem. Nonlinear optimization methods generally offer better solutions, even if they do “get stuck”. This chapter describes non-linear optimization methods based on earlier work [48, 50].

4.1.5 Hybrids

Bosse’s Atlas [5] uses linearized (EKF-based) submaps but stitches them together using nonlinear optimization. During this stitching operation, the submaps are treated as rigid. The resulting “stitched” map generally has small errors where submaps do not align exactly. The decoupling of large maps into smaller maps, however, requires less computation.

4.1.6 State Space

Most map building algorithms use an obvious state space representation in which the state variables corresponds to the *positions* of the nodes. However, this is not the only possibility. Previous authors have used local submaps whose coordinate system is defined by features in that submap [36], while others have used state space

representations in which the *relative* position of nodes is used. This chapter will introduce a new representation that encodes the *motions* of the robot as it moves from pose to pose.

4.2 Overview and Intuition

Our approach is motivated by a simple intuition. Consider a pose graph constructed by a robot as it travels around a large loop. The trajectory of the robot is periodically sampled, with each sample point represented by a pose (a “node”) and connected to its predecessor by an estimate of the robot’s motion (an “edge”). Note that every edge has an uncertainty associated with it, so it is possible to compute its χ^2 error.

Suppose the robot returns to its starting place and recognizes it; this adds an additional edge between the first and last pose, creating a cycle (see Fig. 4-1). This edge is commonly called a “loop closure”, since it closes a large loop of nodes. Recognizing these loop closures was the subject of Chapter 3.

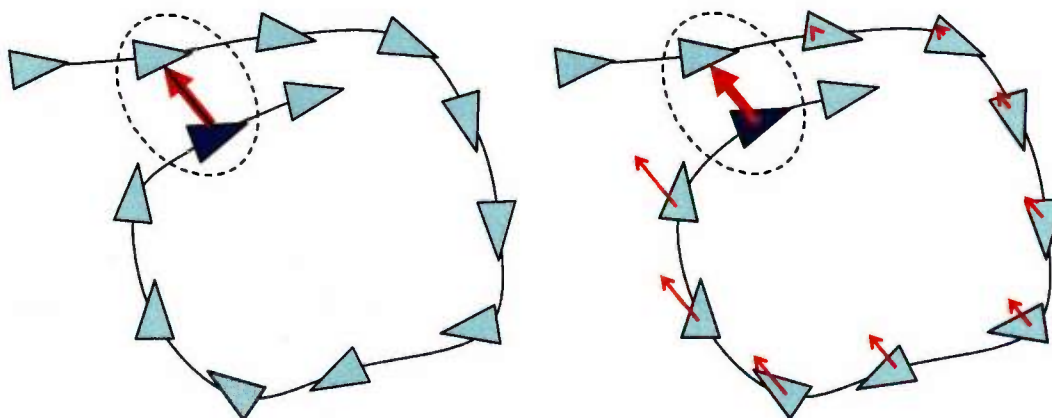


Figure 4-1: Map Optimization Intuition. The χ^2 error of a graph with a poorly-satisfied constraint (left, red arrow) can be reduced by distributing small adjustments around the nodes in the loop (right).

The χ^2 cost of any constraint grows as the square of its residual; a net reduction in cost can be obtained by reducing the error of the high-error constraint at the expense of adding a small amount of error to other constraints. We can do exactly this, adding small motions to each of the constraints so that the final loop closure is more closely satisfied. All of the constraints now have some error, but the χ^2 error has been reduced.

Konolige [34] pointed out that this simple problem can be solved “exactly” (up to linearization error) in $O(N)$ time by linearizing the constraints and recursively dissecting the loop and optimizing each half using a trivial EKF. (We note that this method is not immune to linearization error; in fact, this linearization error can cause divergence when large errors are present.)

In general, pose graphs do not consist of a single loop, but rather a complex web of interconnected constraints. Konolige’s method, which simultaneously optimizes N

constraints, cannot be extended to this case, because the recursive dissection relies on the nodes having only two neighbors.

However, each constraint can be viewed as part of a loop in which the constraint forms a loop relative to the current state estimate. The error of the constraint can be reduced by adjusting the position of elements of the state vector between the beginning and end of the loop, just as in Fig. 4-1.

Unfortunately, it is generally impossible to satisfy all the constraints at the same time. How do we know which constraints to satisfy at the expense of other constraints? Determining this is equivalent to performing a least-squares iteration, and is a slow operation. Instead, our method iteratively finds an equilibrium in which the conflicting updates of antagonistic constraints are in balance. This equilibrium corresponds to our desired solution.

4.3 Method

In this section, we derive our method. The first step is to introduce a novel state space representation that accurately encodes the cumulative nature of robot motion: the position of a robot at time t is the integral of its motions over times $[0, t]$. This state space representation is a key contribution, and is a critical factor in the algorithm’s performance.

Second, we derive an iterative update to the state vector using a single constraint. This update is based on stochastic gradient descent. It is simple, fast to compute, and highly effective at reducing both the χ^2 error and SSE of the graph. These traits make the algorithm competitive with (and often superior to) other state-of-the-art methods. However, our iterative update is particularly good at escaping local minima, giving our algorithm an ability to find good solutions when other algorithms get stuck.

Finally, we describe a data structure that reduces the computational cost of each update from $O(N)$ to $O(\log N)$, where N represents the number of poses in the graph.

In this work, we assume that all constraints between poses are full-rank rigid-body transformations. Rank-deficient constraints can, in principle, be treated as well; however, we do not examine this case. (We note that rank-deficient constraints, such as the bearing-only constraints arising from vision data, can often be combined to form full-rank constraints; this is precisely the technique we use on our vision based dataset.) We also limit ourselves to 2D, though our methods also apply in principle to 3D.

4.3.1 Incremental State Representation

The choice of state representation has a large impact on the performance of our optimization algorithm. Picking a state space in which the state variables echo the structure of the underlying random process results in significant performance improvements.

Consider a pose graph consisting of a single robot trajectory sampled at particular times. The conventional state space is the *global absolute* (x_{global}) state space— i.e.,

a list of (x, y, θ) positions for each node. However, the position of a robot at time t is the integral of all of its motions from time 0 through time t . The underlying random processes of the robot trajectory are the *motions* between each pose, not the positions.

Using the x_{global} state space tends to obscure this underlying structure. Suppose a and b are two poses from a robot trajectory and that their positions are constrained by a rigid-body transformation. In the x_{global} state space representation, the gradient of this constraint is zero everywhere except for nodes a and b . In other words, while the constraint gives information about the motion of the robot between a and b , this information is not reflected in the position of other nodes, whose positions are a function of this motion.

Virtually all contemporary algorithms incorporate the effects of *all* the constraints at each update step, which avoids this problem. By including all the constraints (most obviously the odometric constraints), their solutions reflect the underlying structure of the problem. This comes at a cost: it takes more time to incorporate the effects of all the constraints.

However, it is possible to use a state space that more directly reflects the structure of the problem. This allows good state updates *without* having to consider the effects of all the constraints at every iteration.

$$\begin{array}{ccc}
 \begin{bmatrix} x_0 \\ y_0 \\ \theta_0 \\ x_1 \\ y_1 \\ \theta_1 \\ x_2 \\ y_2 \\ \theta_2 \\ \dots \end{bmatrix} & \begin{bmatrix} x_0 \\ y_0 \\ \theta_0 \\ \cos(\theta_0)(x_1 - x_0) + \sin(\theta_0)(y_1 - y_0) \\ -\sin(\theta_0)(x_1 - x_0) + \cos(\theta_0)(y_1 - y_0) \\ \theta_1 - \theta_0 \\ \cos(\theta_1)(x_2 - x_1) + \sin(\theta_1)(y_2 - y_1) \\ -\sin(\theta_1)(x_2 - x_1) + \cos(\theta_1)(y_2 - y_1) \\ \theta_2 - \theta_1 \\ \dots \end{bmatrix} & \begin{bmatrix} x_0 \\ y_0 \\ \theta_0 \\ x_1 - x_0 \\ y_1 - y_0 \\ \theta_1 - \theta_0 \\ x_2 - x_1 \\ y_2 - y_1 \\ \theta_2 - \theta_1 \\ \dots \end{bmatrix} \\
 \mathbf{x}_{global} & \mathbf{x}_{rel} & \mathbf{x}_{incr}
 \end{array}$$

Figure 4-2: State Space Representations. The commonly used state space x_{global} , the rigid-body transformation state space x_{rel} , and our proposed “global incremental” state space x_{incr} .

One way of “capturing” the cumulative nature of robot motion is to represent the robot trajectory as a series of rigid body transformations. In this case, the position of the trajectory node i in the trajectory is the composition of all the rigid-body transformations $T_0 \oplus T_1 \oplus \dots \oplus T_{i-1}$, and the state space is a list of the rigid-body transformations. We call this the “relative” (x_{rel}) state space (see Fig. 4-2).

This state space, while accurately modeling the motion of the robot, is not stable. In a well-behaved optimization problem, a change to the state vector should have a commensurate effect on the cost function: small steps should lead to small changes in cost, for example. This is not the case when using a relative state space representation. A tiny change to a rotational component in one part of the graph can have a very

large projective effect on another (distant) part of the graph. This can cause enormous increases in the χ^2 error of other constraints. Because these projective effects are not well-modeled due to linearization error, a state change that is predicted to reduce the χ^2 error might actually increase it.

Instead, we propose a state space representation that approximates the cumulative motion of the robot like x_{rel} , but eliminates the projective effects of the relative representation. We call it the “global incremental” state space representation, x_{incr} : the state vector contains the difference between the successive global positions. In the absence of rotations ($\theta_i = 0$), it is identical to the relative state space. With this state space representation, there are no projective effects (the position of a node is a simple sum of global motions, not the composition of rigid-body transformations), and so updates are more stable.

4.3.2 Iterative optimization

Our approach is a hybrid of nonlinear least squares and stochastic gradient descent; we begin by reviewing the “vanilla” versions of these algorithms.

Nonlinear Least Squares

Recall from Chapter 2 that the χ^2 error for a graph, evaluated at a distance d from the current state estimate, can be written in terms of the Jacobians of the constraints J , the residuals r , and the block-diagonal information matrix Σ^{-1} of the constraints:

$$\chi^2 = (Jd - r)^T \Sigma^{-1} (Jd - r) \quad (4.1)$$

We can compute the d that minimizes the χ^2 error by differentiating with respect to d and setting the result to zero:

$$\begin{aligned} \frac{\partial \chi^2}{\partial d} &= 2J_i^T \Sigma_i^{-1} Jd - 2J_i \Sigma_i^{-1} r_i = 0 \\ (J^T \Sigma^{-1} J)d &= J^T \Sigma^{-1} r \end{aligned} \quad (4.2)$$

Note that Eqn. 4.2 is of the form $Ax = b$, with $A = J^T \Sigma^{-1} J$ (the information matrix) and $b = J^T \Sigma^{-1} r$. This equation can be solved for d via a number of methods such as inversion of A , LU decomposition, or Cholesky decomposition.

Algorithm 3 NonlinearLeastSquares

repeat

 compute J and r at the current state estimate.

$d = (J^T \Sigma^{-1} J)^{-1} J^T \Sigma^{-1} r$

$x = x + d$

until $\|d\| < \epsilon$

Nonlinear-least squares computes state updates by simultaneously considering the interaction of all the constraints. Often, only a few iterations are required for a very

good solution. Unfortunately, Alg. 3 can be impractically slow.

Stochastic Gradient Descent

In Stochastic Gradient Descent (SGD), a constraint is selected and a gradient descent step is taken to reduce the error of that constraint. However, a graph generally contains antagonistic constraints, whose steps tend to conflict. Stochastic Gradient Descent suggests a solution: introduce a learning rate λ that modulates the step size, and reduce λ over time. Provided that a few technical conditions (described below) are satisfied, the resulting method will find an equilibrium between antagonistic constraints. This equilibrium indicates that a minimum has been found.

Naturally, we expect that single iteration of stochastic gradient descent will not reduce the χ^2 error as much as an iteration of non-linear least squares. The promise of SGD is that each iteration is so easy to compute that the minimum will be found faster, despite requiring more iterations.

The learning rate must satisfy two conditions in order to guarantee convergence [6]; the first ensures that the algorithm can enough “energy” to explore the whole state space, while the second ensures that the gradients eventually vanish when the solution approaches the optimal value:

$$\sum \lambda_t = \infty \quad (4.3)$$

$$\sum \lambda_t^2 < \infty \quad (4.4)$$

We use Robbins’ original suggestion [56], of a harmonically-decreasing learning rate (i.e., $\lambda_t = 1/t$), which satisfies these properties. Other authors have explored more complex (and adaptive) learning rates [7], but our experiments with these methods did not consistently improve performance over a simple harmonic progression.

Consider the χ^2 error of a single constraint i , evaluated at the current state estimate plus d (see Chapter 2 for more details):

$$\chi_i^2 = (J_i d - r_i)^T \Sigma^{-1} (J_i d - r_i) \quad (4.5)$$

The gradient is:

$$\frac{\partial \chi_i^2}{\partial d} = 2J_i^T \Sigma_i^{-1} J_i d - 2J_i^T \Sigma_i^{-1} r_i \quad (4.6)$$

Evaluated at the current state estimate, where $d = 0$, gives just:

$$\frac{\partial \chi_i^2}{\partial d} = -2J_i^T \Sigma_i^{-1} r_i \quad (4.7)$$

At time step t , let the learning rate λ be $1/t$. The gradient descent step moves in the opposite direction of the gradient, and can be written as:

$$d = \lambda 2J_i^T \Sigma_i^{-1} r_i \quad (4.8)$$

Algorithm 4 Stochastic Gradient Descent Algorithm

```

 $\lambda = 1/3$ 
while not converged do
  select a constraint  $i$  at random
  compute  $J_i$  and  $r_i$  at the current state estimate.
   $d = 2\lambda J_i^T \Sigma_i^{-1} r_i$ 
   $x = x + d$ 
   $\lambda = \lambda / (\lambda + 1)$ 
end while

```

At each iteration, Stochastic Gradient Descent (see Alg. 4) reduces the error of a single constraint (often unintentionally increasing the error of other constraints). In early iterations, it will take large steps in order to reduce the residual, which causes the state estimate to jump around the state space. This behavior allows SGD to escape local minima. Intuitively, while many constraints may be satisfied at a local minimum, there are generally one or more constraints that remain poorly satisfied. These ill-satisfied constraints will cause a comparatively large jump away from that local minimum. Conversely, near the global minimum, constraints tend to be better satisfied: this means smaller jumps. In short, ill-satisfied constraints cause the state estimate to jump around, but the state estimate tends to “stick” near the global minimum once it is found.

The ability to rapidly explore the state space not only allows SGD to escape local minima, but is conducive to rapid reductions in the χ^2 error. This is a valuable feature to the application of map optimization, where the initial state estimate can be very poor. The disadvantage, however, is that this exploration never ends. Even if the global minimum has been found, SGD continues to jump around (though the jumps shrink over time due to the decreasing learning rate). In other words, while convergence is very fast at the beginning, it is ill-suited towards “fine tuning” the solution. As we will illustrate, while the effect is real, the practical consequences are often negligible.

SGD is sensitive to the scaling of the Σ_i^{-1} matrices: this scale proportionately affects the size of the step d . Two optimization problems that differ only by the scaling of their Σ_i^{-1} matrices are equivalent, yet Eqn. 4.7 will yield different steps for them. We can eliminate this sensitivity by normalizing the set of Σ_i^{-1} matrices according to the largest example.

Our Hybrid Method

Our method combines features of nonlinear least squares and stochastic gradient descent. We also incorporate additional domain-specific knowledge to prevent the state updates from making harmfully large steps.

First, note that the nonlinear least squares step (Eqn. 4.2) can be written in terms

of contributions from each individual constraint:

$$d = (J^T \Sigma^{-1} J)^{-1} J^T \Sigma^{-1} r \quad (4.9)$$

$$= \sum_i d_i \quad (4.10)$$

$$= \sum_i (J^T \Sigma^{-1} J)^{-1} J_i^T \Sigma_i^{-1} r_i \quad (4.11)$$

The d_i 's computed by nonlinear least squares and stochastic gradient descent are quite similar. The difference is essentially scaling: nonlinear least squares rescales and reshapes the step according to the inverse of the information matrix. Intuitively, this scaling accomplishes two things: it determines the relative importance of each constraint, and it determines how error should be distributed around a loop. (Since some nodes are more uncertain than others, a larger decrease of χ^2 error can be obtained by distributing more error to less-confident nodes.)

Nonlinear Least Squares	Stochastic Gradient Descent
$d_i = (J^T \Sigma^{-1} J)^{-1} J_i^T \Sigma_i^{-1} r_i$	$d_i = 2\lambda J_i^T \Sigma_i^{-1} r_i$

Our hybrid solution approximates this scaling by finding an $M^{-1} \approx (J^T \Sigma^{-1} J)^{-1}$. We use a relatively low-fidelity approximation, computing M as the block-diagonal of $J^T \Sigma^{-1} J$. This approximation is easy to compute and does a reasonably good job of adjusting how error is distributed around a loop (i.e., it estimates the relative confidence of each node), but it is nearly useless as a means of estimating the absolute importance of each constraint. In other words, the *shape* of M is useful, but the *scale* is not. To negate the effect of M 's scale, we constrain d_i so that it has the same magnitude as $J_i^T \Sigma_i^{-1} r_i$. In summary, we can compare the three update formulas:

Nonlinear Least Squares	Stochastic Gradient Descent	Hybrid
$d_i = (J^T \Sigma^{-1} J)^{-1} J_i^T \Sigma_i^{-1} r_i$	$d_i = 2\lambda J_i^T \Sigma_i^{-1} r_i$	$d_i = \lambda M^{-1} J_i^T \Sigma_i^{-1} r_i$

As mentioned earlier, the scale of Σ_i^{-1} has an impact on the step size. This effect is removed by scaling the steps by largest Σ_i^{-1} in the graph.

Our algorithm, like stochastic gradient descent, takes steps whose size are proportional to the errors of the constraints. In map optimization, however, we can often compute a *maximum* permissible step size. Stepping farther than this maximum would only start increasing the error again, and thus should be avoided. Consequently, we clamp d such that it will not overshoot the constraint's minimum. This improves both the performance and stability of the algorithm.

4.3.3 Rigid-Body Constraints

We now illustrate how to apply the optimization method to a rigid-body constraint. Suppose we have a rigid-body constraint T connecting poses a and b . In x_{global}

coordinates, the observation equations for the rigid-body constraint are:

$$\begin{aligned}x_T &= \cos(\theta_A)(x_B - x_A) + \sin(\theta_A)(y_B - y_A) \\y_T &= -\sin(\theta_A)(x_B - x_A) + \cos(\theta_A)(y_B - y_A) \\\theta_T &= \theta_B - \theta_A\end{aligned}\tag{4.12}$$

Let us write the incremental motions between consecutive x coordinates as $\dot{x}$, such that $\dot{x}_i = x_i - x_{i-1}$. We do the same with $\dot{y}$ and $\dot{\theta}$ such that we can write $x_{incr} = [\dot{x}_1 \dot{y}_1 \dot{\theta}_1 \dot{x}_2 \dot{y}_2 \dot{\theta}_2 \dots]^T$. We can now write Eqns. 4.12 in terms of the x_{incr} state variables.

$$\begin{aligned}x_T &= \cos\left(\sum_0^a \dot{\theta}_i\right) \sum_{a+1}^b \dot{x}_i + \sin\left(\sum_0^a \dot{\theta}_i\right) \sum_{a+1}^b \dot{y}_i \\y_T &= -\sin\left(\sum_0^a \dot{\theta}_i\right) \sum_{a+1}^b \dot{x}_i + \cos\left(\sum_0^a \dot{\theta}_i\right) \sum_{a+1}^b \dot{y}_i \\\theta_T &= \sum_{a+1}^b \dot{\theta}_i\end{aligned}\tag{4.13}$$

The Jacobian of the rigid-body transformation T with respect to the state variables x_{incr} is:

$$\frac{\partial[x_T \ y_T \ \theta_T]}{\partial[\dot{x}_i \ \dot{y}_i \ \dot{\theta}_i]} = \begin{cases} \begin{bmatrix} 0 & 0 & -(x_b - x_a) \sin(\theta_A) + (y_b - y_a) \cos(\theta_A) \\ 0 & 0 & -(x_b - x_a) \cos(\theta_A) - (y_b - y_a) \sin(\theta_A) \\ 0 & 0 & 0 \end{bmatrix} & 0 \leq i \leq a \\ \begin{bmatrix} \cos(\theta_a) & \sin(\theta_a) & 0 \\ -\sin(\theta_a) & \cos(\theta_a) & 0 \\ 0 & 0 & 1 \end{bmatrix} & a < i \leq b \\ \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} & \text{otherwise} \end{cases}\tag{4.14}$$

Given some thought, the block of Jacobians corresponding to $0 \leq i \leq a$ is somewhat odd. Intuitively, this Jacobian suggests that a translational error in the relative positions of a and b can be reduced by rotating the nodes $[0, a - 1]$. This is an artifact of our state representation, which (purposefully) does not model the projective effect of robot motion. If we consider the projective nature of the robot trajectory, we know that rotating a node $0 \leq i < a$ does not affect the relative positions of nodes a and b : such a rotation will have the same projective effect on both nodes, leaving their relative positions unchanged. If we instead computed the Jacobian of the constraint between a and b in the x_{rel} state space (a more accurate, if less stable representation), we would find that the Jacobian for nodes $[0, a - 1]$ is exactly zero. In other words, over the interval $0 \leq i \leq a$, the Jacobian is usually better approximated by zero than by Eqn. 4.14.

In summary, the Jacobian that we use is:

$$\frac{\partial[x_T \ y_T \ \theta_T]}{\partial[\dot{x}_i \ \dot{y}_i \ \dot{\theta}_i]} \approx \begin{cases} \begin{bmatrix} \cos(\theta_a) & \sin(\theta_a) & 0 \\ -\sin(\theta_a) & \cos(\theta_a) & 0 \\ 0 & 0 & 1 \end{bmatrix} & a < i \leq b \\ \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} & \text{otherwise} \end{cases} \quad (4.15)$$

In order to compute a step, we also need to compute the block-diagonal approximation M of the information matrix $J^T \Sigma^{-1} J$. Since each constraint is independent, Σ^{-1} is a block-diagonal matrix composed of the individual Σ_i^{-1} matrices. Thus, we can build up our approximation for M by considering each constraint one at a time:

$$J^T \Sigma^{-1} J = \sum_i J_i^T \Sigma_i^{-1} J_i \quad (4.16)$$

Let R_i be the repeated block of the Jacobian J_i for $a \leq i \leq b$, i.e.:

$$R_i = \begin{bmatrix} \cos(\theta_a) & \sin(\theta_a) & 0 \\ -\sin(\theta_a) & \cos(\theta_a) & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad (4.17)$$

From the repeated-block structure of J_i it is easy to see that the block diagonals will be $R_i^T \Sigma_i^{-1} R_i$ for blocks $a < i \leq b$, and zero elsewhere. M can be computed by adding up these blocks for each constraint, discarding all but the values on the diagonal.

4.3.4 The Error-Distributing Tree

In the previous sections, we showed how the state update d can be computed for a constraint. This computation requires the Jacobian of the constraint, which we derived for the ubiquitous case of a rigid-body constraint. A simple implementation could compute d literally and add it to the state vector x . For a graph with N nodes, this would result in a runtime cost of $O(N)$ per constraint.

However, the Jacobian for a rigid-body constraint has an interesting structure: it is *constant* over the interval $a < i \leq b$, and zero elsewhere. This means that the error is distributed uniformly over the nodes in the loop, subject to the reshaping effect of M . This section describes a specialized data structure for the state that allows updates (including the effects of M) to be performed in $O(\log N)$ time.

D-Trees

This data structure makes extensive use of a simpler novel data structure we call a *distribution tree*, or *dtree*. A dtree conceptually maintains an array of values v_i ($0 \leq i < N$) and supports two operations: compute the value at a particular index

i , and increase all values at indices $\geq i$ by some amount. These operations could be implemented using a trivial array in $O(N)$, but the dtree we describe here is $O(\log N)$.

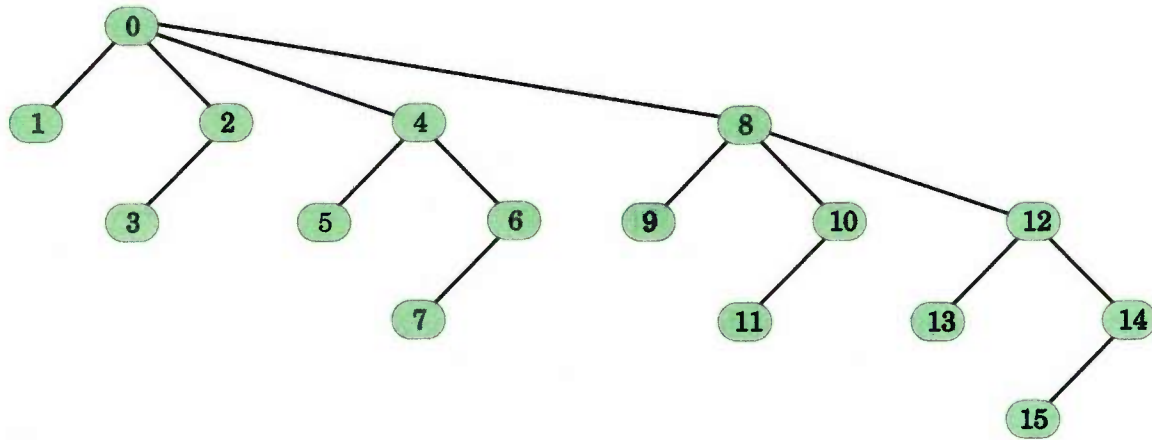


Figure 4-3: Distribution Tree. A dtree (a binomial tree) stores an array of data, allowing an amount to be added to any contiguous portion of the array in $O(\log N)$ time. The value of an array element i is the sum of the node values along the path from node i to the root. The indices of each node are shown.

A dtree is a binomial tree: it has 2^{k-1} nodes (such that $2^k \geq N$) and height k . We give each node an index such that the parent node of node i is $i \& (i - 1)$ (see Fig. 4-3). Each node also maintains a value, n_i . We then define the value of v_i to be the sum of the node values n_j along the path from the root to node i . The value v_i is computed in at most $O(\log N)$ time, as seen in Alg. 4.3.4.

Algorithm 5 dtree-get(i)

```

value = 0
loop
  value = value +  $n_i$ 
  if  $i = 0$  then
    return value
  end if
   $i = i \& (i - 1)$ 
end loop

```

Adding an amount δ to all nodes $[i, N - 1]$ is possible by adding δ to a set of nodes such that every path from the root to node $j \geq i$ goes through exactly one adjusted node. There is only one such set, which can be found by starting at node i and repeatedly traveling to the right sibling, or to the right sibling of the parent if there is no right sibling.

For example, adding δ to all nodes ≥ 3 can be accomplished by adjusting nodes n_3, n_4, n_8, n_{16} , etc. For any given node j , the next node in the sequence is $j + j \& (j - 1)$. This process guarantees that the right-most k bits of the node index are zero after k steps, thus there can be at most $O(\log N)$ steps (see Alg. 4.3.4).

Algorithm 6 dtree-add(i, δ)

```

while  $i < MAX$  do
   $n[i] = n[i] + \delta$ 
  if  $i = 0$  then
    return
  end if
   $i = i + (i \oplus (i - 1))$ 
end while

```

We now note that we can add a δ to any contiguous *subset* of nodes $[a, b]$ by simply performing two updates: one which adds δ to each element after a and another which subtracts δ from each element after $b + 1$. The operation remains $O(\log N)$.

Error Distribution Tree

The error distribution tree will allow a value Δ to be distributed over a set of buckets δ_i for $i \in [a, b]$ in proportion to the “weights” w_i of those buckets. In our map optimization application, we are distributing error around a loop in proportion to the positional uncertainty of each node².

We define the *cumulative weight* C_i of a node to be $C_i = \sum_{j=0}^i w_j$, which allows us to write δ_i as:

$$\delta_i = \delta_i + \begin{cases} 0 & i \leq a \\ \Delta * \frac{C_i - C_a}{C_b - C_a} & a < i < b \\ \Delta & i \geq b \end{cases} \quad (4.18)$$

A simple dtree could easily implement the steps for $i \leq a$ and $i \geq b$, since the quantities are constant over their ranges of indices. The interval from $a < i \leq b$ is more difficult, since it is a function of C_i , which is different for every element in the interval.

Let us implement Eqn. 4.18 in terms of *two* dtrees, *offset* and *scale* such that:

$$\delta_i = \text{offset}(i) + \text{scale}(i) * C_i \quad (4.19)$$

We can now rewrite Eqn. 4.18 so that each update is of the form $K_1 + K_2 * C_i$, such that K_1 and K_2 are constant over their intervals:

$$\delta_i = \delta_i + \begin{cases} 0 + 0 * C_i & i \leq a \\ -\Delta * \frac{C_a}{C_b - C_a} + \Delta \frac{1}{C_b - C_a} * C_i & a < i < b \\ \Delta + 0 * C_i & i \geq b \end{cases} \quad (4.20)$$

We are now done; we can implement Eqn. 4.20 by updating the *offset* dtree according to K_1 and updating the *scale* dtree according to K_2 . These operations require a constant number of *dtree-add* operations, each of which is $O(\log N)$. When

²Each error-distribution supports *scalar* operations. Since each node in the graph has three degrees of freedom, we will use three separate trees in order to be able to distribute error over x , y , and θ .

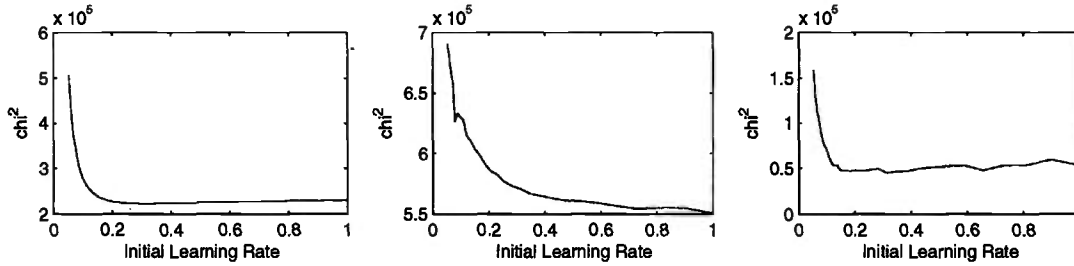


Figure 4-4: Initial Learning Rate effect on χ^2 . The initial learning rate has an impact on the convergence rate of the algorithm, as illustrated here by the χ^2 value after 20 iterations. Very low initial learning rates (less than about 0.1) incur a severe penalty, but the effect of larger learning rates is fairly modest. Left: CSW data set, middle: Freiburg, right: Intel research center.

reading from the error-distribution tree, we must read from both the offset and scale dtrees, multiplying the latter value by C_i as in Eqn. 4.19. Our total computational cost for each of these operations is $O(\log N)$.

By using this data structure, we must pay $O(\log N)$ to compute the position of any pose (i.e., to compute Eqn. 4.19). However, a constraint connects only two nodes, and so requires only two lookups.

The total computational cost of processing a constraint is thus $O(\log N)$, a significant improvement over the naive $O(N)$ implementation.

4.3.5 Initial Learning Rate

A free parameter of the algorithm is the initial, maximum learning rate. Learning rates that are too large result in large steps that often clamp. If two antagonistic constraints clamp in an iteration, little progress is made towards finding an equilibrium between them, since the constraint processed last will override the constraint processed earlier.

Conversely, a learning rate that starts off too small will exhibit slow convergence, and is less likely to escape a local minimum.

In Fig. 4-4, we have plotted the χ^2 error after 20 iterations as a function of the initial learning rate. Since the convergence behavior is also dependent on the particular optimization problem, we have plotted results for three representative data sets. It is quickly evident that low initial learning rates are far more detrimental to convergence than higher learning rates: low initial learning rates have much higher χ^2 error than higher learning rates, and there is virtually no penalty for having “too large” a learning rate. We consequently recommend an initial learning rate of about $1/3$.

4.3.6 Algorithm Summary

In the previous sections, we have described the essential elements of the algorithm. In this section, we provide a consolidated and concise description of the algorithm (see Alg. 7). We also elaborate on some implementation details.

Algorithm 7 SGD-Optimize-Graph

```

1:  $\lambda = 1/3$ 
2: repeat
3:   Randomly permute order of Constraints
4:   {Update approximation  $M = J^T \Sigma^{-1} J$ . Let  $M_i$  be the  $i^{th}$  diagonal block.}
5:   for all  $\{a, b, t_{ab}, \Sigma_{ab}\}$  in Constraints do
6:     compute  $R$  using Eqn. 4.15
7:      $W = R^T \Sigma_{ab}^{-1} R$ 
8:     for all  $i \in [a + 1, b]$  do
9:        $M_i = M_i + W$ 
10:    Update error-distribution trees with  $M$ 
11:  end for
12: end for
13:  $\Gamma = \max(\Sigma_i^{-1})$ 
14:
15: {Modified Stochastic Gradient Descent}
16: for all  $\{a, b, t_{ab}, \Sigma_{ab}\}$  in Constraints do
17:    $r = X_a^{-1} \oplus X_b$ 
18:    $r(3) = \text{mod}2\pi(r(3))$ 
19:   compute  $R$  using Eqn. 4.15
20:    $s = -(b - a)\lambda R^T \Gamma^{-1} \Sigma_{ab}^{-1} r$ 
21:    $s_{max} = X_b - (X_a \oplus T_{ab})$ 
22:    $s = \text{clamp}(s, s_{max})$ 
23:   distribute( $a + 1, b, s$ )
24: end for
25:  $\lambda = \lambda/(\lambda + 1)$ 
26: until converged

```

The essential strategy is to compute the total correction s to the map, i.e., the amount by which node b will be brought closer to the position predicted by the constraint. We can then distribute this total motion over the nodes $[a + 1, b]$, in proportion to the confidence of each individual node (i.e., redistributed according to M .)

The total correction s is easy to compute because of the block structure of the Jacobian: a total of $(b - a)$ state variables will each move by the same amount (subject to subsequent reshaping via matrix M). Since the state variables are in the x_{incr} state space, the total motion is the sum of these individual motions.

4.4 Results

We now consider the performance of our algorithm versus other algorithms. We begin with synthetic data sets, which allow us to more easily manipulate the characteristics of the problem (and provide ground truth), then demonstrate the algorithm on real

data.

Our first dataset, the CSW dataset, will include comparatively exhaustive analyses of the performance of several algorithms (including our method). Much of this analysis generalizes to other datasets.

4.4.1 CSW Dataset

The CSW dataset named after the CSAIL Student Workshop at which it was first used. It has since become a common synthetic benchmark for non-linear optimization algorithms. It has 3500 nodes in a “grid world” configuration with a total of 5598 edges (see Fig. 4-8). This gives it an average node degree of about 1.6, making it as a relatively sparse problem.

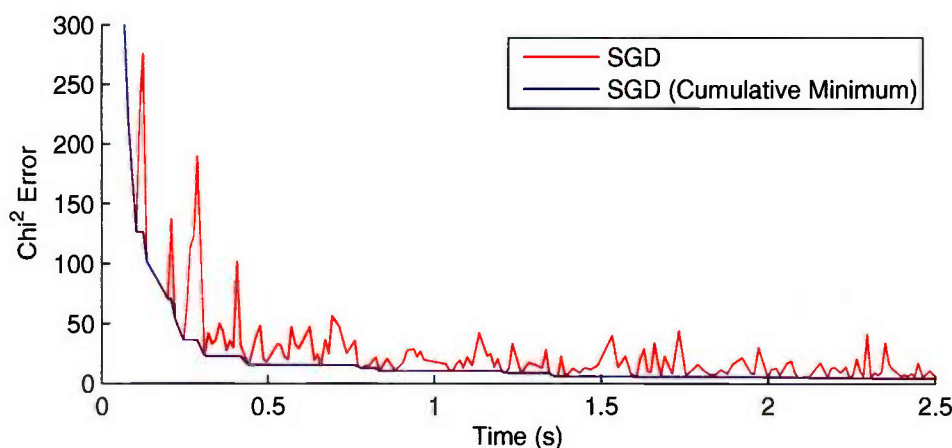


Figure 4-5: CSW Dataset, SGD Algorithm. The stochastic nature of the optimization method causes the error estimate to fluctuate near the minimum. An application that used SGD, however, would use the best-cumulative map (blue).

A typical application using SGD would not use the most recent state estimate (which is constantly exploring new parts of the state space), but would instead use the best known map to-date (as measured by χ^2 error; SSE cannot be computed when the global minimum is not yet known). The quality of the cumulative-best map versus the stochastically optimized map is shown in Fig. 4-5. Note that since SGD is a randomized algorithm, the behavior varies from run to run. We will use the cumulative minimum plot for the remainder of the figures.

As we will see, our algorithm converges to lower-error configurations significantly faster than several representative existing methods (see Fig. 4-6), with the exception of sparse Cholesky Decomposition which achieved similar performance.

Extended Kalman Filter

We also processed the CSW dataset using an Extended Kalman Filter. The total elapsed time was 2.5 hours, more than three orders of magnitude slower than either the sparse Cholesky decomposition or our method. The quadratic growth in computational complexity can be seen in Fig. 4-7. At the end of the dataset, the EKF required

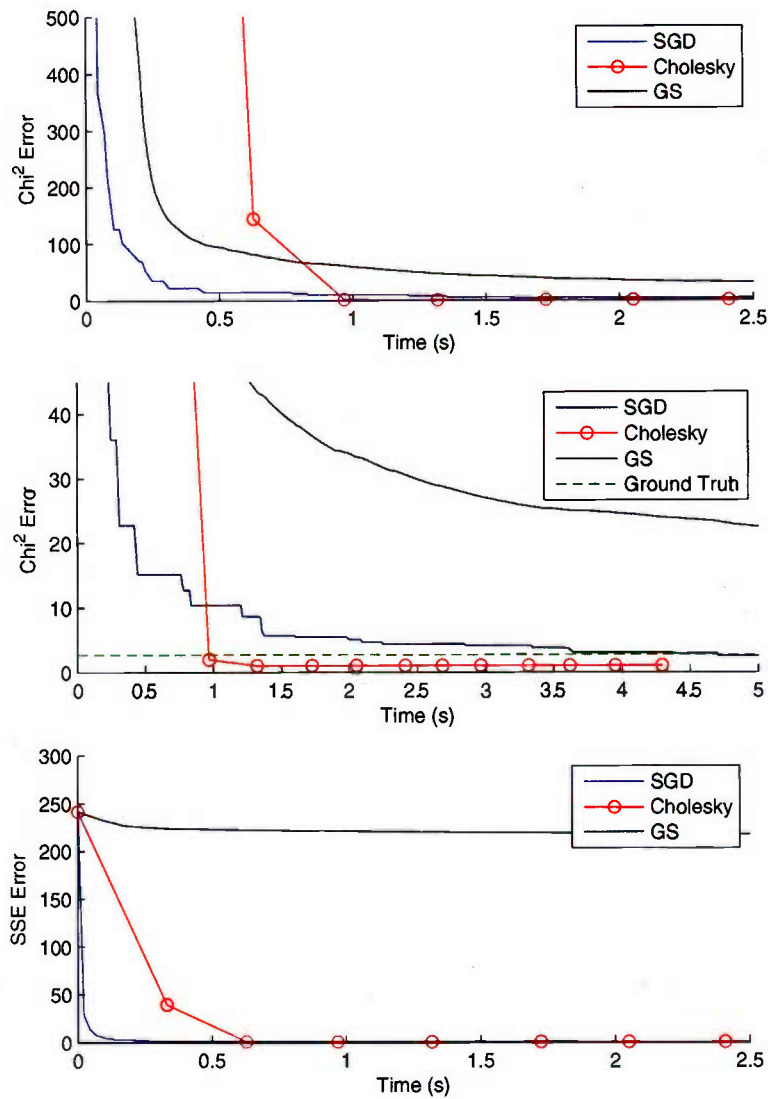


Figure 4-6: CSW Dataset Convergence Performance. The χ^2 error (top, and closeup in middle) and state squared error (SSE, bottom) paint two complementary pictures of optimization performance. Our stochastic gradient method (SGD) converges reduces both the χ^2 error and SSE error very quickly. Cholesky Decomposition converges more slowly in early steps, but finds the globally optimal solution after about one second. Gauss-Seidel relaxation, while appearing to reduce the χ^2 error reasonably well, has actually produced a very poor map estimate, as seen in the SSE plots and in Fig. 4-8.

841 MB of memory to store the covariance matrix, which grows as $O(N^2)$. The large amount of memory usage caused our Java implementation to experience more garbage collection operations, which caused some jitter in the timing measurements (readily visible beginning around constraint 2500).

Cholesky Decomposition

We have implemented a sparse Cholesky decomposition routine very similar to Delaert's $\sqrt{SAM}$ algorithm. It differs in that instead of using COLAMD [8] for deter-

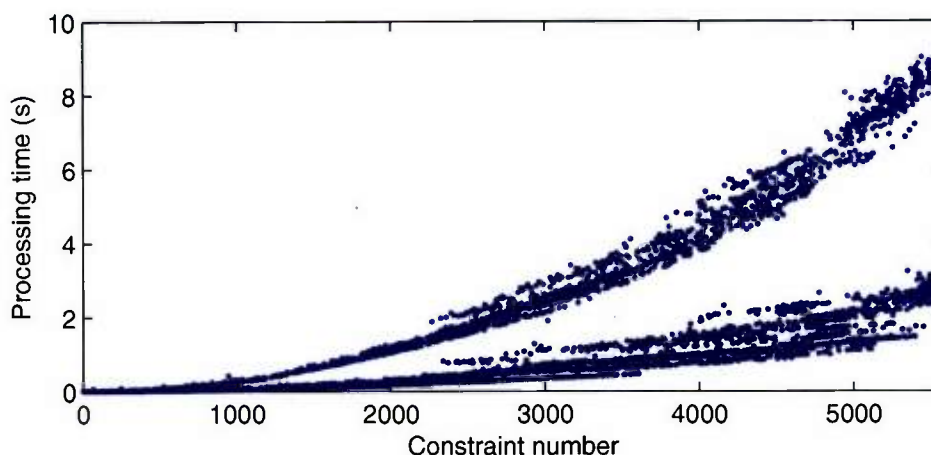


Figure 4-7: CSW Dataset, EKF Performance. Seen above are the CPU time per constraint (total elapsed time was 2.5 hours). Two trends are visible: the trend on top corresponds to the cost of performing an observation step, while the lower trend corresponds to performing a time-projection step (in which the state and covariance matrix are grown to accommodate a new state).

mining a variable ordering, it uses an exact minimum-degree ordering³ that yields less fill-in during factorization at the expense of taking longer to compute the ordering. Since the ordering needs to be computed only once (even if the Cholesky decomposition is iterated many times), and since reduced fill-in has a significant impact on the cost of Cholesky factorization, this trade-off is often a net win. A careful comparison of different variable ordering algorithms in the context of robot mapping problems has not been conducted, however, and remains an area for future study.

In this experiment, the sparse Cholesky Decomposition processed an information matrix of size 10500×10500 . The information matrix was exceptionally sparse (0.081% non-zero entries), due to the low average node degree. The resulting Cholesky factor contained 0.166% non-zero entries. This sparsity allows significant memory savings (only non-zero entries need be stored), and greatly accelerates the actual factorization and subsequent back-solves.

Sparse Cholesky decomposition is one of the fastest map optimization algorithms known, and represents a state-of-the-art algorithm. As the results show in Fig. 4-6, our algorithm reduces the error (both χ^2 and SSE) faster than Cholesky decomposition during the early iterations of optimization. However, once the Cholesky decomposition lands near the global minimum, it exhibits quadratic convergence. In cases where Cholesky decomposition finds the correct minimum, it generally finds a lower χ^2 error. However, this additional reduction in error often represents over-fitting to the data, and is not representative of an increase in map quality. This is illustrated in Fig. 4-8b, in which the χ^2 error of the ground truth is significantly higher than the χ^2 error found by Cholesky decomposition. As with machine learning algorithms in general, over-fitting can occur when a model (in this case a map) is fit to a rel-

³Finding the optimal variable ordering is NP-hard; the minimum-degree ordering is an effective heuristic. COLAMD is, in turn, an approximation of the minimum-degree ordering which can be significantly faster to compute.

atively small amount of data (the constraints). In the map optimization context, small average node degrees are fairly common, making optimization algorithms prone to over-fitting.

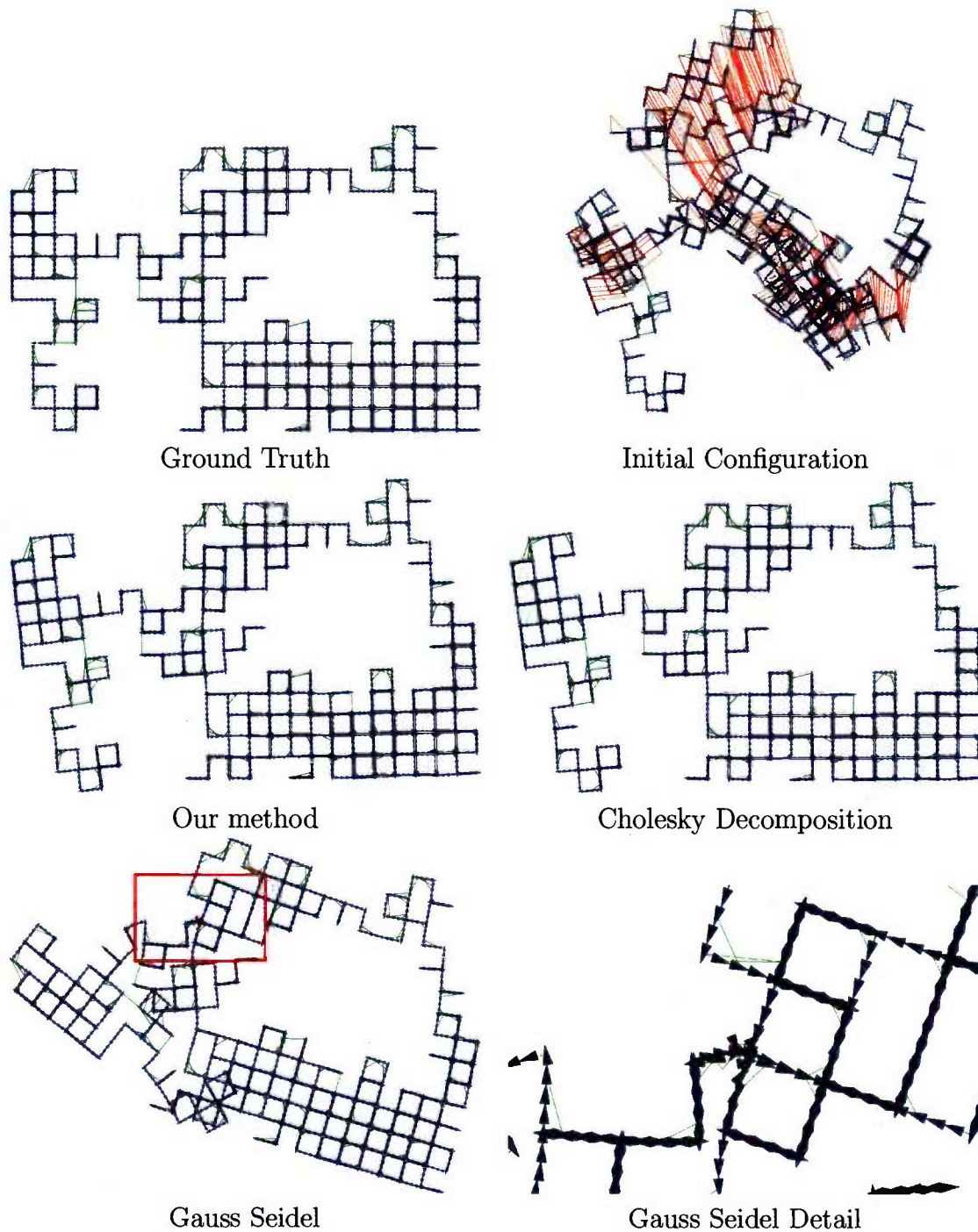


Figure 4-8: CSW Dataset Graphs. Our method and Cholesky Decomposition produce nearly indistinguishable output. Gauss Seidel converges exceptionally slowly, producing the figure above after 1.5 hours of CPU time. The detail region indicates a “knot” which is irreversibly induced early in the optimization.

Gauss Seidel Relaxation

Gauss Seidel typically exhibits poor performance on large datasets that have significant errors in their initial configuration. Intuitively, this is because each node is only affected by its immediate neighbors, but an error in one part of the graph often requires an adjustment to a distant part of the graph. With Gauss Seidel relaxation, it takes a long time for these adjustments to propagate through the graph.

This effect is significant on the CSW dataset, which has two sub-graphs connected by a “thin” chain of poses and edges. Individually, the sub-graphs have a relatively small diameter (due to the nodes’ high average node degree). Consequently, information can flow fairly rapidly throughout the sub-graph. As a result, the sub-graphs converge to low-error configurations fairly quickly. In contrast, information about the relative alignment of these two sub-graphs must pass through the thin chain of poses, which slows the rate at which the two sub-graphs can find a mutually-consistent configuration.

Consider a node j at the junction of the chain and one of the larger pieces: it has many neighbors belonging to the locally-consistent sub-graph and only one neighbor belonging to the chain. Gauss-Seidel iterations for node j will heavily favor the sub-graph’s current configuration, since all but one of its neighbors support that configuration. Thus, any new information propagated along the chain is rapidly attenuated when it enters a sub-graph. As a result, the two sub-graphs converge to a mutually-consistent arrangement very slowly.

The resulting graph can have very low χ^2 error even when the two sub-graphs are poorly aligned with respect to each other. This is because almost all of the edges in the graph belong to one of the two large sub-graphs (which have found locally consistent configurations)—the error is concentrated along just a handful of edges belonging to the chain. The total χ^2 error for the graph is thus quite low, even though the positions of the nodes are far from the global minimum. This illustrates why χ^2 error cannot be used as a quality metric by itself, and why our proposed SSE metric is important for quantitative evaluations.

In early steps of Gauss-Seidel, large errors can cause nodes in the graph to move dramatically. These large state updates generally reduce the χ^2 error dramatically, which leads to smaller steps later on. However, these initial large steps can introduce large errors into the positions of some nodes that later (smaller) steps cannot undo the damage. In the CSW dataset, a node near the junction of the right and left halves of the graph experiences such a large update that it effectively ties a “knot” into the robot trajectory. However, this knot has so much less χ^2 error than the initial configuration that subsequent Gauss Seidel iterations do not have the energy to escape it. In other words, Gauss Seidel has become stuck in a local minima of its own creation. (This knot is shown in more detail in Fig. 4-8.) Given a week of CPU time, the Gauss-Seidel solution macroscopically resembles the solutions given by our method and Cholesky decomposition—however, the knot remains.

A typical way of accelerating Gauss Seidel relaxation is by performing *over-relaxation*. It addresses the fact that nodes tend to move too slowly in response to distant constraints. The intuition is that moving *past* the locally-optimal solution

may in fact move the state estimate closer to the globally-optimal solution. Indeed, over-relaxation *can* increase the convergence rate of Gauss-Seidel (modestly, in our experiments), but the larger step sizes also greatly increase the risk of “knots” and divergence.

4.4.2 High Noise Dataset

Our second dataset illustrates the effect of noisy odometry and loop closures (see Fig. 4-9). It is a synthetic dataset of 501 nodes with 544 edges, i.e., reflecting situations in which loop closures are difficult to obtain.

In other datasets presented in this thesis, it is usually possible to recover a map that subjectively looks like the ground truth. In this high-noise dataset, this is not the case: even the best map is visibly imperfect. Producing the best map *possible*, even under high noise situations, is a key test of an optimization algorithm. Success in these cases allows robots to operate in more difficult environments and use less expensive (noisier) sensors.

In this dataset, the only algorithm to produce a good result is our method. The convergence rate of Gauss Seidel, while initially rapid, quickly drops off and leads to a poor solution in both terms of χ^2 and SSE.

The Extended Kalman Filter solution, which took 17 s to compute, suffered from significant linearization error, leading to both the worst χ^2 score (31.9) and the worst SSE score (45.6). While EIFs, SEIFS, and ESEIFs might be faster than the EKF, they will exhibit similar error. This illustrates the importance of methods that can relinearize its constraints.

The solution computed by Cholesky decomposition is visibly distorted (see Fig. 4-9), and has a significantly higher SSE error (see Fig. 4-10) than our solution. The first few iterations of Cholesky decomposition were very chaotic; the violent updates to the state estimate resulted in a “knot” in the center of the map, much like the knot that resulted when using Gauss-Seidel on the CSW data set. Additional iterations of Cholesky decomposition have no effect on this knot.

The similarity of the χ^2 errors (3.5 for Cholesky, 3.06 for our method), illustrates that χ^2 error is not always a good indicator of solution quality; SSE is more revealing in this case.

4.4.3 P5K20K Dataset

P5K20K is a synthetically generated dataset with 5000 nodes and 20000 edges. This gives it a fairly high average node degree, which, with the fairly large size of the state vector, makes it is a challenging optimization problem (see Fig. 4-11 and Fig. 4-12).

The Cholesky factorization is much slower on this problem than on previous examples due to the fairly high average node degree and relatively large sensing range. The result is that the Cholesky factor is an order of magnitude more dense than on the CSW dataset: the information matrix has a similar density (0.086%), but the Cholesky factor has a density of 1.96% (versus 0.166% on the CSW dataset).

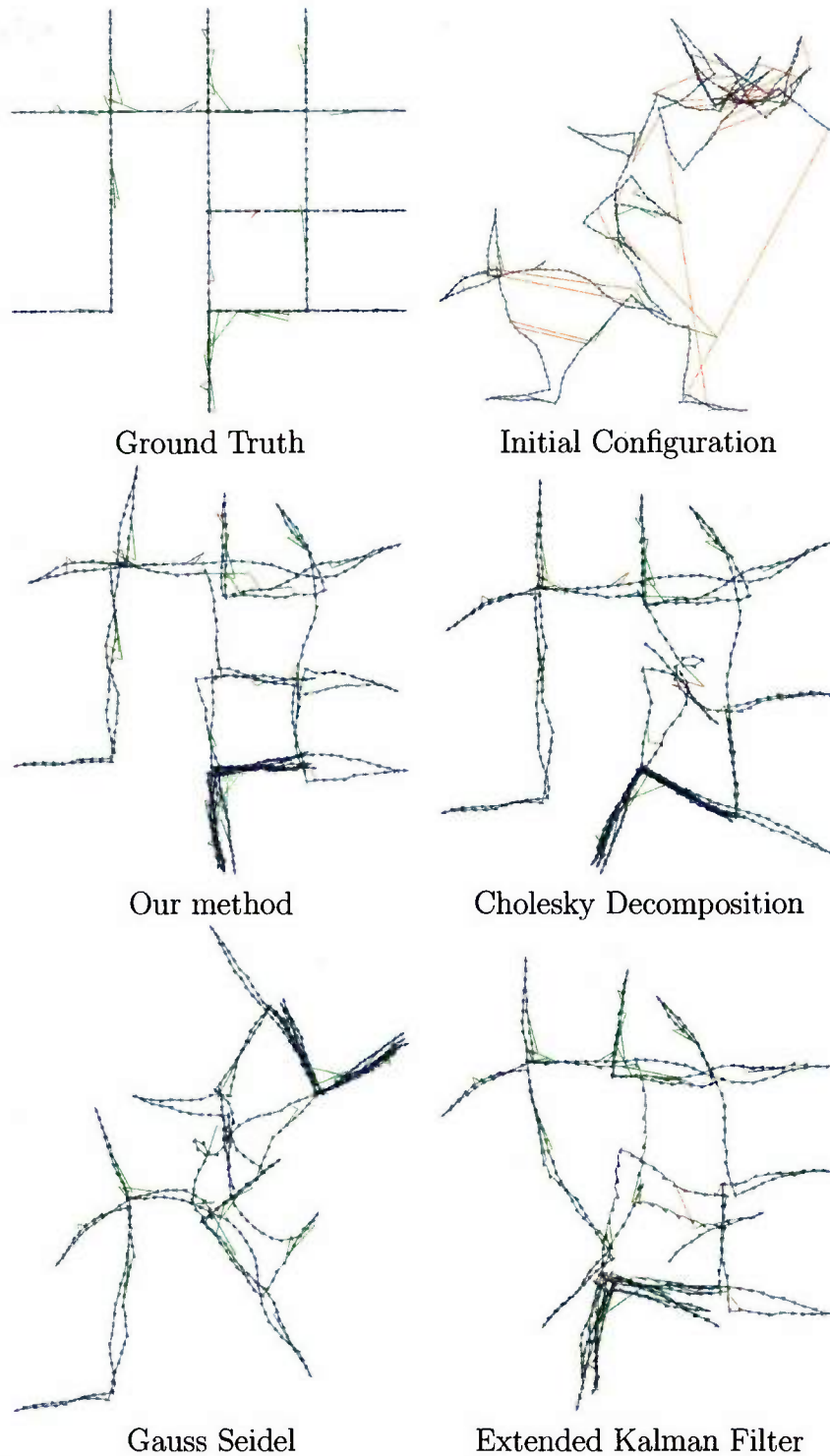


Figure 4-9: High Noise Dataset. The true configuration (top left) and odometry-derived initial configuration (top right) are shown. In this high-noise case, only our method finds a reasonable solution. Cholesky Decomposition has become stuck in a local minimum (note the knot in the center of the map), while Gauss Seidel and the Extended Kalman Filter have both produced poor maps.

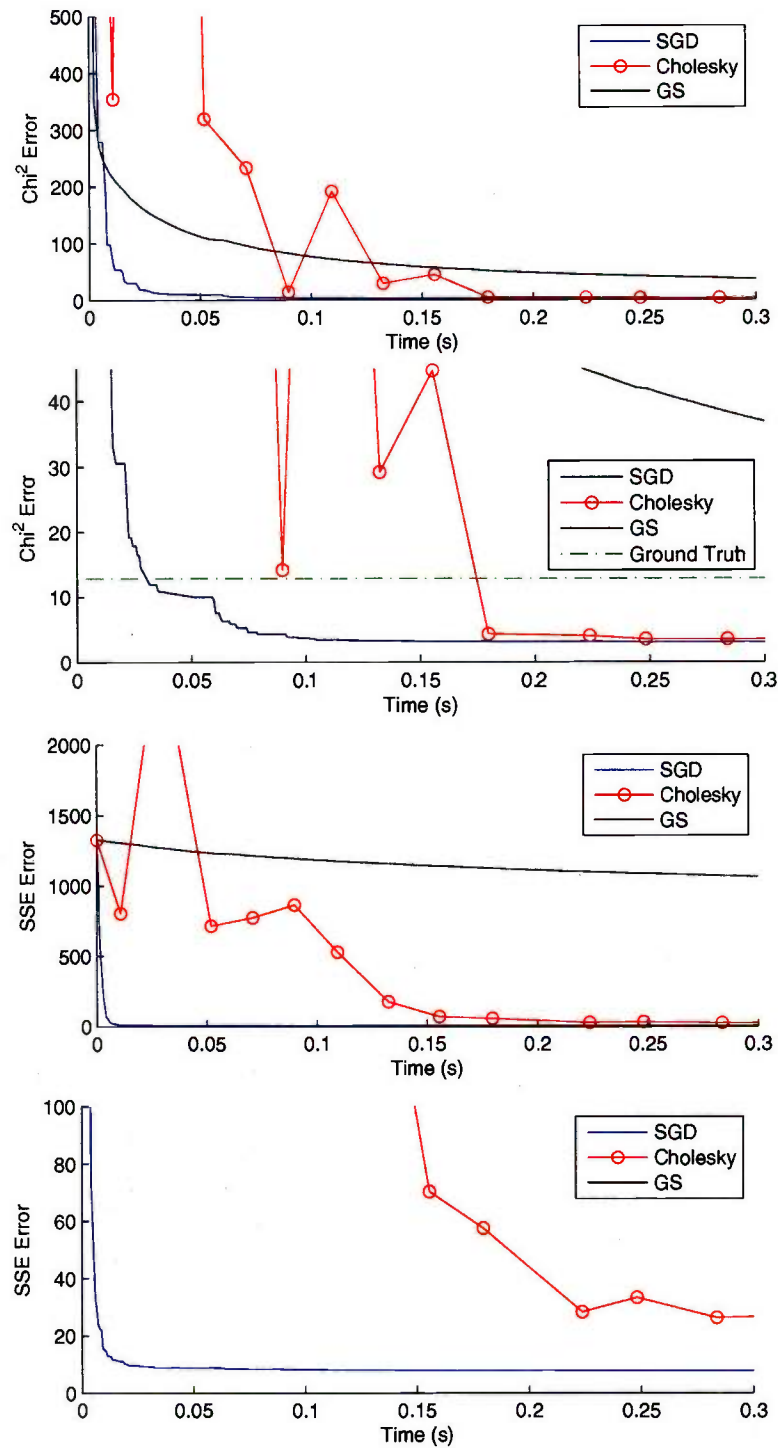


Figure 4-10: High Noise Results. The χ^2 error and SSE error are plotted for three different algorithms. Each error metric is plotted twice: the second plot is an enlargement of the first to show detail. Only our method finds the global minimum. Cholesky decomposition gets stuck in a local minimum with fairly low χ^2 , but large SSE . (Not shown: EKF took 17.2s, and had $\chi^2 = 31.9$ and $SSE = 45.6$.)

It takes Cholesky Decomposition three iterations totalling 91 seconds of computa-

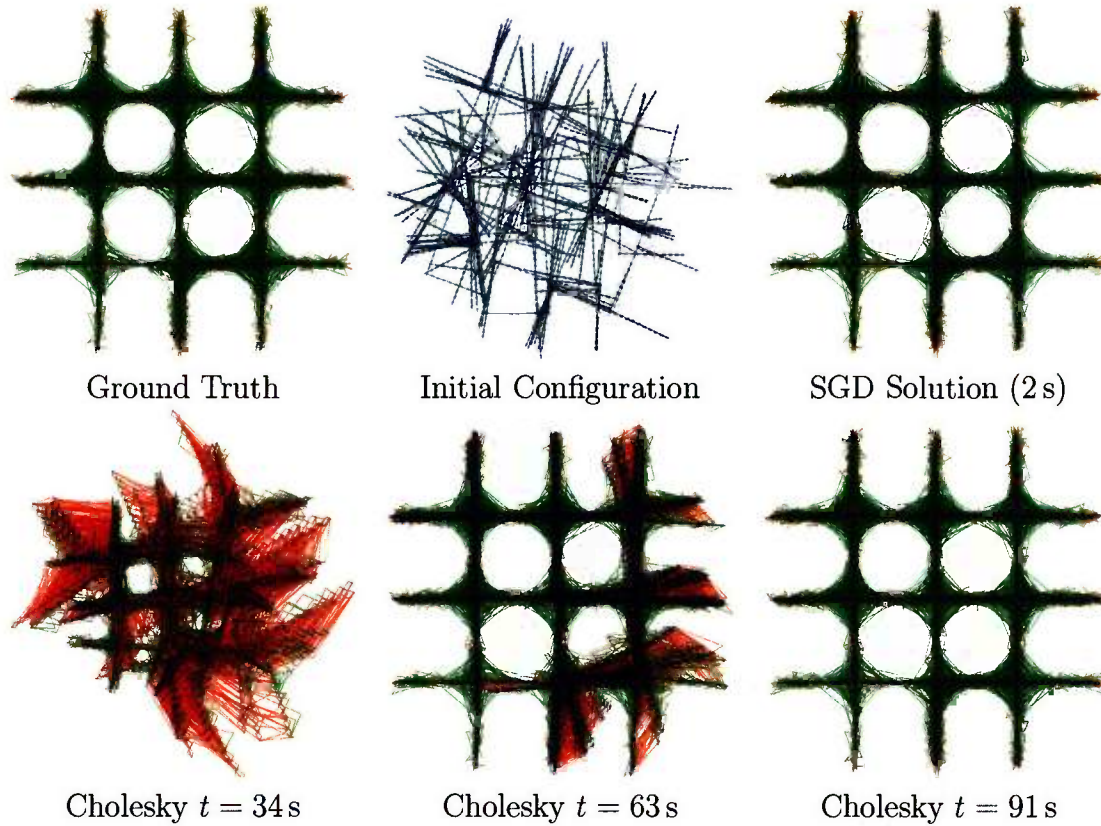


Figure 4-11: P5K20K Graphs. The true configuration (top left) and an odometry-derived initial configuration (top middle) are shown. The configuration resulting from three successive iterations of Cholesky Decomposition are shown (bottom, left to right). After 91 seconds, its solution is comparable to our solution (top right), which was obtained in 2 seconds.

tion to produce a solution ($\chi^2 = 1.47$, $\text{SSE}=0.0387$) similar to the solution produced by our method in 2 seconds ($\chi^2 = 2.9$, $\text{SSE}=0.0307$). Given a fourth iteration, Cholesky Decomposition improves the estimate to $\chi^2 = 0.9983$ and $\text{SSE}=0.00744$, but with a total elapsed time of 124 s. Interestingly, this solution is numerically optimal. Our method cannot typically recover the numerically optimal solution due to its stochastic exploration of the state space and the effects of approximations made in the algorithm. However, as this dataset illustrates, our method produces good maps in a fraction of the time.

The χ^2 error of the solution produced by our algorithm can be reduced significantly by using a few iterations of Gauss-Seidel relaxation. Such a hybrid method combines SGD's ability to rapidly recover the basic structure of the graph with Gauss-Seidel's ability to fine-tune local structure (see Fig. 4-13).

The hybrid SGD/Gauss-Seidel method can noticeably reduce the χ^2 error, but has little effect on SSE error. Tiny perturbations in the positions of individual poses can have a significant impact on the error of individual constraints, but so long as these perturbations do not have a cumulative effect, their effect on SSE is small. SGD's stochastic exploration of state space tends to add noise to the position of each node in the graph: this causes an increase in χ^2 error of the constraints connected

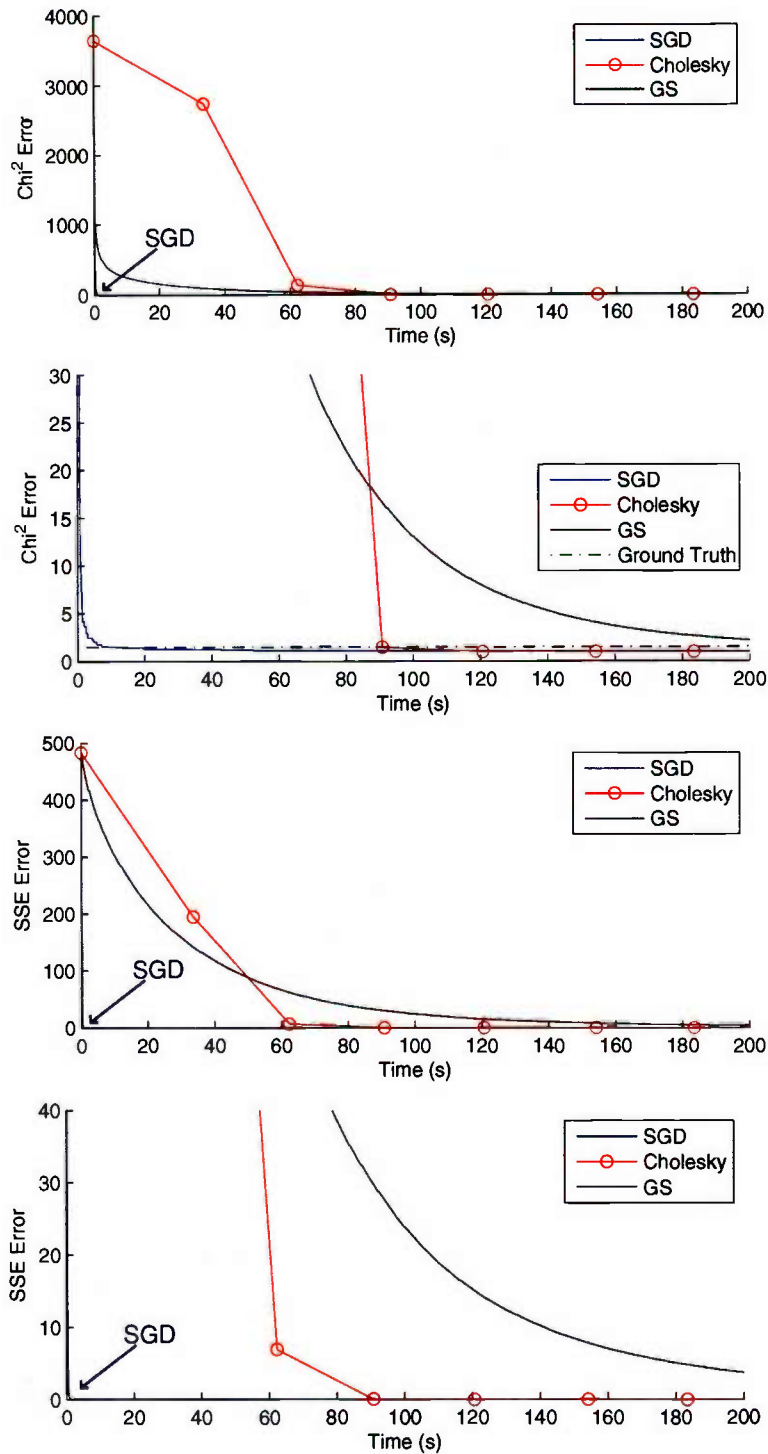


Figure 4-12: P5K20K Results. χ^2 and SSE error are plotted (with enlarged versions for each) for several algorithms. SGD converges much more quickly than the other methods. Unlike previous data sets, however, all methods ultimately converge on a good solution.

to those nodes, even though those nodes are very close to their optimal position. Running Gauss-Seidel allows the noise to be corrected, significantly reducing the χ^2 error by making minute adjustments to the position of each node. However, these

minute adjustments have little effect on SSE. This is precisely the behavior we see in Fig. 4-13. This behavior also illustrates that χ^2 error, by itself, is not necessarily indicative of map quality.

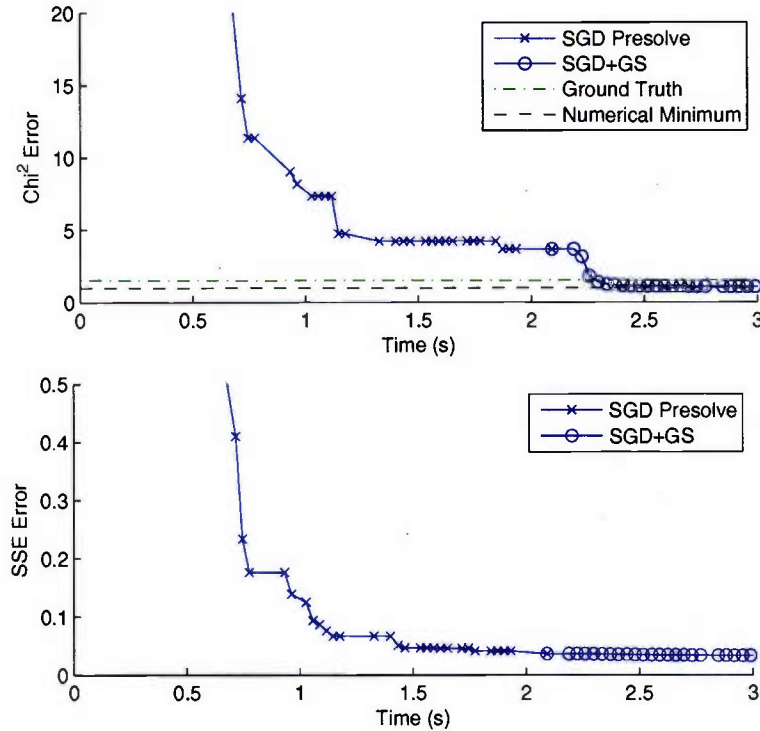


Figure 4-13: Hybrid Method Results on P5K20K Dataset. Our method can be combined with Gauss-Seidel to produce results with lower χ^2 error by using Gauss-Seidel to refine a configuration produced by our algorithm. In contrast, the effect on SSE error is minimal; this shows that SGD produces graphs whose quality is higher than χ^2 alone would indicate.

4.4.4 Intel Research Center Dataset

The Intel Research Center dataset is composed of 875 nodes connected via 15605 edges, and represents a real data set spanning 44.9 minutes [29]. Cholesky, Gauss-Seidel, and our method all perform well on this dataset. Although our method does very well with respect to SSE error⁴, our method lags behind in terms of χ^2 error. Our method is an approximation that neglects certain projective effects (as seen in Fig. 4-16). That, coupled with its stochastic exploration of the state space, limits its ability to find extremely low-error configurations. In this case, all the methods achieved very low errors ($\chi^2 < 1$), and so SGD's larger χ^2 error is more conspicuous. However, as we have seen in earlier datasets, this does not mean that the map is worse. If minimizing χ^2 error is critical, Gauss-Seidel relaxation can be used to fine-tune the solution computed by SGD.

⁴In the absence of an actual ground truth, we compute SSE with respect to the global minimum solution.

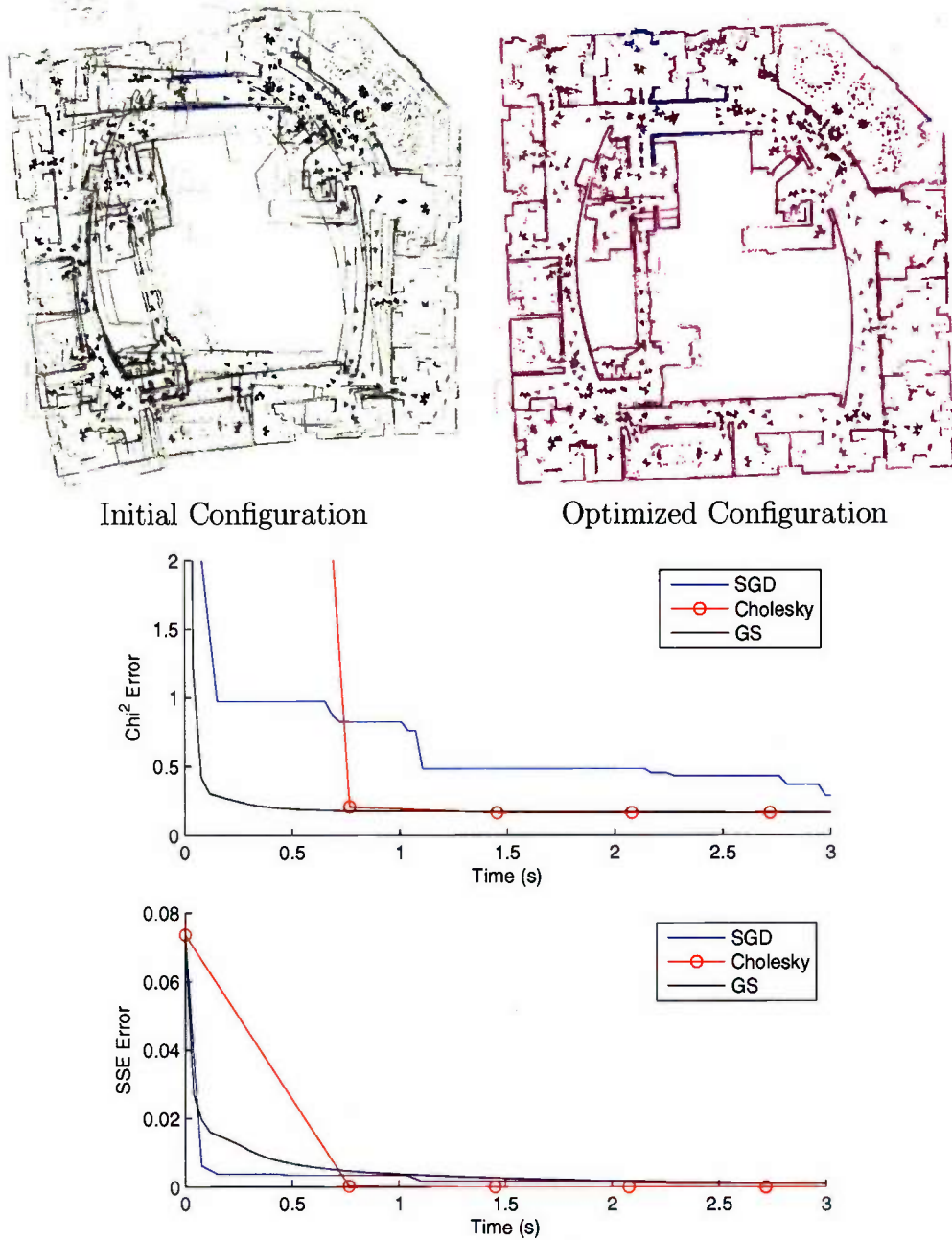


Figure 4-14: Intel Research Center. All the methods achieve very low error, though their convergence rates differ. Note that our method takes a quick and early lead in SSE error, though it lags in χ^2 error.

4.4.5 Stanford Gates Dataset

The Stanford Gates dataset includes 34.6 minutes of data. As processed by our lidar loop-recognition algorithms (described in Chapter 3), the raw data resulted in a graph containing 679 nodes and 7121 edges. Fig. 4-15 shows the initial configuration and optimized configuration, with timing results for three algorithms.

Only Gauss-Seidel produces a poor result: Cholesky Decomposition, EKF, and

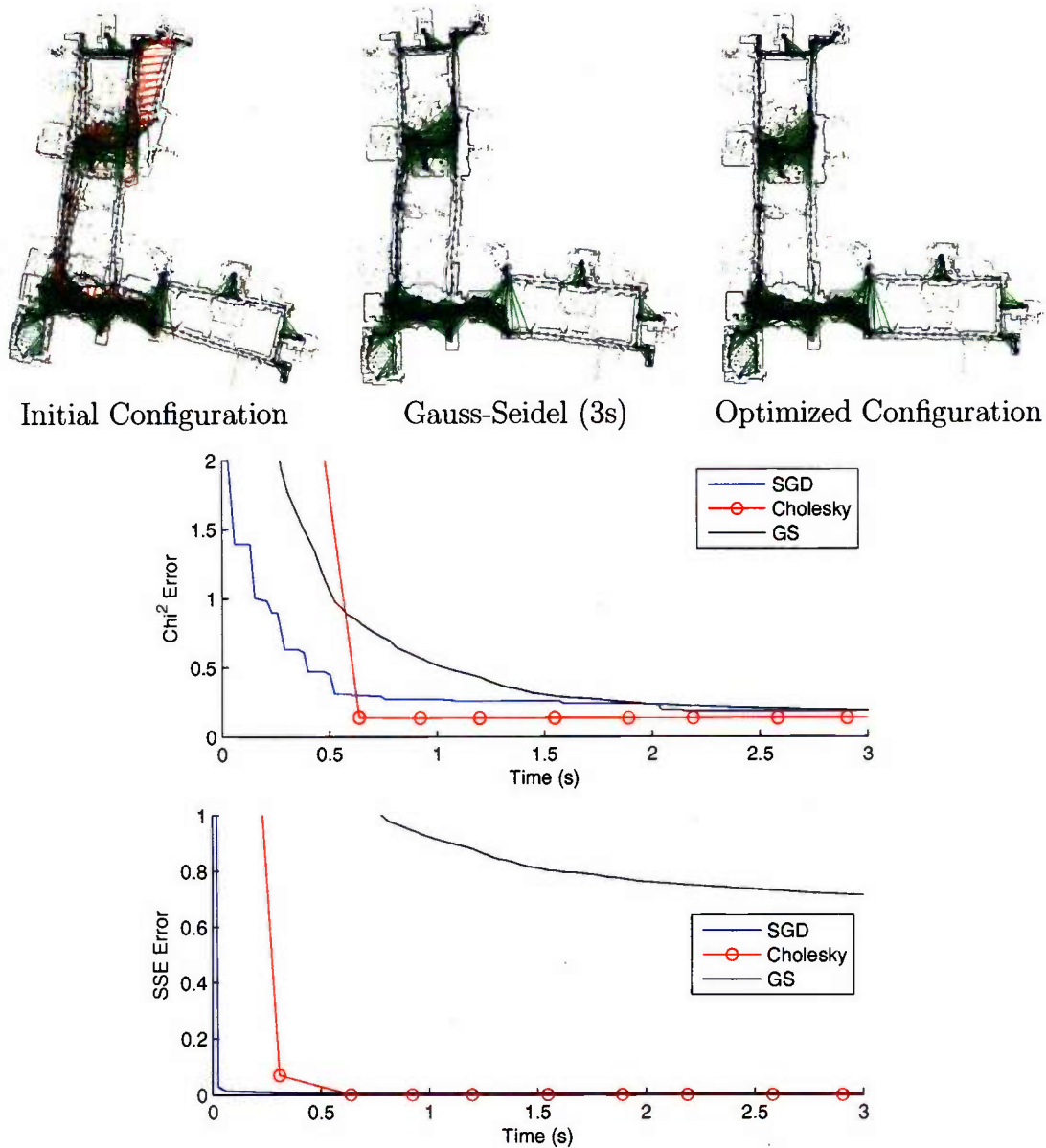


Figure 4-15: Stanford Gates Building Results. Once again, Gauss-Seidel quickly reduced the χ^2 error, but results in a map with noticeable distortions (and large SSE). Both Cholesky decomposition and our method quickly produced good maps, with our method producing low-error maps earlier. (Not pictured: a map produced by an EKF with $\chi^2 = 0.134$ and $SSE=0.00031$ in 881 s.)

our method all produce very good maps. Our method converges faster than other methods initially, though Cholesky Decomposition ultimately produces a graph with lower χ^2 error. The Extended Kalman Filter is orders of magnitude slower than either Cholesky Decomposition or SGD.

4.4.6 Dog Leg Configuration

In certain contrived cases, errors due to our approximation of the Jacobian can be observed. We have purposefully eliminated many projective effects in both our state

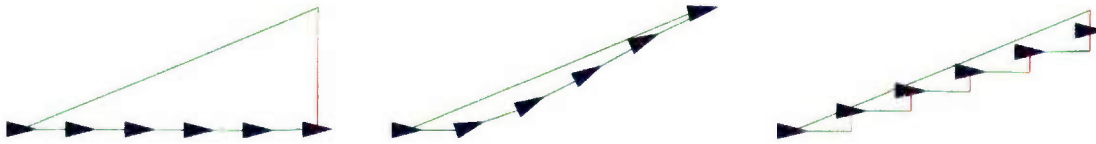


Figure 4-16: Dog leg configuration. The dog leg initial condition (left) has a single ill-satisfied constraint; the optimal solution is shown in the middle. Our proposed algorithm converges to an approximate solution that does not reflect all of the projective constraints in the configuration. This effect is minor in non-contrived cases, however.

representation and in our Jacobians. These allow fast and robust updates, but they are not exact.

Fig. 4-16 shows the “Dog Leg” configuration⁵. The nodes are initialized along a straight line. A constraint is added between the first and last node that suggests a different position (but *not* a different orientation) of the last node. In the initial configuration, the orientation error for every constraint is zero, and so the state updates computed by our method include no rotational component. A lower χ^2 error can be obtained by rotating the intermediate nodes, as seen in the middle of the figure.

Real-world datasets rarely exhibit artifacts as noticeable as those in this contrived case. This type of error is significant only when there are large rotational errors between physically distant nodes and there are no other constraints available to help guide the orientation of the intermediate nodes.

4.4.7 Robustness

Compared to other algorithms, our proposed method is substantially more robust to initialization error. Specifically, our method can find the global minimum even when other methods get stuck in local minima.

This robustness can be measured. The likelihood of an algorithm getting stuck in a local minimum increases with the noise of the observations in the graph: noisy observations are more likely to cause the algorithm to erroneously step in a sub-optimal direction. We can test the robustness of an optimization algorithm by generating a large number of synthetic datasets with varying degrees of noise, and counting the number of graphs that each algorithm successfully solves. We detect failure by comparing the normalized χ^2 to 1.0, its expected value at the global minimum: convergence to significantly larger values indicates a local minimum.

Using a synthetic graph generator, we generated about 15000 graphs. Each graph consisted of 1000 nodes and 200 landmarks; nodes were connected by odometry measurements (whose noise was the independent variable). Finally, 5000 landmark observations were generated with constant-magnitude noise.

We expect, a priori, that algorithms will have an increasing tendency to get stuck in a local minimum with increasing noise. For this study, we continue to optimize until either a generous time bound is exceeded or until the algorithm’s convergence

⁵We thank Udo Frese for suggesting the Dog Leg configuration.

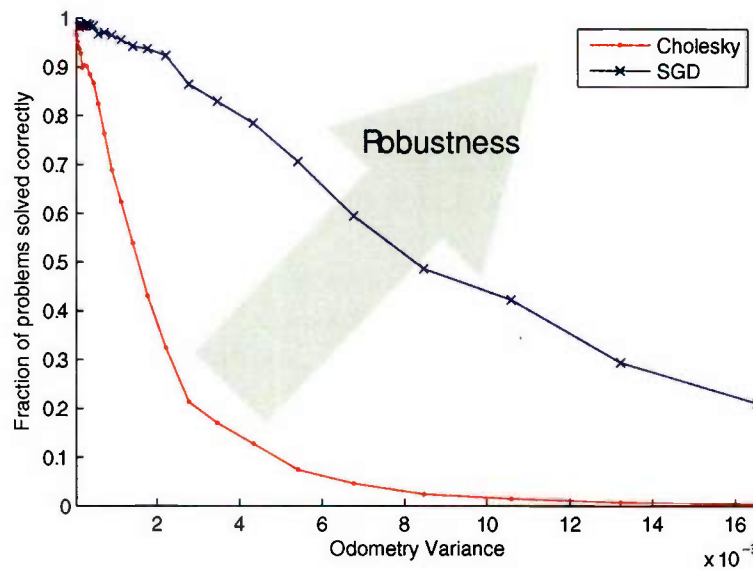


Figure 4-17: Convergence Robustness. On problems with significant noise and poor initial estimates, our algorithm finds the global minimum much more often than a Cholesky-Decomposition based method.

rate has slowed dramatically. We cannot reliably distinguish between a true local minimum versus a long valley that ultimately does lead to the global minimum, but from a practical standpoint, these two cases are equally problematic to applications.

We compare our approach to a Cholesky Decomposition-based method similar to Dellaert’s $\sqrt{SAM}$. This Cholesky Decomposition based approach is one of the most stable and fastest performing algorithms that we have tested (and generally outperforms Gauss-Seidel and non-iterated forms of the Kalman filter and Information Filter). However, our algorithm consistently produces better results (normalized χ^2 values near the expected value of 1.0) at much higher noise levels than Cholesky Decomposition. In practical terms, this means that our algorithm can produce correct answers when Cholesky Decomposition (and other similar methods) produce unusable maps.

Our algorithm selects constraints in a pseudo-random way, taking care to ensure that every constraint is processed before any constraint is processed a second time. Processing the constraints in random order increases the algorithm’s ability to explore the state space (and thus to escape local minima).

We have observed that when our algorithm fails to produce a good map, the optimization often diverges: the χ^2 increases by orders of magnitude. Since χ^2 error has an expected value of 1.0, such divergences are easy to detect. A robot could conceivably perform some fail-safe behavior when failure was detected. In contrast, when Cholesky Decomposition produces a bad map, the χ^2 error is often large, but not terribly so (e.g., less than 100). In cases where sensor noise covariances have been poorly modelled, it could be difficult to differentiate between map divergence and covariance modelling error.

4.5 Summary

We have presented a new map optimization that is not only fast, but remarkably robust to poor initial conditions and noisy observations. Our approach adapts stochastic gradient descent, a learning algorithm previously used in training neural networks, and introduces a state-space representation that approximates the cumulative nature of robot motion.

We compared our method to the previous fastest non-linear method, which is based on sparse Cholesky factorization. Numerous experiments demonstrate that our method is competitive with, or faster than, existing methods. More critically, we demonstrated that Cholesky Decomposition and similar methods are significantly more susceptible to getting stuck in local minima than our algorithm. In short, these methods produce poor maps in cases where our algorithm succeeds.

We also introduced a new error metric, the mean squared error of the state estimate (SSE). Our results show that it provides a useful measure of the graph quality, and that χ^2 error alone does not paint a complete picture of optimization performance.

CHAPTER 5

Incremental Optimization of Pose Graphs

When a robot obtains new information about the environment, this information must be rapidly incorporated into the model of the world so that the robot’s obstacle avoidance and path planning activities can take that information into account. An incremental optimization algorithm allows new information (nodes and edges) to be added to a pose graph, and provides for an efficient incremental update of the posterior map.

Most existing optimization methods are inherently incremental¹. Recursive linear solutions (e.g., Extended Kalman Filter (EKF), Extended Information Filter (EIF), and their many variants) explicitly compute the importance of a new observation in relation to all previous observations: this allows new information to be incorporated in a single step. However, the cost per step increases quadratically in the size of the map, and poor maps can result due to the effects of linearization error.

Non-linear optimization methods like Gauss-Seidel [12] and Multi-Level Relaxation (MLR) [20] maintain no state aside from the state estimate itself. The graph can be updated between iterations without any complications. TreeMap [19] and Thin Junction Tree Filters [52] do maintain some auxiliary state (a pruned tree), but adding new nodes is relatively straightforward. Once nodes are added to the tree, the actual optimization algorithm is unchanged.

Dellaert’s $\sqrt{SAM}$ [10] method also maintains additional state, namely a variable reordering. More problematic, however, is the fact that the information matrix must be factored at each iteration—a potentially slow operation. Kaess has proposed modifications that make incremental usage more efficient, including incremental variable reordering strategies and using QR (instead of Cholesky) decomposition [31].

Our batch algorithm, based upon stochastic gradient descent, has been shown to be both fast and robust. Naturally, these attributes would be very useful in an incremental context. Stochastic gradient descent (SGD), however, assumes that the

¹The terms *iterative* and *incremental* describe distinct concepts. An *iterative* algorithm is one in which the quality of the solution can be improved by running the algorithm repeatedly. An *incremental* algorithm is one in which the underlying problem can be modified without requiring the algorithm to restart from initial conditions. They are orthogonal features of an algorithm. Both the batch algorithm described in Chapter 4 and the algorithm described here are iterative, but this chapter adds an incremental capability.

underlying optimization problem does not change over time. SGD uses a learning rate that varies as a function of time: this learning rate modulates the step size, allowing an equilibrium between all the constraints to be found.

Quickly adapting the posterior estimate to reflect new information can require large steps, which are hampered by small learning rates. Consequently, the learning rate may need to be increased when new information is added to the graph.

A simple strategy would be to reset the learning rate to its initial value, but this would be equivalent to restarting the batch optimization. This is undesirable: even though SGD rapidly decreases the error of the posterior, the robot may have already expended significant computational resources optimizing the map. Turning the learning rate back up will undo much of this work, since *all* the constraints will once again take large steps.

In this chapter, we describe an incremental version of stochastic gradient descent. The central contribution is a spatially-varying learning rate that allows us to incorporate new information without obliterating the results of previous iterations. To our knowledge, this is the first instance of stochastic gradient descent being used to estimate the parameters of a dynamic optimization problem.

We also describe a technique for performing iterations on only the parts of the graph that are most poorly optimized. New information most often perturbs areas near the robot much more than distant areas. Our method allows computational effort to be focused on optimizing the area near the robot, without needlessly refining already-optimized areas farther away.

5.1 Method

The learning rate scales the update step produced by each constraint. If we wish to rapidly incorporate information from a new constraint, the learning rate may need to be increased. In the case of the batch algorithm, in which there is a single (global) learning rate, an increase of the learning rate will cause *every* constraint to start taking larger steps. As a result, even if the effect of the new constraint is limited to a small portion of the graph, the whole graph will be perturbed away from the minimum.

Our solution is to allow different parts of the graph to have different learning rates. When a new constraint is added to the graph, the learning rate can then be selectively increased. This allows unaffected parts of the graph to retain their low-error configurations while volatile parts of the graph are allowed to rapidly find a new equilibrium.

Specifically, instead of a single global learning rate λ , we will give each node i its own learning rate Λ_i . These learning rates represent the volatility of each node in the graph. The bulk of this chapter is devoted to a set of rules describing how these learning rates should be adjusted in order to accommodate new information. The batch algorithm can be described as a special case of the incremental algorithm: in the absence of new observations, the learning rates Λ_i will all equal the learning rate λ of the batch algorithm.

In general, we note that the maintenance of learning rates is a matter of *convergence speed*, rather than correctness. Even if the learning rate is *never* increased, stochastic gradient descent will ultimately arrive near a local minimum—but it might take a long time to do so. The methods in this chapter comprise an effective set of heuristics for managing the learning rates; they model the interactions of constraints so that the learning rate can be increased sufficiently to allow rapid convergence. Simultaneously, they attempt to avoid unnecessary increases in learning rates, but as we saw in Chapter 4, there is generally a bigger performance penalty associated with picking too small a learning rate than picking too large a learning rate.

There are three basic operations that affect the learning rates: adding a new constraint (which can selectively increase learning rates), processing existing constraints (which may propagate the effects of previous learning rate increases to additional learning rates), and reducing the learning rates according to a generalized harmonic progression.

5.1.1 Adding a new constraint

When adding a new constraint, we must estimate how large a step should be taken to reduce its residual. Once determined, we can compute the learning rate that will permit a step of that size.

The graph's current state estimate already reflects the effects of a number of other constraints. The step resulting from the addition of a new constraint should reflect the certainty of the new constraint relative to the certainties of the constraints already incorporated into the graph. Let gain β be the fraction of a full-step that would optimally fuse the previous estimate and the new constraint. β can be derived by differentiating the χ^2 cost of two Gaussian observations of the same quantity, or manipulated from the Kalman gain equation:

$$\beta = \Sigma_i^{-1}(\Sigma_i^{-1} + \Sigma_{graph}^{-1})^{-1} \quad (5.1)$$

Computing β exactly would be a computationally expensive operation (equivalent to performing a Kalman step). Instead, we estimate Σ_{graph}^{-1} from the diagonals of the information matrix. We have already approximated these uncertainties in our diagonal approximation to the information matrix M , which we described in Chapter 4. This is an approximation because, in truth, the states are correlated. However, it still yields a serviceable approximation.

We now compute the learning rate that will allow a total correction s of size βr_i . Consider the state update equation (unchanged from Chapter 4), where s represents the total correction:

$$s_i = \lambda(b - a)R_i^T \Sigma_i^{-1} r_i = \beta r_i \quad (5.2)$$

Using Eqn. 5.2, we can solve for the learning rate that would result in a step of size $s = \beta r_i$. Because there are three degrees-of-freedom per pose, we obtain three simultaneous equations for λ ; we could maintain separate learning rates for each, but

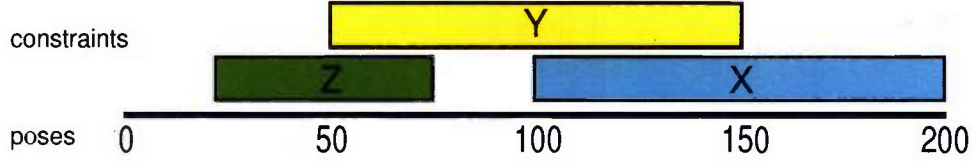


Figure 5-1: Three overlapping constraints. A disturbance in constraint X will be propagated through Y to Z, even though X and Z do not overlap. At each step, the effect of the disturbance is attenuated.

we use the maximum value for all three. With $\oslash$ representing row-by-row division, we can write:

$$\lambda = \max_{\text{row}} \left(\frac{1}{b-a} (\beta r_i \oslash \Omega R \Sigma_i^{-1} r_i) \right) \quad (5.3)$$

This value of λ is then propagated to all of the poses after pose a , since the resulting state update will adjust all of those nodes. This is due to our use of the x_{incr} state space: we are estimating the robot's motion at each time, and the position of the robot is the sum of its previous motions. Note that monotonicity of learning rates Λ_i is important since an adjustment to an incremental state variable i has an effect on all the subsequent positions $j \geq i$:

$$\Lambda'_i = \max(\Lambda_i, \lambda) \quad \text{for } i > a. \quad (5.4)$$

5.1.2 Processing existing constraints

When processing an old constraint, we must determine what *effective learning rate* should be used when calculating its step size. If a new constraint has been added to the graph, it may upset existing constraints. In order for all of these constraints to rapidly find a new equilibrium, the effective learning rate of other constraints needs to increase in response to the newly added constraint.

The learning rates for each node are set in proportion to how much they are perturbed by new constraints. Thus, the average amount of perturbation affecting a constraint between a and b is the average of learning rates Λ_i for $i \leq a \leq b$. Constraints that overlap with a new constraint will see an increase in their effective learning rate in proportion to their overlap with the new constraint. The resulting state update will perturb all subsequent poses (again due to our use of the x_{incr} state space); as before, we update the learning rates so that all the affected poses have a learning rate at least as large as the effective learning rate.

Consider an example with three constraints (see Fig. 5-1) with a newly-added constraint X between poses 100 and 200, and existing constraints Y (between poses 50 and 150) and Z (between poses 25 and 75). The learning rate increase caused by constraint X will cause an increase in the effective learning rate for constraint Y. On the next iteration, constraint Z will also see an increase in its effective learning rate, due to the learning rate increases caused by Y. In other words, constraint Z will be affected by constraint X, even though they have no poses in common. Their

interaction is mediated by constraint Y . Each interaction attenuates the learning rate increase in a manner related to the overlap between the constraints. This “percolating” effect is important in order to accommodate new information, even though the effect is generally small.

5.1.3 Algorithm Summary

In summary, adding a new constraint (or constraints) to the graph requires the following initialization:

1. Compute the constraint’s effective learning rate λ_{eff} using Eqn. 5.3 and perform a step.
2. Increase the learning rates, as necessary:

$$\Lambda'_i = \max(\Lambda_i, \lambda_{eff}) \quad \text{for } i > a \quad (5.5)$$

Updating an existing constraint between poses a and b involves three steps:

1. Compute the constraint’s effective learning rate λ_{eff} by computing the mean value of the learning rates for each pose spanned by the constraint:

$$\lambda_{eff} = \frac{1}{b-a} \sum_{a+1}^b \Lambda_i \quad (5.6)$$

2. Compute and apply the constraint step, using learning rate λ_{eff} ;
3. Update the learning rates of the affected poses, as in Eqn. 5.5.

After processing all of the constraints, the learning rates Λ_i are decreased according to a generalized harmonic progression, e.g.:

$$\Lambda'_i = \frac{\Lambda_i}{1 + \Lambda_i} \quad (5.7)$$

These rules guarantee the increasing monotonicity of Λ_i with i . I.e., at any given time step, $\Lambda_i \leq \Lambda_j$ for all $i < j$.

5.2 Partial Iterations

In the case of the batch algorithm, a “full iteration” includes a step for *every* constraint in the graph. The learning rate is controlled globally, and the graph converges more-or-less uniformly throughout the graph.

In the incremental algorithm that we have described, the learning rate is not uniform and different parts of the graph can be in dramatically different states of convergence. In fact, it is often the case that older parts of the graph have much

lower learning rates (and are closer to the minimum-error configuration) than newer parts, which are far from a minimum. This is because new constraints are generally added near the robot.

Suppose a robot explores two buildings, first A then B. While the robot is in B, the map for A is more-or-less unaffected, whereas the map for building B may be undergoing dramatic changes as new loop closures are identified.

A reasonable goal for an optimization algorithm is to reduce χ^2 error per CPU time used. Building B, in this simple example, represents low-hanging fruit. It would be highly advantageous if we spend more CPU time optimizing building B than A.

Suppose we have a graph with a maximum learning rate of λ_{max} . If we performed a full iteration (without adding any new constraints), the maximum learning rate in the graph would be reduced to $\lambda'_{max} = \lambda_{max}/(\lambda_{max} + 1)$. Now suppose that there is a constraint whose effective learning rate is less than λ'_{max} . Processing that constraint will not significantly improve the state estimate, since other constraints are taking larger steps (which will have a dominant effect on χ^2 error). That constraint will also not reduce the largest learning rate in the graph. Consequently, this constraint will have virtually no effect on the state estimate and can be skipped. In short, processing constraints whose effective learning rate is less than λ'_{max} is unproductive.

The following algorithm exploits this property to optimize only a recent subgraph of the pose graph. It does not actually compute the effective learning rate for each constraint; instead, it identifies constraints whose effective learning rate *must* be lower than λ'_{max} by virtue of not including any nodes whose learning rate is higher. As a result, some constraints that could be skipped will be processed anyway, but the resulting algorithm is simpler.

1. Look up the maximum learning rate in the graph, e.g., $\lambda_{max} = \Lambda_{nposes-1}$. If we performed a full iteration, the maximum learning rate after the iteration would be $\lambda'_{max} = \lambda_{max}/(1 + \lambda_{max})$.
2. Find the oldest pose p such that $\lambda_p \geq \lambda'_{max}$
3. Perform steps only on those constraints that involve a pose $\geq p$.
4. Set $\lambda_i = \lambda'_{max}$ for all $i \geq p$

In other words, this algorithm reduces the maximum learning rate in the graph by a full harmonic progression by computing and operating on only a subset of the graph's constraints. The procedure above conservatively identifies the set of constraints that should be considered.

In many cases, large reductions in learning rate can be achieved by considering only a handful of constraints. In the Freiburg data set, the technique is very effective, with an average of only 6.73% of the constraints updated at each iteration (see Fig. 5-2). Since computational cost is linear in the number of constraints processed, this yields a speed-up of almost 15x. The χ^2 error is naturally somewhat higher (see Fig. 5-6) in absolute terms, since older parts of the graph are not being fine-tuned when there are high learning-rate subgraphs. However, when the total amount of

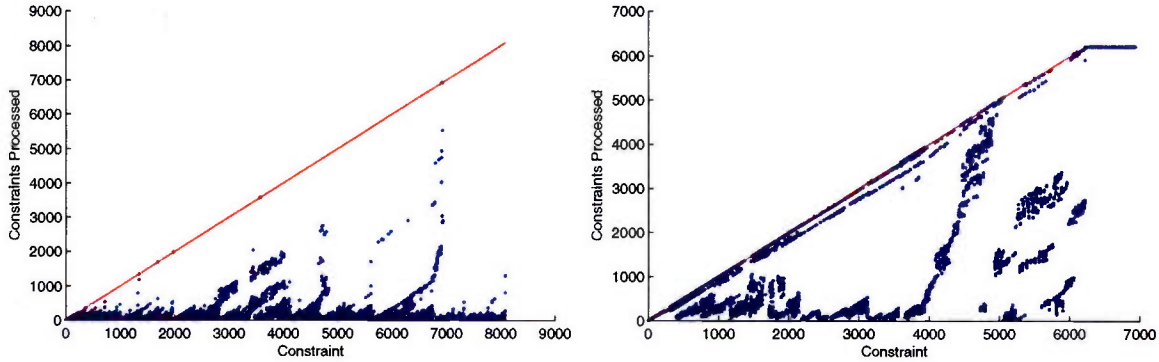


Figure 5-2: Partial Updates. Left: Freiburg, Right: Intel Research Center. Selective processing of constraints leads to large speed-ups. In the figure, the line indicates the total number of constraints in the graph; the points show the actual number of constraints that were processed. In the Intel dataset, more constraints tend to be processed since the robot repeatedly revisits the same areas, causing new constraints to disturb larger portions of the graph.

CPU time is taken into account, the selective method is substantially more efficient at reducing the χ^2 error.

The same approach on the Intel Research Center data set processes 59% of the constraints on average. The lower performance is due to the fact that the robot is orbiting the entire facility: many constraints perturb virtually the whole graph. Still, on this near worst-case data set, a significant speedup of 69% is achieved.

The approach outlined here is somewhat greedy: it attempts to reduce the worst-case learning rate to Λ'_{max} . This goal determines both the oldest pose p and thus the number of constraints that are necessary to perform the partial update. On very-well connected graphs with many large loop closures, the learning rate tends to be relatively flat. This leads to very small values of p and thus large numbers of constraints to be processed. It is possible that picking a larger Λ'_{max} (i.e., settling for less than a full harmonic progression) would result in a more economical update—this is an area of future work.

Similarly, if a robot knew that it was leaving an area and that it would not return for some time, it would be reasonable to expend additional iterations performing complete updates, lowering the learning rate throughout the graph and thus improving the quality of the map.

5.3 Learning Rate Data Structure

An obvious data structure for maintaining the values Λ_i is an array. The operations required by Eqn. 5.5 and 5.6 would run in $O(N)$ time for a graph with N poses. Since every constraint step requires updating the learning rates, this would increase the time complexity per constraint from $O(\log N)$ to $O(N)$.

An augmented binary tree can be used to implement both of these operations in $O(\log N)$ time. The learning rates for each pose correspond to the leaves of the tree. Each non-leaf node in the tree nominally stores the minimum, maximum, and sum of

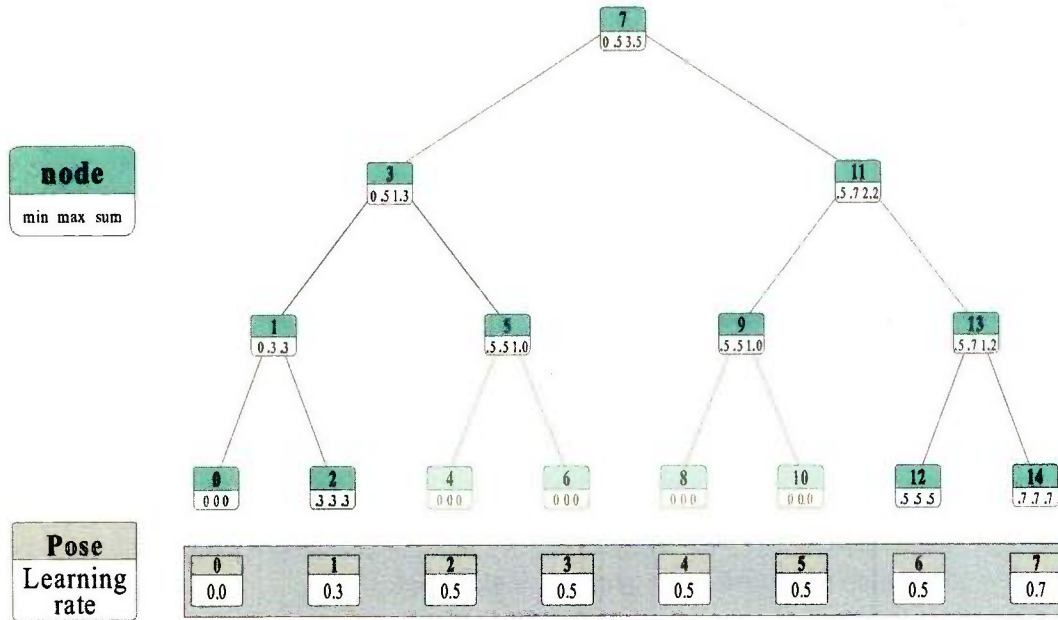


Figure 5-3: Learning-Rate Data Structure. Our learning-rate data structure, an augmented binary tree, allows all critical learning-rate functions to be implemented in $O(\log N)$ time. Each node in the tree tracks the minimum, maximum, and sum of all the elements beneath it; however, any parent can override all of its children by having $\text{min} = \text{max}$. The node indexing scheme allows the tree to be stored in-place in an array.

its children. However, a parent overrides the value of its children when its minimum field equals its maximum field. The values stored in the descendant nodes of such a parent are invalid.

The tree supports a variety of operations, each of which runs in $O(\log N)$ time. Since any node can override its children, operations begin at the root and descend towards the leaves; if they encounter a node at which $\text{min} == \text{max}$ (which signals that the parent is overriding the data stored in its descendants), the tree descent ends.

Most operations terminate at the first overriding node. (Note that leaves always have $\text{min} == \text{max}$ and thus can be considered “overriding” nodes, even though they have no children.) The procedure *setLowerLimitRecurse* is somewhat more complicated, since it must traverse all the way to a leaf, possibly passing an overriding node along the way. During its recursion, any overridden value is passed along and used to “patch up” the tree as it goes.

The *fixup* procedure recomputes a node’s min, max, and sum values from its children. The procedure *mean* can be trivially computed from *cumulativeSum*.

The node indexing scheme (see Fig. 5-3) permits computation of parent, child, sibling, and number of children using constant time arithmetic expressions. For example, the arithmetic mean of two child node ids gives the parent node id.

Aging the learning rates requires $O(N)$ time: the learning rates are extracted from the tree into an array (via a recursive traversal of the tree), aged, and then reinserted back into the tree. However, aging the learning rates only occurs after all the constraints have been updated. The amortized time is $O(N/M)$ for N nodes

Algorithm 8 `getValue(node)` : *Return the value v_{node}*

```

i = ROOT
loop
  if  $min_i == max_i$  then
    return  $min_i$ 
  end if
  i = descendTowardsNode(i, node)
end loop

```

Algorithm 9 `cumulativeSum(idx)` : *Compute $\sum_0^{idx} v_i$*

```

sum = 0
i = ROOT
loop
  if  $min_i == max_i$  then
    return  $sum + min_i * (idx - \text{leftMostDescendant}(i) + 1)$ 
  end if
  i = descendTowardsChild(i, idx)
  if isRightChild(i) then
     $sum = sum + sum_{\text{leftSibling}(i)}$ 
  end if
end loop

```

and M constraints; since M must increase *at least* linearly with N , this runtime is typically sub-linear.

5.4 Analysis

Our approach for updating the learning rates in order to perform incremental stochastic optimization has a number of appealing properties.

- When the robot travels to a new location, a new pose and a new constraint connecting that pose to the graph are created. Since the residual of this new constraint is zero (because there is only a single constraint to that node), no learning rate increase results.
- When a new loop closure is identified, but the residual is small (perhaps because the area is already well-mapped), little or no learning rate increase results.
- When a new loop closure is identified, and the residual is large (closing a very large loop for the first time, for example), a larger learning rate increase results.

Algorithm 10 `setUpperLimit(ulimit)` : *For all i , set $v'_i = \min(v_i, ulimit)$*

```

setUpperLimitRecurse(ROOT, ulimit)

```

Algorithm 11 setUpperLimitRecurse(i , $ulimit$)

```

if  $max_i \leq ulimit$  then
    return
end if
if  $min_i == max_i$  then
     $min_i = ulimit$ 
     $max_i = ulimit$ 
     $sum_i = ulimit * (\text{numberOfChildren}(i) + 1)$ 
end if
setUpperLimitRecurse(leftChild( $i$ ),  $ulimit$ )
setUpperLimitRecurse(rightChild( $i$ ),  $ulimit$ )
fixup( $i$ )

```

Algorithm 12 setLowerLimit(idx , $llimit$) : *For all $i \geq idx$, set $v'_i = \max(v_i, llimit)$*

```

setLowerLimitRecurse( $idx$ , ROOT,  $llimit$ , false, 0)

```

This is to be expected, since the positions of many nodes must be updated. However, any portion of the graph prior to the loop closure will be insulated from this increase.

On a graph with N poses and M constraints, memory usage of the algorithm is $O(N + M)$, the same as the batch algorithm. Runtime per constraint also remains $O(\log N)$.

The actual CPU requirements, as with the batch algorithm, are difficult to specify. The nature of the stochastic optimization provides no guarantees that the error will go down at each iteration. It is also the case that the amount of map optimization required for any particular application varies. Consequently, we cannot prescribe a specific number of iterations to compute after adding a constraint.

That said, the convergence of the algorithm is typically quite rapid, especially in correcting gross errors. Further, the fact that individual iterations are inexpensive affords users a great deal of granularity in controlling their CPU requirements versus their map quality requirements.

5.5 Results

We compared the runtime performance characteristics of our approach to that of LU decomposition (LU), the Extended Kalman Filter (EKF), and Gauss-Seidel Relaxation (GS). Our assumption is that the robot requires a full state estimate after every observation; this makes the performance of the EKF comparable to that of an information-form filter. Fig. 5-6 shows the cumulative computational time versus step (with one constraint added per step). For the Gauss-Seidel implementation, every state is iteratively updated once after each constraint. Our method is plotted twice: first, using all the constraints in the graph (“Whole”), and second, using only a subset of the constraints (“Partial”).

Algorithm 13 setLowerLimitRecurse(*idx*, *i*, *llimit*, *overriding*, *overrideValue*)

```

if overriding then
    mini = overrideValue
    maxi = overrideValue
    sumi = overrideValue * (numberOfChildren(i) + 1)
end if
if descendants of i are all < idx then
    return
end if
if mini == maxi and descendants of i are all ≥ idx then
    mini = llimit
    maxi = llimit
    sumi = llimit * (numberOfChildren(i) + 1)
    return
end if
if mini == maxi and not overriding then
    overriding = true
    overrideValue = mini
end if
setLowerLimitRecurse(idx, leftChild(i), llimit, overriding, overrideValue)
setLowerLimitRecurse(idx, rightChild(i), llimit, overriding, overrideValue)
fixup(i)

```

Our approach, especially with partial updates enabled, is significantly faster than any of the other methods. In terms of quality (as measured by χ^2 error), our approach produced somewhat worse maps. However, the difference is very subtle. When using the constraint selection algorithm (plots labeled “partial”), our algorithm is significantly faster than the other approaches. The CPU time and χ^2 results on the Freiburg data set are shown in Fig. 5-6. Similar behavior occurred on the Intel Research Center dataset (see Fig. 5-7).

A brute force LU decomposition at each iteration is the slowest method by far. The EKF’s $O(N^2)$ time complexity amounted to 588s, making it an unappealing approach. Gauss-Seidel took 110s, and our approach took 15.1s. The EKF, Gauss-Seidel (GS), and our method (not using the partial updates optimization) had similar χ^2 errors at each step, tracking the actual minimum quite well. When selectively processing constraints, our method’s χ^2 is naturally somewhat higher; subjectively, the quality of the maps was acceptable. We note that Gauss-Seidel relaxation does not perform this well in all cases, as was demonstrated in [48] and in Chapter 4.

The algorithm, even if all constraints must be processed, is very fast. Processing all 8000 constraints on the Freiburg graph with 906 poses required a total of 16.8ms. When using the partial updates algorithm, only a small fraction of these constraints need to be processed: the algorithm took an average of 1.1ms to add each of the final 10 constraints on the Freiburg graph: for each, it only needed to consider an average of 73 constraints. Several Freiburg maps are illustrated in Fig. 5-5, including

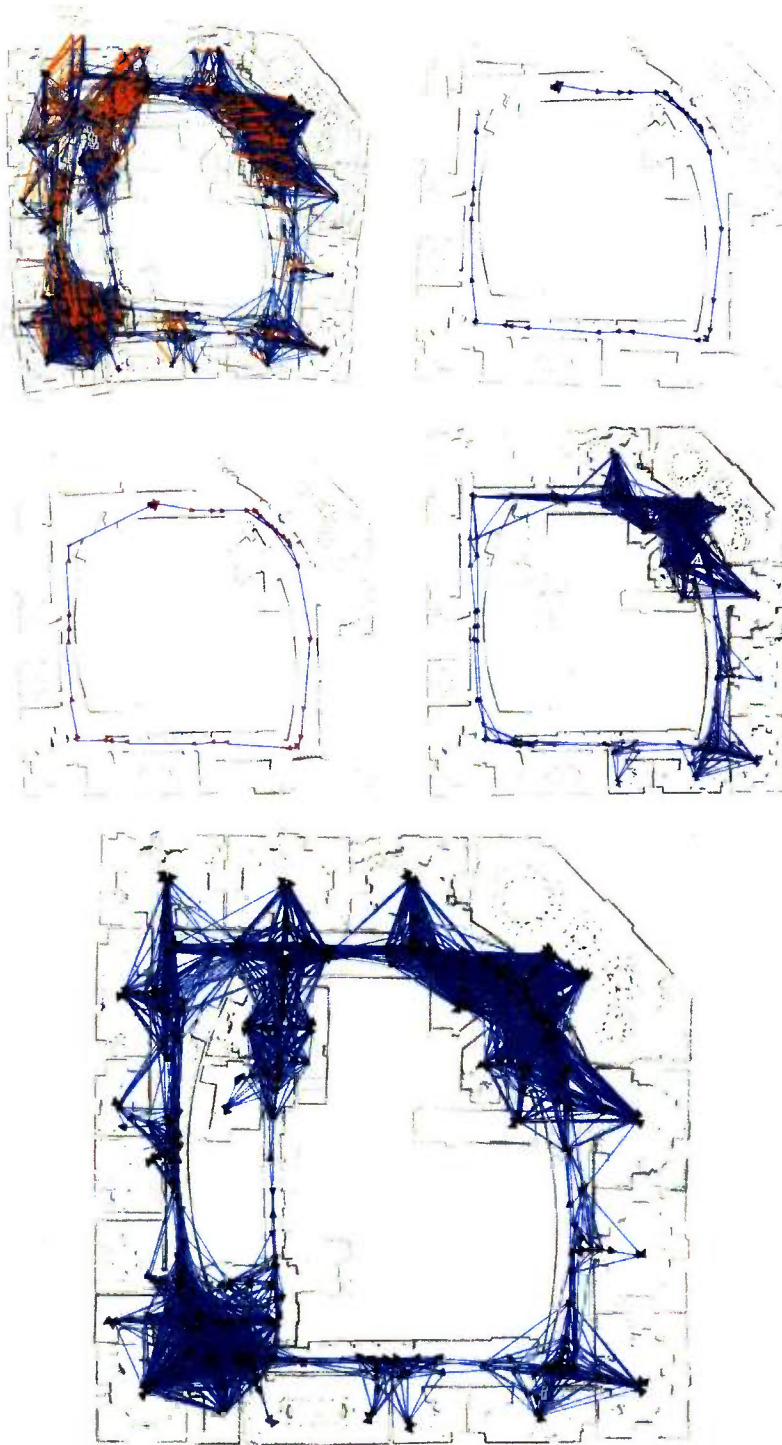


Figure 5-4: Intel Research Center. Top-left: open loop trajectory; top-right: just before closing the first loop, middle-left: just after closing the first loop; middle-right: about half way through the data set; bottom: the posterior map.

the learning rates (as a function of pose).

Putting these numbers in perspective, the Intel Research Center dataset (Fig. 5-4) represents 45 minutes of data; incorporating observations one at a time (and out-

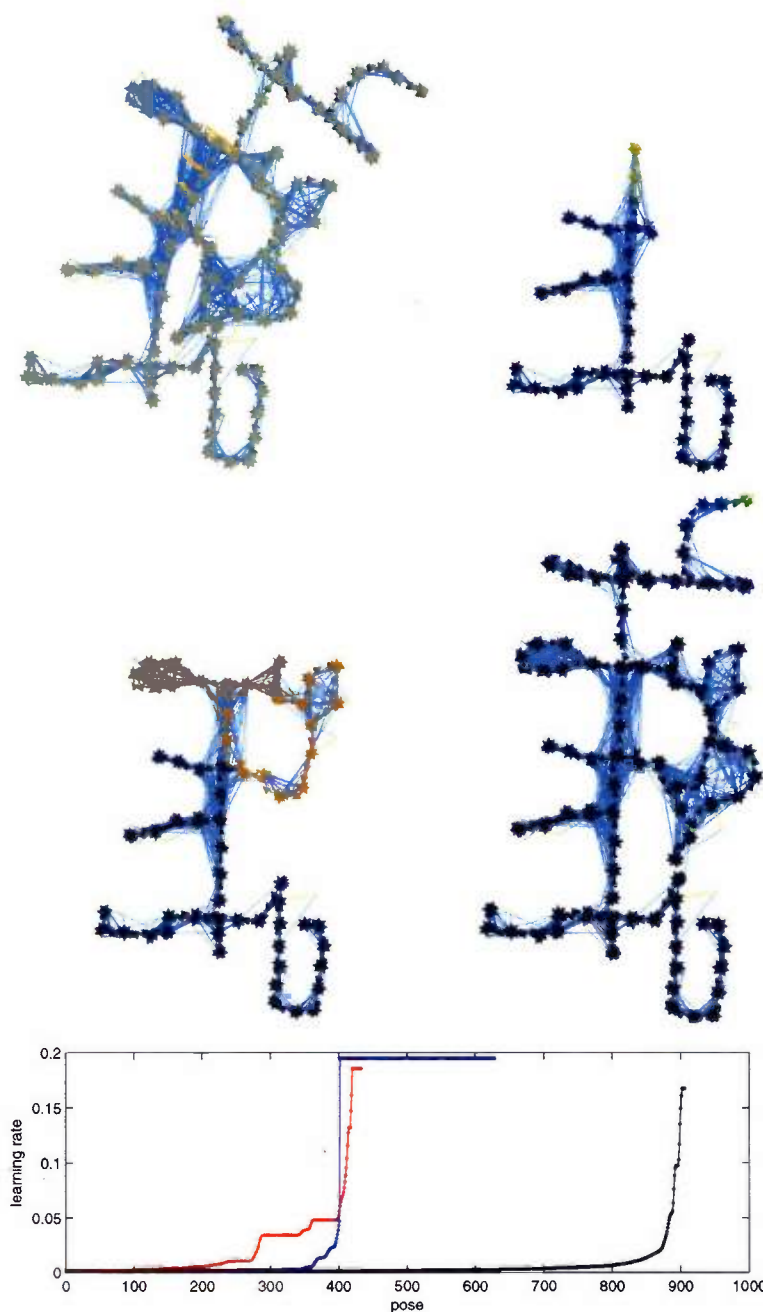


Figure 5-5: Incremental Processing of Freiburg dataset. The open-loop graph (top-left) is incrementally optimized; the state of the graph is shown at two intermediate configurations and the final configuration. The colors used in the maps indicate the learning rates Λ_i , which are also plotted on the bottom as a function of pose number; earlier parts of the graph are clearly insulated from learning rate increases affecting poses closer to the robot. When closing large loops (middle-left figure), the learning rate is increased over a larger portion of the graph.

putting a posterior map after every observation) required a total cumulative time of 27.9s with partial updates enabled, and 50.9s without.

Several maps from the Intel Research Center are shown in Fig. 5-4. The open-loop trajectory is shown, as well as several intermediate maps. In the first map, a

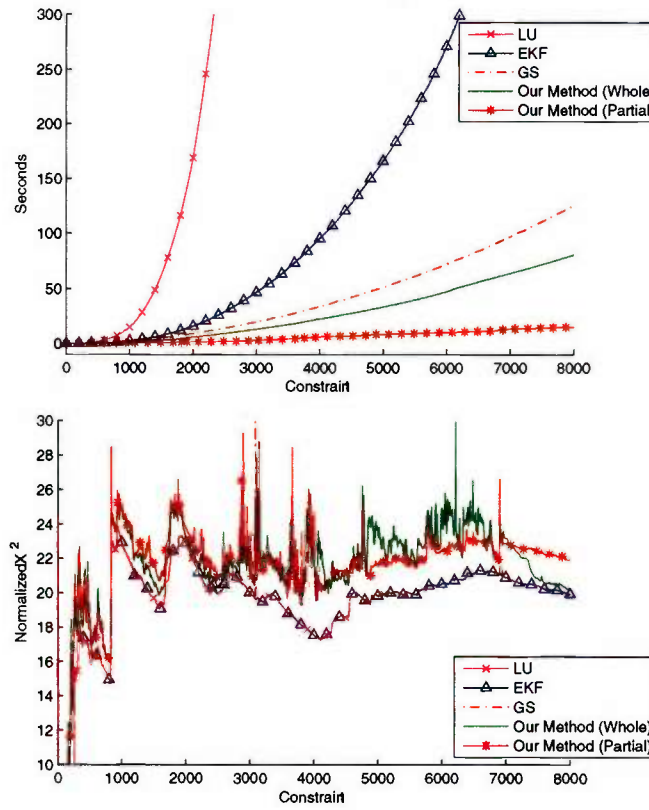


Figure 5-6: Cumulative Runtime and Error Comparisons, Freiburg dataset. Each constraint was added one at a time. EKF and LU computational time dwarfs the others. Our proposed method (with partial updates) is by far the fastest at 21 s; our method (without partial updates) beats out Gauss-Seidel relaxation. In terms of quality, LU, EKF, and Gauss-Seidel all produce nearly optimal results; our methods have marginally higher χ^2 error, as expected, but the maps are subjectively difficult to distinguish. (Partial: 15.5 s; Whole: 82.5 s; Gauss Seidel: 128.3 s, EKF: 650 s.)

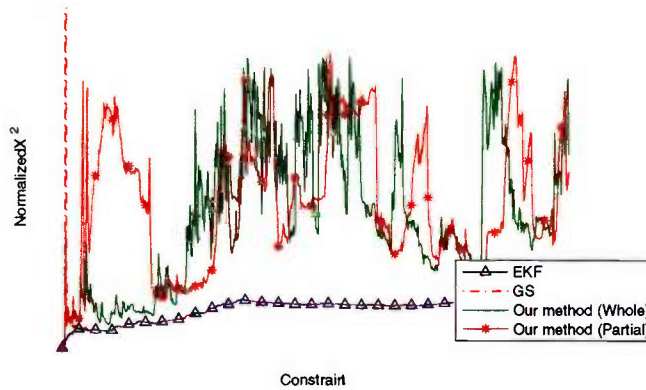


Figure 5-7: Error Comparisons, Intel Research Center. Our methods are able to stay relatively close to the nearly-optimal χ^2 error produced by the EKF and Gauss-Seidel, however, did so at a fraction of the run time. (Partial: 14.5 s, Whole: 45.4 s, Gauss-Seidel: 65.2 s, EKF: 132.3 s)

loop closure is just about to occur; prior to this, the learning rate is low everywhere. After the loop closure, the learning rate is high everywhere. The final map exhibits

sharp walls and virtually no feature doubling; the incremental algorithm matches the quality of the batch algorithm. We also validated our algorithm on countless synthetic datasets with consistently good results.

As with the batch algorithm, the optimization rapidly finds a solution *near* the global minimum, but once near the minimum, other algorithms can “fine tune” more efficiently. On a large synthetic data set, our method requires 24.18s for incremental processing; after a post-processing using 2.0s of Gauss-Seidel relaxation, the normalized χ^2 error drops to 1.18. This is far better than Gauss-Seidel can do on its own: it requires 168s to reduce the χ^2 error by the same amount.

5.6 Summary

We have presented an incremental map optimization algorithm based upon stochastic gradient descent. Our algorithm provides very fast updates, rapidly incorporating new data into the map estimate. We introduced the idea of spatially-dependent learning rates as a means of isolating different parts of the graph. This allows distant and less volatile parts of the graph to be largely unaffected by the addition of new constraints. We also described a data structure that allows the spatially-dependent learning rates to be efficiently updated over time.

Finally, we showed how to optimize only the subgraph that has the largest learning rate. This allows the χ^2 error of the graph to be rapidly decreased by focusing computational effort on the part of the graph that needs it most, making the algorithm even more computationally efficient.

CHAPTER 6

Conclusion

Building maps is a fundamental capability for robots operating in unknown or changing environments. In some instances, the map itself is a useful output of the robot, while in others, the map allows the robot to avoid obstacles and plan paths.

Map building is difficult. Environments are often cluttered and ambiguous, and are perceived only through imperfect sensor systems. Existing methods can construct maps in many environments, but when environments are large, or when the sensor data is ambiguous, those methods can become slow or produce incorrect answers.

Map building can be decomposed into two basic pieces, both of which are critical to success. The first is loop closing: recognizing when a robot has revisited a place it has been before. Each loop closure represents a constraint on the trajectory of the robot; the more over-constrained the trajectory is, the more accurately the trajectory—and thus the map—can be estimated. Loop closing is difficult, however, due to ambiguities in sensor data. Many environments can appear similar, making it hard to determine whether two sets of sensor readings correspond to the same or different places.

The second challenge is to compute the maximum-likelihood map once the loop closures have been detected. This optimization problem is difficult due to the size of the state vector, the non-linear nature of the constraints, and the poor quality of the initial state estimates.

6.1 Contributions

This thesis has described algorithmic improvements for both loop closing and map optimization. We show how relatively easy-to-generate loop-closure hypotheses can be filtered so that only correct loop-closure hypotheses remain. Our approach is based on the spectral properties of the pair-wise consistency matrix. A key feature of our approach is that the ratio of the first and second eigenvalues can be interpreted as a confidence metric, allowing perceptually ambiguous sensor data to be detected. In this thesis, we simply reject ambiguous hypothesis sets; a natural direction for future work would be to resolve ambiguous hypothesis sets by collecting additional data.

The second major contribution of this thesis is a new family of map optimization algorithms. These algorithms are based on stochastic gradient descent, which



Figure 6-1: The author with several indoor robots.

improves the state estimate by considering a single constraint at a time. Stochastic Gradient Descent has previously been used in machine learning (particularly in the training of neural networks), but we showed how the method could be adapted to the map optimization problem. Our resulting algorithm rapidly explores the state space, giving it two key advantages: it escapes local minima that trap existing methods, and it can rapidly find solutions near the global minimum. This thesis describes both a batch and online version of the algorithm and evaluates the algorithms on a variety of real and synthetic data sets.

6.2 Future Work

The methods described in this thesis allow robots to operate in larger and more perceptually ambiguous environments than before. Still, the map building problem has not been definitively solved. Both loop-closing and map optimization continue to offer difficult challenges.

Loop closing is primarily difficult due to perceptual ambiguity. Robots are unable to recognize places as well as a human, even though a robot's sensors are superior in some important respects. For example, planar lidars can easily measure distances with centimeter precision—far better than an unaided human. Humans make up for this, at least in part, by having a better understanding of the environment. A human might recognize an office copier room by noting the presence of a copy machine, whereas to a lidar, a copier is essentially indistinguishable from any other piece of furniture. In the short term, there is room for improvement in both low-level sensor

processing (like feature detectors) as well as higher-level processing (like the loop-closing method described in Chapter 3).

Current optimization methods, augmented by the techniques described in this thesis, can often successfully solve map optimization problems. The problem can be made arbitrarily harder, of course, by increasing the size of the problem: more nodes and more sensor observations will tend to bog down any method. However, with several fast methods now available, *robustness* becomes a distinguishing factor. If an algorithm arrives at the wrong answer (by getting stuck in a local minimum, for example), the speed of that algorithm is largely irrelevant.

Problems in which there is a great deal of noise make the optimization harder, but the *type* of observation also has an effect on difficulty. This thesis, for example, focused on edges that fully constrain the position of two nodes. Range-only and bearing-only observations do not fully constrain the position of two nodes: they constrain the position of one node to lie on either a circle or ray (respectively).

Range-only observations have proven to be fairly easy to incorporate [47, 49]. Bearing-only observations, in contrast, are more difficult. Bearing-only measurements, such as those naturally obtained from monocular cameras, exhibit a singularity: if the predicted position of a landmark is near the robot, small changes in the robot's position cause large changes in the predicted bearing to the landmark. In the worst case, if the robot happens to be on top of the landmark's predicted position, the predicted bearing to the landmark is undefined. While these effects can be mitigated to some degree (by skipping observations that are near a singularity, for example), the likelihood of the optimization failing increase. Failure modes include both local minima and divergence.

It is important that map building performance continue to improve. Better mapping algorithms will allow robots to operate in environments that are currently too complex. For example, autonomous cars have the potential to reduce the number of traffic fatalities— with humans at the wheel, more than 40,000 die every year on roadways in the United States [1]. Other potential applications, from charting abandoned (and hazardous) mines to helping the elderly move about their homes, all illustrate the practical usefulness of robotic map building.

APPENDIX A

Sensor Processing

A.1 Overview

This appendix describes some of the implementation details of the sensor processing methods used in this thesis. The central contributions of this thesis are largely sensor agnostic, but an understanding of the underlying systems can build an appreciation for how complexities that arise from real data.

The bulk of this thesis uses two sensor modalities: vision and laser scanners (lidars). This appendix discusses each in turn.

A.2 Vision Processing

A.2.1 Data Collection Platform

Our vision data set was collected by MIT's DARPA Grand Challenge vehicle (see Fig. A-1) during an autonomous run of the National Qualifying Event. Five calibrated cameras recorded a nearly 360-degree field of view. For these experiments, each camera's data stream was reduced to 376×240 at 10 fps. These cameras were essentially monocular since their fields of view did not significantly overlap. In other words, the range to objects in the environment could not be determined from a single set of camera images.

The motion between poses is estimated by the inertial measurement unit (IMU) attached to the robot. This IMU, coupled with a three-antenna GPS system, provides a good ground truth estimate. For many of our experiments, we purposefully degraded the IMU data in order to mimic the lower-quality odometry data available on a typical vehicle.



Figure A-1: Talos. Our vision test-bed was MIT’s DARPA Grand Challenge vehicle. The vehicle has five cameras (two forward, left, right, rear), and a high-quality IMU/GPS system for both dead-reckoning and ground-truthing.

A.2.2 Feature Detection

We applied Lowe’s Scale-Invariant Feature Transform (SIFT) [38, 39] detector to our images, resulting in a large number of SIFT detections (see Fig. A-2)¹. SIFT features occur at so-called *interest points*, which corresponding to extrema of the difference-of-Gaussian function applied over a multi-resolution image pyramid. Each feature is described by a 128-dimensional vector that allows features to be reliably associated over time.

Using known camera calibrations, the ray on which the SIFT feature lies can be computed.

A.2.3 Feature Tracking and Landmark Initialization

Unlike some previous SIFT-based feature tracking and mapping work [59, 60], we did not use stereo/trinocular vision to determine the range to detected SIFT features: the minimal overlap of our cameras’ fields-of-view precluded this. This means that individual observations constrain the position of a feature to a ray—the distance to that feature cannot be determined from a single observation. Davison accounted for unknown range by tracking features with a particle filter whose particles were distributed all along the ray [9]. Our approach is to triangulate the position of a feature by using several observations of the same collected while the robot moves².

Whenever a camera frame arrives, a new set of SIFT features are extracted. These features are matched against a set of “tracks”: each track maintains a “reference” SIFT descriptor and a list of observations matching that descriptor. Each observation includes both a vehicle position and the bearing to the SIFT feature.

¹My thanks to Albert Huang for generating the SIFT detections and for guidance on how to process them.

²Initializing features based on bearing-only observations shares similarities with using range-only observations [49].

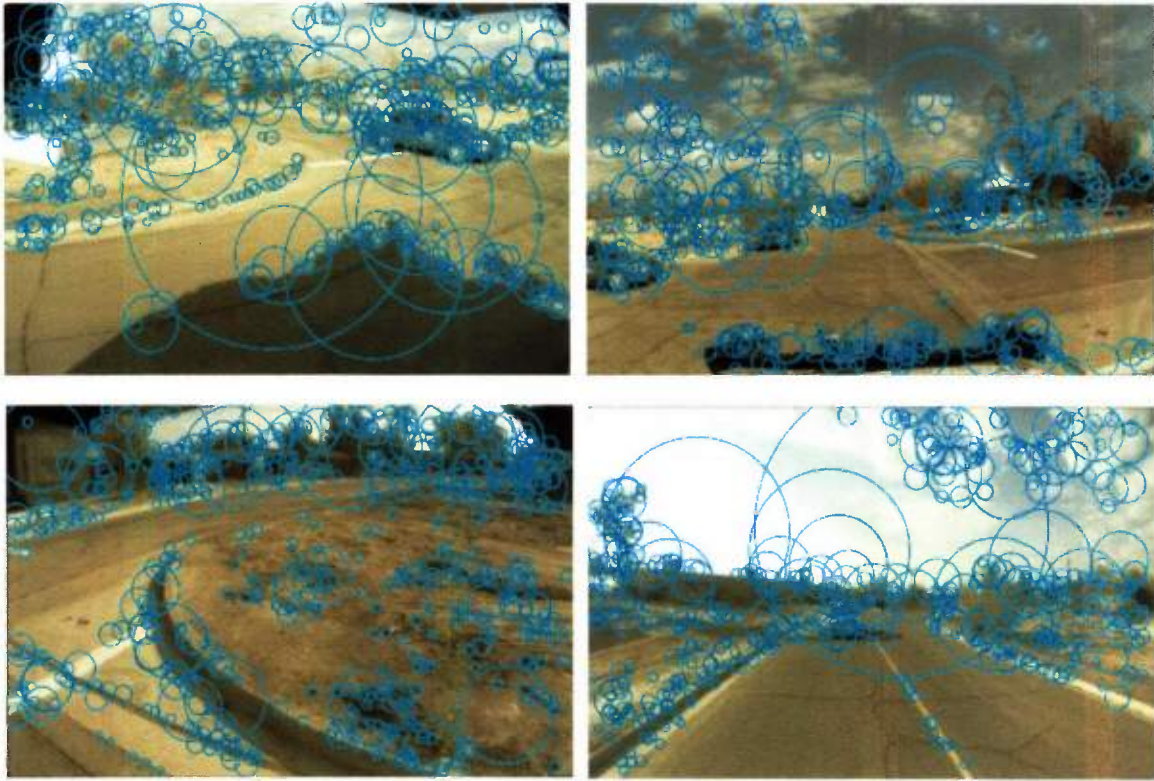


Figure A-2: Extracted SIFT features. Clockwise: left camera, forward camera, right camera, rear camera. Cyan circles represent extracted SIFT features: the size of the circle encodes the scale of the feature.

Observations are matched to tracks using the SIFT descriptor alone. We use commonly-accepted matching parameters: in order for a match to be valid, the mean squared error corresponding to the best match must be less than 60% of the mean squared error of the next best match. Additionally, if two newly detected features match to the same track, both matches are discarded. (This is similar to “marriage”, in which two descriptors a and b match only if a is the best possible match for b , and b is the best possible match for a . Our faster rule is “monogamous”, in that it requires that a is the only feature whose best match is b .)

The set of observations for each track are periodically tested to see if they can be used to localized the landmark. The position of the landmark is estimated by taking the pair-wise intersection of every observation ray, with a maximum distance set to 100m. For N observations, this yields up to $N(N - 1)/2$ points. The mean and covariance of these points is used as the landmark position estimate and its uncertainty. If the landmark’s covariance is under a threshold (2 m in both $\hat{x}$ and $\hat{y}$), and the landmark has been observed over at least 5 degrees of motion, a landmark is initialized (see Fig. A-3). When a landmark is initialized, the track is reset so that the landmark can be independently observed and estimated again. The result is that the same landmarks are repeatedly reobserved with respect to the car’s current position.

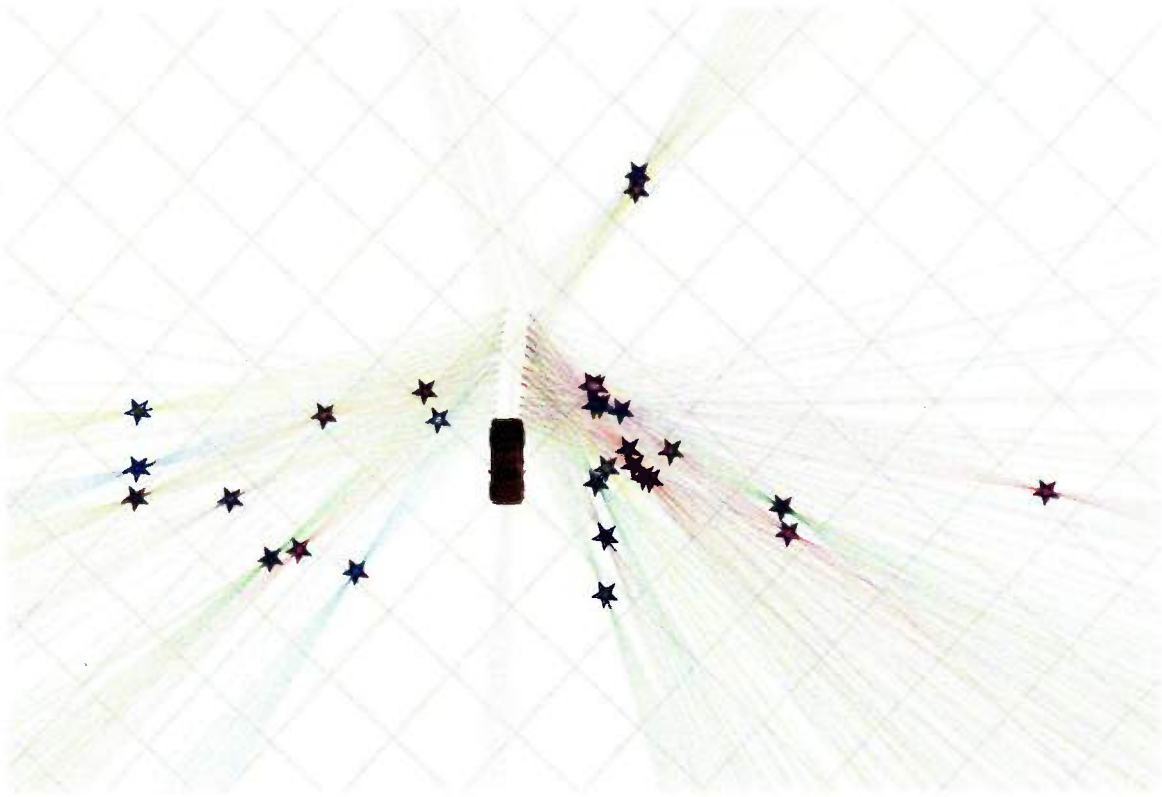


Figure A-3: SIFT Landmark Initialization. Landmarks are computed by tracking SIFT features over short motions of the robot: when the intersection of the bearings converges near a single point, a landmark is initialized.

A.2.4 Landmark matching

When initializing landmarks, a fairly restrictive set of parameters are used to ensure that features are not erroneously associated. When closing loops, however, the robot may be viewing the features from a different angle, distance, and under different lighting conditions. While SIFT features are somewhat robust to these variations, a conservative set of matching requirements (like those used in tracking) can result in very few matches. Instead, we independently compute the best match for each feature, without any additional restrictions such as “marriage” or “monogamy”.

Suppose that a robot has initialized a set of landmarks A from one pose, and a different set of landmarks B from a different pose. Using this greedy matching process, we find the best matches between these two sets by comparing the SIFT descriptors of each landmark. Suppose the indices of the landmarks A and B are assigned such that (A_i, B_i) represent a matched pair.

Given two correspondences, e.g. $(A_i, B_i), (A_j, B_j)$, we can compute a rigid-body transformation that would approximately align the landmarks. For example, one correspondence could be used to establish the translational component of the rigid-body transformation, while the second correspondence could be used to compute the rotational component. (In practice, a lower-error rigid-body transformation can be found using Horn’s algorithm [28]).

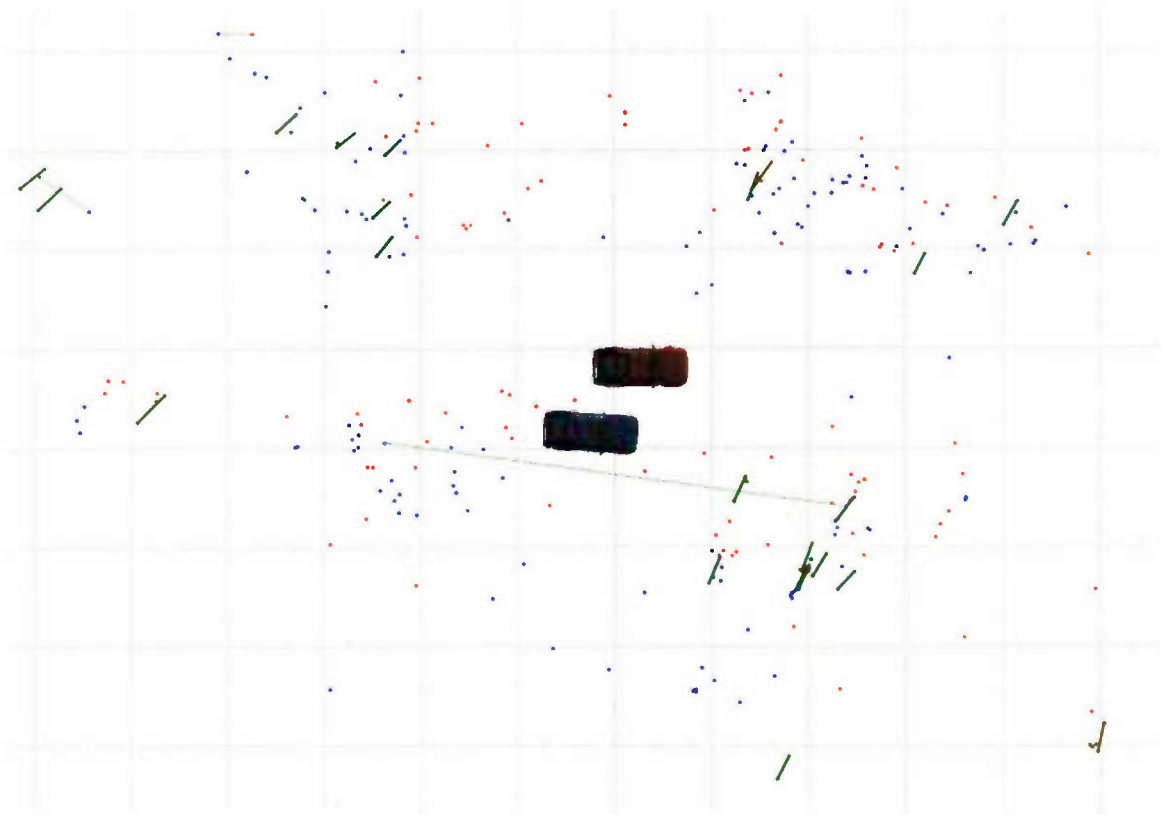


Figure A-4: SIFT Landmark Matching. The SIFT features observed from two different visits to the same place are shown: blue dots correspond to SIFT features observed from the blue pose, red dots correspond to SIFT features observed from the red pose. Matched SIFT features are connected by lines: heavier lines indicate matches that were accepted by the RANSAC validation.

The best rigid-body transformation is then found via RANSAC (see Fig. A-4). Each putative rigid-body transformation is scored by counting the number of features B_i that, when projected, lie near their counterpart A_i . We used a distance threshold of 1 m in our implementation. The rigid-body transformation with the largest score is output as a candidate loop closure hypothesis, which can be filtered using the techniques in Chapter 3.

A.3 Laser Scan Processing

Laser scanners measure the range and distance to other objects in the environment. A typical SICK laser scanner (see Fig. A-5) captures 180 points at one degree intervals with a range accurate of a few centimeters. Bearings are accurate to a fraction of a degree.

We use laser scanners (often called “lidars”) in two different ways. First, we use lidars to produce an initial estimate of the robot’s trajectory by *scan matching* consecutive scans. Our approach is to search for a rigid-body transform that aligns two scans by using a brute-force search (i.e., by considering a range of translations



Figure A-5: SICK Laser Scanner on a small indoor robot.

and rotations). This incremental scan matching generally produces a much better initial estimate of the robot trajectory than the robot's wheel-derived odometry.

We also use laser scanners as a means of generating loop closure hypotheses (see Chapter 3). In this case, we attempt to determine the alignment of two scans taken at very different times: the initial uncertainty is much larger than in the incremental scan matching case, making a brute-force search impractical. Our system uses a feature detector in order to generate rough candidate alignments; these candidate alignments are then refined by a much smaller brute-force search.

In this section, we describe our both aspects of our lidar data processing. First, however, we begin by describing some of the challenges inherent in lidar data.

A.3.1 Challenges

Laser data is often thought to be a panacea, providing precise and accurate data that makes map building trivial. While laser range finders produce accurate metrical descriptions of the environment, they also present a number of challenges.

The most significant limitation of laser scanners is that they measure only the *shape* of the environment (and even that, only on a single plane). In many common cases, such as in office buildings, matching laser scans can be ambiguous. This is due to the fact that the environments are composed of objects that are hard to distinguish from laser data alone. For example, in an office environment, virtually all rooms contain a similar-looking set of walls and corners.

Lidar data, often consisting of only a single cross-sectional slice of the environment, often lacks cues useful for navigation. In a long straight hallway, for example, longitudinal motion can be difficult to detect since the cross-sectional profile of the environment does not change (see Fig. A-6).

It is often difficult to exploit the subtle cues that *are* present, since algorithms must be robust to small changes in the environment. Algorithms must continue to work, for example, even if a chair has been moved or if a human is walking down a

hallway with the robot.

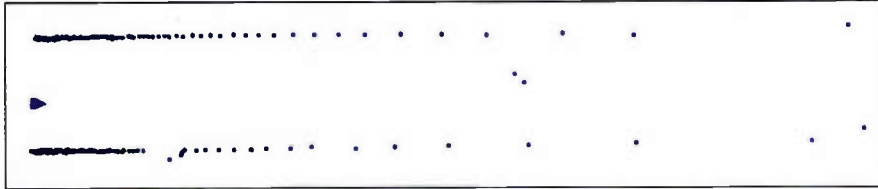


Figure A-6: Ambiguous long corridor. As a robot travels down a long corridor, the data from the lidar changes very little; this makes it difficult for the lidar to estimate the robot's motion from the lidar data. This data also includes a pedestrian (two dots near the center of the figure).

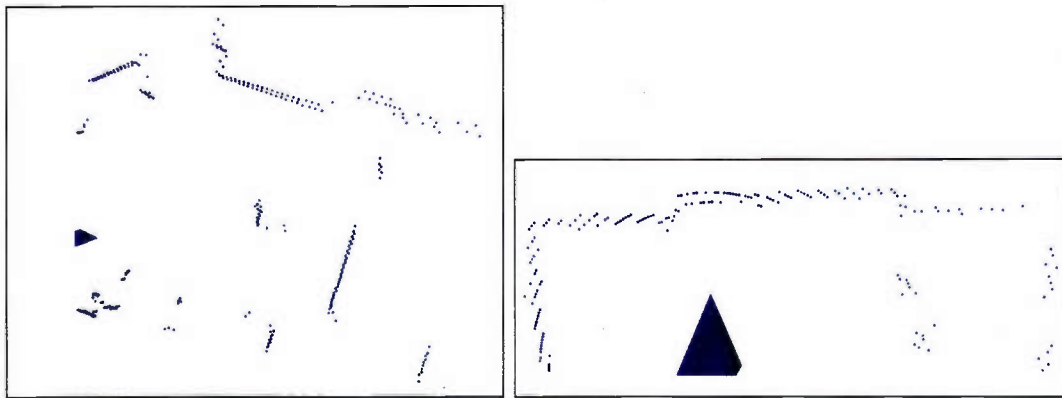


Figure A-7: Laser challenges. Left: an interlaced scan providing 0.5 degree resolution is combined from two separate scans taken 13.3ms apart. When the robot is moving, visible artifacts result. Right: coarse range quantization creates false “scaloped” textures.

Planar laser scanners are also difficult to use in cluttered outdoor scenes. Since they scan in only a single plane, any pitching or rolling of the sensor results in different cross-sectional views of the environment. Not only can this make the appearance of objects change (since they are being sampled at different heights above the ground), it introduces range distortions when the scanning plane is no longer horizontal.

Laser scanners do not directly capture the shape of the room: instead, they sample the shape at discrete points. The position of these sample points varies as the robot moves around, meaning that the set of points from a scan taken at time $t + 1$ may not all be near the points taken from a scan at time t . Consequently, it is necessary to infer the actual contours of an environment so that the “closeness” of two scans can be meaningfully computed. Since laser scanners sample points at a fixed angular intervals, the *spatial* resolution decreases rapidly with range: this makes any inferences about the geometry of the environment increasingly error-prone with range. Partially transparent surfaces complicate this inference (see Fig. A-8).

One common error made with laser scanners manufactured by SICK is to configure them to produce *interleaved* sub-degree resolution scans. A typical SICK sensor can only produce samples at one degree angular spacing; in order to produce a scan with quarter degree spacing, four scans (each with an angular spacing of one degree) are performed sequentially. Each scan is offset a quarter degree from the previous scan,



Figure A-8: Partially transparent surfaces. Inferring the correct geometry of a scene is complicated by partially transparent surfaces. Left: the laser data can partially see through the guard rail, detecting both the guard rail and wall on the opposite side. Right: a camera's view of the same environment.

and the results from all four scans are then interleaved. Motion of the robot between these scans, especially rotation, can cause artifacts in the interleaved scan (see Fig. A-7). Instead, it is better to operate SICK sensors in “dithered mode”: each successive scan is still offset by a quarter degree, but each scan is individually output. This allows the motion of the robot to be incorporated for each individual scan, avoiding interlacing artifacts. This is the ideal mode for SICKs, as it increases the effective resolution of the sensor when the robot is stationary, but does not create artifacts if it is moving.

A.3.2 Scan Matching

Scan matching is the process of finding a rigid-body transformation that best aligns two scans. Incremental scan matching is the special case of examining temporally-consecutive scans: it is used to enhance or replace conventional odometry sensors.

In Fig. A-9, two scans acquired approximately 133ms apart are shown in an poorly-aligned configuration (the alignment corresponds to the odometry-derived estimate).

Scan matching is an optimization problem with three free parameters corresponding to the parameters of a rigid-body transformation. For each rigid-body transformation, a cost function is computed that indicates the quality of the alignment. An optimization algorithm then attempts to find the minimum-cost alignment. This section describes these steps.

Contour Extraction

As previously discussed, each lidar scan samples the shape of the environment at a set of discrete points. When matching two scans, the point samples do not generally correspond to the same points in the environment. Consequently, matching points directly can produce poor results. Instead, we explicitly estimate the shape of the

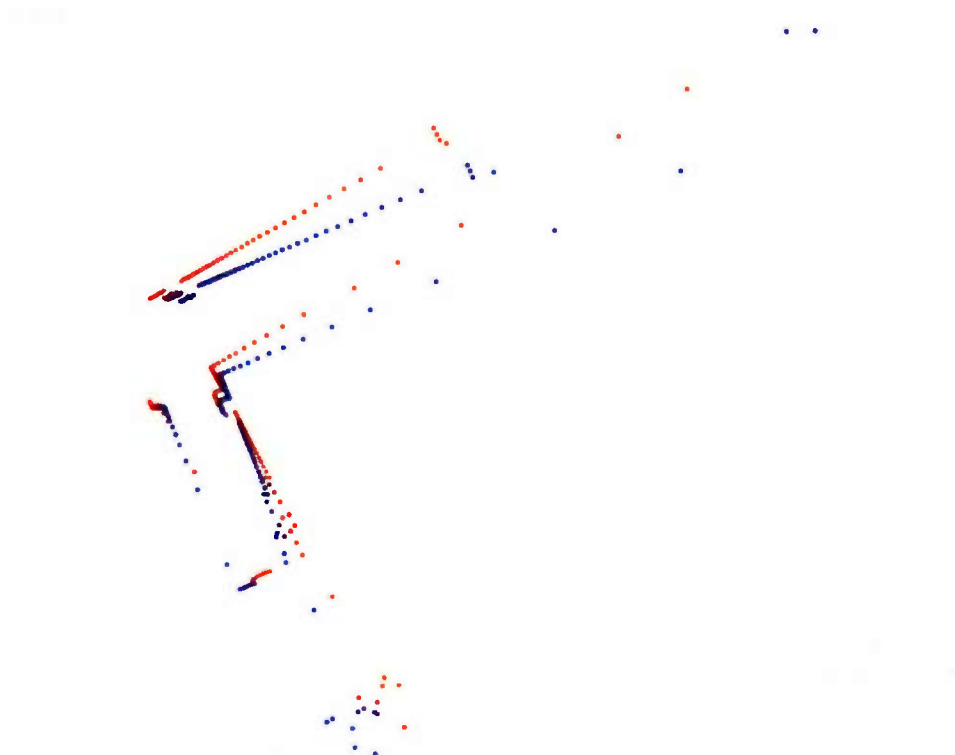


Figure A-9: Scan Matching. Two scans taken 133ms apart are poorly aligned. Scan-matching is the process of finding a rigid-body transformation that improves the alignment of two scans. This alignment corresponds to the motion of the robot.

environment that led to a set of samples. Given a shape estimate, we can then test if a second set of point samples is consistent with that shape. We represent the shape of the environment with a set of poly-lines we call contours. This section describes our contour extraction algorithm.

Initially, each point forms a zero-length contour. Contours are agglomeratively merged by repeatedly finding the pair of contours whose endpoints are the closest. If these two endpoints are within a threshold distance, the contours are merged. This process repeats until there are no more nearby endpoints. This algorithm preferentially connects nearby points, and produces reasonable contours even in environments with partially-transparent surfaces (as in Fig. A-8).

Our implementation maintains a MinHeap [55] of candidate contour merges, sorted according to the distance between their endpoints. We repeatedly extract the next best candidate from the MinHeap: if the endpoints of the candidate's contours have not already been merged with other contours, then the candidate's contours are merged. The algorithm is very fast: on a typical scan with 180 points, it takes about 0.2ms to extract the contours.

Computing the Probability of an Alignment

We wish to determine the cost associated with a particular alignment of one laser scan with another. The noise model of the laser scanner involves uncertainty in

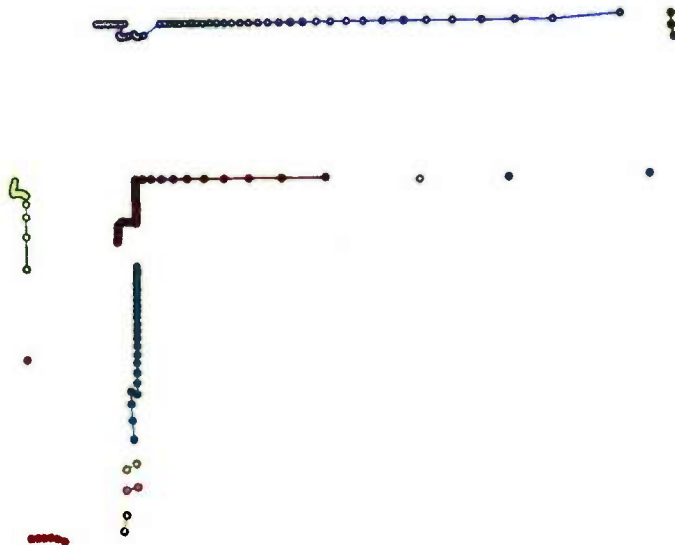


Figure A-10: Contour Extraction. A laser scanner measures only a discrete set of points; it is necessary to infer the *contours* of the surface being sampled as shown here.

both range and bearing. Range is typically more uncertain than bearing, and so the probability distributions resemble radially-oriented ellipses. In other words, the probability distribution of an observed point is dependent upon the position of that point relative to the scanner. In practice, we approximate the distribution as a circle (regardless of its position). Specifically, given a laser scan point (x, y) and the distance d to the nearest contour C , we approximate the probability of a single lidar point as:

$$P(x, y \mid C) \propto e^{-d^2/\sigma^2} \quad (\text{A.1})$$

The variance parameter σ is determined by the sensor’s noise characteristics. This approximation yields good results, and allows a lookup table for $P(x, y)$ to be easily constructed. This lookup table greatly accelerates scan matching since each scan matching operation evaluates the probability for a large number of rotations and translations.

We compute a look-up table by rendering each of the extracted contours with an intensity pattern corresponding to the log-likelihood of Eqn. A.1. The log-likelihood of a scan with N points is proportional to the sum of the N lookups.

Laser scanners are strictly line-of-sight, and are susceptible to occlusions. In general, better scan matching results can be obtained by building the cost lookup table using *several* previous scans. This increases the completeness of the lookup table (and thus the likelihood of getting a good match) at the expense of potentially introducing errors into the lookup table if the alignment of those earlier scans is incorrect.

While Carmen’s “Vasco” scan matcher[45] assumes a good initial estimate and relies on gradient descent to achieve a good solution, our approach is more brute-force, computing the cost function over a wide range of translations and rotations. The cost surface can be very complicated, even in simple environments as seen in

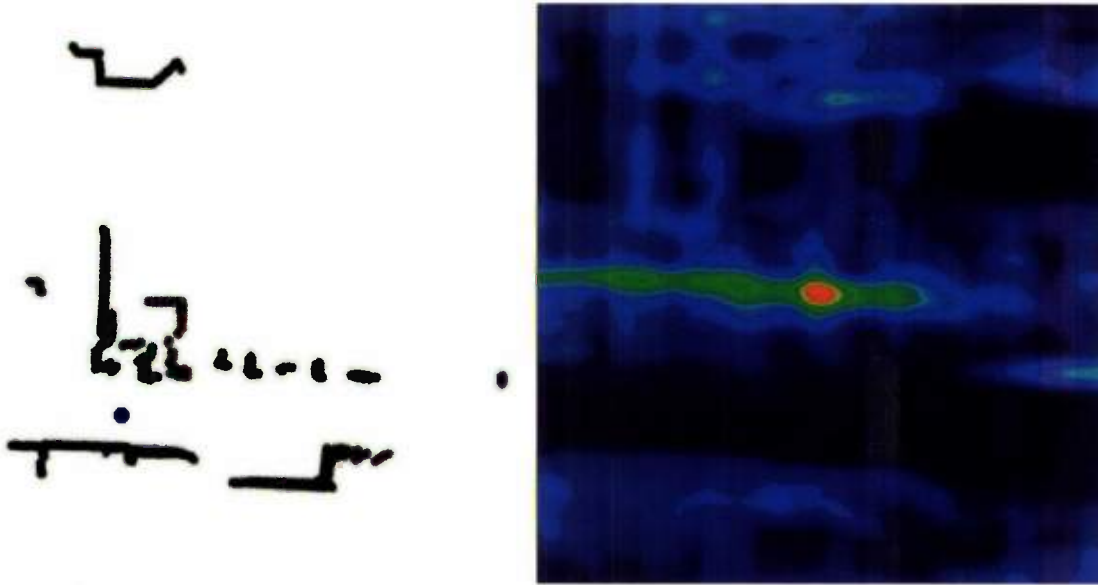


Figure A-11: Scan Matching Cost Lookup Table. The cost function is precomputed into a 2D lookup table (left) allowing the probability of a particular alignment to be computed very quickly. The cost surface for an incoming scan, parameterized by the rigid-body transformation, can very complicated (right). This cost surface is one slice of a 3-dimensional surface, corresponding to pure translations given the correct rotation.

Fig. A-11. This complexity can cause gradient-descent methods to get stuck in local minima. The primary motivation behind our brute-force approach is to improve the scan matcher’s ability to find better solutions. This robustness comes at the cost of increased computational complexity, but our method can still run much faster than real-time on modern hardware if it is implemented carefully.

Scan Matching Optimization

Our brute-force approach to scan matching is robust, but is also fairly computationally expensive. This section describes an optimization that helps make the algorithm run faster than real time.

Our goal is to compute the cost function for a range of rigid-body transformations, outputting the rigid-body transformation with the minimum cost. Each rigid-body transformation corresponds to a point in a three-dimensional parameter space; our algorithm will evaluate the cost function on a 3D grid of points in that parameter space.

A straight-forward implementation would independently consider each rigid-body transformation, projecting each lidar point in order to find the appropriate cell in the lookup table and summing the resulting values. However, the computational cost of repeatedly projecting the lidar points can be substantial. Our optimization reduces the number of projections by simultaneously considering all the translations for a particular rotation.

Let us consider one slice of the 3D grid of rigid-body transformations: this 2D slice corresponds to the set of possible translations for a fixed rotation. For a single lidar

point, the cost of a translation is computed by projecting the point into the lookup map and sampling the value. Importantly, translating the lidar point is equivalent to translating the lookup table by the same amount. This means that the 2D cost map for that point can be obtained by simply copying data out a rectangular window of the lookup table. This operation is particularly fast if the 2D search grid is the same resolution as the lookup table.

The cost for all of the translations can be determined as a group by summing the rectangular windows corresponding to each point. The best translation for that particular rotation can then be determined. Our scan-matching algorithm repeats this process for each candidate rotation.

Naturally, there are an infinite number of possible translations and rotations. A prior limits the range of rotations and translations that we search, while a quality parameter determines how densely we sample the cost function within that region. The fact that the cost function is the sum of a large number of lookups (one for each lidar point) helps to reduce the effects of quantization noise stemming from the finite resolution of the lookup table. Our experiments use a lookup table with 2.5 cm resolution, but the accuracy is typically *better* than this.

A.3.3 Loop Closing Hypothesis Generation

In contrast to the incremental scan-matching case, the positional uncertainty between two scans when closing a loop can be very large. Gradient-Descent methods are more prone to failure since their initial estimate may be far from the optimal answer. With the scan matching method described above, large uncertainties require large (and slow) searches.

Instead, we extract features from the laser scans and match the features between scans. If a feature match can be found, an approximate alignment between the scans can be computed. This rough solution greatly reduces the uncertainty, making it possible to use scan matching to refine the match.

Feature Extraction

Line segments are easy to detect and are common in most indoor environments. Even in outdoor environments, lines can be fit around non-linear features like trees and shrubs: despite their poorer quality-of-fit, these lines can still be used to bootstrap the scan-matcher in order to compute good loop-closure hypotheses.

Line fitting is performed on one contour at a time: in cases of semi-transparent objects like those in Fig. A-8, this allows us to fit lines to each guard rail individually.

Our line extraction algorithm is similar to our contour extraction method. A set of lines are initialized for each pair of adjacent points within a contour (we constrain our line features to include only contiguous sets of points from the contours). Nearby line features are then repeatedly merged until the quality-of-fit of the merged lines exceeds a threshold. When merging two line features, the resulting line represents the optimal line fit of all the points associated with the input lines. Fortunately, the optimal line can be computed based on a set of sufficient statistics of constant

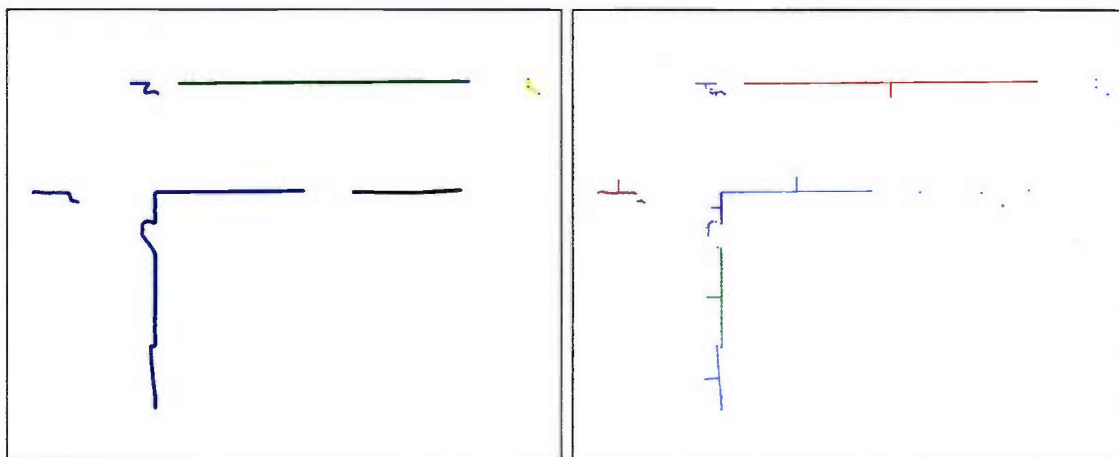


Figure A-12: Feature Extraction. Beginning with contours, lines are agglomeratively clustered until the mean squared error of the line exceeds a threshold. Left: the input contours. Right: the extracted line features (normals have been added for visualization purposes).

size [40, 42]: this makes the computational complexity of merging two lines $O(1)$ instead of linear in the number of points.

As with contour extraction, we use a MinHeap to keep track of the relative merits of all possible line merges. Given that the computational complexity of inserts and removals for a MinHeap is $O(\log N)$, and that there can be (at most) $N - 1$ merges, the runtime is $O(N \log N)$. On a 2.4GHz Intel Core2 system, line extraction takes about 0.8ms for a 180 point scan: i.e., its runtime is negligible.

Feature Matching and Refinement

Suppose that we extract line features from two different lidar scans. Given two lines from each scan (and correspondences between those lines), a rigid-body transformation can be computed that approximately aligns those lines. The intersection of the two lines serves as a common anchor point (determining the translational component), and the angles of the lines with respect to that intersection allow the rotational component to be determined.³

However, we do not initially know the correspondences. Instead, we select the K best lines from each scan and compute the rigid-body transformation that results from all possible correspondences amongst those lines. The number of possible pairs of correspondences grows as $O(K^4)$, making it necessary to keep K relatively small. The best lines are selected based on a heuristic incorporating the length of the line and the number of points on the line. This heuristic favors lines resulting from long and straight walls. In our experiments, we set K to 10.

The cost of each rigid-body transformation is computed using the lookup table; the lowest-cost transformation is then refined using a brute-force scan-matching operation. The line-based alignment greatly reduces the search space, making the run-time

³In 3D, there is an additional “flip” ambiguity, but in 2D, we can force a unique solution by forbidding reflections.

cost manageable. The result of this refinement becomes a loop closure “hypothesis”, as discussed in Chapter 3.

A.4 Summary

This appendix described our low-level sensor processing algorithms for both cameras and laser scanners. The goal of these algorithms is to compute rigid-body transformations that relate the positions of the robot at different times.

Sensor processing algorithms tend to be difficult to design and implement, since they must simultaneously minimize both their error rate and their runtime costs. The methods described in this appendix are computationally efficient and generally produce good results (i.e., have a low error rate). However, they can still produce incorrect rigid-body transformations; the methods of Chapter 3 show how the error rate can be reduced further.

Bibliography

- [1] National Highway Traffic Safety Administration. Fatality analysis reporting system (FARS), 2007.
- [2] Tim Bailey. *Mobile Robot Localisation and Mapping in Extensive Outdoor Environments*. PhD thesis, Australian Centre for Field Robotics, University of Sydney, August 2002.
- [3] Y. Bar-Shalom, X. Rong Li, and T. Kirubarajan. *Estimation with Applications to Tracking and Navigation*. John Wiley & Sons, Inc., New York, 2001.
- [4] Kostas E. Bekris, Max Glick, and Lydia E. Kavraki. Evaluation of algorithms for bearing-only SLAM. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 1937–1943. IEEE, 2006.
- [5] M. Bosse, P. Newman, J. Leonard, and S. Teller. Simultaneous localization and map building in large-scale cyclic environments using the Atlas framework. *International Journal of Robotics Research*, 23(12):1113–1139, December 2004.
- [6] Léon Bottou. Stochastic learning. In Olivier Bousquet and Ulrike von Luxburg, editors, *Advanced Lectures on Machine Learning*, number LNAI 3176 in Lecture Notes in Artificial Intelligence, pages 146–168. Springer Verlag, Berlin, 2004.
- [7] Christian Darken, Joseph Chang, and John Moody. Learning rate schedules for faster stochastic gradient search. In *Proceedings of Neural Networks for Signal Processing 2*. IEEE Press, 1992.
- [8] Timothy A. Davis, John R. Gilbert, Stefan I. Larimore, and Esmond G. Ng. A column approximate minimum degree ordering algorithm. *ACM Transactions on Mathematical Software*, 30(3):353–376, 2004.
- [9] A.J. Davison. Real-time simultaneous localisation and mapping with a single camera. In *Proceedings of the IEEE International Conference on Computer Vision*, volume 2, pages 1403–1410, 2003.
- [10] F. Dellaert and M. Kaess. Square root SAM: Simultaneous localization and mapping via square root information smoothing. *International Journal of Robotics Research*, 25(12):1181–1203, December 2006.

- [11] Chris Ding, XiaoFeng He, Hongyuan Zha, Ming Gu, and Horst D. Simon. A Min-MaxCut spectral method for data clustering and graph partitioning. Technical Report 54111, Lawrence Berkeley National Laboratory, 2003.
- [12] T. Duckett, S. Marsland, and J. Shapiro. Learning globally consistent maps by relaxation. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, volume 4, pages 3841–3846, San Francisco, CA, 2000.
- [13] R. Eustice, H. Singh, and J. Leonard. Exactly sparse delayed-state filters. In *Proceedings of the 2005 IEEE International Conference on Robotics and Automation*, pages 2428–2435, Barcelona, Spain, April 2005.
- [14] R.M. Eustice, H. Singh, J.J. Leonard, and M.R. Walter. Visually mapping the RMS Titanic: Conservative covariance estimates for SLAM information filters. *International Journal of Robotics Research*, 25(12):1223–1242, December 2006.
- [15] M.A. Fischler and R.C. Bolles. Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, 24(6):381–395, June 1981.
- [16] John Folkesson and H. I. Christensen. Graphical SLAM - a self-correcting map. *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 791–798, 2004.
- [17] John Folkesson and Henrik I. Christensen. Closing the loop with graphical SLAM. *IEEE Transactions on Robotics*, 23(4):731–741, 2007.
- [18] U. Frese. A proof for the approximate sparsity of SLAM information matrices. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 331–337, Barcelona, Spain, April 2005.
- [19] U. Frese. Treemap: An $\mathcal{O}(\log n)$ algorithm for simultaneous localization and mapping. In C. Freksa, editor, *Spatial Cognition IV*, pages 455–476. Springer-Verlag, 2005.
- [20] U. Frese, P. Larsson, and T. Duckett. A multilevel relaxation algorithm for simultaneous localization and mapping. *IEEE Transactions on Robotics*, 21(2):196–207, April 2005.
- [21] G. Gallo, M. D. Grigoriadis, and R. E. Tarjan. A fast parametric maximum flow algorithm and applications. *SIAM J. Comput.*, 18(1):30–55, 1989.
- [22] G. Grisetti, C. Stachniss, and W. Burgard. Improving grid-based SLAM with Rao-Blackwellized particle filters by adaptive proposals and selective resampling. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 2432–2437, Barcelona, April 2005.

-
- [23] Giorgio Grisetti, Dario Lodi Rizzini, Cyrill Stachniss, Edwin Olson, and Wolfram Burgard. Online constraint network optimization for efficient maximum likelihood map learning. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, 2008.
 - [24] Giorgio Grisetti, Cyrill Stachniss, Slawomir Grzonka, and Wolfram Burgard. A tree parameterization for efficiently computing maximum likelihood maps using gradient descent. In *Proceedings of Robotics: Science and Systems (RSS)*, Atlanta, GA, USA, 2007.
 - [25] Andreas Haeberlen, Eliot Flannery, Andrew M. Ladd, Algis Rudys, Dan S. Wallach, and Lydia E. Kavraki. Practical robust localization over large-scale 802.11 wireless networks. In *MobiCom '04: Proceedings of the 10th annual international conference on Mobile computing and networking*, pages 70–84, New York, NY, USA, 2004. ACM.
 - [26] D. Hähnel, W. Burgard, B. Wegbreit, and S. Thrun. Towards lazy data association in SLAM. In *Proceedings of the International Symposium of Robotics Research (ISRR)*, Sienna, Italy, 2003. Springer.
 - [27] R.I. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, second edition, 2004.
 - [28] Berthold K. P. Horn. Closed-form solution of absolute orientation using unit quaternions. *Journal of the Optical Society of America. A*, 4(4):629–642, Apr 1987.
 - [29] Andrew Howard and Nicholas Roy. The robotics data set repository (radish), 2003.
 - [30] M.I. Jordan. *Learning in Graphical Models*. MIT Press, 1999.
 - [31] Michael Kaess, Ananth Ranganathan, and Frank Dellaert. Fast incremental square root information smoothing. In *International Joint Conference on Artificial Intelligence (IJCAI)*, pages 2129–2134, Hyderabad; India, 2007.
 - [32] Yan Ke and Rahul Sukthankar. PCA-SIFT: A more distinctive representation for local image descriptors. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, 02:506–513, 2004.
 - [33] Alexander Kleiner, Johann Prediger, and Bernhard Nebel. RFID technology-based exploration and SLAM for search and rescue. *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 4054–4059, October 2006.
 - [34] Kurt Konolige. Large-scale map-making. In *Proceedings of the National Conference on Artificial Intelligence (AAAI)*, pages 457–463, 2004.

- [35] Benjamin Kuipers and Patrick Beeson. Bootstrap learning for place recognition. In *Proceedings of the National Conference on Artificial Intelligence (AAAI)*, pages 174–180, Edmonton, Canada, 2002.
- [36] J. Leonard and P. Newman. Consistent, convergent, and constant-time SLAM. In *International Joint Conference on Artificial Intelligence (IJCAI)*, pages 1143–1150, Acapulco, Mexico, August 2003. IJCAI.
- [37] John Leonard, David Barrett, Jonathan How, Seth Teller, and et al. Team MIT DARPA urban challenge technical report. Technical report, Massachusetts Institute of Technology, April 2007.
- [38] David G. Lowe. Object recognition from local scale-invariant features. In *International Conference on Computer Vision*, pages 1150–1157, Corfu, Greece, 1999.
- [39] David G. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60(2):91–110, November 2004.
- [40] F. Lu and E. Milios. Robot pose estimation in unknown environments by matching 2d range scans. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 935–938, 1994.
- [41] F. Lu and E. Milios. Globally consistent range scan alignment for environment mapping. *Autonomous Robots*, 4(4):333–349, April 1997.
- [42] F. Lu and E. Milios. Robot pose estimation in unknown environments by matching 2D range scans. *Journal of Intelligent and Robotic Systems*, 18(249-275), 1997.
- [43] Peter S. Maybeck. *Stochastic Models, Estimation, and Control*, volume 1. Academic Press, New York, NY, 1979.
- [44] Michael Montemerlo. *FastSLAM: A Factored Solution to the Simultaneous Localization and Mapping Problem with Unknown Data Association*. PhD thesis, Robotics Institute, Carnegie Mellon University, Pittsburgh, PA, July 2003.
- [45] Michael Montemerlo, Nicholas Roy, and Sebastian Thrun. Perspectives on standardization in mobile robot programming: The Carnegie Mellon navigation (CARMEN) toolkit. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, volume 3, pages 2436–2441, Las Vegas, NV, October 2003.
- [46] Jose Neira and Juan D. Tardos. Data association in stochastic mapping using the joint compatibility test. *IEEE Transactions on Robotics and Automation*, 17(6):890–897, December 2001.

- [47] Paul M. Newman and John Leonard. Pure range-only sub-sea SLAM. In *Proceedings of the IEEE Conference on Robotics and Automation (ICRA '02)*, pages 1921–1926, 2003.
- [48] Edwin Olson, John Leonard, and Seth Teller. Fast iterative optimization of pose graphs with poor initial estimates. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 2262–2269, 2006.
- [49] Edwin Olson, John Leonard, and Seth Teller. Robust range-only beacon localization. *IEEE Journal of Oceanic Engineering*, 31(4):949–958, October 2006.
- [50] Edwin Olson, John Leonard, and Seth Teller. Spatially-adaptive learning rates for online incremental SLAM. In *Proceedings of Robotics: Science and Systems*, Atlanta, GA, USA, June 2007.
- [51] Edwin Olson, Matthew Walter, John Leonard, and Seth Teller. Single cluster graph partitioning for robotics applications. In *Proceedings of Robotics Science and Systems*, pages 265–272, 2005.
- [52] M.A. Paskin. Thin junction tree filters for simultaneous localization and mapping. Technical Report UCB/CSD-02-1198, University of California, Berkeley, September 2002.
- [53] Son Bao Pham, Bernhard Hengst, Darren Ibbotson, and Claude Sammut. Stochastic gradient descent localisation in quadruped robots. In Andreas Birk 0002, Silvia Coradeschi, and Satoshi Tadokoro, editors, *RoboCup*, volume 2377 of *Lecture Notes in Computer Science*, pages 447–452. Springer, 2001.
- [54] Nissanka Priyantha, Anit Chakraborty, and Hari Balakrishnan. The cricket location-support system. In *Proceedings of the ACM International Conference on Mobile Computing and Networking (MobiCom)*, pages 32–34, Boston, MA, August 2000.
- [55] Ronald L. Rivest and Charles E. Leiserson. *Introduction to Algorithms*. McGraw-Hill, Inc., New York, NY, USA, 1990.
- [56] H. Robbins and S. Monro. A stochastic approximation method. *Annals of Mathematical Statistics*, 22:400–407, 1951.
- [57] Nicholas Roy and Joelle Pineau. *Gerontechnology: Growing Old in a Technological Society*, chapter Robotics and Independence for the Elderly, pages 209–242. Charles C. Thomas Publisher Ltd., 2007. Co-authors: Nicholas Roy (MIT).
- [58] Brad Schumitsch, Sebastian Thrun, Gary Bradski, and Kunle Olukotun. The information-form data association filter. In Y. Weiss, B. Schölkopf, and J. Platt, editors, *Advances in Neural Information Processing Systems (NIPS)*, pages 1193–1200, Cambridge, MA, 2006. MIT Press.

- [59] S. Se, D. Lowe, and J. Little. Vision-based mobile robot localization and mapping using scale-invariant features. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 2051–2058, Seoul, Korea, May 2001.
- [60] S. Se, D. Lowe, and J. Little. Mobile robot localization and mapping with uncertainty using scale-invariant visual landmarks. *International Journal of Robotics Research*, 21(8):735–758, 2002.
- [61] H. Shatkay and L. Kaelbling. Learning topological maps with weak local odometric information. In *International Joint Conference on Artificial Intelligence (IJCAI)*, pages 920–929, 1997.
- [62] J. Shi and J. Malik. Normalized cuts and image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pages 888–905, August 2000.
- [63] Sudipta Sinha, Jan-Michael Frahm, Marc Pollefeys, and Yakup Genc. Feature tracking and matching in video using programmable graphics hardware. *Machine Vision and Applications*, November 2007.
- [64] J. Sivic and A. Zisserman. Video Google: Efficient visual search of videos. In J. Ponce, M. Hebert, C. Schmid, and A. Zisserman, editors, *Toward Category-Level Object Recognition*, volume 4170 of *LNCS*, pages 127–144. Springer, 2006.
- [65] R. Smith, M. Self, and P. Cheeseman. A stochastic map for uncertain spatial relationships. In O. Faugeras and G. Giralt, editors, *Proceedings of the International Symposium of Robotics Research (ISRR)*, pages 467–474, 1988.
- [66] C. Stachniss, G. Grisetti, and W. Burgard. Recovering particle diversity in a Rao-Blackwellized particle filter for SLAM after actively closing loops. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 667–672, Barcelona, Spain, 2005.
- [67] Gilbert Strang. *Introduction to Linear Algebra*. Wellesley-Cambridge Press, 1993.
- [68] J.D. Tardós, J. Neira, P.M. Newman, and J.J. Leonard. Robust mapping and localization in indoor environments using sonar data. *International Journal of Robotics Research*, 21(4):311–330, 2002.
- [69] S. Thrun and Y. Liu. Multi-robot SLAM with sparse extended information filters. In *Proceedings of the International Symposium of Robotics Research (ISRR)*, Sienna, Italy, 2003.
- [70] S. Thrun, Y. Liu, D. Koller, A.Y. Ng, Z. Ghahramani, and H. Durrant-Whyte. Simultaneous localization and mapping with sparse extended information filters. Technical report, Carnegie Mellon University, Pittsburgh, PA, April 2003.

- [71] S. Thrun, Y. Liu, D. Koller, A.Y. Ng, Z. Ghahramani, and H. Durrant-Whyte. Simultaneous localization and mapping with sparse extended information filters. *International Journal of Robotics Research*, 23(7-8):693–716, July-August 2004.
- [72] S. Thrun and M. Montemerlo. The GraphSLAM algorithm with applications to large-scale mapping of urban structures. *International Journal of Robotics Research*, 25(5-6):403–430, May-June 2006.
- [73] Walker. Degrees of freedom. *Journal of Educational Psychology*, 31(4):253–269, 1940.
- [74] M. Walter, R. Eustice, and J. Leonard. A provably consistent method for imposing exact sparsity in feature-based SLAM information filters. In S. Thrun, R. Brooks, and H. Durrant-Whyte, editors, *Proceedings of the International Symposium of Robotics Research (ISRR)*, pages 214–234, San Francisco, CA, October 2005. Springer.
- [75] M. Walter, F. Hover, and J. Leonard. SLAM for ship hull inspection using exactly sparse extended information filters. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, Pasadena, CA, May 2008.
- [76] Matthew R. Walter, Ryan M. Eustice, and John J. Leonard. Exactly sparse extended information filters for feature-based SLAM. *Int. J. Rob. Res.*, 26(4):335–359, 2007.

Attachment D

297537764-MIT.pdf

1 / 129100%+

1

2

3

4

5

Robust and Efficient Robotic Mapping

by

Edwin B. Olson

Department of
Engineering and Computer Science
Requirements for the degree of
Master Science and Engineering
e
INSTITUTE OF TECHNOLOGY
2008
III. All rights reserved.
permission to reproduce and
atic copies of this thesis document
in part.

Engineering and Computer Science
May 16, 2008

Certified by...
Seth Teller
Professor of Computer Science and Engineering
Thesis Supervisor

Certified by...
John Leonard
Professor of Mechanical and Ocean Engineering
Thesis Supervisor

Terry P. Orlando

Document properties

File name:297537764-MIT.pdf

File size:58.6 MB

Title:-

Author:-

Subject:-

Keywords:-

Created:2/8/09, 10:06:25 PM

Modified:2/9/09, 8:12:52 PM

Application:-

PDF producer:-

PDF version:1.6

Page count:129

Page size:8.50 × 11.00 in (portrait)

Fast web view:Yes

Close

Lightshot Screenshot

MASSACHUSETTS INSTITUTE

Attachment E

DSpace@MIT

Show simple item record

Robust and efficient robotic mapping

dc.contributor.advisor	Seth Teller and John Leonard.	en_US
dc.contributor.author	Olson, Edwin B. (Edwin Brock), 1977-	en_US
dc.contributor.other	Massachusetts Institute of Technology. Dept. of Electrical Engineering and Computer Science.	en_US
dc.date.accessioned	2009-03-16T19:33:04Z	
dc.date.available	2009-03-16T19:33:04Z	
dc.date.copyright	2008	en_US
dc.date.issued	2008	en_US
dc.identifier.uri	http://hdl.handle.net/1721.1/44713	
dc.description	Thesis (Ph. D.)--Massachusetts Institute of Technology, Dept. of Electrical Engineering and Computer Science, 2008.	en_US
dc.description	Includes bibliographical references (p. 123-129).	en_US

dc.description.abstract	<p>Mobile robots are dependent upon a model of the environment for many of their basic functions. Locally accurate maps are critical to collision avoidance, while large-scale maps (accurate both metrically and topologically) are necessary for efficient route planning. Solutions to these problems have immediate and important applications to autonomous vehicles, precision surveying, and domestic robots. Building accurate maps can be cast as an optimization problem: find the map that is most probable given the set of observations of the environment. However, the problem rapidly becomes difficult when dealing with large maps or large numbers of observations. Sensor noise and non-linearities make the problem even more difficult especially when using inexpensive (and therefore preferable) sensors. This thesis describes an optimization algorithm that can rapidly estimate the maximum likelihood map given a set of observations. The algorithm, which iteratively reduces map error by considering a single observation at a time, scales well to large environments with many observations. The approach is particularly robust to noise and non-linearities, quickly escaping local minima that trap current methods. Both batch and online versions of the algorithm are described. In order to build a map, however, a robot must first be able to recognize places that it has previously seen. Limitations in sensor processing algorithms, coupled with environmental ambiguity, make this difficult. Incorrect place recognitions can rapidly lead to divergence of the map. This thesis describes a place recognition algorithm that can robustly handle ambiguous data. We evaluate these algorithms on a number of challenging datasets and provide quantitative comparisons to other state-of-the-art methods, illustrating the advantages of our methods.</p>	en_US
dc.description.statementsofresponsibility	by Edwin B. Olson.	en_US
dc.format.extent	129 p.	en_US
dc.language.iso	eng	en_US
dc.publisher	Massachusetts Institute of Technology	en_US

dc.rights	M.I.T. theses are protected by copyright. They may be viewed from this source for any purpose, but reproduction or distribution in any format is prohibited without written permission. See provided URL for inquiries about permission.	en_US
dc.rights.uri	http://dspace.mit.edu/handle/1721.1/7582	en_US
dc.subject	Electrical Engineering and Computer Science.	en_US
dc.title	Robust and efficient robotic mapping	en_US
dc.type	Thesis	en_US
dc.description.degree	Ph.D.	en_US
dc.contributor.department	Massachusetts Institute of Technology. Department of Electrical Engineering and Computer Science	
dc.identifier.oclc	297537764	en_US

Files in this item



Name:
297537764-MIT.pdf
Size:
58.63Mb
Format:
PDF
Description:
Full printable version

[View/Open](#)

This item appears in the following Collection(s)

- [Doctoral Theses](#)

[Show simple item record](#)

[Show Statistical Information](#)

[Privacy](#) | [Permissions](#) | [Accessibility](#) | [Contact us](#)

Content created by the MIT Libraries, [CC BY-NC](#) unless otherwise noted. [Notify us about copyright concerns.](#)

Attachment F

Articles

About 125 results (0.02 sec)

My profile

My library

Any time

Since 2025

Since 2024

Since 2021

Custom range

2008

2015

Search

Citations

2012

2013

2014

2015

Sort by relevance

Sort by date

Create alert

Robust and efficient robotic mapping

Search within citing articles

A tutorial on graph-based SLAM

G. Grisetti, R. Kümmerle, C. Stachniss... - IEEE Intelligent ..., 2011 - ieeexplore.ieee.org

Being able to build a map of the environment and to simultaneously localize within this map is an essential skill for mobile robots navigating in unknown environments in absence of ...

☆ Save 99 Cite Cited by 1986 Related articles All 20 versions

Appearance-only SLAM at large scale with FAB-MAP 2.0

M. Cummins, P. Newman - The International Journal of ..., 2011 - journals.sagepub.com

We describe a new formulation of appearance-only SLAM suitable for very large scale place recognition. The system navigates in the space of appearance, assigning each new ...

☆ Save 99 Cite Cited by 863 Related articles All 4 versions

Autonomous multi-floor indoor navigation with a computationally constrained MAV

S. Shen, N. Michael, V. Kumar - 2011 IEEE International ..., 2011 - ieeexplore.ieee.org

In this paper, we consider the problem of autonomous navigation with a micro aerial vehicle (MAV) in indoor environments. In particular, we are interested in autonomous navigation in ...

☆ Save 99 Cite Cited by 560 Related articles All 12 versions

Real-time correlative scan matching

E.B. Olson - 2009 IEEE International conference on robotics and ..., 2009 - ieeexplore.ieee.org

Scan matching, the problem of registering two laser scans in order to determine the relative positions from which the scans were obtained, is one of the most heavily relied-upon tools ...

☆ Save 99 Cite Cited by 549 Related articles All 10 versions

Cooperative localization for autonomous underwater vehicles

A. Bahi, J. Leonard, M.E. Fallon - The International Journal of ..., 2009 - journals.sagepub.com

This paper describes an algorithm for distributed acoustic navigation for Autonomous Underwater Vehicles (AUVs). Whereas typical AUV navigation systems utilize pre-calibrated ...

☆ Save 99 Cite Cited by 585 Related articles All 27 versions 86

RANGE—Robust autonomous navigation in GPS-denied environments

A. Bachrach, S. Prentice, R. He... - Journal of Field Robotics, 2011 - Wiley Online Library

This paper addresses the problem of autonomous navigation of a micro air vehicle (MAV) in GPS-denied environments. We present experimental validation and analysis for our system ...

☆ Save 99 Cite Cited by 500 Related articles All 14 versions

On measuring the accuracy of SLAM algorithms

R. Kümmerle, B. Steder, C. Dornhege, M. Ruhnke... - Autonomous ..., 2009 - Springer

In this paper, we address the problem of creating an objective benchmark for evaluating SLAM approaches. We propose a framework for analyzing the results of a SLAM approach ...

☆ Save 99 Cite Cited by 505 Related articles All 30 versions

Inference on networks of mixtures for robust robot mapping

E. Olson, P. Agarwal - The International Journal of Robotics ..., 2013 - journals.sagepub.com

The central challenge in robotic mapping is obtaining reliable data associations (or "loop closures") state-of-the-art inference algorithms can fail catastrophically if even one ...

☆ Save 99 Cite Cited by 353 Related articles All 23 versions

[PDF] Information-Theoretic Planning with Trajectory Optimization for Dense 3D Mapping.

B. Charrow, G. Kahn, S. Patil, S. Liu... - Robotics: Science ..., 2015 - roboticsproceedings.org

We propose an information-theoretic planning approach that enables mobile robots to autonomously construct dense 3D maps in a computationally efficient manner. Inspired by ...

☆ Save 99 Cite Cited by 239 Related articles All 12 versions 86

Information-theoretic mapping using cauchy-schwarz quadratic mutual information

B. Charrow, S. Liu, V. Kumar... - 2015 IEEE International ..., 2015 - ieeexplore.ieee.org

We develop a computationally efficient control policy for active perception that incorporates explicit models of sensing and mobility to build 3D maps with ground and aerial robots. Like ...

☆ Save 99 Cite Cited by 218 Related articles All 2 versions

[PDF] researchgate.net

[PDF] utk.edu

[PDF] archive.org

[PDF] dl.ic.mil

[PDF] mit.edu

[PDF] diva-portal.org

[PDF] uni-freiburg.de

[PDF] roboticsproceedings.org

Google

1 2 3 4 5 6 7 8 9 10 Next

Privacy

Terms

Help

Avidbots Ex1008
Page 168 of 545

Google Scholar

Articles

Page 2 of about 126 results (0.02 sec)

My profile

My library

Any time

Since 2025

Since 2024

Since 2021

Custom range

2008

2015

Search

Citations

2012

2013

2014

2015

Sort by relevance

Sort by date

Create alert

Robust and efficient robotic mapping

Search within citing articles

Hierarchical optimization on manifolds for online 2D and 3D mapping

G. Grisetti, R. Kümmerle, C. Stachniss - ... on Robotics and ... 2010 - IEEE explore. IEEE.org

In this paper, we present a new hierarchical optimization solution to the graph-based simultaneous localization and mapping (SLAM) problem. During online mapping, the ...

☆ Save 90 Cite Cited by 316 Related articles All 19 versions

Stereo vision and laser odometry for autonomous helicopters in GPS-denied indoor environments

M. Achtelik, A. Bachrach, R. He - ... Unmanned Systems ... 2009 - spiedigitallibrary.org

This paper presents our solution for enabling a quadrotor helicopter to autonomously navigate unstructured and unknown indoor environments. We compare two sensor suites ...

☆ Save 90 Cite Cited by 326 Related articles All 14 versions

Autonomous flight in unknown indoor environments

A. Bachrach, R. He, N. Roy - International journal of micro air ... 2009 - journals.sagepub.com

This paper presents our solution for enabling a quadrotor helicopter, equipped with a laser rangefinder sensor, to autonomously explore and map unstructured and unknown indoor ...

☆ Save 90 Cite Cited by 277 Related articles All 9 versions

Autonomous robot navigation in highly populated pedestrian zones

R. Kümmerle, M. Ruhnke, B. Steder - Journal of Field ... 2015 - Wiley Online Library

In the past, there has been a tremendous amount of progress in the area of autonomous robot navigation, and a large variety of robots have been developed that demonstrated ...

☆ Save 90 Cite Cited by 171 Related articles All 9 versions

Large scale graph-based SLAM using aerial images as prior information

R. Kümmerle, B. Steder, C. Dornhege, A. Kleinert - Autonomous ... 2011 - Springer

The problem of learning a map with a mobile robot has been intensively studied in the past and is usually referred to as the simultaneous localization and mapping (SLAM) problem ...

☆ Save 90 Cite Cited by 202 Related articles All 33 versions

Lifelong localization in changing environments

G.D. Tipaldi, D. Meyer-Delius - ... International Journal of ... 2013 - journals.sagepub.com

Robot localization systems typically assume that the environment is static. Ignoring the dynamics inherent in most real-world settings. Corresponding scenarios include ...

☆ Save 90 Cite Cited by 170 Related articles All 8 versions

CAT-SLAM: probabilistic localisation and mapping using a continuous appearance-based trajectory

W. Maddern, M. Milford, G. Wyeth - The International Journal ... 2012 - journals.sagepub.com

This paper describes a new system, dubbed Continuous Appearance-based Trajectory Simultaneous Localisation and Mapping (CAT-SLAM), which augments sequential ...

☆ Save 90 Cite Cited by 164 Related articles All 6 versions

A comparison of SLAM algorithms based on a graph of relations

W. Burgard, C. Stachniss, G. Grisetti - 2009 IEEE/RSJ ... 2009 - IEEE explore. IEEE.org

In this paper, we address the problem of creating an objective benchmark for comparing SLAM approaches. We propose a framework for analyzing the results of SLAM approaches ...

☆ Save 90 Cite Cited by 181 Related articles All 21 versions

Information-theoretic compression of pose graphs for laser-based SLAM

H. Kretschmar, C. Stachniss - The International Journal of ... 2012 - journals.sagepub.com

In graph-based simultaneous localization and mapping (SLAM), the pose graph grows over time as the robot gathers information about the environment. An ever growing pose graph ...

☆ Save 90 Cite Cited by 150 Related articles All 10 versions

Extracting general-purpose features from LIDAR data

Y. Li, E.B. Olson - ... IEEE International conference on robotics and ... 2010 - IEEE explore. IEEE.org

The detection of features from Light Detection and Ranging (LIDAR) data is a fundamental component of feature-based mapping and SLAM systems. Existing detectors tend to exploit ...

☆ Save 90 Cite Cited by 149 Related articles All 6 versions

[PDF] academia.edu

[PDF] spiedigitallibrary.org

[PDF] sagepub.com

[PDF] researchgate.net

[PDF] diva-portal.org

[PDF] uni-freiburg.de

[PDF] qut.edu.au

[PDF] diva-portal.org

[PDF] uni-bonn.de

[PDF] lth.se

< Google >

Previous 1 2 3 4 5 6 7 8 9 10 Next

Privacy Terms Help

Avidbots Ex1008
Page 169 of 545

Articles

Any time
Since 2025
Since 2024
Since 2021
Custom range
2008 — 2015
Search

Citations
2012 2013 2014 2015

Sort by relevance
Sort by date
Create alert

Page 3 of about 126 results (0.02 sec)

My profileMy library

Robust and efficient robotic mapping

Search within citing articles

[PDF] Autonomous navigation and exploration of a quadrotor helicopter in GPS-denied indoor environments

Machtellis, A. Bachrach, R. He. . . - First symposium on . . . , 2009 - aerialroboticscompetition.org

This paper presents our solution for enabling a quadrotor helicopter to autonomously navigate, explore and locate objects of interest in unstructured and unknown indoor . . .

☆ Save 00 Cite Cited by 187 Related articles All 6 versions 00

[PDF] aerialroboticscompetition . . .

Progress toward multi-robot reconnaissance and the MAGIC 2010 competition

E. Olson, J. Sirom, R. Merton, A. Richardson. . . - Journal of Field . . . , 2012 - Wiley Online Library

Tasks like search-and-rescue and urban reconnaissance benefit from large numbers of robots working together, but high levels of autonomy are needed to reduce operator . . .

☆ Save 00 Cite Cited by 121 Related articles All 7 versions

[PDF] academia.edu

Robust place recognition with stereo sequences

C. Cadena, D. Galvez-Lopez. . . - IEEE Transactions on . . . , 2012 - ieeexplore.ieee.org

We propose a place recognition algorithm for simultaneous localization and mapping (SLAM) systems using stereo cameras that considers both appearance and geometric . . .

☆ Save 00 Cite Cited by 118 Related articles All 9 versions

[PDF] unizar.es

Autonomous flight in unstructured and unknown indoor environments

A.G. Bachrach - 2009 - dspace.mit.edu

This thesis presents the design, implementation, and validation of a system that enables a micro air vehicle to autonomously explore and map unstructured and unknown indoor . . .

☆ Save 00 Cite Cited by 148 Related articles All 7 versions 00

[PDF] mit.edu

Consistent sparsification for graph optimization

G. Huang, M. Kassas, J. Leonard. . . - 2013 European Conference . . . , 2013 - ieeexplore.ieee.org

In a standard pose-graph formulation of simultaneous localization and mapping (SLAM), due to the continuously increasing numbers of nodes (states) and edges (measurements) . . .

☆ Save 00 Cite Cited by 95 Related articles All 10 versions

[PDF] mit.edu

A heuristic loop closing technique for large-scale 6d slam

J. Sprickerhof, A. Nüchter, K. Lingemann, J. Hertzberg - Automatica. 2011 - Taylor & Francis

This paper presents a novel heuristic for correcting scan pose estimations after loop closing in SLAM using 3D laser scans. Contrary to state of the art approaches, the built SLAM graph . . .

☆ Save 00 Cite Cited by 115 Related articles All 17 versions

[PDF] tandfonline.com

A navigation system for robots operating in crowded urban environments

R. Kümmerle, M. Ruhnke, B. Steder. . . - on Robotics and . . . , 2013 - ieeexplore.ieee.org

Over the past years, there has been a tremendous progress in the area of robot navigation. Most of the systems developed thus far, however, are restricted to indoor scenarios, non . . .

☆ Save 00 Cite Cited by 93 Related articles All 11 versions

[PDF] uni-freiburg.de

[PDF] Incremental Block Cholesky Factorization for Nonlinear Least Squares in Robotics.

L. Polak, V. Ila, M. Solomy, P. Smrz. . . - Robotics: Science and . . . , 2013 - researchgate.net

Efficiently solving nonlinear least squares (NLS) problems is crucial for many applications in robotics. In online applications, solving the associated nonlinear systems every step may . . .

☆ Save 00 Cite Cited by 78 Related articles All 8 versions 00

[PDF] researchgate.net

[HTML] Fast fingerprint database maintenance for indoor positioning based on UGV SLAM

J. Tang, Y. Chen, L. Chen, J. Liu, J. Hyppä, A. Kukko. . . - Sensors, 2015 - mdpi.com

Indoor positioning technology has become more and more important in the last two decades. Utilizing Received Signal Strength Indicator (RSSI) fingerprints of Signals of . . .

☆ Save 00 Cite Cited by 63 Related articles All 19 versions 00

[HTML] mdpi.com

Recognizing places using spectrally clustered local matches

E. Olson - Robotics and Autonomous Systems, 2009 - Elsevier

Place recognition is a fundamental perceptual problem at the heart of many basic robot operations, most notably mapping. Failures can result from ambiguous sensor readings and . . .

☆ Save 00 Cite Cited by 97 Related articles All 6 versions

< Google >
Previous 1 2 3 4 5 6 7 8 9 10 Next

PrivacyTermsHelp

Avidbots Ex1008
Page 170 of 545

Articles

Page 4 of about 126 results (0.03 sec)

My profile

My library

Any time

Since 2025

Since 2024

Since 2021

Custom range

2008

2015

Search

Citations

2012

2013

2014

2015

Sort by relevance

Sort by date

Create alert

Robust and efficient robotic mapping

Search within citing articles

Study of parameterizations for the rigid body transformations of the scan registration problem

A Nuchter, J Elieberg, P Schneider, D Paulus

Computer Vision and ...

2010

Elsevier

The iterative closest point (ICP) algorithm is the de facto standard for geometric alignment of three-dimensional models when an initial relative pose estimate is available. The basis of ...

☆ Save

00 Cite

Cited by 95

Related articles

All 8 versions

[PDF] uni-wuerzburg.de

(PDF) Fab-map: Appearance-based place recognition and mapping using a learned visual vocabulary model

M J Cummings, PM Newman

Proceedings of the 27th International ...

2010

robots.ox.ac.uk

We present an overview of FAB-MAP, an algorithm for place recognition and mapping developed for infrastructure-free mobile robot navigation in large environments. The system ...

☆ Save

99 Cite

Cited by 75

Related articles

All 7 versions

00

[PDF] ox.ac.uk

Efficient information-theoretic graph pruning for graph-based SLAM with laser range finders

H Kretschmar, C Stachniss

...

2011

IEEE/RSJ ...

2011

ieeexplore.ieee.org

In graph-based SLAM, the pose graph encodes the poses of the robot during data acquisition as well as spatial constraints between them. The size of the pose graph has a ...

☆ Save

99 Cite

Cited by 83

Related articles

All 16 versions

[PDF] uni-freiburg.de

Simultaneous calibration, localization, and mapping

R Kumar, M G. S. ...

...

2011

IEEE/RSJ ...

2011

ieeexplore.ieee.org

The calibration parameters of a mobile robot play a substantial role in navigation tasks. Often these parameters are subject to variations that depend either on environmental ...

☆ Save

99 Cite

Cited by 70

Related articles

All 13 versions

[PDF] researchgate.net

(HTML) NAVIS-An UGV indoor positioning system using laser scan matching for large-area real-time applications

J Tang, Y Chen, A Jaakkola, J Liu, J Jyysa, H Hyppa

Sensors

2014

mdpi.com

Laser scan matching with grid-based maps is a promising tool for real time indoor positioning of mobile Unmanned Ground Vehicles (UGVs). While there are critical ...

☆ Save

99 Cite

Cited by 58

Related articles

All 16 versions

00

[HTML] mdpi.com

Structure tensors for general purpose LIDAR feature extraction

Y Li, FB Olson

...

2011

IEEE International Conference on ...

2011

ieeexplore.ieee.org

The detection of features from Light Detection and Ranging (LIDAR) data is a fundamental component of feature-based mapping and SLAM systems. Classical approaches are often ...

☆ Save

99 Cite

Cited by 64

Related articles

All 8 versions

[PDF] archive.org

Simultaneous localization and mapping with multimodal probability distributions

M Pfingsthorn, A Brix

The International Journal of Robotics ...

2013

journals.sagepub.com

Simultaneous Localization and Mapping (SLAM) has focused on noisy but unique data associations resulting in linear Gaussian uncertainty models. However, a unique decision is ...

☆ Save

99 Cite

Cited by 56

Related articles

All 5 versions

[PDF] sagepub.com

Variable reordering strategies for SLAM

P Agarwal, E Olson

...

2012

IEEE/RSJ International Conference ...

2012

ieeexplore.ieee.org

State of the art methods for state estimation and perception make use of least-squares optimization methods to perform efficient inference on noisy sensor data. Much of this ...

☆ Save

99 Cite

Cited by 57

Related articles

All 9 versions

[PDF] uni-freiburg.de

Robust place recognition with stereo cameras

C Cadena, D Gálvez-López, F Ramos

...

2010

IEEE/RSJ ...

2010

ieeexplore.ieee.org

Place recognition is a challenging task in any SLAM system. Algorithms based on visual appearance are becoming popular to detect locations already visited, also known as loop ...

☆ Save

99 Cite

Cited by 72

Related articles

All 11 versions

[PDF] academia.edu

(HTML) A general purpose feature extractor for light detection and ranging data

Y Li, FB Olson

Sensors

2010

mdpi.com

Feature extraction is a central step of processing Light Detection and Ranging (LIDAR) data. Existing detectors tend to exploit characteristics of specific environments: corners and lines ...

☆ Save

99 Cite

Cited by 63

Related articles

All 13 versions

00

[HTML] mdpi.com

< Google >

Previous 1 2 3 4 5 6 7 8 9 10 Next

Privacy Terms Help

Avidbots Ex1008
Page 171 of 545

Articles

Any time
Since 2025
Since 2024
Since 2021
Custom range

2008

2015

Search

Citations

2012

2013

2014

2015

Sort by relevance
Sort by date

Create alert

Page 5 of about 126 results (0.03 sec)

My profileMy library

Robust and efficient robotic mapping

Search within citing articles

[book] Probabilistic localization and mapping in appearance space

M.Cummins - 2009 - search.proquest.com

This thesis is concerned with the problem of place recognition for mobile robots. How can a robot determine its location from an image or sequence of images, without any prior ...

☆ Save 99 Cite Cited by 57 Related articles All 4 versions

[PDF] academia.edu

Simultaneous parameter calibration, localization, and mapping

R.Kummert, G.Grisetti, W.Burgard - Advanced Robotics, 2012 - Taylor & Francis

The calibration parameters of a mobile robot play a substantial role in navigation tasks. Often these parameters are subject to variations that depend either on changes in the ...

☆ Save 99 Cite Cited by 53 Related articles All 11 versions

[PDF] uniroma1.it

Towards a reliable SLAM back-end

G.Hu, K.Khosoussi, S.Huang - 2013 IEEE/RSJ International ..., 2013 - IEEE

In the state-of-the-art approaches to SLAM, the problem is often formulated as a non-linear least squares. SLAM back-ends often employ iterative methods such as Gauss-Newton or ...

☆ Save 99 Cite Cited by 45 Related articles All 11 versions

[PDF] uts.edu.au

Exploration and mapping with autonomous robot teams

E.Olson, J.Strom, R.Goeddel, R.Morton - Communications of the ..., 2013 - dl.acm.org

Exploration and mapping with autonomous robot teams Page 1 62 coMMunicATions of The AcM | MARCH 2013 | Vol. 56 | No. 3 contributed articles Urban reConnaissanCe and search-and-rescue ...

☆ Save 99 Cite Cited by 57 Related articles All 4 versions

[PDF] acm.org

Feature based graph-SLAM in structured environments

P.de la Puente, D.Rodriguez-Losada - Autonomous Robots, 2014 - Springer

Introducing a priori knowledge about the latent structure of the environment in simultaneous localization and mapping (SLAM), can improve the quality and consistency results of its ...

☆ Save 99 Cite Cited by 46 Related articles All 5 versions

[PDF] academia.edu

Fast covariance recovery in incremental nonlinear least square solvers

V.Jila, L.Polok, M.Soloviy, P.Smrz - 2015 IEEE International ..., 2015 - IEEE

Many estimation problems in robotics rely on efficiently solving nonlinear least squares (NLS). For example, it is well known that the simultaneous localisation and mapping (SLAM) ...

☆ Save 99 Cite Cited by 42 Related articles All 6 versions

[PDF] academia.edu

Obtaining lift-off indoors: Autonomous navigation in confined indoor environments

S.Shen, N.Michael, V.Kumar - IEEE Robotics & Automation ..., 2013 - IEEE

In this article, we consider the problem of autonomous navigation with a microaerial vehicle (MAV) in three-dimensional (3-D) confined indoor environments with multiple floors. We ...

☆ Save 99 Cite Cited by 48 Related articles All 2 versions

[PDF] d-nb.info

[PDF] Robust graph-based localization and mapping

P.Agarwal - 2015 - d-nb.info

Localization and mapping are fundamental requirements to enable mobile robots to operate robustly in their environments. The solution of these two problems will allow robots to assist ...

☆ Save 99 Cite Cited by 35 Related articles All 4 versions

[PDF] d-nb.info

[book] Information-theoretic active perception for multi-robot teams

B.Chaffrow - 2015 - search.proquest.com

Multi-robot teams that intelligently gather information have the potential to transform industries as diverse as agriculture, space exploration, mining, environmental monitoring ...

☆ Save 99 Cite Cited by 37 Related articles All 5 versions

[PDF] upenn.edu

FLIRT: Interest regions for 2D range data with applications to robot navigation

G.D.Tipaldi, M.Braun, K.O.Arras - Experimental Robotics: The 12th ..., 2014 - Springer

In this paper we present the Fast Laser Interest Region Transform (FLIRT), a multi-scale interest region operator for 2D range data. FLIRT combines a detector based on a geodesic ...

☆ Save 99 Cite Cited by 53 Related articles All 5 versions

[PDF] uni-freiburg.de

< Google >

Previous 1 2 3 4 5 6 7 8 9 10 Next

Privacy Terms Help

Avidbots Ex1008
Page 172 of 545

Google Scholar

Articles

Page 6 of about 126 results (0.03 sec)

My profile

My library

Any time

Since 2025

Since 2024

Since 2021

Custom range

2008

2015

Search

Citations

2012

2013

2014

2015

Sort by relevance

Sort by date

Create alert

Robust and efficient robotic mapping

Search within citing articles

Where am I? Creating spatial awareness in unmanned ground robots using SLAM: A survey

NK Dhiman, D Deodhare, D Khemani - Sadhana, 2015 - Springer

This paper presents a survey of Simultaneous Localization And Mapping (SLAM) algorithms for unmanned ground robots. SLAM is the process of creating a map of the environment ...

☆ Save 90 Cite Cited by 29 Related articles All 6 versions

Autonomous navigation in complex indoor and outdoor environments with micro aerial vehicles

S. Shao - 2014 - search.proquest.com

Micro aerial vehicles (MAVs) are ideal platforms for surveillance and search and rescue in confined indoor and outdoor environments due to their small size, superior mobility, and ...

☆ Save 90 Cite Cited by 33 Related articles All 4 versions

The University of Pennsylvania MAGIC 2010 multi-robot unmanned vehicle system

J. Butrick, K. Daniilidis, A. Kushleyev - Journal of Field ..., 2012 - Wiley Online Library

In this report, we describe the technical approach and algorithms that have been used by the University of Pennsylvania in the MAGIC 2010 competition. We have constructed and ...

☆ Save 90 Cite Cited by 39 Related articles All 5 versions

Lifelong map learning for graph-based slam in static environments

H. Kretschmar, G. Grisetti, C. Stachniss - KI-Künstliche Intelligenz, 2010 - Springer

In this paper, we address the problem of lifelong map learning in static environments with mobile robots using the graph-based formulation of the simultaneous localization and ...

☆ Save 90 Cite Cited by 40 Related articles All 15 versions

Using embodied data for localization and mapping

J. Schwendner, S. Jayasuriya, E. Kitchner - Journal of Field Robotics, 2014 - Wiley Online Library

Mobile autonomous robots have finally emerged from the confined spaces of structured and controlled indoor environments. To fulfill the promises of ubiquitous robotics in unstructured ...

☆ Save 90 Cite Cited by 26 Related articles All 4 versions

IPJC: The incremental posterior joint compatibility test for fast feature cloud matching

Y. Li, E. B. Olson - 2012 IEEE/RSJ International Conference on ..., 2012 - ieeexplore.ieee.org

One of the fundamental challenges in robotics is data-association: determining which sensor observations correspond to the same physical object. A common approach is to consider ...

☆ Save 90 Cite Cited by 28 Related articles All 7 versions

Uncertainty estimation for a 6-dof spectral registration method as basis for sonar-based underwater 3d slam

M. Pfingsthorn, A. Birk, H. Buelow - 2012 IEEE International ..., 2012 - ieeexplore.ieee.org

An uncertainty estimation method for 6 degree of freedom (6-DoF) spectral registration is introduced here. The underlying 6-DoF registration method based on Phase Only Matched ...

☆ Save 90 Cite Cited by 30 Related articles All 4 versions

基于图优化的同时定位与地图创建综述

梁镇杰, 阮华伟, 罗安华 - 机器人, 2013 - researchgate.net

基于图优化的同时定位与地图创建(SLAM)是当前机器人领域的研究热点。从帧间配准、环形闭合检测以及优化技术3个主要方面对基于图优化的同时定位与地图创建进行综述 ...

☆ Save 90 Cite Cited by 56 Related articles All 4 versions

A lightweight slam algorithm for indoor autonomous navigation

P. Tulpieghin, M. Umetti, N. Giordani - Proc. Australas. Conf ..., 2014 - academia.edu

Abstract: Simultaneous Localization and Mapping (SLAM) algorithms require huge computational power. Most of the state-of-the-art implementations employ dedicated ...

☆ Save 90 Cite Cited by 20 Related articles All 7 versions

Loop closing detection in RGB-D SLAM combining appearance and geometric constraints

H. Zhang, Y. Liu, J. Tan - Sensors, 2015 - mdpi.com

A kind of multi feature points matching algorithm fusing local geometric constraints is proposed for the purpose of quickly loop closing detection in RGB-D Simultaneous ...

☆ Save 90 Cite Cited by 24 Related articles All 9 versions

[PDF] ias.ac.in

[PDF] upenn.edu

[PDF] academia.edu

[PDF] academia.edu

[PDF] archive.org

[PDF] researchgate.net

[PDF] researchgate.net

[PDF] academia.edu

[HTML] mdpi.com

< Google >

Previous 1 2 3 4 5 6 7 8 9 10 Next

Privacy Terms Help

Avidbots Ex1008
Page 173 of 545

Google Scholar

Page 7 of about 126 results (0.03 sec)

My profile

My library

Articles

Any time

Since 2025

Since 2024

Since 2021

Custom range

2008 — 2015

Search

Citations

2012 2013 2014 2015

Sort by relevance

Sort by date

Create alert

Robust and efficient robotic mapping

Search within citing articles

Efficient implementation for block matrix operations for nonlinear least squares problems in robotic applications

L Polak, M Solomy, V Ila, P Smrz, ... - 2013 IEEE International ..., 2013 - [ieeexplore.ieee.org](#)

A large number of robotic, computer vision and computer graphics applications rely on efficiently solving the associated sparse linear systems. Simultaneous localization and ...

☆ Save 00 Cite Cited by 27 Related articles All 3 versions

Towards persistent localization and mapping with a continuous appearance-based topology

W Maddam, M Mafford, G Wyeth - Robotics: Science and Systems, 2013 - [books.google.com](#)

Appearance-based localization can provide loop closure detection at vast scales regardless of accumulated metric error. However, the computation time and memory requirements of ...

☆ Save 00 Cite Cited by 22 Related articles All 19 versions

Robust pose graph optimization using stochastic gradient descent

J Wang, E Olson - 2014 IEEE international conference on ..., 2014 - [ieeexplore.ieee.org](#)

Robust SLAM methods can allow robots to recover correct maps even in the presence of incorrect loop closures. While these approaches improve robustness to outliers, they are ...

☆ Save 00 Cite Cited by 17 Related articles All 6 versions

Toward cognitive robotics

J E Latif - Unmanned Systems Technology XI, 2009 - [spiedigitallibrary.org](#)

Our long-term goal is to develop autonomous robotic systems that have the cognitive abilities of humans, including communication, coordination, adapting to novel situations, and ...

☆ Save 00 Cite Cited by 26 Related articles All 10 versions

Robust sensor characterization via max-mixture models: GPS sensors

B Martini, E Olson - 2013 IEEE/RSJ International Conference ..., 2013 - [ieeexplore.ieee.org](#)

Large position errors plague GNSS-based sensors (eg, GPS) due to poor satellite configuration and multipath effects, resulting in frequent outliers. Due to quadratic cost ...

☆ Save 00 Cite Cited by 17 Related articles All 6 versions

Simultaneous parameter calibration, localization, and mapping for robust service robotics

R Kummerle, G Grisetti, C Stachniss, ... - Robotics and its ..., 2011 - [ieeexplore.ieee.org](#)

Modern service robots are designed to be deployed by end-users and not to be monitored by experts during operation. Most service robotics applications require reliable navigation ...

☆ Save 00 Cite Cited by 12 Related articles All 12 versions

Autonomous aerial manipulation using a quadrotor

V Ghadikolaei - 2011 - [search.proquest.com](#)

This paper presents an implementation of autonomous indoor aerial gripping using a low-cost, custom-built quadrotor. Such research extends the typical functionality of micro air ...

☆ Save 00 Cite Cited by 16 Related articles All 5 versions 00

Place recognition using near and far visual information

C Cadena, J McDonald, J Leonard, J Neira - IFAC Proceedings Volumes, 2011 - Elsevier

In this paper we show how to carry out robust place recognition using both near and far information provided by a stereo camera. Visual appearance is known to be very useful in ...

☆ Save 00 Cite Cited by 17 Related articles All 14 versions

Hybrid Hessians for flexible optimization of pose graphs

MK Grimes, D Anguelov, ... - 2010 IEEE/RSJ ..., 2010 - [ieeexplore.ieee.org](#)

We present a novel "hybrid Hessian" six-degrees-of-freedom simultaneous localization and mapping (SLAM) algorithm. Our method allows for the smooth trade-off of accuracy for ...

☆ Save 00 Cite Cited by 12 Related articles All 7 versions

Robust simultaneous localization and mapping via information matrix estimation

MC Graham, JP How - 2014 IEEE/ION Position, Location and ..., 2014 - [ieeexplore.ieee.org](#)

One of the major challenges for current SLAM systems is the impact of outliers and incorrectly modeled measurement noise on the final mapping solution. Outliers, such as ...

☆ Save 00 Cite Cited by 9 Related articles All 3 versions

[PDF] academia.edu

[PDF] qut.edu.au

[PDF] archive.org

[PDF] spiedigitallibrary.org

[PDF] archive.org

[PDF] uni-freiburg.de

[PDF] usu.edu

[PDF] maynoothuniversity.ie

[PDF] academia.edu

< Google >

Previous 2 3 4 5 6 7 8 9 10 11 Next

Privacy Terms Help

Avidbots Ex1008
Page 174 of 545

Google Scholar

Articles

Page 8 of about 126 results (0.03 sec)

My profile

My library

Any time

Since 2025

Since 2024

Since 2021

Custom range

2008

2015

Search

Citations

2012

2013

2014

2015

Sort by relevance

Sort by date

Create alert

Robust and efficient robotic mapping

Search within citing articles

Incremental Cholesky factorization for least squares problems in robotics

L Polak · M Solomy · V Ila · P Smrz · P Zemaník · IFAC Proceedings Volumes, 2013 · Elsevier

Online applications in robotics, computer vision, and computer graphics rely on efficiently solving the associated nonlinear systems every step. Iteratively solving the non-linear system ...

☆ Save 99 Cite Cited by 8 Related articles All 2 versions

[PDF] academia.edu

[PDF] State estimation and optimization for mobile robot navigation

R Kummert · 2013 · freidok.uni-freiburg.de

Robust autonomous navigation is a key feature of a mobile robot realizing services such as transportation, cleaning, search and rescue, and surveillance. In addition to that, navigation ...

☆ Save 99 Cite Cited by 17 Related articles All 4 versions

[PDF] uni-freiburg.de

Fuzzy ART-based place recognition for visual loop closure detection

K Rebal · O Azouaoui · N Achour · Biological cybernetics, 2013 · Springer

The automatic place recognition problem is one of the key challenges in SLAM approaches for loop closure detection. Most of the appearance-based solutions to this problem share the ...

☆ Save 99 Cite Cited by 10 Related articles All 11 versions

Localisation et cartographie simultanées pour un robot mobile équipé d'un laser à balayage

CoreSLAM

O El Hamzaoui · 2012 · pastel.hal.science

La thématique de la navigation autonome constitue l'un des principaux axes de recherche dans le domaine des véhicules intelligents et des robots mobiles. Dans ce contexte, on ...

☆ Save 99 Cite Cited by 14 Related articles All 5 versions

[PDF] hal.science

[HTML] Mapping, localization, and trajectory estimation with mobile robots using long-range passive RFID

P Vorst · 2011 · academia.edu

Abstract: This thesis addresses the use of radio-frequency identification (RFID) for three fundamental tasks in mobile robotics: mapping, self-localization, and trajectory estimation ...

☆ Save 99 Cite Cited by 7 Related articles

[HTML] academia.edu

[PDF] A full 2D/3D GraphSLAM system for globally consistent mapping based on manifolds

F Neuhaus · 2012 · fneuhaus.de

Robotics research today is primarily about enabling autonomous, mobile robots to seamlessly interact with arbitrary, previously unknown environments. One of the most basic ...

☆ Save 99 Cite Cited by 7 Related articles All 5 versions

[PDF] fneuhaus.de

[PDF] Generalized Simultaneous Localization and Mapping (SLAM) on Graphs with Multimodal Probabilities and Hyperedges

M Pfingsthorn · 2014 · researchgate.net

Abstract Simultaneous Localization and Mapping (SLAM) is one of the cornerstones of robotics research. Any mobile robot which is to operate in a previously unknown area ...

☆ Save 99 Cite Cited by 3 Related articles All 3 versions

[PDF] researchgate.net

Registration of multiple ToF camera point clouds

T Hedlund · 2010 · diva-portal.org

Buildings, maps and objects et cetera, can be modeled using a computer or reconstructed in 3D by data from different kinds of cameras or laser scanners. This thesis concerns the latter ...

☆ Save 99 Cite Cited by 5 Related articles All 3 versions

[PDF] diva-portal.org

[BOOK] Efficient ranging-sensor navigation methods for indoor aircraft

DM Sobers Jr · 2010 · search.proquest.com

Abstract Unmanned Aerial Vehicles (UAVs) are often used for reconnaissance, search and rescue, damage assessment, exploration, and other tasks that are dangerous or ...

☆ Save 99 Cite Cited by 5 Related articles All 2 versions

[PDF] gatech.edu

[BOOK] Real time multi-sensor data acquisition and processing for a road mapping system

X Luo · 2014 · search.proquest.com

Road assets condition has a critical impact on road safety and efficiency. Accurate and efficient monitoring and management of road assets is a challenge. This research is focused ...

☆ Save 99 Cite Cited by 3 Related articles All 3 versions

[PDF] proquest.com

< Google >

Previous 3 4 5 6 7 8 9 10 11 12 Next

Privacy Terms Help

Avidbots Ex1008
Page 175 of 545

Google Scholar

Page 9 of about 126 results (0.04 sec)

My profile

My library

Articles

Any time
Since 2025
Since 2024
Since 2021
Custom range.
2008 — 2015
Search

Citations

2012201320142015

Sort by relevance
Sort by date

Create alert

Robust and efficient robotic mapping

Search within citing articles

Fast and accurate PoseSLAM by combining relative and global state spaces

B Peasley, S Blachfeld - 2014 IEEE International Conference ..., 2014 - IEEE explore.IEEE.org

We revisit the question of state space in the context of performing loop closure. Although a relative state space has been previously discounted, we show that such a state space is ...

☆ Save 00 Cite Cited by 3 Related articles All 9 versions

[CITATION] Implicit data association from spectrally clustered local matches

E Olson - Inside Data Association, held in RSS 2008, 2008

☆ Save 00 Cite Cited by 4 Related articles All 3 versions

A hierarchical iterative closest point algorithm for simultaneous localization and mapping of mobile robot

QZ Zhang, YL Zhou - Proceedings of the 10th World Congress ..., 2012 - IEEE explore.IEEE.org

Simultaneous localization and mapping (SLAM) problem of a mobile robot is studied in this paper. An improved particle filters approach is adopted to reduce the number of particles. A ...

☆ Save 00 Cite Cited by 3 Related articles All 2 versions

[CITATION] Machinery Perception and the Application to Exoskeleton Robots

Y Li, Y Wang, J Sun, F Shuang, Y Ge - CiteSeer

☆ Save 00 Cite Cited by 2 Related articles

A multi-hypothesis constraint network optimizer for maximum likelihood mapping

DL Rizzini, S Caselli - 2011 IEEE International Conference on ..., 2011 - IEEE explore.IEEE.org

Loop closure is one of the most difficult task in localization and mapping problems since it suffers from perceptual aliasing. Multi-hypothesis topological SLAM algorithms have been ...

☆ Save 00 Cite Cited by 2 Related articles All 6 versions

[PDF] Handling local and global ambiguities via a generalized graph SLAM framework based on multimodal and hyperedge constraints

M Pfingsthorn, A Birk - ICRA Workshop on robust and multimodal ..., 2013 - tu-chemnitz.de

Graph-based Simultaneous Localization and Mapping (SLAM) has experienced a recent surge towards robust methods. These methods take the combinatorial aspect of data ...

☆ Save 00 Cite Cited by 2 Related articles All 2 versions 00

[CITATION] 一种广泛适用的激光雷达数据特征提取方法

李阳阳, 盛庆虎 - 机器人, 2010

☆ Save 00 Cite Cited by 5 Related articles All 2 versions

Attentional behaviors for environment modeling by a mobile robot

P Bachiller, P Bustos, Lj Manso - Stereo Vision, 2011 - books.google.com

Building robots capable of interacting in an effective and autonomous way with their environments requires to provide them with the ability to model the world. That is to say, the ...

☆ Save 00 Cite Cited by 2 Related articles All 14 versions 00

[PDF] Feature-based 3D perception for mobile robots

B Steder - 2013 - freidok.uni-freiburg.de

Perception is one of the key topics in robotics research. It is about the processing of external sensor data and its interpretation. Strong perceptual abilities are a basic requirement for a ...

☆ Save 00 Cite Cited by 5 Related articles All 5 versions 00

UPenn Multi-Robot Unmanned Vehicle System (MAGIC)

DD Lee - 2014 - apps.dtic.mil

In this report, we describe the technical approach and algorithms that have been used by the University of Pennsylvania for multi-robot perception, planning, and control in the context of ...

☆ Save 00 Cite Cited by 1 Related articles All 2 versions 00

[PDF] clemson.edu

[PDF] academia.edu

[PDF] tu-chemnitz.de

[PDF] aston.ac.uk

[PDF] uni-freiburg.de

[PDF] dtic.mil

< Google >

Previous 3 4 5 6 7 8 9 10 11 12 Next

Privacy Terms Help

Avidbots Ex1008
Page 176 of 545

Google Scholar

Articles

Page 10 of about 126 results (0.04 sec)

My profile

My library

Any time

Since 2025

Since 2024

Since 2021

Custom range

2008

—

2015

Search

Citations

2012

2013

2014

2015

Sort by relevance

Sort by date

Create alert

Robust and efficient robotic mapping

Search within citing articles

Scalable online decentralized smoothing and mapping

AG Cunningham - 2014 - repository.gatech.edu

Many applications for field robots can benefit from large numbers of robots, especially applications where the objective is for the robots to cover or explore a region. A key enabling ...

☆ Save 90 Cite Cited by 1 Related articles All 9 versions

Consolidation de relevés laser d'intérieurs construits: pour une approche probabiliste initialisée par géolocalisation

J-F Huile - 2013 - theses.hal.science

La préparation d'interventions de maintenance dans les installations industrielles a d'ordinaire recours à des outils d'étude, de modélisation et de simulation basés sur ...

☆ Save 90 Cite Cited by 4 Related articles All 8 versions

An Incremental Hybrid Approach to Indoor Modeling.

MA Gutiérrez, P Bachiller, L Manso, P Bustos, P Nimez - ECMR, 2011 - academia.edu

Most of mobile robots basic functions are highly dependent on a model of their environment. Proper modeling is crucial for tasks such as local navigation, localization or route planning ...

☆ Save 90 Cite Cited by 1 Related articles All 9 versions

Contributions à la localisation de personnes par vision monoculaire embarquée

W Eloumi - 2012 - theses.hal.science

Une des techniques alternatives au GPS pour le développement d'un système d'assistance à la navigation piédestre en milieux urbains est la vision embarquée. La localisation du ...

☆ Save 90 Cite Cited by 3 Related articles All 4 versions

Highly scalable appearance-only SLAM-FAB-MAP 2.0

M Cummins, P Newman - 2010 - direct.mit.edu

We describe a new formulation of appearance-only SLAM suitable for very large scale navigation. The system navigates in the space of appearance, assigning each new ...

☆ Save 90 Cite Cited by 430 Related articles All 4 versions

Map building, localization and exploration for multi-robot systems

JÁ Castellanos Gómez, G Grisetti - zaguan.unizar.es

The idea of having robots performing the task for which they have been designed completely autonomously and interacting with the environment has been the main objective since the ...

☆ Save 90 Cite Related articles All 2 versions

Gutachter: Prof. Dr. Frank Kirchner Prof. Dr. Udo Frese

J Hertzberg, R Drechsler, Y Kassahun

☆ Save 90 Cite Related articles

JOMAR: Joint Operations with Mobile Autonomous Robots

E Olson - 2015 - apps.dtic.mil

Under this grant, we formulated and implemented a variety of novel algorithms that address core problems in multi-robot systems. These contributions roughly fall into two categories ...

☆ Save 90 Cite Related articles

Map building, localization and exploration for multi-robot systems

MT Lázaro Grañón, JÁ Castellanos Gómez, G Grisetti - zaguan.unizar.es

The idea of having robots performing the task for which they have been designed completely autonomously and interacting with the environment has been the main objective since the ...

☆ Save 90 Cite Related articles

Communication aware mobile robot teams

J Stephan - 2015 - search.proquest.com

The type of scenarios that could benefit from a team of robots that are able to self configure into an ad-hoc multi-hop mobile communication network while completing a task in an ...

☆ Save 90 Cite Related articles All 5 versions

< Google >

Previous 3 4 5 6 7 8 9 10 11 12 Next

Privacy

Terms

Help

Avidbots Ex1008
Page 177 of 545

Articles

Page 11 of about 126 results (0.06 sec)

My profile

My library

Any time

Since 2025

Since 2024

Since 2021

Custom range.

2008

2015

Search

Citations

2012 2013 2014 2015

Sort by relevance

Sort by date

Create alert

Robust and efficient robotic mapping

Search within citing articles

[PDF] Accurate and compact surface models for mobile robots

M.Rubels - 2014 - researchgate.net

Mobile robots require environment models to perform autonomous navigation tasks, like the delivery of goods, cleaning up homes, surveillance and environmental monitoring. Besides ...

☆ Save 00 Cite Related articles All 4 versions 00

[PDF] researchgate.net

Exploration and Mapping with Autonomous Robot Teams

E.AndrewRichardson - academia.edu

The potential impact of autonomous robotics is magnified when those robots are deployed in teams: a team of cooperating robots can greatly increase the effectiveness of a human ...

☆ Save 00 Cite Related articles All 3 versions 00

[PDF] academia.edu

Flexible-Cost SLAM

MK.Grimes - 2012 - search.proquest.com

Flexible-Cost SLAM Page 1 Flexible-Cost SLAM by Matthew Koichi Grimes A dissertation submitted in partial fulfillment of the requirements for the degree of Doctor of Philosophy ...

☆ Save 00 Cite Related articles All 3 versions 00

[PDF] spiedigitallibrary.org

Hierarchical loop detection for mobile outdoor robots

D.Lang, C.Winkens, M.Hasalich - 2012 - spiedigitallibrary.org

Loop closing is a fundamental part of 3D simultaneous localization and mapping (SLAM) that can greatly enhance the quality of long term mapping. It is essential for the creation of ...

☆ Save 00 Cite Related articles All 5 versions 00

[PDF] spiedigitallibrary.org

Heuristička metoda zatvaranja petlje za 6D SLAM velikih dimenzija

J.Sprickerhof, A.Nichter, K.Lingemann - 2011 - hrcak.srce.hr

Sažetak Članak prikazuje novu heuristiku čija je svrha korekcija estimacije pozicija obližnja 3D lasera nakon zatvaranja petlje u SLAM-u. U suprotnosti s postojećim pristupima ...

☆ Save 00 Cite Related articles 00

[PDF] srce.hr

Real-Time Target Following Using an Unmanned Rotorcraft with a Laser Rangefinder

BS.Pircock - 2012 - search.proquest.com

Micro-unmanned aerial rotorcraft are quickly gaining acceptance as indoor platforms for performing stealth, surveillance, and rescue and reconnaissance missions. These rotorcraft ...

☆ Save 00 Cite Related articles All 2 versions 00

[PDF] byu.edu

Long term appearance-based mapping with vision and laser

R.Paul - 2012 - ox.ac.uk

This thesis is about appearance-based topological mapping for mobile robots using vision and laser. Our goal is life-long continual operation in outdoor unstructured workspaces. We ...

☆ Save 00 Cite Related articles All 2 versions 00

[PDF] ox.ac.uk

Progress toward multi-robot reconnaissance and the MAGIC 2010 competition

A.Hsieh, S.Lacroix - 2012 - deepblue.lib.umich.edu

Tasks like search and rescue and urban reconnaissance benefit from large numbers of robots working together, but high levels of autonomy are needed to reduce operator ...

☆ Save 00 Cite Related articles 00

[PDF] umich.edu

Approches 2D/2D pour le SFM à partir d'un réseau de caméras asynchrones

R.Mhini - 2015 - theses.hal.science

Les systèmes d'aide à la conduite et les travaux concernant le véhicule autonome ont atteint une certaine maturité durant ces dernières années grâce à l'utilisation de technologies ...

☆ Save 00 Cite Cited by 1 Related articles All 6 versions 00

[PDF] hal.science

on Mobile Robots ECMR'2011

A.J.Li, J.Lienhai, T.Duckett - 2011 - researchgate.net

Most of mobile robots basic functions are highly dependent on a model of their environment. Proper modelling is crucial for tasks such as local navigation, localization or route planning ...

☆ Save 00 Cite Related articles 00

[PDF] researchgate.net

< Google >

Previous 3 4 5 6 7 8 9 10 11 12 Next

Privacy Terms Help

Avidbots Ex1008
Page 178 of 545

Articles

Page 12 of about 126 results (0.04 sec)

My profile

My library

Any time

Since 2025

Since 2024

Since 2021

Custom range.

2008

2015

Search

Citations

2012

2013

2014

2015

Sort by relevance

Sort by date

Create alert

Robust and efficient robotic mapping

Search within citing articles

Body-relative navigation guidance using uncalibrated cameras

Q Koch · 2010 · dspace.mit.edu

The ability to navigate through the world is an essential capability to humans. In a variety of situations, people do not have the time, the opportunity or the capability to learn the layout of ...

☆ Save

🔗 Cite

Related articles

All 4 versions

🔗

PDF] mit.edu

Localization with Improved Proposals

S Oswald · C Gonsior · researchgate.net

Solving localization and navigation tasks reliably is an essential objective for autonomous mobile systems and robots. A popular technique for estimating the robot's pose is ...

☆ Save

🔗 Cite

Related articles

All 3 versions

🔗

PDF] researchgate.net

Trajectory-based Comparison of SLAM Algorithms

WBCSG Grisetti, B Steder, RKCDM Ruhnke · gki.informatik.uni-freiburg.de

In this paper, we address the problem of creating an objective benchmark for comparing SLAM approaches. We propose a framework for analyzing the results of SLAM approaches ...

☆ Save

🔗 Cite

Related articles

All 5 versions

🔗

PDF] uni-freiburg.de

Life-long Map Learning for Graph-based Simultaneous Localization and Mapping

W Burgard · 2009 · ais.informatik.uni-freiburg.de

Maps of the environment are needed for a wide range of robotic applications, such as delivery, surveillance, and pursuit-evasion. The ability to generate a map of the environment ...

☆ Save

🔗 Cite

Related articles

All 2 versions

🔗

PDF] uni-freiburg.de

Large Scale Graph-Based SLAM using Aerial Images as Prior Information

RKISD Dornhege, AKGGW Burgard · gki.informatik.uni-freiburg.de

Robot navigation within large-scale scenarios in general requires a globally consistent map of the environment, eg, for reaching target locations safely. This problem has been ...

☆ Save

🔗 Cite

Related articles

All 2 versions

🔗

PDF] uni-freiburg.de

Localisation et cartographie simultanées pour un robot mobile de type voiture

K Rebai · 2015 · ccdz.cerist.dz

Résumé L'approche de SLAM proposée, dans cette thèse, est inspirée de la neurobiologie et particulièrement des caractéristiques des cellules de vue. Elle consiste à construire une ...

☆ Save

🔗 Cite

Related articles

🔗

(CITATION) 基于子图分割和自适应噪声方差的 2D 移动机器人定位方法

张晨, 刘国良, 李海君, 侯发峰 · 计算机科学, 2014

☆ Save

🔗 Cite

Related articles

All 2 versions

🔗

移動機器人在稀疏環境中之同時定位與建圖

陳建華 · 國立臺灣國際大學電機工程學系學位論文, 2014 · aimilibrary.com

此論文研究移動式機器人在稀疏環境中同時定位與建圖問題, 採用分佈式圖增型卡爾曼濾波之粒子濾波器估計機器人位置及周遭環境特徵並建立地圖。在目前FastSLAM 算法是基於把聯合 ...

☆ Save

🔗 Cite

Related articles

All 3 versions

🔗

probabilistic mapping with mobile robots in structured environments

P Puenti Yusty · 2012 · oa.upm.es

La mayor parte de los entornos diseñados por el hombre presentan características geométricas específicas. En ellos es frecuente encontrar formas poligonales, rectangulares ...

☆ Save

🔗 Cite

Related articles

All 3 versions

🔗

PDF] upm.es

Computation and Time constraints in Localization and Mapping Problems

D Lodi Rizzi · 2009 · repository.unipr.it

Research on simultaneous localization and mapping problems has been extensively carried out by robotics community in the last decade and several subproblems—like data ...

☆ Save

🔗 Cite

Related articles

🔗

PDF] unipr.it

< Google >

Previous 3 4 5 6 7 8 9 10 11 12 Next

Privacy Terms Help

Avidbots Ex1008
Page 179 of 545

Attachment G

On measuring the accuracy of SLAM algorithms

Rainer Kümmerle · Bastian Steder · Christian Dornhege · Michael Ruhnke ·
Giorgio Grisetti · Cyrill Stachniss · Alexander Kleiner

Received: 10 February 2008 / Accepted: 18 August 2009 / Published online: 30 September 2009
© Springer Science+Business Media, LLC 2009

Abstract In this paper, we address the problem of creating an objective benchmark for evaluating SLAM approaches. We propose a framework for analyzing the results of a SLAM approach based on a metric for measuring the error of the corrected trajectory. This metric uses only relative relations between poses and does not rely on a global reference frame. This overcomes serious shortcomings of approaches using a global reference frame to compute the error. Our method furthermore allows us to compare SLAM approaches that use different estimation techniques or different sensor modalities since all computations are made based on the corrected trajectory of the robot.

R. Kümmerle (✉) · B. Steder · M. Ruhnke · G. Grisetti ·
C. Stachniss
Dept. of Computer Science, University of Freiburg,
Georges Köhler Allee 79, 79110 Freiburg, Germany
e-mail: kummerle@informatik.uni-freiburg.de

B. Steder
e-mail: steder@informatik.uni-freiburg.de

M. Ruhnke
e-mail: ruhnke@informatik.uni-freiburg.de

G. Grisetti
e-mail: grisetti@informatik.uni-freiburg.de

C. Stachniss
e-mail: stachnis@informatik.uni-freiburg.de

C. Dornhege · A. Kleiner
Dept. of Computer Science, University of Freiburg,
Georges Köhler Allee 52, 79110 Freiburg, Germany

C. Dornhege
e-mail: dornhege@informatik.uni-freiburg.de

A. Kleiner
e-mail: kleiner@informatik.uni-freiburg.de

We provide sets of relative relations needed to compute our metric for an extensive set of datasets frequently used in the robotics community. The relations have been obtained by manually matching laser-range observations to avoid the errors caused by matching algorithms. Our benchmark framework allows the user to easily analyze and objectively compare different SLAM approaches.

Keywords SLAM · Mapping accuracy · Benchmarking

1 Introduction

Models of the environment are needed for a wide range of robotic applications including transportation tasks, guidance, and search and rescue. Learning maps has therefore been a major research focus in the robotics community in the last decades. Robots that are able to acquire an accurate model of their environment are regarded as fulfilling a major precondition of truly autonomous agents.

In the literature, the mobile robot mapping problem under pose uncertainty is often referred to as the *simultaneous localization and mapping* (SLAM) or *concurrent mapping and localization* (CML) problem (Smith and Cheeseman 1986; Dissanayake et al. 2000; Gutmann and Konolige 1999; Hähnel et al. 2003; Montemerlo et al. 2003; Thrun 2001; Leonard and Durrant-Whyte 1991). SLAM is considered to be a complex problem because to localize itself a robot needs a consistent map and for acquiring the map the robot requires a good estimate of its location. This mutual dependency among the pose and the map estimates makes the SLAM problem hard and requires searching for a solution in a high-dimensional space.

Whereas dozens of different techniques to tackle the SLAM problem have been presented, there is no gold standard for comparing the results of different SLAM algo-

ri thms. In the community of feature-based estimation techniques, researchers often measure the distance or Mahalanobis distance between the estimated landmark location and the true location (if this information is available). As we will illustrate in this paper, comparing results based on an absolute reference frame can have shortcomings. In the area of grid-based estimation techniques, people often use visual inspection to compare maps or overlays with blueprints of buildings. This kind of evaluation becomes more and more difficult as new SLAM approaches show increasing capabilities and thus large scale environments are needed for evaluation. In the community, there is a strong need for methods allowing meaningful comparisons of different approaches. Ideally, such a method is capable of performing comparisons between mapping systems that apply different estimation techniques and operate on different sensing modalities. We argue that meaningful comparisons between different SLAM approaches require a common performance measure (metric). This metric should enable the user to compare the outcome of different mapping approaches when applying them on the same dataset.

In this paper, we propose a novel technique for comparing the output of SLAM algorithms. We aim to establish a benchmark that allows for objectively measuring the performance of a mapping system. We propose a metric that operates only on relative geometric relations between poses along the trajectory of the robot. Our approach allows for making comparisons even if a perfect ground truth information is not available. This enables us to present benchmarks based on frequently used datasets in the robotics community such as the MIT Killian Court or the Intel Research Lab dataset. The disadvantage of our method is that it requires manual work to be carried out by a human that knows the topology of the environment. The manual work, however, has to be done only once for a dataset and then allows other researchers to evaluate their methods easily. In this paper, we present manually obtained relative relations for different datasets that can be used for carrying out comparisons. We furthermore provide evaluations for the results of three different mapping techniques, namely scan-matching, SLAM using Rao-Blackwellized particle filter (Grisetti et al. 2007b; Stachniss et al. 2007b), and a maximum likelihood SLAM approach based on the graph formulation (Grisetti et al. 2007c; Olson 2008).

The remainder of this paper is organized as follows. First, we discuss related work in Sect. 2 and present the proposed metric based on relative relations between poses along the trajectory of the robot in Sect. 3. Then, in Sects. 4 and 5 we explain how to obtain such relations in practice. In Sect. 6, we briefly discuss how to benchmark if the tested SLAM system does not provide pose estimates. Next, in Sect. 7 we provide a brief overview of the datasets used for benchmarking and in Sect. 8 we present our experiments which illus-

trate different properties of our method and we give benchmark results for three existing SLAM approaches.

2 Related work

Learning maps is a frequently studied problem in the robotics literature. Mapping techniques for mobile robots can be classified according to the underlying estimation technique. The most popular approaches are extended Kalman filters (EKF) (Leonard and Durrant-Whyte 1991; Smith et al. 1990), sparse extended information filters (Eustice et al. 2005a; Thrun et al. 2004), particle filters (Montemerlo et al. 2003; Grisetti et al. 2007b), and least square error minimization approaches (Lu and Milios 1997; Frese et al. 2005; Gutmann and Konolige 1999; Olson et al. 2006). For some applications, it might even be sufficient to learn local maps only (Hermosillo et al. 2003; Thrun et al. 2006; Yguel et al. 2007).

The effectiveness of the EKF approaches comes from the fact that they estimate a fully correlated posterior about landmark maps and robot poses. Their weakness lies in the strong assumptions that have to be made on both, the robot motion model and the sensor noise. If these assumptions are violated the filter is likely to diverge (Julier et al. 1995; Uhlmann 1995).

Thrun et al. (2004) proposed a method to correct the poses of a robot based on the inverse of the covariance matrix. The advantage of sparse extended information filters (SEIFs) is that they make use of the approximative sparsity of the information matrix. Eustice et al. (2005a) presented a technique that more accurately computes the error-bounds within the SEIF framework and therefore reduces the risk of becoming overly confident.

Dellaert and colleagues proposed a smoothing method called square root smoothing and mapping (SAM) (Dellaert 2005; Kaess et al. 2007; Ranganathan et al. 2007). It has several advantages compared to EKF-based solutions since it better covers the non-linearities and is faster to compute. In contrast to SEIFs, it furthermore provides an exactly sparse factorization of the information matrix. In addition to that, SAM can be applied in an incremental way (Kaess et al. 2007) and is able to learn maps in 2D and 3D.

Frese's TreeMap algorithm (Frese 2006) can be applied to compute nonlinear map estimates. It relies on a strong topological assumption on the map to perform sparsification of the information matrix. This approximation ignores small entries in the information matrix. In this way, Frese is able to perform an update in $\mathcal{O}(\log n)$ where n is the number of features.

An alternative approach to find maximum likelihood maps is the application of least square error minimization. The idea is to compute a network of constraints given the

sequence of sensor readings. It should be noted that our approach for evaluating SLAM methods presented in this paper is highly related to this formulation of the SLAM problem.

Lu and Milios (1997) introduced the concept of graph-based or network-based SLAM using a kind of brute force method for optimization. Their approach seeks to optimize the whole network at once. Gutmann and Konolige (1999) proposed an effective way for constructing such a network and for detecting loop closures while running an incremental estimation algorithm. Duckett et al. (2002) propose the usage of Gauss-Seidel relaxation to minimize the error in the network of relations. To make the problem linear, they assume knowledge about the orientation of the robot. Frese et al. (2005) propose a variant of Gauss-Seidel relaxation called multi-level relaxation (MLR). It applies relaxation at different resolutions. MLR is reported to provide very good results in flat environments especially if the error in the initial guess is limited.

Olson et al. (2006) presented an optimization approach that applies stochastic gradient descent for resolving relations in a network efficiently. Extensions of this work have been presented by Grisetti et al. (2007a, 2007c). Most approaches to graph-based SLAM such as the work of Olson et al., Grisetti et al., Frese et al., and others focus on computing the best map and assume that the relations are given. The ATLAS framework (Bosse et al. 2003), hierarchical SLAM (Estrada et al. 2005), or the work of Nüchter et al. (2005), for example, can be used to obtain the constraints. In the graph-based mapping approach used in this paper, we followed the work of Olson (2008) to extract constraints and applied (Grisetti et al. 2007c) for computing the minimal error configuration.

Activities related to performance metrics for SLAM methods, such as the work described in this paper, can roughly be divided into three major categories: First, competition settings where robot systems are competing within a defined problem scenario, such as playing soccer, navigating through a desert, or searching for victims. Second, collections of publicly available datasets that are provided for comparing algorithms on specific problems. Third, related publications that introduce methodologies and scoring metrics for comparing different methods.

The comparison of robots within benchmarking scenarios is a straight-forward approach for identifying specific system properties that can be generalized to other problem types. For this purpose numerous robot competitions have been initiated in the past, evaluating the performance of cleaning robots (EPFL and IROS 2002), robots in simulated Mars environments (ESA 2008), robots playing soccer or rescuing victims after a disaster (RoboCup 2009), and cars driving autonomously in an urban area (Darpa 2007). However, competition settings are likely to generate additional noise due to differing hardware and software settings.

For example, when comparing mapping solutions in the RoboCup Rescue domain, the quality of maps generated using climbing robots can greatly differ from those generated on wheel-based robots operating in the plane. Furthermore, the approaches are often tuned to the settings addressed in the competitions.

Benchmarking of systems from datasets has reached a rather mature level in the vision community. There exist numerous data bases and performance measures, which are available via the Internet. Their purpose is to validate, for example, image annotation (Torralba et al. 2007), range image segmentation (Hoover et al. 1996), and stereo vision correspondence algorithms (Scharstein and Szeliski 2002). These image databases provide ground truth data (Torralba et al. 2007; Scharstein and Szeliski 2002), tools for generating ground truth (Torralba et al. 2007) and computing the scoring metric (Scharstein and Szeliski 2002), and an online ranking of results from different methods (Scharstein and Szeliski 2002) for direct comparison.

In the robotics community, there are some well-known web sites providing datasets (Howard and Roy 2003; Bonarini et al. 2006) and algorithms (Stachniss et al. 2007a) for mapping. However, they neither provide ground truth data nor recommendations on how to compare different maps in a meaningful way.

Some preliminary steps towards benchmarking navigation solutions have been presented in the past. Amigoni et al. (2007) presented a general methodology for performing experimental activities in the area of robotic mapping. They suggested a number of issues that should be addressed when experimentally validating a mapping method. For example, the mapping system should be applied to publicly available data, parameters of the algorithm should be clearly indicated (and also effects of their variations presented), as well as parameters of the map should be explained. When ground truth data is available, they suggest to utilize the Hausdorff metric for map comparison.

Wulf et al. (2008) proposed the idea of using manually supervised Monte Carlo Localization (MCL) for matching 3D scans against a reference map. They suggested that a reference map be generated from independently created CAD data, which can be obtained from the land registry office. The comparison between generated map and ground truth has been carried out by computing the Euclidean distance and angle difference of each scan, and plotting these over time. Furthermore, they provided standard deviation and maximum error of the track for comparisons. We argue that comparing the absolute error between two tracks might not yield a meaningful assertion in all cases as illustrated in the initial example in Sect. 3. This effect gets even stronger when the robot makes a small angular error especially in the beginning of the dataset (and when it does not return to this place again). Then, large parts or the overall map are likely

to be consistent, the error, however, will be huge. Therefore, the method proposed in this paper favors comparisons between relative poses along the trajectory of the robot. Based on the selection between which pose relations are considered, different properties can be highlighted.

Balaguer et al. (2007) utilize the USARSim robot simulator and a real robot platform for comparing different open source SLAM approaches. They demonstrated that maps resulting from processing simulator data are very close to those resulting from real robot data. Hence, they concluded that the simulator engine could be used for systematically benchmarking different approaches of SLAM. However, it has also been shown that noise is often but not always Gaussian in the SLAM context (Stachniss et al. 2007b). Gaussian noise, however, is typically used in most simulation systems. In addition to that, Balaguer et al. do not provide a quantitative measure for comparing generated maps with ground truth. As with many other approaches, their comparisons were carried out by visual inspection.

The paper presented here extends our work (Burgard et al. 2009) with a more detailed description of the approach, a technique for extracting relations from aerial images, and a significantly extended experimental evaluation.

3 Metric for benchmarking SLAM algorithms

In this paper, we propose a metric for measuring the performance of a SLAM algorithm *not by comparing the map itself but by considering the poses of the robot during data acquisition*. In this way, we gain two important properties: First, it allows us to compare the result of algorithms that generate different types of metric map representations, such as feature-maps or occupancy grid maps. Second, the method is invariant to the sensor setup of the robot. Thus, a result of a graph-based SLAM approach working on laser range data can be compared, for example, with the result of vision-based FastSLAM. The only property we require is that the SLAM algorithm estimates the trajectory of the robot given by a set of poses at which observations are made. All benchmark computations will be performed on this set.

3.1 A measure for benchmarking SLAM results

Let $x_{1:T}$ be the poses of the robot estimated by a SLAM algorithm from time step 1 to T . Let $x_{1:T}^*$ be the reference poses of the robot, ideally the true locations. A straightforward error metric could be defined as

$$\varepsilon(x_{1:T}) = \sum_{t=1}^T (x_t \ominus x_t^*)^2, \quad (1)$$

where $\oplus$ is the standard motion composition operator and $\ominus$ its inverse. Let $\delta_{i,j} = x_j \ominus x_i$ be the relative transformation that moves the node x_i onto x_j and accordingly $\delta_{i,j}^* =$

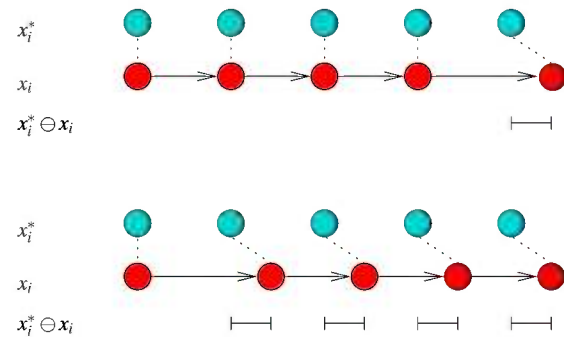


Fig. 1 This figure illustrates a simple example where the metric in Eq. 1 fails. The blue (light gray) circles show the reference positions of the robot $\{x_i^*\}$ while the red (dark gray) circles show the estimated positions of the robot $\{x_i\}$. The correspondence between the estimated locations and the ground truth is shown with dashed lines, and the direction of motion of the robot is highlighted with arrows. In the situation shown in the upper part, the robot makes a small mistake at the end of the path. This results in a small error. Conversely, in the situation illustrated on the bottom part of the figure the robot makes a small error of the same entity, but at the beginning of the travel, thus resulting in a much bigger global error

$x_j^* \ominus x_i^*$. Equation 1 can be rewritten as

$$\varepsilon(x_{1:T}) = \sum_{t=1}^T ((x_1 \oplus \delta_{1,2} \oplus \dots \oplus \delta_{t-1,t}) \ominus (x_1^* \oplus \delta_{1,2}^* \oplus \dots \oplus \delta_{t-1,t}^*))^2 \quad (2)$$

We claim that this metric is suboptimal for comparing the result of a SLAM algorithm. To illustrate this, consider the following 1D example in which a robot travels along a straight line. Let the robot make a translational error of e during the first motion, $\delta_{1,2} = \delta_{1,2}^* + e$, and perfect estimates at all other points in time $\delta_{t,t+1} = \delta_{t,t+1}^*$ for $t > 1$. Thus, the error according to Eq. 2, will be $T \cdot e$, since $\delta_{1,2}$ is contained in every pose estimate for $t > 1$. If, however, we estimate the trajectory backwards starting from x_T to x_1 or alternatively by shifting the whole map by e , we obtain an error of e only. This indicates, that such an error estimate is suboptimal for comparing the results of a SLAM algorithm. See also Fig. 1 for an illustration.

In the past, the so-called NEES measure proposed in Bar-Shalom et al. (2001) as

$$\varepsilon(x_{1:T}) = \sum_{t=1}^T (x_t - x_t^*)^T \Omega_t (x_t - x_t^*), \quad (3)$$

has often been used to evaluate the results of a SLAM approach (e.g., Eustice et al. 2005b). Here Ω_t represents the information matrix of the pose x_t . The NEES measure, however, suffers from a similar problem as Eq. 1 when computing ε . In addition to that, not all SLAM algorithms provide an estimate of the information matrix and thus cannot be compared based on Eq. 3.

Based on this experience, we propose a measure that considers the deformation energy that is needed to transfer the estimate into the ground truth. This can be done—similar to the ideas of the graph mapping introduced by Lu and Milios (1997)—by considering the nodes as masses and connections between them as springs. Thus, our metric is based on the *relative* displacement between robot poses. Instead of comparing x to x^* (in the global reference frame), we do the operation based on δ and δ^* as

$$\varepsilon(\delta) = \frac{1}{N} \sum_{i,j} \text{trans}(\delta_{i,j} \ominus \delta_{i,j}^*)^2 + \text{rot}(\delta_{i,j} \ominus \delta_{i,j}^*)^2, \quad (4)$$

where N is the number of relative relations and $\text{trans}(\cdot)$ and $\text{rot}(\cdot)$ are used to separate and weight the translational and rotational components. We suggest that both quantities be evaluated individually. In this case, the error (or transformation energy) in the above-mentioned example will be consistently estimated as the single rotational error no matter where the error occurs in the space or in which order the data is processed.

Our error metric, however, leaves open which relative displacements $\delta_{i,j}$ are included in the summation in Eq. 4. Using the metric and selecting relations are two related but different problems. Evaluating two approaches based on a different set of relative pose displacements will obviously result in two different scores. As we will show in the remainder of this section, the set δ and thus δ^* can be defined to highlight certain properties of an algorithm.

Note that some researchers prefer the absolute error (absolute value, not squared) instead of the squared one. We prefer the squared one since it derives from the motivation that the metric measures the energy needed to transform the estimated trajectory into ground truth. However, one can also use the metric using the non-squared error instead of the squared one. In the experimental evaluation, we actually provide both values.

3.2 Selecting relative displacements for evaluation

Benchmarks are designed to compare different algorithms. In the case of SLAM systems, however, the task the robot finally has to solve should define the required accuracy and this information should be considered in the measure.

For example, a robot generating blueprints of buildings should reflect the geometry of a building as accurately as possible. In contrast to that, a robot performing navigation tasks requires a map that can be used to robustly localize itself and to compute valid trajectories to a goal location. To carry out this task, it is sufficient in most cases that the map is topologically consistent and that its observations can be locally matched to the map, i.e., its spatial structure is correctly representing the environment. We refer to a map

having this property as being locally consistent. Figure 3 illustrates the concept of locally consistent maps which are suited for a robot to carry out navigation tasks.

By selecting the relative displacements $\delta_{i,j}$ used in Eq. 4 for a given dataset, the user can highlight certain properties and thus design a measure for evaluating an approach given the application in mind.

For example, by adding only known relative displacements between nearby poses based on visibility, a local consistency is highlighted. In contrast to that, by adding known relative displacements of far away poses, for example, provided by an accurate external measurement device or by background knowledge, the accuracy of the overall geometry of the mapped environment is enforced. In this way, one can incorporate additional knowledge (for example, that a corridor has a certain length and is straight) into the benchmark.

4 Obtaining reference relations in indoor environments

In practice, the key question regarding Eq. 4 is how to determine the *true relative displacements* between poses. Obviously, the true values are not available. However, we can determine close-to-true values by using the information recorded by the mobile robot and the background knowledge of the human recording the datasets, which, of course, involves manual work.

Please note, that the metric presented above is independent of the actual sensor used. In the remainder of this paper, however, we will concentrate on robots equipped with a laser range finder, since they are probably the most popular sensors in robotics at the moment. To evaluate an approach operating on a different sensor modality, one has two possibilities to generate relations. One way would be to temporarily mount a laser range finder on the robot and calibrate it in the robot coordinate frame. If this is not possible, one has to provide a method for accurately determining the relative displacements between two poses from which an observation has been taken that observes the same part of the space.

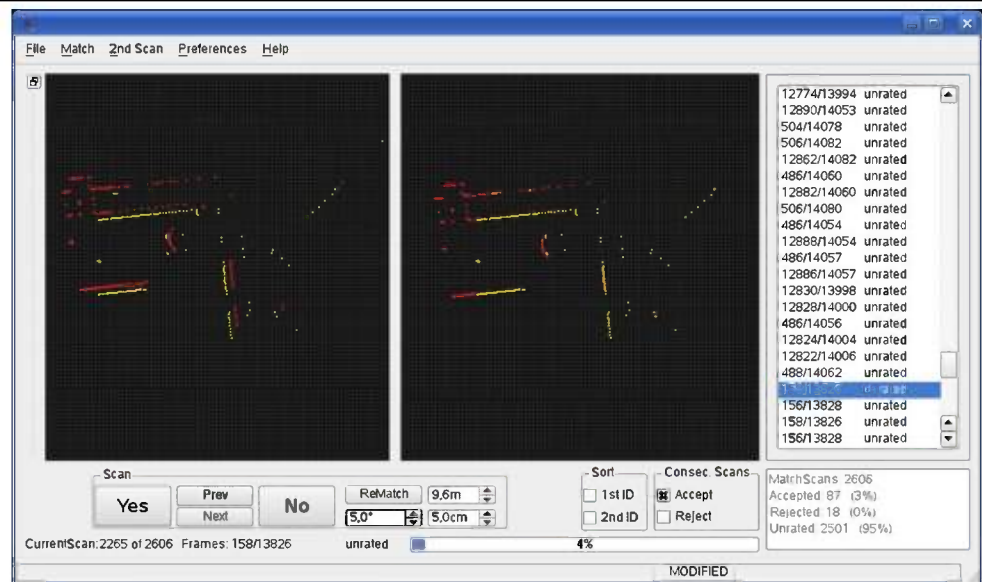
4.1 Initial guess

In our work, we propose the following strategy. First, one tries to find an initial guess about the relative displacement between poses. Based on the knowledge of the human, a wrong initial guess can be easily discarded since the human “knows” the structure of the environment. In a second step, a refinement is proposed based on manual interaction.

4.1.1 Symeo system

One way for obtaining good initial guesses with no or only very few interactions can be the use of the Symeo Positioning System LPR-B (Symeo GmbH 2008). It works similar to

Fig. 2 User interface for matching, accepting, and discarding pairs of observations



a local GPS system but indoors and can achieve a localization accuracy of around 5 cm to 10 cm. The problem is that such a system designed for industrial applications is typically not present at most robotics labs. If available, however, it is well suited for a rather accurate initial guess of the robot's position.

4.1.2 Initial estimate via SLAM approaches

In most cases, however, researchers in robotics will have SLAM algorithms at hand that can be used to compute an initial guess about the poses of the robot. In the recent years, several accurate methods have been proposed to serve as such a guess (see Sect. 2). By manually inspecting the estimates of the algorithm, a human can accept, refine, or discard a match and also add missing relations.

It is important to note that the output is not more than an initial guess and it is used to estimate the visibility constraints which will be used in the next step.

4.2 Manual matching refinement and rejection

Based on the initial guess about the position of the robot for a given time step, it is possible to determine which observations in the dataset should have covered the same part of the space or the same objects. For a laser range finder, this can easily be achieved. Between each visible pair of poses, one adds a relative displacement into a candidate set.

In the next step, a human processes the candidate set to eliminate wrong hypotheses by visualizing the observation in a common reference frame. This requires manual interaction but allows for eliminating wrong matches and outliers with high precision, since the user is able to incorporate his background knowledge about the environment.

Since we aim to find the best possible relative displacement, we perform a pair-wise registration procedure to refine the estimates of the observation registration method. It furthermore allows the user to manually adjust the relative offset between poses so that the pairs of observations fit better. Alternatively, the pair can be discarded.

This approach might sound labor-intensive but with an appropriate user interface, this task can be carried out without a large waste of resources. For example, for a standard dataset with 1700 relations, it took an unexperienced user approximately four hours to extract the relative translations that then served as the input to the error calculation. Figure 2 shows a screen-shot of the user interface used for evaluation.

It should be noted that for the manual work described above some kind of structure in the environment is required. The manual labor might be very hard in highly unstructured scenes.

4.3 Other relations

In addition to the relative transformations added upon visibility and matching of observations, one can directly incorporate additional relations resulting from other sources of information, for example, given the knowledge about the length of a corridor in an environment. By adding a relation between two poses—each at one side of the corridor—one can incorporate knowledge about the global geometry of an environment if this is available. This fact is, for example, illustrated by the black dashed line in Fig. 3 that implies a known distance between two poses in a corridor that are not adjacent. Figure 4 plots the corresponding error identified by the relation.

In the experimental evaluation, we will show one example for such additional relations used in real world datasets.

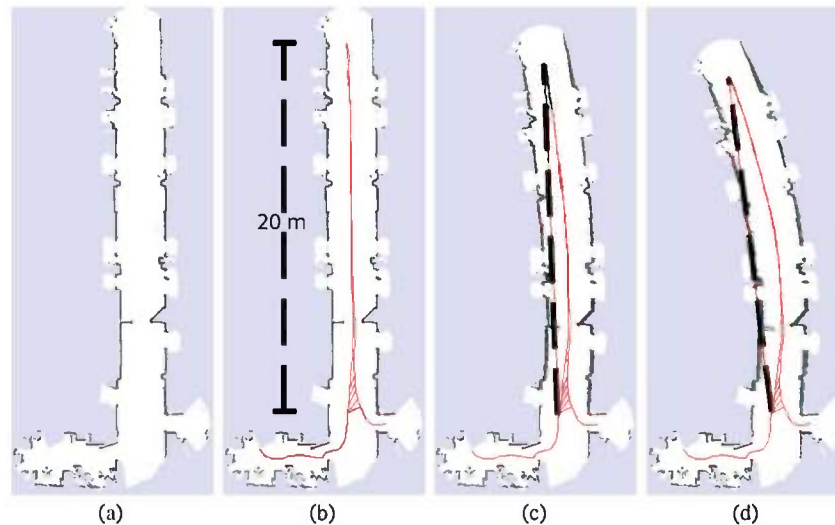


Fig. 3 Example of the performance measure on maps generated using different sensor setups. The relations between close-by positions are determined by a human assisted scan-alignment procedure performed on scans acquired at close-by locations. The *long dashed line* represents a relation added by manually measuring the relative distance at two locations of the robot: (a) the reference map obtained from the relative measurements, (b) the reference map superimposed with the network of relative measurements, (c) a map obtained by scan matching

using a 4 meters range sensor, with the superimposed relation (this map is still usable for navigating a robot), (d) a map obtained by cropping the range of the sensor to 3 meters. Whereas the quality of the right-most map is visibly decreased, it is also adequate for robot navigation since it preserves a correct topology of the environment (all doorways are still visible) and it correctly reflects the local spatial structure of the corridor. Therefore, it is locally consistent, but not globally consistent as (a). See also Fig. 4 for corresponding error plots

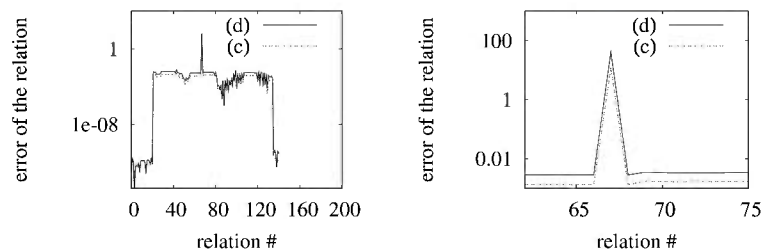


Fig. 4 This figure shows the behavior of the error metric for the maps (c) and (d) in Fig. 3. On the *left* we plot the error introduced by the individual relations. The *right plot* is a magnification of the *left one* in the

region corresponding to the manually introduced relations marked on the images with the *dashed line*. This results in a substantial increase of the global ε of SLAM results under comparison

In this example, we utilize relations derived from satellite image data.

5 Obtaining reference relations in outdoor environments

The techniques described in the previous section can be used to obtain a close-to-ground-truth for indoor environments. In outdoor scenarios however, the manual validation of the data is usually less practical due to the reduced structure and the large size. In wide open areas it may be difficult for a human operator to determine whether a potential alignment between laser scans is good or not due to the limited range of the scanner. Furthermore the poor structure of the environ-

ment makes this procedure hard even when the laser senses a high number of obstacles.

GPS is commonly used to bound the global uncertainty of a vehicle moving outdoors. Unfortunately, GPS suffers from outages or occlusions so that a robot relying on GPS might encounter substantial positioning errors. Especially in urban environments, GPS is known to be noisy. Even sophisticated SLAM algorithms cannot fully compensate for these errors as there still might be lacking relations between observations combined with large odometry errors that introduce a high uncertainty in the current position of the vehicle.

As an alternative to GPS, it is possible to use aerial images to determine relations close to the ground truth. We investigated this approach in our previous work (Kümmerle et al. 2009) and we show that this solution yields a better global consistency of the resulting map, if we consider the

prior information. Satellite images of locations are widely available on the web by popular tools like Google-Earth or Microsoft Live-Earth. This data can be used as prior information to localize a robot equipped with a 3D laser range finder.

The overall approach is based on the Monte-Carlo localization framework (Dellaert et al. 1998). The satellite images are captured from a viewpoint significantly different from the one of the robot. However, by using 3D scans we can extract 2D information which is more likely to be consistent with the one visible in the reference map. In this way, we can prevent the system from introducing inconsistent prior information.

In the following, we explain how we adapted Monte Carlo Localization (MCL) to operate on aerial images and how to select points from 3D scans to be considered in the observation model of MCL. This procedure returns a set of candidate robot locations $\hat{x}_t$. From those positions, we then select a subset of pairs of locations from which to compute the reference displacements $\hat{\delta}_{i,j}$ to be used in the metric.

5.1 Monte Carlo localization

To estimate the pose x of the robot in its environment, we consider probabilistic localization, which follows the recursive Bayesian filtering scheme. The key idea of this approach is to maintain a probability density $p(x_t | z_{1:t}, u_{0:t-1})$ of the location x_t of the robot at time t given all observations $z_{1:t}$ and all control inputs $u_{0:t-1}$. This posterior is updated as follows:

$$\begin{aligned} p(x_t | z_{1:t}, u_{0:t-1}) \\ = \alpha \cdot p(z_t | x_t) \cdot \int p(x_t | u_{t-1}, x_{t-1}) \cdot p(x_{t-1}) dx_{t-1}. \end{aligned} \quad (5)$$

Here, α is a normalization constant which ensures that $p(x_t | z_{1:t}, u_{0:t-1})$ sums up to one over all x_t . The terms to be described in Eq. 5 are the prediction model $p(x_t | u_{t-1}, x_{t-1})$ and the sensor model $p(z_t | x_t)$. One contribution of this work is an appropriate computation of the sensor model in the case that a robot equipped with a 3D range sensor operates in a given birds-eye map.

MCL is a variant of particle filtering (Doucet et al. 2001) where each particle corresponds to a possible robot pose and has an assigned weight $w^{[i]}$. The belief update from Eq. 5 is performed according to the following two alternating steps:

1. In the prediction step, we draw for each particle with weight $w^{[i]}$ a new particle according to $w^{[i]}$ and to the prediction model $p(x_t | u_{t-1}, x_{t-1})$.
2. In the correction step, a new observation z_t is integrated. This is done by assigning a new weight $w^{[i]}$ to each particle according to the sensor model $p(z_t | x_t)$.

Furthermore, the particle set needs to be re-sampled according to the assigned weights to obtain a good approximation of the pose distribution with a finite number of particles.

So far, we have described the general framework of MCL. In the next section, we will describe the sensor model for determining the likelihood $p(z_t | x_t)$ of perceiving the 3D scan z_t from a given robot position x_t within an aerial image. For convenience, we will drop the time index t in the remainder of this section.

5.2 Sensor model for 3D range scans in aerial images

The task of the sensor model is to determine the likelihood $p(z | x)$ of a reading z given the robot is at pose x . In our current system, we apply the so called endpoint model or likelihood fields (Thrun et al. 2005). Let z^k be the endpoints of a 3D scan z . The endpoint model computes the likelihood of a reading based only on the distances between a scan point $\hat{z}^k$ re-projected onto the map according to the pose x of the robot and the point in the map $\hat{d}^k$ which is closest to $\hat{z}^k$ as:

$$p(z | x) = f(\|\hat{z}^1 - \hat{d}^1\|, \dots, \|\hat{z}^k - \hat{d}^k\|). \quad (6)$$

If we assume that the beams are independent and the sensor noise is normally distributed we can rewrite Eq. 6 as

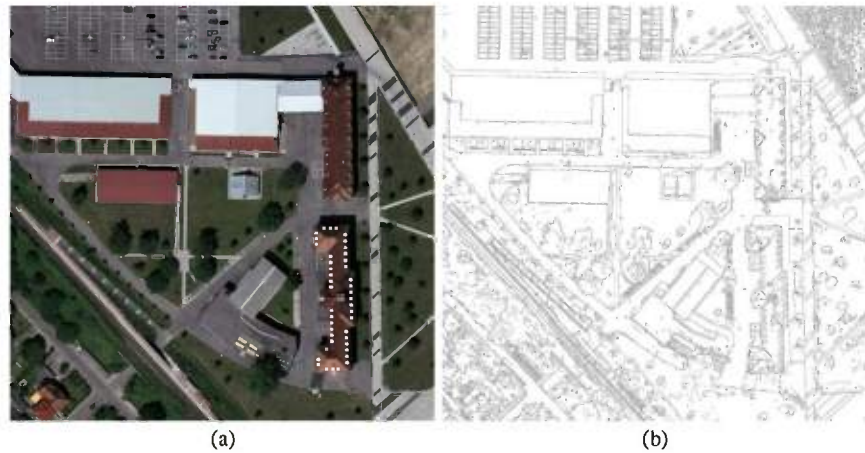
$$f(\|\hat{z}^1 - \hat{d}^1\|, \dots, \|\hat{z}^k - \hat{d}^k\|) \propto \prod_j e^{-\frac{(\hat{z}^j - \hat{d}^j)^2}{\sigma^2}}. \quad (7)$$

Since the aerial image only contains 2D information about the scene, we need to select a set of beams from the 3D scan, which are likely to result in structures that can be identified and matched in the image. In other words, we need to transform both the scan and the image into a set of 2D points which can be compared via the function $f(\cdot)$.

To extract these points from the image we employ the standard Canny edge extraction procedure (Canny 1986). The idea behind this is that if there is a height gap in the aerial image, there will often also be a visible change in intensity in the aerial image and we assume that this intensity change is detected by the edge extraction procedure. In an urban environment, such edges typically correspond to borders of roofs, trees, fences or other structures. Of course, the edge extraction procedure returns a lot of false positives that do not represent any actual 3D structure, like street markings, grass borders, shadows, and other flat markings. All these aspects have to be considered by the sensor model.

A straightforward way to address this problem is to select a subset of beams z^k from the 3D scan which will then be used to compute the likelihood. The beams which should be considered are the ones which correspond to significant variations along the z direction of the 3D scan. For vertical structures, a direct matching between the extracted edges and the measurements of a horizontal 2D laser range scanner can

Fig. 5 (a) A Google Earth image of the Freiburg campus. (b) The corresponding Canny image. Despite the considerable clutter, the structure of the buildings and the vertical elements are clearly visible



be performed, as discussed by Früh and Zakhori (2004). If a 3D laser range finder is available, we also attempt to match variations in height that are not purely vertical structures, like trees or overhanging roofs. This procedure is illustrated by the sequence of images in Fig. 6.

In the current implementation, we considered variations in height of 0.5 m and above as possible positions of edges that could also be visible in the aerial image. The positions of these variations relative to the robot can then be matched against the Canny edges of the aerial image in a point-by-point fashion, similar to the matching of 2D-laser scans against an occupancy grid map. Additionally, we employ a heuristic to detect when the prior is not available, i.e., when the robot is inside of a building or under overhanging structures. This is based on the 3D perception. If there is a ceiling which leads to range measurements above the robot no global relations from the localization are integrated, since we assume that the area the robot is sensing is not visible in the aerial image.

Figure 7 shows an example trajectory estimated with this technique (in red) and the GPS positions (in blue). As can be seen, the estimates are more accurate than the GPS data. Thus the improved guess facilitates the manual verification of the data. Note that the approach presented here is used to obtain the candidate relations for outdoor datasets. A human operator has to accept or decline all relations found by the approach.

6 Benchmarking for algorithms without trajectory estimates

A series of SLAM approaches estimate the trajectory of the robot as well as a map. However, in the context of the EKF, researchers often exclude an estimate of the full trajectory to lower the computational load. To facilitate evaluation one could store the current pose for each processing step and use it to build a trajectory. This would lead to good results

if only local accuracy is considered. However, global corrections appearing later in run-time are not represented correctly.

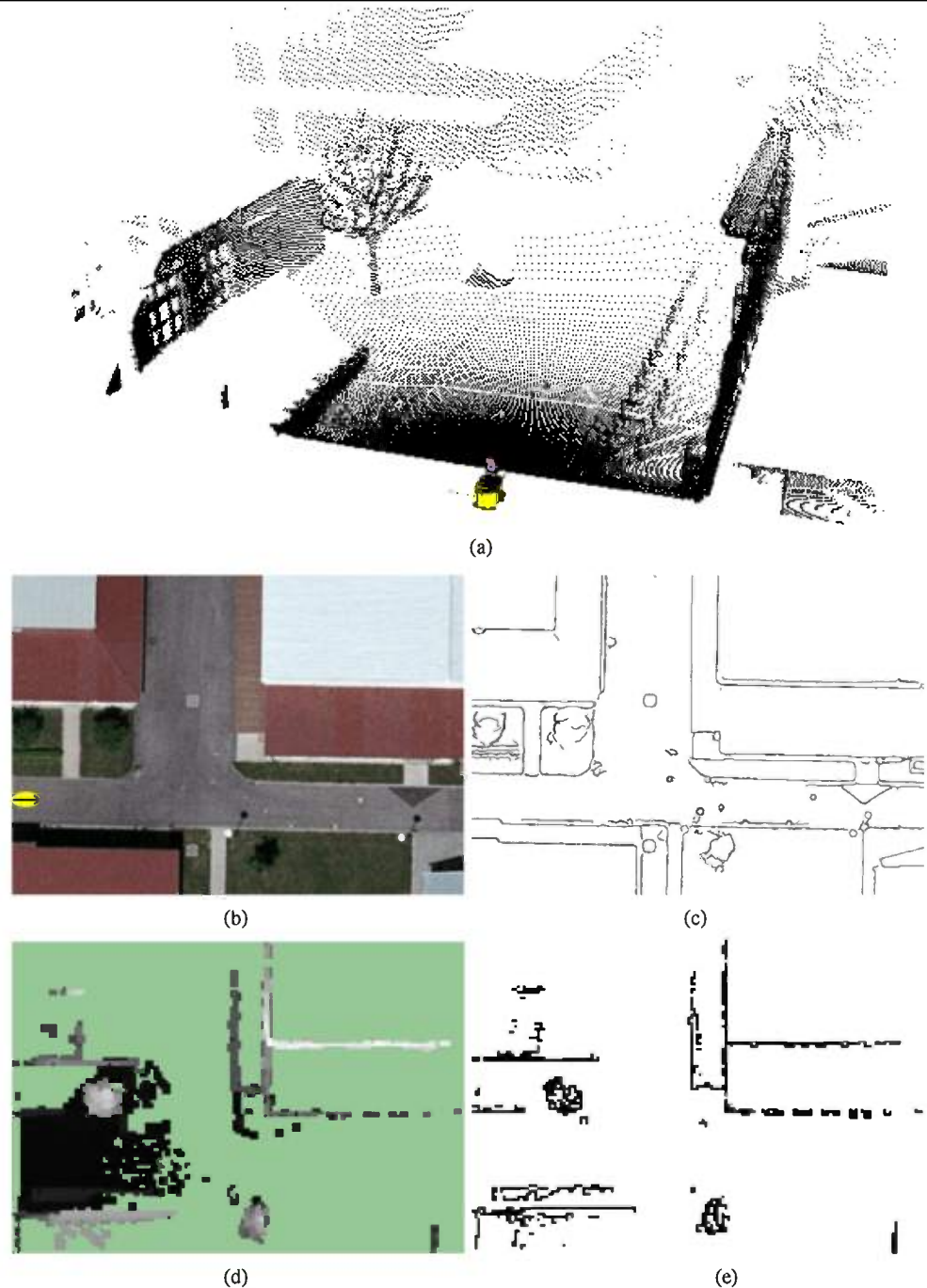
We see two solutions to overcome this problem: (a) depending on the capabilities of the sensor, one can recover the trajectory as a post processing step given the feature locations and the data association estimated by the approach. This procedure could be quite easily realized by a localization run in the built map with given data association (the data association of the SLAM algorithm). (b) In some settings this strategy can be difficult and one might argue that a comparison based on the landmark locations is more desirable. In this case, one can apply our metric operating on the landmark locations instead of based on the poses of the robot. In this case, the relations $\delta_{i,j}^*$ can be determined by measuring the relative distances between landmarks using, for example, a highly accurate measurement device.

The disadvantage of this approach is that the data association between estimated landmarks and ground truth landmarks is not given. Depending on the kind of observations, a human can manually determine the data association for each observation of an evaluation datasets as done by Frese (2008). This, however, might get intractable for SIFT-like features obtained with high frame rate cameras. Note that all metrics measuring an error based on landmark locations require such a data association as given. Furthermore, it becomes impossible to compare significantly different SLAM systems using different sensing modalities. Therefore, we would recommend the first option to evaluate techniques such as EKF.

7 Datasets for benchmarking

To validate the metric, we selected a set of datasets representing different kinds of environments from the publicly available datasets. We extracted relative relations between

Fig. 6 (a) A 3D scan represented as a point cloud. (b) The aerial image of the scene. (c) The Canny edges extracted from (b). (d) A view from the top, where the gray value represents the maximal height per cell. The darker the color the lower the height. (e) Extracted height variations from (d)



robot poses using the methods described in the previous sections by manually validating every single observation between pairs of poses.

As a challenging indoor corridor-environment with a non-trivial topology including nested loops, we selected the MIT Killian Court dataset.¹ (Infinite Corridor) and the dataset of the ACES building at the University of Texas,

Austin.² As a typical office environment with a significant level of clutter, we selected the dataset of building 079 at the University of Freiburg, the Intel Research Lab dataset,³ and a dataset acquired at the CSAIL at MIT. For addressing outdoor environments, we recorded a new dataset at the park area of the University Hospital, Freiburg. To give a visual impression of the scanned

¹Courtesy of Mike Bosse.

²Courtesy of Patrick Beeson.

³Courtesy of Dirk Haehnel.

Fig. 7 Trajectory estimated using satellite images versus GPS data overlaid on the image of the ALU-FR campus



environments, Fig. 8 illustrates maps obtained by executing state-of-the-art SLAM algorithms (Grisetti et al. 2007b, 2007c; Olson 2008). All datasets, the manually verified relations, and map images are available online at: <http://ais.informatik.uni-freiburg.de/slamevaluation/>.

8 Experimental evaluation

This evaluation is designed to illustrate the properties of our method. We selected three popular mapping techniques, namely scan matching, a Rao-Blackwellized particle filter-based approach, and a graph-based solution to the SLAM problem and processed the datasets discussed in the previous section.

We provide the scores obtained from the metric for all combinations of SLAM approach and dataset. This will allow other researchers to compare their own SLAM approaches against our methods using the provided benchmark datasets. In addition, we also present sub-optimally corrected trajectories in this section to illustrate how inconsistencies affect the score of the metric. We will show that our error metric is well-suited for benchmarking and this kind of evaluation.

8.1 Evaluation of existing approaches using the proposed metric

In this evaluation, we considered the following mapping approaches:

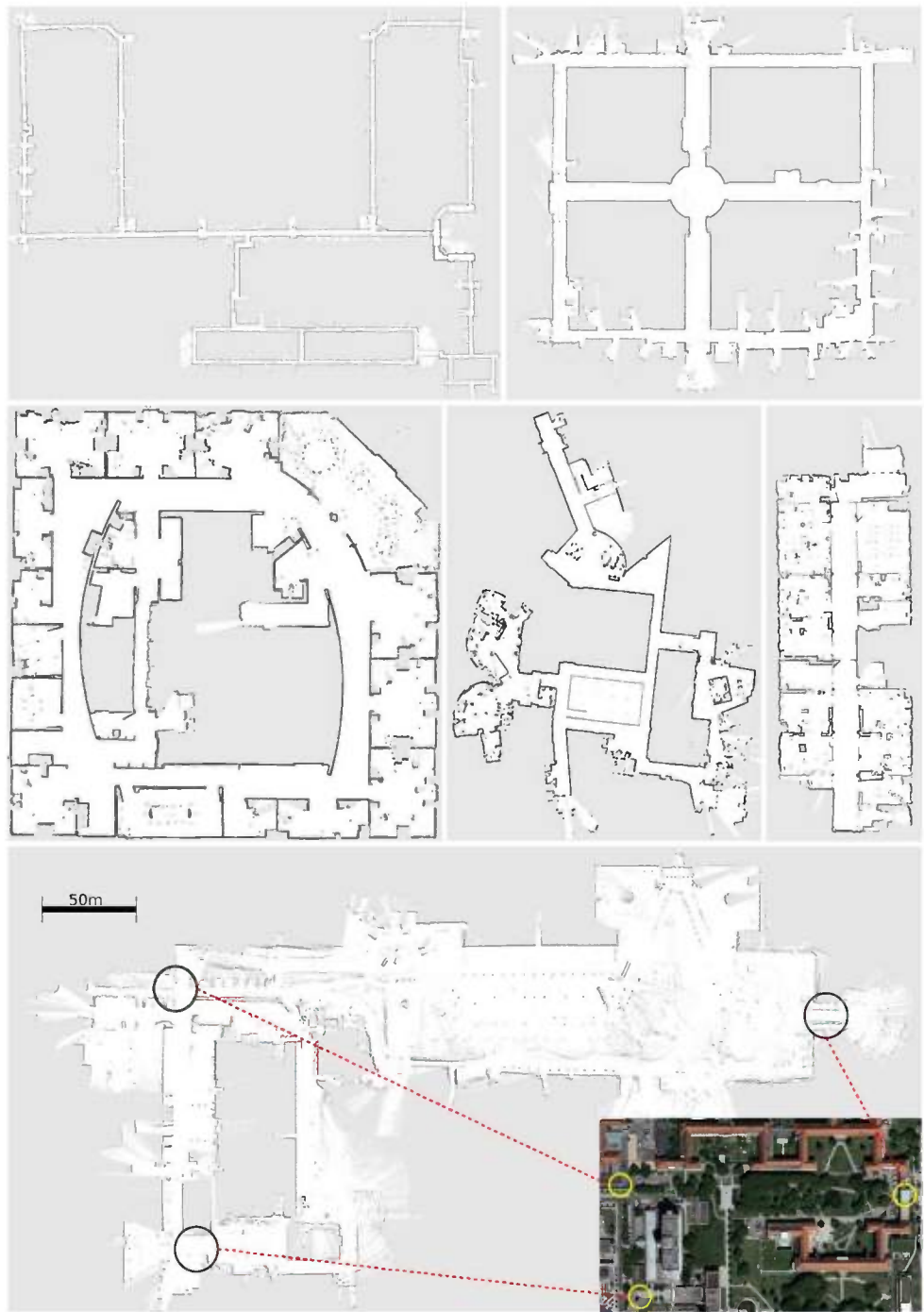
Scan Matching: Scan matching is the computation of the incremental, open loop maximum likelihood trajectory of

the robot by matching consecutive scans (Lu and Milios 1994; Censi 2006). In small environments, a scan matching algorithm is generally sufficient to obtain accurate maps with a comparably small computational effort. However, the estimate of the robot trajectory computed by scan matching is affected by an increasing error which becomes visible whenever the robot reenters in known regions after visiting large unknown areas (loop closing or place revisiting).

Grid-based Rao-Blackwellized Particle Filter (RBPF) for SLAM: We use the RBPF implementation described in Grisetti et al. (2007b), Stachniss et al. (2007b) which is available online (Stachniss et al. 2007a). It estimates the posterior over maps and trajectories by means of a particle filter. Each particle carries its own map and a hypothesis of the robot pose within that map. The approach uses an informed proposal distribution for particle generation that is optimized to laser range data. In the evaluation presented here, we used 50 particles. Note that a higher number of samples may improve the performance of the algorithm.

Graph Mapping: This approach computes a map by means of graph optimization (Grisetti et al. 2007c). The idea is to construct a graph out of the sequence of measurements. Every node in the graph represents a pose along the trajectory taken by the robot and the corresponding measurement obtained at that pose. Then, a least square error minimization approach is applied to obtain the most-likely configuration of the graph. In general, it is non-trivial to find the constraints, often referred to as the data association problem. Especially in symmetric environments or in situations with large noise, the

Fig. 8 Maps obtained by the reference datasets used to validate our metric. From top to bottom and left to right: MIT Killian Court (Boston), ACES Building (Austin), Intel Research Lab (Seattle), MIT CS Building (Boston), building 079 University of Freiburg, and the University Hospital in Freiburg. The depicted map of the University Hospital was obtained by using the background information extracted from the satellite images



edges in the graph may be wrong or imprecise and thus the resulting map may yield inconsistencies. In our current implementation of the graph mapping system, we followed the approach of Olson (2008) to compute constraints.

For our evaluation, we manually extracted the relations for all datasets mentioned in the previous section. The manually extracted relations are available online, see Sect. 7. We then carried out the mapping approaches and used the corrected trajectory for computing the error according to our

metric. Please note, that the error computed according to our metric (as well as for most other metrics too) can be separated into two components: a translational error and a rotational error. Often, a “weighting-factor” is used to combine both error terms into a single number, see, for example, Pfaff et al. (2006). In our evaluation, however, we provide both terms separately for a better transparency of the results.

We processed all benchmark datasets from Sect. 7 using the algorithms listed above. A condensed view of each algorithm’s performance is given by the averaged error over

Table 1 Quantitative results of different approaches/datasets on the translation error as well as the corresponding standard deviation and the maximum error

Translational error m (abs)/ m^2 (sqr)	Scan Matching	RBPF (50 part.)	Graph Mapping
Aces			
Equation 4 using absolute errors	0.173 ± 0.614	0.060 ± 0.049	0.044 ± 0.044
Equation 4 using squared errors	0.407 ± 2.726	0.006 ± 0.011	0.004 ± 0.009
Maximum absolute error of a relation	4.869	0.433	0.347
Intel			
Equation 4 using absolute errors	0.220 ± 0.296	0.070 ± 0.083	0.031 ± 0.026
Equation 4 using squared errors	0.136 ± 0.277	0.011 ± 0.034	0.002 ± 0.004
Maximum absolute error of a relation	1.168	0.698	0.229
MIT Killian Court			
Equation 4 using absolute errors	1.651 ± 4.138	0.122 ± 0.386^a	0.050 ± 0.056
Equation 4 using squared errors	19.85 ± 59.84	0.164 ± 0.814^a	0.006 ± 0.029
Maximum absolute error of a relation	19.467	2.513^a	0.765
MIT CSAIL			
Equation 4 using absolute errors	0.106 ± 0.325	0.049 ± 0.049^a	0.004 ± 0.009
Equation 4 using squared errors	0.117 ± 0.728	0.005 ± 0.013^a	0.0001 ± 0.0005
Maximum absolute error of a relation	3.570	0.508^a	0.096
Freiburg bldg 79			
Equation 4 using absolute errors	0.258 ± 0.427	0.061 ± 0.044^a	0.056 ± 0.042
Equation 4 using squared errors	0.249 ± 0.687	0.006 ± 0.020^a	0.005 ± 0.011
Maximum absolute error of a relation	2.280	0.856^a	0.459
Freiburg Hospital			
Equation 4 using absolute errors	0.434 ± 1.615	0.637 ± 2.638	0.143 ± 0.180
Equation 4 using squared errors	2.79 ± 18.19	7.367 ± 38.496	0.053 ± 0.272
Maximum absolute error of a relation	15.584	15.343	2.385
Freiburg Hospital, only global relations (see text)			
Equation 4 using absolute errors	13.0 ± 11.6	12.3 ± 11.7	11.6 ± 11.9
Equation 4 using squared errors	305.4 ± 518.9	288.8 ± 626.3	276.1 ± 516.5
Maximum absolute error of a relation	70.9	65.1	66.1

^aScan matching has been applied as a preprocessing step to improve the odometry

all relations. In Table 1, we give an overview on the translational error of the various algorithms, while Table 2 shows the rotational error. Comparing two algorithms can be done by comparing the values given in the tables, namely the maximum error as well as the average error. It can be seen that the more advanced algorithms (Rao-Blackwellized particle filter and graph mapping) usually outperform scan matching. This is mainly caused by the fact that scan matching only locally optimizes the result and will introduce topological errors in the maps, especially when large loops have to be closed. A distinction between RBPF and graph mapping seems difficult as both algorithms perform well in general. On average, graph mapping seems to be slightly better

than a RBPF for mapping. It should also be noted that for the outdoor dataset (Freiburg hospital), the RBPF mapper was not able to close the large loop and therefore was substantially worse than the graph mapper.

To visualize the results and to provide more insights about the metric, we do not provide the scores only but also plots showing the error of each relation. In case of high errors in a block of relations, we label the relations in the maps. This enables us to see not only where an algorithm fails, but can also provide insights as to why it fails. Inspecting those situations in correlation with the map helps to understand the properties of algorithms and gives valuable insights on its capabilities. For three datasets, a detailed

Table 2 Quantitative results of different approaches/datasets on the rotational error as well as the corresponding standard deviation and the maximum error

Rotational error deg (abs)/deg ² (sqr)	Scan Matching	RBPF (50 part.)	Graph Mapping
Aces			
Equation 4 using absolute errors	1.2 ± 1.5	1.2 ± 1.3	0.4 ± 0.4
Equation 4 using squared errors	3.7 ± 10.7	3.1 ± 7.0	0.3 ± 0.8
Maximum absolute error of a relation	12.1	7.9	3.5
Intel			
Equation 4 using absolute errors	1.7 ± 4.8	3.0 ± 5.3	1.3 ± 4.7
Equation 4 using squared errors	25.8 ± 170.9	36.7 ± 187.7	24.0 ± 166.1
Maximum absolute error of a relation	4.5	34.7	6.4
MIT Killian Court			
Equation 4 using absolute errors	2.3 ± 4.5	0.8 ± 0.8^a	0.5 ± 0.5
Equation 4 using squared errors	25.4 ± 65.0	0.9 ± 1.7^a	0.9 ± 0.9
Maximum absolute error of a relation	21.6	7.4 ^a	5.4
MIT CSAIL			
Equation 4 using absolute errors	1.4 ± 4.5	0.6 ± 1.2^a	0.05 ± 0.08
Equation 4 using squared errors	22.3 ± 111.3	1.9 ± 17.3^a	0.01 ± 0.04
Maximum absolute error of a relation	26.3	18.2 ^a	0.8
Freiburg bldg 79			
Equation 4 using absolute errors	1.7 ± 2.1	0.6 ± 0.6^a	0.6 ± 0.6
Equation 4 using squared errors	7.3 ± 14.5	0.7 ± 2.0^a	0.7 ± 1.7
Maximum absolute error of a relation	9.9	6.4 ^a	5.4
Freiburg Hospital			
Equation 4 using absolute errors	1.3 ± 3.0	1.3 ± 2.3	0.9 ± 2.2
Equation 4 using squared errors	10.9 ± 50.4	7.1 ± 42.2	5.5 ± 46.2
Maximum absolute error of a relation	27.4	28.0	29.6
Freiburg Hospital, only global relations (see text)			
Equation 4 using absolute errors	6.3 ± 5.2	5.5 ± 5.9	6.3 ± 6.2
Equation 4 using squared errors	66.1 ± 101.4	64.6 ± 144.2	77.2 ± 154.8
Maximum absolute error of a relation	27.3	35.1	38.6

^aScan matching has been applied as a preprocessing step to improve the odometry

analysis using these plots is presented in Sects. 8.2 to 8.4. The overall analysis provides the intuition that our metric is well-suited for evaluating SLAM approaches.

8.2 MIT Killian Court

The MIT Killian Court dataset has been acquired in a large indoor environment, where the robot mainly observed corridors lacking structures that support accurate pose correction. The robot traverses multiple nested loops—a challenge especially for the RBPF-based technique. We extracted close to 5000 relations between nearby poses that are used for

evaluation. Figure 9 shows three different results and the corresponding error distributions to illustrate the capabilities of our method. Regions in the map with high inconsistencies correspond to relations having a high error. The absence of significant structure along the corridors results in a small or medium re-localization error of the robot in all compared approaches. In sum, we can say the graph-based approach outperforms the other methods and that the score of our metric reflects the impression of a human about map quality obtained by visually inspecting the mapping results (the vertical corridors in the upper part are supposed to be parallel).

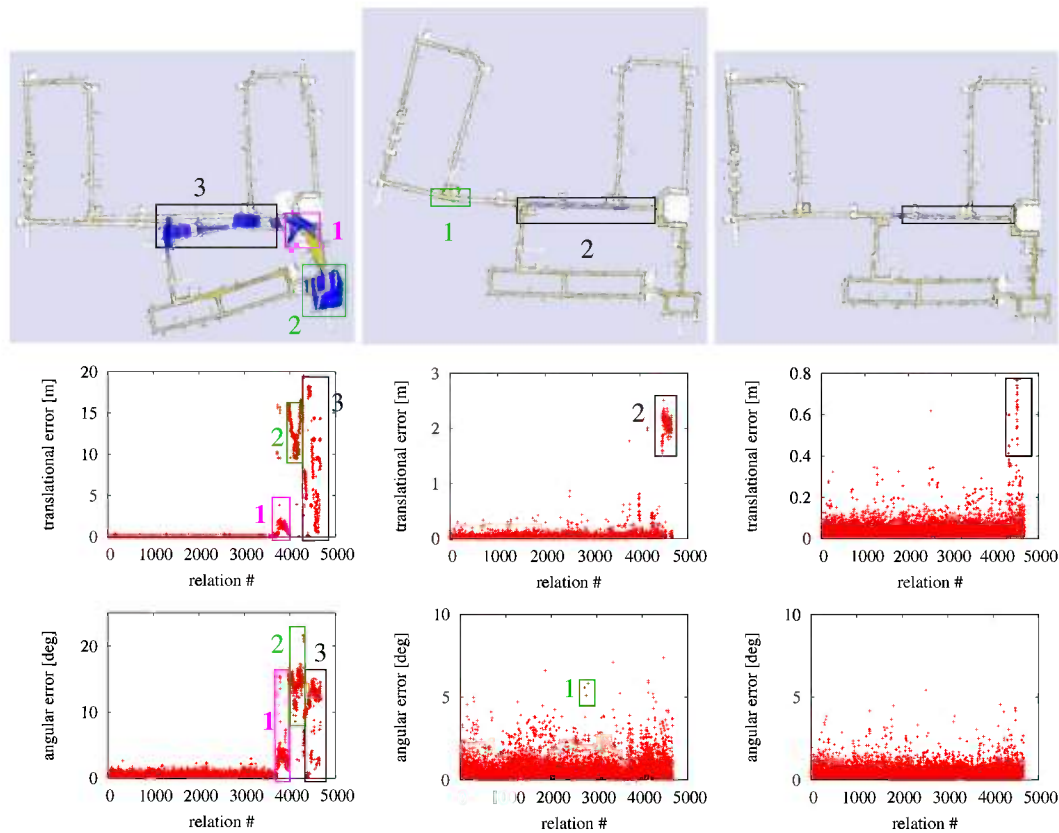


Fig. 9 This figure illustrates our metric applied to the MIT Killian Court dataset. The reference relations are depicted in *yellow* (light gray), while the relations marked in the plots are shown in *blue* (dark gray). The *left column* shows the results of pure scan-matching, the *middle column* the result of a RBPF-based technique with 50 samples, and the *right column* shows the result of a graph-based approach. The

regions marked in the map correspond to regions in the error plots having high error. Due to its inability of dealing with loop closures scan matching has a high error when revisiting known regions. However, the absence of significant structure along the corridors for scan registration is an issue for both the graph-based and the RBPF approach. All in all, the graph-based approach outperforms the other methods

8.3 Freiburg Indoor Building 079

The building 079 of the University of Freiburg is an example for an indoor office environment. The building consists of one corridor which connects the individual rooms. Figure 10 depicts the results of the individual algorithms (scan matching, RBPF, graph-based). In the first row of Fig. 10, the relations having a translational error greater than 0.15 m are highlighted in blue.

In the left plot showing the scan matching result, the relations plotted in blue are generated when the robot revisits an already known region. These relations are visible in the corresponding error plots (Fig. 10 first column, second and third row). As can be seen from the error plots, the relations with a number greater than 1000 have a larger error than the rest of the dataset. The fact that the pose estimate of the robot is sub-optimal and that the error accumulates can also be seen by the rather blurry map and that some walls occur twice. In contrast to that, the more sophisticated algorithms, namely RBPF and graph mapping, are able to produce consistent and accurate maps in this environment. Only very few

relations show an increased error (illustrated by dark blue relations).

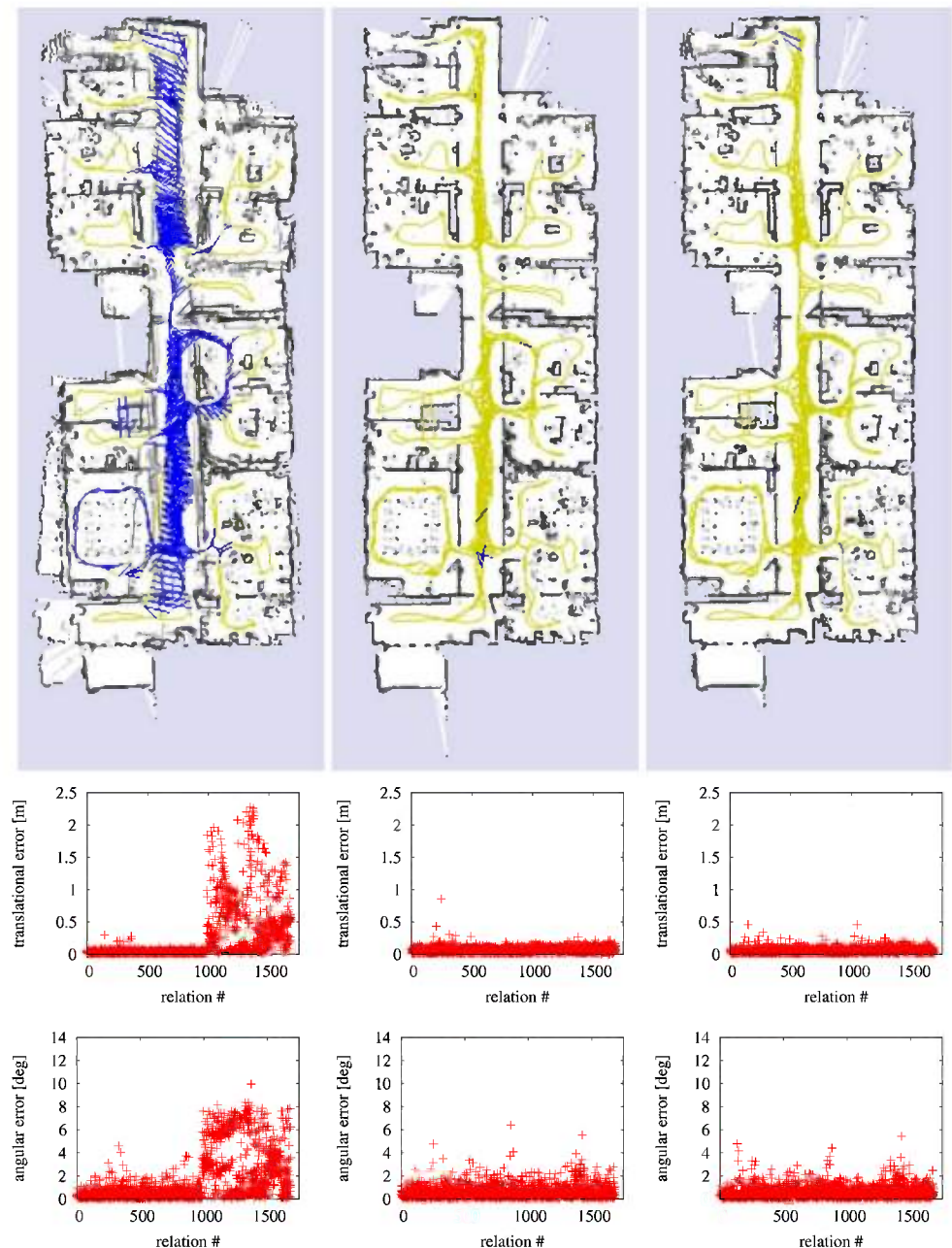
8.4 Freiburg University Hospital

This dataset consists of 2D and 3D laser range data obtained with one statically mounted SICK scanner and one mounted on a pan-tilt unit. The robot was steered through a park area that contains a lot of bushes and which is surrounded by buildings. Most of the time, the robot was steered along a bike lane with cobble stone pavement. The area is around 500 m by 250 m in size.

Figure 11 depicts the three mapping results, one obtained with scan matching (left), one with the RBPF (middle), and one with the graph mapper (right). The quality of all maps is lower than the quality of the map depicted in Fig. 8. The reason for that is that while building the map in Fig. 8, we also used the satellite image data which was not available for the algorithms under evaluation.

Based on the error plots in Fig. 11 as well as the overall score depicted in the tables, we can see that graph map-

Fig. 10 This figure shows the Freiburg Indoor Building 079 dataset. Each column reports the results of one approach. *Left*: scan-matching, *middle*: RBPF and *right* a graph based algorithm. Within each column, the *top image* shows the map, the *middle plot* is the translational error and the *bottom one* is the rotational error



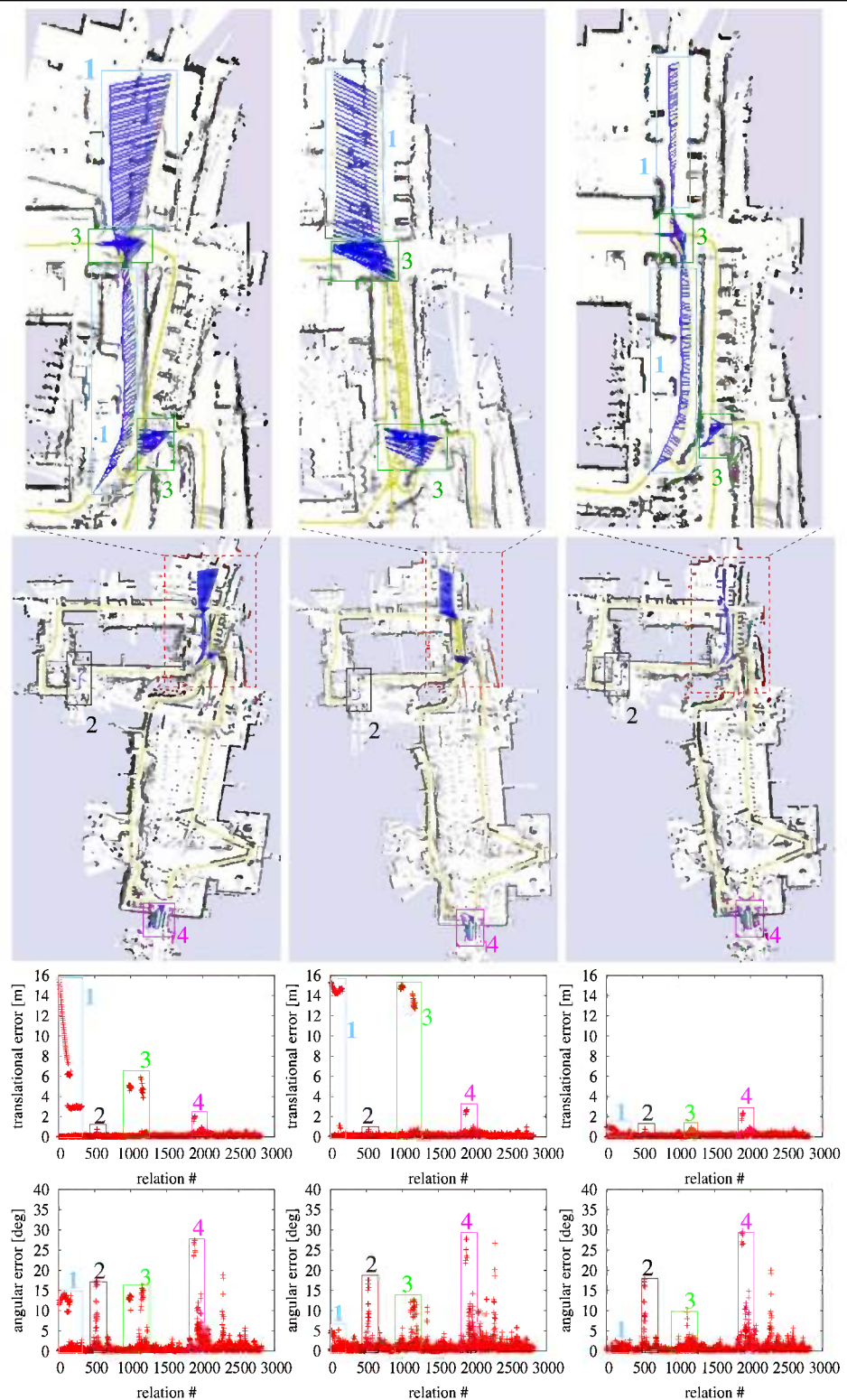
ping outperforms the RBPF and scan matching. The RBPF was not able to close the large loop and therefore performed similar to scan matching. However, note that in most parts of the map, the results of the scan matcher and RBPF are comparable to the one of graph mapping. Significant differences can be observed in the areas labeled as 1 and 3. Here, the two approaches fail to build a consistent map which is the reason for the significantly higher overall error.

In the area labeled as 4, the results of all algorithms yield matching errors. In that area, the robot makes a 180 degree turn and looks towards the open park area where almost no structure that would allow for pose correction is visible.

Therefore, none of the tested algorithms was able to build a perfect map here.

Note that for this dataset, we present two alternative sets of relations. One using only local relations based on the sensor range. In addition, we provide a set where the relations are generated for pairs of randomly sampled poses. This set should be used if global consistency is desired. A comparison between the two data sets can be seen in Fig. 12. The histograms count relations based by the difference in the time indices of the connected poses. As can be seen from the left image, using local relations based on the sensor range leads to a peaked histogram, since the relations only cover a small time frame. Additional minor peaks occur if the ro-

Fig. 11 Maps and error plots of the Freiburg University Hospital. Each column reports the results of one approach. *Left*: scan-matching, *middle*: RBPF and *right* a graph based algorithm. The *second row* depicts the created maps. The *first row* shows close-ups of areas in these maps. The error plots in the *middle* are regarding translation and on the *bottom* regarding rotation. There are corresponding areas marked in the plots and the maps



bot re-visits a region. In contrast, the set of relations used to evaluate the global consistency of the map is less peaked. Here, the relations uniformly sub-sample all available pairwise combinations of robot poses.

8.4.1 Utilizing additional relations

To illustrate that it is possible to incorporate additional relations as claimed in Sect. 4.3, we added in a further ex-

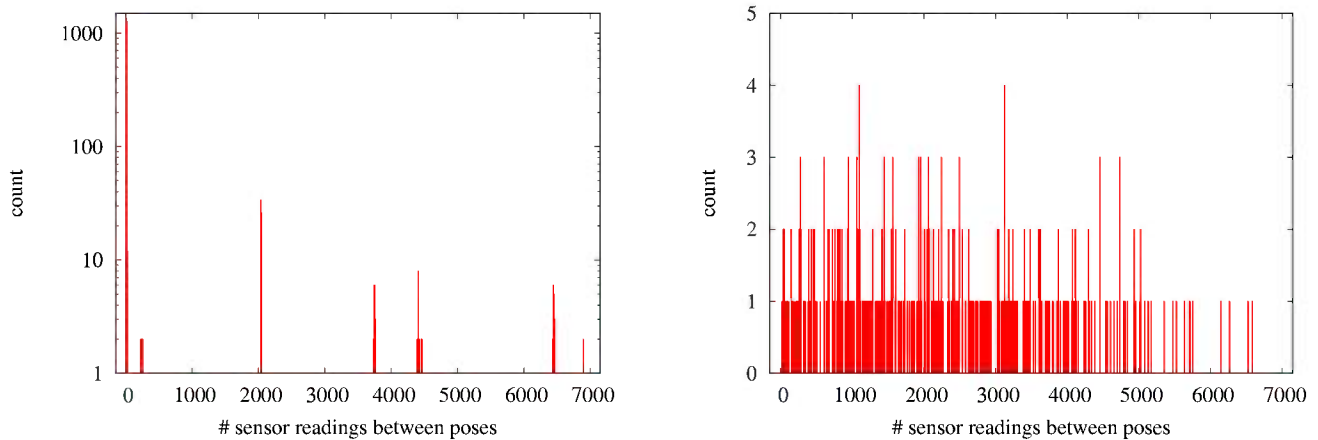


Fig. 12 Comparison of local relations with global relations based on their histograms. The *abscissa* shows the number of sensor readings between the two positions in a relation (a bin size of 10 was chosen).

On the *left side* the histogram for the local relation set is shown, while the *right side* displays the global relations

periment the satellite image data which was used to obtain the close-to-true pose information for the Freiburg hospital. These additional relations favor approaches that are able to generate global consistency as it is desired for robots that, for example, build blueprints.

The resulting scores for such a setting are given in the last rows of Tables 1 and 2, respectively. As expected, the error in case of the evaluation including global relations is higher.

8.5 Summary of the experiments

Our evaluation illustrates that the proposed metric provides a ranking of the results of mapping algorithms that is likely to be compatible with a ranking obtained from visual inspection by humans. Inconsistencies yield increased error scores since in the wrongly mapped areas the relations obtained from manual matching are not met. By visualizing the error of each relation as done in the plots in this section, one can identify regions in which algorithms fail and we believe that this helps to understand where and why different approaches have problems to build accurate maps.

We furthermore encourage authors to evaluate their algorithms based on multiple datasets and not just using a single in order to illustrate the generality of the method and not being optimized for a single dataset.

9 Conclusion

In this paper, we have presented a framework for analyzing the results of SLAM approaches that allows for creating objective benchmarks. We proposed a metric for measuring the error of a SLAM system based on the corrected trajectory. Our metric uses only relative relations between poses and

does not rely on a global reference frame. This overcomes serious shortcomings of approaches using a global reference frame to compute the error. The metric even allows for comparing SLAM approaches that use different estimation techniques or different sensor modalities.

In addition to the proposed metric, we provide robotic datasets together with relative relations between poses for benchmarking. These relations have been obtained by manually matching observations and yield a high matching accuracy. We present relations for self-recorded datasets with laser range finder data as well as for a set of log-files that are frequently used in the SLAM community to evaluate approaches. In addition, we provide an error analysis for three mapping systems including two modern laser-based SLAM approaches, namely a graph-based approach as well as system based on a Rao-Blackwellized particle filter. We believe that our results are a valuable benchmark for SLAM researchers since we provide a framework that allows for objectively and comparably easy analyzing the results of SLAM systems.

Acknowledgements This work has partly been supported by the DFG under contract number SFB/TR-8 and the European Commission under contract numbers FP6-2005-IST-6-RAWSEEDS, FP7-231888-EUROPA, and FP6-IST-045388-INDIGO. The authors gratefully thank Mike Bosse, Patrick Beeson, and Dirk Haehnel for providing the MIT Killian Court, the ACES, and the Intel Research Lab datasets.

References

- Amigoni, F., Gasparini, S., & Gini, M. (2007). Good experimental methodologies for robotic mapping: A proposal. In *Proc. of the IEEE int. conf. on robotics & automation (ICRA)*.
- Balaguer, B., Carpin, S., & Balakirsky, S. (2007). Towards quantitative comparisons of robot algorithms: Experiences with SLAM in simulation and real world systems. In *IROS 2007 workshop*.

- Bar-Shalom, Y., Li, X. R., & Kirubarajan, T. (2001). *Estimation with application to tracking and navigation*. New York: Wiley.
- Bonarini, A., Burgard, W., Fontana, G., Matteucci, M., Sorrenti, D. G., & Tardos, J. D. (2006). Rawseeds a project on SLAM benchmarking. In *Proceedings of the IROS'06 workshop on benchmarks in robotics research*. Available online at <http://www.robot.uji.es/EURON/pdfs/LectureNotesIROS06.pdf>.
- Bosse, M., Newman, P. M., Leonard, J. J., & Teller, S. (2003). An ALTAS framework for scalable mapping. In *Proc. of the IEEE int. conf. on robotics & automation (ICRA)* (pp. 1899–1906). Taipei, Taiwan.
- Burgard, W., Stachniss, C., Grisetti, G., Steder, B., Kümmerle, R., Dornhege, C., Ruhnke, M., Kleiner, A., & Tardós, J. D. (2009, to appear). A comparison of slam algorithms based on a graph of relations. In *Proc. of the int. conf. on intelligent robots and systems (IROS)*.
- Canny, J. (1986). A computational approach to edge detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 8(6), 679–698.
- Censi, A. (2006). Scan matching in a probabilistic framework. In *Proc. of the IEEE int. conf. on robotics & automation (ICRA)* (pp. 2291–2296).
- Darpa (2007). Darpa Urban Challenge. <http://www.darpa.mil/grandchallenge/>.
- Dellaert, F. (2005). Square Root SAM. In *Proc. of robotics: science and systems (RSS)* (pp. 177–184). Cambridge, MA, USA.
- Dellaert, F., Fox, D., Burgard, W., & Thrun, S. (1998). Monte Carlo localization for mobile robots. In *Proc. of the IEEE int. conf. on robotics & automation (ICRA)*. Leuven, Belgium.
- Dissanayake, G., Durrant-Whyte, H., & Bailey, T. (2000). A computationally efficient solution to the simultaneous localisation and map building (SLAM) problem. In *Proc. of the IEEE int. conf. on robotics & automation (ICRA)* (pp. 1009–1014).
- Doucet, A., de Freitas, N., & Gordon, N. (Eds.) (2001). *Sequential Monte-Carlo methods in practice*. Berlin: Springer.
- Duckett, T., Marsland, S., & Shapiro, J. (2002). Fast, on-line learning of globally consistent maps. *Autonomous Robots*, 12(3), 287–300.
- EPFL and IROS (2002). Cleaning Robot Contest. <http://robotika.cz/competitions/cleaning2002/en>.
- ESA (2008). Lunar robotics challenge. http://www.esa.int/esaCP/SEM4GKRTKMF_index_0.html.
- Estrada, C., Neira, J., & Tardós, J. D. (2005). Hierarchical SLAM: Real-time accurate mapping of large environments. *IEEE Transactions on Robotics*, 21(4), 588–596.
- Eustice, R., Singh, H., & Leonard, J. J. (2005a). Exactly sparse delayed-state filters. In *Proc. of the IEEE int. conf. on robotics & automation (ICRA)* (pp. 2428–2435).
- Eustice, R., Walter, M., & Leonard, J. J. (2005b). Sparse extended information filters: Insights into sparsification. In *Proc. of the int. conf. on intelligent robots and systems (IROS)* (pp. 641–648). Edmonton, Canada.
- Frese, U. (2006). Treemap: An $o(\log n)$ algorithm for indoor simultaneous localization and mapping. *Autonomous Robots*, 21(2), 103–122.
- Frese, U. (2008). Dlr spatial cognition data set. <http://www.informatik.uni-bremen.de/agebv/en/DlrSpatialCognitionDataSet>.
- Frese, U., Larsson, P., & Duckett, T. (2005). A multilevel relaxation algorithm for simultaneous localisation and mapping. *IEEE Transactions on Robotics*, 21(2), 1–12.
- Früh, C., & Zakhor, A. (2004). An automated method for large-scale, ground-based city model acquisition. *International Journal of Computer Vision*, 60, 5–24.
- Grisetti, G., Grzonka, S., Stachniss, C., Pfaff, P., & Burgard, W. (2007a). Efficient estimation of accurate maximum likelihood maps in 3D. In *Proc. of the int. conf. on intelligent robots and systems (IROS)*. San Diego, CA, USA.
- Grisetti, G., Stachniss, C., & Burgard, W. (2007b). Improved techniques for grid mapping with Rao-Blackwellized particle filters. *IEEE Transactions on Robotics*, 23, 34–46.
- Grisetti, G., Stachniss, C., Grzonka, S., & Burgard, W. (2007c). A tree parameterization for efficiently computing maximum likelihood maps using gradient descent. In *Proc. of robotics: science and systems (RSS)*.
- Gutmann, J.-S., & Konolige, K. (1999). Incremental mapping of large cyclic environments. In *Proc. of the IEEE int. symposium on computational intelligence in robotics and automation (CIRA)*.
- Hähnel, D., Burgard, W., Fox, D., & Thrun, S. (2003). An efficient FastSLAM algorithm for generating maps of large-scale cyclic environments from raw laser range measurements. In *Proc. of the int. conf. on intelligent robots and systems (IROS)* (pp. 206–211).
- Hermosillo, J., Pradalier, C., Sekhavat, S., Laugier, C., & Baille, G. (2003). Towards motion autonomy of a bi-steerable car: Experimental issues from map-building to trajectory execution. In *Proc. of the IEEE int. conf. on robotics & automation (ICRA)*.
- Hoover, A., Jean-Baptiste, G., Jiang, X., Flynn, P. J., Bunke, H., Goldgof, D. B., Bowyer, K. K., Eggert, D. W., Fitzgibbon, A. W., & Fisher, R. B. (1996). An experimental comparison of range image segmentation algorithms. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 18(7), 673–689.
- Howard, A., & Roy, N. (2003). Radish: The robotics data set repository, standard data sets for the robotics community. <http://radish.sourceforge.net/>.
- Julier, S., Uhlmann, J., & Durrant-Whyte, H. (1995). A new approach for filtering nonlinear systems. In *Proc. of the American control conference* (pp. 1628–1632).
- Kaess, M., Ranganathan, A., & Dellaert, F. (2007). iSAM: Fast incremental smoothing and mapping with efficient data association. In *Proc. of the IEEE int. conf. on robotics & automation (ICRA)*.
- Kümmerle, R., Steder, B., Dornhege, C., Kleiner, A., Grisetti, G., & Burgard, W. (2009). Large scale graph-based SLAM using aerial images as prior information. In *Proc. of robotics: science and systems (RSS)*.
- Leonard, J. J., & Durrant-Whyte, H. F. (1991). Mobile robot localization by tracking geometric beacons. *IEEE Transactions on Robotics and Automation*, 7(4), 376–382.
- Lu, F., & Milios, E. (1994). Robot pose estimation in unknown environments by matching 2d range scans. In *IEEE computer vision and pattern recognition conference (CVPR)* (pp. 935–938).
- Lu, F., & Milios, E. (1997). Globally consistent range scan alignment for environment mapping. *Autonomous Robots*, 4, 333–349.
- Montemerlo, M., Thrun, S., Koller, D., & Wegbreit, B. (2003). FastSLAM 2.0: An improved particle filtering algorithm for simultaneous localization and mapping that provably converges. In *Proc. of the int. conf. on artificial intelligence (IJCAI)* (pp. 1151–1156).
- Nüchter, A., Lingemann, K., Hertzberg, J., & Surmann, H. (2005). 6d SLAM with approximate data association. In *Proc. of the 12th int. conference on advanced robotics (ICAR)* (pp. 242–249).
- Olson, E. (2008). *Robust and efficient robotic mapping*. PhD thesis, Massachusetts Institute of Technology, Cambridge, MA, USA.
- Olson, E., Leonard, J., & Teller, S. (2006). Fast iterative optimization of pose graphs with poor initial estimates. In *Proc. of the IEEE int. conf. on robotics & automation (ICRA)* (pp. 2262–2269).
- Pfaff, P., Burgard, W., & Fox, D. (2006). Robust Monte-Carlo localization using adaptive likelihood models. In H.I. Christensen (Ed.), *STAR Springer tracts in advanced robotics: Vol. 22. European robotics symposium 2006* (pp. 181–194). Berlin: Springer.
- Ranganathan, A., Kaess, M., & Dellaert, F. (2007). Loopy sam. In *Proc. of the int. conf. on artificial intelligence (IJCAI)*.
- RoboCup Federation (2009). RoboCup Competitions. <http://www.robotcup.org>.
- Scharstein, D., & Szeliski, R. (2002). Middlebury stereo vision page. <http://www.middlebury.edu/stereo>.

- Smith, R. C., & Cheeseman, P. (1986). On the representation and estimation of spatial uncertainty. *International Journal of Robotics Research*, 5(4), 56–68.
- Smith, R., Self, M., & Cheeseman, P. (1990). Estimating uncertain spatial relationships in robotics. In I. Cox & G. Wilfong (Eds.), *Autonomous robot vehicles* (pp. 167–193). Berlin: Springer.
- Stachniss, C., Frese, U., & Grisetti, G. (2007a). OpenSLAM. org—give your algorithm to the community. <http://www.openslam.org>.
- Stachniss, C., Grisetti, G., Roy, N., & Burgard, W. (2007b). Evaluation of Gaussian proposal distributions for mapping with Rao-Blackwellized particle filters. In *Proc. of the int. conf. on intelligent robots and systems (IROS)*.
- Symeo GmbH (2008). <http://www.symeo.de>.
- Thrun, S. (2001). An online mapping algorithm for teams of mobile robots. *International Journal of Robotics Research*, 20(5), 335–363.
- Thrun, S., Liu, Y., Koller, D., Ng, A. Y., Ghahramani, Z., & Durrant-Whyte, H. (2004). Simultaneous localization and mapping with sparse extended information filters. *International Journal of Robotics Research*, 23(7/8), 693–716.
- Thrun, S., Burgard, W., & Fox, D. (2005). *Probabilistic robotics*. Cambridge: MIT Press.
- Thrun, S. et al. (2006). Winning the Darpa grand challenge. *Journal on Field Robotics*.
- Torrallba, A., Murphy, K. P., & Freeman, W. T. (2007). Labelme: the open annotation tool. <http://labelme.csail.mit.edu/>.
- Uhlmann, J. (1995). *Dynamic map building and localization: New theoretical foundations*. PhD thesis, University of Oxford.
- Wulf, O., Nüchter, A., Hertzberg, J., & Wagner, B. (2008). Benchmarking urban six-degree-of-freedom simultaneous localization and mapping. *Journal of Field Robotics*, 25(3), 148–163.
- Yguel, M., Keat, C. T. M., Brailon, C., Laugier, C., & Aycard, O. (2007). Dense mapping for range sensors: Efficient algorithms and sparse representations. In *Proc. of robotics: science and systems (RSS)*.



Rainer Kümmerle studied computer science at the University of Freiburg and received his Master degree in April 2007. He is currently working as a PhD-student in the Autonomous Intelligent Systems Lab of the University of Freiburg headed by Wolfram Burgard. His research interests lie in the areas of outdoor navigation, 3D mapping, and localization.



Bastian Steder received his Master's degree in Computer Science from the University of Freiburg (Germany) in April 2007. He is currently working as a PhD-student at the Autonomous Intelligent Systems Lab (University of Freiburg) headed by Wolfram Burgard. His areas of interest are SLAM, Vision, and object recognition in 3D.



Christian Dornhege studied computer science at the University of Freiburg and received his Master degree in April 2007. Since June 2007 he has been working as a PhD-student in the Research Group on the Foundations of Artificial Intelligence headed by Prof. Dr. Bernhard Nebel. His research interests lie in the areas of autonomous robots, symbolic and geometric planning.



Michael Ruhnke studied computer science at the University of Freiburg and received his Master degree in November 2008. His research interests lie in the areas of 3D object modelling, 3D mapping, and localization.



Giorgio Grisetti is working as a Post-doc at the Autonomous Intelligent Systems Lab of Freiburg University headed by Wolfram Burgard. Before, he was a PhD student at University of Rome "La Sapienza" in the Intelligent Systems Lab. His advisor was Daniele Nardi and he received his PhD degree in April 2006. His research interests lie in the areas of mobile robotics. His previous and current works aims to provide effective solutions to mobile robot navigation in all its aspects: SLAM, localization and path planning.



Cyrill Stachniss studied computer science at the University of Freiburg and received his PhD degree in 2006. After his PhD, he was a senior researcher at ETH Zurich. Since 2007, he is an academic advisor at the University of Freiburg in the Laboratory for Autonomous Intelligent Systems. His research interests lie in the areas of robot navigation, exploration, SLAM, as well as learning approaches.



Alexander Kleiner is academic adviser (German: “Akademischer Rat”) in the Research Group on Foundations of Artificial Intelligence in Freiburg, headed by Prof. Dr. Bernhard Nebel. He received his PhD degree from the University of Freiburg in February 2008. Since 2006, he is member of the executive committee of RoboCup. His research areas are autonomous robot exploration, simultaneous localization and mapping (SLAM), and mixed-initiative teams of humans and multi-robot systems. The main

concern of his work is on field robotics, e.g. to develop robotic solutions for supporting humans in hostile situations, such as urban search and rescue (USAR).

Attachment H

Probabilistic Localization and Mapping in Appearance Space

Mark Cummins
Balliol College



Supervisor:
Dr Paul Newman

Robotics Research Group
Department of Engineering Science
University of Oxford

October 2009

“Begin at the beginning”, the King said gravely, “and go on till you come to the end: then stop.”

– *Lewis Carroll, Alice’s Adventures in Wonderland*

Abstract

This thesis is concerned with the problem of place recognition for mobile robots. How can a robot determine its location from an image or sequence of images, without any prior knowledge of its position, even in a world where many places look identical? We outline a new probabilistic approach to the problem, which we call Fast Appearance Based Mapping or FAB-MAP. Our map of the environment consists of a set of discrete locations, each with an associated appearance model. For every observation collected by the robot, we compute a probability distribution over the map, and either create a new location or update our belief about the appearance of an existing location. The technique can be seen as a new type of SLAM algorithm, where the appearance of locations (rather than their position) is subject to estimation. Unlike existing SLAM systems, our appearance based technique does not rely on keeping track of the robot in any metric coordinate system. Thus it is applicable even when informative observations are available only intermittently. Solutions to the loop closure detection problem, the kidnapped robot problem and the multi-session mapping problem arise as special cases of our general approach.

Our probabilistic model introduces several technical advances. The model incorporates correlations between visual features in a novel way, which is shown to improve system performance. Additionally, we explicitly compute an approximation to the partition function in our Bayesian formulation, which provides a natural probabilistic measure of when a new observation should be assigned to a location not already present in the map. The technique is applicable even in visually repetitive environments where many places look the same.

Finally, we define two distinct approximate inference procedures for the model. The first of these is based on concentration inequalities and has general applicability beyond the problem considered in this thesis. The second approach, built on inverted index techniques, is tailored to our specific problem of place recognition, but achieves extreme efficiency, allowing us to apply FAB-MAP to navigation problems on the largest scale. The thesis concludes with a visual SLAM experiment on a trajectory 1,000 km long. The system successfully detects loop closures with close to 100% precision and requires average inference time of only 25 ms by the end of the trajectory.

Statement of Authorship

This thesis is submitted to the Department of Engineering Science, University of Oxford, in fulfilment of the requirements for the degree of Doctor of Philosophy. This thesis is entirely my own work, and except where otherwise stated, describes my own research.

Mark Cummins, Balliol College

Funding

The work reported in this thesis was funded partly by the Systems Engineering for Autonomous Systems (SEAS) Defence Technology Centre established by the UK Ministry of Defence and partly by the Engineering and Physical Sciences Research Council.

We would also like to acknowledge Google Inc. for providing the City Block data set, which was helpful during the development of the work reported here.

Acknowledgements

First and foremost, thanks are due to my supervisor Paul Newman. Paul was everything a good supervisor should be. Some wise words at the right times, but more than anything else, the freedom to explore interesting things. There was also an unforgettable moment in an airport.

Thanks are also due to the many people who have given me a helping hand at one point or another. To lead Rezek for some early pointers that proved very fruitful. To Rohan Paul, without whom the 1,000 km data set would never have been possible. Thanks also for teaching me a lot of Indian history along the way – that was a good week, apart from the roundabouts. And of course, thanks to all the members of MRG and AVL; for discussions, for friendship and for good times.

Finally, but most importantly, to my family, who have been there through thick and thin, before and after.

Contents

1	Introduction	8
1.1	Motivation	8
1.2	Outline of the Problem	10
1.3	Contributions	10
1.4	Structure of the Thesis	11
1.5	Publications	12
2	Background	13
2.1	The Robot Navigation Problem	13
2.1.1	Overview	14
2.1.2	Extended Kalman Filter SLAM	14
2.1.3	Particle Filter SLAM	15
2.1.4	Other Estimation Techniques	16
2.1.5	Navigation and Mapping without a Global Coordinate Frame	17
2.1.6	Challenging Problems in SLAM	18
2.1.6.1	Local Data Association	18
2.1.6.2	The Loop Closure Detection Problem	20
2.1.6.3	The Kidnapped Robot Problem and Multi-Session Mapping	23
2.2	Content Based Image Retrieval Systems	23
2.2.1	The Bag of Words Representation	24

2.2.2	Interest Point Detectors	26
2.2.3	Interest Point Descriptors	27
2.2.4	Detector and Descriptor used in this Thesis	28
2.2.5	Partitioning of Descriptor Space	28
2.2.6	Notable Image Retrieval Systems	29
2.3	Appearance Based Techniques in Robotics	31
2.3.1	Localization using Appearance	32
2.3.2	Loop Closure Detection	33
2.3.3	Recent Developments	34
3	The FAB-MAP Model	36
3.1	Approximating High Dimensional Discrete Distributions	37
3.1.1	Chow Liu Trees	38
3.1.2	Sample Complexity of Learning	40
3.1.3	Smoothing Probability Estimates	40
3.1.4	What the Chow Liu Tree Learns	42
3.2	Probabilistic Navigation using Appearance	43
3.2.1	Representing Appearance	43
3.2.2	Representing Locations	44
3.2.3	Inference in the FAB-MAP Model	45
3.2.4	Evaluating The Model	46
3.2.5	Independence Assumptions, Approximations and Input Parameters	54
3.2.6	Discussion: Alternative Generative Models	56
3.3	Implementing the Model	60
3.3.1	Converting Images to Bag-of-Words Representation	60
3.3.2	Learning the Vocabulary Model	61
3.3.3	Algorithm Configuration	62
3.4	Performance Evaluation	63

3.4.1	Data Sets	65
3.4.2	Performance Metrics	66
3.4.3	Results	66
3.4.4	Generalization Performance	75
3.4.5	Use with Metric SLAM Systems	77
3.5	Chapter Summary	80
4	Approximate Inference using Concentration Inequalities	81
4.1	Introduction	81
4.2	Bail-Out Strategies	81
4.3	Probabilistic Bail-Out using Bennett's Inequality	82
4.4	Application to Appearance-Only SLAM	85
4.4.1	Applicability of Bennett's Inequality	86
4.4.2	Ranking Features	87
4.4.3	Calculating M and v	88
4.4.4	Calculating a pdf over Hypotheses	89
4.5	Results	89
4.6	Chapter Summary	95
5	Inverted Index Techniques for Large Scale Place Recognition	96
5.1	Related Work	97
5.2	System Description	98
5.2.1	Inverted Indices	98
5.2.2	FAB-MAP 2.0 - An Approximation to FAB-MAP for Large Scale Applications	98
5.2.3	Geometric Verification	103
5.2.4	Visual Vocabulary Learning At Large Scale	106
5.3	Data Set	108
5.4	Results	110
5.4.1	Test Conditions	110

5.4.2	Overall Performance	112
5.4.3	Comparison to tf-idf	114
5.4.4	Timing	117
5.4.5	Comparison to Original System	118
5.5	Chapter Summary	118
6	Discussion	127
6.1	Summary of Contribution	127
6.2	Future Work	128
6.3	Concluding Remark	131
A	Data Collection Platforms	132
B	Chow Liu Trees	135
C	Manipulations	138
C.1	FAB-MAP Model	138
C.2	Bennett Bound	139

Nomenclature

e_q	The event that an object (scene element) which generates observations of type z_q exists at a location.
L_i	The i^{th} location in the map.
$\mathcal{L}^k$	The set of all locations in the map at time k .
Z_k	The k^{th} observation collected by the robot.
z_q	The event that the q^{th} word of the visual vocabulary was observed.
z_{p_q}	An observation of word p_q , the parent of word q in the Chow Liu tree.
$\mathcal{Z}^k$	The set of all observations up to time k .

Chapter 1

Introduction

1.1 Motivation

The overall aim of this thesis is to contribute to the development of autonomous mobile robots. For several decades, the practical application of robotics has remained limited to carefully controlled environments such as assembly lines. Automation of the vast majority of real-world tasks remains out of reach, because our best systems cannot handle the complexity of unconstrained environments. The research community continues to make steady progress, but the critical level of robustness has yet to be attained. Crossing the threshold will open up a huge field of new possibilities – from personal service robots in the home to habitat monitoring in the oceans. We continue to pursue that goal.

In particular, this thesis will be concerned with autonomous navigation. To be useful, a robot must be able to reliably find its way between locations in its environment. For some applications, solutions based on human-constructed maps and external infrastructure such as the Global Positioning System (GPS) are sufficient, but in many cases the navigation problem must be solved on the basis of the robot's internal sensors alone. This might be dictated by considerations of flexibility and costs, or by necessity. For example, no infrastructure is available to planetary exploration robots, and GPS is unavailable in many terrestrial environments such as indoors, near tall buildings, under foliage, underground and underwater. In these situations the robotic system must solve the navigation problem without external



(a) Perceptual Variability



(b) Perceptual Aliasing



Figure 1.1: Appearance based navigation systems must assess the probability that two images come from the same place. This is difficult because place appearance can vary, (a), and more challengingly, because different places in the environment can appear identical, (b). The appearance based navigation problem is thus distinct from typical content based image retrieval tasks.

assistance. This problem of navigation without prior knowledge and without external infrastructure is known as Simultaneous Localization and Mapping (SLAM), and has been a core area of research in mobile robotics for several decades. While much progress has been made, a robust and general solution is still lacking. This thesis aims to increase the robustness of navigation systems by formulating a new approach to SLAM based on appearance rather than metric information. Our approach allows the robot to incrementally construct a model of the appearance of locations in the environment, and to globally re-localize by recognizing places it has previously visited. As a special case it solves a problem known as loop closure detection which has been a persistent difficulty for SLAM systems.

1.2 Outline of the Problem

Concretely, the problem that will be addressed in this thesis is the following: Given two observations or sequences of observations (typically images), what is the probability that these observations were collected at the same location?

This is a hard problem for a number of reasons. Firstly, the world is dynamic. Two images of the same location may look different due to changes in lighting or viewpoint, arrival or departure of vehicles, changes in weather, etc. Secondly, and more challengingly, the world is visually repetitive. Common textures such as brickwork and foliage are present everywhere. Mass produced objects and symbols such as road markings appear throughout the environment. These problems are illustrated in Figure 1.1. Under many reasonable metrics, the image pair shown in Figure 1.1b is more similar than the pair in 1.1a. In our bag-of-words representation (introduced later), 1.1b share 28% of their visual words, whereas 1.1a share only 15%. However, we would like to build a system that asserts with high confidence that 1.1a shows two images of the same location, whereas 1.1b probably does not. To do this, we must go beyond a simple computation of image similarity, and build a model that allows us to evaluate the *distinctiveness* of images. This will involve learning a probability distribution over the space of images, which captures the fact that the scene in Figure 1.1b is common in the environment, so despite the similarity the images are unlikely to come from the same location.

1.3 Contributions

This thesis makes the following contributions to the field:

- The first fully probabilistic framework for localization and mapping using appearance which is naturally robust to perceptual aliasing. Our technique can be used alone, or as a loop closure detection component of a larger robot navigation system.
- A direct approach to calculating the probability that a new observation comes from a location not already in the map, which proceeds by computing an approximation of the partition function in our probabilistic model. To our knowledge this is the first method that goes beyond simple

heuristics such as fixed similarity thresholds. In our experience the new approach is a dramatic improvement, particularly in that it naturally handles challenging cases such as perceptual aliasing.

- A novel technique for taking account of correlations between visual words when computing observation likelihoods. We learn an approximation to the joint distribution over visual words, which captures the fact that objects in the data generate correlated sets of features. We explore efficient learning methods which allow us to apply the technique to large visual vocabularies ($>100,000$ words), and demonstrate that it improves performance. The approach should have applicability beyond the problem at hand.
- An approximate inference technique based on the use of concentration inequalities. The technique is general in its applicability, and allows large speed increases to be realized in return for a small user-controlled probability of generating an incorrect result. For our problem, the technique yielded speed improvements in the range 25x-50x with minimal degradation in accuracy.
- Several public benchmark data sets for evaluating appearance-only SLAM systems, including a 1,000 km set which is, to our knowledge, the most extensive benchmark set available by a considerable margin. The set contains 20 Hz stereo imagery, omni-directional imagery every 5-10 m and GPS ground truth.

1.4 Structure of the Thesis

In Chapter 2 we present a review of relevant background material and related work in image retrieval and robot navigation. Chapter 3 describes a new probabilistic framework for appearance-based navigation, our “Fast Appearance Based Mapping” or “FAB-MAP” model. Chapter 4 describes a novel approach to approximate inference for more rapidly computing position under this model. Chapter 5 describes a second approximation technique suitable for very large scale applications of appearance-based navigation, and presents experimental results on a 1,000 km test trajectory. Conclusions and an outline of possible future research directions are presented in Chapter 6.

1.5 Publications

The system described in Chapter 3 was published in ICRA 2007 [30] with subsequent extension in IJRR [32]. The techniques in Chapter 4 appeared in ICRA 2008 [31], and were awarded the Best Vision Paper prize. Those of Chapter 5 appeared in RSS 2009 [33]. Various applications of the system and related work appeared in [110, 109, 91, 164, 163, 107, 125, 124].

Chapter 2

Background

This chapter introduces background material relevant to the thesis. We draw on two general strands of literature, namely the work in the robotics community on the Simultaneous Localization and Mapping (SLAM) problem, and work in the computer vision community on content-based image retrieval system. We begin by discussing the SLAM problem in robotics.

2.1 The Robot Navigation Problem

"The true logic of this world is in the calculus of probabilities."

– *James Clerk Maxwell.*

A mobile robot exists in a state of continual uncertainty. Commands sent to actuators may not be reliably carried out. Sensor readings are subject to noise and bias. Motion models are only approximations. Yet for a robot to be able to carry out useful tasks, it must be able to reliably navigate between locations in its environment. For reasons of flexibility and cost, we would like the robot to achieve this task entirely autonomously, without the need for manually constructed maps or robot-specific navigational aids such as beacons. How this may be achieved has been at the core of robotics research for several decades, where it is referred to as the Simultaneous Localization and Mapping (SLAM) problem. Early navigation algorithms that ignored uncertainty were brittle and subject to silent failure. Essentially all

successful approaches to SLAM (and indeed most problems in robotics) embrace the uncertainty of the robot’s perceptual world by framing the problem as a Bayesian inference task. We begin this section by introducing the concept of probabilistic navigation and describe some of the different algorithms used, along with a discussion of their failure modes.

2.1.1 Overview

A SLAM algorithm must maintain a probability distribution x that describes the robot’s belief about its position and the position of objects in the world, given a stream of observations Z (typically laser range measurements, camera images, odometry, etc). If the k^{th} observation is denoted Z_k and the set of all observations up to time k is denoted $\mathcal{Z}^k$, then SLAM can be formulated as a recursive Bayesian estimation problem:

$$p(x|\mathcal{Z}^k) = \frac{p(Z_k|x)p(x|\mathcal{Z}^{k-1})}{p(Z_k|\mathcal{Z}^{k-1})} \quad (2.1)$$

Depending on the choice of underlying world representation and the representation of the pdf x , this equation leads to the various SLAM algorithms. The most prominent approaches are based on the Extended Kalman Filter and the Rao-Blackwellized Particle Filter. The details of these algorithms will not concern us in this thesis; we limit ourselves to a brief discussion of their salient performance characteristics only. A more detailed introduction can be found in [41, 3].

2.1.2 Extended Kalman Filter SLAM

The seminal 1990 paper of Smith, Self and Cheeseman [146] introduced the widely used Extended Kalman Filter (EKF) formulation of SLAM, where the world is represented as a set of landmarks, and the vehicle and landmark covariances are approximated by Gaussian distributions. The EKF-based approach to SLAM was a revolutionary improvement and continues to be widely used. However it suffered from a number of performance-limiting issues, most prominently computational scaling, linearization and the assumption of Gaussianity.

Scaling issues (the fact that the computation needed to maintain a map increases quadratically in the number of landmarks) have largely been resolved by subsequent research – successful approaches

include submapping strategies [120, 13] and information form methods [159]. The fastest methods claim $O(1)$ complexity per update [78], at the expense of slower filter convergence.

However, the issue of linearization largely remains. “Linearization errors” arise from the fact that the Kalman filter is fundamentally applicable only to linear processes. The Extended Kalman Filter extends the technique to the non-linear case by linearizing all functions of interest about their current estimates. This approximation is a source of error, particularly if the function being estimated is highly non-linear, or if the state estimate is biased. The problem is partly addressed by techniques such as the Unscented Kalman Filter [68] and Iterated Extended Kalman Filter [4], and partly by submapping techniques with appropriate choice of coordinate frames [21]. However, linearization remains an essentially unavoidable problem within the EKF framework.

Finally, for many purposes the largest limitation of EKF-based methods is the assumption that the underlying vehicle and landmark covariances are well-approximated by Gaussians. There are many situations where this is not the case, for example in the presence of large angular errors, or when the pdf becomes multi-modal. In such cases a navigation system must use techniques beyond the EKF framework, or suffer an inconsistent map estimate.

2.1.3 Particle Filter SLAM

An alternative solution to SLAM based upon a Rao-Blackwellized particle filter was first proposed by Montemerlo, Thrun, Koller and Wegbreit in 2002 [100] and later improved in [101]. The solution (known as FastSLAM) exploits the fact that the landmark position estimates are conditionally independent given the trajectory of the vehicle. The FastSLAM algorithm maintains the distribution over vehicle position using a set of particles, where each particle possesses N low-dimensional EKFs to represent the N landmarks. In contrast to EKF SLAM, the particle filter based approach can accurately represent the arbitrary distributions over vehicle position that may arise from non-linear vehicle kinematics or situations where the position uncertainty is multi-modal. Additionally, the approach does not suffer from significant linearization issues and in combination with some efficient data structures allows updates to be computed in $O(M \log N)$ time for M particles (as opposed to $O(N^2)$ for the EKF).

The main difficulty with FastSLAM is maintaining sufficient *diversity* in the particle set over

long trajectories. FastSLAM applies a resampling step during filter updates, which causes a loss of information about trajectory history and is strictly only applicable for systems which “*exponentially forget*” their past (i.e. where the current state of the system tends towards independence from earlier states) [29]. The SLAM problem does not have this exponential forgetting property, and as a result FastSLAM and related techniques can have difficulties building consistent maps over longer trajectories containing multiple loops.

2.1.4 Other Estimation Techniques

The core estimation problem of SLAM has been approached from a number of directions in addition to the popular EKF and particle filter formulations. Within the Bayesian filtering framework, the Thin Junction Tree Filter proposed by Paskin [118, 119] approximates inter-landmark correlations by a thin junction tree. The algorithm achieves linear complexity in both time and space, and is related to submapping approximations to the EKF.

Square Root Information Smoothing [37, 70] proposed by Dellaert in 2005 moves away from the Bayesian filtering perspective entirely and instead treats SLAM as incremental smoothing problem, where the complete trajectory of the robot is recovered. This is in contrast to filtering approaches to SLAM that only maintain an estimate of the robot’s current location. In the smoothing framework, the entire past trajectory of the robot is subject to revision at each update. This means that smoothing approaches, unlike filtering ones, never make irreversible linearization errors, so the full non-linear solution can be obtained. The disadvantage of this scheme is that, because no marginalization of past poses is ever carried out, the number of variables in the estimator increases continuously while the robot is operating. However, because the relevant matrices typically remain sparse, the system can be solved in real time even for large environments. The same theme of applying incremental non-linear optimization techniques to SLAM was also proposed by Frese *et al.* [49] in 2005, who introduce a multi-level relaxation algorithm for SLAM that is derived from multigrid methods used for solving partial differential equations.

Recently popular are so-called pose graph optimization methods, for example the work of Olson *et al.* [116] and Grisetti *et al.* [54, 53]. These techniques apply a variant of Stochastic Gradient Descent

(a non-linear optimization method) to minimize the error in the network of constraints between robot poses. The techniques recover maximum likelihood map estimates only, but typically converge rapidly and recover good trajectories even when initial pose estimates are poor. Some non-linear optimization of this kind is usually necessary to recover good global map structure after an event known as *loop closure*, where the robot revisits a previously mapped area after traversing a large loop through unmapped terrain. In this condition the robot's position estimate and uncertainty may differ significantly from the ground truth, a condition which is not well handled by linearizing filters such as the EKF.

2.1.5 Navigation and Mapping without a Global Coordinate Frame

The dominant theme within the SLAM literature has been an approach known as *metric SLAM*, which aims to recover a globally consistent map of the environment in a single Cartesian coordinate frame. All algorithms discussed so far fall into this category. However, a robot can successfully perform most tasks with a map that satisfies much weaker requirements — for example, a map which is metrically consistent on the local scale, and merely topologically consistent on a global scale. In some cases a purely topological map, which specifies connectivity between landmarks but contains no metric information whatsoever, will suffice.

Topological theories of SLAM have existed for some considerable time [73, 132, 23], however these early topological mapping algorithms were not probabilistic, and hence difficult to make robust. Recently Ranganathan and Dellaert outlined a fully probabilistic framework for topological SLAM [127, 129, 128], which enabled principled treatment of the uncertainty in landmark identification. However, the computational complexity of the approach has so far limited its applicability to small environments containing only a few tens of landmarks.

Recent work by Sibley *et al.* [139, 138] proposes an interesting compromise which is metric in construction, but abandons the notion of a single Cartesian coordinate frame, thus leading to a global structure which is topological in character. Relative transformations are estimated between consecutive robot poses, but closed loops through the transformation graph are not constrained to be the identity transformation. This allows the map to represent unusual types of connectivity, such as between two floors of a building via an elevator, where the robot's ego-motion through the elevator is unobservable.

The resulting “map” does not necessarily have a Cartesian embedding, yet is nevertheless perfectly applicable for navigation tasks such as path planning and obstacle avoidance.

2.1.6 Challenging Problems in SLAM

A shared difficulty for all the SLAM algorithms described so far, regardless of estimation technique or map representation, is an issue known as *data association*. Data association is the problem of relating sensor measurements to objects in the robot’s map — a distance measurement to an object doesn’t tell us much about where we are unless we know which object we are observing. Data association also involves determining if measurements are spurious, or belong to a previously unobserved object not yet in the map. Correct data association is essential for successful navigation; errors rapidly lead to an incorrect map.

Theoretically, data association could be treated as simply another uncertain quantity to be estimated concurrently with the rest of the robot state. However, in practice this approach is intractable because the possible assignments of measurements to objects grow combinatorially¹. In deriving SLAM algorithms, it is frequently assumed that an “oracle” is available that supplies the data association information. In practice, of course, no oracle is available and some method of relating measurements to objects must be found. This is a non-trivial problem, and the core topic of this thesis. The limitations of the heuristic data association methods currently in use are the root cause of some of the most traditionally challenging problems in SLAM — the *loop closure detection* problem, the *kidnapped robot* problem and the *multi-session mapping* problem. A truly reliable method for general data association solves all these problems at once.

2.1.6.1 Local Data Association

The problem of associating measurements to map landmarks is commonly solved with very simple heuristic methods. One of the most common approaches is nearest neighbour gating (sometimes known as maximum likelihood data association), where a measurement is assigned to the “most likely” landmark in the map. Typically “most likely” is defined as the nearest landmark in the sense of Mahalanobis

¹The closest approach to treating the problem directly is [128].

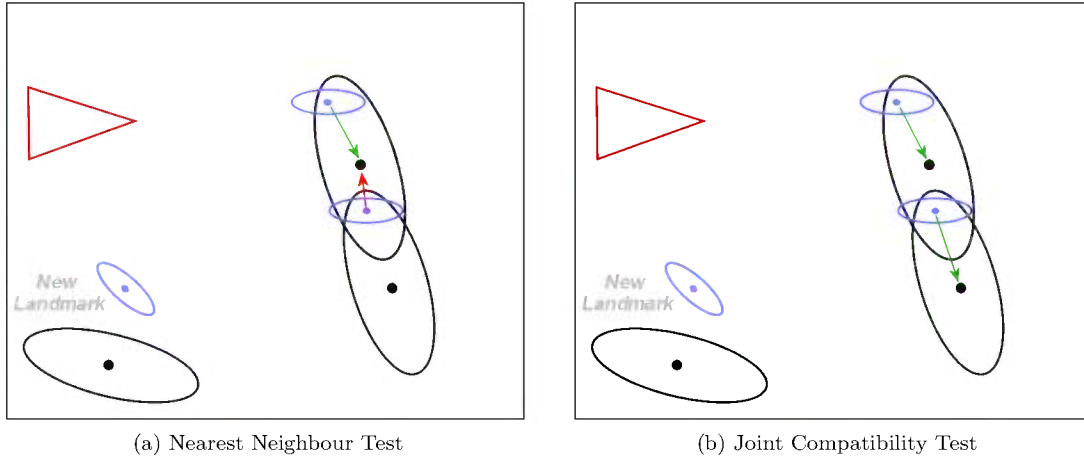


Figure 2.1: The local data association problem. Black points indicate estimated landmark locations with associated uncertainty ellipses. Measurements are shown in blue. In (a), nearest-neighbour assignment results in two measurements being assigned to a single landmark, corrupting the robot’s map estimate. In (b), the joint compatibility test [106] results in correct data association. Note that both the nearest neighbour and joint compatibility tests rely on a good prior estimate of robot location. Also shown is a measurement lies further than the gating threshold from any landmark, which will cause a new landmark to be added to the map.

distance — a distance metric which accounts for the fact that position uncertainty is not uniform in all directions [86]. Measurements are assigned to a new landmark if the most likely match is farther than some threshold. The scheme is illustrated in Figure 2.1.

The nearest neighbour gating scheme is very brittle. It is really applicable only when environmental clutter is low, sensor precision is high and vehicle pose uncertainty is low. Nevertheless, the basic approach is in common use. This is despite the fact that errors in data association typically cause catastrophic divergence of the map. To increase the robustness of the technique, a number of more advanced variants have been described. For example, the Joint Compatibility Branch and Bound technique [106] considers multiple measurements simultaneously, and so decreases the error rate by taking into account local compatibility constraints between measurements (Figure 2.1(b)). Another approach is to make use of additional sensory information associated with landmarks. Laser scan matching algorithms make use of larger scale shape information [134], visual SLAM systems make use of texture information associated with landmarks [36], etc. A second level of robustness can be achieved through multi-hypothesis tracking, where ambiguous data association cases are handled by tracking

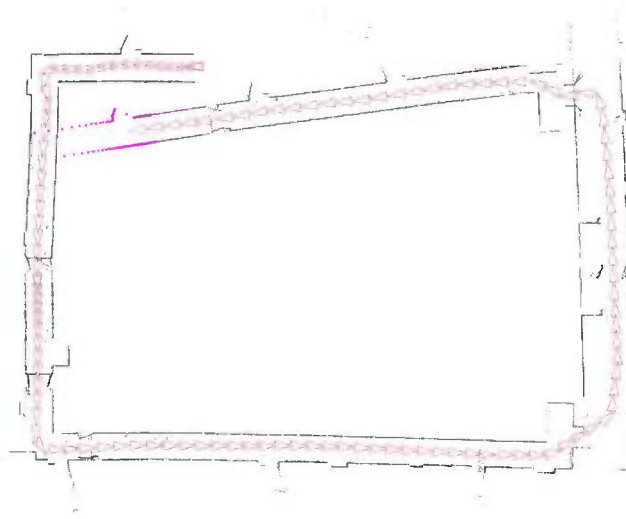


Figure 2.2: A example of loop closure detection failure in an indoor environment. Triangles show vehicle poses. As the robot travels along the corridor, linearization and perception errors accumulate. When the robot returns to the starting point, its true location lies outside the three-sigma bound on vehicle uncertainty (not shown). The nearest neighbour data association technique fails, and the robot's map becomes corrupted. (Figure courtesy of Oxford Mobile Robotics Group)

both possibilities in the hope that the ambiguity will be resolved by subsequent measurements [131]. However, all of these techniques are essentially similar in that they address the local data association problem — that is, they work only if we have a good initial guess for the robot's position within the map. Such information is usually not available in a situation known as loop closure, where other data association techniques are required.

2.1.6.2 The Loop Closure Detection Problem

A particularly difficult instance of the data association problem is known as *loop closure detection*. Loop closure detection is essentially the problem of data association without a good prior guess for the robot's position within the map. The problem commonly arises when the robot is returning to a previously mapped location after traversing a large loop through an unmapped area. In such cases the robot's position estimate can differ markedly from the ground truth. This does not imply that the robot made gross errors at any point, rather that small errors have been integrated over a large path length — for example in Figure 2.3, small angular errors along the path result in the end points

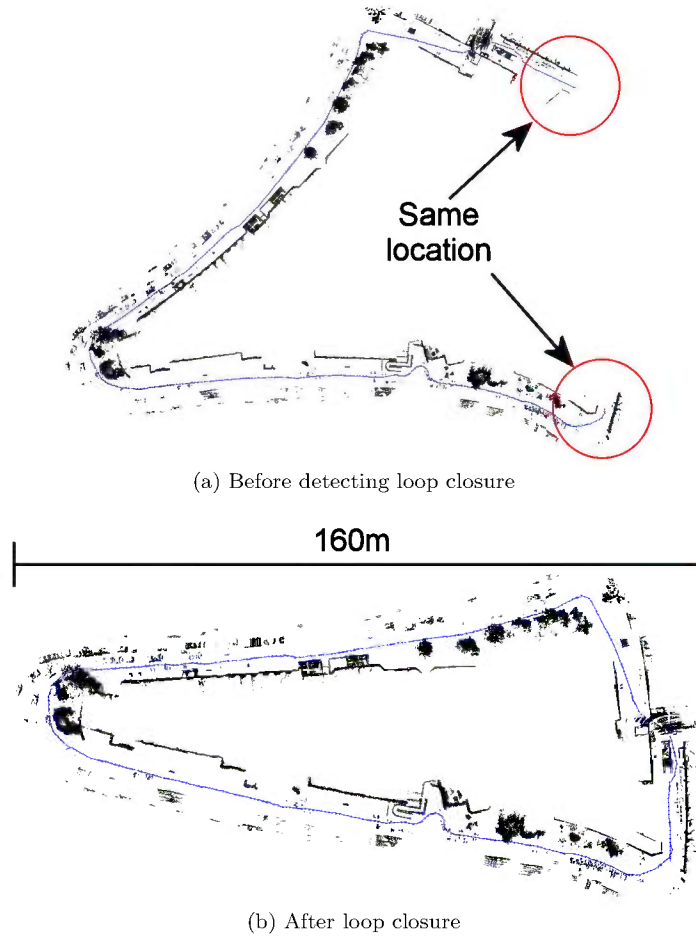


Figure 2.3: An illustration of the difficulty of loop closure detection using global metric estimates of vehicle position. (Figure courtesy of Alastair Harrison and Dave Cole.) Sub-figure (a) shows the vehicle trajectory estimate produced by a scan-matching SLAM system operating in an outdoor environment [111]. The end points of the trajectory are at the same location; however small angular errors along the trajectory have produced a large difference between the true and estimated positions of the vehicle. The corrected trajectory is shown in sub figure (b), where the loop closure has been detected on the basis of an image match. It would seem that in large-scale environments, such a loop closure detection system is essential for reliable navigation.

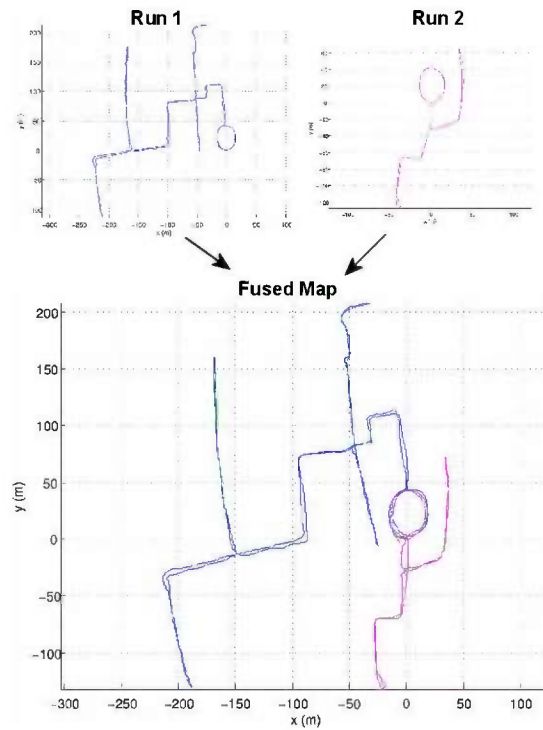


Figure 2.4: Multi-session mapping. Two partial maps of the environment were collected on different days. The transformation between the two map coordinate frames is unknown and must be recovered to produce the fused map.

of the loop having widely separated global position estimates. The problem is compounded by the fact that, for trajectories involving uncertain rotations, the vehicle position uncertainty cannot be correctly represented by Gaussian models such as the EKF. A second failure example, from an indoor environment, is shown in Figure 2.2.

In the case of loop closure, where the robot's estimate of its position is in gross error, local data association techniques such as those discussed in the previous section are inapplicable. Relying on such techniques alone will result in previously mapped areas being remapped in a new global location and a vehicle that is effectively lost. A great many techniques have been proposed to solve the loop closure detection problem; we will review them in detail in Section 2.3.

2.1.6.3 The Kidnapped Robot Problem and Multi-Session Mapping

The kidnapped robot problem and the multi-session mapping problem are two further tasks that have traditionally been challenging for SLAM systems. The kidnapped robot problem is the task of determining a robot's pose within a map, without prior position information. It might occur if a robot is deactivated and then moved, experiences a temporary sensor failure, etc. The multi-session mapping problem is the task of aligning two partial maps of the environment collected by the robot during different periods of operation, illustrated in Figure 2.4. The rotation, translation (and in some cases scale) between the two maps is unknown and must be determined automatically. Both of these tasks are common and important elements of an autonomous system capable of life-long operation.

As with loop closure detection, both kidnapped robot and multi-session mapping can be seen as a subset of the data association problem without prior position information, although problem-specific techniques have been proposed that solve these problems individually without tackling the more general case. We will review some of these techniques when we consider the SLAM literature on place recognition in Section 2.3. First, however, we introduce relevant background work in computer vision.

2.2 Content Based Image Retrieval Systems

The second strand of literature drawn upon in this thesis is the work on content based image retrieval systems in the computer vision community. Given a “query image”, these systems attempt to rapidly locate other images of the same or similar objects within a large image collection, even in the presence of image transformations such as perspective or lighting change, partial occlusion, etc. Notable examples of such systems include Sivic and Zisserman [144], Nister and Stewénus [114], Philbin *et al.* [122] and Jégou *et al.* [63]. Essentially all of these systems are based on quantized invariant local features, also known as the “bag-of-words”, “bag-of-visual-words” or “bag-of-visual-features” representation. We begin our discussion by introducing this image representation.

2.2.1 The Bag of Words Representation

To efficiently find image matches in large databases under viewpoint transformation, the choice of representation of the image data is crucial. Some early image retrieval systems represented images using global features, where a single descriptor is computed for an entire image (for example, global colour histograms [149, 156] or PCA components [154, 72]). Such global features are not very robust to effects such as variable lighting, perspective change or partial occlusion of the scene. An alternative approach pioneered in the late 1990s by Schmid and Mohr [137] locates *interest points* in the image and computes *invariant features* (or “*descriptors*”) in the neighbourhood of these interest points. The interest point detectors and local descriptors can be designed such that they are repeatable even after a variety of image transformations such as scaling, rotation and linear lighting changes. To perform image search, the local features are computed in the query image, and for each local feature a “vote” is cast for images in the image database which contains a similar local feature. The voting procedure enables object recognition even when the query contains a partially occluded view of the object being searched for, and also provides robustness against a detector/descriptor which is only stochastically repeatable. Note that the representation discards information about the relative geometric configuration of the interest points. This local-feature paradigm proved highly successful, especially after the introduction of the SIFT descriptor by Lowe [83].

Image search via direct local descriptor matching is effective in image collections of moderate size (hundreds to thousands of images), but becomes computationally difficult for larger image collections (tens of thousands to millions of images). The computational bottleneck is the matching of descriptors in the query image to near neighbours in the image database. A typical image may yield thousands of local descriptors, so matching a descriptor from a query image typically involves searching through tens to hundred of millions of descriptors in the image database. Descriptors are typically high-dimensional (for example, the SIFT descriptor is 128 dimensional), and nearest neighbour search for high dimensional data is computationally challenging. The most widely used practical search algorithm, the kd-tree [102], is not much more efficient than linear search when the dimensionality is high. Approximate nearest neighbours can be found much more quickly, for example using the Best Bin First variant of

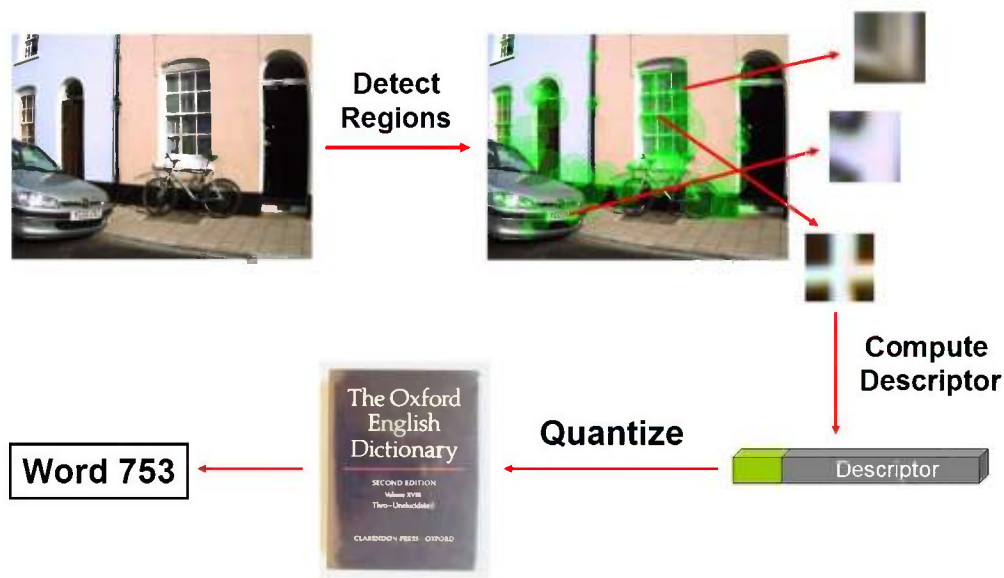


Figure 2.5: Illustration of how an image is converted to the visual word representation. First, regions of interest are detected in the image. An invariant descriptor is computed for each interest region. These descriptors are then quantized with respect to a dictionary of prototypical features, and the descriptor is replaced by an integer identifying the nearest prototypical descriptor in the dictionary (“*visual word id*”). The whole image can then be represented by a short list of integers specifying which visual words occurred in the image, and how often each word occurred.

kd-tree search [8], but for very large image search problems even these techniques are both too slow and (especially) too memory intensive to be practical.

The key innovation that enabled search of very large image collections was the “bag of words” model [136, 80, 144], which quantizes image descriptors against a “vocabulary” of prototypical descriptors. The vocabulary is learned by clustering descriptors from training data. Quantization allows a high-dimensional image descriptor to be replaced by a single integer (or *word ID*) specifying which prototypical descriptor in the vocabulary it is most similar to. The process is illustrated in Figure 2.5. A *word* thus corresponds to a region in the space of invariant descriptors, the assumption being that descriptors within this region can be considered identical for the purposes of the task at hand. Images can now be represented as a short list of integers specifying which visual words occurred in the image, and how often they occurred — a representation identical to that used for text documents in internet search engines. The representation thus enables fast visual search through the application of methods developed for

text retrieval. These techniques were first applied to large scale visual search in the work of Sivic and Zisserman [144].

For the purposes of this thesis we will take the visual word representation as fundamental. The details of the interest point detectors and descriptors that underlie the representation will not be of particular concern to us. It is sufficient to know that we have some algorithm that takes an image and produces a list of vector-valued features that are somewhat repeatable and somewhat invariant to common image transformations. Our primary concern will be in constructing a probabilistic inference framework on top of this representation. For completeness, however, we briefly review the various detectors and descriptors in the following sections.

2.2.2 Interest Point Detectors

A large number of interest point detectors have been proposed in the literature. Popular examples include Harris corners [56], Harris-Laplace and Hessian-Laplace points and their affine-invariant extensions the Harris-Affine and Hessian-Affine points [96], the Difference of Gaussians scale space extrema used in SIFT [84], Maximally Stable Extremal Regions (MSERs) [90], and the box filter approximation to the determinant of the Hessian used in SURF (known as Fast-Hessian) [6]. Most of these detectors select local maxima of simple operators in image scale space, and typically respond to structures such as corners or blobs. An extensive performance evaluation was carried out by Mikolajczyk *et al.* [98], where the repeatability of the interest points was tested with respect to viewpoint change, zoom, rotation, blur, JPEG compression and lighting change. Overall the Hessian-Affine and MSER detectors were found to perform best, MSERs being particularly robust to viewpoint and lighting change and Hessian-Affine being better in the presence of blur and JPEG compression. The Fast-Hessian operator used by SURF was later found to have performance comparable or slightly superior to Hessian-Laplace at much lower computational cost [6], though not competitive with Hessian-Affine in the case of large viewpoint changes.

2.2.3 Interest Point Descriptors

A wide variety of interest point descriptors have also been described. The dominant paradigm is the histogram-type descriptor introduced by Lowe [84], which is based on the spatial distribution of local features within the interest point neighbourhood. Lowe's SIFT descriptor is a histogram of local image gradient responses in 8 orientations over a 4x4 spatial grid, yielding a 128D vector descriptor. Rotation and illumination invariance is achieved by intensity normalizing the image patch and rotating in the direction of dominant local gradient. These histogram-type descriptors have excellent performance because they capture sufficient information so as to be highly discriminative, yet are tolerant of small image deformations and errors in interest point localization. An extensive evaluation by Mikolajczyk and Schmid [95] found them to consistently outperform a wide variety of alternatives.

Several papers have investigated variations on the original SIFT design. The SURF descriptor, based on histograms of Haar wavelet responses, broadly retains the idea of a descriptor based on the spatial distribution of local gradients, but changes the details to aid efficient implementation. The resulting descriptor was shown to have matching performance superior to SIFT, while also being faster to compute [6, 7]. Further performance improvements were achieved by Winder and Brown [165, 166], where the form of the descriptor was learned directly so as to maximize matching performance on training data. The resulting descriptor designs substantially out-perform SIFT in terms of both error rate and descriptor size – the best configuration resulting in an error rate half that of SIFT and a memory footprint 10x smaller. Other recent innovations include GPU-based implementations of descriptors and detectors, which enables feature extraction in only a few milliseconds on typical 640x480 images [28]. Finally, a notable alternative to the SIFT-like descriptor is the randomized tree technique of Lepetit *et al.* [79], which poses keypoint matching as a classification problem. Randomized trees are favoured in some real-time systems because they can be computed very rapidly even on standard hardware.

2.2.4 Detector and Descriptor used in this Thesis

As noted previously, the work described in this thesis is descriptor-agnostic. During development we experimented with several possibilities, principally the Harris-Affine detector with SIFT descriptor, and the SURF detector/descriptor. The real-time performance requirements of our robotics application lead us to choose the SURF descriptor for all final results. We use the U-SURF variant with 128D descriptor [6]. We disable rotation invariance because this version of the descriptor is faster to compute and, because in-plane rotation is a rare occurrence on our robot platform, it also provides a modest increase in matching performance. Some of the more recent results in descriptor design, notably those of [166], might provide performance improvements for future versions of our system.

2.2.5 Partitioning of Descriptor Space

The final step in producing a bag-of-words representation of an image is to construct a partitioning of descriptor space that defines the mapping from interest point descriptor to visual word. The standard approach is to cluster a set of descriptors and use the cluster centres to implicitly divide the descriptor space into Voronoi regions. See Figure 2.6. The most common choice of clustering algorithm is k-means [144]. For larger vocabularies hierarchical k-means [114] or approximate k-means [122] are used for tractability. Others have investigated incremental radius-based clustering [69, 60, 45] to overcome some problems with k-means, an issue we will discuss in detail in Chapter 5. Other techniques such as agglomerative clustering have also been considered [77, 97], though primarily for problems such as object class recognition where vocabularies are smaller and so more expensive clustering procedures are feasible. An alternative to clustering was explored in [155] where a regular grid was used to discretize the descriptor space.

In our experience the choice of clustering algorithm is of secondary importance, at least for smaller retrieval problems. Indeed, an early version of our system had a bug in the clustering code that resulted in very poor quality clusterings. This went unnoticed for several weeks because the system continued to detect loop closures successfully. This behaviour should not be too surprising — the primary purpose of the clustering is to partition the descriptor space into regions, rather than to locate structure in

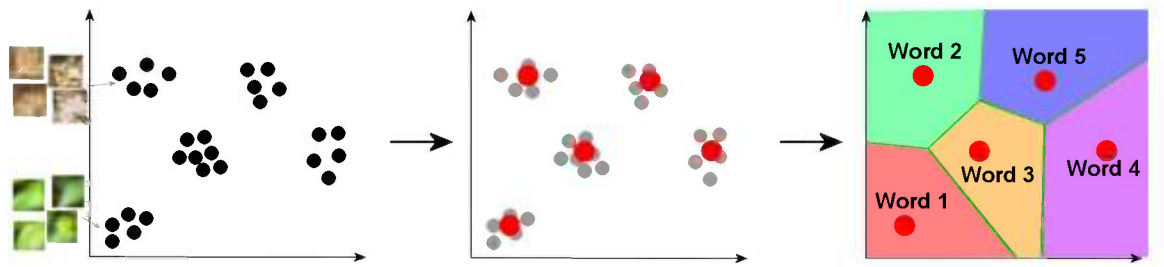


Figure 2.6: Descriptor space partitioning (shown in 2D for illustration). Descriptors from similar real-world objects tend to cluster in descriptor space - for example, foliage may form one cluster and brickwork another. Cluster centres (shown in red) are found via an unsupervised clustering algorithm such as k-means. The descriptor space is then implicitly divided into visual words, defined by the Voronoi regions of the cluster centres.

the data. While intuition would suggest that good quality clusterings should improve performance because they keep quantization boundaries away from dense regions of the data (which would tend to amplify quantization error due to small changes in descriptors), retrieval should still be possible with a random partitioning of descriptor space. Only highly degenerate partitionings where a small number of visual words occupy a large volume of descriptor space are likely to actually “break” the system. These issues are relevant when considering generalization performance – for example, should we expect a robot whose visual vocabulary has been trained on indoor images to be able to navigate in an outdoor environment? As might be clear from the preceding discussion, we expect system performance to degrade gracefully in such situations. We will observe this behaviour in practice in Chapter 3.

2.2.6 Notable Image Retrieval Systems

Large scale image retrieval using the bag-of-words approach was first introduced by Sivic and Zisserman in [144]. Later Nister and Stewénus [114] developed the vocabulary tree which combined vocabulary generation, quantization and indexing into a single structure based on a hierarchical k-means clustering of the descriptors in the image database. The paper explored the use of very large visual vocabularies (up to 10^6 words), and demonstrated that this gave a significant increase in performance. A hard drive based scheme for extending the approach beyond the RAM limits of a single machine was discussed in [48].

Philbin *et al.* [122] also experimented with the use of large visual vocabularies, and demonstrated that an approximate k-means clustering combined with a randomized forest of kd-trees for quantization was superior to the hierarchical k-means approach of [114]. Philbin *et al.* additionally introduced a fast method for verifying the geometric consistency of putative image matches returned by the bag-of-words ranking procedure. The approach took advantage of the shape information in the affine-invariant interest points, yielding a method fast enough to apply to the top several hundred images in the ranking. The geometric verification step gave a sizable performance improvement, particularly with smaller visual vocabularies or larger retrieval sets, however at the cost of increasing the memory requirements of the index. Subsequent work by the same authors extracted further performance improvements through the use of relevance feedback [26] and explored the impact of quantization accuracy on system performance [123] through the use of soft assignment. Soft assignment gave performance improvement, but again at the cost of increasing the storage requirements of the index. Recently Perd'och *et al.* [121] proposed a technique to reduce the memory required to store the shape information of the interest points used during the geometric verification stage.

Jégou *et al.* [63, 64] introduced a technique called the Hamming Embedding for improving quantization beyond the granularity of visual words, by storing a short binary code for each point descriptor. They also introduced a method to build weak geometric consistency directly into the ranking stage (rather than as a post-processing step). The geometric ranking required some restrictive assumptions, but proved to increase performance on typical data.

A common denominator of the systems discussed so far is that similarity between bag-of-words vectors used in the core ranking stage of retrieval is computed using a term-frequency inverse-document-frequency (tf-idf) weighted cosine distance. Tf-idf is a standard ranking measure used throughout information retrieval [87], which builds on the intuition that two documents containing a rare word should be more similar than those that share a common word. However, it has relatively little theoretical justification beyond the fact that it has good empirical performance on text retrieval tasks. Most production text retrieval systems apply many tweaks to the measure to overcome various deficiencies [87]. A core argument of this thesis will be that the tf-idf ranking is not particularly suitable for image data, and that by paying attention to the specific qualities of imagery, much better ranking measures

can be derived. Some very recent literature has explored similar ideas; for example [65], where some heuristic ranking measures are proposed to better account for the “bursty” statistics of visual words. Also of note are some recent papers which, with different motivation, have exploited the co-occurrence of visual features. For example [153] uses fast query-dependent AND searches to rapidly locate a small set of images for more complex ranking, and [168] is one of several examples of approaches that combine multiple visual features into meta-features.

Finally, some interesting techniques have been proposed in papers dealing with problems closely related to image search, such as image clustering and object recognition with large image databases. Chum *et al.* introduced geometric min-Hash [25], a hashing based retrieval scheme that incorporates the local geometry of interest points directly into the retrieval stage. This was demonstrated in the context of automatically discovering repeated views of objects in the image set. Torralba *et al.* [151, 161] investigated a number of techniques for learning small-code approximations to image descriptors. An entire image is represented by a code of only a few hundred bits, giving a retrieval scheme which is fast to search and extremely memory efficient. A search index for tens of millions of images can be stored in less than 1 GB RAM. However, the technique has been evaluated only for retrieving images that are weakly similar to the query image; it is not clear how the method would perform for the exact matching tasks considered by the other papers discussed here.

2.3 Appearance Based Techniques in Robotics

We now return to considering the appearance based navigation problem in robotics. In Section 2.1 we introduced the loop closure detection problem and discussed the difficulty of addressing it within a typical metric SLAM framework. The root cause of this difficulty is the simplicity of the heuristics used for data association, which typically require a very good prior position estimate to be effective. To better address the general data association problem, *appearance based techniques* have been developed which exploit the rich appearance information from sensors such as cameras to determine when the robot is revisiting a previously mapped place. Typically these techniques can perform a global search through previously seen locations, so they are not reliant on position priors and are a useful approach

to loop closure detection. Further, because location estimates based on appearance have uncertainty essentially orthogonal to that in metric estimates, they are a useful complement to metric SLAM systems in general. We review the development of appearance based techniques in this section.

2.3.1 Localization using Appearance

We begin our review of the literature by considering appearance based localization techniques. Although the localization problem is considerably simpler than SLAM (the difficult issue of new place detection does not arise), appearance based approaches to the two problems share many common concerns. In particular, the appearance based localization literature has extensively examined the issue of how best to represent and match sensory appearance information. Early examples include representing appearance as a set of image histograms [156], and as ordered sequences of edge and colour features [76]. Several authors explored the use of more general dimensionality reduction techniques. For example, Kröse *et al.* [72] used Principal Component Analysis to reduce the dimensionality of the incoming images. Places are then represented as a Gaussian density in the low dimensional space, which enables principled probabilistic localization. Ramos *et al.* [126] employed a dimensionality reduction technique combined with variational Bayes learning to find a generative model for each place. (A drawback of both of these approaches is that they require significant supervised training phases to learn their generative place models). Bowling *et al.* [18] described an unsupervised approach that uses a sophisticated dimensionality reduction technique called Action Respecting Embedding; however the method suffered from high computational cost [17] and yielded localization results in a “subjective” representation which is not straight forward to interpret. Applying image-specific features has generally been a more successful approach — for example, Torralba *et al.* [152] represented places by a set of texture features, and described a system where localization is cast as estimating the state of a hidden Markov model.

The methods described so far represent appearance using global features, where a single descriptor is computed for an entire image. As discussed in our overview of the computer vision literature in Section 2.2, representations based on sets of local features were found to be a more robust approach. Recent appearance-based techniques use this representation. An early example of the approach was described in [143, 40]. More recently, [167] used an image retrieval system based on invariant features as the basis of

a Monte Carlo localization scheme. The issue of selecting the most salient local features for localization was considered in [82]. Most recently the bag-of-words representation has become popular, and many authors have proposed extensions to the basic approach. Wang *et al.* [160] confirmed putative matches in a geometric post-verification step. Filliant described a system where the visual vocabulary is learned online [45]. Ferreira *et al.* employed a Bernoulli mixture model to capture some of the conditional dependencies between words in the vocabulary [44]. Schindler *et al.* described how to tailor visual vocabulary generation so as to yield more discriminative visual words [135], and discuss the application of the technique to city-scale localization with a database of 30,000 images.

Finally, a recent paper proposed an interesting alternative to the local feature model, known as the *epitomic* representation [112]. The paper reports strong place recognition performance; however performance was compared only to the work of Torralba *et al.* [152]. Relative performance with respect to bag-of-words methods remains to be investigated.

2.3.2 Loop Closure Detection

The work discussed so far has been mainly concerned with the localization problem, where the map is known a-priori and the current sensory view is guaranteed to come from somewhere within the map. To use a similar appearance-based approach to detect loop closure in SLAM, the system must be able to deal with the possibility that the current view comes from a previously unvisited place and has no match within the map. As pointed out in [55], this makes the problem considerably more difficult. Particularly challenging is the *perceptual aliasing* problem — the fact that different parts of the workspace may appear the same to the robot’s sensors. As noted in [140], this can be a problem even when using rich sensors such as cameras due to repetitive features in the environment, for example mass-produced objects or repeating architectural elements. Gutmann and Konolige [55] tackle the problem by computing a similarity score between a patch of the map around the current pose and older areas of the map to identify possible loop-closures. They then employ a set of heuristics to decide if the putative match is a genuine loop closure or not. While these heuristic methods achieved some success, they did not provide a satisfactory solution to the perceptual aliasing problem. Goede-mé described an approach to the issue using Dempster-Shafer theory, using sequences of observations to confirm or

reject putative loop closures [52]. The approach involved separate mapping and localization phases, so was not suitable for unsupervised mobile robotics applications.

A number of methods based on *similarity matrices* were proposed as a way to extend appearance-based matching beyond pure localization tasks. These methods define a similarity score between observations, then compute the pairwise similarity between all observations to yield a square similarity matrix. If the robot observes frequently as it moves through the environment and observations are ordered by time, then loop closures appear as off-diagonal stripes in this matrix. Levin and Szeliski describe such a method [81] that uses a cascade of similarity functions based on global colour histograms, image moments and epipolar geometry. Silpa-Anan and Hartley describe a similar system [140, 141] which employs SIFT features. Zivkovic *et al.* [170] consider the case where the order in which the images were collected is unknown, and use graph cuts to identify clusters of related images. A series of papers by Ho and Newman [111, 108, 59, 58, 60] advanced similarity matrix based techniques by considering the problem of perceptual aliasing. Their approach is based on a singular value decomposition of the similarity matrix that eliminates the effect of repetitive structure. They also address the issue of deciding if a putative match is indeed a loop closure based on examining an extreme value distribution related to the similarity matrix.

Similarity matrix based techniques were effective, however the only method which explicitly tackled the perceptual aliasing problem (Ho and Newman) was limited by $O(n^3)$ complexity. Further, the similarity matrix approaches were not truly probabilistic in construction, and so did not integrate very smoothly into a more general SLAM framework. In 2007 we outlined a new approach to appearance based navigation which addressed the perceptual aliasing problem in a probabilistic framework, and furthermore had $O(n)$ runtime complexity [30]. The details of this algorithm we will discuss in Chapter 3.

2.3.3 Recent Developments

While appearance-based navigation has a long history within robotics, there has been considerable development in the field within the timespan of this thesis. Appearance-based navigation and loop closure detection systems operating on trajectories on the order of a few kilometers in length are now

commonplace [2, 47, 20, 169, 85]. Indeed, place recognition systems similar in character to the one described in this thesis have become an important component even of single-camera SLAM systems designed for small-scale applications [42, 163].

Use of these systems on the scale of tens of kilometers or more has also begun to be feasible. For example Milford and Wyeth [99] described a large scale appearance-based navigation experiment using a system based on set of biologically inspired techniques. The system achieved successful loop closure detection and mapping in a collection of more than 12,000 images from a 66 km trajectory, with processing time of less than 100 ms per image. The appearance-recognition component of the system is based on direct template matching, so scales linearly with the size of the environment. Operating at a similar scale, Bosse and Zlot describe a place recognition system based on distinctive keypoints extracted from 2D lidar data [14], and demonstrate good precision-recall performance over an 18 km suburban data set. Related results, though based on a less scalable correlation-based submap matching method, were also described in [15].

A number of recent papers also describe integrated systems which tightly couple appearance and metric information. Olson, for example, described an approach to increasing the robustness of general loop closure detection systems by using both appearance and relative metric information to select a single consistent set of loop closures from a larger number of candidates [115]. The method was evaluated over several kilometers of urban data and shown to recover high-precision loop closures even with the use of artificially poor image features. Blanco *et al.* described a system where metric and topological position information is considered jointly in the estimator [11]. More loosely coupled systems were also described in [71, 110].

Finally, authors have also begun to consider the problem of place recognition in the presence of major environmental variation — for example, initial results for cross-season place recognition were presented in [157].

Chapter 3

The FAB-MAP Model

This chapter introduces a new probabilistic model for navigation and mapping using appearance data, which we call Fast Appearance Based Mapping or FAB-MAP. The technique allows a mobile robot to compute a probability distribution over its location based on the sensory appearance of the local environment alone, without using any metric information. Navigation methods based on sensory similarity have been proposed before – this work goes beyond previously published methods in that it assesses not the similarity of two observations but rather the probability that they originate from the same location. This is achieved by learning a generative model of appearance, which enables us to assign identical but common observations a low probability of having come from the same place. The best previous solutions to the appearance-based navigation problem in visually repetitive environments had complexity cubic in the number of locations in the map [60]. The algorithm presented here has linear complexity, extending its applicability to much larger environments, and making it particularly suitable for online loop closure detection in mobile robotics.

The basic form of our system is inspired by the bag-of-words image retrieval systems developed in the computer vision community [144, 114]. However, we extend the approach by learning a generative model for the bag-of-words data. This generative model captures the fact that certain combinations of appearance words tend to co-occur, because they are generated by common objects in the environment. Our system effectively learns a model of these common objects (without supervision, see Figure 3.9),

which allows us to improve inference by reasoning about which sets of features are likely to appear or disappear together (see Figure 3.18). We present results which demonstrate that the system is capable of recognising places even when observations have few features in common, while rejecting false matches due to perceptual aliasing even when such observations share many features. Our approach also has reasonable computational cost – fast enough for online loop closure detection in realistic mobile robotics problems where the map contains several thousand places.

The focus in this chapter will be on the development and validation of the probabilistic model. We leave other issues relevant to improving system performance, such as geometric checks and details of visual word generation, for discussion in the context of the large scale experimental verification in Chapter 5.

3.1 Approximating High Dimensional Discrete Distributions

This section introduces some necessary background material on machine learning techniques used by our system. Readers familiar with Chow Liu trees may wish to skip to Section 3.2.

Consider a distribution $P(Z)$ on n discrete variables, $Z = \{z_1, z_2, \dots, z_n\}$. We wish to learn the parameters of this distribution from a set of samples. If $P(Z)$ is a general distribution without special structure, the space needed to represent the distribution is exponential in n — as n increases dealing with such distributions quickly becomes intractable. A solution to this problem is to approximate $P(Z)$ by another distribution $Q(Z)$ possessing some special structure that makes it tractable to work with, and that is in some sense similar to the original distribution. The natural similarity metric in this context is the Kullback-Leibler divergence [74], defined as:

$$D_{KL}(P, Q) = \sum_{x \in X} P(x) \log \frac{P(x)}{Q(x)} \quad (3.1)$$

where the summation is carried out over all possible states in the distribution. The KL divergence is zero when the distributions are identical, and strictly larger otherwise.

The Naive Bayes approximation is an example of a structural constraint that allows for tractable

learning of $Q(Z)$. Under this approximation $Q(Z)$ is restricted such that each variable must be independent of all others. This extreme structural constraint limits the degree to which $Q(Z)$ can represent $P(Z)$. A less severe constraint is to require each variable in $Q(Z)$ to be conditioned on at most one other variable — this restricts the graphical model of $Q(Z)$ to be a tree. Unlike the Naive Bayes case, where there is only one possible graphical model, in this case there are n^{n-2} possible tree-structured distributions that satisfy the structural constraint [162]. Chow and Liu [24] describe a polynomial time algorithm to select the best such distribution, which we elaborate on below.

More generally we could constrain $Q(Z)$ such that each variable is conditioned on at most n other variables. This structure is known as a *thin junction tree*. It was shown in [148] that the problem of determining the optimal thin junction tree is NP-hard, though there are several methods to find a good approximate solution (for example [150]).

3.1.1 Chow Liu Trees

The Chow Liu algorithm approximates a discrete distribution $P(Z)$ by the closest tree-structured Bayesian network $Q(Z)_{opt}$, in the sense of minimizing the Kullback-Leibler divergence. The structure of $Q(Z)_{opt}$ is determined by considering a mutual information graph $\mathcal{G}$. For a distribution over n variables, $\mathcal{G}$ is the complete graph with n nodes and $\frac{n(n-1)}{2}$ edges, where each edge (z_i, z_j) has weight equal to the mutual information $I(z_i, z_j)$ between variable i and j :

$$I(z_i, z_j) = \sum_{z_i \in \Omega, z_j \in \Omega} p(z_i, z_j) \log \frac{p(z_i, z_j)}{p(z_i)p(z_j)} \quad (3.2)$$

and the summation is carried out over all possible states of z . Mutual information measures the degree to which knowledge of the value of one variable predicts the value of another. It is zero if two variables are independent, and strictly larger otherwise.

Chow and Liu prove that the maximum-weight spanning tree [27] of the mutual information graph will have the same structure as $Q(Z)_{opt}$ (see Figure 3.1). Intuitively, the dependencies between variables that exist in $P(Z)$ but are excluded from $Q(Z)$ have as little mutual information as possible, and so are the best ones to approximate as independent. Chow and Liu further prove that the conditional

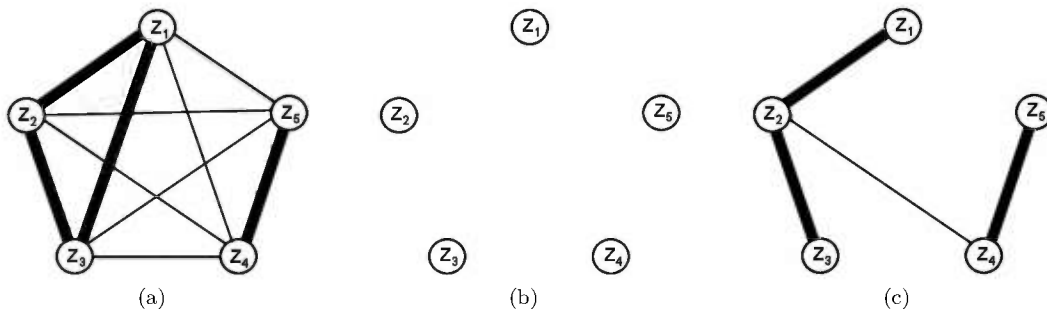


Figure 3.1: **(a)** Graphical model of the underlying distribution $P(Z)$. Mutual information between variables is shown by the thickness of the edge. **(b)** Naive Bayes approximation. **(c)** Chow Liu tree.

probabilities $p(z_i|z_j)$ for each edge in $Q(Z)_{opt}$ are the same as the conditional probabilities in $P(Z)$ – thus maximum likelihood estimates can be obtained directly from co-occurrence frequency in training data. To mitigate potential problems due to limited training data, we estimate these probabilities using a pseudo-Bayesian smoothing estimator described in Section 3.1.3 below, rather than the maximum likelihood estimator. This prevents any probabilities from having degenerate values of 0 or 1.

In the following section we will use Chow Liu trees to approximate large discrete distributions. Because we are interested in learning distributions over a large number of variables ($\sim 10,000$) the $O(n^2)$ mutual information graph to be computed is very large, typically too large to be stored in RAM. To deal with this, we use a semi-external spanning tree algorithm [39]. The mutual information graph is required only temporarily at learning time – it is discarded once the maximal spanning tree has been computed. Meilă [92] has described a more efficient, though complex, method of learning Chow Liu trees that avoids the explicit computation of the full mutual information graph, and provides significant speed-up for “sparse data” such as ours. In Chapter 5 we will work with even larger Chow Liu trees over $\sim 100,000$ variables, in which case use of Meilă’s algorithm will become essential.

We have chosen to use Chow Liu trees because they are tractable to compute even for very high-dimensional distributions, are guaranteed to be the optimal approximation within their model class, and require only first order conditional probabilities, which can be reliably estimated from available training data. However, we are essentially interested only in capturing dependencies between variables and there is nothing that restricts us to using a Chow Liu tree for this purpose. Our approach could

substitute a more complex model such as a mixture of trees [94], a thin junction tree of wider tree-width [150, 43], or with some modification make use of other classes of learning techniques, for example deep belief networks [130]. We will revisit these possibilities in Chapter 6.

A simple example to illustrate the performance of Chow Liu trees for density estimation is given in Appendix B.

3.1.2 Sample Complexity of Learning

When learning Bayesian networks from data, the question arises as to how much training data is required – this is known as the sample complexity of learning. The question can be asked separately for the problems of learning the *structure* of the network, and for the *parameters* of a network with known structure. The question will appear particularly relevant in Chapter 5 when we will consider learning Chow Liu trees over a 100,000 variables from only thousands of training examples.

There is much support for the practice of learning large networks from relatively small quantities of training data. Recent theoretical work by Abbeel [1] has shown that for bounded fan-in Bayesian networks (such as Chow Liu trees), the sample complexity of parameter learning is independent of the number of variables in the network. For the structure learning problem, sample complexity in the bounded fan-in case is known to be at worst logarithmic in the number of variables in the network, at least when the network structure is required to be only close to optimal [62]. Empirical results are in agreement with these predictions. In [94], Chow Liu trees were shown to be effective as classifiers even when available training data was limited relative to the dimensionality of the data. The results of [22] show similar behaviour when using Chow Liu trees as density estimators. Notably, they also demonstrate that in the regime where training data is limited relative to problem dimensionality, the Chow Liu algorithm consistently out-performs models which attempt to learn more complex large-tree-width Bayesian networks.

3.1.3 Smoothing Probability Estimates

Given the structure of the Bayesian network, the relevant conditional probabilities that specify the distribution must also be learned from the training data. When training data is limited, the maximum

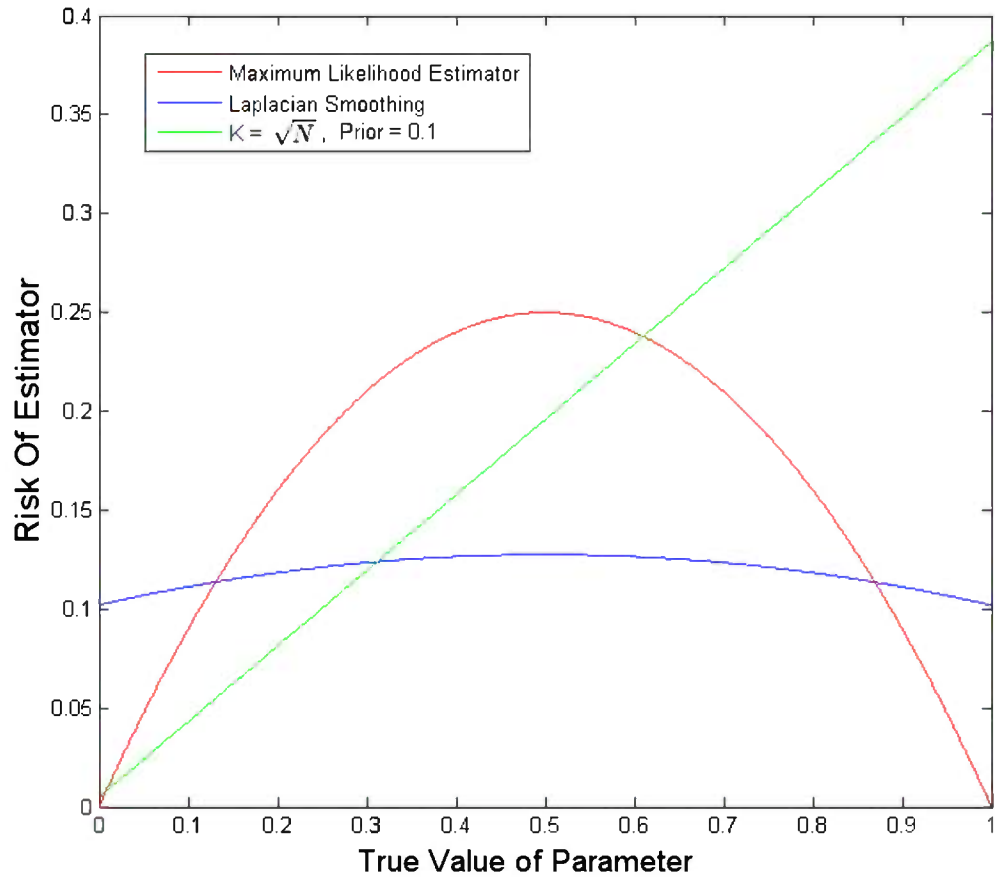


Figure 3.2: Risk function of different estimators, for a binary random variable estimated from five observations. The horizontal axis is the true probability of the random variable being observed, and the vertical axis is the risk of the estimator. The maximum likelihood estimator is shown in red, Laplacian smoothing in blue, and an estimator with $K = \sqrt{N}$ and $p_{prior} = 0.1$ in green.

likelihood estimate of some probabilities $p(z_i = s_i | z_j = s_j)$ can be 0 or 1, due to sampling error. To avoid this effect, the probability estimates must be smoothed. A variety smoothing techniques exist [34], typically of the form

$$p_{smooth} = \left(\frac{N}{N+K}\right)p_{mle} + \left(\frac{K}{N+K}\right)p_{prior} \quad (3.3)$$

where p_{mle} is the maximum likelihood estimate of the parameter, p_{prior} is a prior, N is the number of observations and K determines the relative weighting between the prior and the maximum likelihood estimate. For example, choosing $K = 2$ and $p_{prior} = 0.5$ gives Laplacian smoothing. We use the estimator described in [10], where $K = \sqrt{N}$ and the prior for $p(A|B)$ is set to $P(A)$.

The quality of such an estimator may be characterized by its risk function

$$R(\hat{p}, p) = E \left(\|\hat{p} - p\|^2 \right) \quad (3.4)$$

which is the expectation of the squared distance between the estimate $\hat{p}$ and the true value p . The risk functions for several different estimators are illustrated in Figure 3.2.

3.1.4 What the Chow Liu Tree Learns

To conclude this section, it is useful to build a little intuition about what the Chow Liu tree is capturing, and how it will be useful to an appearance-based SLAM system. The model we will employ in the rest of this chapter takes as input sets of visual words, which represent the content of images captured by the robot. When it comes to assessing the similarity of two of these bag-of-words vectors, a simple approach would be to compute some standard distance metric between them such as L2 distance or cosine distance. The commonly used tf-idf ranking measure essentially takes this approach, with some weighting applied to the dimensions. However, these simple metrics assume independence between visual words, which is far from the case. Real-world objects such as cars and pedestrians typically give rise to many visual word detections. Thus two images of a location which differ by the presence of some such object can lie far apart in visual word space, even though they are semantically close. By capturing some of the correlations between sets of visual words using the Chow Liu tree, we will arrive

at a similarity function that is sensitive to the fact that the real world is composed of objects rather than independent visual elements. The learned similarity function is thus in some sense closer to the underlying semantic distance between images. Typical correlations learned by the model are shown in Figure 3.9, and some examples of improved inference performance are shown in Figure 3.18.

3.2 Probabilistic Navigation using Appearance

We now describe the construction of a probabilistic framework for appearance-based navigation. In overview, the world is modeled as a set of discrete locations, each location being described by a distribution over appearance words. Incoming sensory data is converted into a bag-of-words representation; for each location, we can then ask how likely it is that the observation came from that location’s distribution. We also find an expression for the probability that the observation came from a place not in the map. This yields a pdf over location, which we can use to make a data association decision and update our belief about the appearance of each place. Essentially this is a SLAM algorithm in a discrete world. The method is outlined in detail in the following sections.

3.2.1 Representing Appearance

As previously mentioned, raw sensor data received by the system is converted into bag-of-words representation. In this representation, data is described by the presence or absence of “appearance words” chosen from a vocabulary of size $|v|$ (see discussion in Section 2.2). An observation of local scene appearance, visual or otherwise, captured at time k is denoted $Z_k = \{z_1, \dots, z_{|v|}\}$, where z_q is a binary variable indicating the presence (or absence) of the q^{th} word of the vocabulary. Furthermore $\mathcal{Z}^k$ is used to denote the set of all observations up to time k . Note that while we will use the term “bag-of-words” to refer to this representation, strictly this is a “set-of-words” approach which discards count information about the number of instances of an appearance word in an observation. We choose this representation because it decreases the complexity of learning and inference. We will discuss generalizations to the bag-of-words case in Chapter 6.

The results in this thesis employ binary features derived from imagery, based on quantized SURF

descriptors (see Section 3.3); however binary features from any sensor or combination of sensors could be used. For example, laser or sonar data could be represented using quantized versions of the features proposed in [51, 104, 66].

3.2.2 Representing Locations

At time k , our map of the environment is a collection of n_k discrete and disjoint locations $\mathcal{L}^k = \{L_1, \dots, L_{n_k}\}$. Each of these locations has an associated appearance model, which we parameterize in terms of unobservable “scene elements”, e_q . A detector yields visual word observations z_q , which are noisy measurements of the existence of the underlying scene element e_q . The appearance model of a location in the map is our belief about the existence of each scene element at that location:

$$L_i : \{p(e_1 = 1|L_i), \dots, p(e_{|v|} = 1|L_i)\} \quad (3.5)$$

where each of the scene elements e_q are generated independently by the location. A detector model relates scene elements e_q to feature detection z_q . The detector is specified by

$$\mathcal{D} : \begin{cases} p(z_q = 1|e_q = 0), & \text{false positive probability.} \\ p(z_q = 0|e_q = 1), & \text{false negative probability.} \end{cases} \quad (3.6)$$

The generative model, which we will refer to as the FAB-MAP model, is illustrated in Figure 3.3.

The reason for introducing the hidden scene element variables e_q is twofold. Firstly, it provides a natural framework for incorporating data from multiple sensors with different (and possibly time-varying) error characteristics. Secondly, as outlined in the next section, it facilitates factoring the distribution $p(Z|L_i)$ into two parts — a simple model for each location composed of independent variables e_q , and a more complex model that captures the correlations between detections of appearance words $p(z_q|Z_k)$. Because the location model has a simple structure, it can be estimated online from the few observations available for any given location (combined with a prior). The more complex model that captures the correlations between observations is learned offline from abundant training data, and can be combined with the simple location models on the assumption that the conditional

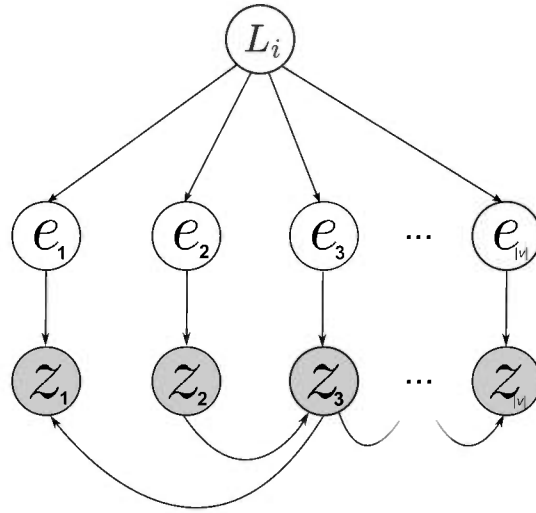


Figure 3.3: The FAB-MAP generative model. Observed variables are shaded, latent variables unshaded. The environment generates locations L_i . Locations independently generate scene elements e_q . Scene elements generate word observations z_q , which have tree-structured dependencies between them. Some additional independence assumptions are noted in Section 3.2.5.

dependencies between appearance words are independent of location. We describe how this is achieved in the following section. We will discuss alternative generative models in Section 3.2.6 where their relative merits may be more easily understood.

3.2.3 Inference in the FAB-MAP Model

We now describe how localization and mapping is achieved using our model. The algorithm we present is a recursive Bayes filter. We must begin our explanation somewhere within the recursive loop – so assume for the moment that a partial map of the environment has already been created. The robot receives a new observation, and we wish to calculate the probability of being at each of the locations in the map, given all the observations collected so far. Calculating this probability, $p(L_i|\mathcal{Z}^k)$ for each location L_i , can be formulated as a recursive Bayes estimation problem:

$$p(L_i|\mathcal{Z}^k) = \frac{p(Z_k|L_i, \mathcal{Z}^{k-1})p(L_i|\mathcal{Z}^{k-1})}{p(Z_k|\mathcal{Z}^{k-1})} \quad (3.7)$$

Here $p(L_i|\mathcal{Z}^{k-1})$ is our prior belief about our location, $p(Z_k|L_i, \mathcal{Z}^{k-1})$ is the observation likelihood, and $p(Z_k|\mathcal{Z}^{k-1})$ is a normalizing term. We discuss the evaluation of each of these terms below, with reference to the graphical model in Figure 3.3. We will also make some additional independence assumptions which are introduced in the course of the derivation. These additional assumptions are summarized in Section 3.2.5. Overview pseudocode is given in Algorithms 3.1 and 3.2.

3.2.4 Evaluating The Model

Observation Likelihood

To simplify evaluation of the observation likelihood $p(Z_k|L_i, \mathcal{Z}^{k-1})$, we first assume independence between the current and past observations, conditioned on the location. The simplified observation likelihood $p(Z_k|L_i)$ can then be expanded as:

$$p(Z_k|L_i) = p(z_n|z_1, z_2, \dots, z_{n-1}, L_i)p(z_{n-1}|z_1, z_2, \dots, z_{n-2}, L_i) \dots p(z_2|z_1, L_i)p(z_1|L_i) \quad (3.8)$$

or the equivalent expansion for any other variable ordering. This expression cannot be evaluated directly because of the intractability of learning the high-order conditional dependencies between appearance words. The simplest approximation is to make a Naive Bayes assumption, yielding:

$$p(Z_k|L_i) \approx \prod_{q=1}^{|v|} p(z_q|L_i) \quad (3.9)$$

Referring to the graphical model in Figure 3.3, each factor can be further expanded as:

$$p(z_q|L_i) = \sum_{s \in \{0,1\}} p(z_q|e_q = s, L_i)p(e_q = s|L_i) \quad (3.10)$$

Now, making the assumption that detector errors are independent of position in the world (i.e. $p(z_q|e_q, L_i) = p(z_q|e_q)$), this becomes:

$$p(z_q|L_i) = \sum_{s \in \{0,1\}} p(z_q|e_q = s)p(e_q = s|L_i) \quad (3.11)$$

which can now be evaluated directly — $p(z_q|e_q = s)$ are the components of the detector model specified in Equation 3.6 and $p(e_q = s|L_i)$ is a component of the place appearance model.

While a Naive Bayes approach yields reasonable results (see Section 3.3), substantially better results can be obtained by instead employing a Chow-Liu approximation to capture more of the dependencies between appearance words. Using a Chow Liu assumption, the appearance likelihood of Equation 3.8 can be approximated as

$$p(Z_k|L_i) \approx p(z_r|L_i) \prod_{q=2}^{|v|} p(z_q|z_{p_q}, L_i) \quad (3.12)$$

where z_r is the root of the tree and z_{p_q} is the parent of z_q in the tree. Contrasting Equation 3.12 with Equation 3.9, note that the Chow Liu model takes account of some of the dependencies between word observations z_q . As before, the observation factors in Equation 3.12 can be further expanded as:

$$p(z_q | z_{p_q}, L_i) = \sum_{s_{e_q} \in \{0,1\}} p(z_q | e_q = s_{e_q}, z_{p_q}, L_i) p(e_q = s_{e_q} | z_{p_q}, L_i) \quad (3.13)$$

Again making the assumption that detector errors are independent of position in the world (i.e. $p(z_q|e_q, L_i) = p(z_q|e_q)$), the expression becomes:

$$p(z_q | z_{p_q}, L_i) = \sum_{s_{e_q} \in \{0,1\}} p(z_q | e_q = s_{e_q}, z_{p_q}) p(e_q = s_{e_q} | z_{p_q}, L_i) \quad (3.14)$$

We now make the approximation that $p(e_q = s_{e_q} | z_{p_q}, L_i)$ can be replaced by $p(e_q = s_{e_q} | L_i)$ (We make this approximation for tractability. See discussion in Section 3.2.5), yielding:

$$p(z_q | z_{p_q}, L_i) = \sum_{s_{e_q} \in \{0,1\}} p(z_q | e_q = s_{e_q}, z_{p_q}) p(e_q = s_{e_q} | L_i) \quad (3.15)$$

The term $p(z_q | e_q, z_{p_q})$ can be expanded (see Appendix C.1) as:

$$p(z_q = s_{z_q} | e_q = s_{e_q}, z_p = s_{z_p}) = (1 + \frac{\alpha}{\beta})^{-1} \quad (3.16)$$

where $s_{z_q}, s_{e_q}, s_{z_p} \in \{0, 1\}$ and

$$\alpha = p(z_q = s_{z_q})p(z_q = \overline{s_{z_q}} \mid e_q = s_{e_q})p(z_q = \overline{s_{z_q}} \mid z_p = s_{z_p}) \quad (3.17)$$

$$\beta = \underbrace{p(z_q = \overline{s_{z_q}})}_{\text{prior}} \underbrace{p(z_q = s_{z_q} \mid e_q = s_{e_q})}_{\text{detector model}} \underbrace{p(z_q = s_{z_q} \mid z_p = s_{z_p})}_{\text{conditional from training}} \quad (3.18)$$

where $\overline{s_z}$ denotes the opposite state to s_z . The terms α and β are now expressed entirely in terms of quantities which can be estimated from training data (as indicated by the under-braces). Hence $p(Z_k \mid L_i)$ can now be computed directly.

New Place or Old Place?

We now turn our attention to the denominator of Equation 2.1, $p(Z_k \mid \mathcal{Z}^{k-1})$. Knowing $p(Z_k \mid L)$ as calculated in the previous section identifies the most likely location in the map to have generated the current observation. However, we would like to be able to deal with the possibility that the observation comes from a new location, not currently in the map. This requires the knowledge of the partition function $p(Z_k \mid \mathcal{Z}^{k-1})$, which converts the appearance likelihood into a probability of loop closure. Computation of the partition function is the key difference between a localization system and a true appearance-only SLAM system. It can be observed that $p(Z_k \mid \mathcal{Z}^{k-1})$ is a measure of the distinctiveness of an observation, and thus directly related to the problem of perceptual aliasing.

To calculate $p(Z_k \mid \mathcal{Z}^{k-1})$, we divide the world into the set of places in our current map, $\mathcal{L}^k$, and the set of unmapped places $\overline{\mathcal{L}^k}$, so that

$$p(Z_k \mid \mathcal{Z}^{k-1}) = \sum_{m \in \mathcal{L}^k} p(Z_k \mid L_m)p(L_m \mid \mathcal{Z}^{k-1}) + \sum_{u \in \overline{\mathcal{L}^k}} p(Z_k \mid L_u)p(L_u \mid \mathcal{Z}^{k-1}) \quad (3.19)$$

where we have applied our assumption that observations are conditionally independent given location. The second summation cannot be evaluated directly because it involves all possible unknown locations. We have compared two different approximations to this term. The first is a mean field approximation [67] :

$$p(Z_k|\mathcal{Z}^{k-1}) \approx \sum_{m \in \mathcal{L}^k} p(Z_k|L_m)p(L_m|\mathcal{Z}^{k-1}) + p(Z_k|L_{avg}) \sum_{u \in \overline{\mathcal{L}^k}} p(L_u|\mathcal{Z}^{k-1}) \quad (3.20)$$

where L_{avg} is the “average place” where the e_q values are set to their marginal probability, and $\sum_{u \in \overline{\mathcal{L}^k}} p(L_u|\mathcal{Z}^{k-1})$ is the prior probability that we are at a new location, which can be obtained from our previous position estimate and a motion model (discussed in the next section).

An alternative to the mean field approach is to approximate the second summation via sampling. The procedure is to sample location models L_u according to the distribution by which they are generated by the environment, and evaluate $\sum_{u \in \overline{\mathcal{L}^k}} p(Z_k|L_u)p(L_u|\mathcal{Z}^{k-1})$ for the sampled location models. In order to do this, some method of sampling location models L_u is required. Here we make an approximation for ease of implementation — we instead sample an observation Z and use the sampled observation to create a place model. The reason for doing this is that sampling an observation Z is extremely easy — our sampler simply consists of selecting at random from a large collection of observations. For our robot which navigates in outdoor urban environments using imagery, large collections of street-side imagery for sampling are readily available – for example from previous runs of the robot, or from sources such as Google Street View¹. In general this sampling procedure will not create location models according to their true distribution because models may have been created from multiple observations of the location. However, it will be a good approximation when the robot is exploring a new environment, where most location models will have only a single observation associated with them. In practice we have found that it works very well, and it has the advantage of being very easy to implement.

Having sampled a location model, we must evaluate $p(Z_k|L_u)p(L_u|\mathcal{Z}^{k-1})$ for the sample. Calculating $p(Z_k|L_u)$ has already been described. For convenience we assume $p(L_u|\mathcal{Z}^{k-1})$, the prior probability of our sampled place model with respect to our history of observations, to be uniform over the samples. Equation 3.19 thus becomes

$$p(Z_k|\mathcal{Z}^{k-1}) \approx \sum_{m \in \mathcal{L}^k} p(Z_k|L_m)p(L_m|\mathcal{Z}^{k-1}) + p(L_{new}|\mathcal{Z}^{k-1}) \sum_{u=1}^{n_s} \frac{p(Z_k|L_u)}{n_s} \quad (3.21)$$

where n_s is the number of samples used, and $p(L_{new}|\mathcal{Z}^{k-1})$ is our prior probability of being at a new

¹<http://www.google.com/help/maps/streetview/>

place.

The approximation could be improved further by using a non-uniform prior $p(L_u|\mathcal{Z}^{k-1})$ over the samples. For example, if robot has recently seen a sequence of images from an urban area, we would expect the next observation to be similar, rather than of a forest, say. This kind of conditional sampling can easily be achieved by maintaining a “location estimate” over the set of images used for Monte Carlo sampling in the same way that we maintain a location estimate over the map, provided that the sampling set consists of sequentially collected images (i.e. that samples u and $u + 1$ were collected at physically close locations). In practice we have found it more convenient to use random collections of images for the sampling set, so have not taken advantage of this possibility.

To conclude, let us try to build some intuition about how the calculation of the partition function relates to perceptual aliasing. Consider what happens after the system has collected a new observation which is a good match for one of the locations in the map. To evaluate $p(Z_k|\mathcal{Z}^{k-1})$ we compute the likelihood that the observation came from randomly selected locations in the sampling set. If the similarity between the current observation and the location in the map is caused by some repetitive feature such as brickwork or foliage, it is likely that we will find an equally good match for our observation among the randomly sampled locations. This means that $p(Z_k|\mathcal{Z}^{k-1})$ will be comparable in magnitude to $p(Z_k|L_i)$ (the likelihood of the matched location), and so the probability assigned to loop closure will be low. On the other hand, finding a good match in the sampling set is very unlikely in the case of a distinctive loop closure. Thus the partition function serves to suppress spurious loop closures caused by perceptual aliasing. The procedure can break down when the robot first encounters a repetitive feature which is not part of its sampling set (e.g. Figure 3.19), however in the limit of sampling set size it will correctly handle progressively rarer causes of false loop closure.

Location Prior

The last term to discuss is the location prior $p(L_i|\mathcal{Z}^{k-1})$. This prior transmits information such that a sequence of observations can result in a higher confidence loop closure. The most straight forward way to obtain a prior is to transform the previous position estimate via a motion model, and use this as our prior for location at the next time step. For the purpose of calculating a prior we assume that

sequentially collected observations come from adjacent places, so that if the robot is at place i at time t , it is likely to be at one of the places $\{i - 1, i + 1\}$ at time $t + 1$, with equal probability. For places with unknown neighbours (e.g. the next place after the most recently collected observation, where place $i + 1$ may be a new place not already in the map), part of the probability mass is assigned to a “new place” node and the remainder is spread evenly over all places in the map. The split is governed by the probability that a topological link with unknown endpoint leads to a new place, which is a user-defined parameter in our system. Clearly this prior could be improved via a better topology estimate or through the use of some odometry information — however, we have found that the effect of the prior is relatively weak, so these issues are not critical to performance.

If the assumption that sequential observations come from adjacent places is not valid, the prior can be simply left uniform. While this results in some increase in the number of false matches, performance is still largely sufficient, especially over smaller trajectories (see quantities evaluation in Chapter 5). Alternatively, if we have an estimate for the topology of the environment, but no estimate of our position within this topology, we could obtain a prior as the limit of a random walk on the topology. This can be obtained efficiently as the dominant eigenvector of the normalized link matrix of the topology [117].

Dealing with Outliers

We have found that the performance of the inference procedure outlined above is strongly dependent on an accurate estimation of the denominator term in Equation 2.1, $p(Z_k | \mathcal{Z}^{k-1})$. Ideally we would like to perform the Monte Carlo integration in Equation 3.21 over a large set of place models L_u that fully capture the visual variety of the environment. In practice, we are limited by running time and available data. The consequence is that occasionally we will incorrectly assign two similar images high probability of having come from the same place, when in fact the similarity is due to perceptual aliasing.

An example of this is illustrated in Figure 3.19, where two images from different places, both of which contain iron railings, are incorrectly assigned high probability of having come from the same place. The reason for this mistake is that the railings generate a large number of features, and there were no examples of railings in the sampling set used to calculate $p(Z_k | \mathcal{Z}^{k-1})$, so the system cannot know these sets of features are correlated.

In practice, due to limited data and computation time, there will always be some aspects of repetitive environmental structure that we fail to capture. We thus employ some secondary techniques to make the system more robust to these occasional spurious matches. One of the most effective approaches is to check a putative loop closure for consistent epipolar geometry [57]. However, cases still exist where this technique may fail, particularly due to repetitive man-made objects such as the situation illustrated in Figure 3.19 (b). A complementary approach that we have found to be successful is to apply a smoothing operation to our likelihood estimates:

$$p(Z_k|L_i) \longrightarrow \sigma p(Z_k|L_i) + \frac{(1-\sigma)}{n_k} \quad (3.22)$$

where n_k is the number of places in the map and σ is the smoothing parameter, which for our experiments we set to 0.99.

The effect of this smoothing of the data likelihood values is to prevent the system asserting loop closure with high confidence on the basis of a single similar image pair. Smoothing the likelihood values effectively boosts the importance of the prior term $p(L_i|\mathcal{Z}^{k-1})$, so that high probability of loop closure can now only be achieved by accumulating evidence from a sequence of matched observations. While in principle mistakes due to our approximate calculation of $p(Z_k|\mathcal{Z}^{k-1})$ can still occur, in practice we have found that this modification successfully rejects almost all outliers, while still allowing true loop closures to achieve high probability after a sequence of only two or three corresponding images.

Updating Place Models

We have now outlined how to compute a pdf over location given a new observation. The final issue to address is data association – at some point it is necessary to make a hard decision and associate the observation with a particular location. The rule used to make data association decisions is discussed in the next section. Once the data association decision has been taken, we can update the relevant place appearance model, which consists of the set $\{p(e_1 = 1|L_i), \dots, p(e_{|v|} = 1|L_i)\}$. When a new place is created its appearance model is initialized so that all scene elements exist with some prior probability $p(e_q = 1)$, which we derive from the word frequencies in training data. Note that this prior probability

is different for each word. Given an observation that relates to the place, each component of the appearance model can be updated according to:

$$p(e_q = 1|L_i, \mathcal{Z}^k) = \frac{p(Z_k|e_q = 1, L_i)p(e_q = 1|L_i, \mathcal{Z}^{k-1})}{p(Z_k|L_i)} \quad (3.23)$$

where we have again applied our assumption that observations are conditionally independent given location.

Data Association Rule

To decide data association we apply a fixed decision rule: if the current observation matches a location with probability higher than some user specified threshold (e.g. $p > 0.999$), we associate the observation with that location. If no location (including the “new place”) meets the probability threshold, we default to creating a new location. Note that the threshold used for triggering data association need not be the same as that at which the system outputs a loop closure notification. A decision rule of this form is preferable to doing maximum-likelihood data association because of the asymmetric consequences of incorrect data association decisions. Reminiscent of the situation with metric SLAM algorithms, we find that incorrectly creating a new location has only minor consequences, whereas incorrectly associating an observation with an existing location rapidly leads to failure of the filter.

Our intuition for this phenomenon is as follows. When an incorrect data association decision is made, the location whose appearance model is updated will now have two very different observations associated with it. The appearance model which results from this mixture of visually different observations tends towards the appearance of the “average place”, which is a reasonably good match for many observations. New observations collected by the robot which are not a good match for any location in the sampling set used in Equation 3.21 and have no good match within the map will tend to match best with this average location. The result is that further observations will be incorrectly associated with the location, making its appearance closer to average and reinforcing the effect. A single wrong data association decision can thus rapidly break the filter.

In contrast, the effect of incorrectly creating a new location is relatively minor. The result is that

Algorithm 3.1 Overview pseudocode for FAB-MAP. Supplementary functions are shown in Algorithm 3.2.

```

//This function processes each new observation collected by the vehicle.
// $\mathcal{L}^k$  is the current map, and  $\mathcal{Z}^k$  is the set of observations up to time  $k$ .
//The most recent observation in  $\mathcal{Z}^k$  is  $Z_k$ , and the  $i^{th}$  loca-
tion in the map is  $L_i$ .
Process_Observation( $\mathcal{L}^k, Z_k$ ):
    pdf = Compute_Pdf_Over_Location( $\mathcal{L}^k, Z_k$ )
    Decide_Data_Association(pdf)
     $\mathcal{L}^{k+1}$  = Initialize_New_Place_Or_Update_Existing_Place( $\mathcal{L}^k, Z_k$ )

```

a single location in the world will be represented by two distinct location models in the robot’s map. In principle this could prevent the robot from recognising this location in future because, given an observation from this location, the probability mass could be split evenly between the two location models in the map, so that neither reaches the threshold necessary to declare loop closure. In practice we find that the symmetry is rarely so exact – typically most of the probability mass will be assigned to one or the other of the locations in the map, and loop closure will still be detected. In fact, we find that it is possible to run the system while *never* performing data association, instead creating a new location model for every new observation. While this mode of operation will certainly lead to a gradual degradation in system performance when the robot remains in the same area for a long time, it is not subject to the unpredictable catastrophic failure that results from a bad data association decision. Therefore, in our typical operation of the system, and in the results presented in this thesis, we opt for this mode of operation.

Clearly the approach we have taken here is not ideal (although it is effective). A better approach might be to maintain a pdf over topologies using a particle filter, as described in [129]. In that way, data association decisions could effectively be postponed until more evidence is available to confirm or reject them. Note that this approach is only effective in conjunction with vehicle motion information.

3.2.5 Independence Assumptions, Approximations and Input Parameters

We briefly recap the approximations and user-defined terms in our model. For tractability, we have made a number of independence assumptions:

Algorithm 3.2 Pseudocode for functions involved in processing a new observation.

```

Compute_Pdf_Over_Location( $\mathcal{L}^k, Z_k$ ):
    //First, compute the likelihood that each location in the map
    //generated the current observation.
    for  $L_i$  in  $\mathcal{L}^k$  do:
        likelihood[ $L_i$ ] = (Compute as per Equation 3.12)
        //(for numerical precision we actually work with log-likelihoods)
    //Also compute the likelihood that the observation came from
    //a location not in the current map (" $L_{new}$ ")
    likelihood[ $L_{new}$ ] = (Compute as per Equation 3.19)
    //Mix in the prior
    location_prior = Get_Location_Prior()
    for  $i$  in range(likelihood) do:
        pdf[i] = likelihood[i] * location_prior[i]
    //(Finally normalize and optionally also apply
    //smoothing as described in Equation 3.22)
    return pdf

Get_Location_Prior()
    //Transform the last pdf over location through a motion model.
    //See discussion under "Location Prior" in Section 3.2.4
    //The prior transmits information such that a sequence of
    //observations yields a higher confidence loop closure.
    return Apply_Motion_Model(pdf_over_location_at_last_timestep)

Decide_Data_Association(pdf):
    //Decide if any location meets the data association criteria
    //See discussion under "Data Association Rule" in Section 3.2.4
    //Note that results reported in this thesis use a rule
    //that creates a new place for each new observation.

Initialize_New_Place_Or_Update_Existing_Place( $\mathcal{L}^k, Z_k$ ):
    //Given the data association decision, update the map.
    if (data association decision for  $Z_k$  is to a location already in the map)
        //Update the relevant location model ( $L_{da}$ )
        //according to Equation 3.23
         $L_{da}$  = Update_Place_Model( $L_{da}, Z_k$ )
         $\mathcal{L}^{k+1} = \mathcal{L}^k$ 
        //No new locations were added to the map,
        //(but one location in the map was updated).
    else
        //Create a new location  $L_{n_k+1}$ 
         $L_{n_k+1}$  = (A new location model, where all
            scene elements exist with prior probability).
        //Update the location model according to Equation 3.23
         $L_{n_k+1}$  = Update_Place_Model( $L_{n_k+1}, Z_k$ )
        //Add the new location to the map
         $\mathcal{L}^{k+1} = \mathcal{L}^k \cup L_{n_k+1}$ 
         $n_k = n_k + 1$ 
    return  $\mathcal{L}^{k+1}$ 

```

1. Sets of observations are conditionally independent given position: $p(Z_k|L_i, \mathcal{Z}^{k-1}) = p(Z_k|L_i)$.
2. Detector behaviour is independent of position: $p(z_q|e_q, L_i) = p(z_q|e_q)$.
3. Location models are generated independently by the environment.

In addition to these independence assumptions, we also make an approximation for tractability:

1. The existence of scene element e_q is not informed by observations other than z_q . That is, an observation of $z_p \forall p \neq q$ provides no information about the existence of e_q , so that $p(e_q|Z_k) \approx p(e_q|z_{q_k})$. We also make the related approximation that $p(e_q | z_p, L_i)$ can be approximated by $p(e_q | L_i)$, when $p \neq q$. While strictly this does discard some information that could be recovered at the cost of more complex inference, in effect these approximations only mean that we will be slightly less certain about the existence of scene elements than if we made full use of the available data.

Input Parameters

The only user-specified inputs to the algorithm are the detector model, $p(z_q = 1|e_q = 0)$ and $p(z_q = 0|e_q = 1)$ (two scalars), the smoothing parameter σ and a term that sets the prior probability that a topological link with an unknown endpoint leads to a new place. Of these, the algorithm is particularly sensitive only to the detector model, which can be determined from a calibration discussed in Section 3.3.3.

3.2.6 Discussion: Alternative Generative Models

Finally, before evaluating the performance of the system, we return briefly to our choice of generative model outlined in Section 3.2.2. Several alternative generative models can be imagined, some of which are more intuitive, and it is useful to discuss the reasons for our model choice.

Model A1

Perhaps the most intuitive generative model, illustrated in Figure 3.4 and which we refer to as “Model A1”, eliminates the hidden layer of e variables and maintains $p(Z|L_i)$ directly. Likelihood evaluations

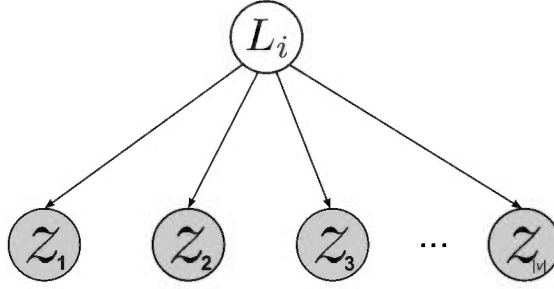


Figure 3.4: Model A1.

in the model would be of the form

$$p(Z_k|L_i) = \prod_{q=1}^{|v|} p(z_q|L_i)$$

In this model, which assumes independence between each appearance word, $p(z_q|L_i)$ could be maintained as a beta distribution initialized to some prior and whose parameters could be updated each time a new observation is associated with the place. The difficulty with this approach is that it implicitly assumes that $p(z_q|L_i)$ is a fixed process, and so does not provide a principled way to deal with observations of varying quality. By contrast the FAB-MAP model, by separating existence of underlying scene elements, e_q , from observations, z_q , can naturally deal with variable confidence observations by adjusting the detector terms in Equation 3.6. This provides a framework for dealing with sensor performance that varies with time, for example due to adverse lighting conditions or environmental effects such as fog, or for combining observations from sensors of different quality, for example cameras with different resolution or noise characteristics. While this thesis will not exploit this flexibility, it is nevertheless a more natural way to parameterize the generative model. Additionally, introducing the layer of e variables is crucial for efficient treatment of correlations between appearance words, as we will see in the discussion of Model A2.

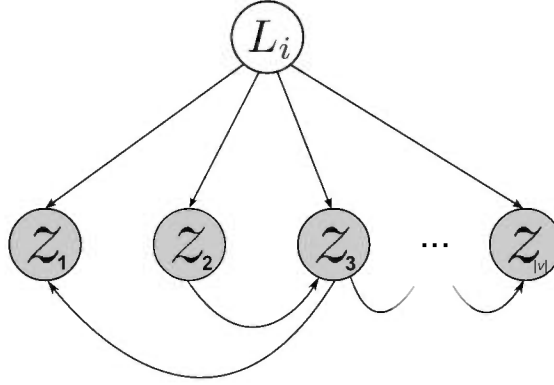


Figure 3.5: Model A2.

Model A2

Model A2 extends Model A1 by introducing tree-structured dependencies between the observed variables z . As in A1, the scheme maintains an estimate of $p(Z|L_i)$ directly for each location. Likelihood evaluations in the model would be of the form

$$p(Z_k|L_i) = p(z_r|L_i) \prod_{q=2}^{|v|} p(z_q|z_{p_q}, L_i)$$

where $p(z_q|z_{p_q}, L_i)$ would be explicitly maintained for each location. This is the TAN model discussed in [50].

The difficulty with this model is twofold. Firstly, due to the elimination of the e variables, it suffers from the same lack of flexibility as Model A1 in dealing with variable confidence observations. Secondly, and more pertinently, because in this model $p(z_q|z_{p_q}, L_i)$ cannot be factored but rather is required directly (cf. Equation 3.13), a separate Chow Liu tree must be constructed for each location in the map². This is in contrast to the FAB-MAP model where only a single global Chow Liu tree is required.

Learning separate Chow Liu trees for each location is not a desirable approach for a number of reasons. The core problem is that most locations in the map have very few associated observations (typically only one). Thus we could not hope to learn a separate Chow Liu model for each location

²It should be noted that the trees could differ in their parameters only, rather than in their structure.

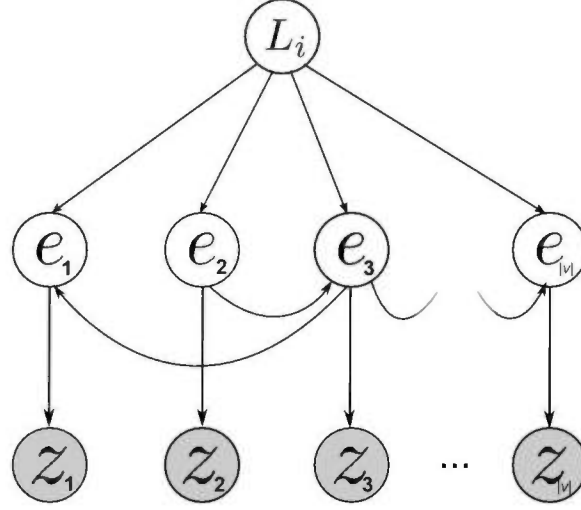


Figure 3.6: Model B.

from such limited training data, at least not by using the ML learning algorithm of [24]. One possibility would be to use the MAP Chow Liu learning technique of [94, 93], which makes use of a prior over Chow Liu trees. Interestingly, though the space of Chow Liu trees is super-exponential (n^{n-2}), a decomposable prior can be specified in only $O(n^2)$ space. The prior would be learned offline in much the same way as in the FAB-MAP model, and used to initialize each new location. The difficulty with this scheme is practical — maintaining the $O(n^2)$ prior for large numbers of variables requires enormous storage, whereas the FAB-MAP model makes use only of the final Chow Liu tree which requires only $O(n)$ space. Considering for example the 100,000 word visual vocabulary used in Chapter 5, maintaining the prior over Chow Liu trees would require approximately 80 GB of storage, whereas the Chow Liu tree itself requires only about 800 KB. For this reason, and because it also eliminates the flexibility of the detector, we discard model A2.

Model B

Finally we consider Model B, which has the same form as the FAB-MAP model, but imposes the tree-structured dependencies between the hidden variables e rather than the observed variables z . In some ways this is a more intuitive model than FAB-MAP, because the dependencies are imposed

between word existence rather than word observation. However, it shares with Model A2 the problem that the Chow Liu dependencies between variables may be different for each location, and so are much more expensive to represent. Additionally, dependencies between hidden variables e are less straight-forward to learn than those between observed variables z .

Thus we arrive back at the FAB-MAP model shown in Figure 3.3. The hidden variables e provide a natural way to deal with variable confidence observations, and also allow the likelihood $P(Z_k|L_i)$ to be factored so that it depends on a simple model for the locations (captured by $p(e_q|L_i)$) and a more complex but compactly representable model of the co-occurrence of visual words, $p(z_q|z_{p_q})$, which applies globally to all locations.

3.3 Implementing the Model

This section discusses some implementation details of the system: how images are converted to bag-of-words representation, how the Chow Liu tree is learned, and how the free parameters of the model are chosen.

3.3.1 Converting Images to Bag-of-Words Representation

Data must be supplied to the system in the bag-of-words representation. There are many possible ways to convert input sensory data to a bag-of-words. As described in Section 2.2.4, our system uses the SURF detector/descriptor [7] to extract regions of interest from the images, and compute 128D non-rotation-invariant descriptors for these regions. These descriptors are quantized against a visual vocabulary using a randomized forest of eight kd-trees [142, 105]. The visual vocabulary is learned by clustering all the descriptors from a set of training images. Note that these training images are generic, and do not come from the area where robot navigation is tested. However, they are similar in character to images for the navigation task – when navigating outdoors, we train the vocabulary on outdoor imagery, etc. In Section 3.4.3 we will discuss the consequences of learning the visual vocabulary on training data which is visually dissimilar to the test data.

The results presented in this chapter employ a fixed-radius incremental cluster algorithm where

the data points are inspected sequentially and a new cluster centre is initialized for every data point that lies further than a fixed threshold from all existing clusters. The threshold is chosen so as to give a visual vocabulary with approximately the desired number of words. In Chapter 5 we will examine the performance impact of different choices of clustering algorithm, including k-means and a hybrid of k-means and the radius-based approach. We find that these alternatives have a non-negligible, but relatively modest, impact on performance. Their effect is less pronounced at high levels of precision, which is the regime that will concern us most.

3.3.2 Learning the Vocabulary Model

Given a visual vocabulary, the next step is to construct a Chow Liu tree to capture the co-occurrence statistics of the visual words. To do this, we construct the mutual information graph as described in Section 3.1.1. Each node in the graph corresponds to a visual word, and the edge weights (mutual information) between node i and j are calculated as per Equation 3.2. The Chow Liu tree is then the maximum weight spanning tree of the graph. We define co-occurrence to mean that the two visual words are present in the same image.

If the Chow Liu tree is to be a good approximation to the true distribution, it must be computed from a large number of observations. To prevent bias, these observations should be independent samples from the observation distribution. We collected 2,800 images from 28km of urban streets using the robot's camera. The images were taken 10m apart, perpendicular to the robot's motion, so that they are non-overlapping and approximate independent samples from the observation distribution. From this data set we compute the clustering of SURF features that form our vocabulary, then compute the Chow Liu tree for the vocabulary. The clustering procedure generated a vocabulary of approximately 11,000 words. The combined process takes 2 hours on a 3GHz Pentium IV. This is a one-off computation that occurs offline.

Illustrative results are shown in Figures 3.7, 3.8 and 3.9, which show sample visual words learned by the system and a visualization of a portion of the Chow Liu tree. These words (which correspond to parts of windows) were determined to have high pairwise mutual information, and so are neighbours in

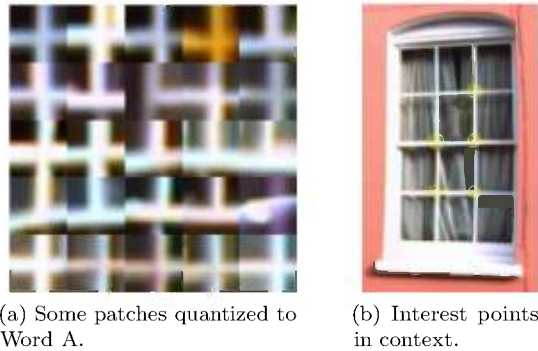


Figure 3.7: A sample word in the vocabulary, showing typical image patches (a) and an example of the interest points in context (b). Interest points quantized to this word typically correspond to the cross-piece of windows. The most correlated word in the vocabulary is shown in Figure 3.8.

the tree. The joint probability of observing the five words shown in the tree is 4,778 times higher under the Chow Liu model than under a Naive Bayes assumption. Effectively the system is learning about the presence of objects that generate the visual words. This significantly improves inference (see Figure 3.18).

3.3.3 Algorithm Configuration

The user defined parameters in our model were set as follows. We modeled our detector by $p(z_q=1|e_q=1) = 0.39$ and $p(z_q=1|e_q=0) = 0$ – i.e. a per-word true positive rate of 39%, and no false positives. To set these parameters, we first assumed a false-positive rate of zero. The false negative rate was then determined by collecting multiple images at a test location – $p(z_q = 0|e_q = 1)$ comes directly from the ratio of the number of words detected in any one image to the number of unique words detected in union of all the images from that location. Finally, we adjusted the value slightly so as to maximize performance on some trial data.

It is worth noting some alternative approaches that could be taken to the calibration of the detector terms. In principle different detector terms could be learned for each word – for example words generated by cars might be less reliably re-observable than words generated by buildings. Exploiting this possibility is discussed in Chapter 6. Additionally, a more precise (but more laborious) detector

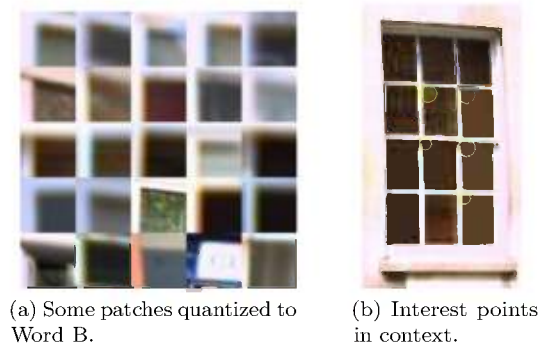


Figure 3.8: A sample word in the vocabulary, showing typical image patches and an example of the interest points in context. Interest points quantized to this word typically correspond to the top-left corner of windows. The most correlated word in the vocabulary is shown in Figure 3.7.

calibration could be achieved by hand-labelling image data and inspecting interest point regions to determine how often the interest point detector fires and how often an interest point is quantized incorrectly.

The number of samples used in the Monte Carlo integration for determining the denominator term $p(Z_k|Z^{k-1})$ was fixed at 2,800 – the set of images used to learn the vocabulary. The smoothing parameter σ was 0.99, and the prior that a new topological link leads to a new place was 0.9. The location prior used a motion model which assumed sequentially collected images, as discussed in Section 3.2. To evaluate the effect of the various other alternative approximations discussed in Section 3.2, the next section presents several sets of results comparing Naive Bayes vs. Chow Liu approximations to $p(Z|L_i)$ and Mean Field vs. Monte Carlo approximations to $p(Z_k|Z^{k-1})$.

3.4 Performance Evaluation

This section presents a performance evaluation of the FAB-MAP model, using imagery collected by a mobile robot. Note that the purpose of this chapter is to evaluate the performance of the FAB-MAP model alone. Therefore, the results presented here do not include a geometric check to verify putative matches, which might mask performance differences between different versions of the model. An evaluation of a more complete system is presented in Chapter 5.

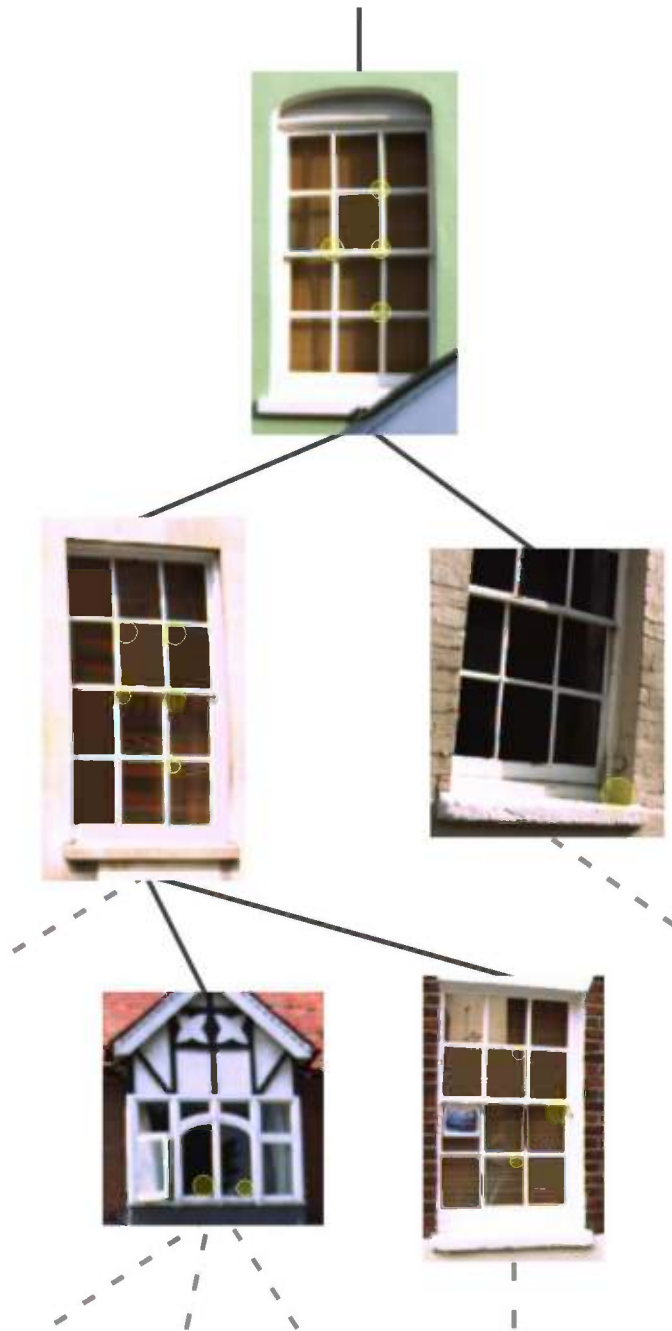


Figure 3.9: Visualization of a section the Chow Liu tree computed for our urban vocabulary. Each word in the tree is represented by a typical example. Clockwise from top, the words correspond to the cross-pieces of window panes, right corners of window stills, top-right corners of window panes, bottom-right corners of window panes and top-left corners of window panes. Under the Chow Liu model the joint probability of observing these words together is 4,778 times higher than under a Naive Bayes assumption.



Figure 3.10: Typical imagery from the environment of the New College data set.



Figure 3.11: Typical imagery from the City Centre data set.

3.4.1 Data Sets

We tested the system on two outdoor urban data sets collected by our mobile robot. Both data sets are available at http://www.robots.ox.ac.uk/~mobile/IJRR_2008_Dataset/. Details of the robot are given in Appendix A. As the robot travels through the environment, it collects images³ to the left and right of its trajectory approximately every 1.5m. The area of our test data sets did not overlap with the region where the training data was collected. The first data set – New College – was chosen to test the system’s robustness to perceptual aliasing. It features several large areas of strong visual repetition, including a medieval cloister with identical repeating archways and a garden area with a long stretch of uniform stone wall and bushes. Typical imagery is shown in Figure 3.10. The second data set – labeled City Centre – was chosen to test matching ability in the presence of scene change. It was collected along public roads near the city centre, and features many dynamic objects such as traffic and pedestrians. Additionally it was collected on a windy day with bright sunshine, which makes

³Image resolution: 640x480.

the abundant foliage and shadow features unstable. Typical imagery is shown in Figure 3.11. Ground truth loop closures were labeled manually for both data sets, on the basis of visual inspection and our knowledge of the robot trajectory.

3.4.2 Performance Metrics

We report precision-recall metrics for the system. Precision is defined as the ratio of true positive loop closure detections to total detections. Recall is the ratio of true positive loop closure detections to the number of ground truth loop closures. Note that images for which no loop closure exists cannot contribute to the true positive rate, however they can generate false positives. Likewise true loop closures which are incorrectly assigned to a “new place” depress recall but do not impact our precision metric. These metrics provide a good indication of how useful the system would be for loop closure detection as part of a metric SLAM system – recall at 100% precision indicates the percentage of loop closures that can be detected without any false positives that would cause filter divergence.

Average precision (AP), or the area under the precision-recall curve, is another commonly quoted performance measure for image retrieval systems. We find AP gives a misleading picture of relative algorithm performance with respect to our application, so do not quote it. AP tends to be heavily impacted by “terminal recall”; that is, the recall achieved when every observation is assigned to some location. A system which achieves a relatively low precision but a high terminal recall can have a high AP score, but would be of limited use for our SLAM application where high precision is critical. Finally, note that a typical loop closure consists of a sequence of several images, so even a recall rate of 20% or 30% is sufficient to detect most loop closure events, provided the detections have uniform spatial distribution.

3.4.3 Results

We now discuss the overall performance of the system. Figures 3.12 and 3.13 show navigation results overlaid on an aerial photo. These results were generated using the Chow Liu and Monte Carlo approximations, which we found to give the best performance. The system correctly identifies a large proportion of possible loop closures with high confidence. There are no false positives that meet the

probability threshold. Videos of these results are available at the same URL as the data set.

Precision recall curves are shown in Figure 3.14. The curves were generated by varying the probability at which a loop closure was accepted. We are primarily interested in the recall rate at 100% precision – if the system were to be used to complement a metric SLAM algorithm, an incorrect loop closure could cause unrecoverable mapping failure. At 100% precision, the system achieves 48% recall on the New College data set, and 37% on the more challenging City Centre data set. As noted previously, because a typical loop closure is composed of multiple images, even a recall rate of 37% is sufficient to detect almost all loop closures.

Some examples of typical image matching results are presented in Figures 3.15 and 3.16. Figure 3.16 highlights robustness to perceptual aliasing. Here very similar images that originate from different locations are correctly assigned low probability of having come from the same place. We emphasize that these results are not outliers; they show typical system performance. The result is possible because most of the probability mass for the query image is captured by locations in the sampling set – effectively the system has learned that images like these are common in the environment. Of course, had these examples been genuine loop closures they might also have been assigned low probability values. We would argue that this is correct behaviour, modulo the fact that the probabilities in 3.16a and 3.16b seem too low. The very low probabilities for these examples are due to the fact that good matches for the query images are found in the sampling set, capturing almost all the probability mass. This is less likely in the case of a true but ambiguous loop closure.

Figure 3.15 shows matching performance in the presence of scene change. Many of these image pairs have far fewer visual words in common than the examples of perceptual aliasing, yet are assigned high probability of having come from the same place. To give a quantitative example, loop closures detected by the system sometimes have as few as 8% of their visual words in common, whereas perceptual aliasing examples often share most of their visual words – for example, the images shown in Figure 3.16b have 46% of their words in common.

Poor probability estimates do occasionally occur – some examples of images incorrectly assigned high probability of a place match are shown in Figure 3.19. Note however that the typical true loop

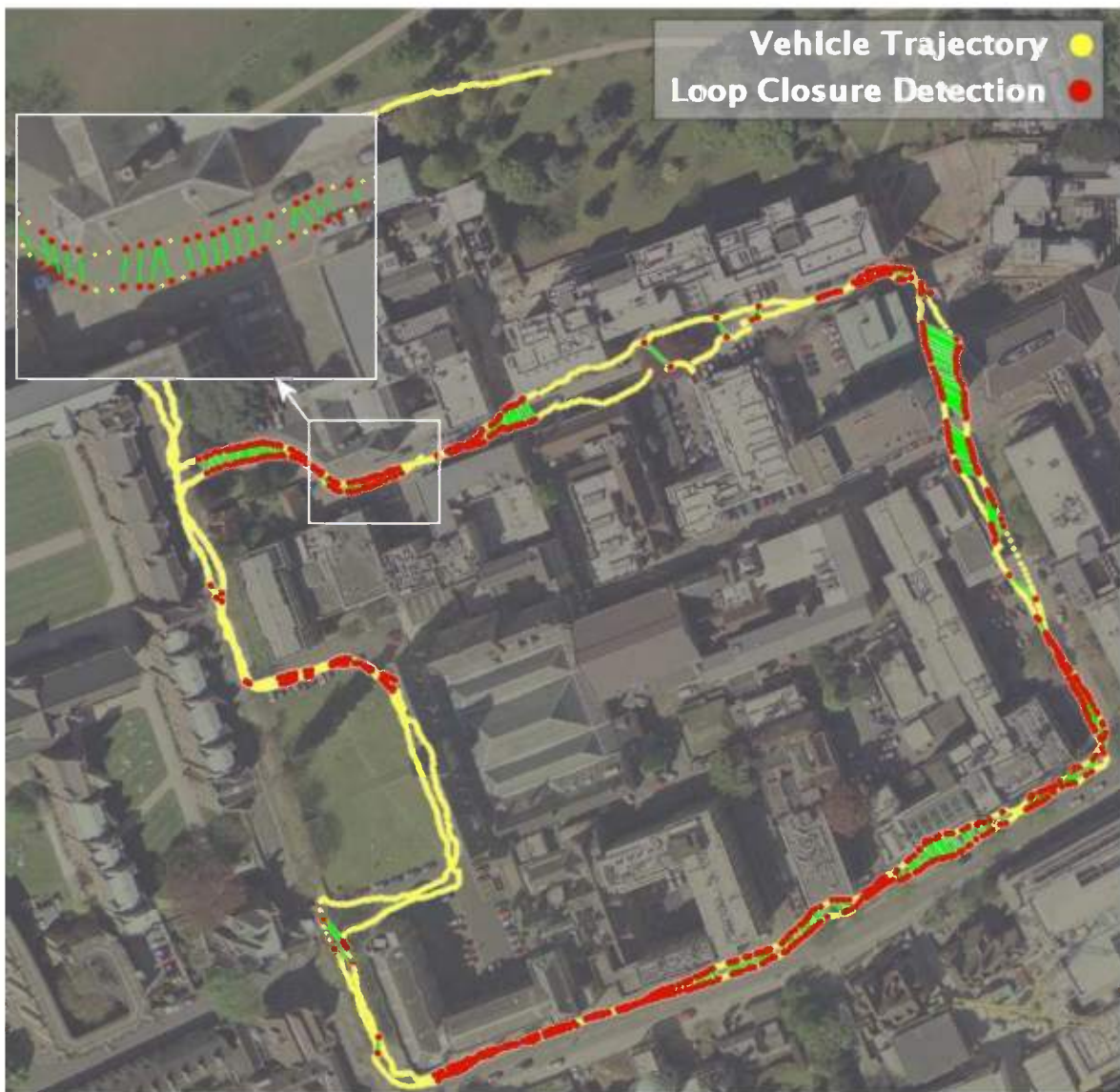


Figure 3.12: Appearance-based matching results for the City Centre data set overlaid on an aerial photograph. The robot travels twice around a loop with total path length 2km, collecting 2,474 images. Positions (from hand-corrected GPS) at which the robot collected an image are marked with a yellow dot. Two images that were assigned a probability $p \geq 0.99$ of having come from the same location (on the basis of appearance alone) are marked in red and joined with a line. There are no incorrect matches that meet this probability threshold. This result is best viewed as a video (available at the same URL as the data set).



Figure 3.13: Appearance-based matching results for the New College data set overlaid on an aerial photograph. The robot traverses a complex trajectory of 1.9km with multiple loop closures. 2,146 images were collected. Positions (from hand-corrected GPS) at which the robot collected an image are marked with a yellow dot. Two images that were assigned a probability $p \geq 0.99$ of having come from the same location (on the basis of appearance alone) are marked in red and joined with a line. There are no incorrect matches that meet this probability threshold. This result is best viewed as a video (available at the same URL as the data set).

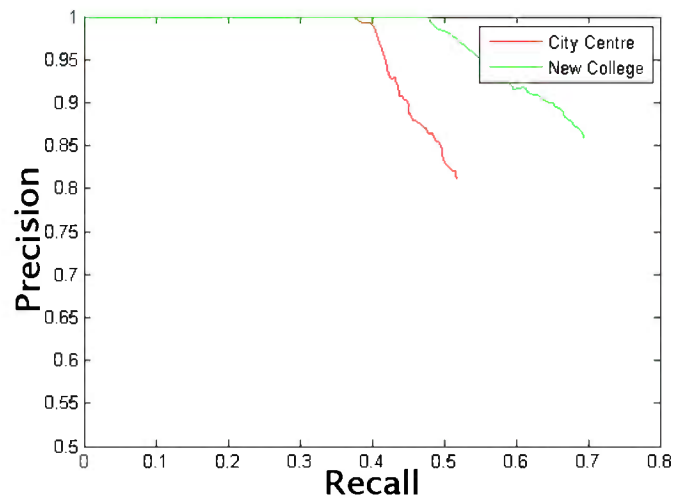


Figure 3.14: Precision-Recall curves for the City Centre and New College data sets. Notice the scale.

closure receives a much higher probability of a match than these examples, primarily due to temporal support from surrounding loop closures. Neither of the image pairs in Figure 3.19 met the loop closure notification threshold of $p > 0.99$.

Comparing Approximations

This section presents a comparison of the alternative approximations discussed in Section 3.2 — Naive Bayes vs. Chow Liu approximations to $p(Z|L_i)$ and Mean Field vs. Monte Carlo approximations to $p(Z_k|\mathcal{Z}^{k-1})$. Figure 3.17 shows precision-recall curves from the New College data set for the four possible combinations. Timing and accuracy results are summarized in Table 3.1. The Chow Liu and Monte Carlo approximations both considerably improve performance. Some examples of typical loop closures detected by the Chow Liu approximation but not by the Naive Bayes are shown in Figure 3.18.

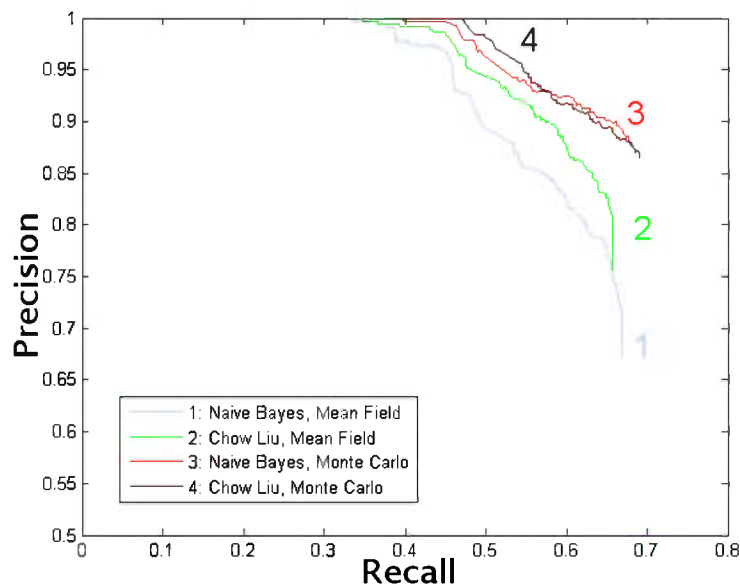


Figure 3.17: Precision-Recall curves for the four variant algorithms on the New College data sets. Notice the scale. All results include the location prior which allows evidence to accumulate through a sequence of observations. Relative performance on the City Centre data set is comparable.

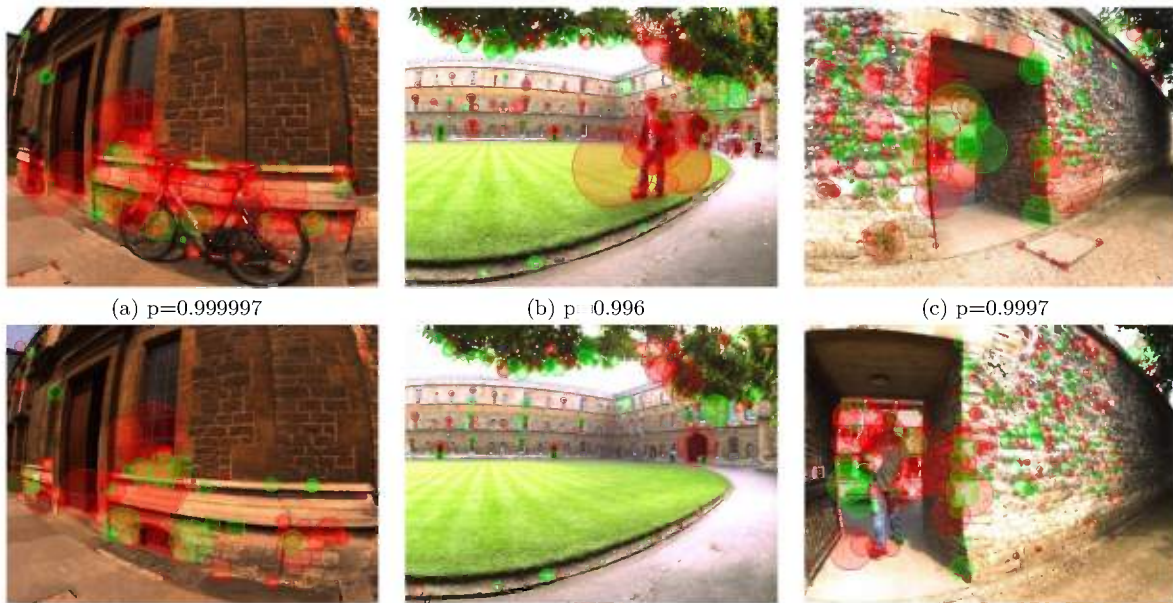


Figure 3.15: Some images assigned high probability of having come from the same place, despite scene change. Words common to both images are shown in green, others in red. Loop closure probability indicated between the pairs.

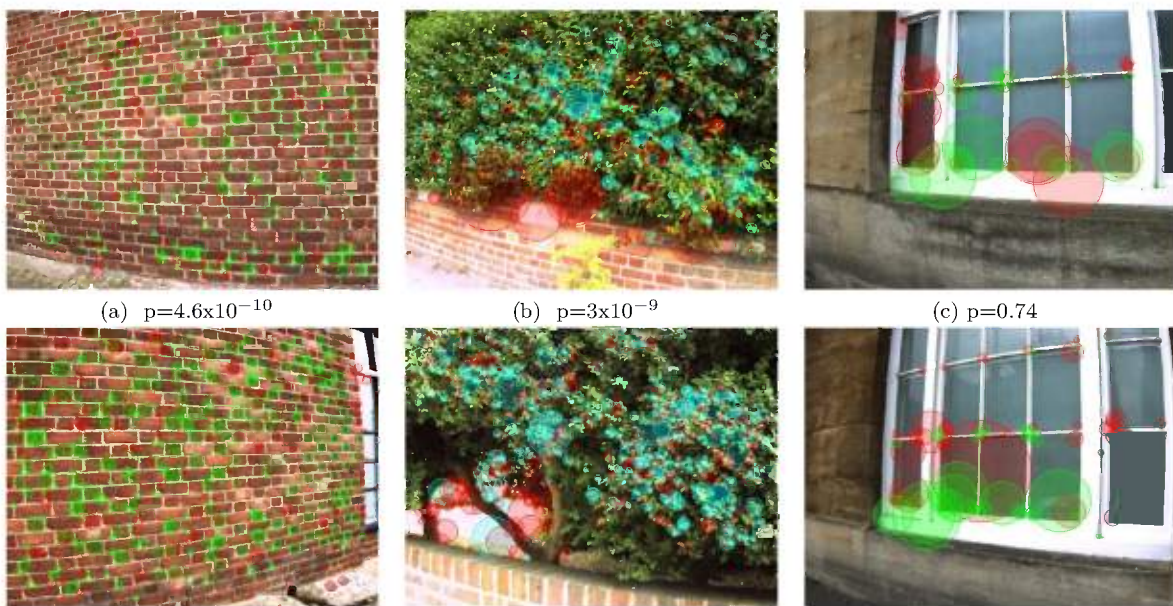


Figure 3.16: Some similar-looking images from different parts of the workspace correctly assigned low probability of having come from the same place. Words common to both images are shown in green (in blue for (b)), others in red.

Algorithm	Recall - New College	Recall - City Centre	Run Time
Mean Field, Naive Bayes	33%	16%	0.6ms/place
Mean Field, Chow Liu	35%	31%	1.1ms/place
Monte Carlo, Naive Bayes	40%	31%	0.6ms/place + 1.71 secs sampling
Monte Carlo, Chow Liu	47%	37%	1.1ms/place + 3.15 secs sampling

Table 3.1: Comparison of the four different approximations. The recall rates quoted are at 100% precision. The time to process a new observation is given as a function of the number of places already in the map, plus a fixed cost to perform the sampling. Timing results are for a 3GHZ Pentium IV.

The extra performance comes at the cost of some increase in computation time; however even the slowest version using Chow Liu and Monte Carlo approximations is still relatively fast. Running times are summarized in Table 3.1. The maximum time taken to process a new observation over all data sets was 5.9 seconds. As the robot collected images approximately every two seconds, this is somewhat slower than required. In Chapters 4 and 5 we will discuss approximate inference techniques that will enable real time performance with minimal accuracy penalty. Additionally, we note that the dominant computational cost is the calculation of $p(Z|L_i)$ for each place model in the map and each sampled place. Each of these calculations is independent; the algorithm is highly parallelizable and will perform well on multi-core processors.

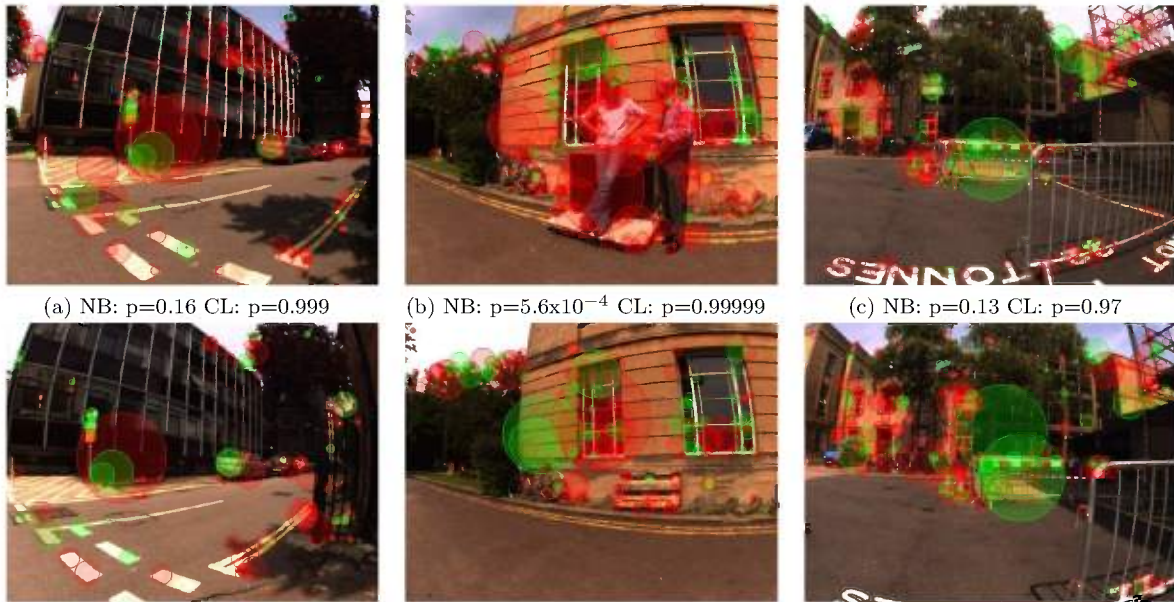


Figure 3.18: Some examples of situations where the Chow Liu approximation outperforms Naive Bayes. In (a), a change in lighting means that the feature detector does not fire on the windows of the building. In (b), the people are no longer present. In (c), the foreground text and the scaffolding in the top right are not present in the second image. In each of these cases, the missing features are known to be correlated by the Chow Liu approximation, hence the more accurate probability. Words common to both images are shown in green, others in red. The probability that the two images come from the same place (according to both the Chow Liu and Naive Bayes models) is indicated between the pairs.



Figure 3.19: Some images from different locations incorrectly assigned high probability of having come from the same place. In (a), the training set contains no examples of railings, so the matched features are not known to be correlated. In (b), we encounter the same truck again in a different part of the workspace. Errors of this type are particularly challenging. Notice that while both images are assigned high probability of a match, a typical true loop closure is assigned much higher probability. Neither of these image pairs met our $p = 0.99$ data association threshold.



Figure 3.20: Some data sets on which the system has been tested.

3.4.4 Generalization Performance

Because FAB-MAP relies on a learned appearance model to reject perceptual aliasing and improve inference, a natural question is to what extent system performance will degrade in environments very dissimilar to the training set. We have not fully investigated this issue because we have only a limited number of data sets for which labeled ground truth was produced. However, we have informally run the algorithm on a much larger number of data sets and our experience suggests that performance degrades gracefully. Recall typically drops somewhat, but the system almost always detects a useful fraction of the loop closures, with no or very few false positives.

A selection of the different types of data used in testing is shown in Figure 3.20. Note that we processed all of these data sets using the same visual vocabulary used for the New College and City Centre experiments. The data sets shown are:

1. A handheld video sequence collected in an indoor environment using a consumer camcorder (Figure 3.20a). Notable for motion blur.
2. Data from the “Wildcat” autonomous vehicle provided by BAE Systems, Filton (Figure 3.20b). Notable for severe interlacing artifacts.
3. A sequence collected with a webcam, for a monocular SLAM experiment (Figure 3.20c). Data is from [164].
4. Frames extracted directly from an internet-quality video (Figure 3.20d). This is the results video from [103].
5. An image sequence from the NASA Jet Propulsion Laboratory Mars Yard test facility (Figure 3.20e).
6. The DLR Spatial Cognition data set [75] from the Inside Data Association workshop at Robotics: Science and Systems 2008 (Figure 3.20f).

To provide some quantitative measure of system performance on data dissimilar to the training set, we manually labeled ground truth for the indoor image sequence shown in Figure 3.20a, and processed it using the same vocabulary and Chow Liu tree used for the New College and City Centre experiments. Videos of the indoor results can be found at http://www.robots.ox.ac.uk/~mobile/IJRR_2008_Dataset/. While the indoor data set is too small to draw strong conclusions from (only 188 images), results are nevertheless interesting. Firstly, the quality of the imagery in this indoor data set is considerably poorer than that from the robot, due to adverse lighting conditions and greater motion blur. Despite this, system performance is largely similar to our outdoor results (46% recall at 100% precision), although it should be noted that the numbers are not strictly comparable because of the different sizes of the two data sets. The fact that the system works indoors despite using an outdoor vocabulary should not be so surprising in light of the discussion in Chapter 2 (Section 2.6). To investigate how large an effect the vocabulary has on performance, we learned an indoor-specific vocabulary from a set of 1,679 images collected with the same hand-held video camera as used for the data set. The area where these images were collected did not overlap with the area used for the navigation experiment. As

expected, performance improves with the more appropriate vocabulary, reaching 59% recall at 100% precision.

One effect that we did not expect is that when using the outdoor vocabulary on the indoor data set, the Chow Liu model continues to outperform Naive Bayes (46% recall as opposed to 39%). This is surprising, and suggests perhaps that some of the correlations being learned by the Chow Liu tree model generic low-level aspects of imagery. We should emphasise however that the data set is not large enough to fully support such a conclusion.

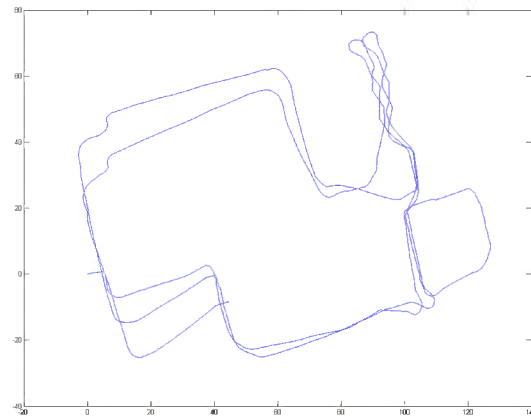
Overall, the quantitative results on this indoor sequence are in line with our experience on a large number of data sets – system performance degrades gracefully in environments dissimilar to the training data.

3.4.5 Use with Metric SLAM Systems

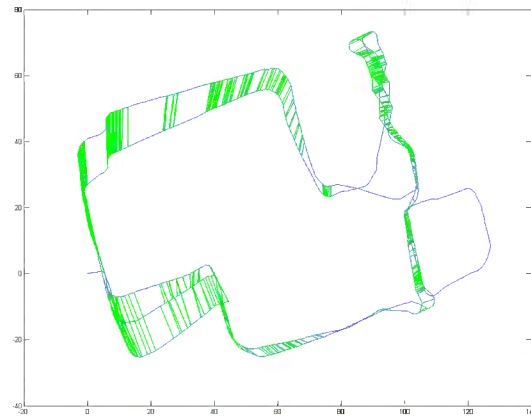
The motivation for the development of FAB-MAP was to solve the loop closure detection and multi-session mapping problems for metric SLAM systems. This section briefly presents some results demonstrating the use of FAB-MAP in that context. Typical results are shown in Figure 3.21. The vehicle trajectory, consisting of a set of vehicle poses and the relative metric transformations between them (with associated uncertainty), is determined by a visual SLAM system developed by Mei *et al.* [91]. Loop closure constraints between poses are determined by FAB-MAP as described in this chapter. These constraints are shown in Figure 3.21b as green links between different parts of the trajectory. The relative metric transformation for these constraints is determined by the visual SLAM system using stereo image pairs collected by the robot at each location. The final trajectory estimate is obtained by relaxing the pose graph using the method described in [110].

An example of multi-session mapping using the same basic approach is shown in Figure 3.22. Here, the alignment between two partially overlapping data sets collected on different days is recovered. This is achieved in exactly the same way as for the single-session mapping example. The only difference is that when the robot begins the second mapping session, we load the place models learned by FAB-MAP during the previous session, so that these places can be recognised.

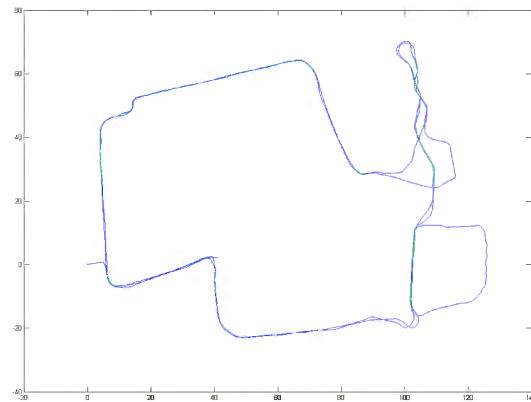
More details of the experiments summarized here may be found in [110]. The data set used is that of [145].



(a) Trajectory from the visual SLAM system. The robot travels twice around the loop. The position estimate is clearly drifting over time.

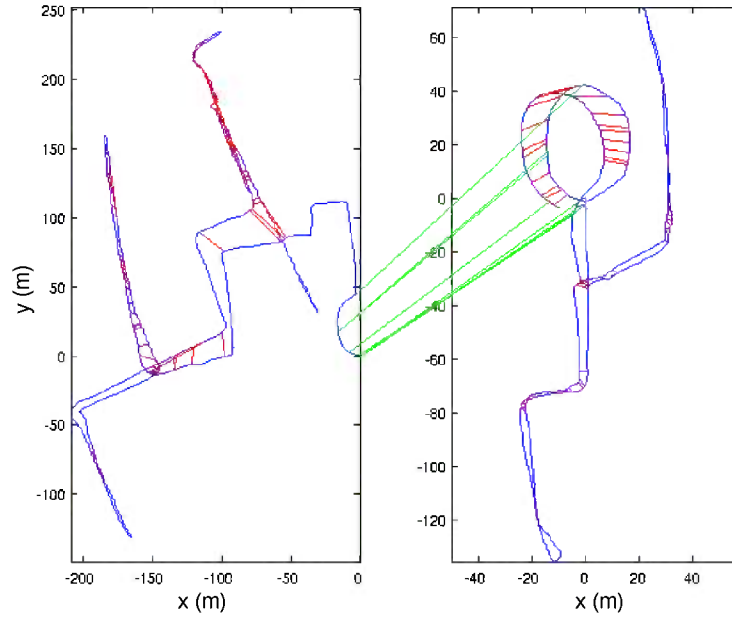


(b) Place recognition constraints from FAB-MAP.

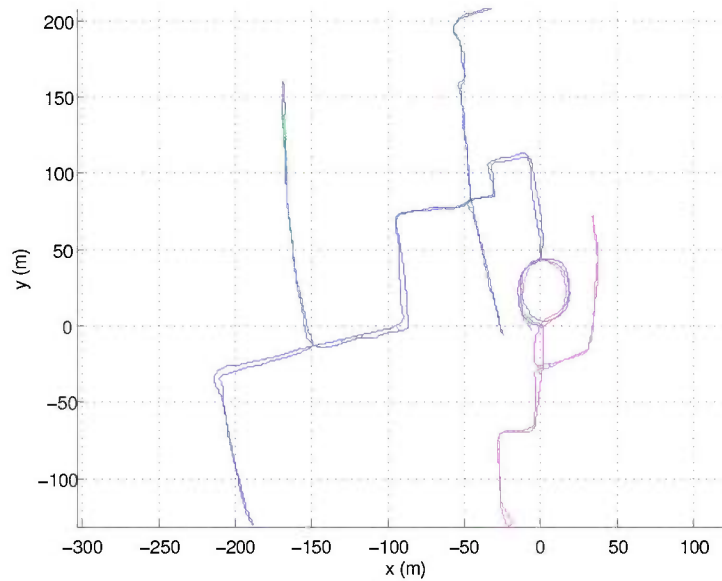


(c) Final trajectory estimate after relaxation.

Figure 3.21: Improving a metric SLAM estimate using FAB-MAP.



(a) Two input maps.



(b) Single output map.

Figure 3.22: Use of FAB-MAP for multi-session mapping. Sub-figure (a) shows two robot trajectories collected on different days (blue), which have an unknown transformation between them. Place recognition constraints between poses in these trajectories are detected by FAB-MAP. Constraints within each trajectory are shown as red links, and between the two trajectories as green links. Pose graph relaxation yields the final output trajectory shown in (b). The two trajectories now share a common coordinate frame.

3.5 Chapter Summary

This chapter has introduced the FAB-MAP algorithm, a probabilistic framework for navigation and mapping which relies on appearance information only. The system is robust even in visually repetitive environments and is fast enough for online loop closure detection in realistic mobile robotics applications. In our evaluation, FAB-MAP was successful in detecting large portions of loop closures in challenging outdoor environments, even when we require a confidence level such that there are no false positives.

Three aspects of our results are particularly noteworthy. Firstly, we outline a principled approach for determining when an observation comes from a location not already in the map. To our knowledge this is the first approach to the problem that goes beyond simple heuristics such as fixed similarity thresholds. The more principled approach gives a large performance improvement. It also naturally handles perceptual aliasing. Our second notable contribution was to show that learning a generative model of the bag-of-words observations yields a marked improvement in performance. We think that this technique will have applications beyond the specific problem addressed in this thesis. Thirdly, we have observed that the system appears to perform well even in environments quite dissimilar to the training data. This suggests that the approach is practical for deployment in unconstrained navigation tasks, as a natural complement to more typical metric SLAM algorithms.

Chapter 4

Approximate Inference using Concentration Inequalities

4.1 Introduction

This chapter is concerned with speed improvements to the appearance-only SLAM system introduced in Chapter 3. We show that by employing a probabilistic bail-out test in the core likelihood calculation required by FAB-MAP, speed improvements of between 25 and 50 times are possible, with only slight accuracy penalty. We achieve this performance improvement by introducing a novel approach to multihypothesis testing based on concentration inequalities, the applicability of which is quite general. Typical filter update times for the improved system are on the order of 150ms for maps which contain several thousand locations. This enables real-time loop closure detection on a mobile robot for loops tens of kilometers in length.

4.2 Bail-Out Strategies

In Chapter 3 we introduced the probabilistic model for our appearance-only SLAM system. The limiting computational cost of the method was the calculation of the observation likelihood term $p(Z_t|L_i, \mathcal{Z}^{t-1})$

for each location in the map and for each sample in the Monte Carlo sampling set. Typically, only a small number of these locations will yield non-negligible probability of having generated the current observation. The core idea of this chapter is that, by identifying locations that will have insignificant likelihood before the calculation is fully complete, many locations can be excluded quickly and large speed increases can be realized.

Closely related ideas have been described elsewhere in computer vision, notably in the context of efficient RANSAC algorithms [113, 89]. Matas and Chum showed that for RANSAC, the sequential probability ratio test (SPRT) yields the optimal solution. The SPRT approach was originally designed for testing two hypotheses under a sequence of identical and equally informative observations [158]. Extensions exist for the multihypothesis case [5]. However, stopping boundaries for the SPRT are not easy to derive when the observations are not equally informative. We describe an alternative approach based on concentration inequalities [16]. Unlike the SPRT, this approach is straight-forward to apply even when there are multiple hypotheses and the observations are not equally informative. We have noted related ideas in other fields [88], however we believe our approach is novel in this context.

We begin by introducing our bail out test in the most general form in Section 4.3, before discussing how to apply it to our appearance-based navigation task in Section 4.4.

4.3 Probabilistic Bail-Out using Bennett's Inequality

Let $\mathcal{H} = \{H^1, \dots, H^K\}$ be a set of K hypotheses (models) and let $Z = \{z_1, \dots, z_N\}$ be an observation consisting of N features. The likelihood of the observation under the k^{th} hypothesis is given by

$$p(Z|H^k) = p(z_1|z_2, \dots, z_N, H^k) \dots p(z_{N-1}|z_N, H^k) p(z_N|H^k) \quad (4.1)$$

Define the log-likelihood of the first i features under the k^{th} hypothesis as

$$D_i^k = \sum_{j=1}^i d_j^k \quad (4.2)$$

where

$$d_i^k = \ln(p(z_i|z_{i+1}, \dots, z_N, H_k)) \quad (4.3)$$

is the log-likelihood of the i^{th} feature under the k^{th} hypothesis. We would like to determine, as rapidly as possible, the hypothesis H^* for which the total log-likelihood D_N^* is maximized. Finding H^* with certainty requires a complete evaluation of the likelihood of each hypothesis, which may be too slow for applications of interest. Consequently, we consider the problem of finding a hypothesis $H^\#$, subject to the constraint that $p(H^\# \neq H^*) < \epsilon$, where ϵ is some user-specified probability.

In overview, our approach is to calculate the likelihoods of all hypotheses in parallel, and terminate the likelihood calculation for hypotheses that have fallen too far behind the current leading hypothesis. “Too far” can be quantified using concentration inequalities, which yield a bound on the probability that a hypothesis will overtake the leader, given their current difference in likelihoods and some statistics about the properties of the features which remain to be evaluated. Figure 4.1 illustrates the idea.

Consider two hypotheses $H^x, H^y \in \mathcal{H}$ and let

$$X_i = d_i^x - d_i^y \quad (4.4)$$

the difference in the log-likelihood of feature i under hypothesis H^x and H^y . X_i can be considered as a random variable before its value has been calculated. This is useful because we can calculate some key statistics about X_i more cheaply than we can determine its exact value. Now define

$$S_n = \sum_{i=n+1}^N X_i \quad (4.5)$$

If after evaluating n features, the log-likelihood of some hypothesis is Δ less than the current best hypothesis, then the probability of failing to locate H^* if we discard this hypothesis is given by $p(S_n > \Delta)$ – i.e. by the probability that the trailing hypothesis will overtake the leader after the evaluation of the remaining features. Thus, knowing the distribution of S_n allows the creation of a probabilistic bail-out test for discarding hypotheses subject to an error constraint. Calculating an explicit distribution on S_n is infeasible, however concentration inequalities – which bound the probability that a function of

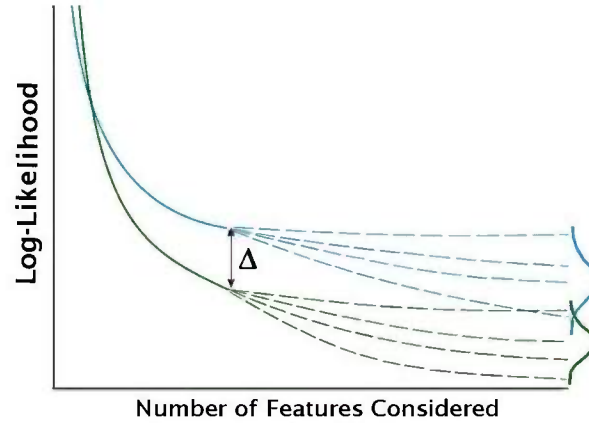


Figure 4.1: Conceptual illustration of the bail-out test. After considering the first i features, the difference in log-likelihoods between two hypotheses is Δ . Given some statistics about the remaining features, it is possible to compute a bound on the probability that the evaluation of the remaining features will cause one hypothesis to overtake the other. If this probability is sufficiently small, the trailing hypothesis can be discarded.

random variables will deviate from its mean value – can be applied to yield bounds on $p(S_n > \Delta)$.

A large variety of concentration inequalities exist, many of which apply under very general conditions, including cases where the component distributions are not identically distributed, not independent, and are combined using arbitrary functions. The more information available about the component distributions X_i , the tighter the bound. For an overview see [16]. For our application we will find it convenient to use the Bennett inequality for sums of symmetric random variables [9]. This inequality is specified in terms of two parameters — M , a bound on the maximum value of any component X_i , and v , a bound on the sum of the variances of the components X_i .

Formally, let $\{X_i\}_{i=n+1}^N$ be a collection of independent mean-zero random variables with symmetric distributions (corresponding to the log-likelihood changes due to those features not yet considered), and satisfying the conditions

$$p(|X_i| < M) = 1, \forall i : \{n+1 < i < N\} \quad (4.6)$$

$$\sum_{i=n+1}^N E[X_i^2] < v \quad (4.7)$$

and as before

$$S_n = \sum_{i=n+1}^N X_i \quad (4.8)$$

then the Bennett inequality states that

$$p(S_n > \Delta) < \exp \left(\frac{v}{M^2} \cosh(f(\Delta)) - 1 - \frac{\Delta M}{v} f(\Delta) \right) \quad (4.9)$$

where

$$f(\Delta) = \sinh^{-1} \left(\frac{\Delta M}{v} \right) \quad (4.10)$$

Thus, after considering the first n features, and given bounds on the maximum value (M) and sum of variances (v) of the inter-hypothesis log-likelihood changes (X_i) for the features yet to be evaluated ($n+1 : N$), we can solve the Bennett inequality for Δ , such that $p(S_n > \Delta)$ meets the user specified error probability ϵ . We solve for Δ using a combination of bisection and Newton-Raphson iteration. After considering a given feature, any hypothesis whose log-likelihood is at least Δ less than the current leading hypothesis can be discarded, because the probability of it overtaking the current leader before the end of the calculation is less than ϵ . Note that the values of M and v are defined on the features yet to be evaluated ($n+1 : N$), thus as the calculation progresses and the number of unconsidered features decreases, M and v will decrease. The bail-out threshold thus converges to zero as the number of unconsidered features decreases (see Figure 4.2).

4.4 Application to Appearance-Only SLAM

We now turn our attention to applying this bail-out condition to our appearance-only SLAM system. Recalling our notation from Chapter 3, we have a set of locations L_i whose likelihood we are evaluating with respect to some observation Z . Each of our features z_i is a binary variable indicating whether or not the i^{th} word of the visual vocabulary was present in the current observation. We will denote the difference in log-likelihood of two locations L_x and L_y under feature z_i by

$$\begin{aligned}
X_i &= d_i^x - d_i^y \\
&= \ln(p(z_i|z_{p_i}, L_x)) - \ln(p(z_i|z_{p_i}, L_y))
\end{aligned} \tag{4.11}$$

which we will consider as a random variable before its value has been calculated. To apply our bail-out test, we must establish that our random variables X_i meet the conditions for applicability of Bennett's inequality. We must also define an order in which to consider the features, and outline how M and v may be calculated.

4.4.1 Applicability of Bennett's Inequality

To make use of Equation 4.9, we require that the random variables $\{X_i\}_{i=1}^N$ be independent, mean-zero and have symmetric distributions. We also require that their maximum values be bounded.

To understand why these conditions are satisfied, we note that, given that the values of z_i and z_{p_i} are fixed, the likelihood of a feature being observed at a particular location, $p(z_i|z_{p_i}, L_x)$, depends only on the number of previous observations of feature z_i at location L_x (see Equation 3.13). Thus, assuming the number of observations relating to any given place is finite, X_i has a multinomial distribution because $p(z_i|z_{p_i}, L_x)$ can be one of a fixed set of discrete values. If we now make the assumption that, a-priori, all locations are equally likely to generate any given feature, X_i must have a symmetric mean-zero distribution. This is because if $X_i = \delta$ for some history of previous observations at L_x and L_y , then $X_i = -\delta$ for the symmetric case where the observation history of L_x and L_y is swapped, which occurs with equal probability.

To illustrate this, consider an example. Let z_i be a feature which is observed with probability η , and let L_x and L_y be two location models which each have exactly one associated observation. Further assume that the state of the feature z_i in the observation associated with L_x and L_y (i.e. $z_i = 0$ or $z_i = 1$) is unknown. For compactness, denote the event that location L_x has a previous associated observation with $z_i = 0$ by $L_x\{0\}$. There are four possible cases:

Case	Probability	Value of X_i
$L_x\{0\}, L_y\{0\}$	$p = (1 - \eta)^2$	0
$L_x\{0\}, L_y\{1\}$	$p = \eta(1 - \eta)$	δ
$L_x\{1\}, L_y\{0\}$	$p = \eta(1 - \eta)$	$-\delta$
$L_x\{1\}, L_y\{1\}$	$p = \eta^2$	0

which occur with the probabilities shown. Also shown is the log-likelihood difference (X_i) in the four cases, where δ is the value of $\ln(p(z_i|z_{p_i}, L_x\{0\})) - \ln(p(z_i|z_{p_i}, L_y\{1\}))$. Observe that X_i has a symmetric, mean-zero multinomial distribution:

$$X_i : \begin{cases} \delta & \text{with probability } \eta(1 - \eta) \\ 0 & \text{with probability } (1 - \eta)^2 + \eta^2 \\ -\delta & \text{with probability } \eta(1 - \eta) \end{cases} \quad (4.12)$$

When the location models have more than one associated observation this multinomial distribution has a similar form, but a larger number of states.

Finally, Bennett's inequality requires that the variables X_i be independent. Our Chow Liu model captures much but not all of the conditional dependence between features. Thus the variables X_i may have weak residual dependence. However, our experimental results (Section 4.5) would appear to indicate that this is not a problem in practice.

4.4.2 Ranking Features

Next, we must define an order in which to consider the features during the bail-out test. While the bail-out test applies to any ordering, it is natural to rank the features by information gain. That way, the hypotheses will converge most rapidly toward their final log-likelihood values and poor hypotheses can be identified earliest (see Figure 4.2).

Under our observation model, each feature z_i is conditionally dependent on one other feature z_{p_i} . If we observe $z_i = s_i$ and $z_{p_i} = s_{p_i}$ (with $s \in \{0, 1\}$), then the information gain associated with this

observation under our model is

$$I = -\ln p(z_i = s_i | z_{p_i} = s_{p_i}) \quad (4.13)$$

Typically observations of rare words are the highest ranked features, though, perhaps surprisingly, failure to observe a word can sometimes also have high information gain – for example, if two words are almost always observed together, then failure to observe one while observing the other is an informative observation.

Note that in our implementation the probabilities in Equation 4.13 come from the training data on which we learnt the model of our visual words. As such we are calculating the information gain with respect to the places in the training data. Strictly we should consider the information gain with respect to the set of places in our current map – for example, some feature might be very rare in the training set but very common in the map. In practice we observe that the difference between the two values is usually small, so maintaining a separate set of probabilities is unnecessary. As the bail-out test applies to any feature ordering, the effect of this simplified implementation is merely that hypotheses may be discarded slightly later than optimal.

4.4.3 Calculating M and v

Finally, to apply Bennett’s inequality, we must calculate M and v , the parameters of the concentration inequality which depend on the component random variables X_i . As per Equation 4.6, M is a bound on the maximum value of $X_i, \forall i : \{n + 1 < i < N\}$ – that is, the maximum of the log-likelihood differences due to feature i between some trailing location hypothesis L_x and the leading hypothesis at the time when feature i is considered, over all indices i between $n + 1$ and N . Calculating an exact value for M would require knowledge of the leading hypothesis at a future point in the calculation, which is unknown. However, as M is a *bound* on the maximum value of X_i , it can be calculated as the maximum inter-hypothesis log-likelihood change over *all* hypothesis pairs, and all remaining features. This bound can easily be calculated by keeping track, for each feature i , of the location that was most and least likely to have generated that feature, as these location pairs maximize X_i .

The value of v can be determined directly as the sum of the variances of the distributions shown

in Equation 4.12. This is practical when the number of observations associated with each location in the map is small. However, as exploration continues and the number of observations associated with a typical place in the map increases, the calculation of this variance rapidly becomes expensive. At some point it may be beneficial to switch from using Bennett’s inequality to Hoeffding’s inequality [61], a similar concentration inequality that requires knowledge only of the maximum value of each X_i . Hoeffding’s inequality gives a weaker bound, but this is compensated for by the fact that by the time the variance becomes expensive to compute, the place models themselves are more differentiated, and so their likelihoods will diverge faster.

4.4.4 Calculating a pdf over Hypotheses

One remaining issue is that our appearance-only SLAM system requires a pdf over hypotheses, whereas our discussion so far has concerned locating only the best hypothesis. Computing a pdf requires a simple modification to the bail-out scheme. Consider that instead of locating only the best hypothesis H^* , we would like to locate all hypotheses whose log-likelihood is at most C less than that of H^* . C is a user-specified constant chosen so that hypotheses less likely than this can be considered to have zero probability with minimal error. Simply increasing our bail-out distance by C will retain all those hypotheses whose final likelihood may be within this likelihood range, thus giving us a close approximation to the pdf over hypotheses.

4.5 Results

We tested the system on the New College and City Centre data sets introduced in Chapter 3. The New College data set is 1.9 km and features a number of challenging cases of perceptual aliasing. The City Centre data set is 2 km in length and is particularly rich in dynamic objects. We also add a third data set – Parks Road – featuring a typical suburban environment. In all three data sets the robot collected images to the left and right of its trajectory approximately every 1.5m. The robot’s appearance model was built from a fourth data set collected in a different region of the city, the area of which did not overlap with the test sets.

Navigation results for these data sets were generated using both the original system outlined in Chapter 3 and the accelerated system incorporating the bail-out test. All data sets were processed using the same visual vocabulary and algorithm parameters. The bail-out boundary was set so that the probability of incorrectly discarding the best hypothesis at any step was $< 10^{-6}$. This value can be varied to trade off speed against accuracy.

Results are summarized in the figures below. Figure 4.2 illustrates the bail-out calculation on some real data. Ordering the features by information gain clearly has a dramatic effect on the effectiveness of the bail-out test. Precision-recall curves for the full and accelerated algorithms on the City Centre data set are shown in Figure 4.4. The curves were generated by varying the probability at which a loop closure was accepted. Recall rates are quoted in terms of image-to-image matches. As a typical loop closure is composed of multiple images, even a recall rate of 35% is sufficient to detect almost all loop closures. The relative performance of the two algorithms on the other data sets is summarized in Table 4.1.

Figure 4.3 visualizes the overall performance of the accelerated algorithm on the City Centre data set. The system correctly identifies a large proportion of possible loop closures with high confidence. There are no false positives that meet the probability threshold. Figures 4.5 and 4.6 show some examples of place recognition performance, highlighting the fact that matching ability in the presence of scene change and robustness to perceptual aliasing is not significantly compromised by the bail-out test. The robustness to perceptual aliasing is particularly noteworthy. Of course, had the examples shown in Figure 4.6 been genuine loop closures they might also have received low probability of having come from the same place. We would argue that this is correct behaviour, modulo the fact that the probabilities in (a) and (b) seem too low. The very low probabilities in (a) and (b) are due to the fact that the best matches for the query images are found in the sampling set, capturing almost all the probability mass. This is less likely in the case of a true but ambiguous loop closure, particularly because in the case of a true loop closure the ambiguity can be resolved by temporal information via the prior term in Equation 2.1.

Dataset	Full Calculation		Fast Bail-Out		Speed-Up
	Recall	Mean Time	Recall	Mean Time	
City Centre	37%	5015 ms	35%	141 ms	35.5
New College	46%	4818 ms	42%	178 ms	27.0
Parks Road	44%	4267 ms	40%	79 ms	53.6

Table 4.1: Comparison of the performance of the SLAM system using full and accelerated likelihood calculations. The recall rates quoted are at 100% precision. Timing results are for the filter update, on a 3GHz Pentium IV. Feature generation adds an extra 330 ms on average. Update time for the accelerated calculation is data dependent and varies from observation to observation. Time quoted is the average over the data set.

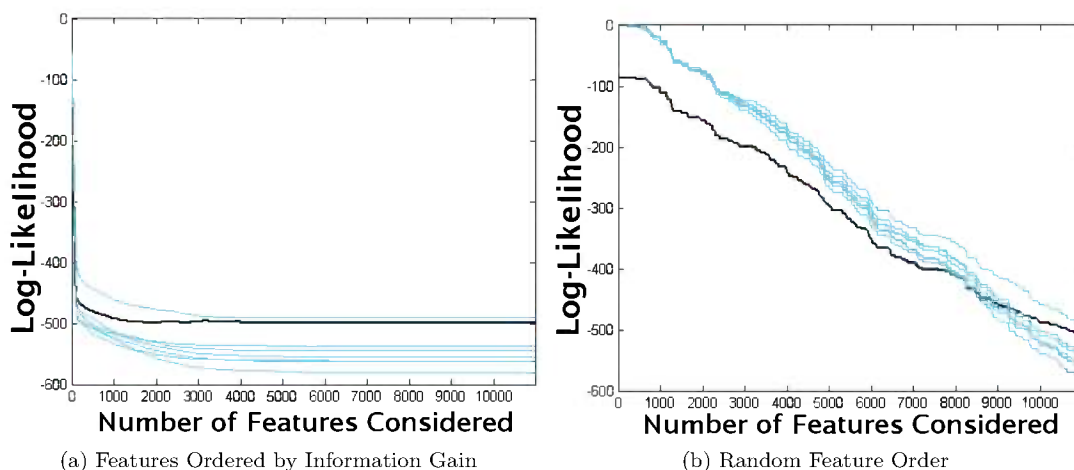


Figure 4.2: Bail-out test on real data. Here the blue lines show the log-likelihoods of each place versus number of features considered. Typically there are thousands of places - here only a few are shown for clarity. The black line is the bail-out threshold. Once the likelihood of a place hypothesis falls below the bail-out threshold, its likelihood calculation can be terminated (the remainder of the likelihood calculation is shown above for illustration). The calculations shown in (a) and (b) are the same, the only difference being the order in which features are considered. In (a), observations are ordered by information gain; in (b) they are ordered randomly. Note that ordering the features by information gain results in much faster convergence toward final likelihood values, and hence a much more effective bail-out test. The bail-out threshold does not converge to the leading hypothesis because of the offset constant C .

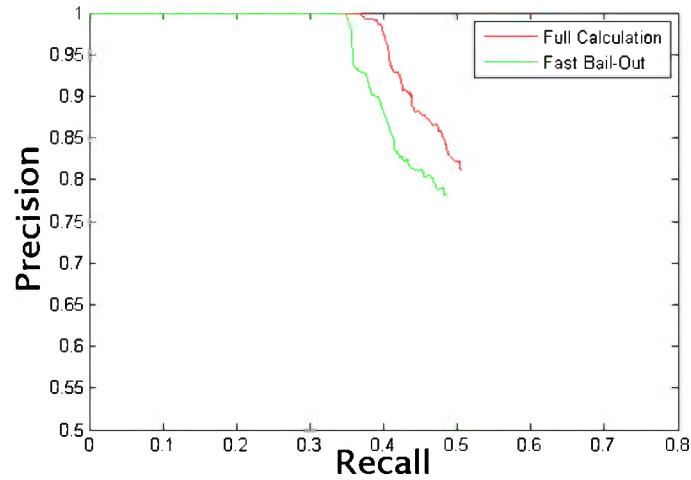


Figure 4.4: Precision-Recall curves for the City Centre data set, showing the full likelihood calculation (red) and the accelerated calculation using the bail-out test (green). Notice the scale.

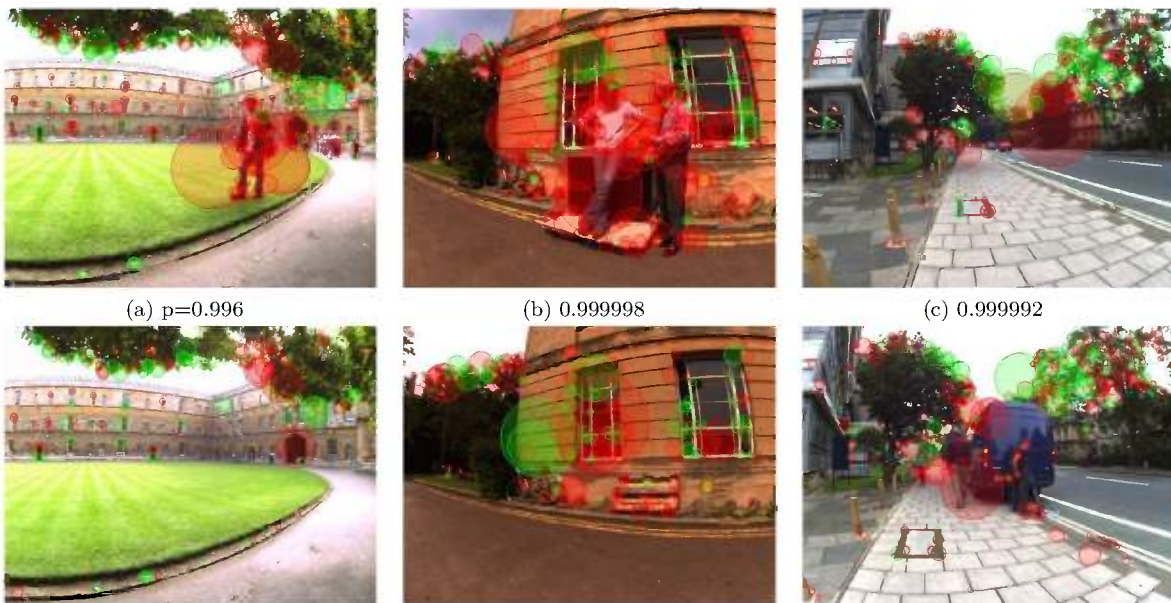


Figure 4.5: Some examples of images that were assigned high probability of having come from the same place, despite scene change. Results were generated using the accelerated likelihood calculation. Words common to both images are shown in green, others in red. The probability that the two images come from the same place is indicated between the pairs.

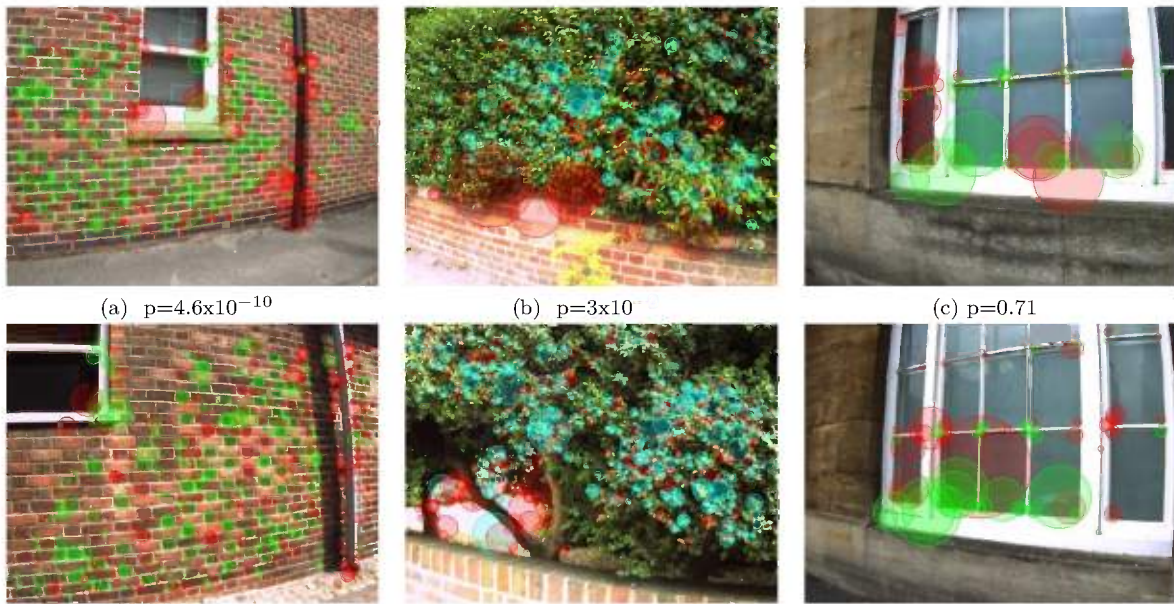


Figure 4.6: Some examples of remarkably similar-looking images from different parts of the workspace that were correctly assigned low probability of having come from the same place. Results were generated using the accelerated likelihood calculation. The examples represent typical system performance. This result is possible because most of the probability mass is captured by locations in the sampling set – effectively the system has learned that images like these are common in the environment. Words present in both images are shown in green, others in red. (Common words shown in blue in (b) for better contrast). The probability that the two images come from the same place is indicated between the pairs.

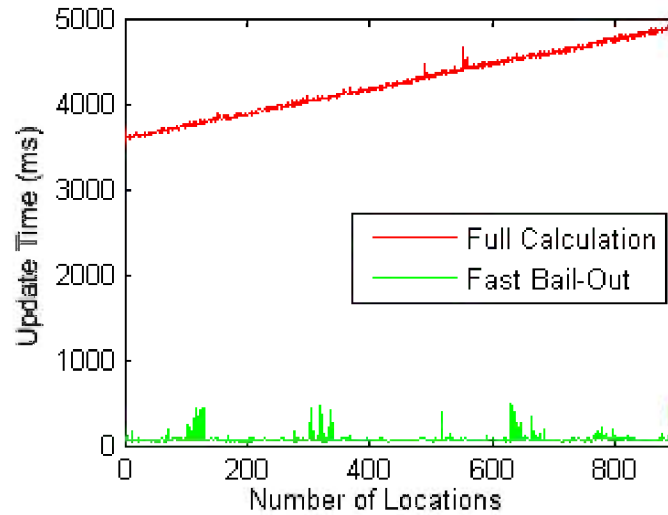


Figure 4.7: Filter update time versus the number of locations in the map, for the Parks Road data set. Update time with zero locations is non-zero due to the fixed cost of evaluating the partition function. Calculation time with the bail-out test grows linearly, however the slope is too small to be seen on this graph.

4.6 Chapter Summary

This chapter has presented a new approach to rapid multihypothesis testing using a probabilistic bail-out condition based on concentration inequalities. Concentration inequalities exist that apply under very general conditions, even for arbitrary functions of non-iid random variables, hence our basic idea should be applicable to a wide variety of problems. We have applied the bail-out test to accelerate our appearance-only SLAM system. The speed increase is data-dependent, but acceleration factors in the range 25x-50x are typical in our tests. The location recognition performance of the accelerated system is only marginally less than that of the full solution, and more than sufficient for reliable online loop closure detection in mobile robotics applications. We have presented results demonstrating online loop-closure detection over 2km loops, though the system is fast enough to scale to loops of tens of kilometres in length while maintaining sub-second filter update times. However, for truly long-term navigation on the order of hundreds or thousands of kilometers, a different approach will be necessary. We investigate appearance-only navigation in this regime in the next chapter.

Chapter 5

Inverted Index Techniques for Large Scale Place Recognition

This chapter is concerned with the problem of appearance-based place recognition at very large scales. Thus far we have considered systems suitable for navigation over a few kilometers (Chapter 3), and approximate inference techniques which extend applicability to a few tens of kilometers (Chapter 4). This chapter builds on the probabilistic framework introduced in those chapters, but modifies the structure of the model to support efficient inference over maps several orders of magnitude larger than previously considered. In seeking such a model, there are some compromises to be made between the fully Bayesian approach considered before and a system which meets the efficiency needs of large scale applications. We describe a formulation which preserves almost all the key features of our earlier model, but allows for the exploitation of the sparsity of visual word data to achieve large reductions in computation and memory requirements. We validate the work on a 1,000 km data set; to date this is the largest experiment conducted with systems of this kind by a considerable margin. The data set, including omni-directional imagery, 20Hz stereo imagery and 5Hz GPS, is available for use by other researchers and is intended to serve as a benchmark for future systems. The chapter concludes with an extensive performance evaluation for the new system, including an analysis of a modified visual



Figure 5.1: Segments of the 1,000 km evaluation trajectory (ground truth).

vocabulary learning stage which is shown to increase performance, and a comparison to the commonly used tf-idf ranking measure, which is considerably out-performed by our new approach.

5.1 Related Work

Appearance-based place recognition for trajectories much above a kilometer long has only recently begun to be considered by SLAM researchers. The small number of relevant papers were introduced in Chapter 2 (Section 2.3.3), and notably include the contributions of Milford and Wyeth [99], and Bosse and Zlot [14]. In the computer vision community, image retrieval from large databases is more commonplace [144, 114, 122, 63]. However, the problem of retrieval from a fixed database is considerably easier than the appearance-based navigation problem, because it is possible to tune the system directly on the images to be recognized, and the difficult issue of new place detection does not arise. We believe the results presented in this chapter currently represent the largest scale system that fully addresses these issues of incrementality and perceptual aliasing.

5.2 System Description

This section discusses the development of a system suitable for appearance-based navigation for environments where the map may contain hundreds of thousands of places or more. We first introduce a modified probabilistic model, and then discuss systems issues that arise. Visual vocabulary and Chow Liu tree learning are revisited in light of the scalability requirements of the new system, and a geometric verification stage is introduced which we found to be almost essential on our largest data sets.

The probabilistic model employed in the new system builds directly on the scheme outlined in Chapter 3. For a highly scalable system, we modify the model so that it is suitable for implementation using an inverted index architecture. We begin by introducing the inverted index.

5.2.1 Inverted Indices

An inverted index is a simple data structure used throughout information retrieval, which enables efficient search of large document collections [87]. If a document is considered as a list of word identifiers, then an inverted index maintains the inverse mapping from words to documents. That is, for each word in the vocabulary, a list of the documents in which that word appears is maintained. Finding all documents that contain a word or set of words is then a very cheap operation. In computational terms, the inverted index approach scales to document collections that are arbitrarily large [19]. The data structure is illustrated in Figure 5.2.

5.2.2 FAB-MAP 2.0 - An Approximation to FAB-MAP for Large Scale Applications

We would like to find a probabilistic model that can take advantage of the scalability of inverted index techniques. Our FAB-MAP model is not directly implementable using an inverted index, because the appearance likelihood $p(Z_k|L_i)$ requires evaluation of Equation 3.12, $\prod_{q=2}^{|v|} p(z_q|z_{p_q}, L_i)$. The computation pattern is illustrated in Figure 5.3. Every observation component contributes to the appearance likelihood, including *negative* observations – those where $z_q = 0$, words not detected in the current image. As such, it does not have the sparsity structure that enables inverted index approaches

Document	Words
1	"September", "rain", "falls", "on", "the", "house"
2	"the", "little", "moons", "fall", "down", "like", "tears"
3	"With", "crayons", "the", "child", "draws", "a", "rigid", "house"
4	"Time", "to", "plant", "tears", "says", "the", "almanac"
5	"and", "the", "child", "draws", "another", "inscrutable", "house"

Word	Documents
a	3
almanac	4
and	5
another	5
child	3, 5
crayons	3
down	2
draws	3, 5
fall	2
falls	1
house	1, 3, 5

tears	2, 4
the	1, 2, 3, 4, 5
Time	4
to	4

(a) Documents in the database.

(b) Inverted index.

Figure 5.2: Illustration of an inverted index data structure. A mapping from words to documents is maintained, enabling fast retrieval.

to scale. Perhaps surprisingly, we have found that simply ignoring the negative observations has a detrimental impact on place recognition performance. Thus we seek a formulation that will enable efficient implementation, but preserve the information inherent in the negative observations.

To enable an inverted index implementation, we modify the probabilistic model in two ways. Firstly, we place some restrictions on the probabilities in the location models. Recalling Equation 3.5, location models are parametrized as $\{p(e_1 = 1|L_i), \dots, p(e_{|v|} = 1|L_i)\}$, that is, by a set of beliefs about the existence of scene elements that give rise to observations of the words in the vocabulary. Let $p(e_q|L_i)|_{\{0\}}$ denote one of these beliefs, where the subscript $\{0\}$ indicates the history of observations that have been associated with the location. Thus $\{0\}$ denotes one associated observation with $z_q = 0$, and $\{0, 0, 1\}$ denotes three associated observations, with $z_q = 1$ in one of those observations. Further, let $p(e_q|L_i)|_0$ indicate that in all observations associated with the location, $z_q = 0$.

In the general probabilistic model described in Section 3.2, $p(e_q|L_i)|_0$ can take on a range of values – for example, $p(e_q|L_i)|_{\{0\}} \neq p(e_q|L_i)|_{\{0,0\}}$, as the belief in the non-existence of the scene elements increases as more supporting observations become available. In FAB-MAP 2.0, the model is restricted

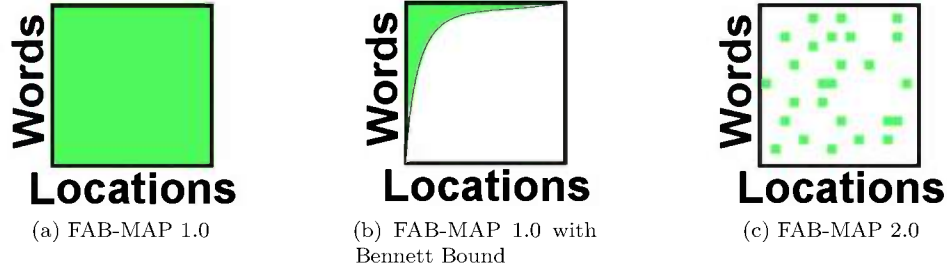


Figure 5.3: Illustration of the amount of computation performed by the different models. The shaded region of each block represents the appearance likelihood terms $p(z_q|z_{p_q}, L_i)$ which must be evaluated. In (a), FAB-MAP 1.0 (Chapter 3), the likelihood must be computed for all words in all locations in the map. Using the Bennett bound approximate inference procedure defined in Chapter 4, unlikely locations are discarded during the computation, yielding the evaluation pattern shown in (b). The restrictions imposed in FAB-MAP 2.0 allow a fully sparse evaluation, (c).

so that $p(e_q|L_i)|_0$ must take the same value for all locations; it is clamped at the value $p(e_q|L_i)|_{\{0\}}$. This restriction enables an efficient likelihood calculation, illustrated in Figure 5.4.

Consider the calculation of one term of the observation likelihood, as per Equation 3.13, across all locations in the map. That is, we wish to compute the term $p(z_q|z_{p_q}, L_i)$ for some visual word q , for all L_i in the map. Recalling Chapter 3, the term is given by

$$p(z_q|z_{p_q}, L_i) = \sum_{s_{e_q} \in \{0,1\}} p(z_q|e_q = s_{e_q}, z_{p_q})p(e_q = s_{e_q}|L_i)$$

In the unrestricted model, this will involve computing one term for each location, as illustrated in Figure 5.4a.

L_1	L_2	L_3	L_4
$p(z_q z_{p_q}, L_1)$	$p(z_q z_{p_q}, L_2)$	$p(z_q z_{p_q}, L_3)$	$p(z_q z_{p_q}, L_4)$

In the restricted model, Figure 5.4b, the term takes a single common value for all locations where word q was not previously observed (in those locations, the word exists with probability $p(e_q|L_i)|_0$, which determines the value of Equation 3.13. We denote this value by $p(z_q|z_{p_q}, L)|_0$).

$p(z_q z_{p_q}, L_1)$	$p(z_q z_{p_q}, L_2) _0$	$p(z_q z_{p_q}, L_3) _0$	$p(z_q z_{p_q}, L_4)$
-----------------------	--------------------------	--------------------------	-----------------------

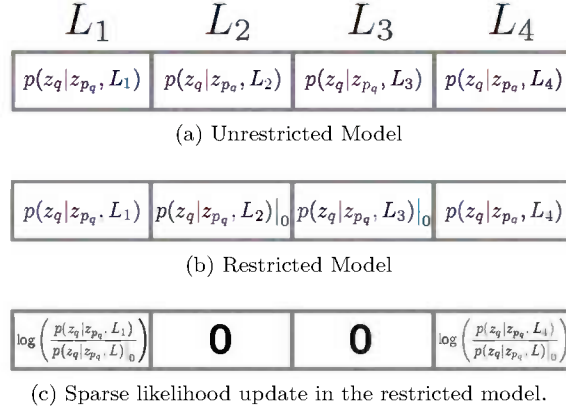


Figure 5.4: Illustration of the calculation of one term of Equation 3.12, i.e. the observation likelihood due to a particular word in the vocabulary. This example shows a map with four locations. The total observation likelihood for a given location, $p(Z|L)$, is calculated by evaluating the illustrated terms for all words in the vocabulary. In (a), the model is unrestricted, and the likelihood term can take a different value for each location. In (b), the restricted model, the likelihood term in all locations where the currently considered word was not previously observed is constrained to take the same value. The calculation can now be organized so that it has a sparse structure, (c).

Working with log-likelihoods, and given that the distribution will later be normalized, the calculation can be reorganized so that it has a sparse structure (Figure 5.4c).



This allows for efficient implementation using an inverted index. The terms shown in Figure 5.4c can be thought of as the weights of the votes a word casts for a location.

We emphasize the fact that the restriction we have placed on the model is slight, and most of the power of the original model is retained. During the exploration phase, when only one observation is associated with each location, the two schemes are identical¹. The “fixed” terms $p(e_q|L_i)|_0$ can (and do) vary with q (word ID), and in principle also with time. Treatment of correlations between words, of perceptual aliasing, and of the detector model remains unaffected.

The second change we make to the model concerns data association. Previously, data association was carried out via Equation 3.23, updating the beliefs $p(e_q|L_i)$. Effectively this amounts to capturing the

¹Assuming the detector model does not change with time.

average appearance of a location. For example, if a location has a multi-modal distribution over word occurrence, such as a door that may be either open or shut, then the location appearance model will approach the mean of this distribution. In FAB-MAP 1.0, when computation increased swiftly with the number of appearance models to be evaluated, this was a reasonable design choice. For FAB-MAP 2.0 we switch to representing locations in a sample-based fashion, which better handles these multi-modal appearance effects. Locations now consist of a set of appearance models as defined in Equation 3.5, with each new observation associated with the location defining a new such model. This change is necessary because of the restrictions placed on scene element existence probabilities, but is also largely beneficial. In addition to improving our ability to deal with multi-modal location appearance, it also makes data association a reversible operation, and makes the implementation of a scheme that maintains multiple hypotheses over data association decisions (e.g. [128]) very efficient. While it means that inference time now increases with every observation collected, the system is sufficiently scalable that this is not of immediate relevance, and the greater ability to deal with variable location appearance is preferred.

Algorithm 5.1 gives pseudocode for the calculation. The complexity of this implementation is $O(\#vocab)$, the number of words in the visual vocabulary. This straight-forward implementation is in fact fast enough for practical use, however it can be improved by a caching scheme which yields an algorithm with complexity effectively independent of vocabulary size. The key observation is that for a given word and a given location, the log-likelihood increments (or “votes”) to be calculated in Algorithm 5.1, $\log\left(\frac{p(z_q=s_q|z_{p_q}=s_{p_q},L_i)}{p(z_q=s_q|z_{p_q}=s_{p_q},L)}\right)_0$, have four possible states:

$$\begin{aligned}
\text{Case 1 :} & \quad (s_q = 1, s_{p_q} = 1) \\
\text{Case 2 :} & \quad (s_q = 1, s_{p_q} = 0) \\
\text{Case 3 :} & \quad (s_q = 0, s_{p_q} = 1) \\
\text{Case 4 :} & \quad (s_q = 0, s_{p_q} = 0)
\end{aligned} \tag{5.1}$$

which depend on whether or not the word q and its parent word p_q are present in the current observation Z_k . As observations are typically sparse, case four, $(s_q = 0, s_{p_q} = 0)$, will be by far the most common. We can exploit this fact by pre-calculating the likelihood of a location L_i as if all votes were from case

Algorithm 5.1 Log-likelihood update using the inverted index.

```

for  $q$  in vocabulary do:
    //Get all locations where word  $q$ 
    //was observed
    locations = inverted_index[ $q$ ]
    for  $L_i$  in locations do:
        //Update the log-likelihood
        //of each of these locations
        loglikelihood[ $L_i$ ] +=  $\log\left(\frac{p(z_q=s_q|z_{p_q}=s_{p_q},L_i)}{p(z_q=s_q|z_{p_q}=s_{p_q},L)_0}\right)$ 
```

four. This likelihood is calculated once, when the location is added to the map, and cached. We refer to this as the location’s “default likelihood”. When processing a new observation, we need only adjust the default likelihood of a location to take account of those words which actually lie in cases one to three for the current observation. Calculating this adjustment involves only those words that are present in the current observation (cases 1,2), or that are children of these words in the Chow Liu tree (case 3). The number of words present in a given observation is typically a small constant independent of vocabulary size. The number of words that relate to case three depends on the structure of the Chow Liu tree, and in pathological cases could still be $O(\#vocab)$. However, in practice we observe it to be a small multiple of the number of observed words. Pseudocode for the algorithm which exploits this sparsity is given in Algorithm 5.2. In our experiments using a 100,000 word vocabulary, we observed an order of magnitude speed increase with this approach.

5.2.3 Geometric Verification

While a navigation system based entirely on the bag-of-words likelihood is possible (Chapter 3), we have found in common with others [122] that a post-verification stage, which checks that the matched images satisfy geometric constraints, considerably improves performance. The impact is particularly noticeable as data set size increases - it is helpful on our 70 km data set but almost essential on the 1,000 km set.

We apply the geometric verification to a “shortlist” of the 100 most likely locations (those which maximize $p(Z_k|L_i, \mathcal{Z}^{k-1})p(L_i|\mathcal{Z}^{k-1})$) and to the 100 most likely samples (the location models used to

Algorithm 5.2 Log-likelihood update using the inverted index and exploiting observation sparsity.

```

Update, Part A (Default Likelihood):
//Each location's likelihood is initialized
//to appropriate "default likelihood"
//which assumes an "null" observation with  $z_q = 0, \forall q$ 
//This can be thought of as the sum of "default votes"  $D_q$ 
//for each observed word at the location
// $D_q = \log\left(\frac{p(z_q=0|z_{pq}=0, L_i)}{p(z_q=0|z_{pq}=0, L)}\right)_0$ 
//Note that the default likelihood will be
//different for each location.
Initialize_Locations_To_Default_Likelihood()
Update, Part B (Observations such that  $z_q = 1$ ):
//Now, adjust the votes based on the content of the current observation.
for  $z_q$  in  $Z$ , such that  $z_q = 1$  do:
    //Get all locations where word  $q$ 
    //was observed
    locations = inverted_index[q]
    for  $L_i$  in locations do:
        //Update the log-likelihood
        //of each of these locations
        //by removing the default vote  $D_q$ 
        //and adding the appropriate vote.
        loglikelihood[ $L_i$ ] +=  $\log\left(\frac{p(z_q=1|z_{pq}=s_{pq}, L_i)}{p(z_q=1|z_{pq}=s_{pq}, L)}\right)_0 - D_q$ 

Update, Part C (Observations such that  $z_q = 0$  and  $z_{pq} = 1$ ):
//Same as Part B, but for unobserved words that are
//children of observed words in the CL tree.
for  $z_q$  in  $Z$ , such that  $z_q = 0$  and  $z_{pq} = 1$  do:
    locations = inverted_index[q]
    for  $L_i$  in locations do:
        loglikelihood[ $L_i$ ] +=  $\log\left(\frac{p(z_q=0|z_{pq}=1, L_i)}{p(z_q=0|z_{pq}=1, L)}\right)_0 - D_q$ 

```



Figure 5.5: A example of geometric verification, showing inliers identified by RANSAC. Note that we are not recovering an exact pose-to-pose transformation; the correspondences are only approximately geometrically consistent.

evaluate the normalizing term $p(Z_k | \mathcal{Z}^{k-1})$. For each of these locations we check geometric consistency with the current observation using RANSAC [46]. Candidate interest point correspondences are derived from the bag-of-words assignment already computed. Because our aim is only to verify approximate geometric consistency rather than recover exact pose to pose transformations, we assume a highly simplified model where the transformation between poses is constrained to be a pure rotation about the vertical axis. A single point correspondence then defines a transformation. Due to this simplified model, and also because our point correspondences typically have few outliers, the geometric verification is very rapid. Only a few RANSAC iterations are required – we assume 65% inliers and so only 13 RANSAC iterations are needed to recover a model with an expected 10^{-6} error rate. The pure-rotation model is a gross approximation, but given the constrained motion of our vehicle mounted camera, it is good enough to give a substantial boost to recognition performance, while imposing very little computational overhead. We accommodate some translation between poses by allowing large inlier regions for point correspondences (up to 50 pixels in x and y, and a factor of 4 in scale). Typical “consistent” correspondences are shown in Figure 5.5. Having recovered a set of inliers using RANSAC we recompute the location’s likelihood by setting $z_q = 0$ for all those visual words not part of the inlier set. A likelihood of zero is assigned to all locations not subject to geometric verification. For the 1,000km experiment, the mean time taken to geometrically verify and re-rank all 200 shortlisted locations was only 10 ms and the maximum time was 145 ms.

5.2.4 Visual Vocabulary Learning At Large Scale

Clustering

A number of challenges arise in learning visual vocabularies at large scale. The number of SURF features extracted from training images is typically very large; our relatively small training set of 1,921 images produces 2.5 million 128-dimensional SURF descriptors occupying 3.2 GB. Even the most scalable clustering algorithms such as k-means are too slow to be practical. Instead we apply the fast approximate k-means algorithm discussed in [122], where, at the beginning of each k-means iteration, a randomized forest of kd-trees [142, 105] is constructed over the cluster centres, which is then used for fast (approximate) distance calculations. This procedure has been shown to outperform alternatives such as hierarchical k-means [114] in terms of visual vocabulary retrieval performance.

As k-means clustering typically converges only to a local minimum of its error metric, the quality of the visual vocabulary is sensitive to the initial cluster locations supplied to k-means. Nevertheless, random initial locations are commonly used. We have found that this leads to poor visual vocabularies, because there are very large density variations in the feature space. In these conditions, randomly chosen cluster centres tend to lie largely within the densest region of the feature space, and the final clustering over-segments the dense region, with poor clustering elsewhere. For example, in our vehicle-collected data, huge numbers of very similar features are generated by road markings, whereas rarer objects (more useful for place recognition) may only have a few instances in the training set. Randomly initialized k-means yields a visual vocabulary where a large fraction of the words correspond to road markings, with tiny variations between words. Similar effects were observed by Jurie and Triggs [69]. Examples are shown in Figure 5.6.

To avoid these effects, we choose the initial cluster centres for k-means using a fixed-radius incremental pre-clustering, where the data points are inspected sequentially, and a new cluster centre is initialized for every data point that lies further than a fixed threshold from all existing clusters. This is similar to the furthest-first initialization technique [35], but more computationally tractable for large data sets. We also modify k-means by adding a cluster merging heuristic. After each k-means iteration, if any two cluster centres are closer than a fixed threshold, one of the two cluster centres is reinitialized to a

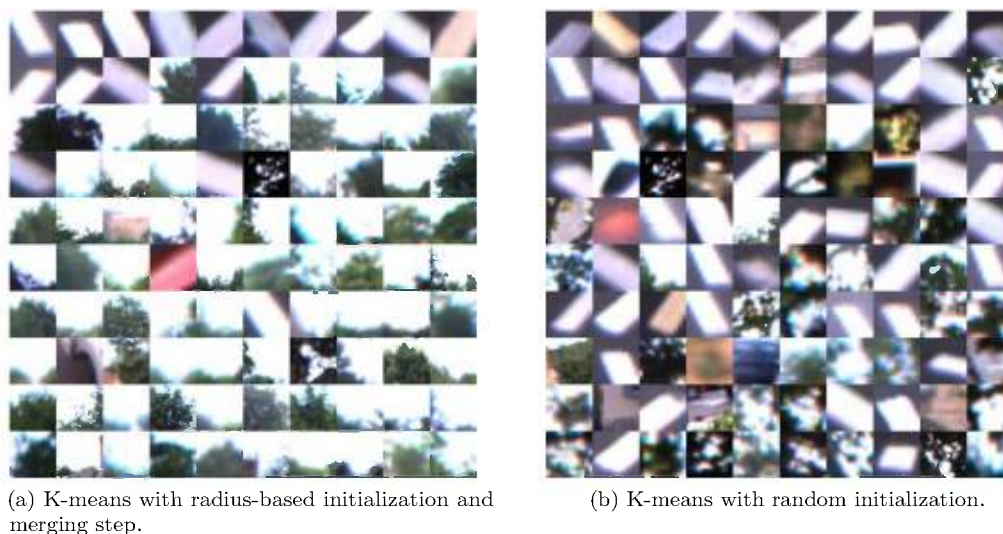


Figure 5.6: The 100 most common visual words in the vocabularies used for the car-based experiments, showing one exemplar per word. With random initialization (Sub-figure (b)) k-means tends to over-segment the densest regions of feature space, leading to a visual vocabulary with many highly similar visual words (in this case, many words corresponding to near-identical views of road markings). Using radius-based initialization and cluster merging (Sub-figure (a)) produces a visual vocabulary with words that are better separated in feature space.

random location.

The modified clustering gives a robust boost to system performance (Figure 5.7). We have observed the effect on multiple data sets and under various different system configurations (with and without geometric verification, etc.). Curiously, however, we do not see the effect when using tf-idf ranking. We have no intuitive explanation for why tf-idf does not benefit in a similar way to FAB-MAP.

Chow Liu Tree Learning

Chow Liu tree learning is also challenging at large scale. The standard algorithm for learning the Chow Liu tree involves computing a (temporary) mutual information graph of size $|v|^2$, so the computation time is quadratic in the vocabulary size. For the 100,000 word vocabulary discussed in Section 5.4, the relevant graph would require 80 GB of storage. Happily, there is an efficient algorithm for learning Chow Liu trees when the data of interest is sparse [92]. Meilă's algorithm has complexity $O(s^2 \log s)$,

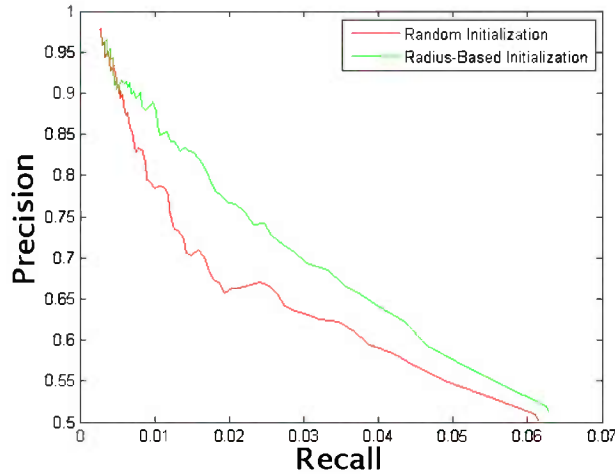


Figure 5.7: Precision-recall curves showing the effect of k-means initialization on overall performance. Performance shown is for a subset of the 1,000km data set, ranked according to the baseline FAB-MAP 2.0 model (i.e. not including motion model or geometric check). Detector terms were tuned separately for each vocabulary, so as to make the comparison fair. The performance difference persists after geometric re-ranking is added to the model. Curiously, we do not observe the effect when using tf-idf ranking.

where s is a sparsity measure, equal to the maximum number of visual words present in any training image. Visual word data is typically very sparse, with only a small fraction of the vocabulary present in any given image. This allows efficient Chow Liu tree learning even for large vocabulary sizes. For example, the tree of the 100,000 word vocabulary used in Section 5.4 was learned in 31 minutes on a 3GHZ Pentium IV.

For both the clustering and Chow Liu learning, we use external memory techniques to deal with the large quantities of data involved [38].

5.3 Data Set

For a truly large scale evaluation of the system, the experiments in this chapter make use of a 1,000km data set. The data was collected by a car-mounted sensor array (see Appendix A), and consists of omni-directional imagery from a Point Grey Ladybug2, 20Hz stereo imagery from a Point Grey

Bumblebee², and 5Hz GPS data. Omni-directional image capture was triggered every 4 meters on the basis of GPS. The omni-directional images were captured at 1920x512 resolution, and the stereo images at 512x384.

The data set was collected over six days in December, with a total length of slightly less than 21 hours, and includes a mixture of urban, rural and motorway environments. The total set comprises 803 GB of imagery (including stereo) and 177 GB of extracted features. There are 103,256 omni-directional images, of which 49,493 are loop closures. The median distance between image captures is 8.7m – this is larger than the targeted 4 m because the Ladybug2 could not provide the necessary frame rate during faster portions of the route. The median time between image captures is 0.48 seconds, which provides our benchmark for real-time image retrieval performance.

Two supplemental data sets were also collected. A set of 1,921 omni-directional images collected 30m apart was used to train the visual vocabulary and Chow Liu tree, and also served as the sampling set for the Monte Carlo integration required in Equation 3.21. The area where this training set was collected did not overlap with that of the test data sets. A second smaller data set of 70 km was also collected in August, four months prior to the main 1,000 km data set. This serves as a smaller-scale test and could also be used for testing cross-season matching performance. The data sets are summarized in Table 5.1.

The 1,000 km data set, collected in mid-December, provides an extremely challenging benchmark for place recognition systems. Due to the time of year, the sun was low on the horizon, so that scenes typically have high dynamic range and quickly varying lighting conditions. We developed custom auto-exposure controllers for the cameras that largely ensured good image quality, however, there is unavoidable information loss in such conditions. Additionally, large sections of the route feature self-similar motorway environments, which provide a challenging test of the system’s ability to deal with perceptual aliasing. The smaller data set collected during August features more benign imaging conditions and will demonstrate the performance that can be typically expected from the system.

Finally, collecting a data set of this magnitude highlights some practical challenges for any truly robust field robotics deployment. We encountered significant difficulty in keeping the camera lenses

²Not used in these results.

Table 5.1: Data set summary.

Data Set	No. of Images	No. of Loop Closures	Median distance between images	Extracted Features	Environment
1,000 km	103,256	48,493	8.7 m	177 GB	Motorways, Urban, Rural
70 km	9,575	4,757	6.7 m	16 GB	Urban, Rural

clean – in winter from accumulating moisture and particulate matter, in summer from fly impacts. For this experiment we periodically cleaned the cameras manually – a more robust solution seems a worthy research topic.

All data sets are available to researchers upon request.

5.4 Results

We now present the system performance evaluation. Overall performance is outlined, the impact of the Chow Liu tree is examined, and the system is benchmarked against the common tf-idf weighting function.

5.4.1 Test Conditions

The system was tested on the two data sets, respectively 70 km and 1,000 km. As input to the system, we used 128D non-rotationally invariant SURF descriptors. These features were quantized to visual words using a randomized forest of eight kd-trees. The visual vocabulary and Chow Liu tree were trained using the system described in Section 5.2.4 and the 1,921 image training set described in Section 5.3. In order to ensure an unbiased Chow Liu tree, the images in the training set were collected 30 m apart, so that as far as possible they do not overlap in viewpoint, and thus approximate independent samples from the distribution over images.

We investigate two different visual vocabularies, of 10,000 and 100,000 words respectively. The detector model (Equation 3.6), the main user-configurable parameter of our system, was determined by a grid search to maximize performance on a set of training loop closures. The detector model primarily captures the effects of variability in SURF interest point detection and feature quantization error. For

the 10,000 word vocabulary we set $p(z = 1|e = 1) = 0.39$ and $p(z = 1|e = 0) = 0.005$. For the 100,000 word vocabulary, the values were $p(z = 1|e = 1) = 0.2$ and $p(z = 1|e = 0) = 0.005$. The likelihood smoothing term σ introduced in Section 3.2.4 was set to 0.99, except in the case where the geometric check was used, where we found it to be unnecessary. This means that when the geometric check was applied, the system could accept a loop closure on the basis of a single image. Finally, we also investigate the importance of learning the Chow Liu tree by comparing against a Naive Bayes formulation which neglects the correlations between words. We refer to these different system configurations as “100k, CL” and “100k, NB”, and similarly for the 10k word vocabulary.

Performance of the system was measured against ground truth loop closures determined from the GPS data. GPS errors and dropouts were corrected manually. Any pair of matched images that were separated by less than 40 m on the basis of GPS was accepted as a correct correspondence. Note that while 40 m may seem too distant for a correct correspondence, on divided highways the minimum distance between correct loop closing poses was sometimes as large as this. Almost all loop closures detected by the system are well below the 40 m limit: 89% were separated by less than 5 m, and 98% by less than 10 m (See Figure 5.8).

We report precision-recall metrics for the system. The evaluation metrics are the same as those outlined in Chapter 3 (Section 3.4.2). To recap, precision is defined as the ratio of true positive loop closure detections to total detections. Recall is the ratio of true positive loop closure detections to the number of ground truth loop closures. Images for which no loop closure exists cannot contribute to the true positive rate, however they can generate false positives. Likewise true loop closures which are incorrectly assigned to a “new place” depress recall but do not impact our precision metric. These metrics provide a good indication of how useful the system would be for loop closure detection as part of a metric SLAM system – recall at 100% precision indicates the percentage of loop closures that can be detected without any false positives that would cause filter divergence. Finally, note that a typical loop closure consists of a sequence of several images, so even a recall rate of 20% or 30% is sufficient to detect most loop closure events, provided that the detections have uniform spatial distribution.

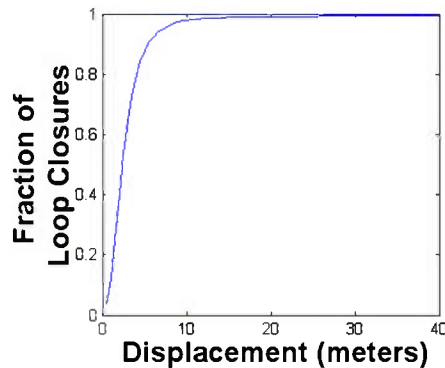


Figure 5.8: The typical displacement in meters between two images identified as loop closures. While any pair separated by less than 40 m is accepted as a true positive, because the ground truth separation can occasionally be this large, 89% of detected loop closures are separated by less than 5m, and 98% by less than 10 m.

5.4.2 Overall Performance

Overall, we found the system to have excellent performance on the 70 km data set, while the 1,000 km data set was more challenging. Precision recall curves for the two data sets are shown in Figure 5.9, and given numerically in Table 5.2. Loop closing performance is also visualized in the maps shown in Figures 5.14 and 5.15. Loop closures are often detected even in the presence of large changes in appearance, typical examples are shown in Figures 5.16 and 5.17. It is worth noting also that there are many examples of loop closures correctly detected by FAB-MAP but not by GPS, particularly under foliage and in city centres.

The performance contributions of the motion model and the geometric verification step are analysed in Figure 5.9 and presented numerically in Table 5.2. The geometric check in particular is useful in maintaining recall at higher levels of precision. The motion model is largely unnecessary on the 70 km set. On this set we detect 44% of all pose-to-pose correspondences at 100% precision, *without using any temporal information*. These loop closures are detected on the basis of a single image. At 99% precision, the recall rises to 69.9%. On the 1,000 km set, the motion model makes a more noticeable contribution. Note, however, that motion model we use is very weak. Stronger motion constraints, for example from a visual odometry system, would be expected to have a much larger impact. In combination with such motion information, it seems that it should be possible to achieve close to 100% recall on the 70 km set.

The effect of vocabulary size and the Chow Liu tree on performance is shown in Figure 5.10 and Table 5.3. In common with other authors [114, 122], we find that performance increases strongly with vocabulary size. The Chow Liu tree also boosts performance on all data sets and at all vocabulary sizes. The effect is weaker at the very highest levels of precision. The reason for this effect is that while the Chow Liu tree will on average improve the likelihood estimates assigned, some individual likelihoods may get worse. The recall at 100% precision is determined by the likelihood assigned to the very last false positive to be eliminated. While on average we expect the Chow Liu tree to improve this likelihood estimate, the opposite may be observed in some fraction of data sets. Below 100% precision the results are sensitive to the likelihood estimates for a larger number of false positives, and so the improvement due to the Chow Liu tree is more robustly observable.

The recall rate for the 70 km data set is 48.4% at 100% precision, rising to 73.2% at 99% precision. The spatial distribution of these loop closures is uniform over the trajectory – thus essentially every pose will be either detected as a loop closure, or a lie within a few meters of a loop closure. There are two short segments of the trajectory where this is not the case, one in a forest with poor lighting conditions, another in open fields with few visual landmarks. For practical purposes this data set can be considered “solved”.

By contrast, the recall for the 1,000 km data set at 100% precision is only 3.1%. However, this figure requires careful interpretation – the data set contains hundreds of kilometers of motorways, where the environment is essentially devoid of distinctive visual features (see Figure 5.20). It is perhaps not reasonable to expect appearance-based loop closure detection in such conditions. To examine performance more closely, we considered separately the results for portions of the trajectory where the vehicle is travelling below 50 km/h (mainly urban areas). We refer to this evaluation as “1,000 km Urban” in Tables 5.2 and 5.3. For these images (31% of the data set) the recall is 6.5% at 100% precision, rising to 18.5% at 99% precision. Note that the retrieval here is performed against the complete 1,000 km data set, the only salient difference being the distinctiveness of the query images. Given that the loop closures have an even distribution over the trajectory (Figure 5.14), even a recall rate of 6.5% is likely sufficient to support a good metric SLAM system.

Both data sets exhibit a sharp drop in recall between 100% and 99% precision. This drop is caused

Data Set	70 km		1,000 km			1,000 km Urban	
Precision	100%	99%	100%	99%	90%	100%	99%
Recall							
Motion Model and Geometric Check	48.5	73.2	3.1	8.3	14.3	6.5	18.5
Geometric Check Only	44.0	69.9	2.6	5.1	9.7	6.9	12.2
Motion Model Only	23.1	41.8	0.2	0.2	4.7	0.6	0.7
Baseline	-	-	-	-	0.5	-	-

Table 5.2: Recall figures at specified precision showing the effect of different system components. A dash indicates that there is no threshold that produces the specified precision level. The same information is presented as a precision-recall curve in Figure 5.9. “Baseline” refers to the system without the geometric check and with a uniform position prior at each timestep. “Motion model” includes the position prior $p(L_i|Z^{k-1})$, allowing loop closures to benefit from temporal support. “Geometric Check” re-ranks the top 100 most likely locations by considering the geometric consistency of matched image interest points.

by particularly challenging cases of perceptual aliasing, such as encountering rare-but-repetitive objects in environment. Figure 5.18 shows the two highest confidence false positives from the 1,000 km set, typical of these difficult cases. The scenes have high similarity in both a bag-of-words and geometric sense, however the primary reason that they are difficult to identify as false positives is that the repetitive objects they contain are relatively uncommon in the environment, and so are not easily captured by the sampling set. By contrast, the scenes shown in Figure 5.20 do not cause such problems, despite high similarity, because the content of the image is common in the environment. Given that the system’s sampling set consists of less than 2,000 images, these effects are perhaps not surprising. If a second navigation experiment were conducted using all of the 1,000km data for training, we may begin to develop robustness even to occasional repeated features such as those in Figure 5.18. Other natural methods to deal with these cases include relying more heavily on temporal support, or perhaps some level of semantic verification such as rejecting the loop closure in Figure 5.18b because the matched object is a vehicle.

5.4.3 Comparison to tf-idf

Term-frequency inverse-document-frequency (tf-idf) is a standard ranking metric used in most existing visual search engines [144, 122, 63, 87]. To compare FAB-MAP against this baseline in the most transparent way possible, we examined performance on a pure retrieval task. For each image in our

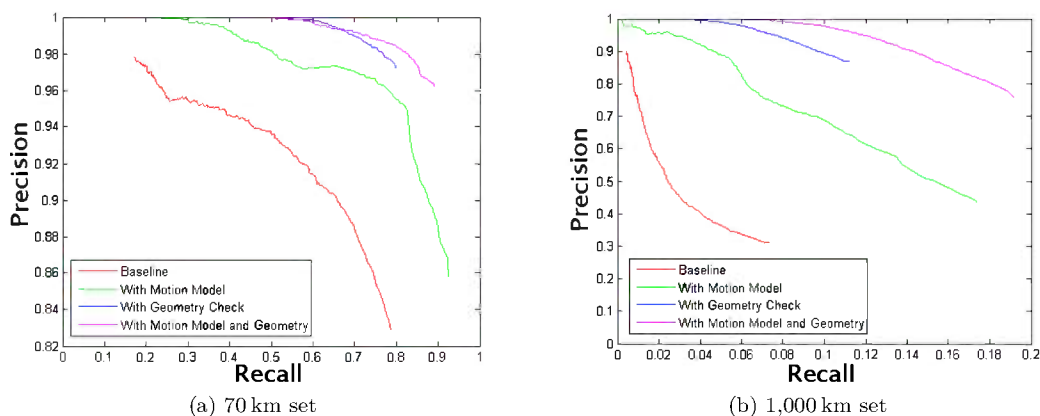


Figure 5.9: Precision-recall curves showing the effect of the different system components on performance. Note the scaling on the axes. Results shown are for the 100k vocabulary with Chow Liu tree. Relative performance in other configurations is similar. “Baseline” refers to the system without the geometric check and with a uniform position prior at each timestep. “Motion model” includes the position prior $p(L_i|Z^{k-1})$, allowing loop closures to benefit from temporal support. “Geometric Check” re-ranks the top 100 most likely locations by considering the geometric consistency of matched image interest points.

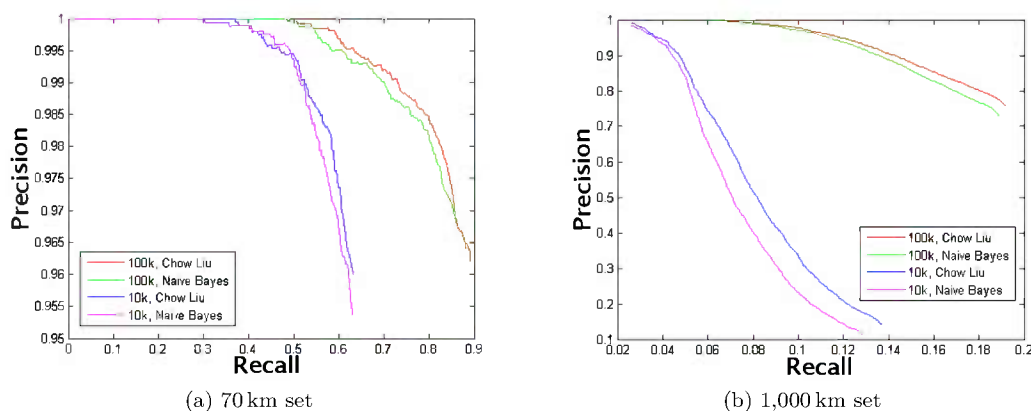


Figure 5.10: Precision-recall curves showing the effect of the vocabulary size and the Chow Liu tree on performance. Note the scaling on the axes. Performance shown includes motion model and geometric check. Performance increases strongly with vocabulary size. The Chow Liu tree also increases performance, for all vocabulary sizes.

Data Set	70 km		1,000 km			1,000 km Urban	
Precision	100%	99%	100%	99%	90%	100%	99%
Recall							
100k CL	48.5	73.2	3.1	8.3	14.3	6.5	18.5
100k NB	49.1	70.0	3.7	7.9	13.5	7.5	17.9
10k CL	37.0	52.3	-	2.7	4.7	-	5.2
10k NB	30.1	51.5	-	-	4.4	-	-

Table 5.3: Recall figures at specified precision for varying vocabulary size and with/without the Chow Liu tree. A dash indicates that there is no threshold that produces the specified precision level. The same information is presented as a precision-recall curve in Figure 5.10. Recall improves with increasing vocabulary size at all levels of precision. The Chow Liu tree also improves recall in all cases with the exception of the 100k vocabulary at 100% precision. The 100% precision figure is sensitive to the probability assigned to all possible false positives, so can be skewed by a single outlier with a high likelihood. So whereas the Chow Liu tree yields better probability estimates in general, the effect is more robustly observable at lower levels of precision, where it cannot be masked by a small number of outliers.

data sets where at least one valid match exists, we computed the ranking according to tf-idf weighted cosine distance and also according to the FAB-MAP likelihood $p(Z_k|L_i)^3$. This ranking-only task is intended to examine the likelihood function alone, so does not involve new place detection, motion model effects or geometric re-ranking. For the 1,000 km data set there are 48,493 images that have at least one valid loop closure; for the 70 km set there are 4,757. Retrieval was performed against the set of images collected up to the point of loop closure. There are a variety of ways to perform the tf-idf weighting - we have followed [63] and have verified that our implementation gives results identical to those reported there.

Precision-recall curves showing relative performance are given in Figure 5.11. FAB-MAP substantially outperforms tf-idf, the difference being particularly dramatic on the 1,000 km data set.

The performance of tf-idf results could perhaps be improved by applying various known tweaks to the measure - for example by taking account of word burstiness [65] or using pivoted normalized document lengths [87] among others. However, it seems to us that increasing performance in this way essentially amounts to finding heuristics by trial-and-error. Indeed, to achieve the performance reported here already required considerable experimentation with various aspects of the tf-idf measure, such as whether to use L1 or L2 normalization, whether to apply tf-idf weighting to query or document vectors

³Note that the tf-idf measure has access to word count (tf) information which is not used by FAB-MAP.

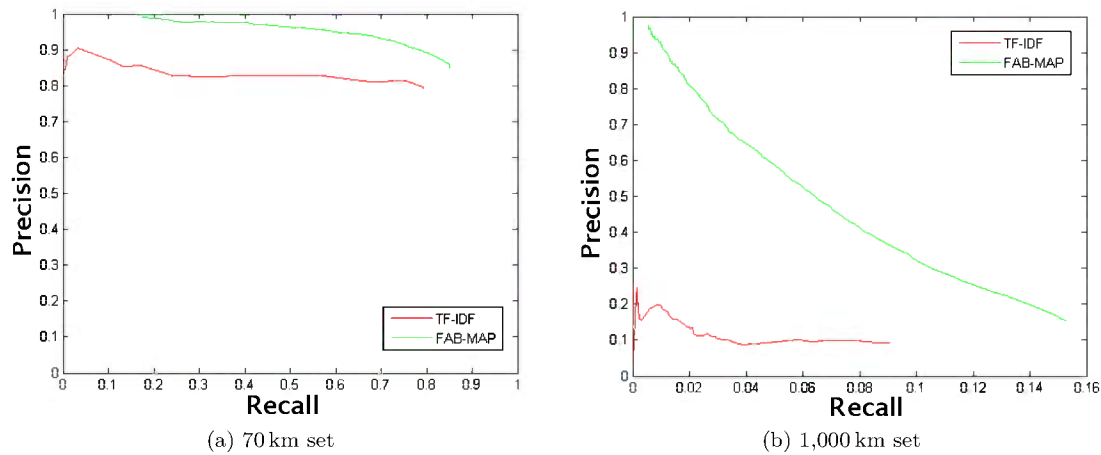
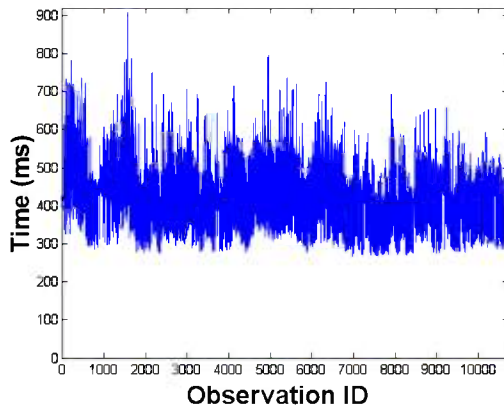


Figure 5.11: Comparison to tf-idf ranking. FAB-MAP substantially outperforms this standard ranking metric, particularly on the larger data set. To ensure the fairest possible comparison, these performance figures relate to a retrieval-only task that excluded new place detection. See Section 5.4.3 for details.

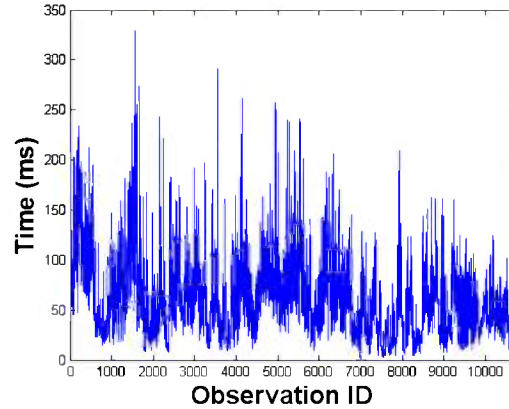
or both, how the tf counts should be normalized, etc. Some of these choices, particularly the choice of vector normalization, have a dramatic impact on ranking performance, without any clear intuition as to why. FAB-MAP by contrast is a natural generative framework which provides clear rationale for the structure of the ranking function and offers paths to improved performance via extensions to the generative model (see Chapter 6). Of course, the existing model also substantially out-performs tf-idf for our application of interest.

5.4.4 Timing

Timing performance is presented in Figure 5.13. Average filter update time over the 1,000 km data set, including the geometric check, was 14 ms. The time quoted was measured on a single core of a 2.40 GHZ Intel Core 2 processor. SURF feature extraction and kd-tree quantization adds an overhead of 484 ms on average, with typical variance illustrated in Figure 5.12. The cost is dominated by 423 ms for SURF. Recent GPU-based implementations can largely eliminate this overhead [28]. However, even including feature detection, our real time requirement of 480 ms could be achieved by simply spreading the processing over two cores.



(a) SURF generation time.



(b) Quantization time using the kd-tree.

Figure 5.12: Bag-of-words generation time (per Ladybug2 panoramic image) for a representative sample of the 1,000 km data set, using the 100k vocabulary. The time is dominated by SURF generation, (a), which takes 423 ms on average. Quantization using the randomized kd-trees, (b), takes on average 60 ms.

5.4.5 Comparison to Original System

In comparison to the original system described in Chapter 3, the inference times of the system described here are on average 4,400 times faster, with comparable precision-recall performance. Equally important, the sparse representation means that location models now require only $O(1)$ memory, as opposed to $O(\#vocabulary)$. For the 100k vocabulary, a typical sparse location model requires 4 KB of memory as opposed to 400 KB previously. This enables the use of large vocabularies which improve performance, and is crucial for scalability because the size of the mappable area is effectively limited by available RAM.

5.5 Chapter Summary

This chapter has outlined a new, highly scalable architecture for appearance-only SLAM. We have defined a new model that permits efficient inverted index implementation, while preserving the key benefits of our original Bayesian approach to the problem. The framework is fully probabilistic, and deals with challenging issues such as perceptual aliasing and new place detection. In addition to these benefits, as a pure ranking function it has been shown to considerably out-perform the baseline tf-idf

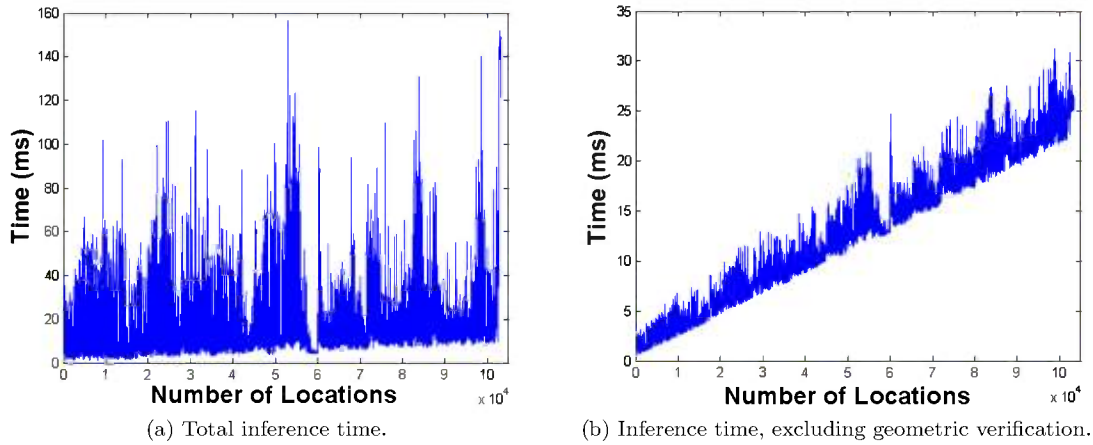
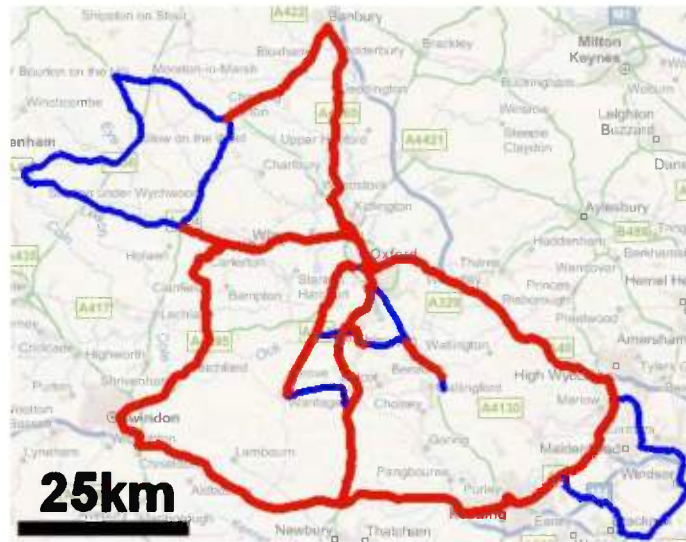


Figure 5.13: Filter update times on the 1,000 km data set for the 100k vocabulary, (a). Mean filter update time is 14 ms and maximum update time is 157 ms. The cost is dominated by the RANSAC geometric verification, which has $O(1)$ complexity. The core ranking stage excluding RANSAC, (b), exhibits linear complexity but with a very small constant - taking 25 ms on average with 100,000 locations in the map.

approach. The chapter also discussed techniques necessary for visual vocabulary generation and Chow Liu tree learning at large scale. On the issue of vocabulary learning, we have demonstrated the benefit of good cluster centre initialization on overall performance. Finally, we have evaluated the system on two substantial data sets, of 70 km and 1,000 km. Both experiments are larger than any existing result we are aware of. Our approach shows very strong performance on the 70 km experiment, in conditions of challenging perceptual aliasing. For practical purposes this set can be considered solved, and moreover this performance can be achieved on the basis of single images, without temporal information. The 1,000 km experiment is more challenging, and we do not consider it fully solved, nevertheless our system's performance is already sufficient to provide a useful competency for an autonomous vehicle operating at this scale. Our data sets are available to the research community, and we hope that they will serve as a benchmark for future systems.



(a)



(b)

Figure 5.14: Loop closure maps for the 1,000 km data set. Sections of the trajectory where loop closures exist are shown in red. (a) The ground truth. (b) Loop closures detected by FAB-MAP (100k CL), showing 99.8% precision and 5.7% recall. There are 2,819 correct loop closures and six false positives. False positives are marked with a green line between poses, however the six present here are spatially close, so are not readily visible on the map. The long section on the right with no detected loop closures is a motorway at dusk. The section on the bottom left with intermittent loop closures is also a motorway.



(a)



(b)

Figure 5.15: Loop closure maps for the 70 km data set. Sections of the trajectory where loop closures exist are shown in red. (a) The ground truth. (b) Detected loop closures using FAB-MAP (100k CL), at 100% precision. The recall rate is 48.4%. However, the system clearly detects loop closures in almost all parts of the trajectory. The recall rate reflects the fact that not every possible image along the trajectory is matched. Two short sections of the trajectory generate fewer loop closures – one is in a forest, where imaging conditions were poor, the other is in open fields, with few visual landmarks. A total of 2,300 loop closures are detected, with no false positives.



(a) At the confidence value of this match, the precision is 99.6%.



(b) At the confidence value of this match, the precision is 100%.



Figure 5.16: Some correct loop closures from the 1,000 km data set. The system typically finds correct matches in the presence of considerable scene change when the image content is distinctive.



(a) At the confidence value of this match, the precision is 100%.



(b) At the confidence value of this match, the precision is 99.9%.



Figure 5.17: Some correct loop closures from the 70km data set. These are not unusual matches. The system typically finds correct matches in the presence of considerable scene change when the image content is distinctive.



(a)



(b)



Figure 5.18: The two highest confidence false positives in the 1,000km data set. Both matches are assigned probabilities very close to 1. In (a), we pass a similar-looking roundabout. The locations are 1km apart. In (b), we encounter the same van twice. The locations are 9km apart. Such rare-but-repetitive objects represent the most challenging class of perceptual aliasing.



(a)



(b)



(c)



(d)

Figure 5.19: Some examples of perceptual aliasing correctly handled by the system. In (a), the locations are 4 km apart. Their highly similar appearance is typical of motorway driving. However, this similarity does not lead to a false positive loop closure detection because this repetitive aspect of the environment has been captured in the sampling set used to evaluate the partition function $p(Z_k|Z^{k-1})$ (see Section 3.2.4). This allows the system to assign the newly collected image a “new place” probability of 0.9997. A similar case from the 70 km set is shown in (b).



(a) First image captured at the location.



(b) Image collected at loop closure.



(c) A second, unrelated location in the map, with very similar appearance.

Figure 5.20: A typical false negative. Figures (a) and (b) come from the same location, but the loop closure is not detected by FAB-MAP. Figure (c) shows a second, unrelated location, to illustrate the self-similar character of the route. The 1,000 km sequence contains hundreds of kilometers of such motorway scenes, so the system's inability to correctly identify this loop closure is unsurprising. This effect depresses the recall in the 1,000 km results. However, the strong perceptual aliasing generates very few false positive detections. In the above case, the new observation is assigned to a new place with probability 0.9994. This is possible because these common modes of perceptual aliasing are easily captured by the sampling set.

Chapter 6

Discussion

This chapter summarizes the contribution of the thesis and discusses possible directions for future research.

6.1 Summary of Contribution

This thesis has made the following contributions to the state of the art:

1. **The first probabilistic framework for localization and mapping using appearance that handles perceptual aliasing (Chapter 3).** We have introduced a natural model for the appearance-based navigation task, which empirically achieves excellent performance for loop closure detection. In addition it provides probabilities for all loop closure assignments, easing integration with metric SLAM systems. Our model is also applicable to some types of image retrieval task. We have compared it to the standard tf-idf ranking function and show superior performance for place recognition.
2. **A novel approach to dealing with correlations between visual words (Chapter 3).** To our knowledge, this is the first time an explicit approximation to the joint distribution of visual word occurrences has been considered in the context of large-scale image retrieval. The technique

we have introduced for modelling word dependencies scales even to large visual vocabularies, and we have demonstrated that it improves performance.

3. **A principled method for deciding when to initialize a new location in the map (Chapter 3).** By computing an approximation of the partition function in our model, we naturally obtain the probability that the current observation comes from a new location not already in the map. To our knowledge this is the first method that goes beyond simple heuristics such as fixed similarity thresholds. We find that our approach is a dramatic improvement over these simpler techniques, particularly in that it handles challenging cases such as perceptual aliasing without any special logic.
4. **An approximate inference technique based on concentration inequalities (Chapter 4).** We show that in return for a small user-controlled probability of generating an incorrect result, inference in our model can be greatly accelerated. Speed improvements in the range 25x-50x were typical for our data, with minimal degradation in accuracy. The approximate inference technique is general in its applicability and should find use beyond the specific problem considered in this thesis.
5. **An efficient version of our model suitable for use with an inverted index (Chapter 5).** By modifying the model to allow an implementation based on inverted indices, we enable the use of our technique on longer trajectories and with larger visual vocabulary sizes. The final system is capable of real time loop closure detection on trajectories over 1,000 km in length.
6. **Four public benchmark data sets (Chapter 5).** This includes a 1,000 km data set which is the most extensive benchmark available. The data set contains 20 Hz stereo imagery, omni-directional imagery every 5-10 m and GPS ground truth.

6.2 Future Work

There are many directions in which the system could be developed. Some interesting possibilities include:

1. **Making greater use of metric information.** In this thesis we have chosen not to use metric information in order to concentrate on what can be achieved with appearance. In a practical system, it seems likely that performance could be improved considerably by replacing our simple topological motion model with one based on real vehicle motion data, for example from a visual SLAM system. While metric position estimates may not be reliably globally (the main motivation for this thesis), they are typically highly accurate locally. This relative motion information is precisely what is required for our position prior. Several authors have recently begun to explore systems which couple metric and appearance information more tightly; for example see [42, 11, 99, 115, 91, 110].
2. **Maintain a distribution over topologies.** As discussed in Chapter 3, the data association step of our method is error-prone. The solution we proposed is effective but not ideal. A better solution might be to maintain a pdf over topologies, as proposed in [128]. An alternative approach is the solution proposed by Olson [115], which selects a single consistent set of loop closures from a larger set of noisy candidates by jointly considering appearance similarity and relative metric information.
3. **Use of depth information.** Often a robot is equipped with either a stereo camera or a laser range finder, which provides depth information. Unlike the 2D geometry of interest points in an image, their relative 3D geometry in space is an invariant property of a location. It should be possible to integrate this relative 3D distance information directly into the ranking function, rather than using 2D geometry in post-verification step. Some initial results by colleagues in our lab suggest that this can significantly improve system performance.
4. **Learning more of the parameters of our generative model.** In this thesis we have used a user-specified constant for $p(z_i|e_i)$, the detector model. In principle this term could instead be learned, and indeed it could be learned as a conditional probability so as to incorporate additional information. For example, certain features are more trustworthy than others — we might not expect words that correspond to parts of dynamic objects such cars or pedestrians to be re-observed at loop closure — and this is naturally captured in a $p(z_i|e_i)$ term which is not constant

across the vocabulary. Lower level properties of an interest point, such its saliency score or vertical height within an image might also be predictive of stability. However, it may be that the general noise in interest point detection dominates these effects. If so, learning might only be worthwhile in combination with a tracking stage to select stable interest points, as was described in [144]. There are other effects that could also be captured by control of $p(z_i|e_i)$, for example if an image is blurry or poorly exposed then $p(z_i|e_i)$ could be dynamically adjusted so as to capture the fact that we expect to see fewer visual words than in a high quality image of the location.

5. **Robustness to rare-but-repetitive features.** It seems likely that the most persistent challenge for appearance-based systems will be dealing with cases similar to those shown in Figure 5.18, where the robot encounters some rare-but-repetitive environmental feature such as a brand logo, advertising billboard, etc. It does not seem feasible to learn a-priori that these are repetitive features, especially in the case of advertising billboards which change regularly. Possible solutions include relying more heavily on sequences of images, detecting false loop closures on the basis of the metric map, or a classifier-based approach to recognising image regions that are likely to contain such untrustworthy features.
6. **Learning stronger approximations to the visual word distribution.** As mentioned in Chapter 3, there is nothing that requires us to use a Chow Liu tree to approximate the joint density over visual words, and it would be interesting to compare performance to alternative learning algorithms. We suspect that the performance improvements that can be gained in this way are modest, given the relative performance of the Naive Bayes and Chow Liu approximations. However, this is only a weak intuition and it would be illuminating to examine the issue thoroughly. In particular, mixture models such as Latent Dirichlet Allocation [12] have quite different representational strengths to the approach we have used, and a comparison would be interesting.
7. **Generalization to non-binary word occurrence.** Our model discards information on the number of visual words present in an image. We chose this simplified representation in order to make the problem of learning correlations between words more tractable. However, a generalization to case of non-binary word occurrence would be interesting and likely offer some performance

improvement. A suitable model, though for the Naive Bayes case only, was described in [128].

8. **Online learning.** At present our visual vocabulary and associated probabilistic model is learned once, and remains static thereafter. It should be possible to update both the visual vocabulary and the model online. Updating the marginal and conditional probabilities that define the probabilistic model is straightforward, though care needs to be taken that the observations used are independent. Incrementally adding new visual words to the vocabulary was explored in [42, 2]. Updating the structure of the Chow Liu tree is efficient for the case of addition of new visual words to the vocabulary [147], but not for the case in which the estimate of the mutual information between existing words changes [133]. Thus relearning the Chow Liu tree might best be achieved as a periodic batch process. We note, however, that our experience with the system's generalization performance suggests that online learning is not a critical issue.

6.3 Concluding Remark

The appearance-based approach to loop closure detection has proven to be remarkably successful. It now seems firmly established as an essential component of SLAM systems. This thesis set out to provide a coherent probabilistic framework in which to treat appearance data. We hope that our approach will be useful for improving the robustness of appearance-based systems and provide a natural way in which to integrate them with metric SLAM techniques.

Appendix A

Data Collection Platforms

This appendix includes details of the vehicles used to collect the data used in this thesis.

The data sets used in Chapters 3 and 4 were collected with a modified iRobot ATRV (*“Marge”*), shown in Figure A.1. The camera used is a Marlin unit from Allied Vision Technologies, mounted on pan-tilt mechanism from Directed Perception.

The 70 km and 1,000 km data sets used in Chapter 5 were collected with a car-mounted sensor rig shown in Figure A.2. The rig consists of a Ladybug2 omni-directional camera and Bumblebee stereo camera, both from Point Grey Research. The cameras were mounted approximately three meters above the road surface. GPS data was collected with a Seres unit from CSI Wireless, mounted on the roof of the car.



Figure A.1: Robot (*"Marge"*) used to collect the data sets used in Chapters 3 and 4.



Figure A.2: Vehicle and sensor rig used to capture the 70 km and 1,000 km data sets used in Chapter 5.

Appendix B

Chow Liu Trees

This appendix illustrates the form and performance of the Chow Liu tree as a general density estimator. For the purposes of explanation we use a system with eight binary features, thus 256 possible distinct observations. Features one to six correspond to the faces of a die. For each observation, the die is rolled and the feature corresponding to the shown face is assigned value 1, the others are assigned value 0. Features seven and eight are generated in a similar way by tossing a coin.

Figure B.1 shows the graphical model for this system, and the ground truth probability of each of the possible 256 observations (Many of these have probability 0 since they correspond to impossible states, such as observing the two faces of the coin simultaneously.)

1000 random samples were generated from this system and used to construct approximations to the pdf. The Chow Liu approximation is shown in Figure B.2. The naive Bayes approximation is shown in Figure B.3.

The three distributions are compared in Figure B.4. The Chow Liu approximation is visibly closer to the true distribution than the naive Bayes approximation. The Kullback-Leibler divergence from the naive approx to the ground truth is 1.61, whereas the Chow Liu approximation has a KL divergence of 0.748.

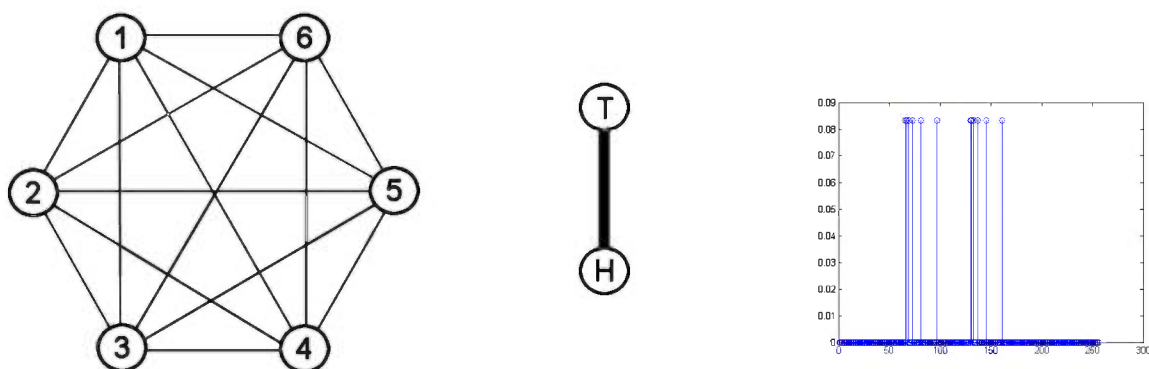


Figure B.1: The underlying system, with its graphical model (left), and the probability of each of the 256 possible observations (right). The thickness of the line in the graphical model corresponds to the mutual information between the variables. There is 1 bit of mutual information between the sides of the coin, and approximately 0.05 bits of mutual information between the faces of the die. The die and coin are independent so share no mutual information.

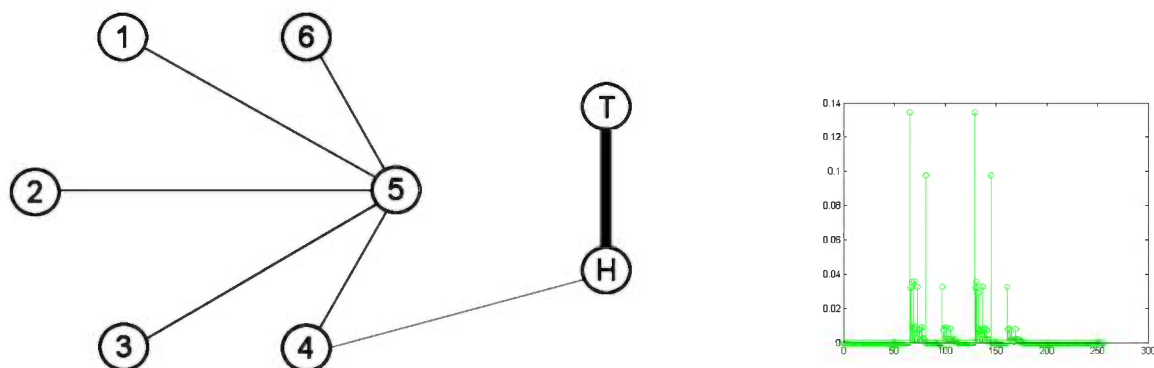


Figure B.2: Chow Liu tree estimated from 1000 samples. Note that the many impossible observations are correctly assigned very low probability. The spurious edge in the graphical model between 4 and H is due to the fact that mutual information between these variables, as estimated from 1000 samples, is not exactly zero. However, the variables have been estimated to be close to independence – the conditional probability $p(H|4) \approx P(H)$, so the effect of this spurious edge in the model is minimal.

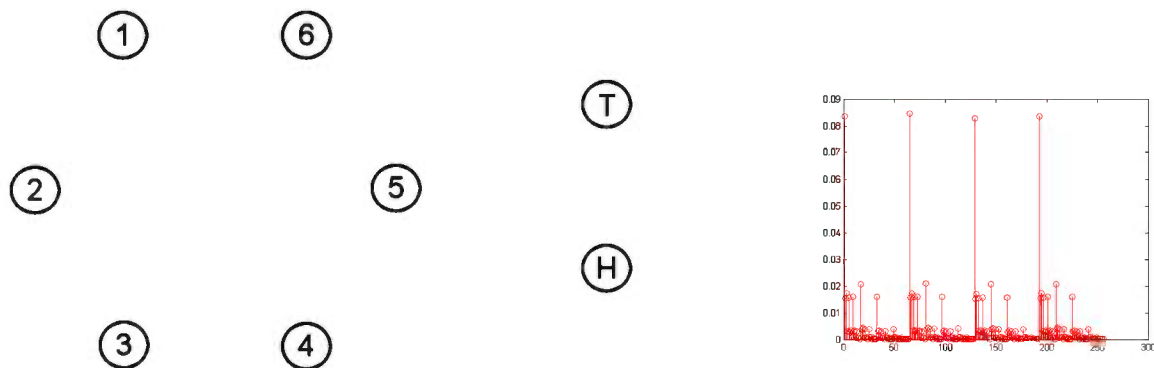


Figure B.3: Naive Bayes. All variables assumed independent. This assumption is far from true in this system, so many impossible observations are estimated to have significant probability (right).

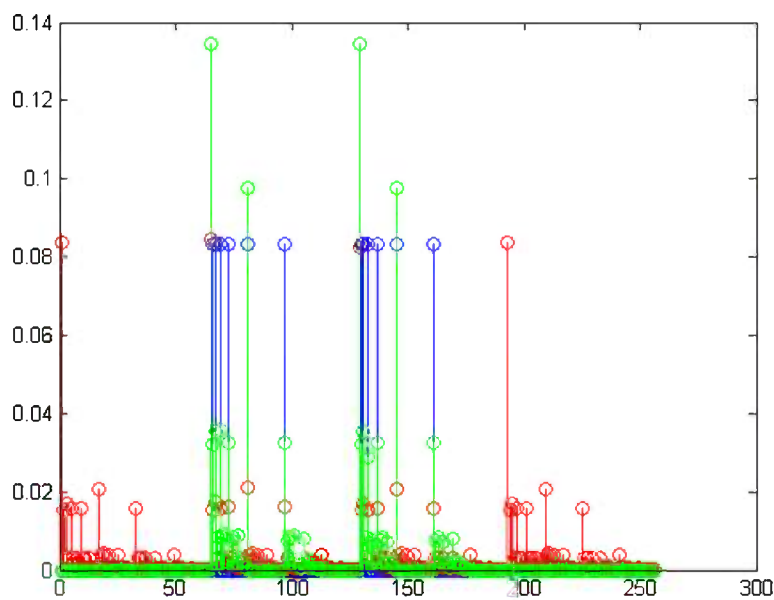


Figure B.4: Comparison of the three distributions – ground truth (blue), Chow Liu (green) and naive Bayes (red). The KL divergence from the naive approx to the ground truth is 1.61, whereas the Chow Liu approximation has a KL divergence of 0.748.

Appendix C

Manipulations

C.1 FAB-MAP Model

This appendix presents the manipulations involved in the expansion of Equation 3.16 from Section 3.2.4. For compactness of notation, in this appendix $z_q = s_{z_q}$ will be denoted z_q and $z_q = \overline{s_{z_q}}$ by $\overline{z_q}$, etc.

We seek to express the term $p(z_q|e_q, z_p)$ as a function of the known quantities $p(z_q)$, $p(z_q|e_q)$, $p(z_q|z_p)$.

Applying Bayes Rule:

$$p(z_q|e_q, z_p) = \frac{p(e_q|z_q, z_p)p(z_q|z_p)}{p(e_q|z_p)}$$

now expanding $p(e_q|z_p)$ as

$$p(e_q|z_p) = p(e_q|z_q, z_p)p(z_q|z_p) + p(e_q|\overline{z_q}, z_p)p(\overline{z_q}|z_p)$$

and making the approximation $p(e_q|z_q, z_p) \approx p(e_q|z_q)$, which is closely related to the approximation we make between Equations 3.14 and 3.15, the expression becomes

$$\begin{aligned}
p(z_q|e_q, z_p) &\approx \frac{p(e_q|z_q)p(z_q|z_p)}{p(e_q|z_q)p(z_q|z_p) + p(e_q|\bar{z}_q)p(\bar{z}_q|z_p)} \\
&= \left(1 + \frac{p(e_q|\bar{z}_q)p(\bar{z}_q|z_p)}{p(e_q|z_q)p(z_q|z_p)}\right)^{-1}
\end{aligned}$$

Now

$$p(e_q|z_q) = \frac{p(z_q|e_q)p(e_q)}{p(z_q)}$$

and similarly for $p(e_q|\bar{z}_q)$, so

$$\frac{p(e_q|\bar{z}_q)}{p(e_q|z_q)} = \frac{p(\bar{z}_q|e_q)p(z_q)}{p(z_q|e_q)p(\bar{z}_q)}$$

substituting back yields

$$p(z_q|e_q, z_p) \approx \left(1 + \frac{\alpha}{\beta}\right)^{-1}$$

where

$$\alpha = p(z_q)p(\bar{z}_q|e_q)p(\bar{z}_q|z_p)$$

$$\beta = p(\bar{z}_q)p(z_q|e_q)p(z_q|z_p)$$

C.2 Bennett Bound

The form of the Bennett inequality used in Chapter 4 differs notationally from the usual statement of the inequality. For completeness, the relevant manipulations are presented here.

Beginning with the specialized form of the Bennett inequality that applies to sums of random variables with symmetric distributions, as per [9]¹:

¹Note that the typesetting in [9] is slightly confusing, as the bracketing might seem to indicate that the argument of the exponential is $\left(\frac{\sigma}{M}\right)^2$. However, the argument is in fact all following terms, i.e. $\left(\frac{\sigma}{M}\right)^2 \left(\cosh(cM) - 1 - cM \frac{M}{\sigma} t\right)$.

$$p(S \geq t\sigma) < \exp\left(\left(\frac{\sigma}{M}\right)^2 \left(\cosh(cM) - 1 - cM \frac{M}{\sigma} t\right)\right)$$

with

$$t = \frac{\sigma}{M} \sinh(cM)$$

Our statement of the inequality simply changes some variables for a more convenient expression.

Define:

$$\Delta = t\sigma$$

so

$$c = \frac{\sinh^{-1}(\frac{\Delta M}{\sigma^2})}{M}$$

and define:

$$f(\Delta) = \sinh^{-1}\left(\frac{\Delta M}{\sigma^2}\right)$$

Substitution yields

$$p(s \geq \Delta) < \exp\left(\left(\frac{\sigma}{M}\right)^2 \left(\cosh(f(\Delta)) - 1 - \frac{M\Delta}{\sigma^2} f(\Delta)\right)\right)$$

Bibliography

- [1] P. Abbeel, D. Koller, and A. Ng. Learning factor graphs in polynomial time and sample complexity. *The Journal of Machine Learning Research*, 7:1743–1788, 2006.
- [2] A. Angeli, D. Filliat, S. Doncieux, and J.A. Meyer. A Fast and Incremental Method for Loop-Closure Detection Using Bags of Visual Words. *IEEE Transactions On Robotics, Special Issue on Visual SLAM*, 24(5):1027–1037, 2008.
- [3] T. Bailey and H. Durrant-Whyte. Simultaneous localization and mapping (SLAM): Part II. *IEEE Robotics & Automation Magazine*, 13(3):108–117, 2006.
- [4] Y. Bar-Shalom, X.R. Li, and T. Kirubarajan. *Estimation with applications to tracking and navigation*. Wiley-Interscience, 2001.
- [5] C.W. Baum and V. Veeravalli. A sequential procedure for multihypothesis testing. *IEEE Transactions on Information Theory*, 40(6):1994–2007, 1994.
- [6] H. Bay, A. Ess, T. Tuytelaars, and L. Van Gool. Speeded-up robust features (SURF). *Computer Vision and Image Understanding*, 110(3):346–359, 2008.
- [7] H. Bay, T. Tuytelaars, and L. Van Gool. SURF: Speeded Up Robust Features. In *Proc 9th European Conf on Computer Vision*, volume 13, pages 404–417, Graz, Austria, May 7 2006.
- [8] J. Beis and D. Lowe. Shape indexing using approximate nearest-neighbour search in high-dimensional spaces. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 1997.

- [9] G. Bennett. Probability inequalities for the sum of independent random variables. *Journal of the American Statistical Association*, 57:33–45, March 1962.
- [10] Y.M. Bishop, S.E. Fienberg, and P.W. Holland. *Discrete Multivariate Analysis: Theory and Practice*. The MIT Press, June 1977.
- [11] J.-L. Blanco, J.-A. Fernandez-Madrigal, , and J. Gonzalez. Towards a unified bayesian approach to hybrid metric-topological SLAM. *IEEE Transactions on Robotics*, 24:259–270, 2008.
- [12] D.M. Blei, A.Y. Ng, and M.I. Jordan. Latent dirichlet allocation. *The Journal of Machine Learning Research*, 3:993–1022, 2003.
- [13] M. Bosse, P. Newman, J. J. Leonard, and S. Teller. SLAM in large-scale cyclic environments using the Atlas framework. *International Journal of Robotics Research*, 23:1113–1139, 2004.
- [14] M. Bosse and R. Zlot. Keypoint design and evaluation for place recognition in 2D lidar maps. In *Robotics: Science and Systems Conference : Inside Data Association Workshop*, 2008.
- [15] M. Bosse and R. Zlot. Map matching and data association for large-scale two-dimensional laser scan-based SLAM. *International Journal of Robotics Research*, 2008.
- [16] S. Boucheron, G. Lugosi, and O. Bousquet. *Concentration Inequalities*, volume Lecture Notes in Artificial Intelligence 3176, pages 208–240. Springer, Heidelberg, Germany, 2004.
- [17] M. Bowling, D. Wilkinson, and A. Ghodsi. Subjective mapping. In *Twenty-First National Conference on Artificial Intelligence*, Boston, Massachusetts, USA, July 2006. AAAI Press.
- [18] M. Bowling, D. Wilkinson, A. Ghodsi, and A. Milstein. Subjective localization with action respecting embedding. In *Proceedings of the International Symposium of Robotics Research*, 2005.
- [19] S. Brin and L. Page. The anatomy of a large-scale hypertextual Web search engine. *Computer Networks and ISDN Systems*, 30, 1998.
- [20] J. Callmer, K. Granstrom, J. Nieto, and F. Ramos. Tree of Words for Visual Loop Closure Detection in Urban SLAM. In *Proceedings of the Australasian Conference on Robotics and Automation*, 2008.

- [21] J.A. Castellanos, R. Martinez-Cantin, J.D. Tardos, and J. Neira. Robocentric map joining: Improving the consistency of EKF-SLAM. *Robotics and Autonomous Systems*, 55(1):21–29, 2007.
- [22] A. Checheta, C. Guestrin, JC Platt, D. Koller, Y. Singer, and S. Roweis. Efficient principled learning of thin junction trees. *Advances in Neural Information Processing Systems*, 20:273–280, 2007.
- [23] H. Choset and K. Nagatani. Topological simultaneous localization and mapping (SLAM): Toward exact localization without explicit localization. *IEEE Transactions on Robotics and Automation*, 17(2):125 – 137, April 2001.
- [24] C.K. Chow and C.N. Liu. Approximating discrete probability distributions with dependence trees. *IEEE Transactions on Information Theory*, IT-14(3), May 1968.
- [25] O. Chum, M. Perdoch, and J. Matas. Geometric min-Hashing: Finding a (Thick) Needle in a Haystack. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR*, 2009.
- [26] O. Chum, J. Philbin, J. Sivic, M. Isard, and A. Zisserman. Total recall: Automatic query expansion with a generative feature model for object retrieval. In *International Conference on Computer Vision*, 2007.
- [27] Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, and Clifford Stein. *Introduction to Algorithms, Second Edition*. The MIT Press, September 2001.
- [28] N. Cornelis and L. Van Gool. Fast scale invariant feature detection and matching on programmable graphics hardware. In *CVPR 2008 Workshop CVGPU*, 2008.
- [29] D. Crisan and A. Doucet. A survey of convergence results on particle filtering methods for practitioners. *IEEE Transactions on signal processing*, 50(3):736–746, 2002.
- [30] M. Cummins and P. Newman. Probabilistic appearance based navigation and loop closing. In *Proc. IEEE International Conference on Robotics and Automation (ICRA’07)*, Rome, April 2007.
- [31] M. Cummins and P. Newman. Accelerated appearance-only SLAM. In *Proc. IEEE International Conference on Robotics and Automation (ICRA’08)*, Pasadena, California, April 2008.

- [32] M. Cummins and P. Newman. FAB-MAP: Probabilistic Localization and Mapping in the Space of Appearance. *The International Journal of Robotics Research*, 27(6):647–665, 2008.
- [33] M. Cummins and P. Newman. Highly scalable appearance-only SLAM - FAB-MAP 2.0. In *Proceedings of Robotics: Science and Systems*, Seattle, USA, June 2009.
- [34] J. Cussens. Bayes and Pseudo-Bayes Estimates of Conditional Probabilities and Their Reliability. *Lecture Notes in Computer Science*, pages 136–136, 1993.
- [35] S. Dasgupta and P.M. Long. Performance guarantees for hierarchical clustering. *Journal of Computer and System Sciences*, 70(4):555–569, 2005.
- [36] A. J. Davison. Real-time simultaneous localisation and mapping with a single camera. In *International Conference on Computer Vision*, Nice France, October 2003.
- [37] F. Dellaert. Square root SAM: Simultaneous location and mapping via square root information smoothing. In *Proceedings of Robotics: Science and Systems*, Cambridge, USA, June 2005.
- [38] R. Dementiev, L. Kettner, and P. Sanders. STXXL: Standard template library for XXL data sets. *Software: Practice and Experience*, 38(6), 2008.
- [39] R. Dementiev, P. Sanders, D. Schultes, and J. Sibeyn. Engineering an external memory minimum spanning tree algorithm. In *Proceedings 3rd International Conference on Theoretical Computer Science*, 2004.
- [40] G. Dudek and D. Jugessur. Robust place recognition using local appearance based methods. In *Proceedings of IEEE International Conference on Robotics and Automation*, volume 2, 2000.
- [41] H. Durrant-Whyte and T. Bailey. Simultaneous localization and mapping: Part I. *IEEE Robotics & Automation Magazine*, 13(2):99–110, 2006.
- [42] E. Eade and T. Drummond. Unified loop closing and recovery for real time monocular slam. In *Proc. 19th British Machine Vision Conference*, 2008.

- [43] G. Elidan and S. Gould. Learning Bounded Treewidth Bayesian Networks. *Journal of Machine Learning Research*, 9:2699–2731, 2008.
- [44] F. Ferreira, V. Santos, and J. Dias. Integration of Multiple Sensors using Binary Features in a Bernoulli Mixture Model. *IEEE International Conference on Multisensor Fusion and Integration for Intelligent Systems*, pages 104–109, 2006.
- [45] D. Filliat. A visual bag of words method for interactive qualitative localization and mapping. In *IEEE International Conference on Robotics and Automation*, pages 3921–3926, 2007.
- [46] M.A. Fischler and R.C. Bolles. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Commun. ACM*, 24(6):381–395, 1981.
- [47] F. Fraundorfer, C. Engels, and D. Nistér. Topological mapping, localization and navigation using image collections. In *International Conference on Intelligent Robots and Systems*, 2007.
- [48] F. Fraundorfer, H. Stewenius, and D. Nister. A binning scheme for fast hard drive based image search. In *IEEE Conference on Computer Vision and Pattern Recognition, 2007. CVPR'07*, pages 1–6, 2007.
- [49] U. Frese, P. Larsson, and T. Duckett. A multilevel relaxation algorithm for simultaneous localisation and mapping. *IEEE Transactions on Robotics*, 21(2):196–207, April 2005.
- [50] N. Friedman, D. Geiger, and M. Goldszmidt. Bayesian Network Classifiers. *Machine Learning*, 29(2):131–163, 1997.
- [51] A. Frome, D. Huber, R. Kolluri, T. Bulow, and J. Malik. Recognizing objects in range data using regional point descriptors. In *Proc. European Conf. on Computer Vision*. Springer, 2004.
- [52] T. Goedemé. *Visual navigation*. PhD thesis, Katholieke Universiteit Leuven, December 2006.
- [53] G. Grisetti, S. Grzonka, C. Stachniss, P. Pfaff, and W. Burgard. Efficient estimation of accurate maximum likelihood maps in 3D. In *Proc. of the IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*, 2007.

- [54] G. Grisetti, C. Stachniss, S. Grzonka, and W. Burgard. A tree parameterization for efficiently computing maximum likelihood maps using gradient descent. In *Proc. of Robotics: Science and Systems (RSS)*, 2007.
- [55] J. Gutmann and K. Konolige. Incremental mapping of large cyclic environments. In *Proceedings of the IEEE International Symposium on Computational Intelligence in Robotics and Automation (CIRA)*, pages 318–325, Monterey, California, November 1999.
- [56] C. G. Harris and M. Stephens. A combined corner and edge detector. In *Alvey vision conference*, pages 147–151, 1988.
- [57] R. I. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, ISBN: 0521623049, 2000.
- [58] K. Ho and P. Newman. Combining visual and spatial appearance for loop closure detection. *Proceedings of European Conference on Mobile Robotics*, September 2005.
- [59] K. Ho and P. Newman. Multiple map intersection detection using visual appearance. In *International Conference on Computational Intelligence, Robotics and Autonomous Systems*, Singapore, December 2005.
- [60] K.L. Ho and P. Newman. Detecting loop closure with scene sequences. *International Journal of Computer Vision*, 74(3):261–286, September 2007.
- [61] W. Hoeffding. Probability Inequalities for Sums of Bounded Random Variables. *Journal of the American Statistical Association*, 58(301):13–30, 1963.
- [62] Klaus-U. Höffgen. Learning and robust learning of product distributions. In *COLT '93: Proceedings of the sixth annual conference on Computational learning theory*, pages 77–83, New York, NY, USA, 1993. ACM.
- [63] H. Jégou, M. Douze, and C. Schmid. Hamming embedding and weak geometric consistency for large scale image search. In Andrew Zisserman David Forsyth, Philip Torr, editor, *European Conference on Computer Vision*, volume I of *LNCS*, pages 304–317. Springer, oct 2008.

- [64] H. Jegou, M. Douze, and C. Schmid. Hamming embedding and weak geometry consistency for large scale image search - extended version. Technical report, INRIA, RR 6709, 2008.
- [65] Hervé Jégou, Matthijs Douze, and Cordelia Schmid. On the burstiness of visual elements. In *IEEE Conference on Computer Vision & Pattern Recognition*, June 2009.
- [66] A. Johnson. *Spin-Images: A Representation for 3-D Surface Matching*. PhD thesis, Robotics Institute, Carnegie Mellon University, Pittsburgh, PA, August 1997.
- [67] M.I. Jordan, Z. Ghahramani, T.S. Jaakkola, and L.K. Saul. An introduction to variational methods for graphical models. *Mach. Learn.*, 37(2):183–233, 1999.
- [68] S. Julier and J. Uhlmann. A new extension of the Kalman filter to nonlinear systems. In *Int. Symp. Aerospace/Defense Sensing, Simul. and Controls, Orlando, FL*, 1997.
- [69] F. Jurie and B. Triggs. Creating efficient codebooks for visual recognition. In *Tenth IEEE International Conference on Computer Vision, 2005. ICCV 2005*, volume 1, 2005.
- [70] Michael Kaess, Ananth Ranganathan, and Frank Dellaert. Fast incremental square root information smoothing. In *International Joint Conference on Artificial Intelligence*, 2007. To appear.
- [71] K. Konolige, J. Bowman, J.D. Chen, P. Mihelich, M. Calonder, V. Lepetit, and P. Fua. View-based maps. In *Proceedings of Robotics: Science and Systems (RSS)*, 2009.
- [72] B. Kröse, N.A. Vlassis, R. Bunschoten, and Y. Motomura. A probabilistic model for appearance-based robot localization. *Image and Vision Computing*, 19(6):381–391, 2001.
- [73] B. Kuipers and Y. Byun. *A robot exploration and mapping strategy based on a semantic hierarchy of spatial representations*, pages 47–63. MIT Press, Cambridge, MA, USA, 1993.
- [74] S. Kullback and R.A. Leibler. On information and sufficiency. *The Annals of Mathematical Statistics*, pages 79–86, 1951.

- [75] J. Kurlbaum and U. Frese. A benchmark data set for data association. Technical Report 017-02/2009, University of Bremen, 2009.
- [76] P. Lamon, I. Nourbakhsh, B. Jensen, and R. Siegwart. Deriving and matching image fingerprint sequences for mobile robot localization. In *Proceedings of the IEEE International Conference on Robotics and Automation*, Seoul, Korea, May 21-26 2001.
- [77] B. Leibe, K. Mikolajczyk, and B. Schiele. Efficient clustering and matching for object class recognition. In *British Machine Vision Conference*, volume 2, 2006.
- [78] J.J. Leonard and P.M. Newman. Consistent, convergent, and constant-time SLAM. In *International Joint Conference on Artificial Intelligence*, Acapulco Mexico, August 2003.
- [79] V. Lepetit and P. Fua. Keypoint recognition using randomized trees. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28(9):1465, 2006.
- [80] T. Leung and J. Malik. Representing and recognizing the visual appearance of materials using three-dimensional textons. *International Journal of Computer Vision*, 43(1):29–44, 2001.
- [81] A. Levin and R. Szeliski. Visual odometry and map correlation. *IEEE Conference on Computer Vision and Pattern Recognition*, 01:611–618, 2004.
- [82] F. Li and J. Kosecka. Probabilistic location recognition using reduced feature set. In *IEEE International Conference on Robotics and Automation*, Orlando, Florida, 2006.
- [83] D.G. Lowe. Object recognition from local scale-invariant features. In *Proceedings of the 7th International Conference on Computer Vision*, pages 1150–1157, Kerkyra, 1999.
- [84] D.G. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60(2):91–110, 2004.
- [85] M. Magnusson, H. Andreasson, A. Nuchter, and A.J. Lilienthal. Appearance-Based Place Recognition from 3D Laser Data Using the Normal Distributions Transform. In *IEEE International Conference on Robotics and Automation*, 2009.

- [86] P.C Mahalanobis. On the generalized distance in statistics. In *Proceedings of the National Institute of Sciences of India*, 1936.
- [87] C. Manning, P. Raghavan, and H. Schütze. *Introduction to Information Retrieval*. Cambridge University Press, 2008.
- [88] O. Maron and A.W. Moore. Hoeffding races: Accelerating model selection search for classification and function approximation. In Jack D. Cowan, Gerald Tesauro, and Joshua Alspecter, editors, *Advances in Neural Information Processing Systems*, volume 6, pages 59–66. Morgan Kaufmann Publishers, Inc., 1994.
- [89] J. Matas and O. Chum. Randomized RANSAC with sequential probability ratio test. In *Proc. IEEE International Conference on Computer Vision (ICCV)*, 2005.
- [90] J. Matas, O. Chum, M. Urban, and T. Pajdla. Robust wide baseline stereo from maximally stable extremal regions. In *Proceedings of British Machine Vision Conference*, 2002.
- [91] C. Mei, G. Sibley, M. Cummins, P. Newman, and I. Reid. A constant time efficient stereo slam system. In *Proceedings of the British Machine Vision Conference (BMVC)*, 2009.
- [92] M. Meilă. An accelerated Chow and Liu algorithm: Fitting tree distributions to high-dimensional sparse data. In *ICML '99: Proceedings of the Sixteenth International Conference on Machine Learning*, pages 249–257, San Francisco, CA, USA, 1999. Morgan Kaufmann Publishers Inc.
- [93] M. Meilă and T. Jaakkola. Tractable Bayesian learning of tree belief networks. *Statistics and Computing*, 16(1):77–92, 2006.
- [94] M. Meilă and M.I. Jordan. Learning with mixtures of trees. *The Journal of Machine Learning Research*, 1:1–48, 2001.
- [95] C. Mikolajczyk and C. Schmid. A performance evaluation of local descriptors. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 18-20 June 2003.
- [96] C. Mikolajczyk and C. Schmid. Scale and affine invariant interest point detectors. *International Journal of Computer Vision*, (1):63–86, 2004.

- [97] K. Mikolajczyk, B. Leibe, B. Schiele, M.I. Syst, and G. Darmstadt. Local features for object class recognition. In *Tenth IEEE International Conference on Computer Vision, 2005. ICCV 2005*, volume 2, 2005.
- [98] K. Mikolajczyk, T. Tuytelaars, C. Schmid, A. Zisserman, J. Matas, F. Schaffalitzky, T. Kadir, and L. Van Gool. A comparison of affine region detectors. *International Journal of Computer Vision*, 65(1):43–72, 2005.
- [99] M.J. Milford and G.F. Wyeth. Mapping a Suburb With a Single Camera Using a Biologically Inspired SLAM System. *IEEE Transactions on Robotics*, 24(5):1038–1053, 2008.
- [100] M. Montemerlo, S. Thrun, D. Koller, and B. Wegbreit. FastSLAM: A factored solution to the simultaneous localization and mapping problem. In *Proceedings of the AAAI National Conference on Artificial Intelligence*, Edmonton, Canada, 2002. AAAI.
- [101] Michael Montemerlo, Sebastian Thrun, Daphne Koller, and Ben Wegbreit. FastSLAM 2.0: An improved particle filtering algorithm for simultaneous localization and mapping that provably converges. In *International Joint Conferences on Artificial Intelligence*, pages 1151–1156, 2003.
- [102] A. Moore. A tutorial on kd-trees. Technical Report 209, University of Cambridge, 1991.
- [103] A.I. Mourikis and S.I. Roumeliotis. A multi-state constraint Kalman filter for vision-aided inertial navigation. In *2007 IEEE International Conference on Robotics and Automation*, pages 3565–3572, 2007.
- [104] O.M. Mozos, C. Stachniss, and W. Burgard. Supervised Learning of Places from Range Data using AdaBoost. *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 1730–1735, 2005.
- [105] M. Muja and D. Lowe. Fast approximate nearest neighbors with automatic algorithm configuration. In *International Conference on Computer Vision Theory and Applications*, 2009.
- [106] J. Neira and J. D. Tardós. Data association in stochastic mapping using the joint compatibility test. *IEEE Transactions on Robotics and Automation*, 17(6):890–897, 2001.

- [107] P. Newman, M. Chandran-Ramesh, D. Cole, M. Cummins, A. Harrison, I. Posner, and D. Schroeter. Describing, navigating and recognising urban spaces - building an end-to-end SLAM system. In *Proc. of the Int. Symposium of Robotics Research (ISRR)*, Hiroshima, Japan, November 2007.
- [108] P. Newman and K. Ho. SLAM - Loop Closing with Visually Salient Features. *IEEE International Conference on Robotics and Automation*, 18-22 April 2005.
- [109] P. Newman, G. Sibley, M. Cummins, M. Smith, A. Harrison, C. Mei, and I. Reid. Toward life long mapping. In *14th International Symposium on Robotics Research*, 2009.
- [110] P. Newman, G. Sibley, M. Smith, M. Cummins, A. Harrison, C. Mei, I. Posner, R. Shade, D. Schröter, L. Murphy, W. Churchill, D. Cole, and I. Reid. Navigating, recognising and describing urban spaces with vision and laser. *The International Journal of Robotics Research*, 2009.
- [111] P. M. Newman, D. M. Cole, and K. Ho. Outdoor SLAM using visual appearance and laser ranging. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, Orlando Florida USA, May 2006.
- [112] K. Ni, A. Kannan, A. Criminisi, and J. Winn. Epitomic location recognition. In *IEEE Conference on Computer Vision and Pattern Recognition, (CVPR)*, pages 1–8, 2008.
- [113] D. Nistér. Preemptive RANSAC for live structure and motion estimation. *Machine Vision and Applications*, 16(5):321–329, 2005.
- [114] D. Nistér and H. Stewenius. Scalable recognition with a vocabulary tree. In *Conf. Computer Vision and Pattern Recognition*, volume 2, pages 2161–2168, 2006.
- [115] E. Olson. *Robust and Efficient Robotic Mapping*. PhD thesis, Massachusetts Institute of Technology, June 2008.
- [116] E. Olson, J. Leonard, and S. Teller. Fast iterative alignment of pose graphs with poor initial estimates. In *Proceedings of International Conference on Robotics and Automation*, Orlando, Florida, USA, May 2006.

- [117] L. Page, S. Brin, R. Motwani, and T. Winograd. The PageRank citation ranking: Bringing order to the web. Technical report, Stanford Digital Library Technologies, 1998.
- [118] M.A. Paskin. Thin junction tree filters for simultaneous localization and mapping. Computer Science Division Technical Report CSD-02-1198, University of California, Berkeley, September 2002.
- [119] M.A. Paskin. Thin junction tree filters for simultaneous localization and mapping. In Georg Gottlob and Toby Walsh, editors, *Proceedings of the Eighteenth International Joint Conference on Artificial Intelligence (IJCAI-03)*, pages 1157–1164, San Francisco, CA, 2003. Morgan Kaufmann Publishers.
- [120] L.M. Paz, J.D. Tardos, and J. Neira. Divide and Conquer: EKF SLAM in $O(n)$. *IEEE Transactions on Robotics*, 24(5):1107–1120, 2008.
- [121] M. Perdoch, O. Chum, and J. Matas. Efficient Representation of Local Geometry for Large Scale Object Retrieval. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR*, 2009.
- [122] J. Philbin, O. Chum, M. Isard, J. Sivic, and A. Zisserman. Object retrieval with large vocabularies and fast spatial matching. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2007.
- [123] J. Philbin, O. Chum, M. Isard, J. Sivic, and A. Zisserman. Lost in quantization: Improving particular object retrieval in large scale image databases. In *IEEE Conference on Computer Vision and Pattern Recognition, 2008. CVPR 2008*, pages 1–8, 2008.
- [124] I. Posner, M. Cummins, and P. Newman. Fast probabilistic labeling of city maps. In *Proc. Robotics: Science and Systems*, Zurich, June 2008.
- [125] I. Posner, M. Cummins, and P. Newman. A generative framework for fast urban labeling using spatial and temporal context. *Autonomous Robots*, 26(2-3):153–170, April 2009.

- [126] F.T. Ramos, B. Upcroft, S. Kumar, and H.F. Durrant-Whyte. A Bayesian approach for place recognition. In *IJCAI Workshop on reasoning with Uncertainty in Robotics*, July 2005.
- [127] A. Ranganathan, E. Menegatti, and F. Dellaert. Bayesian inference in the space of topological maps. *IEEE Transactions on Robotics*, 22(1):92–107, 2006.
- [128] Ananth Ranganathan. *Probabilistic Topological Maps*. PhD thesis, Georgia Institute of Technology, 2008.
- [129] Ananth Ranganathan and Frank Dellaert. A Rao-Blackwellized particle filter for topological mapping. In *Proceedings of International Conference on Robotics and Automation*, Orlando, Florida, USA, May 2006.
- [130] M. Ranzato, Y.L. Boureau, and Y. LeCun. Sparse feature learning for deep belief networks. In *Advances in neural information processing systems*, 2007.
- [131] D.B. Reid. An algorithm for tracking multiple targets. *IEEE Transactions on Automatic Control*, 24(6):843–854, 1979.
- [132] E. Remolina and B. Kuipers. Towards a general theory of topological maps. *Artificial Intelligence*, 152:47–104, 2004.
- [133] J. Roure. Incremental learning of tree augmented naive Bayes classifiers. In *Ibero-American Conference on Artificial Intelligence (IBERAMIA)*, 2002.
- [134] S. Rusinkiewicz and M. Levoy. Efficient variants of the ICP algorithm. In *Proc. 3D Digital Imaging and Modeling*, pages 145–152, 2001.
- [135] G. Schindler, M. Brown, and R. Szeliski. City-Scale Location Recognition. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–7, 2007.
- [136] C. Schmid. Constructing models for content-based image retrieval. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 2, 2001.

- [137] C. Schmid and R. Mohr. Local grayvalue invariants for image retrieval. *IEEE transactions on pattern analysis and machine intelligence*, 19(5):530–535, 1997.
- [138] G. Sibley. Relative bundle adjustment. Technical Report 2307/09, University of Oxford, 2009.
- [139] G. Sibley, C. Mei, I. Reid, and P. Newman. Adaptive Relative Bundle Adjustment. In *Robotics Science and Systems*, August 2009.
- [140] C. Silpa-Anan and R. Hartley. Localisation using an image-map. In *Australian Conference on Robotics and Automation*, 2004.
- [141] C. Silpa-Anan and R. Hartley. Visual localization and loop-back detection with a high resolution omnidirectional camera. In *Workshop on Omnidirectional Vision*, 2005.
- [142] C. Silpa-Anan and R. Hartley. Optimised kd-trees for fast image descriptor matching. In *Computer Vision and Pattern Recognition*, 2008.
- [143] Robert Sim and Gregory Dudek. Mobile robot localization from learned landmarks. *Proceedings of the IEEE International Conference on Intelligent Robots and Systems (IROS)*, 1998.
- [144] J. Sivic and A. Zisserman. Video Google: A text retrieval approach to object matching in videos. In *Proceedings of the International Conference on Computer Vision*, Nice, France, October 2003.
- [145] M. Smith, I. Baldwin, W. Churchill, R. Paul, and P. Newman. The new college vision and laser data set. *International Journal for Robotics Research (IJRR)*, 28(5):595–599, May 2009.
- [146] R. Smith, M. Self, and P. Cheeseman. Estimating uncertain spatial relationships in robotics. *Autonomous Robot Vehicles*, pages 167–193, 1990.
- [147] P. Spira and A. Pan. On finding and updating spanning trees and shortest paths. *SIAM Journal on Computing*, 4:375, 1975.
- [148] N. Srebro. Maximum likelihood bounded tree-width markov networks. In *Proceedings of the 17th Annual Conference on Uncertainty in Artificial Intelligence (UAI-01)*, pages 504–51, San Francisco, CA, 2001. Morgan Kaufmann.

- [149] M.J. Swain and D.H. Ballard. Color indexing. *International journal of computer vision*, 7(1):11–32, 1991.
- [150] M. Teyssier and D. Koller. Ordering-based search: A simple and effective algorithm for learning Bayesian networks. In *Proceedings of the Twenty-first Conference on Uncertainty in AI (UAI)*, pages 584–590, 2005.
- [151] A. Torralba, R. Fergus, and Y. Weiss. Small codes and large image databases for recognition. In *IEEE Conference on Computer Vision and Pattern Recognition, 2008. CVPR 2008*, pages 1–8, 2008.
- [152] A. Torralba, K.P. Murphy, W.T. Freeman, and M.A. Rubin. Context-based vision system for place and object recognition. In *ICCV '03: Proceedings of the Ninth IEEE International Conference on Computer Vision*, page 273, Washington, DC, USA, 2003.
- [153] L. Torresani, M. Szummer, and A. Fitzgibbon. Learning query-dependent prefilters for scalable image retrieval. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2009.
- [154] M. Turk and A. Pentland. Face recognition using eigenfaces. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, volume 591, 1991.
- [155] T. Tuytelaars and C. Schmid. Vector quantizing feature space with a regular lattice. In *IEEE 11th International Conference on Computer Vision, 2007. ICCV 2007*, pages 1–8, 2007.
- [156] I. Ulrich and I. Nourbakhsh. Appearance-based place recognition for topological localization. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, volume 2, pages 1023 – 1029, April 2000.
- [157] C. Valgren and A. Lilienthal. SIFT, SURF and Seasons: Long-term outdoor localization using local features. In *Proc. of 3rd European Conference on Mobile Robots, Freiburg, Germany*, 2007.
- [158] A. Wald. *Sequential Analysis*. Dover Publications, New York, 1947.

- [159] M. Walter, R. Eustice, and J. Leonard. A provably consistent method for imposing sparsity in feature-based SLAM information filters. In *Proc. of the Int. Symposium of Robotics Research (ISRR)*, 2005.
- [160] Junqiu Wang, Hongbin Zha, and Roberto Cipolla. Vision-based global localization using a visual vocabulary. In *IEEE International Conference on Robotics and Automation*, 2005.
- [161] Y. Weiss, A. Torralba, and R. Fergus. Spectral hashing. In *Advances in Neural Information Processing Systems*, volume 21, pages 1753–1760, 2009.
- [162] D.B. West et al. *Introduction to graph theory*. Prentice Hall, NJ, 2001.
- [163] B. Williams, M. Cummins, J. Neira, P. Newman, I. Reid, and J. Tardos. An image-to-map loop closing method for monocular SLAM. In *Proc. International Conference on Intelligent Robots and Systems*, 2008.
- [164] B. Williams, M. Cummins, J. Neira, P. Newman, I. Reid, and J. Tardós. A comparison of loop closing techniques in monocular SLAM. *Robotics and Autonomous Systems*, 2009.
- [165] S. Winder and M. Brown. Learning local image descriptors. In *IEEE Conference on Computer Vision and Pattern Recognition, 2007. CVPR'07*, pages 1–8, 2007.
- [166] S. Winder, G. Hua, and M. Brown. Picking the best daisy. In *Proceedings of the International Conference on Computer Vision and Pattern Recognition (CVPR09)*, Miami, June 2009.
- [167] J. Wolf, W. Burgard, and H. Burkhardt. Robust vision-based localization by combining an image-retrieval system with Monte Carlo localization. *IEEE Transactions on Robotics*, 21(2):208–216, 2005.
- [168] Z. Wu, QF Ke, M. Isard, and J. Sun. Bundling features for large-scale partial-duplicate web image search. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2009.
- [169] Z. Zhu, T. Oskiper, S. Samarasekera, R. Kumar, and H.S. Sawhney. Ten-fold improvement in visual odometry using landmark matching. In *Intl. Conf. on Computer Vision (ICCV)*, 2007.

- [170] Z. Zivkovic, B. Bakker, and B. Kröse. Hierarchical map building using visual landmarks and geometric constraints. In *Proc. IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 7–12, 2005.

Attachment I



46. Simultaneous Localization and Mapping

Cyrril Stachniss, John J. Leonard, Sebastian Thrun

This chapter provides a comprehensive introduction in to the *simultaneous localization and mapping problem*, better known in its abbreviated form as SLAM. SLAM addresses the main perception problem of a robot navigating an unknown environment. While navigating the environment, the robot seeks to acquire a map thereof, and at the same time it wishes to localize itself using its map. The use of SLAM problems can be motivated in two different ways: one might be interested in detailed environment models, or one might seek to maintain an accurate sense of a mobile robot's location. SLAM serves both of these purposes.

We review the three major paradigms from which many published methods for SLAM are derived: (1) the extended Kalman filter (EKF); (2) particle filtering; and (3) graph optimization. We also review recent work in three-dimensional (3-D) SLAM using visual and red green blue dis-

46.1 SLAM: Problem Definition	1154
46.1.1 Mathematical Basis	1154
46.1.2 Example: SLAM in Landmark Worlds	1155
46.1.3 Taxonomy of the SLAM Problem ..	1156
46.2 The Three Main SLAM Paradigms	1157
46.2.1 Extended Kalman Filters	1157
46.2.2 Particle Methods	1159
46.2.3 Graph-Based Optimization Techniques	1162
46.2.4 Relation of Paradigms	1166
46.3 Visual and RGB-D SLAM	1166
46.4 Conclusion and Future Challenges	1169
Video-References	1170
References	1171

tance-sensors (RGB-D), and close with a discussion of open research problems in robotic mapping.

This chapter provides a comprehensive introduction into one of the key enabling technologies of mobile robot navigation: *simultaneous localization and mapping*, or in short *SLAM*. SLAM addresses the problem of acquiring a spatial map of an environment while simultaneously localizing the robot relative to this model. The SLAM problem is generally regarded as one of the most important problems in the pursuit of building truly autonomous mobile robots. It is of great practical importance; if a robust, general-purpose solution to SLAM can be found, then many new applications of mobile robotics will become possible.

While the problem is deceptively easy to state, it presents many challenges, despite significant progress made in this area. At present, we have robust methods for mapping environments that are mainly static, struc-

tured, and of limited size. Robustly mapping unstructured, dynamic, and large-scale environments in an on-line fashion remains largely an open research problem.

The historical roots of methods that can be applied to address the SLAM problem can be traced back to Gauss [46.1], who is largely credited for inventing the least squares method. In the Twentieth Century, a number of fields outside robotics have studied the making of environment models from a moving sensor platform, most notably in *photogrammetry* [46.2–4] and *computer vision* [46.5]. Strongly related problems in these fields are bundle adjustment and structure from motion. SLAM builds on this work, often extending the basic paradigms into more scalable algorithms. Modern SLAM systems often view the estimation problem as solving a sparse graph of constraints and applying

nonlinear optimization to compute the map and the trajectory of the robot. As we strive to enable long-lived autonomous robots, an emerging challenge is to handle massive sensor data streams.

This chapter begins with a definition of the SLAM problem, which shall include a brief taxonomy of different versions of the problem. The centerpiece of this chapter is a layman introduction into the three major paradigms in this field, and the various extensions that exist. As the reader will quickly recognize, there is no single best solution to the SLAM method. The method

chosen by the practitioner will depend on a number of factors, such as the desired map resolution, the update time, and the nature of the features in the map, and so on. Nevertheless, the three methods discussed in this chapter cover the major paradigms in this field.

For more a detailed treatment of SLAM, we refer the reader to Durrant-Whyte and Bailey [46.6, 7], who provide an in-depth tutorial for SLAM, Grisetti et al. for a tutorial on graph-based SLAM [46.8], and Thrun et al., which dedicates a number of chapters to the topic of SLAM [46.9].

46.1 SLAM: Problem Definition

The SLAM problem is defined as follows: A mobile robot roams an unknown environment, starting at an initial location x_0 . Its motion is uncertain, making it gradually more difficult to determine its current pose in global coordinates. As it roams, the robot can sense its environment with a noisy sensor. The SLAM problem is the problem of building a map of the environment while simultaneously determining the robot's position relative to this map given noisy data.

46.1.1 Mathematical Basis

Formally, SLAM is best described in probabilistic terminology. Let us denote time by t , and the robot location by x_t . For mobile robots on a flat ground, x_t is usually a three-dimensional vector, comprising its two-dimensional (2-D) coordinate in the plane plus a single rotational value for its orientation. The sequence of locations, or *path*, is then given as

$$X_T = \{x_0, x_1, x_2, \dots, x_T\}. \quad (46.1)$$

Here T is some terminal time (T might be ∞). The initial location x_0 often serves as a point of reference for the estimation algorithm; other positions cannot be sensed.

Odometry provides relative information between two consecutive locations. Let u_t denote the odometry that characterized the motion between time $t-1$ and time t ; such data might be obtained from the robot's wheel encoders or from the controls given to those motors. Then the sequence

$$U_T = \{u_1, u_2, u_3, \dots, u_T\} \quad (46.2)$$

characterizes the relative motion of the robot. For noise-free motion, U_T would be sufficient to recover the poses from the initial location x_0 . However, odometry mea-

surements are noisy, and path integration techniques inevitably diverge from the truth.

Finally, the robot senses objects in the environment. Let m denote the *true* map of the environment. The environment may be comprised of landmarks, objects, surfaces, etc., and m describes their locations. The environment map m is often assumed to be time-invariant, i. e., static.

The robot measurements establish information between features in m and the robot location x_t . If we, without loss of generality, assume that the robot takes exactly one measurement at each point in time, the sequence of measurements is given as

$$Z_T = \{z_1, z_2, z_3, \dots, z_T\}. \quad (46.3)$$

Figure 46.1 illustrates the variables involved in the SLAM problem. It shows the sequence of locations and sensor measurements, and the causal relationships between these variables. This diagram represents a *graph*-

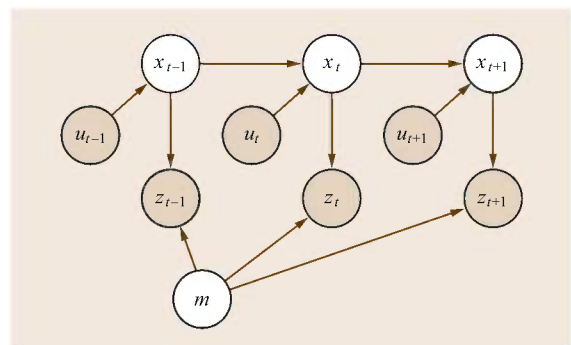


Fig. 46.1 Graphical model of the SLAM problem. Arcs indicate causal relationships, and shaded nodes are directly observable to the robot. In SLAM, the robot seeks to recover the unobservable variables

ical model for SLAM. It is useful in understanding the dependencies in the problem at hand.

The SLAM problem is now the problem of recovering a model of the world m and the sequence of robot locations X_T from the odometry and measurement data. The literature distinguishes two main forms of the SLAM problem, which are both of equal practical importance. One is known as the *full SLAM problem*: it involves estimating the posterior over the entire robot path together with the map

$$p(X_T, m \mid Z_T, U_T). \quad (46.4)$$

Written in this way, the full SLAM problem is the problem of calculating the joint posterior probability over X_T and m from the available data. Notice that the variables right of the conditioning bar are all directly observable to the robot, whereas those on the left are the ones that we want. As we shall see, algorithms for the full SLAM problem are often batch, that is, they process all data at the same time.

The second, equally important SLAM problem is the *online SLAM problem*. This problem is defined via

$$p(x_t, m \mid Z_t, U_t). \quad (46.5)$$

Online SLAM seeks to recover the present robot location, instead of the entire path. Algorithms that address the online problem are usually incremental and can process one data item at a time. In the literature, such algorithms are typically called *filters*.

To *solve* the SLAM problem, the robot needs to be endowed with two more models: a mathematical model that relates odometry measurements u_t to robot locations x_{t-1} and x_t ; and a model that relates measurements z_t to the environment m and the robot location x_t . These models correspond to the arcs in Fig. 46.1.

In SLAM, it is common to think of those mathematical models as probability distributions: $p(x_t \mid x_{t-1}, u_t)$ characterizes the probability distribution of the location x_t assuming that a robot started at a known location x_{t-1} and measured the odometry data u_t . And likewise, $p(z_t \mid x_t, m)$ is the probability for measuring z_t if this measurement is taken at a known location x_t in a known environment m . Of course, in the SLAM problem we do *not* know the robot location, and neither do we know the environment. As we shall see, Bayes rule takes care of this, by transforming these mathematical relationships into a form where we can *recover* probability distributions over those latent variables from the measured data.

46.1.2 Example: SLAM in Landmark Worlds

One common setting of SLAM involves an assumption that the environment is populated by point-landmarks.

When building 2-D maps, point-landmarks may correspond to door posts and corners of rooms, which, when projected into a 2-D map are characterized by a point coordinate. In a 2-D world, each point-landmark is characterized by two coordinate values. Hence the world is a vector of size $2N$, where N is the number of point-landmarks in the world. In a commonly studied setting, the robot can sense three things: the relative range to nearby landmarks, their relative bearing, and the identity of these landmarks. The range and bearing may be noisy, but in the most simple case the identity of the sensed landmarks is known perfectly. Determining the identity of the sensed landmarks is also known as the *data association problem*. In practice, it is one of the most difficult problems in SLAM.

To model the above described setup, one begins with defining the *exact*, noise-free measurement function. The measurement function h describes the workings of the sensors: it accepts as input a description of the environment m and a robot location x_t , and it computes the measurement

$$h(x_t, m). \quad (46.6)$$

Computing h is straightforward in our simplified landmark setting; it is a simple exercise in trigonometry. The probabilistic measurement model can be derived from this measurement function by adding a noise term. It is a probability distribution that peaks at the noise-free value $h(x_t, m)$ but allows for measurement noise, for example,

$$p(z_t \mid x_t, m) = \mathcal{N}(h(x_t, m), \mathbf{Q}_t). \quad (46.7)$$

Here $\mathcal{N}$ denotes the 2-D normal distribution, which is centered at $h(x_t, m)$. The 2-by-2 matrix $\mathbf{Q}_t$ is the noise covariance, indexed by time.

The motion model is derived from a kinematic model of robot motion. Given the location vector x_{t-1} and the motion u_t , textbook kinematics tells us how to calculate x_t . Let this function be denoted by g

$$g(x_{t-1}, u_t). \quad (46.8)$$

The motion model may then be defined by a normal distribution centered at $g(x_{t-1}, u_t)$ but subject to Gaussian noise

$$p(x_t \mid x_{t-1}, u_t) = \mathcal{N}(g(x_{t-1}, u_t), \mathbf{R}_t). \quad (46.9)$$

Here $\mathbf{R}_t$ is a covariance. It is of size 3-by-3, since the location is a three-dimensional 3-D vector.

With these definitions, we have all we need to develop a SLAM algorithm. While in the literature,

point-landmark problems with range-bearing sensing are by far the most studied, SLAM algorithms are not confined to landmark worlds. But no matter what the map representation and the sensor modality, any SLAM algorithm needs a similarly crisp definition of the features in m , the measurement model $p(z_t | x_t, m)$, and the motion model $p(x_t | x_{t-1}, u_t)$. Note that none of those distributions has to be restricted to Gaussian noise as done in the example above.

46.1.3 Taxonomy of the SLAM Problem

SLAM problems are distinguished along a number of different dimensions. Most important research papers identify the type of problems addressed by making the underlying assumptions explicit. We already encountered one such distinction: full versus online. Other common distinctions are as follows:

Volumetric Versus Feature-Based

In volumetric SLAM, the map is sampled at a resolution high enough to allow for photo-realistic reconstruction of the environment. The map m in volumetric SLAM is usually quite high-dimensional, with the result that the computation can be quite involved. Feature-based SLAM extracts sparse features from the sensor stream. The map is then only comprised of features. Our point-landmark example is an instance of feature-based SLAM. Feature-based SLAM techniques tend to be more efficient, but their results may be inferior to volumetric SLAM due to the fact that the extraction of features discards information in the sensor measurements.

Topological Versus Metric

Some mapping techniques recover only a qualitative description of the environment, which characterizes the relation of basic locations. Such methods are known as topological. A topological map might be defined over a set of distinct places and a set of qualitative relations between these places (e.g., place A is adjacent to place B). Metric SLAM methods provide metric information between the relation of such places. In recent years, topological methods have fallen out of fashion, despite ample evidence that humans often use topological information for navigation.

Known Versus Unknown Correspondence

The correspondence problem is the problem of relating the identity of sensed things to other sensed things. In the landmark example above, we assumed that the identity of landmarks is known. Some SLAM algorithms make such an assumption, others do not. The ones that do not provide special mechanisms for estimating the correspondence of measured features to

previously observed landmarks in the map. The problem of estimating the correspondence is known as *data association problem*. It is one of the most difficult problems in SLAM.

Static Versus Dynamic

Static SLAM algorithms assume that the environment does not change over time. Dynamic methods allow for changes in the environment. The vast literature on SLAM assumes static environments. Dynamic effects are often treated just as measurement outliers. Methods that reason about motion in the environment are more involved, but they tend to be more robust in most applications.

Small Versus Large Uncertainty

SLAM problems are distinguished by the degree of location uncertainty that they can handle. The most simple SLAM algorithms allow only for small errors in the location estimate. They are good for situations in which a robot goes down a path that does not intersect itself, and then returns along the same path. In many environments it is possible to reach the same location from multiple directions. Here the robot may accrue a large amount of uncertainty. This problem is known as the *loop closing problem*. When closing a loop, the uncertainty may be large. The ability to close loops is a key characteristic of modern-day SLAM algorithms. The uncertainty can be reduced if the robot can sense information about its position in some absolute coordinate frame, e.g., through the use of a satellite-based global positioning system (GPS) receiver.

Active Versus Passive

In passive SLAM algorithms, some other entity controls the robot, and the SLAM algorithm is purely observing. The vast majority of algorithms are of this type; they give the robot designer the freedom to implement arbitrary motion controllers, and pursue arbitrary motion objectives. In active SLAM, the robot actively explores its environment in the pursuit of an accurate map. Active SLAM methods tend to yield more accurate maps in less time, but they constrain the robot motion. There exist hybrid techniques in which the SLAM algorithm controls only the pointing direction of the robot's sensors, but not the motion direction.

Single-Robot Versus Multi-Robot

Most SLAM problems are defined for a single robot platform, although recently the problem of multi-robot exploration has gained in popularity. Multi-robot SLAM problems come in many flavors. In some, robots get to observe each other, in others, robots are told their relative initial locations. Multirobot SLAM problems

are also distinguished by the type of communication allowed between the different robots. In some, the robots can communicate with no latency and infinite bandwidth. More realistic are setups in which only nearby robots can communicate, and the communication is subject to latency and bandwidth limitations.

Any-Time and Any-Space

Robots that do all computations onboard have limited resources in memory and computation power. Any-time and any-space SLAM systems are an alternative

to traditional methods. They enable the robot to compute a solution given the resource constraints of the system. The more resources available, the better the solution.

As this taxonomy suggests, there exists a flurry of SLAM algorithms. Most modern-day conferences dedicate multiple sessions to SLAM. This chapter focuses on the very basic SLAM setup. In particular it assumes a static environment with a single robot. Extensions are discussed towards the end of this chapter, in which the relevant literature is discussed.

4.6.2 The Three Main SLAM Paradigms

This section reviews three basic SLAM paradigms, from which most others are derived. The first, known as EKF SLAM, is in robotics historically the earliest but has become less popular due to its limiting computational properties and issues resulting from performing single linearizations only. The second approach uses nonparametric statistical filtering techniques known as particle filters. It is a popular method for online SLAM and provides a perspective on addressing the data association problem in SLAM. The third paradigm is based on graphical representations and successfully applies sparse nonlinear optimization methods to the SLAM problem. It is the main paradigm for solving the full SLAM problem and recently also incremental techniques are available.

4.6.2.1 Extended Kalman Filters

Historically, the EKF formulation of SLAM is the earliest, and perhaps the most influential, SLAM algorithm. EKF SLAM was introduced in [46.10, 11] and [46.12, 13], which were the first papers to propose the use of a single state vector to estimate the locations of the robot and a set of features in the environment, with an associated error covariance matrix representing the uncertainty in these estimates, including the correlations between the vehicle and feature state estimates. As the robot moves through its environment taking measurements, the system state vector and covariance matrix are updated using the extended Kalman filter [46.14, 15]. As new features are observed, new states are added to the system state vector; the size of the system covariance matrix grows quadratically.

This approach assumes a metrical, feature-based environmental representation, in which objects can be effectively represented as points in an appropriate parameter space. The position of the robot and the locations of features form a network of uncertain spatial relationships. The development of appropriate repre-

sentations is a critical issue in SLAM, and intimately related to the topics of sensing and world modeling discussed in Chap. 36 and in Part C.

The EKF algorithm represents the robot estimate by a multivariate Gaussian

$$p(x_t, m \mid Z_t, U_t) = \mathcal{N}(\mu_t, \Sigma_t). \quad (46.10)$$

The high-dimensional vector μ_t contains the robot's best estimate of its own current location x_t and the location of the features in the environment. In our point-landmark example, the dimension of μ_t would be $3 + 2N$, since we need three variables to represent the robot location and $2N$ variables for the N landmarks in the map.

The matrix Σ_t is the covariance of the robot's assessment of its expected error in the guess μ_t . The matrix Σ_t is of size $(3 + 2N) \times (3 + 2N)$ and it is positive semi-definite. In SLAM, this matrix is usually dense. The off-diagonal elements capture the correlations in the estimates of different variables. Nonzero correlations come along because the robot's location is uncertain, and as a result the locations of the landmarks in the maps are uncertain.

The EKF SLAM algorithm is easily derived for our point-landmark example. Suppose, for a moment, the motion function g and the measurement function h were *linear* in their arguments. Then, the vanilla Kalman filter, as described in any textbook on Kalman filtering, would be applicable. EKF SLAM linearizes the functions g and h using Taylor series expansion. In its most basic form and in the absence of any data association problems, EKF SLAM is basically the application of the EKF to the online SLAM problem.

Figure 46.2 illustrates the EKF SLAM algorithm for an artificial example. The robot navigates from a start pose that serves as the origin of its coordinate system. As it moves, its own pose uncertainty increases, as indicated by uncertainty ellipses of growing diameter. It

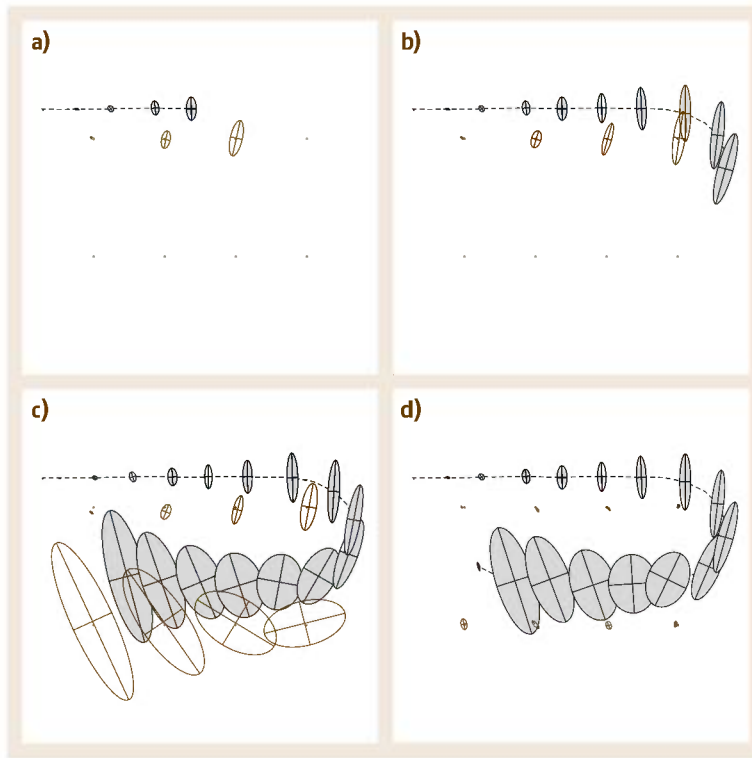


Fig.46.2a–d EKF applied to the online SLAM problem. The robot's path is a *dotted line*, and its estimates of its own position are *shaded ellipses*. Eight distinguishable landmarks of unknown location are shown as *small dots*, and their location estimates are shown as *white ellipses*. In (a–c) the robot's positional uncertainty is increasing, as is its uncertainty about the landmarks it encounters. In (d) the robot senses the first landmark again, and the uncertainty of *all* landmarks decreases, as does the uncertainty of its current pose (image courtesy of Michael Montemerlo, Stanford University)

also senses nearby landmarks and maps them with an uncertainty that combines the fixed measurement uncertainty with the increasing pose uncertainty. As a result, the uncertainty in the landmark locations grows over time. The interesting transition happens in Fig. 46.2d: Here the robot observes the landmark it saw in the very beginning of mapping, and whose location is relatively well known. Through this observation, the robot's pose error is reduced, as indicated in Fig. 46.2d – notice the very small error ellipse for the final robot pose. This observation also reduces the uncertainty for other landmarks in the map. This phenomenon arises from a correlation that is expressed in the covariance matrix of the Gaussian posterior. Since most of the uncertainty in earlier landmark estimates is caused by the robot pose, and since this very uncertainty persists over time, the location estimates of those landmarks are correlated. When gaining information on the robot's pose, this information spreads to previously observed landmarks. This effect is probably the most important characteristic of the SLAM posterior [46.16]. Information that helps localize the robot is propagated through the map, and as a result improves the localization of other landmarks in the map.

With a few adaptations, EKF SLAM can also be applied in the presence of *uncertain data association*. If the identity of observed features is unknown, the basic

EKF idea becomes inapplicable. The solution here is to reason about the most likely data association when a landmark is observed. This is usually done based on proximity: *which of the landmarks in the map corresponds most likely to the landmark just observed?* The proximity calculation considers the measurement noise and the actual uncertainty in the poster estimate, and the metric used in this calculation is known as a Mahalanobis distance, which is a weighted quadratic distance. To minimize the chances of false data associations, many implementations use visible features to distinguish individual landmarks and associate groups of landmarks observed simultaneously [46.17, 18], although distinct features can also be computed from laser data [46.19, 20]. Typical implementations also maintain a *provisional landmark list* and only add landmarks to the internal map when they have been observed sufficiently frequently [46.16, 21]. With an appropriate landmark definition and careful implementation of the data association step, EKF SLAM has been applied successfully in a wide range of environments, using airborne, underwater, indoor, and various other platforms.

The basic formulation of EKF SLAM assumes that the location of features in the map is fully observable from a single position of the robot. The method has been extended to situations with partial observability,

with range-only [46.22] or angle-only [46.23, 24] measurements. The technique has also been utilized using a feature-less representation, in which the state consists of current and past robot poses, and measurements take the form of constraints between the poses (derived for example from laser scan matching or from camera measurements) [46.25, 26].

A key concern of the EKF approach to SLAM lies in the quadratic nature of the covariance matrix. A number of researchers have proposed extensions to the EKF SLAM algorithms that achieve scalability, for example through submap decomposition [46.27–30]. A related family of approaches [46.31–34] employs the Extended Information Filter, which operates on the inverse of the covariance matrix. A key insight is that whereas the EKF covariance is densely populated, the information matrix is sparse when the full robot trajectory is maintained, leading to the development of efficient algorithms and providing a conceptual link to the pose graph optimization methods described in Sect. 46.2.3.

The issues of consistency and convergence in EKF SLAM have been investigated in [46.35, 36]. Observability-based rules for designing consistent EKF SLAM estimators are presented in [46.37].

46.2.2 Particle Methods

The second principal SLAM paradigm is based on particle filters. Particle filters can be traced back to [46.38], but they have become popular only in the last two decades. Particle filters represent a posterior through a set of *particles*. For the novice in SLAM, each particle is best thought as a concrete guess as to what the true value of the state may be. By collecting many such guesses into a set of guesses, or set of particles, the particle filter approximates the posterior distribution. Under mild conditions, the particle filter has been shown to approach the true posterior as the particle set size goes to infinity. It is also a nonparametric representation that represents multimodal distributions with ease.

The key problem with the particle filter in the context of SLAM is that the space of maps and robot paths is huge. Suppose we have a map with 100 features. How many particles would it take to populate that space? In fact, particle filters scale exponentially with the dimension of the underlying state space. Three or four dimensions are thus acceptable, but 100 dimensions are generally not.

The trick to make particle filters amenable to the SLAM problem goes back to [46.39, 40] and is known as Rao–Blackwellization. It has been introduced into the SLAM literature in [46.41], followed by [46.42],

who coined the name fastSLAM (fast simultaneous localization and mapping). Let us first explain the basic FastSLAM algorithm on the simplified point-landmark example, and then discuss the justification for this approach.

At any point in time, FastSLAM maintains K particles of the type

$$X_t^{[k]}, \mu_{t,1}^{[k]}, \dots, \mu_{t,N}^{[k]}, \Sigma_{t,1}^{[k]}, \dots, \Sigma_{t,N}^{[k]}. \quad (46.11)$$

Here $[k]$ is the index of the sample. This expression states that a particle contains:

- A sample path $X_t^{[k]}$, and
- A set of N 2-D Gaussians with means $\mu_{t,n}^{[k]}$ and variances $\Sigma_{t,n}^{[k]}$, one for each landmark in the environment.

Here n is the index of the landmark (with $1 \leq n \leq N$). From that it follows that K particles possess K path samples. It also possesses KN Gaussians, each of which models exactly one landmark for one of the particles.

Initializing FastSLAM is simple: just set each particle's robot location to the starting coordinates, typically $(0, 0, 0)^T$, and zero the map. The particle update then proceeds as follows:

- When an odometry reading is received, new location variables are generated stochastically, one for each of the particles. The distribution for generating those location particles is based on the motion model

$$x_t^{[k]} \approx p(x_t | x_{t-1}^{[k]}, u_t). \quad (46.12)$$

Here $x_{t-1}^{[k]}$ is the previous location, which is part of the particle. This probabilistic sampling step is easily implemented for any robot whose kinematics can be computed.

- When a measurement z_t is received, two things happen: first, FastSLAM computes for each particle the probability of the new measurement z_t . Let the index of the sensed landmark be n . Then the desired probability is defined as follows

$$w_t^{[k]} = \mathcal{N}(z_t | x_t^{[k]}, \mu_{t,n}^{[k]}, \Sigma_{t,n}^{[k]}). \quad (46.13)$$

The factor $w_t^{[k]}$ is called the *importance weight*, since it measures how *important* the particle is in the light of the new sensor measurement. As before, $\mathcal{N}$ denotes the normal distribution, but this time it is calculated for a specific value, z_t . The importance weights of all particles are then normalized so that they sum to 1.

Next, FastSLAM draws with replacement from the set of existing particles a set of new particles. The probability of drawing a particle is its normalized

importance weight. This step is called *resampling*. The intuition behind resampling is that particles for which the measurement is more plausible have a higher chance of surviving the resampling process.

Finally, FastSLAM updates for the new particle set the mean $\mu_{i,n}^{[k]}$ and covariance $\Sigma_{i,n}^{[k]}$, based on the measurement z_i . This update follows the standard EKF update rules – note that the extended Kalman filters maintained in FastSLAM are, in contrast to EKF SLAM, all low-dimensional (typically 2-D).

This all may sound complex, but FastSLAM is quite easy to implement. Sampling from the motion model usually involves simple kinematic calculations. Computing the importance of a measurement is often straightforward too, especially for Gaussian measurement noise. And updating a low-dimensional particle filter is also not complicated.

FastSLAM has been shown to approximate the full SLAM posterior. The derivation of FastSLAM exploits three techniques: Rao–Blackwellization, conditional independence, and resampling. Rao–Blackwellization is the following concept. Suppose we would like to compute a probability distribution $p(a, b)$, where a and b are arbitrary random variables. The vanilla particle filter would draw particles from the joint distributions, that is, each particle would have a value for a and one for b . However, if the conditional $p(b | a)$ can be described in closed form, it is equally legitimate to just draw particles from $p(a)$, and attach to each particle a closed-form description of $p(b | a)$. This trick is known as Rao–

Blackwellization, and it yields better results than sampling from the joint. FastSLAM applies this technique, in that it samples from the path posterior $p(X_i^{[k]} | U_i, Z_i)$ and represents the map $p(m | X_i^{[k]}, U_i, Z_i)$ in Gaussian form.

FastSLAM also breaks down the posterior over maps (conditioned on paths) into sequences of low-dimensional Gaussians. The justification for this decomposition is subtle. It arises from a specific conditional independence assumption that is native to SLAM. Fig. 46.3 illustrates the concept graphically. In SLAM, knowledge of the robot path renders all landmark estimates independent. This is easily shown for the graphical network in Fig. 46.3: we find that if we remove the path variables from Fig. 46.3, then the landmark variables are all disconnected [46.43]. Thus, in SLAM *any dependence between multiple landmark estimates is mediated through the robot path*. This subtle but important observation implies that even if we used a large, monolithic Gaussian for the entire map (one per particle, of course), the off-diagonal element between different landmarks would simply remain zero. It is therefore legitimate to implement the map more efficiently, using N small Gaussians, one for each landmark. This explains the efficient map representation in FastSLAM.

Figure 46.4 shows results for a point-feature problem; here the point features are the centers of tree trunks as observed by an outdoor robot. The dataset used here is known as the Victoria Park dataset [46.44]. Fig. 46.4a shows the path of the vehicle obtained by integrating the vehicle controls, without perception. As can be seen, controls are a poor predictor of location for this vehicle; after 30 min of driving, the estimated position of the vehicle is well over 100 m away from its GPS position.

The FastSLAM algorithm has a number of interesting properties. First, it solves both full and online SLAM problems. Each particle has a sample of an entire path but the actual update equation only uses the most recent pose. This makes FastSLAM a filter. Second, FastSLAM can maintain multiple data association hypotheses. It is straightforward to make data association decisions on a per-particle basis, instead of having to adopt the same hypothesis for the entire filter. While we will not give any mathematical justification, we note that the resulting FastSLAM algorithm can even deal with unknown data association – something that the extended Kalman filter cannot claim. And third, FastSLAM can be implemented very efficiently using advanced tree methods to represent the map estimates, the update can be performed in time logarithmic in the size of the map N , and linear in the number of particles M .

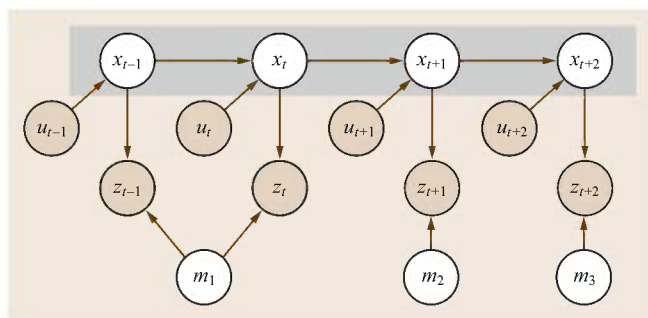


Fig. 46.3 The SLAM problem depicted as Bayes network graph. The robot moves from location x_{t-1} to location x_{t+2} , driven by a sequence of controls. At each location x_t it observes a nearby feature in the map $m = \{m_1, m_2, m_3\}$. This graphical network illustrates that the location variables *separate* the individual features in the map from each other. If the locations are known, there remains no other path involving variables whose value is not known, between any two features in the map. This lack of a path renders the posterior of any two features in the map conditionally independent (given the locations)

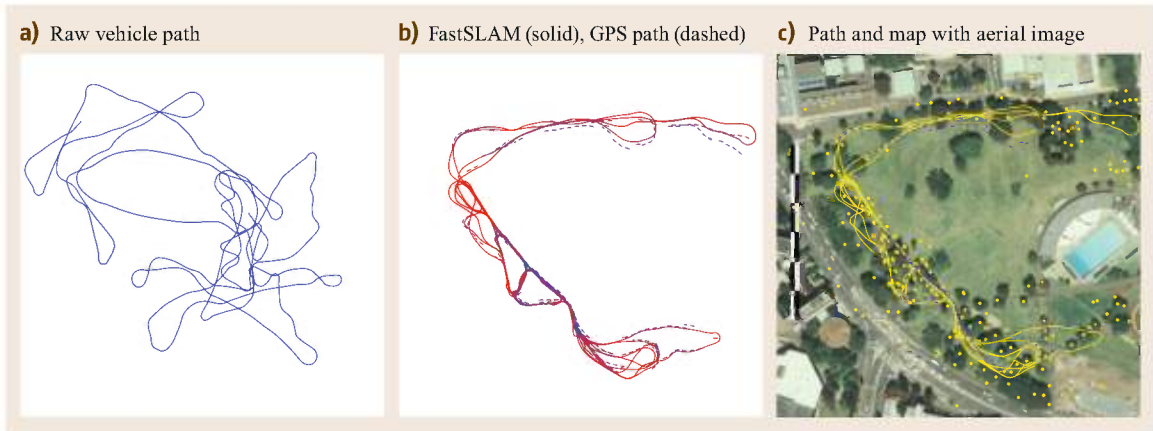


Fig. 46.4 (a) Vehicle path predicted by the odometry; (b) True path (*dashed line*) and FastSLAM 1.0 path (*solid line*); (c) Victoria Park results overlaid on aerial imagery with the GPS path in *blue (dashed)*, average FastSLAM 1.0 path in *yellow (solid)*, and estimated features as *yellow dots* (data and aerial image courtesy of José Guivant and Eduardo Nebot, Australian Centre for Field Robotics)



Fig. 46.5 Occupancy grid map generated from laser range data and based on pure odometry (image courtesy of Dirk Hähnel, University of Freiburg)

FastSLAM has been extended in several ways. One set of variants are grid-based versions of FastSLAM, in which the Gaussians used to model point landmarks are replaced by an occupancy grid map [46.45–47]. The variant of [46.46] is illustrated in Fig. 46.5.

Figure 46.6 illustrates a simplified situation with three particles just before closing a large loop. The three different particles each stand for different paths, and they also possess their own local maps. When the loop is closed importance resampling selects those particles

whose maps are most consistent with the measurement. A resulting large-scale map is shown in Fig. 46.5. Further extensions can be found in [46.48, 49], whose methods are called DP-SLAM and operate on ancestry trees to provide efficient tree update methods for grid-based maps. Related to that, approximations to FastSLAM in which particles share their maps have been proposed [46.50].

The works in [46.45, 47, 51] provide ways to incorporate new observations into the location sampling

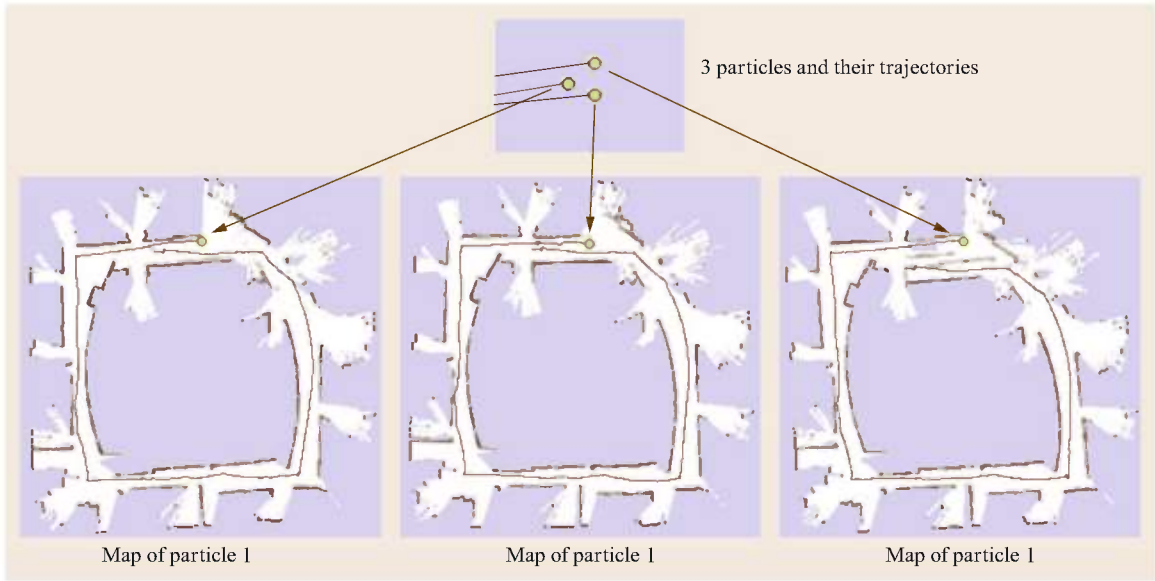


Fig. 46.6 Application of the grid-based variant of the FastSLAM algorithm. Each particle carries its own map and the importance weights of the particles are computed based on the likelihood of the measurements given the particle's own map

process for landmarks and grid maps, based on prior work in [46.52]. This leads to an improved sampling process

$$x_t^{[k]} \approx \frac{p(z_t | m_{t-1}^{[k]}, x_t) p(x_t | x_{t-1}^{[k]}, u_t)}{p(z_t | m_{t-1}^{[k]}, x_{t-1}^{[k]}, u_t)}, \quad (46.14)$$

which incorporates the odometry and the observation at the same time. Using an improved proposal distribution leads to more accurately sampled locations. This in turn leads to more accurate maps and requires a smaller number of particles compared to approaches using the sampling process given in (46.12). This extension makes FastSLAM and especially its grid-based variants robust tools for addressing the SLAM problem.

Finally, there are approaches that aim to overcome the assumption that the observations show Gaussian characteristics. As shown in [46.47], there are several situations in which the model is nonGaussian and also multimodal. A sum of Gaussians model on a per-particle bases, however, can be efficiently considered in the particle filter and it eliminates this problem in practice without introducing additional computational demands.

The so-far developed particle filters-based SLAM systems suffer from two problems. First, the number of samples that are required to compute consistent maps is often set manually by making an *educated guess*. The larger the uncertainty that the filter needs to represent during mapping, the more critical becomes this

parameter. Second, nested loops combined with extensive re-visits of previously mapped areas can lead to particle depletion, which in turn may prevent the system from estimating a consistent map. Adaptive resampling strategies [46.45], particles sharing maps [46.50], or filter backup approaches [46.53] improve the situation but cannot eliminate this problem in general.

46.2.3 Graph-Based Optimization Techniques

A third family of algorithms solves the SLAM problem through nonlinear sparse optimization. They draw their intuition from a graphical representation of the SLAM problem and the first working solution in robotics was proposed in [46.54]. The graph-based representation used here is closely related to a series of papers [46.55–64]. We note that most of the earlier techniques are offline and address the full SLAM problem. In more recent years, new incremental versions that effectively re-use the previously computed solution have been proposed such as [46.65–67].

The basic intuition of graph-based SLAM is as follows. Landmarks and robot locations can be thought of as nodes in a graph. Every consecutive pair of locations x_{t-1}, x_t is tied together by an edge that represents the information conveyed by the odometry reading u_t . Further edges exist between the nodes that correspond to locations x_t and landmarks m_i , assuming that at time t the robot sensed landmark i . Edges in this graph are

soft constraints. Relaxing these constraints yields the robot's best estimate for the map and the full path.

The construction of the graph is illustrated in Fig. 46.7. Suppose at time $t = 1$, the robot senses landmark m_1 . This adds an arc in the (yet highly incomplete) graph between x_1 and m_1 . When caching the edges in a matrix format (which happens to correspond to a quadratic equation defining the resulting constraints), a value is added to the elements between x_1 and m_1 , as shown on the right hand side of Fig. 46.7a.

Now suppose the robot moves. The odometry reading u_2 leads to an arc between nodes x_1 and x_2 , as shown in Fig. 46.7b. Consecutive application of these two basic steps leads to an graph of increasing size, as illustrated in Fig. 46.7c. Nevertheless this graph is *sparse*, in that each node is only connected to a small number of other nodes (assuming a sensor with limited sensing range). The number of constraints in the graph is (at worst) *linear* in the time elapsed and in the number of nodes in the graph.

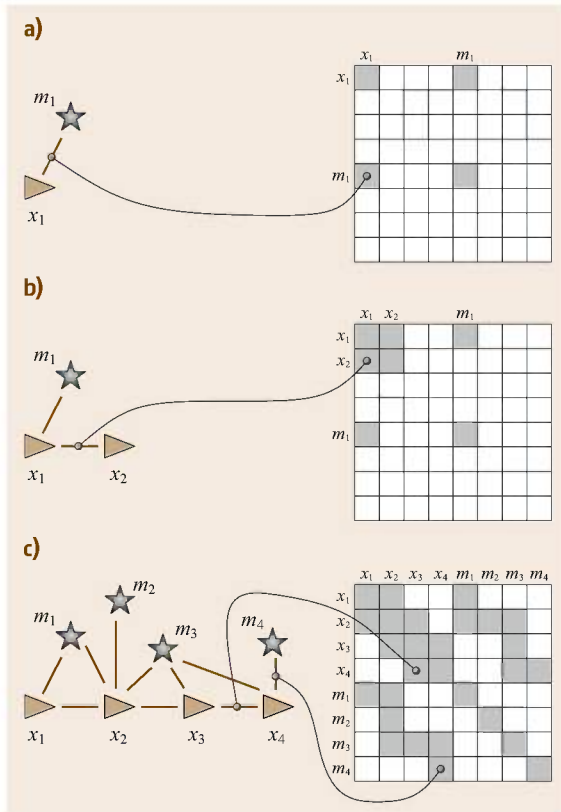


Fig. 46.7a–c Illustration of the graph construction. The (a) diagram shows the graph, the (b) the constraints in matrix form. (a) Observation 1s landmark m_1 . (b) Robot motion from x_1 to x_2 . (c) Several steps later

If we think of the graph as a spring-mass model [46.60], computing the SLAM solution is equivalent to computing the state of minimal energy of this model. To see, we note that the graph corresponds to the log-posterior of the full SLAM problem (46.4)

$$\log p(X_T, m | Z_T, U_T). \quad (46.15)$$

Without derivation, we state that this logarithm is of the form

$$\begin{aligned} \log p(X_T, m | Z_T, U_T) \\ = \text{const} + \sum_t \log p(x_t | x_{t-1}, u_t) \\ + \sum_t \log p(z_t | x_t, m), \end{aligned} \quad (46.16)$$

assuming independence between the individual observations and odometry readings. Each constraint of the form $\log p(x_t | x_{t-1}, u_t)$ is the result of exactly one robot motion event, and it corresponds to an edge in the graph. Likewise, each constraint of the form $\log p(z_t | x_t, m)$ is the result of one sensor measurement, to which we can also find a corresponding edge in the graph. The SLAM problem is then simply to find the mode of this equation, i. e.,

$$X_T^*, m^* = \underset{X_T, m}{\operatorname{argmax}} \log p(X_T, m | Z_T, U_T). \quad (46.17)$$

Without derivation, we note that under the Gaussian noise assumptions, which was made in the point-landmark example, this expression resolves to the following quadratic form

$$\begin{aligned} \log p(X_T, m | Z_T, U_T) = \text{const} \\ + \sum_t \underbrace{[x_t - g(x_{t-1}, u_t)]^T R_t^{-1} [x_t - g(x_{t-1}, u_t)]}_{\text{odometry reading}} \\ + \sum_t \underbrace{[z_t - h(x_t, m)]^T Q_t^{-1} [z_t - h(x_t, m)]}_{\text{feature observation}}. \end{aligned} \quad (46.18)$$

This quadratic form yields a sparse system of equations and a number of efficient optimization techniques can be applied. Common choices include direct methods such as sparse Cholesky and QR decomposition, or iterative ones such as gradient descent, conjugate gradient, and others. Most SLAM implementations rely on iteratively linearizing the functions g and h , in which case the objective in (46.18) becomes quadratic in all of its variables.

Extensions to support an effective correction of large-scale graphs are hierarchical methods. One of

the first is the ATLAS framework [46.25], which constructs a two-level hierarchy combining a Kalman filter that operates in the lower level and a global optimization at the higher level. Similar to that, Hierarchical SLAM [46.68] is a technique for using independent local maps, which are merged in case of re-visiting a place. A fully hierarchical approach has been presented in [46.65]. It builds a multilevel pose-graph and employs an incremental, lazy optimization scheme that allows for optimizing large graphs and at the same time can be executed at each step during mapping. An alternative hierarchical approach is [46.69], which recursively partitions the graph into multiple-level submaps using the nested dissection algorithm.

When it comes to computing highly accurate environment reconstructions, approaches that do not only optimize the poses of the robot but also each individual measurement of a dense sensor often provide better results. In the spirit of bundle adjustment [46.4], approaches for laser scanners [46.70] and Kinect cameras [46.71] have been proposed.

The graphical paradigm can be extended to handle the data association problems as we can integrate additional knowledge on data association into (46.18). Suppose some oracle informed us that landmarks m_i and m_j in the map corresponded to one and the same physical landmark in the world. Then, we can either remove m_j from the graph and attach all adjacent edges to m_i , or we can add a soft correspondence constraint [46.72] of the form

$$(m_j - m_i)^T \Gamma (m_j - m_i). \quad (46.19)$$

Here Γ is 2-by-2 diagonal matrix whose coefficients determine the penalty for *not* assigning identical locations to two landmarks (hence we want Γ to be large). Since graphical methods are usually used for the full SLAM problem, the optimization can be interleaved with the search for the optimal data association.

Data association errors typically have a strong impact in the resulting map estimate. Even a small number of wrong data associations is likely to result in inconsistent map estimates. Recently, novel approaches have been proposed that are robust under a certain number of false associations. For example, [46.73, 74] propose an iterative procedure that allows for disabling constraints, an action that is associated with a cost. A generalization of this method introduced in [46.75] formulates [46.74] as a robust cost function also reducing the computational requirements. Such approaches can deal with a significant number of false associations and still provide high-quality maps. Consistency checks for loop closure hypotheses can be found in other approaches as well, both in the front-end [46.76] and in the opti-

mizer [46.77]. There has also been an extension that can deal with multimodal constraints [46.78], proposing a max-mixture representation for maintaining efficiency of the log likelihood optimizing in (46.16). As a result of that, the multimodal extension has only little impact on the runtime and can easily be incorporated in most optimizers. Also robust cost function are used for SLAM, for example pseudo Huber and several alternatives [46.75, 79–81].

Graphical SLAM methods have the advantage that they scale to much higher-dimensional maps than EKF SLAM, exploiting the sparsity of the graph. The key limiting factor in EKF SLAM is the covariance matrix, which takes space (and update time) quadratic in the size of the map. No such constraint exists in graphical methods. The update time of the graph is constant, and the amount of memory required is linear (under some mild assumptions). A further advantage of graph-based methods over the EKF is their ability to constantly re-linearize the error function which often leads to better results. Performing the optimization can be expensive, however. Technically, finding the optimal data association is suspected to be an NP-hard problem, although in practice the number of plausible assignments is usually small. The continuous optimization of the log likelihood function in (46.18) depends among other things on the number and size of loops in the map. Also the initialization can have a strong impact on the result and a good initial guess can simplify the optimization substantially [46.8, 82, 83].

We note that the graph-based paradigm is very closely linked to information theory, in that the soft constraints constitute the information the robot has on the world (in an information-theoretic sense [46.92]). Most methods in the field are offline and they optimize for the entire robot path. If the robot path is long, the optimization may become cumbersome. Over the last five years, however, incremental optimization techniques have been proposed that aim at providing a sufficient but not necessarily perfect model of the environment at every point in time. This allows a robot to make decisions based on the current model, for example, to determine exploration goals. In this context, incremental variants [46.93, 94] of stochastic gradient descent techniques [46.8, 91] have been proposed that estimate which part of the graph requires re-optimization given new sensor data. Incremental methods [46.66, 79, 95] in the smoothing and mapping framework can be executed at each step of the mapping process and achieve the performance by variable ordering and selective re-linearization. As also evaluated in [46.96] for the SLAM problem, variable ordering impacts the performance of the optimization. Others use hierarchical data structures [46.89] and pose-graphs [46.97] combined with

Table 46.1 Recent open-source graph-based SLAM implementations

Name	Comment
Dynamic covariance scaling (DCS) [46.75]	Optimization with a robust cost function for dealing with outliers
g^2o [46.80]	Integrated into g^2o Flexible and easily extendable optimization framework for SLAM Comes with different optimization approaches and error functions Supports external plugins
GTSAM2.1 [46.79]	Flexible optimization framework for SLAM and SFMstructure from motion Implements direct and iterative optimization techniques Implements smoothing and mapping (SAM), iSAM, and iSAM2 Implements bundle adjustment for Visual SLAM and SFM
HOG-Man [46.65]	Incremental optimization approach via hierarchical pose graphs and lazy optimization Requires pose-graphs with full rank constraints
iSAM2 [46.66]	General incremental nonlinear optimization with variable elimination Variable re-ordering to retain sparsity On-demand re-linearization of selected variables
KinFu (KinectFusion reimplemented)	Open source reimplementation of KinectFusion [46.84] within the point cloud library (PCL) Dense and highly accurate reconstruction using a Kinect camera Currently limited to medium sized rooms
MaxMixture [46.78]	Optimization for multimodal constraints and outliers Robust to outliers Plugin for g^2o
Parallel tracking and mapping (PTAM) [46.85]	System for tracking a hand-held monocular camera and observed features Operates on comparably small workspaces
RGBD-SLAM [46.86]	Kinect-frontend for HOG-Man and g^2o Fairly standard combination of SURF matching and RANSAC
ScaViSLAM [46.87]	SLAM system for stereo and Kinect-style cameras Combines local bundle adjustment with sparse global optimization for on-the-fly processing
SLAM6-D [46.88]	SLAM system that operates on point clouds from 3-D laser data Applies iterative closest point algorithm (ICP) and global relaxation
Sparse surface adjustment (SSA) [46.70, 71]	Optimizes robot poses and proximity sensor data jointly Provides smooth surface estimates Assumes a range sensor (e.g., laser scanner, Kinect, or similar)
TreeMap [46.89]	Incremental optimization approach Update in $O(\log N)$ time Provides only a mean estimate
TORO [46.90]	Optimization approach that extends stochastic gradient descent (SGD) [46.91] Robust under bad initial guesses Recovers quickly from large errors but slow convergence at minimum Assumes that constraints have roughly spherical covariance matrices Provides only a mean estimate
Vertigo [46.74]	Switchable constraints for robust optimization Plugin for g^2o

a lazy optimization for on-the-fly mapping [46.65]. As an alternative to global methods, relative optimization approaches [46.98] aim at computing locally consistent geometric maps but only topological maps on the global scale. Hybrid approaches [46.87] seek to combine the best of both worlds.

There also exists a number of cross-overs that manipulate the graph online so as to factor out past robot location variables. The resulting algorithms are filters [46.25, 33, 99, 100], and they tend to be intimately related to information filter methods. Many of the original attempts to decompose EKF SLAM representations into smaller submaps to scale up are based

on motivations that are not dissimilar to the graphical approach [46.27, 28, 101].

Recently, researchers addressed the problem of long-term operation and frequent revisits of already mapped terrain. To avoid densely connected pose-graphs that lead to slow convergence behavior, the robot can switch between SLAM and localization, can merge nodes to avoid a growth of the graph [46.90, 102], or can discard nodes or edges [46.32, 103–105].

Graphical and optimization-based SLAM algorithms are still subject of intense research and the paradigm scales to maps large numbers of nodes [46.25, 55, 57, 59, 63–65, 89, 90, 106, 107]. Arguably, the graph-based

paradigm has generated some of the largest SLAM maps ever built. Furthermore, the SLAM community started to release flexible optimization frameworks and SLAM implementations under open source licenses to support further developments and to allow for efficient comparisons, (Table 46.1). Especially the optimization frameworks [46.66, 79, 80] are flexible and powerful state of the art tools for developing graph-based SLAM systems. They can be either used as a *black box* or can be easily extended through plugins.

46.2.4 Relation of Paradigms

The three paradigms just discussed cover the vast majority of work in the field of SLAM. As discussed, EKF SLAM comes with a computational hurdle that poses serious scaling limitations and the linearization may lead to inconsistent maps. The most promising extensions of EKF SLAM are based on building local submaps; however, in many ways the resulting algorithms resemble the graph-based approach.

Particle filter methods sidestep some of the issues arising from the natural inter-feature correlations in the map – which hindered the EKF. By sampling from robot poses, the individual landmarks in the map

become independent, and hence are decorrelated. As a result, FastSLAM can represent the posterior by a sampled robot pose, and many local, independent Gaussians for its landmarks. The particle representation offers advantages for SLAM as it allows for computationally efficient updates and for sampling over data associations. On the negative side, the number of necessary particles can grow very large, especially for robots seeking to map multiple nested loops.

Graph-based methods address the full SLAM problem, hence are in the standard formulation not online. They draw their intuition from the fact that SLAM can be modeled by a sparse graph of soft constraints, where each constraint either corresponds to a motion or a measurement event. Due to the availability of highly efficient optimization methods for sparse nonlinear optimization problems, graph-based SLAM has become the method of choice for building large-scale maps. Recent developments have brought up several graph-based methods for incremental map building that can be executed at every time step during navigation. Data association search can be incorporated into the basic mathematical framework and different approaches that are even robust under wrong data associations are available today.

46.3 Visual and RGB-D SLAM

A popular and important topic in recent years has been Visual SLAM – the challenge of building maps and tracking the robot pose in full 6-DOF using data from cameras [46.108] or RGB-D (Kinect) sensors [46.86, 109]. Visual sensors offer a wealth of information that enables the construction of rich 3-D models of the world. They also enable difficult issues such as loop-closing to be addressed in novel ways using appearance information [46.110]. Visual SLAM is anticipated to be a critical area for future research in perception for robotics, as we seek to develop low-cost systems that are capable of intelligent physical interaction with the world.

Attempting SLAM with monocular, stereo, omnidirectional, or RGB-D cameras raises the level-of-difficulty of many of the SLAM components, such as data association and computational efficiency, described above. A key challenge is robustness. Many visual SLAM applications of interest, such as augmented reality [46.85], entail handheld camera motions, which present greater difficulties for state estimation, in comparison to the motion of a wheeled robot across a flat floor.

Visual navigation and mapping was a key early goal in the mobile robotics community [46.111, 112], but

early approaches were hampered by the lack of sufficient computational resources to handle massive video data streams. Early approaches were typically based on extended Kalman filters [46.113–116], but did not compute the full covariance for the feature poses and camera trajectory, resulting in a loss of consistency. Visual SLAM is closely related to the structure from motion (SFM) problem in computer vision [46.4, 5]. Historically, SFM was primarily concerned with off-line batch processing, whereas SLAM seeks to achieve a solution for online operation, suitable for closed-loop interaction of a robot or user with its environment. In comparison to laser scanners, cameras provide a fire hydrant of information, making online processing nearly impossible until recent increases in computation have become available.

Davison was an early pioneer in developing complete visual SLAM systems, initially using a real-time active stereo head [46.121] that tracked distinctive visual features with a full covariance EKF approach. Subsequent work developed the first real-time SLAM system that operated with a single freely moving camera as the only data source [46.23, 122]. This system could build sparse, room-size maps of indoor scenes at 30 Hz frame-rate in real-time, a notable historical achievement

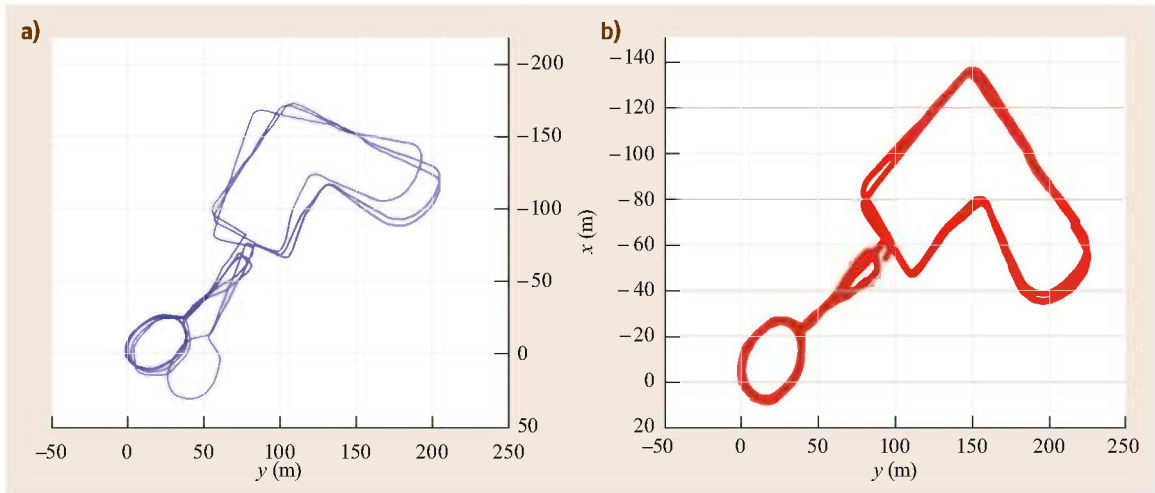


Fig. 46.8 (a) A 2-km path and 50 000 frames estimated for the New College Dataset (after [46.117]) using relative bundle adjustment (after [46.98]). (b) the relative bundle adjustment solution is easily improved by taking FAB-MAP (after [46.118]) loop-closures into account – this is achieved without global optimization (after [46.119]). Sibley et al. advocate that relative metric accuracy and topological consistency are the requirements for autonomous navigation, and these are better achieved using a relative manifold representation instead of using a conventional single Euclidean representation [46.120]

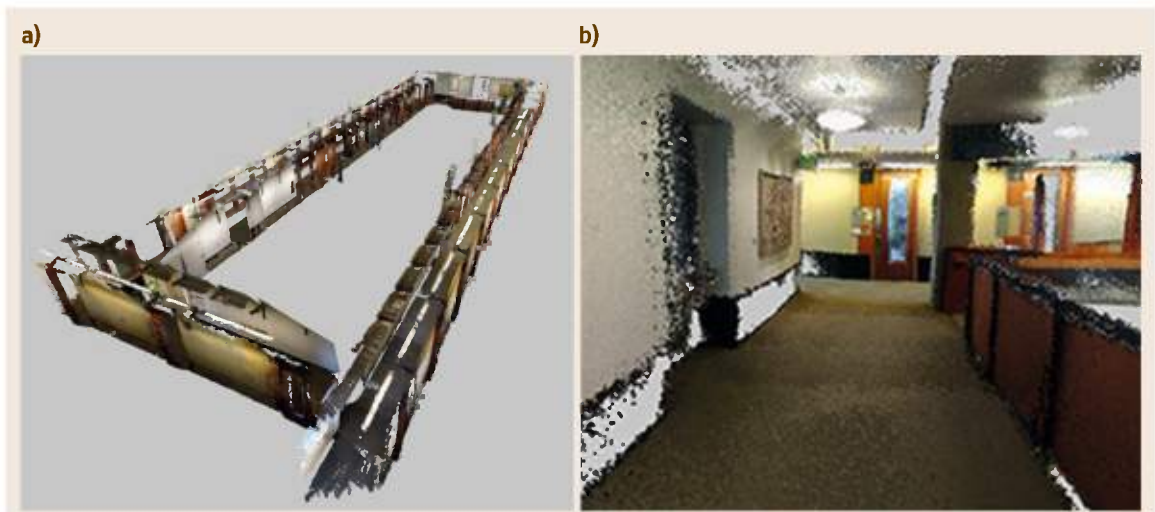


Fig. 46.9 (a) 3-D Model and (b) close-up view of a corridor environment in the Paul G. Allen building at University of Washington built from Kinect data (after [46.109]; image courtesy of Peter Henry, University of Washington)

in visual SLAM research. A difficulty encountered with initial monocular SLAM [46.23] was coping with the initialization of points that were far away from the camera, due to nonGaussian distributions of such feature locations resulting from poor depth information. This limitation was overcome in [46.24], introducing an inverse depth parameterization for monocular SLAM, a key development for enabling a unified treatment of initialization and tracking of visual features in real-time.

A milestone in creating robust visual SLAM systems was the introduction of keyframes in parallel tracking and mapping (PTAM) [46.85], which separated the tasks of keyframe mapping and localization into parallel threads, improving robustness and performance for online processing. Keyframes are now a mainstream concept for complexity reduction in visual SLAM systems. Related approaches using keyframes include [46.87, 123–126]. The work

in [46.108] analyzes the tradeoffs between filtering and keyframe-based bundle adjustment in visual SLAM, and concluded that keyframe bundle adjustment outperforms filtering, as it provides the most accuracy per unit of computing time.

As pointed out by Davison and other researchers, an appealing aspect of visual SLAM is that camera measurements can provide odometry information, and indeed visual odometry is a key component of modern SLAM systems [46.127]. Here, [46.128] and [46.129] provide an extensive tutorial of techniques for visual odometry, including feature detection, feature matching, outlier rejection, and constraint estimation, and trajectory optimization. Finally, a publicly available visual odometry library [46.130] that is optimized for efficient operation on small unmanned aerial vehicles is available today.

Visual information offers a tremendous source of information for loop closing, not present in the canonical 2-D laser SLAM systems developed in the early 2000s. The work in [46.110] was one of the first to employ techniques for visual object recognition [46.131] to location recognition. More recently FAB-MAP [46.118, 132] has demonstrated appearance-only place recognition at large scale, mapping trajectories with a length of 1000 km. Combining a bag-of-features approach with a probabilistic place model and Chow–Liu tree inference leads to place recognition that is robust against perceptual aliasing while remaining computationally efficient. Other work on place recognition includes [46.133], which combines bag-of-words loop closing with tests of geometrical consistency based on conditional random fields (Fig. 46.8).

The techniques described above have formed the basis for a number of notable large-scale SLAM systems developed in recent years. A 2008 special issue of the IEEE Transactions on Robotics provides a good snapshot of recent state-of-the-art SLAM techniques [46.135]. Other notable recent examples include [46.98, 119, 126, 136–138]. The idea of employing relative bundle adjustment [46.98] to compute a full maximum likelihood solution in an online fashion, even for loop closures, by employing a manifold representation that does not attempt to enforce Euclidean constraints results in maps that can be computed at high frame rate (see also Fig. 46.8). Finally, view-based mapping systems [46.126, 136, 137] aim at large-scale and/or life-long visual mapping based on the pose graph optimization techniques described above in Section 46.2.3.

Several compelling 3-D mapping and localization have been created in recent years with RGB-D (Kinect) sensors. The combination of direct range measurements with dense visual imagery can enable dramatic im-

provements in mapping and navigation systems for indoor environments. State-of-the-art RGB-D SLAM systems include [46.109] and [46.86]. Figure 46.9 shows examples of the output of these systems.

Other researchers aim at exploiting the surface properties of scanned environments to correct for sen-

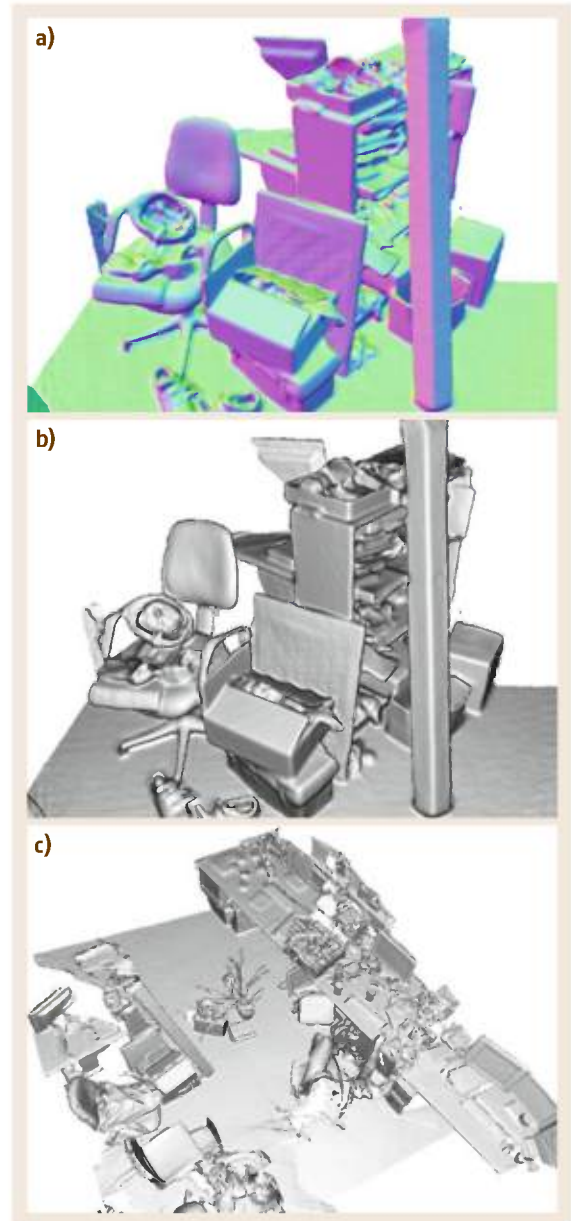


Fig. 46.10a–c Results obtained with KinectFusion (after [46.134]). (a) A local scene as a normal map and (b) as a Phong-shaded rendering. The (c) image depicts a larger scene (image courtesy of Richard Newcombe, Imperial College London)

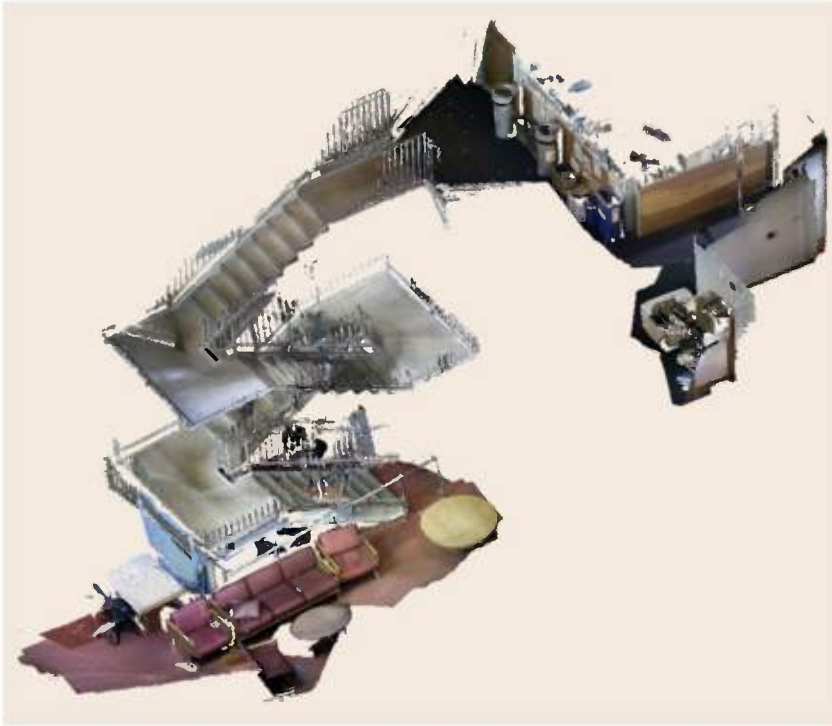


Fig. 46.11 Spatially extended KinectFusion output produced in real-time with Kintinuous (after [46.139])

sor noise of range sensors such as the Kinect [46.71]. They jointly optimize the poses of the sensor and the positions of the surface points measured and iteratively refine the structure of the error function by recomputing the data associations after each optimization, resulting in accurate smooth models of the environment.

An emerging area for future research is the development of fully dense processing methods that exploit recent advances in commodity graphical processing unit (GPU) technology. Kinect-based dense tracking and mapping, a fully-dense method for small-scale visual tracking and reconstruction is described in [46.140]. Dense modeling and tracking are achieved

via highly parallelized operations on commodity GPU hardware to yield a system that outperforms previous methods such as PTAM for challenging camera trajectories. Dense methods offer an interesting perspective from which to address long-standing problems, such as visual odometry, from a fresh perspective, without requiring explicit feature detection and matching [46.141]. KinectFusion [46.84, 134] is a dense modeling system that tracks the 3-D pose of a handheld Kinect while concurrently reconstructing high-quality scene 3-D models in real-time. See Fig. 46.10 for an example. KinectFusion has been applied to spatially extended environments in [46.139, 142]. An example is shown in Fig. 46.11.

46.4 Conclusion and Future Challenges

This chapter has provided an introduction into SLAM, which is defined as the problem faced by a mobile platform roaming an unknown environment, and seeking to localize itself while concurrently building a map of the environment. The chapter discussed three main paradigms in SLAM, which are based on the extended Kalman filter, particle filters, and graph-based sparse optimization techniques, and then described recent progress in Visual/Kinect SLAM.

The following references provide an in-depth tutorial on SLAM and much greater depth of coverage on the details of popular SLAM algorithms. Furthermore, several implementations of popular SLAM systems, including most of the approaches listed in Table 46.1, can be found in online resources such as <http://www.openslam.org> or in the references [46.6, 9, 21, 62].

The considerable progress in SLAM in the past decade is beyond doubt. The core state estimation at

the heart of SLAM is now quite well understood, and a number of impressive implementations have been developed, including several widely used open source software implementations and some commercial projects. None-the-less, a number of open research challenges remain for the general problem of robotic mapping in complex and dynamic environments over extended periods of time, including robots sharing, extending, and revising previously built models, efficient failure recovery, zero user intervention, and operation on resource-constrained systems. Another exciting area for the future is the further development of fully dense visual mapping systems exploiting the latest advances in GPU hardware development.

An ultimate goal is to realize the challenge of *persistent* navigation and mapping – the capability for a robot to perform SLAM robustly for days, weeks, or months at a time with minimal human supervision, in complex and dynamic environments. Taking the limit as $t \rightarrow \infty$ poses difficult challenges to most current algorithms; in fact, most robot mapping and navigation algorithms are doomed to fail with the passage of time, as errors inevitably accrue. Despite recent encouraging solutions [46.102, 105], more research is needed for techniques that can recover from mistakes and enable robots to deal with changes in the environment and enabling a long-term autonomous existence.

Video-References

-  VIDEO 439 Deformation-based loop closure for Dense RGB-D SLAM
available from <http://handbookofrobotics.org/view-chapter/46/videodetails/439>
-  VIDEO 440 Large-scale SLAM using the Atlas framework
available from <http://handbookofrobotics.org/view-chapter/46/videodetails/440>
-  VIDEO 441 Graph-based SLAM
available from <http://handbookofrobotics.org/view-chapter/46/videodetails/441>
-  VIDEO 442 Graph-based SLAM
available from <http://handbookofrobotics.org/view-chapter/46/videodetails/442>
-  VIDEO 443 Graph-based SLAM
available from <http://handbookofrobotics.org/view-chapter/46/videodetails/443>
-  VIDEO 444 Graph-based SLAM
available from <http://handbookofrobotics.org/view-chapter/46/videodetails/444>
-  VIDEO 445 Graph-based SLAM
available from <http://handbookofrobotics.org/view-chapter/46/videodetails/445>
-  VIDEO 446 Graph-based SLAM using TORO
available from <http://handbookofrobotics.org/view-chapter/46/videodetails/446>
-  VIDEO 447 Sparse pose adjustment
available from <http://handbookofrobotics.org/view-chapter/46/videodetails/447>
-  VIDEO 449 Pose graph compression for laser-based SLAM
available from <http://handbookofrobotics.org/view-chapter/46/videodetails/449>
-  VIDEO 450 Pose graph compression for laser-based SLAM
available from <http://handbookofrobotics.org/view-chapter/46/videodetails/450>
-  VIDEO 451 Pose graph compression for laser-based SLAM
available from <http://handbookofrobotics.org/view-chapter/46/videodetails/451>
-  VIDEO 452 DTAM: Dense tracking and mapping in real-time
available from <http://handbookofrobotics.org/view-chapter/46/videodetails/452>
-  VIDEO 453 MonoSLAM: Real-time single camera SLAM
available from <http://handbookofrobotics.org/view-chapter/46/videodetails/453>
-  VIDEO 454 SLAM++: Simultaneous localisation and mapping at the level of objects
available from <http://handbookofrobotics.org/view-chapter/46/videodetails/454>
-  VIDEO 455 Extended Kalman filter SLAM
available from <http://handbookofrobotics.org/view-chapter/46/videodetails/455>

References

- 46.1 C.F. Gauss: *Theoria Motus Corporum Coelestium (Theory of the Motion of the Heavenly Bodies Moving about the Sun in Conic Sections)* (Perthes and Bessen, Hamburg 1809), Republished in 1857 and by Dover in 1963
- 46.2 D.C. Brown: The bundle adjustment – Progress and prospects, *Int. Arch. Photogramm.* **21**(3), 3:3–3:35 (1976)
- 46.3 G. Konecny: *Geoinformation: Remote Sensing, Photogrammetry and Geographical Information Systems* (Taylor Francis, London 2002)
- 46.4 B. Triggs, P. McLauchlan, R. Hartley, A. Fitzgibbon: Bundle adjustment – A modern synthesis, *Lect. Notes Comput. Sci.* **62**, 298–372 (2000)
- 46.5 R. Hartley, A. Zisserman: *Multiple View Geometry in Computer Vision* (Cambridge Univ. Press, Cambridge 2003)
- 46.6 T. Bailey, H.F. Durrant-Whyte: Simultaneous localisation and mapping (SLAM): Part II, *Robotics Autom. Mag.* **13**(3), 108–117 (2006)
- 46.7 H.F. Durrant-Whyte, T. Bailey: Simultaneous localisation and mapping (SLAM): Part I, *Robotics Autom. Mag.* **13**(2), 99–110 (2006)
- 46.8 G. Grisetti, C. Stachniss, W. Burgard: Nonlinear constraint network optimization for efficient map learning, *IEEE Trans. Intell. Transp. Syst.* **10**(3), 428–439 (2009)
- 46.9 S. Thrun, W. Burgard, D. Fox: *Probabilistic Robotics* (MIT Press, Cambridge, 2005)
- 46.10 R. Smith, M. Self, P. Cheeseman: A stochastic map for uncertain spatial relationships, *Proc. Int. Symp. Robotics Res. (ISRR)* (MIT Press, Cambridge 1988) pp. 467–474
- 46.11 R. Smith, M. Self, P. Cheeseman: Estimating uncertain spatial relationships in robotics. In: *Autonomous Robot Vehicles*, ed. by I.J. Cox, G.T. Wilfong (Springer Verlag, Berlin, Heidelberg 1990) pp. 167–193
- 46.12 P. Moutarlier, R. Chatila: Stochastic multisensory data fusion for mobile robot location and environment modeling, *5th Int. Symp. Robotics Res. (ISRR)* (1989) pp. 207–216
- 46.13 P. Moutarlier, R. Chatila: An experimental system for incremental environment modeling by an autonomous mobile robot, *1st Int. Sym. Exp. Robotics (ISER)* (1990)
- 46.14 R. Kalman: A new approach to linear filtering and prediction problems, *J. Fluids* **82**, 35–45 (1960)
- 46.15 A.M. Jazwinsky: *Stochastic Processes and Filtering Theory* (Academic, New York 1970)
- 46.16 M.G. Dissanayake, P.M. Newman, S. Clark, H.F. Durrant-Whyte, M. Csorba: A solution to the simultaneous localization and map building (SLAM) Problem, *IEEE Trans. Robotics Autom.* **17**(3), 229–241 (2001)
- 46.17 J. Neira, J. Tardos, J. Castellanos: Linear time vehicle relocation in SLAM, *Proc. IEEE Int. Conf. Robotics Autom. (ICRA)* (2003) pp. 427–433
- 46.18 J. Neira, J.D. Tardos: Data association in stochastic mapping using the joint compatibility test, *IEEE Trans. Robotics Autom.* **17**(6), 890–897 (2001)
- 46.19 G.D. Tipaldi, M. Braun, K.O. Arras: Flirt: interest regions for 2D range data with applications to robot navigation, *Proc. Int. Symp. Exp. Robotics (ISER)* (2010)
- 46.20 G.D. Tipaldi, L. Spinello, W. Burgard: Geometrical flirt phrases for large scale place recognition in 2D range data, *Proc. IEEE Int. Conf. Robotics Autom. (ICRA)* (2013)
- 46.21 T. Bailey: *Mobile Robot Localisation and Mapping in Extensive Outdoor Environments*, Ph.D. Thesis (Univ. of Sydney, Sydney 2002)
- 46.22 J.J. Leonard, R.R. Rikoski, P.M. Newman, M. Bosse: Mapping partially observable features from multiple uncertain vantage points, *Int. J. Robotics Res.* **21**(10), 943–975 (2002)
- 46.23 A.J. Davison: Real-time simultaneous localisation and mapping with a single camera, *Proc. IEEE 9th Int. Conf. Comput. Vis.* (2003) pp. 1403–1410
- 46.24 J.M.M. Montiel, J. Civera, A.J. Davison: Unified inverse depth parametrization for monocular SLAM, *Robotics Sci. Syst.*, Vol. 1 (2006)
- 46.25 M. Bosse, P.M. Newman, J. Leonard, S. Teller: Simultaneous localization and map building in large-scale cyclic environments using the Atlas Framework, *Int. J. Robotics Res.* **23**(12), 1113–1139 (2004)
- 46.26 J. Nieto, T. Bailey, E. Nebot: Scan-SLAM: Combining EKF-SLAM and scan correlation, *Proc. IEEE Int. Conf. Robotics Autom. (ICRA)* (2005)
- 46.27 J. Guivant, E. Nebot: Optimization of the simultaneous localization and map building algorithm for real time implementation, *IEEE Trans. Robotics Autom.* **17**(3), 242–257 (2001)
- 46.28 J.J. Leonard, H. Feder: A computationally efficient method for large-scale concurrent mapping and localization, *Proc. 9th Int. Symp. Robotics Res. (ISRR)*, ed. by J. Hollerbach, D. Koditschek (1999) pp. 169–176
- 46.29 J.D. Tardós, J. Neira, P.M. Newman, J.J. Leonard: Robust mapping and localization in indoor environments using sonar data, *Int. J. Robotics Res.* **21**(4), 311–330 (2002)
- 46.30 S.B. Williams, G. Dissanayake, H.F. Durrant-Whyte: Towards multi-vehicle simultaneous localisation and mapping, *Proc. IEEE Int. Conf. Robotics Autom. (ICRA)* (2002) pp. 2743–2748
- 46.31 R.M. Eustice, H. Singh, J.J. Leonard: Exactly sparse delayed-state filters for view-based SLAM, *IEEE Trans. Robotics* **22**(6), 1100–1114 (2006)
- 46.32 V. Ila, J.M. Porta, J. Andrade-Cetto: Information-based compact pose SLAM, *IEEE Trans. Robotics* **26**(1), 78–93 (2010)
- 46.33 S. Thrun, D. Koller, Z. Ghahramani, H.F. Durrant-Whyte, A.Y. Ng: Simultaneous mapping and localization with sparse extended information filters, *Proc. 5th Int. Workshop Algorithmic Found. Robotics*, ed. by J.-D. Boissonnat, J. Burdick, K. Goldberg, S. Hutchinson (2002)

- 46.34 M.R. Walter, R.M. Eustice, J.J. Leonard: Exactly sparse extended information filters for feature-based SLAM, *Int. J. Robotics Res.* **26**(4), 335–359 (2007)
- 46.35 T. Bailey, J. Nieto, J. Guivant, M. Stevens, E. Nebot: Consistency of the EKF-SLAM algorithm, *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)* (2006) pp. 3562–3568
- 46.36 S. Huang, G. Dissanayake: Convergence and consistency analysis for extended Kalman filter based SLAM, *IEEE Trans. Robotics* **23**(5), 1036–1049 (2007)
- 46.37 G.P. Huang, A.I. Mourikis, S.I. Roumeliotis: Observability-based Rules for Designing Consistent EKF SLAM Estimators, *Int. J. Robotics Res.* **29**, 502–528 (2010)
- 46.38 N. Metropolis, S. Ulam: The Monte Carlo method, *J. Am. Stat. Assoc.* **44**(247), 335–341 (1949)
- 46.39 D. Blackwell: Conditional expectation and unbiased sequential estimation, *Ann. Math. Stat.* **18**, 105–110 (1947)
- 46.40 C.R. Rao: Information and accuracy obtainable in estimation of statistical parameters, *Bull. Calcutta Math. Soc.* **37**(3), 81–91 (1945)
- 46.41 K. Murphy, S. Russell: Rao-Blackwellized particle filtering for dynamic Bayesian networks. In: *Sequential Monte Carlo Methods in Practice*, ed. by A. Doucet, N. de Freitas, N. Gordon (Springer, Berlin 2001) pp. 499–516
- 46.42 M. Montemerlo, S. Thrun, D. Koller, B. Wegbreit: FastSLAM: A factored solution to the simultaneous localization and mapping problem, *Proc. AAAI Natl. Conf. Artif. Intell.* (2002)
- 46.43 J. Pearl: *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference* (Morgan Kaufmann, San Mateo 1988)
- 46.44 J. Guivant, E. Nebot, S. Baiker: Autonomous navigation and map building using laser range sensors in outdoor applications, *J. Robotics Syst.* **17**(10), 565–583 (2000)
- 46.45 G. Grisetti, C. Stachniss, W. Burgard: Improved techniques for grid mapping with Rao-Blackwellized particle filters, *IEEE Trans. Robotics* **23**, 34–46 (2007)
- 46.46 D. Hähnel, D. Fox, W. Burgard, S. Thrun: A highly efficient FastSLAM algorithm for generating cyclic maps of large-scale environments from raw laser range measurements, *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)* (2003)
- 46.47 C. Stachniss, G. Grisetti, W. Burgard, N. Roy: Evaluation of gaussian proposal distributions for mapping with rao-blackwellized particle filters, *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)* (2007)
- 46.48 A. Eliazar, R. Parr: DP-SLAM: Fast, robust simultaneous localization and mapping without predetermined landmarks, *Proc. 16th Int. Jt. Conf. Artif. Intell. (IJCAI)* (2003) pp. 1135–1142
- 46.49 A. Eliazar, R. Parr: DP-SLAM 2.0, *Proc. IEEE Int. Conf. Robotics Autom. (ICRA)*, Vol. 2 (2004) pp. 1314–1320
- 46.50 G. Grisetti, G.D. Tipaldi, C. Stachniss, W. Burgard, D. Nardi: Fast and accurate SLAM with Rao-Blackwellized particle filters, *J. Robotics Auton. Syst.* **55**(1), 30–38 (2007)
- 46.51 D. Roller, M. Montemerlo, S. Thrun, B. Wegbreit: FastSLAM 2.0: An improved particle filtering algorithm for simultaneous localization and mapping that provably converges, *Int. Jt. Conf. Artif. Intell. (IJCAI)* (Morgan Kaufmann, San Francisco 2003) pp. 1151–1156
- 46.52 R. van der Merwe, N. de Freitas, A. Doucet, E. Wan: The unscented particle filter, *Proc. Adv. Neural Inform. Process. Syst. Conf.* (2000) pp. 584–590
- 46.53 C. Stachniss, G. Grisetti, W. Burgard: Recovering particle diversity in a Rao-Blackwellized particle filter for SLAM after actively closing loops, *Proc. IEEE Int. Conf. Robotics Autom. (ICRA)* (2005) pp. 655–660
- 46.54 F. Lu, E. Milios: Globally consistent range scan alignment for environmental mapping, *Auton. Robots* **4**, 333–349 (1997)
- 46.55 F. Dellaert: Square root SAM, *Robotics Sci. Syst.*, ed. by S. Thrun, G. Sukhatme, S. Schaal, O. Brock (MIT Press, Cambridge 2005)
- 46.56 T. Duckett, S. Marsland, J. Shapiro: Learning globally consistent maps by relaxation, *Proc. IEEE Int. Conf. Robotics Autom. (ICRA)* (2000) pp. 3841–3846
- 46.57 T. Duckett, S. Marsland, J. Shapiro: Fast, on-line learning of globally consistent maps, *Auton. Robots* **12**(3), 287–300 (2002)
- 46.58 J. Folkesson, H.I. Christensen: Graphical SLAM: A self-correcting map, *Proc. IEEE Int. Conf. Robotics Autom. (ICRA)* (2004) pp. 383–390
- 46.59 U. Frese, G. Hirzinger: Simultaneous localization and mapping – A discussion, *Proc. IJCAI Workshop Reason. Uncertain. Robotics* (2001) pp. 17–26
- 46.60 M. Golfarelli, D. Maio, S. Rizzi: Elastic correction of dead-reckoning errors in map building, *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)* (1998) pp. 905–911
- 46.61 J. Gutmann, K. Konolige: Incremental mapping of large cyclic environments, *Proc. IEEE Int. Symp. Comput. Intell. Robotics Autom. (CIRA)* (2000) pp. 318–325
- 46.62 G. Grisetti, R. Kümmerle, C. Stachniss, W. Burgard: A Tutorial on Graph-based SLAM, *IEEE Trans. Intell. Transp. Syst. Mag.* **2**, 31–43 (2010)
- 46.63 K. Konolige: Large-scale map-making, *Proc. AAAI Natl. Conf. Artif. Intell.* (MIT Press, Cambridge 2004) pp. 457–463
- 46.64 M. Montemerlo, S. Thrun: Large-scale robotic 3-D mapping of urban structures, *Springer Tract. Adv. Robotics* **21**, 141–150 (2005)
- 46.65 G. Grisetti, R. Kümmerle, C. Stachniss, U. Frese, C. Hertzberg: Hierarchical optimization on manifolds for online 2D and 3D mapping, *Proc. IEEE Int. Conf. Robotics Autom. (ICRA)* (2010)
- 46.66 M. Kaess, H. Johannsson, R. Roberts, V. Ila, J.J. Leonard, F. Dellaert: iSAM2: Incremental smoothing and mapping using the Bayes tree, *Int. J. Robotics Res.* **31**, 217–236 (2012)
- 46.67 M. Kaess, A. Ranganathan, F. Dellaert: iSAM: Incremental Smoothing and Mapping, *IEEE Trans. Robotics* **24**(6), 1365–1378 (2008)

- 46.68 C. Estrada, J. Neira, J.D. Tardós: Hierarchical SLAM: Real-time accurate mapping of large environments, *IEEE Trans. Robotics* **21**(4), 588–596 (2005)
- 46.69 K. Ni, F. Dellaert: Multi-level submap based SLAM using nested dissection, *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)* (2010)
- 46.70 M. Ruhnke, R. Kümmerle, G. Grisetti, W. Burgard: Highly accurate maximum likelihood laser mapping by jointly optimizing laser points and robot poses, *Proc. IEEE Int. Conf. Robotics Autom. (ICRA)* (2011)
- 46.71 M. Ruhnke, R. Kümmerle, G. Grisetti, W. Burgard: Highly accurate 3D surface models by sparse surface adjustment, *Proc. IEEE Int. Conf. Robotics Autom. (ICRA)* (2012)
- 46.72 Y. Liu, S. Thrun: Results for outdoor-SLAM using sparse extended information filters, *Proc. IEEE Int. Conf. Robotics Autom. (ICRA)* (2003)
- 46.73 N. Sünderhauf, P. Protzel: BRIEF-Gist – Closing the loop by simple means, *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)* (2011) pp. 1234–1241
- 46.74 N. Sünderhauf, P. Protzel: Switchable constraints for robust pose graph SLAM, *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)* (2012)
- 46.75 P. Agarwal, G.D. Tipaldi, L. Spinello, C. Stachniss, W. Burgard: Robust map optimization using dynamic covariance scaling, *Proc. IEEE Int. Conf. Robotics Autom. (ICRA)* (2013)
- 46.76 E. Olson: Recognizing places using spectrally clustered local matches, *J. Robotics Auton. Syst.* **57**(12), 1157–1172 (2009)
- 46.77 Y. Latif, C. Cadena Lerma, J. Neira: Robust loop closing over time, *Robotics Sci. Syst.* (2012)
- 46.78 E. Olson, P. Agarwal: Inference on networks of mixtures for robust robot mapping, *Robotics Sci. Syst.* (2012)
- 46.79 F. Dellaert: *Factor graphs and GTSAM: A hands-on introduction*, Tech. Rep. GT-RIM-CP & R-2012-002 (Georgia Tech, Atlanta 2012)
- 46.80 R. Kümmerle, G. Grisetti, H. Strasdat, K. Konolige, W. Burgard: G²o: A general framework for graph optimization, *Proc. IEEE Int. Conf. Robotics Autom. (ICRA)* (2011)
- 46.81 D.M. Rosen, M. Kaess, J.J. Leonard: An incremental trust-region method for robust online sparse least-squares estimation, *Proc. IEEE Int. Conf. Robotics Autom. (ICRA)* (2012) pp. 1262–1269
- 46.82 L. Carlone, R. Aragues, J. Castellanos, B. Bona: A linear approximation for graph-based simultaneous localization and mapping, *Robotics Sci. Syst.* (2011)
- 46.83 G. Grisetti, R. Kümmerle, K. Ni: Robust optimization of factor graphs by using condensed measurements, *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)* (2012)
- 46.84 S. Izadi, R.A. Newcombe, D. Kim, O. Hilliges, D. Molyneaux, S. Hodges, P. Kohli, J. Shotton, A.J. Davison, A. Fitzgibbon: Kinectfusion: Real-time dynamic 3D surface reconstruction and interaction, *ACM SIGGRAPH Talks* (2011) p. 23
- 46.85 G. Klein, D. Murray: Parallel tracking and mapping for small AR workspaces, *IEEE ACM Int. Symp. Mixed Augment. Real. (ISMAR)* (2007) pp. 225–234
- 46.86 F. Endres, J. Hess, N. Engelhard, J. Sturm, D. Cremers, W. Burgard: An evaluation of the RGB-D SLAM system, *Proc. IEEE Int. Conf. Robotics Autom. (ICRA)* (2012)
- 46.87 H. Strasdat, A.J. Davison, J.M.M. Montiel, K. Konolige: Double window optimisation for constant time visual SLAM, *Int. Conf. Computer Vis. (ICCV)* (2011)
- 46.88 A. Nüchter: 3D robot mapping, *Springer Tract. Adv. Robotics* **52** (2009)
- 46.89 U. Frese: Treemap: An $O(\log n)$ algorithm for indoor simultaneous localization and mapping, *Auton. Robots* **21**(2), 103–122 (2006)
- 46.90 G. Grisetti, C. Stachniss, S. Grzonka, W. Burgard: A tree parameterization for efficiently computing maximum likelihood maps using gradient descent, *Robotics Sci. Syst.* (2007)
- 46.91 E. Olson, J.J. Leonard, S. Teller: Fast iterative alignment of pose graphs with poor initial estimates, *Proc. IEEE Int. Conf. Robotics Autom. (ICRA)* (2006) pp. 2262–2269
- 46.92 T.M. Cover, J.A. Thomas: *Elements of Information Theory* (Wiley, New York 1991)
- 46.93 E. Olson, J.J. Leonard, S. Teller: Spatially-adaptive learning rates for online incremental SLAM, *Robotics Sci. Syst.* (2007)
- 46.94 G. Grisetti, D. Lordi Rizzini, C. Stachniss, E. Olson, W. Burgard: Online constraint network optimization for efficient maximum likelihood map learning, *Proc. IEEE Int. Conf. Robotics Autom. (ICRA)* (2008)
- 46.95 M. Kaess, A. Ranganathan, F. Dellaert: Fast incremental square root information smoothing, *Int. Jt. Conf. Artif. Intell. (ISCAI)* (2007)
- 46.96 P. Agarwal, E. Olson: Evaluating variable reordering strategies for SLAM, *Proc. IEEE/RSJ Int. Conf. Intel Robots Syst. (IROS)* (2012)
- 46.97 K. Ni, D. Steedly, F. Dellaert: Tectonic SAM: exact, out-of-core, submap-based SLAM, *Proc. IEEE Int. Conf. Robotics Autom. (ICRA)* (2007) pp. 1678–1685
- 46.98 G. Sibley, C. Mei, I. Reid, P. Newman: Adaptive relative bundle adjustment, *Robotics Sci. Syst.* (2009)
- 46.99 P.M. Newman, J.J. Leonard, R. Rikoski: Towards constant-time SLAM on an autonomous underwater vehicle using synthetic aperture sonar, *11th Int. Symp. Robotics Res.* (2003)
- 46.100 M.A. Paskin: Thin junction tree filters for simultaneous localization and mapping, *Int. Jt. Conf. Artif. Intell. (IJCAI)* (Morgan Kaufmann, New York 2003) pp. 1157–1164
- 46.101 S.B. Williams: Efficient Solutions to Autonomous Mapping and Navigation Problems, Ph.D. Thesis (ACFR Univ. Sydney, Sydney 2001)
- 46.102 H. Johannsson, M. Kaess, M.F. Fallon, J.J. Leonard: Temporally scalable visual SLAM using a reduced pose graph, *RSS Workshop Long-term Oper. Auton. Robotic Syst. Chang. Environ.* (2012)

- 46.103 N. Carlevaris-Bianco, R.M. Eustice: Generic factor-based node marginalization and edge sparsification for pose-graph SLAM, *Proc. IEEE Int. Conf. Robotics Autom. (ICRA)* (2013)
- 46.104 M. Kaess, F. Dellaert: Covariance recovery from a square root information matrix for data association, *J. Robotics Auton. Syst.* **57**(12), 1198–1210 (2009)
- 46.105 H. Kretzschmar, C. Stachniss: Information-theoretic compression of pose graphs for laser-based SLAM, *Int. J. Robotics Res.* **31**(11), 1219–1230 (2012)
- 46.106 U. Frese, L. Schröder: Closing a million-landmarks loop, *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)* (2006)
- 46.107 J. McDonald, M. Kaess, C. Cadena, J. Neira, J.J. Leonard: Real-time 6-DOF multi-session visual SLAM over large scale environments, *J. Robotics Auton. Syst.* **61**(10), 1144–1158 (2012)
- 46.108 H. Strasdat, J.M.M. Montiel, A.J. Davison: Real-time monocular SLAM: Why filter?, *Proc. IEEE Int. Conf. Robotics Autom. (ICRA)* (2010)
- 46.109 P. Henry, M. Krainin, E. Herbst, X. Ren, D. Fox: RGB-D mapping: Using depth cameras for dense 3D modeling of indoor environments, *Int. J. Robotics Res.* **31**(5), 647–663 (2012)
- 46.110 D. Nister, H. Stewenius: Scalable recognition with a vocabulary tree, *Proc. IEEE Int. Conf. Comput. Vis. Pattern Recognit. (ICCVPR)* (2006) pp. 2161–2168
- 46.111 R.A. Brooks: Aspects of mobile robot visual map making, *Proc. Int. Symp. Robotics Res. (ISRR)* (MIT Press, Cambridge 1984) pp. 287–293
- 46.112 H. Moravec: Obstacle Avoidance and Navigation in the Real World by a Seeing Robot Rover, Ph.D. Thesis (Stanford Univ., Stanford 1980)
- 46.113 N. Ayache, O. Faugeras: Building, registering, and fusing noisy visual maps, *Int. J. Robotics Res.* **7**(6), 45–65 (1988)
- 46.114 D. Kriegman, E. Triendl, T. Binford: Stereo vision and navigation in buildings for mobile robots, *IEEE Trans. Robotics Autom.* **5**(6), 792–803 (1989)
- 46.115 L. Matthies, S. Shafer: Error modeling in stereo navigation, *IEEE J. Robotics Autom.* **3**(3), 239–248 (1987)
- 46.116 S. Pollard, J. Porrill, J. Mayhew: Predictive feed-forward stereo processing, *Alvey Vis. Conf.* (1989) pp. 97–102
- 46.117 M. Smith, I. Baldwin, W. Churchill, R. Paul, P. Newman: The new college vision and laser data set, *Int. J. Robotics Res.* **28**(5), 595–599 (2009)
- 46.118 M. Cummins, P.M. Newman: Appearance-only SLAM at large scale with FAB-MAP 2.0, *Int. J. Robotics Res.* **30**(9), 1100–1123 (2010)
- 46.119 P.M. Newman, G. Sibley, M. Smith, M. Cummins, A. Harrison, C. Mei, I. Posner, R. Shade, D. Schroter, L. Murphy, W. Churchill, D. Cole, I. Reid: Navigating, recognising and describing urban spaces with vision and laser, *Int. J. Robotics Res.* **28**, 11–12 (2009)
- 46.120 G. Sibley, C. Mei, I. Reid, P. Newman: Vast-scale outdoor navigation using adaptive relative bundle adjustment, *Int. J. Robotics Res.* **29**(8), 958–980 (2010)
- 46.121 A. Davison, D. Murray: Mobile robot localisation using active vision, *Eur. Conf. Comput. Vis. (ECCV)* (1998) pp. 809–825
- 46.122 A.J. Davison, I. Reid, N. Molton, O. Stasse: MonoSLAM: Real-time single camera SLAM, *IEEE Trans., Pattern Anal. Mach. Intell.* **29**(6), 1052–1067 (2007)
- 46.123 R.O. Castle, G. Klein, D.W. Murray: Wide-area augmented reality using camera tracking and mapping in multiple regions, *Comput. Vis. Image Understand.* **115**(6), 854–867 (2011)
- 46.124 E. Eade, T. Drummond: Unified loop closing and recovery for real time monocular SLAM, *Br. Mach. Vis. Conf.* (2008)
- 46.125 E. Eade, P. Fong, M.E. Munich: Monocular graph SLAM with complexity reduction, *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)* (2010) pp. 3017–3024
- 46.126 K. Konolige, M. Agrawal: FrameSLAM: From bundle adjustment to real-time visual mapping, *IEEE Trans. Robotics* **24**(5), 1066–1077 (2008)
- 46.127 D. Nister, O. Naroditsky, J. Bergen: Visual odometry for ground vehicle applications, *J. Field Robotics* **23**(1), 3–20 (2006)
- 46.128 D. Scaramuzza, F. Fraundorfer: Visual odometry. Part I: The first 30 years and fundamentals, *IEEE Robotics Autom. Mag.* **18**(4), 80–92 (2011)
- 46.129 F. Fraundorfer, D. Scaramuzza: Visual odometry. Part II: Matching, robustness, optimization, and applications, *IEEE Robotics Autom. Mag.* **19**(2), 78–90 (2012)
- 46.130 A.S. Huang, A. Bachrach, P. Henry, M. Krainin, D. Maturana, D. Fox, N. Roy: Visual odometry and mapping for autonomous flight using an RGB-D camera, *Proc. Int. Symp. Robotics Res. (ISRR)* (2011)
- 46.131 J. Sivic, A. Zisserman: Video Google: A text retrieval approach to object matching in videos, *Int. Conf. Computer Vis. (ICCV)* (2003) p. 1470
- 46.132 M. Cummins, P.M. Newman: Probabilistic appearance based navigation and loop closing, *Proc. IEEE Int. Conf. Robotics Autom. (ICRA)* (2007) pp. 2042–2048
- 46.133 C. Cadena, D. Gálvez, F. Ramos, J.D. Tardós, J. Neira: Robust place recognition with stereo cameras, *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)* (2010)
- 46.134 R.A. Newcombe, A.J. Davison, S. Izadi, P. Kohli, O. Hilliges, J. Shotton, D. Molyneaux, S. Hodges, D. Kim, A. Fitzgibbon: Kinectfusion: Real-time dense surface mapping and tracking, *IEEE/ACM Int. Sym. Mixed Augment. Real. (ISMAR)* (2011) pp. 127–136
- 46.135 J. Neira, A.J. Davison, J.J. Leonard: Guest editorial special issue on visual SLAM, *IEEE Trans. Robotics* **24**(5), 929–931 (2008)
- 46.136 K. Konolige, J. Bowman: Towards lifelong visual maps, *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)* (2009) pp. 1156–1163
- 46.137 K. Konolige, J. Bowman, J.D. Chen, P. Mihe-lich, M. Calonder, V. Lepetit, P. Fua: View-

- based maps, *Int. J. Robotics Res.* **29**(8), 941–957 (2010)
- 46.138 G. Sibley, C. Mei, I. Reid, P. Newman: Planes, trains and automobiles – Autonomy for the modern robot, *Proc. IEEE Int. Conf. Robotics Autom. (ICRA)* (2010) pp. 285–292
- 46.139 T. Whelan, H. Johannsson, M. Kaess, J.J. Leonard, J.B. McDonald: Robust real-time visual odometry for dense RGB-D mapping, *IEEE Int. Conf. Robotics Autom. (ICRA)* (2013)
- 46.140 R.A. Newcombe, S.J. Lovegrove, A.J. Davison: DTAM: Dense tracking and mapping in real-time, *Int. Conf. Computer Vis. (ICCV)* (2011) pp. 2320–2327
- 46.141 F. Steinbruecker, J. Sturm, D. Cremers: Real-time visual odometry from dense RGB-D images, *Workshop Live Dense Reconstr. Mov. Cameras Int. Conf. Comput. Vis. (ICCV)* (2011)
- 46.142 H. Roth, M. Vona: Moving volume kinectfusion, *Br. Mach. Vis. Conf.* (2012)

Attachment J

[Home](#) > [Springer Handbook of Robotics](#) > Chapter

Simultaneous Localization and Mapping

| Chapter | First Online: 01 January 2016

| pp 1153–1176 | [Cite this chapter](#)

✓ Access provided by University of Nebraska–Lincoln

Download book PDF

Download book EPUB 



Springer Handbook of Robotics

[Cyrill Stachniss](#) , [John J. Leonard](#) & [Sebastian Thrun](#)

 Part of the book series: [Springer Handbooks](#) ((SHB))

 100k Accesses  52 Citations  2 Altmetric

Abstract

This chapter provides a comprehensive introduction in to the *simultaneous localization and mapping problem*, better known in its abbreviated form as

SLAM

. SLAM addresses the main perception problem of a robot navigating an unknown environment. While navigating the environment, the robot seeks to acquire a map thereof, and at the same time it wishes to localize itself using its map. The use of SLAM problems can be motivated in two different ways: one might be interested in detailed environment models, or one might seek to maintain an accurate sense of a mobile robot's location. SLAM serves both of these purposes.

We review the three major paradigms from which many published methods for SLAM are derived: (1) the extended Kalman filter (

EKF

); (2) particle filtering; and (3) graph optimization. We also review recent work in three-dimensional (3-D

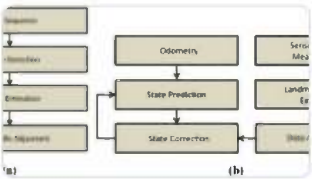
) SLAM using visual and red green blue distance-sensors (RGB-D

), and close with a discussion of open research problems in robotic mapping.



 Access provided by University of Nebraska–Lincoln. [Download chapter PDF](#) 

Similar content being viewed by others



An Overview to Visual Odometry and Visual SLAM: Applications to Mobile Robotics

Article | 13 November 2015



Advances in Simultaneous Localization and Mapping (SLAM) for Autonomous Mobile Robot Navigation

Chapter | © 2024



Extended Kalman Filter (EKF)-Based Local SLAM in Dynamic Environments: A Framework

Chapter | © 2016

Keywords

[Particle Filter](#) [Extended Kalman Filter](#) [Data Association](#) [Bundle Adjustment](#) [Robot Location](#)

These keywords were added by machine and not by the authors. This process is experimental and the keywords may be updated as the learning algorithm improves.

This chapter provides a comprehensive introduction into one of the key enabling technologies of mobile robot navigation: *simultaneous localization and mapping*, or in short *SLAM*. SLAM addresses the problem of acquiring a spatial map of an environment while simultaneously localizing the robot relative to this model. The SLAM problem is generally regarded as one of the most important problems in the pursuit of building truly autonomous mobile robots. It is of great practical importance; if a robust, general-purpose solution to SLAM can be found, then many new applications of mobile robotics will become possible.

While the problem is deceptively easy to state, it presents many challenges, despite significant progress made in this area. At present, we have robust methods for mapping environments that are mainly static, structured, and of limited size. Robustly mapping unstructured, dynamic, and large-scale environments in an online fashion remains largely an open research problem.

The historical roots of methods that can be applied to address the SLAM problem can be traced back to Gauss [46.1], who is largely credited for inventing the least squares method. In the Twentieth Century, a number of fields outside robotics have studied the making of environment models from a moving sensor platform, most notably in *photogrammetry* [46.2, 46.3, 46.4] and *computer vision* [46.5]. Strongly related problems in these fields are bundle adjustment and structure from motion. SLAM builds on this work, often extending the basic paradigms into more scalable algorithms. Modern SLAM systems often view the estimation problem as solving a sparse graph of constraints and applying nonlinear optimization to compute the map and the trajectory of the robot. As we strive to enable long-lived autonomous robots, an emerging challenge is to handle massive sensor data streams.

This chapter begins with a definition of the SLAM problem, which shall include a brief taxonomy of different versions of the problem. The centerpiece of this chapter is a layman introduction into the three major paradigms in this field, and the various extensions that exist. As the reader will quickly recognize, there is no single best solution to the SLAM method. The method chosen by the practitioner will depend on a number of factors, such as the desired map resolution, the update time, and the nature of the features in the map, and so on. Nevertheless, the three methods discussed in this chapter cover the major paradigms in this field.

For more a detailed treatment of SLAM, we refer the reader to Durrant-Whyte and Bailey [46.6, 46.7], who provide an in-depth tutorial for SLAM, Grisetti et al. for a tutorial on graph-based SLAM [46.8], and Thrun et al., which dedicates a number of chapters to the topic of SLAM [46.9].

46.1 SLAM: Problem Definition

The SLAM problem is defined as follows: A mobile robot roams an unknown environment, starting at an initial location x_0 . Its motion is uncertain, making it gradually more difficult to determine its current pose in global coordinates. As it roams, the robot can sense its environment with a noisy sensor. The SLAM problem is the problem of building a map of the environment while simultaneously determining the robot's position relative to this map given noisy data.

46.1.1 Mathematical Basis

Formally, SLAM is best described in probabilistic terminology. Let us denote time by t , and the robot location by x_t . For mobile robots on a flat ground, x_t is usually a three-dimensional vector, comprising its two-dimensional (

2-D

) coordinate in the plane plus a single rotational value for its orientation. The sequence of locations, or *path*, is then given as

$$X_T = \{x_0, x_1, x_2, \dots, x_T\} .$$

(46.1)

Here T is some terminal time (T might be ∞). The initial location x_0 often serves as a point of reference for the estimation algorithm; other positions cannot be sensed.

Odometry provides relative information between two consecutive locations. Let u_t denote the odometry that characterized the motion between time $t - 1$ and time t ; such data might be obtained from the robot's wheel encoders or from the controls given to those motors. Then the sequence

$$U_T = \{u_1, u_2, u_3, \dots, u_T\}$$

(46.2)

characterizes the relative motion of the robot. For noise-free motion, U_T would be sufficient to recover the poses from the initial location x_0 . However, odometry measurements are noisy, and path integration techniques inevitably diverge from the truth.

Finally, the robot senses objects in the environment. Let m denote the *true* map of the environment. The environment may be comprised of landmarks, objects, surfaces, etc., and m describes their locations. The environment map m is often assumed to be time-invariant, i. e., static.

The robot measurements establish information between features in m and the robot location x_t . If we, without loss of generality, assume that the robot takes exactly one measurement at each point in time, the sequence of measurements is given as

$$Z_T = \{z_1, z_2, z_3, \dots, z_T\} .$$

(46.3)

Figure 46.1 illustrates the variables involved in the SLAM problem. It shows the sequence of locations and sensor measurements, and the causal relationships between these variables. This diagram represents a *graphical model* for SLAM. It is useful in understanding the dependencies in the problem at hand.

Fig. 46.1

Graphical model of the SLAM problem. Arcs indicate causal relationships, and shaded nodes are directly observable to the robot. In SLAM, the robot seeks to recover the unobservable variables

The SLAM problem is now the problem of recovering a model of the world m and the sequence of robot locations X_T from the odometry and measurement data. The literature distinguishes two main forms of the SLAM problem, which are both of equal practical importance. One is known as the *full SLAM problem*: it involves estimating the posterior over the entire robot path together with the map

$$p(X_T, m \mid Z_T, U_T) .$$

(46.4)

Written in this way, the full SLAM problem is the problem of calculating the joint posterior probability over X_T and m from the available data. Notice that the variables right of the conditioning bar are all directly observable to the robot, whereas those on the left are the ones that we want. As we shall see, algorithms for the full SLAM problem are often batch, that is, they process all data at the same time.

The second, equally important SLAM problem is the *online SLAM problem*. This problem is defined via

$$p(x_t, m \mid Z_t, U_t) .$$

(46.5)

Online SLAM seeks to recover the present robot location, instead of the entire path. Algorithms that address the online problem are usually incremental and can process one data item at a time. In the literature, such algorithms are typically called *filters*.

To solve the SLAM problem, the robot needs to be endowed with two more models: a mathematical model that relates odometry measurements u_t to robot locations x_{t-1} and x_t ; and a model that relates measurements z_t to the environment m and the robot location x_t . These models correspond to the arcs in Fig. [46.1](#).

In SLAM, it is common to think of those mathematical models as probability distributions: $p(x_t \mid x_{t-1}, u_t)$ characterizes the probability distribution of the location x_t assuming that a robot started at a known location x_{t-1} and measured the odometry data u_t . And likewise, $p(z_t \mid x_t, m)$ is the probability for measuring z_t if this measurement is taken at a known location x_t in a known environment m . Of course, in the SLAM problem we do *not* know the robot location, and neither do we know the environment. As we shall see, Bayes rule takes care of this, by transforming

these mathematical relationships into a form where we can *recover* probability distributions over those latent variables from the measured data.

46.1.2 Example: SLAM in Landmark Worlds

One common setting of SLAM involves an assumption that the environment is populated by point-landmarks. When building 2-D maps, point-landmarks may correspond to door posts and corners of rooms, which, when projected into a 2-D map are characterized by a point coordinate. In a 2-D world, each point-landmark is characterized by two coordinate values. Hence the world is a vector of size $2N$, where N is the number of point-landmarks in the world. In a commonly studied setting, the robot can sense three things: the relative range to nearby landmarks, their relative bearing, and the identity of these landmarks. The range and bearing may be noisy, but in the most simple case the identity of the sensed landmarks is known perfectly. Determining the identity of the sensed landmarks is also known as the *data association problem*. In practice, it is one of the most difficult problems in SLAM.

To model the above described setup, one begins with defining the *exact*, noise-free measurement function. The measurement function h describes the workings of the sensors: it accepts as input a description of the environment m and a robot location x_t , and it computes the measurement

$$h(x_t, m) .$$

(46.6)

Computing h is straightforward in our simplified landmark setting; it is a simple exercise in trigonometry. The probabilistic measurement model can be derived from this measurement function by adding a noise term. It is a probability distribution that peaks at the noise-free value $h(x_t, m)$ but allows for measurement noise, for example,

$$p(z_t | x_t, m) = N(h(x_t, m), Q_t) .$$

(46.7)

Here N denotes the 2-D normal distribution, which is centered at $h(x_t, m)$. The 2-by-2 matrix Q_t is the noise covariance, indexed by time.

The motion model is derived from a kinematic model of robot motion. Given the location vector x_{t-1} and the motion u_t , textbook kinematics tells us how to calculate x_t . Let this function be denoted by g

$$g(x_{t-1}, u_t) .$$

(46.8)

The motion model may then be defined by a normal distribution centered at $g(x_{t-1}, u_t)$ but subject to Gaussian noise

$$p(x_t | x_{t-1}, u_t) = N(g(x_{t-1}, u_t), R_t) .$$

(46.9)

Here $\mathbf{R}_t$ is a covariance. It is of size 3-by-3, since the location is a three-dimensional 3-D vector.

With these definitions, we have all we need to develop a SLAM algorithm. While in the literature, point-landmark problems with range-bearing sensing are by far the most studied, SLAM algorithms are not confined to landmark worlds. But no matter what the map representation and the sensor modality, any SLAM algorithm needs a similarly crisp definition of the features in m , the measurement model $p(z_t | x_t, m)$, and the motion model $p(x_t | x_{t-1}, u_t)$. Note that none of those distributions has to be restricted to Gaussian noise as done in the example above.

46.1.3 Taxonomy of the SLAM Problem

SLAM problems are distinguished along a number of different dimensions. Most important research papers identify the type of problems addressed by making the underlying assumptions explicit. We already encountered one such distinction: full versus online. Other common distinctions are as follows:

46.1.3.1 Volumetric Versus Feature-Based

In volumetric SLAM, the map is sampled at a resolution high enough to allow for photo-realistic reconstruction of the environment. The map m in volumetric SLAM is usually quite high-dimensional, with the result that the computation can be quite involved. Feature-based SLAM extracts sparse features from the sensor stream. The map is then only comprised of features. Our point-landmark example is an instance of feature-based SLAM. Feature-based SLAM techniques tend to be more efficient, but their results may be inferior to volumetric SLAM due to the fact that the extraction of features discards information in the sensor measurements.

46.1.3.2 Topological Versus Metric

Some mapping techniques recover only a qualitative description of the environment, which characterizes the relation of basic locations. Such methods are known as topological. A topological map might be defined over a set of distinct places and a set of qualitative relations between these places (e. g., place A is adjacent to place B). Metric SLAM methods provide metric information between the relation of such places. In recent years, topological methods have fallen out of fashion, despite ample evidence that humans often use topological information for navigation.

46.1.3.3 Known Versus Unknown Correspondence

The correspondence problem is the problem of relating the identity of sensed things to other sensed things. In the landmark example above, we assumed that the identity of landmarks is known. Some SLAM algorithms make such an assumption, others do not. The ones that do not provide special mechanisms for estimating the correspondence of measured features to previously observed landmarks in the map. The problem of estimating the correspondence is known as *data association problem*. It is one of the most difficult problems in SLAM.

46.1.3.4 Static Versus Dynamic

Static SLAM algorithms assume that the environment does not change over time. Dynamic methods allow for changes in the environment. The vast literature on SLAM assumes static environments. Dynamic effects are often treated just as measurement outliers. Methods that reason about motion in the environment are more involved, but they tend to be more robust in most applications.

46.1.3.5 Small Versus Large Uncertainty

SLAM problems are distinguished by the degree of location uncertainty that they can handle. The most simple SLAM algorithms allow only for small errors in the location estimate. They are good for situations in which a robot goes down a path that does not intersect itself, and then returns along the same path. In many environments it is possible to reach the same location from multiple directions. Here the robot may accrue a large amount of uncertainty. This problem is known as the *loop closing problem*. When closing a loop, the uncertainty may be large. The ability to close loops is a key characteristic of modern-day SLAM algorithms. The uncertainty can be reduced if the robot can sense information about its position in some absolute coordinate frame, e. g., through the use of a satellite-based global positioning system (

GPS

) receiver.

46.1.3.6 Active Versus Passive

In passive SLAM algorithms, some other entity controls the robot, and the SLAM algorithm is purely observing. The vast majority of algorithms are of this type; they give the robot designer the freedom to implement arbitrary motion controllers, and pursue arbitrary motion objectives. In active SLAM, the robot actively explores its environment in the pursuit of an accurate map. Active SLAM methods tend to yield more accurate maps in less time, but they constrain the robot motion. There exist hybrid techniques in which the SLAM algorithm controls only the pointing direction of the robot's sensors, but not the motion direction.

46.1.3.7 Single-Robot Versus Multi-Robot

Most SLAM problems are defined for a single robot platform, although recently the problem of multi-robot exploration has gained in popularity. Multi-robot SLAM problems come in many flavors. In some, robots get to observe each other, in others, robots are told their relative initial locations. Multirobot SLAM problems are also distinguished by the type of communication allowed between the different robots. In some, the robots can communicate with no latency and infinite bandwidth. More realistic are setups in which only nearby robots can communicate, and the communication is subject to latency and bandwidth limitations.

46.1.3.8 Any-Time and Any-Space

Robots that do all computations onboard have limited resources in memory and computation power. Any-time and any-space SLAM systems are an alternative to traditional methods. They enable the robot to compute a solution given the resource constraints of the system. The more resources available, the better the solution.

As this taxonomy suggests, there exists a flurry of SLAM algorithms. Most modern-day conferences dedicate multiple sessions to SLAM. This chapter focuses on the very basic SLAM setup. In particular it assumes a static environment with a single robot. Extensions are discussed towards the end of this chapter, in which the relevant literature is discussed.

46.2 The Three Main SLAM Paradigms

This section reviews three basic SLAM paradigms, from which most others are derived. The first, known as EKF SLAM, is in robotics historically the earliest but has become less popular due to its limiting computational properties

and issues resulting from performing single linearizations only. The second approach uses nonparametric statistical filtering techniques known as particle filters. It is a popular method for online SLAM and provides a perspective on addressing the data association problem in SLAM. The third paradigm is based on graphical representations and successfully applies sparse nonlinear optimization methods to the SLAM problem. It is the main paradigm for solving the full SLAM problem and recently also incremental techniques are available.

46.2.1 Extended Kalman Filters

Historically, the EKF formulation of SLAM is the earliest, and perhaps the most influential, SLAM algorithm. EKF SLAM was introduced in [46.10, 46.11] and [46.12, 46.13], which were the first papers to propose the use of a single state vector to estimate the locations of the robot and a set of features in the environment, with an associated error covariance matrix representing the uncertainty in these estimates, including the correlations between the vehicle and feature state estimates. As the robot moves through its environment taking measurements, the system state vector and covariance matrix are updated using the extended Kalman filter [46.14, 46.15]. As new features are observed, new states are added to the system state vector; the size of the system covariance matrix grows quadratically.

This approach assumes a metrical, feature-based environmental representation, in which objects can be effectively represented as points in an appropriate parameter space. The position of the robot and the locations of features form a network of uncertain spatial relationships. The development of appropriate representations is a critical issue in SLAM, and intimately related to the topics of sensing and world modeling discussed in Chap. 36 and in Part C.

The EKF algorithm represents the robot estimate by a multivariate Gaussian

$$p(x_t, m \mid Z_t, U_t) = N(\mu_t, \Sigma_t) .$$

(46.10)

The high-dimensional vector μ_t contains the robot's best estimate of its own current location x_t and the location of the features in the environment. In our point-landmark example, the dimension of μ_t would be $3 + 2N$, since we need three variables to represent the robot location and $2N$ variables for the N landmarks in the map.

The matrix Σ_t is the covariance of the robot's assessment of its expected error in the guess μ_t . The matrix Σ_t is of size $(3 + 2N) \times (3 + 2N)$ and it is positive semi-definite. In SLAM, this matrix is usually dense. The off-diagonal elements capture the correlations in the estimates of different variables. Nonzero correlations come along because the robot's location is uncertain, and as a result the locations of the landmarks in the maps are uncertain.

The EKF SLAM algorithm is easily derived for our point-landmark example. Suppose, for a moment, the motion function g and the measurement function h were *linear* in their arguments. Then, the vanilla Kalman filter, as described in any textbook on Kalman filtering, would be applicable. EKF SLAM linearizes the functions g and h using Taylor series expansion. In its most basic form and in the absence of any data association problems, EKF SLAM is basically the application of the EKF to the online SLAM problem.

Figure 46.2 illustrates the EKF SLAM algorithm for an artificial example. The robot navigates from a start pose that serves as the origin of its coordinate system. As it moves, its own pose uncertainty increases, as indicated by

uncertainty ellipses of growing diameter. It also senses nearby landmarks and maps them with an uncertainty that combines the fixed measurement uncertainty with the increasing pose uncertainty. As a result, the uncertainty in the landmark locations grows over time. The interesting transition happens in Fig. 46.2d: Here the robot observes the landmark it saw in the very beginning of mapping, and whose location is relatively well known. Through this observation, the robot's pose error is reduced, as indicated in Fig. 46.2d – notice the very small error ellipse for the final robot pose. This observation also reduces the uncertainty for other landmarks in the map. This phenomenon arises from a correlation that is expressed in the covariance matrix of the Gaussian posterior. Since most of the uncertainty in earlier landmark estimates is caused by the robot pose, and since this very uncertainty persists over time, the location estimates of those landmarks are correlated. When gaining information on the robot's pose, this information spreads to previously observed landmarks. This effect is probably the most important characteristic of the SLAM posterior [46.16]. Information that helps localize the robot is propagated through the map, and as a result improves the localization of other landmarks in the map.

Fig. 46.2a–d

EKF applied to the online SLAM problem. The robot's path is a *dotted line*, and its estimates of its own position are *shaded ellipses*. Eight distinguishable landmarks of unknown location are shown as *small dots*, and their location estimates are shown as *white ellipses*. In (a–c) the robot's positional uncertainty is increasing, as is its uncertainty about the landmarks it encounters. In (d) the robot senses the first landmark again, and the uncertainty of *all* landmarks decreases, as does the uncertainty of its current pose (image courtesy of Michael Montemerlo, Stanford University)

With a few adaptations, EKF SLAM can also be applied in the presence of *uncertain data association*. If the identity of observed features is unknown, the basic EKF idea becomes inapplicable. The solution here is to reason about the most likely data association when a landmark is observed. This is usually done based on proximity: *which of the landmarks in the map corresponds most likely to the landmark just observed?* The proximity calculation considers the measurement noise and the actual uncertainty in the poster estimate, and the metric used in this calculation is known as a Mahalanobis distance, which is a weighted quadratic distance. To minimize the chances of false data associations, many implementations use visible features to distinguish individual landmarks and associate groups of landmarks observed simultaneously [46.17, 46.18], although distinct features can also be computed from laser data [46.19, 46.20]. Typical implementations also maintain a *provisional landmark list* and only add landmarks to the internal map when they have been observed sufficiently frequently [46.16, 46.21]. With an appropriate landmark definition and

careful implementation of the data association step, EKF SLAM has been applied successfully in a wide range of environments, using airborne, underwater, indoor, and various other platforms.

The basic formulation of EKF SLAM assumes that the location of features in the map is fully observable from a single position of the robot. The method has been extended to situations with partial observability, with range-only [46.22] or angle-only [46.23, 46.24] measurements. The technique has also been utilized using a feature-less representation, in which the state consists of current and past robot poses, and measurements take the form of constraints between the poses (derived for example from laser scan matching or from camera measurements) [46.25, 46.26].

A key concern of the EKF approach to SLAM lies in the quadratic nature of the covariance matrix. A number of researchers have proposed extensions to the EKF SLAM algorithms that achieve scalability, for example through submap decomposition [46.27, 46.28, 46.29, 46.30]. A related family of approaches [46.31, 46.32, 46.33, 46.34] employs the Extended Information Filter, which operates on the inverse of the covariance matrix. A key insight is that whereas the EKF covariance is densely populated, the information matrix is sparse when the full robot trajectory is maintained, leading to the development of efficient algorithms and providing a conceptual link to the pose graph optimization methods described in Sect. 46.2.3.

The issues of consistency and convergence in EKF SLAM have been investigated in [46.35, 46.36]. Observability-based rules for designing consistent EKF SLAM estimators are presented in [46.37].

46.2.2 Particle Methods

The second principal SLAM paradigm is based on particle filters. Particle filters can be traced back to [46.38], but they have become popular only in the last two decades. Particle filters represent a posterior through a set of *particles*. For the novice in SLAM, each particle is best thought as a concrete guess as to what the true value of the state may be. By collecting many such guesses into a set of guesses, or set of particles, the particle filter approximates the posterior distribution. Under mild conditions, the particle filter has been shown to approach the true posterior as the particle set size goes to infinity. It is also a nonparametric representation that represents multimodal distributions with ease.

The key problem with the particle filter in the context of SLAM is that the space of maps and robot paths is huge. Suppose we have a map with 100 features. How many particles would it take to populate that space? In fact, particle filters scale exponentially with the dimension of the underlying state space. Three or four dimensions are thus acceptable, but 100 dimensions are generally not.

The trick to make particle filters amenable to the SLAM problem goes back to [46.39, 46.40] and is known as Rao–Blackwellization. It has been introduced into the SLAM literature in [46.41], followed by [46.42], who coined the name

fastSLAM

(fast simultaneous localization and mapping). Let us first explain the basic FastSLAM algorithm on the simplified point-landmark example, and then discuss the justification for this approach.

At any point in time, FastSLAM maintains K particles of the type

$$X_t^{[k]}, \mu_{t,1}^{[k]}, \dots, \mu_{t,N}^{[k]}, \Sigma_{t,1}^{[k]}, \dots, \Sigma_{t,N}^{[k]}.$$

(46.11)

Here $[k]$ is the index of the sample. This expression states that a particle contains:

- A sample path $X_t^{[k]}$, and
- A set of N 2-D Gaussians with means $\mu_{t,n}^{[k]}$ and variances $\Sigma_{t,n}^{[k]}$, one for each landmark in the environment.

Here n is the index of the landmark (with $1 \leq n \leq N$). From that it follows that K particles possess K path samples. It also possesses KN Gaussians, each of which models exactly one landmark for one of the particles.

Initializing FastSLAM is simple: just set each particle's robot location to the starting coordinates, typically $(0, 0, 0)^T$, and zero the map. The particle update then proceeds as follows:

- When an odometry reading is received, new location variables are generated stochastically, one for each of the particles. The distribution for generating those location particles is based on the motion model

$$x_t^{[k]} \approx p(x_t | x_{t-1}^{[k]}, u_t).$$

(46.12)

Here $x_{t-1}^{[k]}$ is the previous location, which is part of the particle. This probabilistic sampling step is easily implemented for any robot whose kinematics can be computed.

- When a measurement z_t is received, two things happen: first, FastSLAM computes for each particle the probability of the new measurement z_t . Let the index of the sensed landmark be n . Then the desired probability is defined as follows

$$w_t^{[k]} = N(z_t | x_t^{[k]}, \mu_{t,n}^{[k]}, \Sigma_{t,n}^{[k]}).$$

(46.13)

The factor $w_t^{[k]}$ is called the *importance weight*, since it measures how *important* the particle is in the light of the new sensor measurement. As before, N denotes the normal distribution, but this time it is calculated for a specific value, z_t . The importance weights of all particles are then normalized so that they sum to 1.

Next, FastSLAM draws with replacement from the set of existing particles a set of new particles. The probability of drawing a particle is its normalized importance weight. This step is called *resampling*. The intuition behind resampling is that particles for which the measurement is more plausible have a higher chance of surviving the resampling process.

Finally, FastSLAM updates for the new particle set the mean $\mu_{t,n}^{[k]}$ and covariance $\Sigma_{t,n}^{[k]}$, based on the measurement z_t . This update follows the standard EKF update rules – note that the extended Kalman filters maintained in FastSLAM are, in contrast to EKF SLAM, all low-dimensional (typically 2-D).

This all may sound complex, but FastSLAM is quite easy to implement. Sampling from the motion model usually involves simple kinematic calculations. Computing the importance of a measurement is often straightforward too, especially for Gaussian measurement noise. And updating a low-dimensional particle filter is also not complicated.

FastSLAM has been shown to approximate the full SLAM posterior. The derivation of FastSLAM exploits three techniques: Rao–Blackwellization, conditional independence, and resampling. Rao–Blackwellization is the following concept. Suppose we would like to compute a probability distribution $p(a, b)$, where a and b are arbitrary random variables. The vanilla particle filter would draw particles from the joint distributions, that is, each particle would have a value for a and one for b . However, if the conditional $p(b | a)$ can be described in closed form, it is equally legitimate to just draw particles from $p(a)$, and attach to each particle a closed-form description of $p(b | a)$. This trick is known as Rao–Blackwellization, and it yields better results than sampling from the joint. FastSLAM applies this technique, in that it samples from the path posterior $p(X_t^{[k]} | U_t, Z_t)$ and represents the map $p(m | X_t^{[k]}, U_t, Z_t)$ in Gaussian form.

FastSLAM also breaks down the posterior over maps (conditioned on paths) into sequences of low-dimensional Gaussians. The justification for this decomposition is subtle. It arises from a specific conditional independence assumption that is native to SLAM. Fig. 46.3 illustrates the concept graphically. In SLAM, knowledge of the robot path renders all landmark estimates independent. This is easily shown for the graphical network in Fig. 46.3: we find that if we remove the path variables from Fig. 46.3, then the landmark variables are all disconnected [46.43]. Thus, in SLAM *any dependence between multiple landmark estimates is mediated through the robot path*. This subtle but important observation implies that even if we used a large, monolithic Gaussian for the entire map (one per particle, of course), the off-diagonal element between different landmarks would simply remain zero. It is therefore legitimate to implement the map more efficiently, using N small Gaussians, one for each landmark. This explains the efficient map representation in FastSLAM.

Fig. 46.3

The SLAM problem depicted as Bayes network graph. The robot moves from location x_{t-1} to location x_{t+2} , driven by a sequence of controls. At each location x_t it observes a nearby feature in the map $m = \{m_1, m_2, m_3\}$. This graphical network illustrates that the location variables *separate* the individual features in the map from each other. If the locations are known, there remains no other path involving variables whose value is not known, between any two features in the map. This lack of a path renders the posterior of any two features in the map conditionally independent (given the locations)

Figure 46.4 shows results for a point-feature problem; here the point features are the centers of tree trunks as observed by an outdoor robot. The dataset used here is known as the Victoria Park dataset [46.44]. Fig. 46.4a shows the path of the vehicle obtained by integrating the vehicle controls, without perception. As can be seen, controls are a poor predictor of location for this vehicle; after 30 min of driving, the estimated position of the vehicle is well over 100 m away from its GPS position.

Fig. 46.4

(a) Vehicle path predicted by the odometry; (b) True path (*dashed line*) and FastSLAM 1.0 path (*solid line*); (c) Victoria Park results overlaid on aerial imagery with the GPS path in *blue (dashed)*, average FastSLAM 1.0 path in *yellow (solid)*, and estimated features as *yellow dots* (data and aerial image courtesy of José Guivant and Eduardo Nebot, Australian Centre for Field Robotics)

The FastSLAM algorithm has a number of interesting properties. First, it solves both full and online SLAM problems. Each particle has a sample of an entire path but the actual update equation only uses the most recent pose. This makes FastSLAM a filter. Second, FastSLAM can maintain multiple data association hypotheses. It is straightforward to make data association decisions on a per-particle basis, instead of having to adopt the same hypothesis for the entire filter. While we will not give any mathematical justification, we note that the resulting FastSLAM algorithm can even deal with unknown data association – something that the extended Kalman filter cannot claim. And third, FastSLAM can be implemented very efficiently using advanced tree methods to represent the map estimates, the update can be performed in time logarithmic in the size of the map N , and linear in the number of particles M .

FastSLAM has been extended in several ways. One set of variants are grid-based versions of FastSLAM, in which the Gaussians used to model point landmarks are replaced by an occupancy grid map [46.45, 46.46, 46.47]. The variant of [46.46] is illustrated in Fig. 46.5.

Fig. 46.5

Occupancy grid map generated from laser range data and based on pure odometry (image courtesy of Dirk Hähnel, University of Freiburg)

Figure 46.6 illustrates a simplified situation with three particles just before closing a large loop. The three different particles each stand for different paths, and they also possess their own local maps. When the loop is closed importance

resampling selects those particles whose maps are most consistent with the measurement. A resulting large-scale map is shown in Fig. 46.5. Further extensions can be found in [46.48, 46.49], whose methods are called DP-SLAM and operate on ancestry trees to provide efficient tree update methods for grid-based maps. Related to that, approximations to FastSLAM in which particles share their maps have been proposed [46.50].

Fig. 46.6

Application of the grid-based variant of the FastSLAM algorithm. Each particle carries its own map and the importance weights of the particles are computed based on the likelihood of the measurements given the particle's own map

The works in [46.45, 46.47, 46.51] provide ways to incorporate new observations into the location sampling process for landmarks and grid maps, based on prior work in [46.52]. This leads to an improved sampling process

$$x_t^{[k]} \approx \frac{p(z_t | m_{t-1}^{[k]}, x_t) p(x_t | x_{t-1}^{[k]}, u_t)}{p(z_t | m_{t-1}^{[k]}, x_{t-1}^{[k]}, u_t)},$$

(46.14)

which incorporates the odometry and the observation at the same time. Using an improved proposal distribution leads to more accurately sampled locations. This in turn leads to more accurate maps and requires a smaller number of particles compared to approaches using the sampling process given in (46.12). This extension makes FastSLAM and especially its grid-based variants robust tools for addressing the SLAM problem.

Finally, there are approaches that aim to overcome the assumption that the observations show Gaussian characteristics. As shown in [46.47], there are several situations in which the model is nonGaussian and also multimodal. A sum of Gaussians model on a per-particle bases, however, can be efficiently considered in the particle filter and it eliminates this problem in practice without introducing additional computational demands.

The so-far developed particle filters-based SLAM systems suffer from two problems. First, the number of samples that are required to compute consistent maps is often set manually by making an *educated guess*. The larger the uncertainty that the filter needs to represent during mapping, the more critical becomes this parameter. Second, nested loops combined with extensive re-visits of previously mapped areas can lead to particle depletion, which in turn may prevent the system from estimating a consistent map. Adaptive resampling strategies [46.45], particles

sharing maps [46.50], or filter backup approaches [46.53] improve the situation but cannot eliminate this problem in general.

46.2.3 Graph-Based Optimization Techniques

A third family of algorithms solves the SLAM problem through nonlinear sparse optimization. They draw their intuition from a graphical representation of the SLAM problem and the first working solution in robotics was proposed in [46.54]. The graph-based representation used here is closely related to a series of papers [46.55, 46.56, 46.57, 46.58, 46.59, 46.60, 46.61, 46.62, 46.63, 46.64]. We note that most of the earlier techniques are offline and address the full SLAM problem. In more recent years, new incremental versions that effectively re-use the previously computed solution have been proposed such as [46.65, 46.66, 46.67].

The basic intuition of graph-based SLAM is as follows. Landmarks and robot locations can be thought of as nodes in a graph. Every consecutive pair of locations x_{t-1}, x_t is tied together by an edge that represents the information conveyed by the odometry reading u_t . Further edges exist between the nodes that correspond to locations x_t and landmarks m_i , assuming that at time t the robot sensed landmark i . Edges in this graph are soft constraints. Relaxing these constraints yields the robot's best estimate for the map and the full path.

The construction of the graph is illustrated in Fig. 46.7. Suppose at time $t = 1$, the robot senses landmark m_1 . This adds an arc in the (yet highly incomplete) graph between x_1 and m_1 . When caching the edges in a matrix format (which happens to correspond to a quadratic equation defining the resulting constraints), a value is added to the elements between x_1 and m_1 , as shown on the right hand side of Fig. 46.7a.

Fig. 46.7a–c

Illustration of the graph construction. The (a) diagram shows the graph, the (b) the constraints in matrix form. (a) Observation is landmark m_1 . (b) Robot motion from x_1 to x_2 . (c) Several steps later

Now suppose the robot moves. The odometry reading u_2 leads to an arc between nodes x_1 and x_2 , as shown in Fig. 46.7b. Consecutive application of these two basic steps leads to a graph of increasing size, as illustrated in Fig. 46.7c. Nevertheless this graph is *sparse*, in that each node is only connected to a small number of other nodes

(assuming a sensor with limited sensing range). The number of constraints in the graph is (at worst) *linear* in the time elapsed and in the number of nodes in the graph.

If we think of the graph as a spring-mass model [46.60], computing the SLAM solution is equivalent to computing the state of minimal energy of this model. To see, we note that the graph corresponds to the log-posterior of the full SLAM problem (46.4)

$$\log p(X_T, m \mid Z_T, U_T) . \quad (46.15)$$

Without derivation, we state that this logarithm is of the form

$$\log p(X_T, m \mid Z_T, U_T) = \text{const} + \sum_t \log p(x_t \mid x_{t-1}, u_t) + \sum_t \log p(z_t \mid x_t, m) , \quad (46.16)$$

assuming independence between the individual observations and odometry readings. Each constraint of the form $\log p(x_t \mid x_{t-1}, u_t)$ is the result of exactly one robot motion event, and it corresponds to an edge in the graph. Likewise, each constraint of the form $\log p(z_t \mid x_t, m)$ is the result of one sensor measurement, to which we can also find a corresponding edge in the graph. The SLAM problem is then simply to find the mode of this equation, i. e.,

$$X_T^*, m^* = \operatorname{argmax}_{X_T, m} \log p(X_T, m \mid Z_T, U_T) . \quad (46.17)$$

Without derivation, we note that under the Gaussian noise assumptions, which was made in the point-landmark example, this expression resolves to the following quadratic form

$$\log p(X_T, m \mid Z_T, U_T) = \text{const} + \sum_t \underbrace{[x_t - g(x_{t-1}, u_t)]^T R_t^{-1} [x_t - g(x_{t-1}, u_t)]}_{\text{odometry reading}} + \sum_t \underbrace{[z_t - h(x_t, m)]^T Q_t^{-1} [z_t - h(x_t, m)]}_{\text{sensor measurement}} \quad (46.18)$$

This quadratic form yields a sparse system of equations and a number of efficient optimization techniques can be applied. Common choices include direct methods such as sparse Cholesky and QR decomposition, or iterative ones such as gradient descent, conjugate gradient, and others. Most SLAM implementations rely on iteratively linearizing the functions g and h , in which case the objective in (46.18) becomes quadratic in all of its variables.

Extensions to support an effective correction of large-scale graphs are hierarchical methods. One of the first is the ATLAS framework [46.25], which constructs a two-level hierarchy combining a Kalman filter that operates in the lower level and a global optimization at the higher level. Similar to that, Hierarchical SLAM [46.68] is a technique for using independent local maps, which are merged in case of re-visiting a place. A fully hierarchical approach has been presented in [46.65]. It builds a multilevel pose-graph and employs an incremental, lazy optimization scheme that

allows for optimizing large graphs and at the same time can be executed at each step during mapping. An alternative hierarchical approach is [46.69], which recursively partitions the graph into multiple-level submaps using the nested dissection algorithm.

When it comes to computing highly accurate environment reconstructions, approaches that do not only optimize the poses of the robot but also each individual measurement of a dense sensor often provide better results. In the spirit of bundle adjustment [46.4], approaches for laser scanners [46.70] and Kinect cameras [46.71] have been proposed.

The graphical paradigm can be extended to handle the data association problems as we can integrate additional knowledge on data association into (46.18). Suppose some oracle informed us that landmarks m_i and m_j in the map corresponded to one and the same physical landmark in the world. Then, we can either remove m_j from the graph and attach all adjacent edges to m_i , or we can add a soft correspondence constraint [46.72] of the form

$$(m_j - m_i)^T \Gamma (m_j - m_i) .$$

(46.19)

Here Γ is 2-by-2 diagonal matrix whose coefficients determine the penalty for *not* assigning identical locations to two landmarks (hence we want Γ to be large). Since graphical methods are usually used for the full SLAM problem, the optimization can be interleaved with the search for the optimal data association.

Data association errors typically have a strong impact in the resulting map estimate. Even a small number of wrong data associations is likely to result in inconsistent map estimates. Recently, novel approaches have been proposed that are robust under a certain number of false associations. For example, [46.73, 46.74] propose an iterative procedure that allows for disabling constraints, an action that is associated with a cost. A generalization of this method introduced in [46.75] formulates [46.74] as a robust cost function also reducing the computational requirements. Such approaches can deal with a significant number of false associations and still provide high-quality maps. Consistency checks for loop closure hypotheses can be found in other approaches as well, both in the front-end [46.76] and in the optimizer [46.77]. There has also been an extension that can deal with multimodal constraints [46.78], proposing a max-mixture representation for maintaining efficiency of the log likelihood optimizing in (46.16). As a result of that, the multimodal extension has only little impact on the runtime and can easily be incorporated in most optimizers. Also robust cost function are used for SLAM, for example pseudo Huber and several alternatives [46.75, 46.79, 46.80, 46.81].

Graphical SLAM methods have the advantage that they scale to much higher-dimensional maps than EKF SLAM, exploiting the sparsity of the graph. The key limiting factor in EKF SLAM is the covariance matrix, which takes space (and update time) quadratic in the size of the map. No such constraint exists in graphical methods. The update time of the graph is constant, and the amount of memory required is linear (under some mild assumptions). A further advantage of graph-based methods over the EKF is their ability to constantly re-linearize the error function which often leads to better results. Performing the optimization can be expensive, however. Technically, finding the optimal data association is suspected to be an NP-hard problem, although in practice the number of plausible assignments is usually small. The continuous optimization of the log likelihood function in (46.18) depends among other things on

the number and size of loops in the map. Also the initialization can have a strong impact on the result and a good initial guess can simplify the optimization substantially [46.8, 46.82, 46.83].

We note that the graph-based paradigm is very closely linked to information theory, in that the soft constraints constitute the information the robot has on the world (in an information-theoretic sense [46.92]). Most methods in the field are offline and they optimize for the entire robot path. If the robot path is long, the optimization may become cumbersome. Over the last five years, however, incremental optimization techniques have been proposed that aim at providing a sufficient but not necessarily perfect model of the environment at every point in time. This allows a robot to make decisions based on the current model, for example, to determine exploration goals. In this context, incremental variants [46.93, 46.94] of stochastic gradient descent techniques [46.8, 46.91] have been proposed that estimate which part of the graph requires re-optimization given new sensor data. Incremental methods [46.66, 46.79, 46.95] in the smoothing and mapping framework can be executed at each step of the mapping process and achieve the performance by variable ordering and selective relinearization. As also evaluated in [46.96] for the SLAM problem, variable ordering impacts the performance of the optimization. Others use hierarchical data structures [46.89] and pose-graphs [46.97] combined with a lazy optimization for on-the-fly mapping [46.65]. As an alternative to global methods, relative optimization approaches [46.98] aim at computing locally consistent geometric maps but only topological maps on the global scale. Hybrid approaches [46.87] seek to combine the best of both worlds.

There also exists a number of cross-overs that manipulate the graph online so as to factor out past robot location variables. The resulting algorithms are filters [46.100, 46.25, 46.33, 46.99], and they tend to be intimately related to information filter methods. Many of the original attempts to decompose EKF SLAM representations into smaller submaps to scale up are based on motivations that are not dissimilar to the graphical approach [46.101, 46.27, 46.28].

Recently, researchers addressed the problem of long-term operation and frequent revisits of already mapped terrain. To avoid densely connected pose-graphs that lead to slow convergence behavior, the robot can switch between SLAM and localization, can merge nodes to avoid a growth of the graph [46.102, 46.90], or can discard nodes or edges [46.103, 46.104, 46.105, 46.32].

Graphical and optimization-based SLAM algorithms are still subject of intense research and the paradigm scales to maps large numbers of nodes [46.106, 46.107, 46.25, 46.55, 46.57, 46.59, 46.63, 46.64, 46.65, 46.89, 46.90]. Arguably, the graph-based paradigm has generated some of the largest SLAM maps ever built. Furthermore, the SLAM community started to release flexible optimization frameworks and SLAM implementations under open source licenses to support further developments and to allow for efficient comparisons, (Table 46.1). Especially the optimization frameworks [46.66, 46.79, 46.80] are flexible and powerful state of the art tools for developing graph-based SLAM systems. They can be either used as a *black box* or can be easily extended through plugins.

Table 46.1 Recent open-source graph-based SLAM implementations

46.2.4 Relation of Paradigms

The three paradigms just discussed cover the vast majority of work in the field of SLAM. As discussed, EKF SLAM comes with a computational hurdle that poses serious scaling limitations and the linearization may lead to inconsistent maps. The most promising extensions of EKF SLAM are based on building local submaps; however, in many ways the resulting algorithms resemble the graph-based approach.

Particle filter methods sidestep some of the issues arising from the natural inter-feature correlations in the map – which hindered the EKF. By sampling from robot poses, the individual landmarks in the map become independent, and hence are decorrelated. As a result, FastSLAM can represent the posterior by a sampled robot pose, and many local, independent Gaussians for its landmarks. The particle representation offers advantages for SLAM as it allows for computationally efficient updates and for sampling over data associations. On the negative side, the number of necessary particles can grow very large, especially for robots seeking to map multiple nested loops.

Graph-based methods address the full SLAM problem, hence are in the standard formulation not online. They draw their intuition from the fact that SLAM can be modeled by a sparse graph of soft constraints, where each constraint either corresponds to a motion or a measurement event. Due to the availability of highly efficient optimization methods for sparse nonlinear optimization problems, graph-based SLAM has become the method of choice for building large-scale maps. Recent developments have brought up several graph-based methods for incremental map building that can be executed at every time step during navigation. Data association search can be incorporated into the basic mathematical framework and different approaches that are even robust under wrong data associations are available today.

46.3 Visual and RGB-D SLAM

A popular and important topic in recent years has been Visual SLAM – the challenge of building maps and tracking the robot pose in full 6-

DOF

using data from cameras [[46.108](#)] or RGB-D (Kinect) sensors [[46.109](#), [46.86](#)]. Visual sensors offer a wealth of information that enables the construction of rich 3-D models of the world. They also enable difficult issues such as loop-closing to be addressed in novel ways using appearance information [[46.110](#)]. Visual SLAM is anticipated to be a critical area for future research in perception for robotics, as we seek to develop low-cost systems that are capable of intelligent physical interaction with the world.

Attempting SLAM with monocular, stereo, omnidirectional, or RGB-D cameras raises the level-of-difficulty of many of the SLAM components, such as data association and computational efficiency, described above. A key challenge is robustness. Many visual SLAM applications of interest, such as augmented reality [[46.85](#)], entail handheld camera motions, which present greater difficulties for state estimation, in comparison to the motion of a wheeled robot across a flat floor.

Visual navigation and mapping was a key early goal in the mobile robotics community [[46.111](#), [46.112](#)], but early approaches were hampered by the lack of sufficient computational resources to handle massive video data streams. Early approaches were typically based on extended Kalman filters [[46.113](#), [46.114](#), [46.115](#), [46.116](#)], but did not compute

the full covariance for the feature poses and camera trajectory, resulting in a loss of consistency. Visual SLAM is closely related to the structure from motion (

SFM

) problem in computer vision [46.4, 46.5]. Historically, SFM was primarily concerned with off-line batch processing, whereas SLAM seeks to achieve a solution for online operation, suitable for closed-loop interaction of a robot or user with its environment. In comparison to laser scanners, cameras provide a fire hydrant of information, making online processing nearly impossible until recent increases in computation have become available.

Davison was an early pioneer in developing complete visual SLAM systems, initially using a real-time active stereo head [46.121] that tracked distinctive visual features with a full covariance EKF approach. Subsequent work developed the first real-time SLAM system that operated with a single freely moving camera as the only data source [46.122, 46.23]. This system could build sparse, room-size maps of indoor scenes at 30 Hz frame-rate in real-time, a notable historical achievement in visual SLAM research. A difficulty encountered with initial monocular SLAM [46.23] was coping with the initialization of points that were far away from the camera, due to nonGaussian distributions of such feature locations resulting from poor depth information. This limitation was overcome in [46.24], introducing an inverse depth parameterization for monocular SLAM, a key development for enabling a unified treatment of initialization and tracking of visual features in real-time.

A milestone in creating robust visual SLAM systems was the introduction of keyframes in parallel tracking and mapping (PTAM) [46.85], which separated the tasks of keyframe mapping and localization into parallel threads, improving robustness and performance for online processing. Keyframes are now a mainstream concept for complexity reduction in visual SLAM systems. Related approaches using keyframes include [46.123, 46.124, 46.125, 46.126, 46.87]. The work in [46.108] analyzes the tradeoffs between filtering and keyframe-based bundle adjustment in visual SLAM, and concluded that keyframe bundle adjustment outperforms filtering, as it provides the most accuracy per unit of computing time.

As pointed out by Davison and other researchers, an appealing aspect of visual SLAM is that camera measurements can provide odometry information, and indeed visual odometry is a key component of modern SLAM systems [46.127]. Here, [46.128] and [46.129] provide an extensive tutorial of techniques for visual odometry, including feature detection, feature matching, outlier rejection, and constraint estimation, and trajectory optimization. Finally, a publicly available visual odometry library [46.130] that is optimized for efficient operation on small unmanned aerial vehicles is available today.

Visual information offers a tremendous source of information for loop closing, not present in the canonical 2-D laser SLAM systems developed in the early 2000s. The work in [46.110] was one of the first to employ techniques for visual object recognition [46.131] to location recognition. More recently FAB-MAP [46.118, 46.132] has demonstrated appearance-only place recognition at large scale, mapping trajectories with a length of 1000 km. Combining a bag-of-features approach with a probabilistic place model and Chow-Liu tree inference leads to place recognition that is robust against perceptual aliasing while remaining computationally efficient. Other work on place recognition

includes [46.133], which combines bag-of-words loop closing with tests of geometrical consistency based on conditional random fields (Fig. 46.8).

Fig. 46.8

(a) A 2-km path and 50000 frames estimated for the New College Dataset (after [46.117]) using relative bundle adjustment (after [46.98]). (b) the relative bundle adjustment solution is easily improved by taking FAB-MAP (after [46.118]) loop-closures into account – this is achieved without global optimization (after [46.119]). Sibley et al. advocate that relative metric accuracy and topological consistency are the requirements for autonomous navigation, and these are better achieved using a relative manifold representation instead of using a conventional single Euclidean representation [46.120]

The techniques described above have formed the basis for a number of notable large-scale SLAM systems developed in recent years. A 2008 special issue of the IEEE Transactions on Robotics provides a good snapshot of recent state-of-the-art SLAM techniques [46.135]. Other notable recent examples include [46.119, 46.126, 46.136, 46.137, 46.138, 46.98]. The idea of employing relative bundle adjustment [46.98] to compute a full maximum likelihood solution in an online fashion, even for loop closures, by employing a manifold representation that does not attempt to enforce Euclidean constraints results in maps that can be computed at high frame rate (see also Fig. 46.8). Finally, view-based mapping systems [46.126, 46.136, 46.137] aim at large-scale and/or life-long visual mapping based on the pose graph optimization techniques described above in Section 46.2.3.

Several compelling 3-D mapping and localization have been created in recent years with RGB-D (Kinect) sensors. The combination of direct range measurements with dense visual imagery can enable dramatic improvements in mapping and navigation systems for indoor environments. State-of-the-art RGB-D SLAM systems include [46.109] and [46.86]. Figure 46.9 shows examples of the output of these systems.

Fig. 46.9

(a) 3-D Model and (b) close-up view of a corridor environment in the Paul G. Allen building at University of Washington built from Kinect data (after [46.109]); image courtesy of Peter Henry, University of Washington)

Other researchers aim at exploiting the surface properties of scanned environments to correct for sensor noise of range sensors such as the Kinect [46.71]. They jointly optimize the poses of the sensor and the positions of the surface points measured and iteratively refine the structure of the error function by recomputing the data associations after each optimization, resulting in accurate smooth models of the environment.

An emerging area for future research is the development of fully dense processing methods that exploit recent advances in commodity graphical processing unit (

GPU

) technology. Kinect-based dense tracking and mapping, a fully-dense method for small-scale visual tracking and reconstruction is described in [46.140]. Dense modeling and tracking are achieved via highly parallelized operations on commodity GPU hardware to yield a system that outperforms previous methods such as PTAM for challenging camera trajectories. Dense methods offer an interesting perspective from which to address long-standing problems, such as visual odometry, from a fresh perspective, without requiring explicit feature detection and matching [46.141]. KinectFusion [46.134, 46.84] is a dense modeling system that tracks the 3-D pose of a handheld Kinect while concurrently reconstructing high-quality scene 3-D models in real-time. See Fig. 46.10 for an example. KinectFusion has been applied to spatially extended environments in [46.139, 46.142]. An example is shown in Fig. 46.11.

Fig. 46.10a–c

Results obtained with KinectFusion (after [46.134]). (a) A local scene as a normal map and (b) as a Phong-shaded rendering. The (c) image depicts a larger scene (image courtesy of Richard Newcombe, Imperial College London)

Fig. 46.11

Spatially extended KinectFusion output produced in real-time with Kintinuous (after [46.139])

46.4 Conclusion and Future Challenges

This chapter has provided an introduction into SLAM, which is defined as the problem faced by a mobile platform roaming an unknown environment, and seeking to localize itself while concurrently building a map of the environment. The chapter discussed three main paradigms in SLAM, which are based on the extended Kalman filter, particle filters, and graph-based sparse optimization techniques, and then described recent progress in Visual/Kinect SLAM.

The following references provide an in-depth tutorial on SLAM and much greater depth of coverage on the details of popular SLAM algorithms. Furthermore, several implementations of popular SLAM systems, including most of the approaches listed in Table 46.1, can be found in online resources such as <http://www.openslam.org> or in the references [46.21, 46.6, 46.62, 46.9].

The considerable progress in SLAM in the past decade is beyond doubt. The core state estimation at the heart of SLAM is now quite well understood, and a number of impressive implementations have been developed, including several widely used open source software implementations and some commercial projects. None-the-less, a number of open research challenges remain for the general problem of robotic mapping in complex and dynamic environments over extended periods of time, including robots sharing, extending, and revising previously built models, efficient failure recovery, zero user intervention, and operation on resource-constrained systems. Another exciting area for the future is the further development of fully dense visual mapping systems exploiting the latest advances in GPU hardware development.

An ultimate goal is to realize the challenge of *persistent* navigation and mapping – the capability for a robot to perform SLAM robustly for days, weeks, or months at a time with minimal human supervision, in complex and dynamic environments. Taking the limit as $t \rightarrow \infty$ poses difficult challenges to most current algorithms; in fact, most robot mapping and navigation algorithms are doomed to fail with the passage of time, as errors inevitably accrue. Despite recent encouraging solutions [46.102, 46.105], more research is needed for techniques that can recover from mistakes and enable robots to deal with changes in the environment and enabling a long-term autonomous existence.

Abbreviations

2-D: two-dimensional

3-D: three-dimensional

DCS: dynamic covariance scaling

DOF: degree of freedom

EKF: extended Kalman filter

fastSLAM: fast simultaneous localization and mapping

GPS: global positioning system

GPU: graphics processing unit

ICP: iterative closest point

PCL: point cloud library

PTAM: parallel tracking and mapping

RGB-D: red green blue distance

SAM: smoothing and mapping

SFM: structure from motion

SGD: stochastic gradient descent

SLAM: simultaneous localization and mapping

SSA: sparse surface adjustment

References

4.6.1 C.F. Gauss: *Theoria Motus Corporum Coelestium (Theory of the Motion of the Heavenly Bodies Moving about the Sun in Conic Sections)* (Perthes and Bessen, Hamburg 1809), Republished in 1857 and by Dover in 1963

[Google Scholar](#)

4.6.2 D.C. Brown: The bundle adjustment – Progress and prospects, *Int. Arch. Photogramm.* 21(3), 3:3–3:35 (1976)

46.3 G. Konecny: *Geoinformation: Remote Sensing, Photogrammetry and Geographical Information Systems* (Taylor Francis, London 2002)

[Book](#) [Google Scholar](#)

46.4 B. Triggs, P. McLauchlan, R. Hartley, A. Fitzgibbon: Bundle adjustment – A modern synthesis, *Lect. Notes Comput. Sci.* **62**, 298–372 (2000)

[Article](#) [Google Scholar](#)

46.5 R. Hartley, A. Zisserman: *Multiple View Geometry in Computer Vision* (Cambridge Univ. Press, Cambridge 2003)

[MATH](#) [Google Scholar](#)

46.6 T. Bailey, H.F. Durrant-Whyte: Simultaneous localisation and mapping (SLAM): Part II, *Robotics Autom. Mag.* **13**(3), 108–117 (2006)

[Article](#) [Google Scholar](#)

46.7 H.F. Durrant-Whyte, T. Bailey: Simultaneous localisation and mapping (SLAM): Part I, *Robotics Autom. Mag.* **13**(2), 99–110 (2006)

[Article](#) [Google Scholar](#)

46.8 G. Grisetti, C. Stachniss, W. Burgard: Nonlinear constraint network optimization for efficient map learning, *IEEE Trans. Intell. Transp. Syst.* **10**(3), 428–439 (2009)

[Article](#) [Google Scholar](#)

46.9 S. Thrun, W. Burgard, D. Fox: *Probabilistic Robotics* (MIT Press, Cambridge, 2005)

[MATH](#) [Google Scholar](#)

46.10 R. Smith, M. Self, P. Cheeseman: A stochastic map for uncertain spatial relationships, *Proc. Int. Symp. Robotics Res. (ISRR)* (MIT Press, Cambridge 1988) pp. 467–474

[Google Scholar](#)

46.11 R. Smith, M. Self, P. Cheeseman: Estimating uncertain spatial relationships in robotics. In: *Autonomous Robot Vehicles*, ed. by I.J. Cox, G.T. Wilfong (Springer Verlag, Berlin, Heidelberg 1990) pp. 167–193

46.12 P. Moutarlier, R. Chatila: Stochastic multisensory data fusion for mobile robot location and environment modeling, 5th Int. Symp. Robotics Res. (ISRR) (1989) pp. 207–216

[Google Scholar](#)

46.13 P. Moutarlier, R. Chatila: An experimental system for incremental environment modeling by an autonomous mobile robot, 1st Int. Sym. Exp. Robotics (ISER) (1990)

[Google Scholar](#)

46.14 R. Kalman: A new approach to linear filtering and prediction problems, J. Fluids **82**, 35–45 (1960)

[MathSciNet](#) [Google Scholar](#)

46.15 A.M. Jazwinsky: *Stochastic Processes and Filtering Theory* (Academic, New York 1970)

[Google Scholar](#)

46.16 M.G. Dissanayake, P.M. Newman, S. Clark, H.F. Durrant-Whyte, M. Csorba: A solution to the simultaneous localization and map building (SLAM) Problem, IEEE Trans. Robotics Autom. **17**(3), 229–241 (2001)

[Article](#) [Google Scholar](#)

46.17 J. Neira, J. Tardos, J. Castellanos: Linear time vehicle relocation in SLAM, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2003) pp. 427–433

[Google Scholar](#)

46.18 J. Neira, J.D. Tardos: Data association in stochastic mapping using the joint compatibility test, IEEE Trans. Robotics Autom. **17**(6), 890–897 (2001)

[Article](#) [Google Scholar](#)

46.19 G.D. Tipaldi, M. Braun, K.O. Arras: Flirt: interest regions for 2D range data with applications to robot navigation, Proc. Int. Symp. Exp. Robotics (ISER) (2010)

[Google Scholar](#)

46.20 G.D. Tipaldi, L. Spinello, W. Burgard: Geometrical flirt phrases for large scale place recognition in 2D range data, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2013)

46.21 T. Bailey: Mobile Robot Localisation and Mapping in Extensive Outdoor Environments, Ph.D. Thesis (Univ. of Sydney, Sydney 2002)

[Google Scholar](#)

46.22 J.J. Leonard, R.R. Rikoski, P.M. Newman, M. Bosse: Mapping partially observable features from multiple uncertain vantage points, *Int. J. Robotics Res.* **21**(10), 943–975 (2002)

[Article](#) [Google Scholar](#)

46.23 A.J. Davison: Real-time simultaneous localisation and mapping with a single camera, *Proc. IEEE 9th Int. Conf. Comput. Vis.* (2003) pp. 1403–1410

[Google Scholar](#)

46.24 J.M.M. Montiel, J. Civera, A.J. Davison: Unified inverse depth parametrization for monocular SLAM, *Robotics Sci. Syst.*, Vol. 1 (2006)

[Google Scholar](#)

46.25 M. Bosse, P.M. Newman, J. Leonard, S. Teller: Simultaneous localization and map building in large-scale cyclic environments using the Atlas Framework, *Int. J. Robotics Res.* **23**(12), 1113–1139 (2004)

[Article](#) [Google Scholar](#)

46.26 J. Nieto, T. Bailey, E. Nebot: Scan-SLAM: Combining EKF-SLAM and scan correlation, *Proc. IEEE Int. Conf. Robotics Autom. (ICRA)* (2005)

[Google Scholar](#)

46.27 J. Guivant, E. Nebot: Optimization of the simultaneous localization and map building algorithm for real time implementation, *IEEE Trans. Robotics. Autom.* **17**(3), 242–257 (2001)

[Article](#) [Google Scholar](#)

46.28 J.J. Leonard, H. Feder: A computationally efficient method for large-scale concurrent mapping and localization, *Proc. 9th Int. Symp. Robotics Res. (ISRR)*, ed. by J. Hollerbach, D. Koditschek (1999) pp. 169–176

[Google Scholar](#)

46.29 J.D. Tardós, J. Neira, P.M. Newman, J.J. Leonard: Robust mapping and localization in indoor environments using sonar data, *Int. J. Robotics Res.* **21**(4), 311–330 (2002)

[Article](#) [Google Scholar](#)

46.30 S.B. Williams, G. Dissanayake, H.F. Durrant-Whyte: Towards multi-vehicle simultaneous localisation and mapping, *Proc. IEEE Int. Conf. Robotics Autom. (ICRA)* (2002) pp. 2743–2748

[Google Scholar](#)

46.31 R.M. Eustice, H. Singh, J.J. Leonard: Exactly sparse delayed-state filters for view-based SLAM, *IEEE Trans. Robotics* **22**(6), 1100–1114 (2006)

[Article](#) [Google Scholar](#)

46.32 V. Ila, J.M. Porta, J. Andrade-Cetto: Information-based compact pose SLAM, *IEEE Trans. Robotics* **26**(1), 78–93 (2010)

[Article](#) [Google Scholar](#)

46.33 S. Thrun, D. Koller, Z. Ghahramani, H.F. Durrant-Whyte, A.Y. Ng: Simultaneous mapping and localization with sparse extended information filters, *Proc. 5th Int. Workshop Algorithmic Found. Robotics*, ed. by J.-D. Boissonnat, J. Burdick, K. Goldberg, S. Hutchinson (2002)

[Google Scholar](#)

46.34 M.R. Walter, R.M. Eustice, J.J. Leonard: Exactly sparse extended information filters for feature-based SLAM, *Int. J. Robotics Res.* **26**(4), 335–359 (2007)

[Article](#) [Google Scholar](#)

46.35 T. Bailey, J. Nieto, J. Guivant, M. Stevens, E. Nebot: Consistency of the EKF-SLAM algorithm, *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)* (2006) pp. 3562–3568

[Google Scholar](#)

46.36 S. Huang, G. Dissanayake: Convergence and consistency analysis for extended Kalman filter based SLAM, *IEEE Trans. Robotics* **23**(5), 1036–1049 (2007)

[Article](#) [Google Scholar](#)

46.37 G.P. Huang, A.I. Mourikis, S.I. Roumeliotis: Observability-based Rules for Designing Consistent EKF SLAM Estimators, *Int. J. Robotics Res.* **29**, 502–528 (2010)

[Article](#) [Google Scholar](#)

46.38 N. Metropolis, S. Ulam: The Monte Carlo method, J. Am. Stat. Assoc. **44**(247), 335–341 (1949)

[Article](#) [MATH](#) [Google Scholar](#)

46.39 D. Blackwell: Conditional expectation and unbiased sequential estimation, Ann. Math. Stat. **18**, 105–110 (1947)

[Article](#) [MathSciNet](#) [MATH](#) [Google Scholar](#)

46.40 C.R. Rao: Information and accuracy obtainable in estimation of statistical parameters, Bull. Calcutta Math. Soc. **37**(3), 81–91 (1945)

[MathSciNet](#) [Google Scholar](#)

46.41 K. Murphy, S. Russel: Rao--Blackwellized particle filtering for dynamic Bayesian networks. In: *Sequential Monte Carlo Methods in Practice*, ed. by A. Doucet, N. de Freitas, N. Gordon (Springer, Berlin 2001) pp. 499–516

[Chapter](#) [Google Scholar](#)

46.42 M. Montemerlo, S. Thrun, D. Koller, B. Wegbreit: FastSLAM: A factored solution to the simultaneous localization and mapping problem, Proc. AAAI Natl. Conf. Artif. Intell. (2002)

[Google Scholar](#)

46.43 J. Pearl: *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference* (Morgan Kaufmann, San Mateo 1988)

[MATH](#) [Google Scholar](#)

46.44 J. Guivant, E. Nebot, S. Baiker: Autonomous navigation and map building using laser range sensors in outdoor applications, J. Robotics Syst. **17**(10), 565–583 (2000)

[Article](#) [MATH](#) [Google Scholar](#)

46.45 G. Grisetti, C. Stachniss, W. Burgard: Improved techniques for grid mapping with Rao-Blackwellized particle filters, IEEE Trans. Robotics **23**, 34–46 (2007)

[Article](#) [Google Scholar](#)

46.46 D. Hähnel, D. Fox, W. Burgard, S. Thrun: A highly efficient FastSLAM algorithm for generating cyclic maps of large-scale environments from raw laser range measurements, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.

(IROS) (2003)

[Google Scholar](#)

46.47 C. Stachniss, G. Grisetti, W. Burgard, N. Roy: Evaluation of gaussian proposal distributions for mapping with rao-blackwellized particle filters, Proc. IEEE/RSJ Int. Conf. Intell. RobotsSyst. (IROS) (2007)

[Google Scholar](#)

46.48 A. Eliazar, R. Parr: DP-SLAM: Fast, robust simultaneous localization and mapping without predetermined landmarks, Proc. 16th Int. Jt. Conf. Artif. Intell. (IJCAI) (2003) pp. 1135–1142

[Google Scholar](#)

46.49 A. Eliazar, R. Parr: DP-SLAM 2.0, Proc. IEEE Int. Conf. Robotics Autom. (ICRA), Vol. 2 (2004) pp. 1314–1320

[Google Scholar](#)

46.50 G. Grisetti, G.D. Tipaldi, C. Stachniss, W. Burgard, D. Nardi: Fast and accurate SLAM with Rao-Blackwellized particle filters, J. Robotics Auton. Syst. **55**(1), 30–38 (2007)

[Article](#) [Google Scholar](#)

46.51 D. Roller, M. Montemerlo, S. Thrun, B. Wegbreit: FastSLAM 2.0: An improved particle filtering algorithm for simultaneous localization and mapping that provably converges, Int. Jt. Conf. Artif. Intell. (IJCAI) (Morgan Kaufmann, San Francisco 2003) pp. 1151–1156

[Google Scholar](#)

46.52 R. van der Merwe, N. de Freitas, A. Doucet, E. Wan: The unscented particle filter, Proc. Adv. Neural Inform. Process. Syst. Conf. (2000) pp. 584–590

[Google Scholar](#)

46.53 C. Stachniss, G. Grisetti, W. Burgard: Recovering particle diversity in a Rao-Blackwellized particle filter for SLAM after actively closing loops, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2005) pp. 655–660

[Google Scholar](#)

46.54 F. Lu, E. Milios: Globally consistent range scan alignment for environmental mapping, Auton. Robots **4**, 333–349 (1997)

[Article](#) [Google Scholar](#)

46.55 F. Dellaert: Square root SAM, Robotics Sci. Syst., ed. by S. Thrun, G. Sukhatme, S. Schaal, O. Brock (MIT Press, Cambridge 2005)

[Google Scholar](#)

46.56 T. Duckett, S. Marsland, J. Shapiro: Learning globally consistent maps by relaxation, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2000) pp. 3841–3846

[Google Scholar](#)

46.57 T. Duckett, S. Marsland, J. Shapiro: Fast, on-line learning of globally consistent maps, Auton. Robots **12**(3), 287–300 (2002)

[Article](#) [MATH](#) [Google Scholar](#)

46.58 J. Folkesson, H.I. Christensen: Graphical SLAM: A self-correcting map, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2004) pp. 383–390

[Google Scholar](#)

46.59 U. Frese, G. Hirzinger: Simultaneous localization and mapping – A discussion, Proc. IJCAI Workshop Reason. Uncertain. Robotics (2001) pp. 17–26

[Google Scholar](#)

46.60 M. Golfarelli, D. Maio, S. Rizzi: Elastic correction of dead-reckoning errors in map building, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (1998) pp. 905–911

[Google Scholar](#)

46.61 J. Gutmann, K. Konolige: Incremental mapping of large cyclic environments, Proc. IEEE Int. Symp. Comput. Intell. Robotics Autom. (CIRA) (2000) pp. 318–325

[Google Scholar](#)

46.62 G. Grisetti, R. Kümmerle, C. Stachniss, W. Burgard: A Tutorial on Graph-based SLAM, IEEE Trans. Intell. Transp. Syst. Mag. **2**, 31–43 (2010)

[Article](#) [Google Scholar](#)

46.63 K. Konolige: Large-scale map-making, Proc. AAAI Natl. Conf. Artif. Intell. (MIT Press, Cambridge 2004) pp. 457–463

46.64 M. Montemerlo, S. Thrun: Large-scale robotic 3-D mapping of urban structures, Springer Tract. Adv. Robotics **21**, 141–150 (2005)

[Article](#) [Google Scholar](#)

46.65 G. Grisetti, R. Kümmerle, C. Stachniss, U. Frese, C. Hertzberg: Hierarchical optimization on manifolds for online 2D and 3D mapping, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2010)

[Google Scholar](#)

46.66 M. Kaess, H. Johannsson, R. Roberts, V. Ila, J.J. Leonard, F. Dellaert: iSAM2: Incremental smoothing and mapping using the Bayes tree, Int. J. Robotics Res. **31**, 217–236 (2012)

[Article](#) [Google Scholar](#)

46.67 M. Kaess, A. Ranganathan, F. Dellaert: iSAM: Incremental Smoothing and Mapping, IEEE Trans. Robotics **24**(6), 1365–1378 (2008)

[Article](#) [Google Scholar](#)

46.68 C. Estrada, J. Neira, J.D. Tardós: Hierarchical SLAM: Real-time accurate mapping of large environments, IEEE Trans. Robotics **21**(4), 588–596 (2005)

[Article](#) [Google Scholar](#)

46.69 K. Ni, F. Dellaert: Multi-level submap based SLAM using nested dissection, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2010)

[Google Scholar](#)

46.70 M. Ruhnke, R. Kümmerle, G. Grisetti, W. Burgard: Highly accurate maximum likelihood laser mapping by jointly optimizing laser points and robot poses, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2011)

[Google Scholar](#)

46.71 M. Ruhnke, R. Kümmerle, G. Grisetti, W. Burgard: Highly accurate 3D surface models by sparse surface adjustment, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2012)

[Google Scholar](#)

46.72 Y. Liu, S. Thrun: Results for outdoor-SLAM using sparse extended information filters, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2003)

[Google Scholar](#)

46.73 N. Sünderhauf, P. Protzel: BRIEF-Gist – Closing the loop by simple means, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2011) pp. 1234–1241

[Google Scholar](#)

46.74 N. Sünderhauf, P. Protzel: Switchable constraints for robust pose graph SLAM, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2012)

[Google Scholar](#)

46.75 P. Agarwal, G.D. Tipaldi, L. Spinello, C. Stachniss, W. Burgard: Robust map optimization using dynamic covariance scaling, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2013)

[Google Scholar](#)

46.76 E. Olson: Recognizing places using spectrally clustered local matches, J. Robotics Auton. Syst. 57(12), 1157–1172 (2009)

[Article](#) [Google Scholar](#)

46.77 Y. Latif, C. Cadena Lerma, J. Neira: Robust loop closing over time, Robotics Sci. Syst. (2012)

[Google Scholar](#)

46.78 E. Olson, P. Agarwal: Inference on networks of mixtures for robust robot mapping, Robotics Sci. Syst. (2012)

[Google Scholar](#)

46.79 F. Dellaert: *Factor graphs and GTSAM: A hands-on introduction*, Tech. Rep. GT-RIM-CP & R-2012-002 (Georgia Tech, Atlanta 2012)

[Google Scholar](#)

46.80 R. Kümmerle, G. Grisetti, H. Strasdat, K. Konolige, W. Burgard: G²o: A general framework for graph optimization, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2011)

[Google Scholar](#)

46.81 D.M. Rosen, M. Kaess, J.J. Leonard: An incremental trust-region method for robust online sparse least-squares estimation, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2012) pp. 1262–1269

[Google Scholar](#)

46.82 L. Carlone, R. Aragues, J. Castellanos, B. Bona: A linear approximation for graph-based simultaneous localization and mapping, Robotics Sci. Syst. (2011)

[Google Scholar](#)

46.83 G. Grisetti, R. Kümmerle, K. Ni: Robust optimization of factor graphs by using condensed measurements, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2012)

[Google Scholar](#)

46.84 S. Izadi, R.A. Newcombe, D. Kim, O. Hilliges, D. Molyneaux, S. Hodges, P. Kohli, J. Shotton, A.J. Davison, A. Fitzgibbon: Kinectfusion: Real-time dynamic 3D surface reconstruction and interaction, ACM SIGGRAPH Talks (2011) p. 23

[Google Scholar](#)

46.85 G. Klein, D. Murray: Parallel tracking and mapping for small AR workspaces, IEEE/ACM Int. Symp. Mixed Augment. Real. (ISMAR) (2007) pp. 225–234

[Google Scholar](#)

46.86 F. Endres, J. Hess, N. Engelhard, J. Sturm, D. Cremers, W. Burgard: An evaluation of the RGB-D SLAM system, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2012)

[Google Scholar](#)

46.87 H. Strasdat, A.J. Davison, J.M.M. Montiel, K. Konolige: Double window optimisation for constant time visual SLAM, Int. Conf. Computer Vis. (ICCV) (2011)

[Google Scholar](#)

46.88 A. Nüchter: 3D robot mapping, Springer Tract. Adv. Robotics 52 (2009)

[Google Scholar](#)

46.89 U. Frese: Treemap: An $O(\log n)$ algorithm for indoor simultaneous localization and mapping, Auton. Robots 21(2), 103–122 (2006)

46.90 G. Grisetti, C. Stachniss, S. Grzonka, W. Burgard: A tree parameterization for efficiently computing maximum likelihood maps using gradient descent, *Robotics Sci. Syst.* (2007)

[Google Scholar](#)

46.91 E. Olson, J.J. Leonard, S. Teller: Fast iterative alignment of pose graphs with poor initial estimates, *Proc. IEEE Int. Conf. Robotics Autom. (ICRA)* (2006) pp. 2262–2269

[Google Scholar](#)

46.92 T.M. Cover, J.A. Thomas: *Elements of Information Theory* (Wiley, New York 1991)

[Book](#) [MATH](#) [Google Scholar](#)

46.93 E. Olson, J.J. Leonard, S. Teller: Spatially-adaptive learning rates for online incremental SLAM, *Robotics Sci. Syst.* (2007)

[Google Scholar](#)

46.94 G. Grisetti, D. Lordi Rizzini, C. Stachniss, E. Olson, W. Burgard: Online constraint network optimization for efficient maximum likelihood map learning, *Proc. IEEE Int. Conf. Robotics Autom. (ICRA)* (2008)

[Google Scholar](#)

46.95 M. Kaess, A. Ranganathan, F. Dellaert: Fast incremental square root information smoothing, *Int. Jt. Conf. Artif. Intell. (ISCAI)* (2007)

[Google Scholar](#)

46.96 P. Agarwal, E. Olson: Evaluating variable reordering strategies for SLAM, *Proc. IEEE/RSJ Int. Conf. Intel Robots Syst. (IROS)* (2012)

[Google Scholar](#)

46.97 K. Ni, D. Steedly, F. Dellaert: Tectonic SAM: exact, out-of-core, submap-based SLAM, *Proc. IEEE Int. Conf. Robotics Autom. (ICRA)* (2007) pp. 1678–1685

[Google Scholar](#)

46.98 G. Sibley, C. Mei, I. Reid, P. Newman: Adaptive relative bundle adjustment, *Robotics Sci. Syst.* (2009)

46.99 P.M. Newman, J.J. Leonard, R. Rikoski: Towards constant-time SLAM on an autonomous underwater vehicle using synthetic aperture sonar, 11th Int. Symp. Robotics Res. (2003)

[Google Scholar](#)

46.100 M.A. Paskin: Thin junction tree filters for simultaneous localization and mapping, Int. Jt. Conf. Artif. Intell. (IJCAI) (Morgan Kaufmann, New York 2003) pp. 1157–1164

[Google Scholar](#)

46.101 S.B. Williams: Efficient Solutions to Autonomous Mapping and Navigation Problems, Ph.D. Thesis (ACFR Univ. Sydney, Sydney 2001)

[Google Scholar](#)

46.102 H. Johannsson, M. Kaess, M.F. Fallon, J.J. Leonard: Temporally scalable visual SLAM using a reduced pose graph, RSS Workshop Long-term Oper. Auton. Robotic Syst. Chang. Environ. (2012)

[Google Scholar](#)

46.103 N. Carlevaris-Bianco, R.M. Eustice: Generic factor-based node marginalization and edge sparsification for pose-graph SLAM, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2013)

[Google Scholar](#)

46.104 M. Kaess, F. Dellaert: Covariance recovery from a square root information matrix for data association, J. Robotics Auton. Syst. **57**(12), 1198–1210 (2009)

[Article](#) [Google Scholar](#)

46.105 H. Kretzschmar, C. Stachniss: Information-theoretic compression of pose graphs for laser-based SLAM, Int. J. Robotics Res. **31**(11), 1219–1230 (2012)

[Article](#) [Google Scholar](#)

46.106 U. Frese, L. Schröder: Closing a million-landmarks loop, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2006)

[Google Scholar](#)

46.107 J. McDonald, M. Kaess, C. Cadena, J. Neira, J.J. Leonard: Real-time 6-DOF multi-session visual SLAM over large scale environments, *J. Robotics Auton. Syst.* **61**(10), 1144–1158 (2012)

[Article](#) [Google Scholar](#)

46.108 H. Strasdat, J.M.M. Montiel, A.J. Davison: Real-time monocular SLAM: Why filter?, *Proc. IEEE Int. Conf. Robotics Autom. (ICRA)* (2010)

[Google Scholar](#)

46.109 P. Henry, M. Krainin, E. Herbst, X. Ren, D. Fox: RGB-D mapping: Using depth cameras for dense 3D modeling of indoor environments, *Int. J. Robotics Res.* **31**(5), 647–663 (2012)

[Article](#) [Google Scholar](#)

46.110 D. Nister, H. Stewenius: Scalable recognition with a vocabulary tree, *Proc. IEEE Int. Conf. Comput. Vis. Pattern Recognit. (ICCVPR)* (2006) pp. 2161–2168

[Google Scholar](#)

46.111 R.A. Brooks: Aspects of mobile robot visual map making, *Proc. Int. Symp. Robotics Res. (ISRR)* (MIT Press, Cambridge 1984) pp. 287–293

[Google Scholar](#)

46.112 H. Moravec: Obstacle Avoidance and Navigation in the Real World by a Seeing Robot Rover, Ph.D. Thesis (Stanford Univ., Stanford 1980)

[Google Scholar](#)

46.113 N. Ayache, O. Faugeras: Building, registering, and fusing noisy visual maps, *Int. J. Robotics Res.* **7**(6), 45–65 (1988)

[Article](#) [Google Scholar](#)

46.114 D. Kriegman, E. Triendl, T. Binford: Stereo vision and navigation in buildings for mobile robots, *IEEE Trans. Robotics Autom.* **5**(6), 792–803 (1989)

[Article](#) [Google Scholar](#)

46.115 L. Matthies, S. Shafer: Error modeling in stereo navigation, *IEEE J. Robotics Autom.* **3**(3), 239–248 (1987)

[Article](#) [Google Scholar](#)

46.116 S. Pollard, J. Porrill, J. Mayhew: Predictive feed-forward stereo processing, *Alvey Vis. Conf.* (1989) pp. 97–102

[Google Scholar](#)

46.117 M. Smith, I. Baldwin, W. Churchill, R. Paul, P. Newman: The new college vision and laser data set, *Int. J. Robotics Res.* **28**(5), 595–599 (2009)

[Article](#) [Google Scholar](#)

46.118 M. Cummins, P.M. Newman: Appearance-only SLAM at large scale with FAB-MAP 2.0, *Int. J. Robotics Res.* **30**(9), 1100–1123 (2010)

[Article](#) [Google Scholar](#)

46.119 P.M. Newman, G. Sibley, M. Smith, M. Cummins, A. Harrison, C. Mei, I. Posner, R. Shade, D. Schroter, L. Murphy, W. Churchill, D. Cole, I. Reid: Navigating, recognising and describing urban spaces with vision and laser, *Int. J. Robotics Res.* **28**, 11–12 (2009)

[Google Scholar](#)

46.120 G. Sibley, C. Mei, I. Reid, P. Newman: Vast-scale outdoor navigation using adaptive relative bundle adjustment, *Int. J. Robotics Res.* **29**(8), 958–980 (2010)

[Article](#) [Google Scholar](#)

46.121 A. Davison, D. Murray: Mobile robot localisation using active vision, *Eur. Conf. Comput. Vis. (ECCV)* (1998) pp. 809–825

[Google Scholar](#)

46.122 A.J. Davison, I. Reid, N. Molton, O. Stasse: MonoSLAM: Real-time single camera SLAM, *IEEE Trans., Pattern Anal. Mach. Intell.* **29**(6), 1052–1067 (2007)

[Article](#) [Google Scholar](#)

46.123 R.O. Castle, G. Klein, D.W. Murray: Wide-area augmented reality using camera tracking and mapping in multiple regions, *Comput. Vis. Image Underst.* **115**(6), 854–867 (2011)

[Article](#) [Google Scholar](#)

46.124 E. Eade, T. Drummond: Unified loop closing and recovery for real time monocular SLAM, *Br. Mach. Vis. Conf.* (2008)

46.125 E. Eade, P. Fong, M.E. Munich: Monocular graph SLAM with complexity reduction, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2010) pp. 3017–3024

[Google Scholar](#)

46.126 K. Konolige, M. Agrawal: FrameSLAM: From bundle adjustment to real-time visual mapping, IEEE Trans. Robotics **24**(5), 1066–1077 (2008)

[Article](#) [Google Scholar](#)

46.127 D. Nister, O. Naroditsky, J. Bergen: Visual odometry for ground vehicle applications, J. Field Robotics **23**(1), 3–20 (2006)

[Article](#) [MATH](#) [Google Scholar](#)

46.128 D. Scaramuzza, F. Fraundorfer: Visual odometry. Part I: The first 30 years and fundamentals, IEEE Robotics Autom. Mag. **18**(4), 80–92 (2011)

[Article](#) [Google Scholar](#)

46.129 F. Fraundorfer, D. Scaramuzza: Visual odometry. Part II: Matching, robustness, optimization, and applications, IEEE Robotics Autom. Mag. **19**(2), 78–90 (2012)

[Article](#) [Google Scholar](#)

46.130 A.S. Huang, A. Bachrach, P. Henry, M. Krainin, D. Maturana, D. Fox, N. Roy: Visual odometry and mapping for autonomous flight using an RGB-D camera, Proc. Int. Symp. Robotics Res. (ISRR) (2011)

[Google Scholar](#)

46.131 J. Sivic, A. Zisserman: Video Google: A text retrieval approach to object matching in videos, Int. Conf. Computer Vis. (ICCV) (2003) p. 1470

[Google Scholar](#)

46.132 M. Cummins, P.M. Newman: Probabilistic appearance based navigation and loop closing, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2007) pp. 2042–2048

[Google Scholar](#)

46.133 C. Cadena, D. Gálvez, F. Ramos, J.D. Tardós, J. Neira: Robust place recognition with stereo cameras, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2010)

[Google Scholar](#)

46.134 R.A. Newcombe, A.J. Davison, S. Izadi, P. Kohli, O. Hilliges, J. Shotton, D. Molyneaux, S. Hodges, D. Kim, A. Fitzgibbon: Kinectfusion: Real-time dense surface mapping and tracking, IEEE/ACM Int. Sym. Mixed Augment. Real. (ISMAR) (2011) pp. 127–136

[Google Scholar](#)

46.135 J. Neira, A.J. Davison, J.J. Leonard: Guest editorial special issue on visual SLAM, IEEE Trans. Robotics 24(5), 929–931 (2008)

[Article](#) [Google Scholar](#)

46.136 K. Konolige, J. Bowman: Towards lifelong visual maps, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2009) pp. 1156–1163

[Google Scholar](#)

46.137 K. Konolige, J. Bowman, J.D. Chen, P. Mihelich, M. Calonder, V. Lepetit, P. Fua: View-based maps, Int. J. Robotics Res. 29(8), 941–957 (2010)

[Article](#) [Google Scholar](#)

46.138 G. Sibley, C. Mei, I. Reid, P. Newman: Planes, trains and automobiles – Autonomy for the modern robot, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2010) pp. 285–292

[Google Scholar](#)

46.139 T. Whelan, H. Johannsson, M. Kaess, J.J. Leonard, J.B. McDonald: Robust real-time visual odometry for dense RGB-D mapping, IEEE Int. Conf. Robotics Autom. (ICRA) (2013)

[Google Scholar](#)

46.140 R.A. Newcombe, S.J. Lovegrove, A.J. Davison: DTAM: Dense tracking and mapping in real-time, Int. Conf. Computer Vis. (ICCV) (2011) pp. 2320–2327

[Google Scholar](#)

46.141 F. Steinbruecker, J. Sturm, D. Cremers: Real-time visual odometry from dense RGB-D images, Workshop Live Dense Reconstr. Mov. Cameras Int. Conf. Comput. Vis. (ICCV) (2011)

46.142 H. Roth, M. Vona: Moving volume kinectfusion, Br. Mach. Vis. Conf. (2012)

[Google Scholar](#)

Author information

Authors and Affiliations

Institute for Geodesy and Geoinformation, University of Bonn, Nussallee 15, 53115, Bonn, Germany

Cyrill Stachniss

Department of Mechanical Engineering, Massachusetts Institute of Technology, 5-214 77 Massachusetts Avenue, MA 02139, Cambridge, USA

John J. Leonard

Udacity Inc., 2465 Latham Street, 3rd Floor, CA 94040, Mountain View, USA

Sebastian Thrun

Corresponding author

Correspondence to [Cyrill Stachniss](#).

Editor information

Editors and Affiliations

Department of Electrical Engineering, University of Naples Federico II, Via Claudio 21, 80125, Naples, Italy

Bruno Siciliano

Department of Computer Sciences, Artificial Intelligence Laboratory, Stanford University, 450 Serra Mall, CA 94305, Stanford, USA

Oussama Khatib

Video-References

 : Deformation-based loop closure for Dense RGB-D SLAM available from

<http://handbookofrobotics.org/view-chapter/46/videodetails/439>

 : Large-scale SLAM using the Atlas framework available from <http://handbookofrobotics.org/view-chapter/46/videodetails/440>

<http://handbookofrobotics.org/view-chapter/46/videodetails/440>

 : Graph-based SLAM available from <http://handbookofrobotics.org/view-chapter/46/videodetails/441>

 : Graph-based SLAM available from <http://handbookofrobotics.org/view-chapter/46/videodetails/442>

 **VIDEO 43** : Graph-based SLAM available from <http://handbookofrobotics.org/view-chapter/46/videodetails/443>

 **VIDEO 44** : Graph-based SLAM available from <http://handbookofrobotics.org/view-chapter/46/videodetails/444>

 **VIDEO 45** : Graph-based SLAM available from <http://handbookofrobotics.org/view-chapter/46/videodetails/445>

 **VIDEO 46** : Graph-based SLAM using TORO available from <http://handbookofrobotics.org/view-chapter/46/videodetails/446>

 **VIDEO 47** : Sparse pose adjustment available from <http://handbookofrobotics.org/view-chapter/46/videodetails/447>

 **VIDEO 48** : Pose graph compression for laser-based SLAM available from <http://handbookofrobotics.org/view-chapter/46/videodetails/449>

 **VIDEO 49** : Pose graph compression for laser-based SLAM available from <http://handbookofrobotics.org/view-chapter/46/videodetails/450>

 **VIDEO 50** : Pose graph compression for laser-based SLAM available from <http://handbookofrobotics.org/view-chapter/46/videodetails/451>

 **VIDEO 51** : DTAM: Dense tracking and mapping in real-time available from <http://handbookofrobotics.org/view-chapter/46/videodetails/452>

 **VIDEO 52** : MonoSLAM: Real-time single camera SLAM available from <http://handbookofrobotics.org/view-chapter/46/videodetails/453>

 **VIDEO 53** : SLAM++: Simultaneous localisation and mapping at the level of objects available from <http://handbookofrobotics.org/view-chapter/46/videodetails/454>

 **VIDEO 54** : Extended Kalman filter SLAM available from <http://handbookofrobotics.org/view-chapter/46/videodetails/455>

Rights and permissions

[Reprints and permissions](#)

Copyright information

© 2016 Springer-Verlag Berlin Heidelberg

About this chapter

Cite this chapter

[.RIS](#)  [.ENW](#)  [.BIB](#) 

DOI	Published	Publisher Name
https://doi.org/10.1007/978-3-319-32552-1_46	27 July 2016	Springer, Cham
Print ISBN	Online ISBN	eBook Packages
978-3-319-32550-7	978-3-319-32552-1	Engineering
		Engineering (RO)

Share this chapter

Anyone you share the following link with will be able to read this content:

[Get shareable link](#)

Provided by the Springer Nature SharedIt content-sharing initiative

Publish with us

[Policies and ethics](#) 

Attachment K

978-3-319-32552-1_46.pdf

1 / 23100%+

1

2

3

4

5

Multimedia Contents

46. Simultaneous Localization and Mapping

1153

Part E | 46

Document properties

File name:978-3-319-32552-1_46.pdf

File size:2.4 MB

Title:-

Author:-

Subject:-

Keywords:-

Created:7/21/16, 7:38:57 AM

Modified:7/25/16, 6:11:32 AM

Application:pdfsam-console (Ver. 2.4.1e)

PDF producer:Acrobat Distiller 11.0 (Windows)

PDF version:1.4

Page count:23

Page size:7.60 x 10.24 in (portrait)

Fast web view:Yes

Close

SLAM: Problem Definition..... 1154

6.1.1 Mathematical Basis 1154

6.1.2 Example: SLAM in Landmark Worlds 1155

6.1.3 Taxonomy of the SLAM Problem .. 1156

The Three Main SLAM Paradigms 1157

6.2.1 Extended Kalman Filters 1157

6.2.2 Particle Methods 1159

6.2.3 Graph-Based Optimization Techniques..... 1162

6.2.4 Relation of Paradigms..... 1166

Visual and RGB-D SLAM 1166

Conclusion and Future Challenges 1169

References 1170

nces 1171

ensors (RGB-D), and close with a discussion

en research problems in robotic mapping.

and of limited size. Robustly mapping unstruc-

dynamic, and large-scale environments in an on-

tion remains largely an open research problem.

the historical roots of methods that can be applied

to address the SLAM problem can be traced back to

Gauss [46.1], who is largely credited for inventing the

least squares method. In the Twentieth Century, a num-

ber of fields outside robotics have studied the making

of environment models from a moving sensor platform,

most notably in *photogrammetry* [46.2-4] and *com-*

puter vision [46.5]. Strongly related problems in these

fields are bundle adjustment and structure from mo-

tion. SLAM builds on this work, often extending the

basic paradigms into more scalable algorithms. Modern

SLAM systems often view the estimation problem

as solving a sparse graph of constraints and applying

Contacts

Avidbots Ex1008

Page 433 of 545

Attachment L

Bibliographic Record - OCLC Number: 954461883

LDR	Record Status	'c' - Corrected or revised
	Type of Record	'a' - Language material
	Bibliographic Level	'm' - Monograph/Item
	Type of Control	' ' - No specified type of control
	Encoding Level	' ' - Full level
	Descriptive Cataloging Form	'i' - ISBD punctuation included
001	954461883	
005	20250804215546.9	
008	Date Entered on File	'160802'
	Type of Date/Publication Status	's' - Single date
	Date1/Beginning Date of Publication	'2016'
	Date2/Ending Date of Publication	' '
	Place of Publication, Production, or Execution Code	'gw ' - Germany
	Illustrations	'a '
	Target Audience	' ' - Unknown or unspecified
	Form of Item	'o' - Online
	Nature of Contents Code	'bf '
	Government Publication	' ' - Not a government publication
	Conference Publication	'0' - Not a conference publication
	Festschrift	'0' - Not a festschrift
	Index	'1' - Index present
	Literary Form	'0' - Not fiction
	Biography	' ' - No biographical material
	Language Code	'eng' - English
	Modified Record	' ' - Record is not modified
	Cataloging Source Code	'd' - Other
006	Form of Material	'm' - Computer file
	Target Audience	' ' - Unknown or unspecified
	Form of Item	'o' - Online
	Type of Computer File	'd' - Document
	Government Publication	' ' - Not a government publication
007	Category of Resource	'c' - Electronic Resource
	Specific Material Designation (c)	'r' - Remote
	Color (c)	'c' - Multicolored
	Dimensions (c)	'n' - Not applicable
	Sound (c)	'u' - Unknown
	Image Bit Depth (c)	'----
	File Formats (c)	'u' - Unknown
	Quality Assurance Target(s) (c)	'n' - Not applicable
	Antecedent/Source (c)	'u' - Unknown
	Level of Compression (c)	'u' - Unknown
	Reformatting Quality (c)	'u' - Unknown

040 GW5XE \$b eng \$e rda \$e pn \$c GW5XE \$d YDXCP \$d AZU \$d COO \$d VT2 \$d MUB \$d S
OI \$d NST \$d ISM \$d OCLCF \$d QGK \$d UAB \$d IAD \$d JBG \$d ICW \$d Z5A \$d ILO \$d
ICN \$d EBLCP \$d ESU \$d CSAIL \$d IOG \$d REB \$d U3W \$d OCLCQ \$d AU@ \$d OCLCQ
\$d WYU \$d UWO \$d UKMGB \$d AUD \$d UWW \$d UX1 \$d OCLCQ \$d DCT \$d ERF \$d U
KBTH \$d LEATE \$d OCLCQ \$d U9X \$d UBY \$d OCLCQ \$d OCLCO \$d COM \$d OCLCO
\$d OCLCQ \$d OCLCO \$d OCLCL \$d OCLCQ \$d UKKRT \$d OCLCL

015 GBB8K6642 \$2 bnb

016 7 019105073 \$2 Uk

019 959235085 \$a 961871055 \$a 985042917 \$a 1005755666 \$a 1012062741 \$a 1036288222 \$a 1048
230911 \$a 1058224569 \$a 1066679801 \$a 1079209254 \$a 1086472846 \$a 1110968101 \$a 11125
81416 \$a 1113382365 \$a 1113754110 \$a 1122810025 \$a 1127188081 \$a 1136470448 \$a 1140745
464 \$a 1154172068 \$a 1237457558 \$a 1259253666 \$a 1480160416

020 9783319325521 \$q (electronic bk.)

020 3319325523 \$q (electronic bk.)

020 \$z 9783319325507 \$q (print)

020 \$z 3319325507

020 \$z 9783319325507

024 7 10.1007/978-3-319-32552-1 \$2 doi

024 8 99968724117

037 com.springer.onix.9783319325521 \$b Springer Nature

050 4 TJ211

072 7 TJFM \$2 bicssc

072 7 TJFD \$2 bicssc

072 7 TEC004000 \$2 bisacsh

072 7 TEC037000 \$2 bisacsh

072 7 TEC \$x 009000 \$2 bisacsh

072 7 TJFM \$2 thema

072 7 TJFD \$2 thema

082 0 4 629.8/92 \$2 23 \$1 <https://id.oclc.org/worldcat/ddc/E366dwMq7tD7fTMtMkt8K9cdJh>

090 \$b

049 LDLL

245 0 0 Springer handbook of robotics / \$c Bruno Siciliano, Oussama Khatib (eds.).

246 3 0 Handbook of robotics

250 2nd edition.

264 1 Berlin : \$b Springer, \$c 2016.

300 1 online resource (lxxvi, 2227 pages) : \$b illustrations (some color)

336 text \$b txt \$2 rdacontent

337 computer \$b c \$2 rdamedia

338 online resource \$b cr \$2 rdacarrier

347 text file

347 \$b PDF

490 1 Springer Handbooks, \$x 2522-8692

504 Includes bibliographical references and index.

588 0 Online resource; title from PDF title page (SpringerLink, viewed August 2, 2016).

505 0 Part A -- Robotics Foundations -- Part B -- Design -- Part C -- Sensing and Perception -- Part D -- Man
ipulation and Interfaces -- Part E -- Moving in the Environment -- Part F -- Robots at Work -- Part G --

Robots and Humans -- Acknowledgements -- About the Authors -- Subject Index.

520 The second edition of this handbook provides a state-of-the-art cover view on the various aspects in the rapidly developing field of robotics. Reaching for the human frontier, robotics is vigorously engaged in the growing challenges of new emerging domains. Interacting, exploring, and working with humans, the new generation of robots will increasingly touch people and their lives. The credible prospect of practical robots among humans is the result of the scientific endeavour of a half a century of robotic developments that established robotics as a modern scientific discipline. The ongoing vibrant expansion and strong growth of the field during the last decade has fueled this second edition of the Springer Handbook of Robotics. The first edition of the handbook soon became a landmark in robotics publishing and won the American Association of Publishers PROSE Award for Excellence in Physical Sciences & Mathematics as well as the organization's Award for Engineering & Technology. The second edition of the handbook, edited by two internationally renowned scientists with the support of an outstanding team of seven part editors and more than 200 authors, continues to be an authoritative reference for robotics researchers, newcomers to the field, and scholars from related disciplines. The contents have been restructured to achieve four main objectives: the enlargement of foundational topics for robotics, the enlightenment of design of various types of robotic systems, the extension of the treatment on robots moving in the environment, and the enrichment of advanced robotics applications. Further to an extensive update, fifteen new chapters have been introduced on emerging topics, and a new generation of authors have joined the handbook's team. A novel addition to the second edition is a comprehensive collection of multimedia references to more than 700 videos, which bring valuable insight into the contents. The videos can be viewed directly augmented into the text with a smartphone or tablet using a unique and specially designed app.

588 0 Print version record.

650 0 Robotics \$v Handbooks, manuals, etc.

650 6 Robotique \$v Guides, manuels, etc.

650 7 Artificial intelligence. \$2 bicsc

650 7 Tool making. \$2 bicsc

650 7 Technical design. \$2 bicsc

650 7 Automatic control engineering. \$2 bicsc

650 7 TECHNOLOGY & ENGINEERING \$x Engineering (General) \$2 bisacsh

650 7 Robotics \$2 fast

655 2 Handbook

655 7 handbooks. \$2 aat

655 7 Handbooks and manuals \$2 fast

655 7 Handbooks and manuals. \$2 lcgft

655 7 Guides et manuels. \$2 rvmgf

700 1 Siciliano, Bruno, \$d 1959- \$e editor. \$1 <https://id.oclc.org/worldcat/entity/E39PBjd3JpTC9Xphdmd9WYp9Dq>

700 1 Khatib, Oussama, \$e editor.

758 \$i has work: \$a Springer handbook of robotics (Text) \$1 <https://id.oclc.org/worldcat/entity/E39PCFQvkJrch8qpXCRpxyGg83> \$4 <https://id.oclc.org/worldcat/ontology/hasWork>

776 0 8 \$i Print version: \$t Springer handbook of robotics. \$b 2nd edition. \$d Berlin : Springer, 2016 \$z 9783319325507

830 0 Springer Handbooks, \$x 2522-8692

856 4 0 \$3 EBSCOhost \$u <https://search.ebscohost.com/login.aspx?direct=true&scope=site&db=nlebk&db=nlabk&AN=1260895>

856 4 0 \$3 ProQuest Ebook Central \$u <https://public.ebookcentral.proquest.com/choice/publicfullrecord.aspx?p=4614797>

856 4 0 \$3 SpringerLink \$u <https://doi.org/10.1007/978-3-319-32552-1>

856 4 0 **\$3** SpringerLink **\$u** <https://link.springer.com/book/10.1007%2F978-3-319-32550-7>
856 4 0 **\$3** SpringerLink **\$u** <https://link.springer.com/book/10.1007%2F978-3-319-32552-1>
856 4 0 **\$u** <https://sites.smu.edu/cul/apps/EBL/Default.aspx?id=4614797> **\$z** Connect to resource
856 4 **\$u** https://discover.gcu.ac.uk/discovery/openurl?institution=44GLCU_INST&vid=44GLCU_INST:44GLCU_VU2&?u.ignore_date_coverage=true&rft.mms_id=991002484373503836 **\$p** 5370974000003836 **\$x** StGIGCU
856 4 **\$u** https://openurl.ac.uk/ukfed:bat?u.ignore_date_coverage=true&rft.mms_id=991003561026602761 **\$p** 5345589550002761
856 4 0 **\$3** Kortext **\$u** <https://app.kortext.com/borrow/344568>
938 EBSCOhost **\$b** EBSC **\$n** 1260895
938 YBP Library Services **\$b** YANK **\$n** 13098417
938 Kortext **\$b** KTXT **\$n** 344568
029 1 AU@ **\$b** 000058477689
029 1 UKMGB **\$b** 019105073
029 1 DKDLA **\$b** 820120-katalog:999887638805765
029 1 AU@ **\$b** 000075317058

Attachment M



65. Domestic Robotics

Erwin Prassler, Mario E. Munich, Paolo Pirjanian, Kazuhiro Kosuge

When the first edition of this book was published domestic robots were spoken of as a dream that was slowly becoming reality. At that time, in 2008, we looked back on more than twenty years of research and development in domestic robotics, especially in cleaning robotics. Although everybody expected cleaning to be the killer app for domestic robotics in the first half of these twenty years nothing big really happened. About ten years before the first edition of this book appeared, all of a sudden things started moving. Several small, but also some larger enterprises announced that they would soon launch domestic cleaning robots. The robotics community was anxiously awaiting these first cleaning robots and so were consumers. The big burst, however, was yet to come. The price tag of those cleaning robots was far beyond what people were willing to pay for a vacuum cleaner. It took another four years until, in 2002, a small and inexpensive device, which was not even called a cleaning robot, brought the first breakthrough: *Roomba*. Sales of the Roomba quickly passed the first million robots and increased rapidly. While for the first years after Roomba's release, the big players remained on the sidelines, possibly to revise their own designs and, in particular their business models and price tags, some other small players followed quickly and came out with their own products. We reported about these devices and their creators in the first edition. Since then the momentum in the field of domestic robotics has steadily increased. Nowadays most big appliance manufacturers have domestic cleaning robots in their portfolio. We are not only seeing more and more domestic cleaning robots and lawn mowers on the market, but we are also seeing new types of domestic robots, window cleaners, plant watering robots, tele-presence robots, domestic surveillance robots, and robotic sports devices.

Some of these new types of domestic robots are still prototypes or concept studies. Others have already crossed the threshold to becoming commercial products.

For the second edition of this chapter, we have decided to not only enumerate the devices that have emerged and survived in the past five years, but also to take a look back at how it all began, contrasting this retrospection with the burst of progress in the past five years in domestic cleaning robotics. We will not describe and discuss in detail every single cleaning robot that has seen the light of the day, but select those that are representative for the evolution of the technology as well as the market. We will also reserve some space for new types of mobile domestic robots, which will be the success stories or failures for the next edition of this chapter. Further we will look into nonmobile domestic robots, also called smart appliances, and examine their fate. Last but not least, we will look at the recent developments in the area of intelligent homes that surround and, at times, also control the mobile domestic robots and smart appliances described in the preceding sections.

65.1 Mobile Domestic Robotics	1730
65.1.1 Domestic Floor Cleaning.....	1730
65.1.2 Domestic Window Cleaning.....	1737
65.1.3 Pool Cleaning.....	1739
65.1.4 Lawn Mowing.....	1740
65.1.5 Sports Robotics.....	1742
65.1.6 Tele-Presence.....	1745
65.2 Enabling Technologies	1747
65.2.1 Sensing and Obstacle Avoidance.....	1747
65.2.2 Localization and Mapping.....	1749
65.2.3 Navigation and Coverage.....	1753
65.3 Smart Homes	1754
65.3.1 Gator Tech Smart House.....	1755

65.3.2	Aware Home	1756	Video-References	1757
65.3.3	SELF – Sensorized Environment for Life	1756	References	1757

65.1 Mobile Domestic Robotics

The first mention of a domestic cleaning robot dates back to 1985. The device, nicknamed *Robby*, was developed by Hitachi starting in 1983 and officially carried the name HCR-00 (Fig. 65.1). Robby was equipped with a gyroscope to keep track of its position and a rotating sonar scanner for obstacle detection.

65.1.1 Domestic Floor Cleaning

Very similar to today’s cleaning robots described further below, Robby already mapped its environment and used the map for path planning. HRC-00 remained a prototype like its successors HRC-01 to HRC-03.

**First Generation
of Domestic Cleaning Robotics (1985–1999)**
Also in 1985, the Swedish appliance manufacturer Electrolux started the development of a concept vacuum cleaner *Stardust*. The device was equipped with eight fixed sonar sensors and one rotating sensors for obstacle detection. To maintain its orientation Stardust used

an infrared sensor that tracked an infrared light bulb mounted to the ceiling. In 1988 Stardust was presented to the public at Domotechnica in Cologne, Germany, one of the largest fairs worldwide for domestic appliances [65.1].

About five years later, between 1989 and 1991, Panasonic undertook an effort to develop a domestic cleaning robot that led to two prototypes, one of them, *Brownie*, is shown in Fig. 65.1. Brownie was a battery-powered vacuum cleaner with a diameter of 40 cm, and at 18 kg, a super heavy weight compared to today’s domestic cleaning robots. It was equipped with a gyroscope, a ring of sonar sensors, and a dust sensor. Likely, it was limitations in battery technology for driving such a heavy robot, as well as the total cost, which prevented Brownie from become anything more than a prototype.

Six more years went by before Minolta presented its *Minolta Cleaning Robot* and the first European player, Electrolux, its Robot Vacuum Cleaner, later called Trilobite, which was presented to the public in BBC’s TV show *Tomorrows World* in 1996.

The devices had already become significantly smaller and lighter in these six years. The Minolta robot had dimensions of 321 mm × 320 mm × 170 mm and weighted about 8 kg. It was powered by nickel metal hybrid battery and used sonar and tactile sensors for obstacle detection, a cliff-sensor to discover staircases, and a gyroscope to track its position and orientation. Electrolux’ robot also had a sophisticated sonar system, which let it follow contours and even return to a homing position. It already had the size and shape of later floor cleaning robots.

Five more domestic cleaning robots made their first appearance during this time, which we call the infancy of domestic cleaning robotics. Two of them only reached the proof of concept level: *AutoCleaner* from InMach Intelligent Machines – the German startup later developed low-cost navigation systems for the professional cleaning robot Robo40, which became commercially available in 2007 – and Koala from EPFL-Lami. AutoCleaner was the first domestic wet-cleaner proof of concept, which cleaned the floor using a rotating microfiber towel that was pulled through a water tank. Koala used a suction spout to reach corners. Two more robots reached at least the status of industrial prototypes, but were never commercialized: Dyson’s *DC06* and *Cye* from *Probotics*. DC06 was a unique domestic



Fig. 65.1 First generation of domestic floor cleaning robots (1985–1999)



Fig. 65.2 Second generation of domestic floor cleaning robots (2000–2008)

cleaning robot. It differed not only in terms of the announced list price of approximately US\$ 4000, but also in many other aspects: Dyson claimed that it had three onboard CPUs, more than 50 sensors for obstacle avoidance, cliff-detection, localization and more. On top of that came the unique Dual Cyclone cleaning technology. The DC06 never made it to the market. Neither did Cye. Cye was a small mobile robot that could pull and push a (semi-)regular vacuum cleaner. It was announced as the *first personal robot* which could not only vacuum the floor but also serve coffee or deliver mail. It is unclear if Cye ever delivered coffee or mail or vacuumed a real living room. It disappeared as quickly as it appeared. The last out of the nine first robots that we call first generation was the Kärcher RoboCleaner. Its successor RC3000 was amongst the first cleaning robots that had a docking station at which the robot could not only recharge its battery but also unload the dust, which it had collected. We will come back to the Kärcher RC3000 in the next section.

The first generation domestic floor cleaning robots shown in Fig. 65.1 were the ones presented to the public. However, they were by far not the only ones that were developed in the period between 1985 and 1999. By browsing over the patents in the field of cleaning robots during that period it becomes obvious that many of the big appliance and electronic device manufacturers worldwide performed research and development on domestic cleaning robots. Besides the ones mentioned above, one can find such big names as Nintendo, Mat-

sushita, Sanyo, Samsung, Honda, Procter & Gamble, Electrolux, Philips, Henkel and more. So, although only a few players dared to go public, domestic cleaning was already on the radar of many international enterprises. What kept them away from taking the next steps is not hard to guess. It was the risk of getting into a rather conservative and tight market – the cleaning business is extremely conservative – with a semimature technology carrying a price tag that looked quite differently than the one on well-established technology, such as traditional vacuum cleaners.

The Second Generation of Domestic Cleaning Robotics (2000–2008)

Figure 65.2 shows the second generation of domestic cleaning robots. What made the difference between the first and the second generation? The amount of experimentation regarding the overall design decreased, with the majority of manufacturers settling on the disc shape. The number of sensors decreased, as well as their level of sophistication. Sophisticated and expensive sensors such as sonar largely disappeared. Most designs were limited to very few, simple, and inexpensive sensors such as bumpers, simple one-dimensional range sensors, typically based on infrared (IR), and cliff sensors. This was most likely a rather painful lesson learned from the designs in the first generation. What roboticists considered the state of the art in mobile robotics: environmental sensing, map building, range sensors with high quality angular and range res-

olution, and combined localization and mapping, were far too expensive to be built into a domestic cleaning robot. Those robots had to compete with regular vacuum cleaners in a price range of a few hundred dollars. This competition imposed very harsh cost limits for the robotic components that could be built into a domestic cleaning robot: 50 to 100 USD and often less. These cost constraints had a significant impact on a very fundamental expectation of what cleaning robots should achieve: systematic coverage of the area to be cleaned. Cleaning was understood to be synonymous with covering an area with a systematic and intuitive motion while applying some type of cleaning operation. Most of the patents filed for the first generation of domestic cleaning robots mentioned above described cleaning robots that were supposed to cover their workspace in a systematic fashion.

Systematic coverage, however, is impossible without absolute positioning and without decent knowledge of the environment. The second generation of domestic cleaning robots had to provide a solution for this dilemma of systematic coverage with cheap and simple sensors, which must not cost more than several tens of dollars.

The solution to this dilemma was already proposed in some early prototypes. The idea was to waive the requirement of systematic coverage, which would also involve intuitive motion patterns, and instead produce *semisystematic* coverage and *semi-intuitive* motion patterns. This semisystematic coverage was achieved by a combination of random motion, hard-coded motion patterns such as spiral- or meander-shaped motions, which – considered alone – reflects some systematic coverage, and some other systematic and intuitive motion patterns, for example, following contours of walls or other objects in the workspace. The theory of stochastic process states that the mean squared distance of a particle that performs a random walk with respect to the origin of its motion increases proportionally with time. That implies that the particle moving in confined space will cover that space in a finite amount of time.

First-generation domestic cleaning robots like the Electrolux Robotics Vacuum Cleaner (later Trilobite) or the Kärcher Robot Cleaner (later Kärcher RC 3000) already used random motion combined with hard-coded motion patterns to achieve a certain degree of coverage. But apparently the insight that the attribute *robotic* in front of *vacuum cleaner* was not enough of a sales argument to justify a price three to five times higher than that of a regular vacuum cleaner was not so easy to digest. Some of the second-generation cleaning robots were offered for a price on the order of 1500 USD or even more. In the end these robots shared the same fate: they become shelf warmers.

This is also due to the fact that in 2002 a robot was launched which was to herald a breakthrough in domestic cleaning robotics: Roomba. Its creator, iRobot Inc., had learned the lesson that others were still struggling with: if you want to sell a domestic appliance, better sell it for a price that is known for domestic appliances. For the sake of fairness one has to mention that the traditional vacuum cleaner manufacturers had to design the robots so they matched the quality and performance level that could be expected from the brand name. They could not take the risk of making their product too poor. Therefore it easily became over specified compared to what the market required at the time. New manufacturers such as iRobot in contrast had no brand name to defend [65.1].

If it is a totally new device, which may not only cause excitement but also concerns and reservations, sell it for less and not for more. When Roomba entered the market it was sold for 199 USD. That was a price, which did not cause customers to think about whether they really needed it, whether the quality was good enough, or whether the device would get into every corner of a room or underneath every couch or bed. The creators of Roomba were also smart enough not to call it a robot. This that prevented many customers from developing wrong expectation of what robots could or should do.

The robot technology that was built into the Roomba was everything but new or revolutionary. Roomba used a suspended front-shield for contact-sensing, a low-cost infrared range-sensor for contour-following, a cliff-sensor to prevent it from falling down stair-cases, and it could detect photoelectric barriers, so-called *virtual walls*, which kept it from leaving a room or a certain part of its workspace. Roomba further used the combination of random motion and hardcoded area-covering motion patterns to achieve a certain level of coverage. All this technology was known before Roomba.

Still Roomba can be seen as a milestone not only in domestic robotics but also in robotics at large. Why is that? It was the first time in robotics history that robotics was no longer synonymous with high-tech, high-price. Roomba showed that automation of everyday service tasks, such as domestic cleaning, could be achieved with a moderate effort in terms of hardware and at a decent price, given that one accepts some graceful degradation in the overall performance. There is no such thing as a free lunch, not even in robotics and this degradation in the performance is the price to be paid. What turned Roomba into a milestone was the cost-effective design, where the limitations of low-cost hardware – in Roomba's case the sensors – were balanced out by smart heuristics for problem-solving. This

Table 65.1 Technical specification of a selection of second-generation domestic cleaning robots

					
Manufacturer	iRobot	Sharper Image	Electrolux	Kärcher	Yujin
Model	Roomba	eVac	Trilobite 2.0	RC 3000	iclebo
Drive	Differential	Differential	Differential	Differential	Differential
Sensors	Suspended front shield as contact sensor, IR range sensor, four IR cliff sensors, dust sensor	Touch sensor, cliff sensor, IR for wall-following	180° sonar sensors (1 transmitter, 8 receivers), infrared cliff sensor, magnetic stripe detector, suspended front shield as contact sensor	Suspended front shield as contact sensor, four IR cliff sensor	Main brush, side brush anti-bacterial filter
Mapping and localization	n/a	n/a	n/a	n/a	n/a
Navigation/coverage	Random motion with bang and bounce, contour following, spiral motion	Random motion with bang and bounce, spot cleaning with star pattern	Wall-following, random motion with obstacle avoidance	Random motion with bang and bounce, spot cleaning with see-saw motion pattern	Heuristic pat-terns: random, circular, zig-zag, wall following
Cleaning technology	Side brush, two counter-rotating brushes, suction pump	Rotating brushes, vacuum pump	Rotating brush, suction pump	Rotating brush, suction pump	Main brush, side brush
Run time (min)	60–90	15–45	60	20–60	150
Performance	–	–	28 m ² /h	15 m ² /h	–
Docking/recharging station	Yes	No	Yes	Yes	no
Size (Ø/h) (cm)/(cm)	35/8,25	32/14	35/13	28/10	35/9
Weight (kg)	2,7	3	5	2	4
Year of launch	2002	2004	2004	2003	2005
Price range	250 USD	100 USD	1300 EUR	1350 EUR	530 USD

design of a commercially viable product, at a price point competitive with nonrobotized solutions, turned Roomba into the first successful domestic cleaning robot, and the most frequently sold robot in the past 50 years.

In Table 65.1 we give an overview of the technical specifications of some of these second generation robots. A more complete overview is included in 065-bib43

The idea to develop and commercialize a domestic cleaning robot at a price level of 199 USD, comparable to the price of traditional domestic appliances, was very appealing to consumers. However, it was not an overly profitable business model, even if Roomba was manufactured in a low-wage country at a cost of significantly less than 100 USD. Extreme cost pressure, however, often compromises the quality of a product. So a decision to adhere to this low price of below 200 USD may also have been a decision against the quality of the prod-

uct. Today nearly a dozen models of Roomba are on the market, which differ mostly in terms of their cleaning technology and extra features. Their prices are in a range from 250 to 900 USD.

The Third Generation of Domestic Cleaning Robotics (2009–2012)

The second generation of domestic cleaning robots also provided their developers and distributors with a number of painful and partly contradictory insights:

- A domestic cleaning robot sold at a price that is significantly beyond the price of comparable non-robotic devices, runs a high risk of failing, because many customers may not be willing to pay extra money for a cleaning robot just because it is a robot.
- A domestic cleaning robot sold at the same price level as an inexpensive comparable appliance, runs a high risk of failing, because the low price may

compromise the quality and functionality of the product, and many customers may not be willing to pay even a rather low price for a product that is known to be of poor quality.

- For most customers – traditional, as well as technophiles – efficient cleaning requires systematic and efficient coverage of the workspace. A domestic cleaning robot that uses sophisticated sensors to achieve systematic and efficient coverage may easily end up with a price tag that is significantly above the price of a comparable nonrobotic device and hence runs a high risk of failure.
- A domestic cleaning robot, which refrains from using expensive sensors to achieve systematic coverage but instead uses low-cost sensors for collision avoidance, fall protection, and confinement as well as using random motion combined with some hard-coded motion patterns to achieve a certain level of coverage, runs a risk of not satisfying those customers who expect systematic cleaning and coverage.

From a developers point of view these lessons sound as if customers expect not less than to square the circle. The truth is that customers do not care about squares and circles. They expect value for money.

With the insights above one could classify potential customers of domestic cleaning robots into three categories:

- Customers who only care about the price and not so much about quality and functionality. The criterion for this class of customers was obvious: reduce the price as much as possible without ignoring that there may be a bottom line for what customers might expect in terms of quality and functionality.
- Customers who care about the price and quality, but are willing to adjust their expectations of functionality and efficiency. The criterion for such customers was to reduce the price but only to a certain extent, which does not compromise the quality of the product too much. Expectations of such customers in terms of efficiency (i.e., systematic coverage) can be compensated by auxiliary equipment such as automatic charging stations.
- Customers, who care about price, quality, and efficiency. Apparently these customers are the most demanding. None of the second-generation domestic cleaning robots has really managed to satisfy them. The criterion to serve them would be to develop a low-cost navigation system based on low-cost sensors that provides systematic coverage.

As satisfying the last group of customers, which is also likely the largest of the three groups, seemed

nearly impossible, the third generation of domestic cleaning robots have diverged somewhat. Some manufacturers have focused on products that serve the first category of customers, some serve the second group and some even made an effort to square the circle.

May they be successful in the end or not, what cannot be overlooked is the explosion of the number of manufacturers and distributors of domestic cleaning robots in the past five years. At the end of the 2012, twenty-seven years after the first mention of a domestic cleaning robot to the public, Amazon alone listed more than 131 results under the key word *robot vacuum cleaner*, with 14 manufacturers and suppliers. The business to business (B2B) portal www.made-in-china.com lists more than 71 companies for *robot vacuum cleaner* and a total of 875 products. Some of these 875 robot vacuum cleaners look surprisingly similar to the products sold over B2C platforms under very-well known brand names.

In Fig. 65.3 we show some of the products, which primarily intend to serve customers of the first category and therefore stayed under a price of 200 EUR. Figure 65.3a shows a model series of six cleaning robots made by XRobot of the Chinese manufacturer Shenzhen Silver Star Intelligent Electronics Ltd. In B2B trading these robots cost between 64 and 102 USD per piece at a minimum order of 500 pieces. Most models of this series are also sold under the brand name of European and American enterprises. Figure 65.3b shows another series of still rather cheap robotic vacuum cleaners. The attentive reader may notice that the robots in the left and the right figure are not totally unique. Some just have different names and different prices. This is not entirely unintended. It just illustrates that in domestic cleaning robotics the value creation chain is no longer limited to the developers and manufacturers, just as in every other business.

Apart from a few minor details, all robots in Fig. 65.3 use very similar, though not to say the same, technology. They use very few and very cheap (contact, cliff, sometimes dirt) sensors and random motion combined with preprogrammed motion patterns. The vacuum technology consists of a rotating brush, sometimes combined with a small fan. Not surprisingly, robotic vacuum cleaners like the ones in Fig. 65.3 are often called Roomba-clones. The collection of robots in Fig. 65.3 is neither representative nor comprehensive.

Roomba has undoubtedly written robot history. It is the merit of Roomba and its developers that domestic cleaning robots are no longer considered as gadgets but as real appliances. Roomba has opened the door for all



Fig. 65.3 Third generation of domestic floor cleaning robots: low-cost, low-tech



Fig. 65.4 The seventh generation of Roombas

the cleaning robots mentioned above, be they Roomba-clones or not. And Roomba is the by far the best-selling robot ever.

In Fig. 65.4 we show some of the grandchildren and great-grandchildren of the very first Roomba, which started in 2002. The Roomba has gone through several facelifts in these ten years. The latest series, the Roomba 700 is the sixth generation of Roombas. It has matured quite a bit in many respects. It has matured in terms of handling, cleaning technology, obstacle avoidance and navigation technology. At the same time Roomba is delivered in more than a dozen versions, ranging from 250 to 900 USD. These versions differ in the design, in the sophistication of the user interface, in the navigation and coverage strategies, and in the details of the cleaning technology.

According to iRobot more than 6 million units have been sold in the ten years of its existence. Roomba has certainly satisfied quite a number of customers. It is built with good quality, and comes with a self-charging

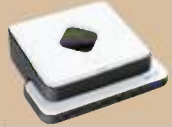
home base and other auxiliary equipment. What has not changed, however, is the basic strategy for covering the workspace. The sixth generation just like the first uses random motion with some precoded motion patterns and heuristics.

This raises one rather fundamental question. Will Roomba's success story continue? Will it maintain its market position? Or has Roomba possibly reached or even passed its summit? These questions will not be answered before the third edition of this book, as any statement in this direction would be pure speculation.

In any case, there are a few recent domestic cleaning robots, which demonstrate that covering a work space in a systematic fashion does not necessarily require expensive sensing, that would push the price for a domestic cleaning robot to a level which customers would not be willing to pay.

Four different key technologies enable the devices shown in Table 65.2 to localize themselves and navigate

Table 65.2 Systematic cleaning at low-cost. The information in this table was partly collected from the websites and the technical documentation of the manufactures and partly from public websites such as www.botroom.com, staubsaugerroboter-test.org, www.robotreviews.com, www.geek.com, gizmodo.com, www.engadget.com

					
Manufacturer	LG	Samsung	Neato Robotics	Iclebo	Evolution Robotics/iRobot
Distributor			Vorwerk	Philips	Dirt Devil EVO/iRobot
Model	Hom-Bot 3.0	Navibot SR 8895 Silencio	Neato XV-21	Philips FC9910/01 HomeRun/ iclebo smart	Mint 4200
CPU	32 bit	2 CPUs			ARM7
Drive	Differential	Differential	Differential	Differential	Differential
Sensors	Camera (ceiling), camera (floor), sonar & infrared for collision avoidance cliff sensors, gyroscope, acceleration	Range sensors camera, cliff sensors, collision sensors, gyro	1-D laser range finder, cliff sensor, gyroscope, acceleration,	Camera (ceiling), cliff sensor, infra red for collision detection, gyroscope,	Northstar Indoor GPS, cliff sensors, bumper, gyro
Mapping and localization	SLAM	SLAM based on ceiling pictures at 30 fps (Visionary Mapping System)	SLAM based on laser range finder (onboard Room Positioning System (RPS))	SLAM based on ceiling pictures	SLAM with Northstar, map-building
Navigation/coverage	Systematic	Systematic	Systematic	Systematic	Systematic
Cleaning technology	Brush roller, 2 side brushes	Brush roller, 2 side brushes	Bristled brush	Brush roller, 2 side brushes, micro-fiber mop	Dry or wet (or pre-moistened) microfiber cloths
Run time (min)	75	90	90	70	180
Performance	–	80 m ² /h	–	40 m ² /h	–
Docking/recharging station	Yes	Yes	Yes	Yes	No
Size (Ø/h) (cm)/(cm)	36/9	35/8	34× 34/10	35/10	31× 11/ 29
Weight (kg)	3,2	3,2	5	4	2
Year of launch	2011	2011	2012	2011	2010
Price range	US\$ 799.00 (Amazon.com)	EUR 349.00 (Amazon.de)	US\$ 350.00 (Amazon.com)	EUR 499.00 (Amazon.de)	US\$ 174.00 (Amazon.com)

in an unknown domestic workspace and systematically cover that space:

- Visual odometry, with a camera pointing to the ceiling, combined with simultaneous localization and mapping (SLAM) used by LG Homebot, Samsung Navibot SR 8xxx, iclebo smart and Philips Home-run,
- Localization, using infra read patterns projected to the ceiling, and mapping based on contact information used by Mint,
- Proprioceptive motion estimation, using inexpensive inertial measurement units (including inexpensive gyroscopes and accelerometers)

- Simultaneous localization and mapping (SLAM), using, for example a one-dimensional laser rangefinder like Neato XV-xx.

These technologies by themselves are not entirely new. As a matter of fact methods such as simultaneous localization and mapping, visual odometry, and virtual landmark based navigation are not state of the technologies form a scientific point of view.

The true achievement of the developers of the above systems, which cannot be assessed high enough, is that they managed to reduce the cost of these technologies and at the same time make them robust enough for 24/7 operation. These key technologies will be described in

more detail in Sect. 65.1.2 Enabling Technologies. A review of domestic vacuum cleaning robots and some criteria to evaluate their performance are presented in  VIDEO 727 and  VIDEO 729.

65.1.2 Domestic Window Cleaning

Window cleaning does not seem to be significantly more pleasant housework as floor cleaning. Notwithstanding, robotic window cleaners have not experienced similar attention or progress as floor cleaners. As a matter of fact, today there are only three commercial robotic window cleaners on the market, shown in Fig. 65.5. www.made-in-china.com does not even list one entry for *window cleaning robot*, although two of the three commercial products are Chinese brands. Why is that so, given the fact that domestic floor cleaning has become a billion-dollar business? The answer to this question has two parts: an economic and a technical one.

The economic one is that windows in private homes are cleaned far less often than floors. Customers may hesitate to buy expensive equipment for a task that needs to be done once a month or even less. So the market for domestic window cleaners is presumably much smaller than that for floor cleaners.

The technical one refers to the technical hurdles that have to be overcome. The technical problems that robotic window cleaners have to face are:

- Adhesion to a vertical, fragile surface, which possibly needs to be moistened in order to be cleaned, and related, the power supply necessary to produce that adhesion.
- Requirements for cleaning performance; people may tolerate if the floor is not 100% clean, but nobody would buy a window cleaner that leaves streaks on the window.

While floor cleaners do not bother about gravity and falling down unless they are near staircases or ledges, gravity is an essential problem for window-cleaning robots, and the solutions are usually not very cheap. Special mechanisms have to be designed for secure motion. Typically special tether mechanisms prevent the robots from falling. Special locomotion mechanisms have to create enough adhesion force to hold a robot attached to a flat, vertical, damageable surface such as glass and at the same time move the body up and down and sideways. These mechanisms have to be small and light and create enough adhesion forces as well as low energy and resource consumption.

The two robotic window cleaners shown in Fig. 65.6, RACOON and QUIRL, are research prototypes, which were developed by Fraunhofer Institute for

Manufacturing Engineering and Automation (IPA) in Germany. In these prototypes the adhesion problem was solved by means of suction cups. RACOON used caterpillar drives that were equipped with passive suction cups. Passive means that the system does not actively create a vacuum in the cup. Rather a small valve aerates or seals the suction cup depending on the position of the cup along the drive. Drives with passive suction cups have the advantage of moderate energy consumption. They have, however, one severe disadvantage. They tend to lose their adhesion after a while. The reason for this is the torque that acts on the center of gravity of the system. Due to this torque there is a traction force acting on the upper cups while at the same time pressure is exerted on the lower cups. Without any attractive force acting on the upper cups the adhesion there gets weaker and eventually the system falls. Therefore passive suction cups are rarely practical.

An apparent solution to this problem is the use of active suction pumps, which generate a vacuum under the upper suction cups. This solution prevents the system from falling. However, supplying the vacuum to the cups makes the system significantly more complex, heavier, and larger. Researchers at Fraunhofer IPA [65.2] have therefore invented a smart solution, which gets by with passive cups, but gets around the problem of decreasing adhesion. The solution uses a spacer at the rear of the vehicle. This spacer neutralizes the torque around the center of gravity which is typical for a systems with passive suction cups. The spacer causes a traction force which acts on the lower suction cups. This traction force creates a torque around the spacer, which counteracts the torque around the center of gravity and also causes a pressure on the upper suction cups. RACOON was presented at the Hannover Fair in 2002 and got quite some attention. But it never became a product. Neither became its successor QUIRL.

In QUIRL, the number of components, the weight, and the size of the system were significantly optimized. The main functions *cleaning*, *holding*, and *moving* were unified in one single component. QUIRL consisted of

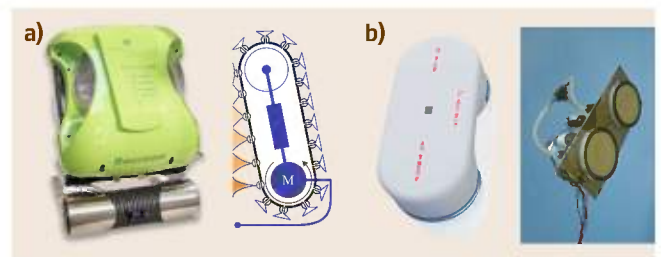


Fig. 65.6 (a) RACOON and (b) QUIRL, two early research prototypes of domestic window cleaning robots



Fig.65.5a–c Commercial robotics window cleaners (a) Windoro, (b) Winbot, (c) Hobot

two vacuum cups that were attached to a common frame which were driven by two separate motors, which rotated independently of each other. The overall motion of QUIRL could be controlled by selecting the velocities and rotational directions of the vacuum cups. If the motor of one vacuum cup was turned off and the cup did not rotate, QUIRL rotated around this fixed cup. If both motors and cups rotated in the same direction this led to an overall rotation of QUIRL about its vertical axis. If both drives rotated in the opposite direction at exactly the same velocity then QUIRL made a linear motion. If both drives rotated in the opposite direction but their velocities were not identical then the translational motion was superimposed by a rotational motion and QUIRL moved along a curved trajectory. In order to clean the surface some cleaning mechanism or tool needed to be fixed to the vacuum cups. By attaching, for example, specific cleaning towels in the cups the abrasion effect was increased and a very good cleaning performance could be achieved.

The field of robotic window cleaning has not made as much progress as domestic floor cleaning, but it has still made progress. In Fig. 65.5 we show three commercially available window cleaners – Windoro WCR 1001, Winbot 7, and Hobot 168 – available today.

In 2010 the South-Korean enterprise Ilshim Global Co. Ltd. introduced its commercial window cleaner Windoro (WCR). Windoro consists of two modules, a navigator module and a cleaning module that are held together by two strong neodymium permanent magnets, whose distance can be adjusted. The navigator module and the cleaning module are placed on the inner and outer side of the window, respectively, like a sandwich with the glass pane in between. The two modules operate as a tandem. The navigator module has a differential drive system with two wide-based rubber wheels. Also the cleaning unit moves on two rubber wheels, which serve as spacers for the four spinning cleaning pads. When the navigator module moves, the cleaning module on the other side of the window moves with it. The linkage by permanent magnets obviously has one rather fundamental advantage. As long as the window does not exceed a certain thickness, namely 28 mm, the permanent magnets hold the robot safely attached

to the window. Windoro cannot fall, even if its battery runs out of power. For that reason Windoro does not need any safety mechanism like a safety rope. Unfortunately there is a price to pay for this: it is very difficult and unhandy if not impossible to use Windoro if you cannot open the window that you want to clean.

Windoro first explores the width and height of the window before it then starts to clean the window from the top to the bottom in a zig-zag motion, with a velocity of 8 cm/s. While the robot moves over the window the cleaning module sprays a cleaning solution onto the window surface. With four spinning microfiber pads the cleaning module removes dirt and sprayed solution from the surface.

The second commercial window-cleaning robot is Winbot, a product of the Chinese company Ecovacs, which has established its own brand. Winbot was first introduced in 2011. The most recent version Winbot 7 was presented at the electronics fair CES (Consumer Electronics Show) 2013 in Las Vegas. Ecovacs also manufactures and distributes the robotic vacuum cleaner family Deebot.

Winbot uses two suction rings and a vacuum pump for the adhesion at the window. The outer ring also serves as a safety mechanism. If the air pressure in the ring increases, that means Winbot has reached the edge of a window plane, it backs up, turns around and moves in a different direction. A second suction cup serves as a safety anchor for Winbot while it moves. It is connected to Winbot by a safety rope and catches the robot if it falls. Winbot can move at a velocity of approximately 15 cm/s. Its drive system consists of two differentially driven anti-slip rubber tracks.

After it has been turned on, Winbot, like Windoro, first explores the height and width of the window, then calculates a zig-zag path that covers the window area and finally executes this path. Winbot has no *active* cleaning technology such as spinning pads. It uses two micro-fiber towels that are attached to two plates at the front and at the rear of the robot. The micro-fiber towel in the front has to be moistened before Winbot starts moving. It resolves and removes the dirt. A rubber blade behind the front towel removes the remaining moisture.



Fig. 65.7 Robotics pool cleaners

The micro-fiber towel in the rear of the robot finally dries the window.

The third, and most recent, system, Hobot 168, is a product of the Taiwanese company Hobot Technology Inc. It was first presented at IFA 2012 (Internationale Funk Ausstellung) in Berlin and launched in summer 2013. It looks surprisingly similar to Quirl in Fig. 65.5 and apparently uses a similar adhesion and locomotion mechanism, namely two rotating vacuum cubs, to move across the window.

The preceding paragraphs read as if the times are over where windows had to be cleaned by hand. Regrettably this is not the case. Although all three devices reasonably solved the adhesion problem their overall performance is modest. Several tests by housewives and magazines came to the same sobering conclusion: There is a lot of noise and very little cleaning. As a matter of fact, although the navigation problem on a vertical rectangular surface, which is free of obstacles apart from the window frame, seems to be a solvable one, Hobot 168 and Winbot showed a rather poor performance in terms of systematic coverage. After a semisuccessful effort to explore the width and height of the window, Winbot, in one of the tests, moved more or less erratically for several minutes, before it gave up somewhere along the road. Several of the currently available commercial window cleaning robots are reviewed in [VIDEO 734](#), [VIDEO 735](#), [VIDEO 736](#), and [VIDEO 737](#).

65.1.3 Pool Cleaning

While robotic floor cleaner and window cleaner were still struggling to get rid of the image of only being the crazy ideas of engineers and researchers at the be-

ginning of the millennium, pool-cleaning robots were already well-established products. This may be due to the fact that the challenge of cleaning a rectangular pool is rather modest and so is the *robotic technology* used in robotic pool cleaners. It may also be due to the fact that pool owners belong to a class of customers who did not create the same price pressure as the ordinary housewife.

First patents on self-propelled pool cleaning devices date back to 1965, three years after the first industrial robot was installed. Ferdinand Chauvier, a South African engineer, could possibly be considered the father of automated pool cleaning. He developed several generations of devices for pool cleaning before he finally marketed *Kreepy Krauly*, the first automated pool cleaner in 1974. Kreepy Krauly was not only the first one of its kind but also the very first domestic service robot ever, 15 years before Joe Engelberg published his book *Robots in Service* at MIT Press in 1989 and coined the term *service robot*. Since then the technology of pool cleaning robots hasn't changed much.

As one can see in Fig. 65.7, most pool cleaners have track drives, which are operated differentially. Typically the motors that drive the tracks are also connected to the front and rear scrubbing brushes, which clean the pool surface, while the robot moves. While tracked drive system were rather common for the early generations of pool cleaners, newer pool cleaners also use wheel drives, for example, the Polaris 9400 from Zodiac Pool Systems. A wheel drive can be advantageous because the space between the bottom of the robot and the ground allows a better water flow and a higher throughput. Also Zodiac claims that Polaris 9400 has higher maneuverability.

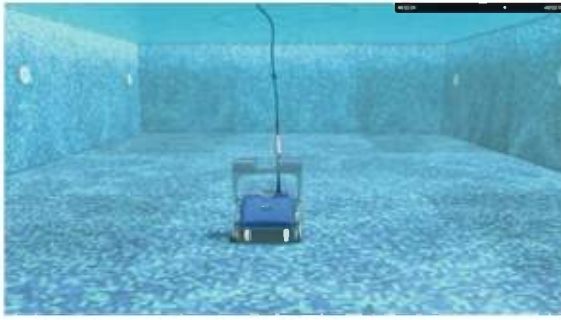


Fig. 65.8 Example of the navigation and cleaning strategy of a robotic pool cleaner

Since most sensors, which are used in mobile robots or in aerial robots do not work under water, the sensor modalities that are used by pool cleaners are manageable. Unfortunately the manufacturers do not disclose much about the sensor technology and technology in general used in their system. So we need to speculate a bit about how the behaviors shown in the commercials are internally implemented.

Most pool cleaners claim to be capable of avoiding obstacles. When they sense a collision, the back up for a certain distance, reverse the direction of motion and then continue, possibly on a track parallel to the one the led to the obstacle. They can *recognize* the walls of a pool, which form the borders of their workspace. They can even climb up the walls of a pool, float along the perimeter and then submerge again. They can *explore* the length and width of a pool. They can drive a certain distance before they change direction.

All these behaviors require a combination of several of the following sensors: *odometry* to measure traveled distance, an *inclinometer* to sense if the robot starts moving upward, e.g., when it keeps pushing against a wall and the front wheels start moving in a vertical direction, *contact sensors* to detect when the robot collides with an obstacle – be it a wall of the pool or a real obstacle on the ground of the pool – *sensor to measure the motor current*, which can be used in addition or instead of a collision sensor to detect if the robot pushes against an object, and possibly an *inertial measurement unit* to correct the heading when driving. Polaris and possibly other systems as well use an accelerometer to constantly determine its position in the pool. A sensor modality, which might be used underwater for obstacle detection and avoidance is laser, with wavelengths in the lower nano-meter range (e.g., 405 nm blue laser). But it is unknown if this principle has been considered for pool-cleaners.

In terms of *navigation and coverage* pool cleaners follow similar strategies to those floor cleaners, which we roughly classified as Roomba-clones. Earlier pool

cleaners used random motion pattern. The newer ones shown in Fig. 65.7 use certain heuristics and strategies to perform some form of localization and exploration of the pool. The Maytronic's Dolphin, for example, first explores the length and the width of a pool. After being dropped into the pool and floating to the bottom of the floor it crosses the pool until it hits the first wall. With the support of a thruster it climbs up the wall until it reaches the surface. There it hovers to the side before it submerges again and moves back to the ground of the pool. Next it moves to the opposite side of the pool, climbs up the wall, hovers to the side and glides back to the ground. While moving on the ground from one wall to the other Dolphin measures the distance from wall to wall. Once it glides back from the second wall it moves halfway back toward the other wall. It then makes a 90° turn and repeats the exploration for the second set of opposite walls. After the exploration is completed, Dolphin knows the length and width of the pool and plans a pattern of parallel and orthogonal tracks as shown in Fig. 65.8 that in the end covers the entire pool.

Removing dirt and debris from the ground and the walls of a pool, requires loosening the dirt – if it is not loose already – and to soak it into a container, otherwise the dirt would only be circulated in the pool. The container is typically a jet pipe with a filter at one end that holds the dirt back while the water flows through the pipe back into the pool. Soaking in water and debris from the floor into a filter and pumping it back into the pool requires significant suction power. This is the reason why pool cleaners typically have an external power supply. This motor serves as a pump and as a thruster at the same time. Both effects together allow pool cleaners to easily climb up vertical walls. Using inflation at the bottom of the robot, and a water jet ejected at the top, enough traction power is created that both tracked and wheeled pool cleaners can drive up vertical surfaces.

As mentioned earlier, the cleaning technology, besides the water pump and the filter to keep back the debris, consists of a system of counter-rotating rubber brushes which brush the debris underneath the pool cleaner, where it ends up in the intake socket of the water pump.  [VIDEO 739](#) and  [VIDEO 740](#) compare a selection of home pool cleaning robots.

65.1.4 Lawn Mowing

Together with robotic pool cleaners and domestic floor cleaners, robotic lawnmowers today count as regular everyday products. People no longer consider them mystical pieces of technology, which have their own life and which at times become so autonomous that the user no longer knows what they are after.

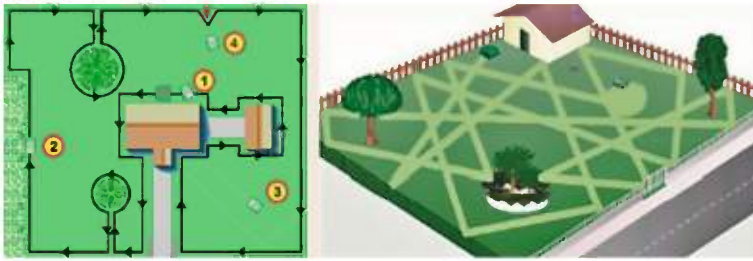


Fig. 65.9 Example for virtual fence and coverage strategy (random motion) of robotics lawnmowers

It is not surprising that robotic lawnmowers have a lot in common with domestic floor cleaners. They have to cover a workspace of a certain size with as little interaction with the owner as possible. They have to perform a certain operation to the surface such as cleaning the floor or mowing the lawn. They must not collide with any obstacles and if they do, they should at least not cause any damage. They must not get stuck anywhere in the environment and they should not leave the workspace without authorization.

The challenge for lawnmowers, much the same for domestic floor cleaners, is the systematic coverage of the workspace, which in turn requires precise positioning and mapping of the workspace. Given the fact that the price pressure is not as back-breaking as for robotic floor cleaners – there are not that many robotic lawnmowers which costs less than 1000 EUR – why not invest a little more in sensing and especially in position sensing and obtain a decent solution for the localization and coverage problem?

The answer to this question is: things are not that simple. Lawnmowers operate outdoors and none of the solution developed for the floor-cleaners in Fig. 65.2 will work. Regular GPS (global positioning system) has an accuracy of several meters and has a tendency to deliver erratic readings, which would lead to equally erratic motions of the lawnmowers. As a matter of fact Automower 220 from Husquana uses GPS, but only as an antitheft protection device. Differential GPS, which would provide accuracy below one meter, would be too expensive for an affordable robotics lawnmower. In a nutshell, absolute positioning or SLAM, which would be necessary to cover a large outdoor area like a garden in a systematic fashion, is not practical.

In order to cover their workspace robotics lawnmowers use similar strategies to the Roomba-like floor cleaners. They refer to heuristics, which do not provide an optimal performance but still show a decent result. Using sensors such as gyroscopes, digital compasses, or inertial measurement units, robotic lawnmowers follow a certain heading and cover the workspace as much as possible by parallel tracks. They move along a straight line until they hit an obstacle or reach the border of the workspace. There they back up to become clear

from the obstacle, make a U-turn by 180° and drive back the way they came. Another heuristic that is frequently applied by robotic lawnmowers is the random motion shown also in Fig. 65.9. The mower moves along a straight line until it hits an obstacle or reaches the border of the workspace but then it does not just reverse but chooses a new direction randomly.

Since robotic lawnmowers move in open space, there is a danger that they leave their workspace and travel to areas where they are not supposed to be. For floor cleaners so-called virtual walls or fences solve this problem. Virtual walls are realized by infra red light beams emitted by some projectors, which can be placed in the workspace. The robots can sense these infrared light beams and consider them as obstacles, which evoke the typical obstacle avoidance behaviors. Lawnmowers use a similar technique, which is based on induction rather than light. To mark the border of the robot's workspace the owner has to place a wire around the area, which the robot must not leave (Fig. 65.9). This wire is connected to a low-voltage alternating current source. When the robot approaches the wire an inductivity sensor senses the current in the wire and causes the robot to reverse. Today most lawn mowers use a more sophisticated so-called *true in/out systems*, where the position is permanently tracked, not only when the robot approaches or passes the virtual fence [65.3].

The very first robotic lawnmowers had to be recharged manually. To avoid an all too frequent involvement of the human into the operation, Husquarna – one of the pioneers in robotic lawn mowing – equipped its first mower with solar panels and called it *SolarMower*. SolarMower was released in 1995, and was one of the first robotics lawnmowers. Nowadays, all lawnmowers come with a base station where they can recharge their batteries without human intervention. Also solar panels are coming back as power source, however only as auxiliary power to increase the performance and runtime between two charges and not as main power supply.

Given the fact, that robotic lawnmowers do have poor positioning capabilities, it is somewhat tricky to guide them back to the docking station once the battery



Fig. 65.10 Robotic lawnmower

gets low and needs to be recharged. Some of the lawnmowers shown in Fig. 65.10 use special wires which radiate from the position of the docking station. This way a robot only needs to follow such a radiating wire to return to the docking station on the fastest way. Another strategy is to follow the wire at the workspace border. That will eventually lead the robot back to the docking station.

An exception is the Bosch Indego, which is equipped with mapping and localization capabilities. This allows Indego to plan a path that leads to a point close to the docking station. Then the robot can follow a wire for the docking maneuver [65.1].

To comply with the safety regulations and to avoid any injuries of humans or animals the cutting mechanism has to be very lightweight and designed such that it is guaranteed to stop if the device tilts or is lifted. The cutting mechanism of robotic lawnmowers typically consists of a rotating disc with three razor-like blades, which automatically retract into their mountings if the robot is stopped unscheduled, for example, if it hits an obstacle or is lifted.

Naturally the lightweight design limits the thickness of the grass that can be cut. Also it is important for the proper functioning of robotic lawnmowers that the lawn, which is to be cut, is not too high. This in turn requires a rather regular if not continuous operation of the lawnmowers. With regular use of the lawnmowers the grass cuttings are short enough to quickly decompose into nutritious compost, so there is no need to remove the cuttings after the lawn is mowed.

Figure 65.10 shows some of the better-known robotic lawnmowers on the market today. Besides the pioneers in robotic lawn mowing Husqvarna, Friendly Robotics and Zucchetti also a number of new players have entered the market, most noticeably the German automotive supplier Bosch and the Japanese car manufacturer Honda, which has set a milestone in non-industrial robotics with its humanoid robot Asimo.

We certainly do not claim that the collection in Fig. 65.10 is complete. Like for domestic floor cleaning, Chinese B2B portals such as www.made-in-china.com or www.alibaba.com list around a hundred products under the category of robotic lawnmowers and 35 suppliers; the market development for robotic lawnmowers in the past five years was not quite as overwhelming as for domestic floor cleaners but was still remarkable. In Table 65.3 we show the technical specifications of a selection of lawnmowers from Fig. 65.10.

65.1.5 Sports Robotics

A subdomain of domestic robotics that was not included in the first edition of this handbook, because it virtually did not exist, or it was not visible at that time, is *sports robotics*. What is a sports robot? Since an official definition of this term does not yet exist – at least we haven't found one – we take the liberty and provide such a definition here. We define a sports robot as a *robotic device*, which either *supports* the human user in their physical exercises as a *coach* or a *companion*, or acts as an opponent in a game. An important aspect of this definition is the physical exercise of the human, which is supported or challenged.

We would like to emphasize that the above definition does not include any form of entertainment robots, which play games such as soccer against each other but do not involve any human activity other than watching.

In Fig. 65.11 we show three examples of *robotic baseball players*. As the name says the *Headless Batman* can act as a batter at least for exercising. It rather successfully hits baseballs thrown at it in many ways. The kinematic structure consists of *two arms, one leg and no head*, as its developer, an industrial designer from Robocross, likes to call it, and is made from auto parts, steel pipes, and pneumatic hoses. The pneumatic hoses are parts of its pneumatic actuation by an air

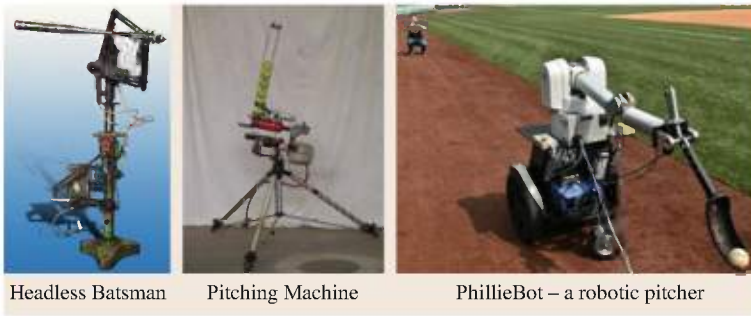


Fig. 65.11 Robotic baseball players

Table 65.3 Technical specification of a selection of robotic lawnmowers

					
Manufacturer	Husqvarna	Friendly Robotics	Zuchetti	Bosch	Honda
Model	Automower 220 AC	Robomow RL1000	Ambrogio	Indego	Miimo
CPU				ARM9 (32 bit) + PowerPC (32 bit)	
Drive	Differential	Differential	Differential 4 driving wheels	Differential	Differential
Sensors	Lift sensor collision/contact sensor	Rain sensor, lift sensor, collision/contact sensor	Grass sensors, safety sensors on handles, lift sensor, sensor for collision sensor and rollover	Lift sensor, collision/contact sensor, gyro, tilt sensor	3 × 360 degree bump sensors, 2 lift sensors
Navigation/coverage	Random	Random		Logicut navigation system: parallel tracks in parcels; localization, mapping, path planning	Random, parallel or mixed
Cutting technology	3 pivoting razor blades	3 razor blades	Rotating disc	3 pivoting blades	3 blades, which bend on hitting hard objects
Virtual fence and guides	Dual guide wires to return to the base station	Boundary wire	Boundary wire	Boundary wire	Boundary wire
Performance (m ²)	1800 (m ²)	2000 (m ²)	400 (m ²)	1000 (m ²)	3000 (m ²)
Ground speed			18 m/min	27 m/min	
Motor power (W)		3 × 150		2 × 20 + 80	2 × 25 + 56
Battery type and capacity	NiMH 30 W		Li-Ion, 6.9Ah	32 V Li-Ion, 3Ah	Li-Ion
Run time	45 min	2 h 30 min	3 h 30 min	50 min	
Docking/recharging station	Automatic docking and recharging		No	Automatic docking and recharging	Automatic docking and recharging
Theft protection	Pin code lock	Personal password		Alarm & Pin code	Pin code lock
Size (L×W×H)	71 × 55 × 30	87,5 × 65 × 31			
Weight (kg)	10	25	7,9	11.1	
Year of launch				2012	
Price range	~\$ 2400,00	\$ 1900,00		~\$ 1850,00	~\$ 2600,00

compressor. The *Headless Batsman* is fully controlled by a human operator via a remote control, which has

three buttons: one to control the robot's hip, a second one that actuates the arms and a third one to lift



Fig. 65.12 (a) JoggoBot and (b) RUFUS two robotic sports companions

and drop the inside shoulder, changing the trajectory of its swing. Similar mechanisms like *Headless Batsman* were developed at Hiroshima University and Tokyo University.

The counterpart of *Headless Batsman* is a pitching machine, which is operated with the same remote control. The barrel of the machine is sawn-off fire extinguisher whose other end is directly connected to a one-inch port poppet valve, which in turn is connected to an 8 bar air pressure tank. The air pressure tank is supplied from a screw compressor. The charging mechanism consists of a double rod actuator with a ring welded horizontally on to the end. Mounted above the ring is a magazine that holds up to 10 balls, which directly drop into the ring. For a pitch the actuator moves the ring with the ball over the edge of the barrel, where it falls on to a small guide. From there the balls rolls backward into the barrel. The actuated ring is covered by a steel strip, which holds back the other balls in the magazine until the actuated ring returns to its initial position. Since both mechanisms are remotely controlled, the human, who controls them, can actually sit in a chair next to the playground and let the robots play against each other. Such a use would clearly violate the above definition of a sports robot but does not seem to be the one that is primarily intended.

Another robot that can throw baseballs is Philliebot (VIDEO 748) developed in University of Pennsylvania's GRASP laboratory. Philliebot was developed in a couple of weeks using only spare parts in the GRASP lab. It uses a Segway as mobile base, a Barret arm, and pneumatically actuated wrist to create the necessary dynamics for the pitch. When the button is pressed the arm moves to the back of the robot and then accelerates its motion toward the target of its pitch. When the arm reaches the highest point of motion the pneumatic wrist cylinder delivers a burst of compressed carbon dioxide to snap the wrist forward and release the ball. What remains is the question of why use robotic equipment worth several tens of thousands of dollars to throw

a baseball, given that pitching machines have existed for many years. According to the developers, the fact that Philliebot is mobile and its software can be tweaked to vary pitch velocity and trajectory was enough to justify the experimentation.

The two sports robots shown in Fig. 65.12 do not really seduce their users to sit in a chair and relax. They both serve as so-called *robotic running coaches* or *running companions*.

Researchers at the Royal Melbourne Institute of Technology in Australia have redesigned a commercially available Parrot AR Drone quadcopter and turned it into an autonomous, flying running partner for joggers, called JoggoBot [65.4]. JoggoBot uses an integrated, front-mounted camera to detect and track a special patten printed on the T-shirt worn by the jogger. JoggoBot takes off when the camera registers the pattern and rises to about the same height as the pattern on the t-shirt. An internal sensor determines JoggoBot's altitude. JoggoBot can be set into a *companion mode*, in which it flies at a steady pace at a relative distance of about three meters to the jogger or in a *coach mode* in which it flies at a slightly more challenging speed.

There are two features of JoggoBot that make the device somewhat limited: first the capacity of the battery limits the flight time to 20 min, which in turns limits the time for exercise; for a short run, this is certainly ok, but for serious training this is insufficient. Second, JoggoBot can only fly in a straight line, to let JoggoBot follow an arbitrary path the jogger needs to remotely control JoggoBot's flight path.

A slightly different concept of a jogging companion is pursued in RUFUS (VIDEO 747), which is developed by runfun (www.runfun.com), a German startup company. RUFUS is an electrically driven, automatically guided, ground vehicle that supports and guides a runner during his/her training. RUFUS plays the role of a *personal running coach*. It fulfills a similar function to a treadmill, which exposes its user to a varying strain by varying its velocity and inclination and thereby improves the fitness, endurance, and resilience of the cardio-vascular system of the user. Unlike a treadmill, RUFUS is not a stationary device, however. It drives ahead of the runner like a pace maker in a marathon or a fake rabbit in a dog race, and sets the runner's speed. RUFUS's velocity is set either manually or automatically via a training program.

If operated in *manual mode* the velocity is either set directly as a velocity set point or indirectly as a heart-beat set point. If the training guidance is based on the heartbeat, then RUFUS controls its velocity such that the runner exercises optimally and continuously within a certain heartbeat interval under a moderate stain of the cardiovascular system.



Fig. 65.13 Tele-presence robots and robotic avatars

This has a twofold use: on the one hand this prevents users from overstressing themselves through overambitious and intensive training modules, possibly from even injuring him or herself. Such a protection function is beneficial for unfit or untrained runners. On the other hand RUFUS facilitates optimal training effect and progress through a careful guidance of the training.

The training effect can be further improved if RUFUS is operated in the *program mode* instead of the manual mode. In this mode RUFUS executes complete training modules, for example pyramid speed interval workouts, which are customized to the user. Such training modules are typically elaborated on by physiotherapists or sports physicians. They can be downloaded to the RUFUS embedded PC like an app from an app store.

RUFUS has a major advantage over JoggoBot. It has a battery capacity that allows it driving for about six hours on a flat road without recharging.

65.1.6 Tele-Presence

In a world in which not only large, but even small and medium-sized enterprises operate globally, in which families are scattered over continents, in which ubiquitous presence seems to become an essential requirement for professional progress, and in which professional services are more and more delivered over the internet,

tele-presence has become a fast growing market in the past years.

Robotics adds a very important aspect to plain *tele-vision* by turning it into *tele-presence*: embodiment and remotely controlled motion. As tele-presence is nothing but the combination of tele-vision and tele-operation using a robotic device, which are often called *tele-presence robots* or *robotic avatars*.

Tele-presence robots offer a whole spectrum of services and applications ranging from plain mobile video-conferencing systems to tele-surveillance, tele-diagnosing, and tele-care, to tele-teaching and tele-commuting. The term *tele-commuting* was coined by Scott Hassan, a Google developer of the early days, nowadays entrepreneur and investor, and founder of WillowGarage and Suitable Technologies, the manufacturer of Beam-RPD (see also Table 65.4).

A *tele-presence robot* typically consists of a mobile robotic platform that:

- Can be tele-operated through some user-interface,
- Carries a camera, which often can be actuated separately (via a pan-tilt unit) and allow the operator to actively explore the remote environment, and
- Carries a display, which allows those at the remote site to see the operator of the tele-presence system and communicate and interact with it.

Figure 65.13 shows a collection of such tele-presence robots. The devices range from a price of

Table 65.4 Technical specification of a selection of tele-presence robots

				
Manufacturer	Double Robotics	Vgo Communications	Gostai	Suitable Technologies
Model	Double	VGo	Jazz Connect	Beam RPD
Height	120–150 cm	120 cm	100.5 cm	5 feet, 2 inch
Weight	7 kg	9 kg	8 kg	45 kg
Screen size	9, 7'' (iPad)	6''	5''	17''
Camera/field of view			High resolution/ wide angle, up to 640 × 480 pixels at 25 fps	Two wide-angle HD cameras
Video conferencing	Open-tok	VGo video conf.	2-way audio and video for remote discussions	–
Network	WiFi	WiFi / 4G / LTE	WiFi	WiFi (two dual-band radios) / 4G
Remote Control	iPad App	VGo App	Intuitive control interface on internet browser	Beam software client, mouse, keyboard, or Xbox controller
Navigation	Tele-operated	Tele-operated	Tele-operated with obstacle detection	Tele-operated
Sensors	Gyroscope, accelerometer	Obstacle and cliff detection sensors	12 ultrasonic sensors, 4 IR receivers (for base docking), telemetric laser for autonomous navigation	–
Drive	Differential (10'' wheels)	Differential	Differential	Differential
Battery	Lithium ion	–	–	–
Run time (h)	8	12	5	8
Docking station	–	Yes auto-docking	Yes auto-docking	Yes

some US\$ 1500 for TeleMe from MantaroBot to more than ten-fold that amount for Beam RPD from Suitable Technologies. The system RP-VITA (Remote Presence Virtual + Independent Telemedicine Assistant) emerging from a cooperation between InTouch Health and iRobot is available only for lease, at a monthly fee of \$ 4700. Giraff is the result of a European research project lead by Giraff Technologies funded by the European Commission and is not commercially available yet.

Not all tele-presence systems shown in Fig. 65.13 can be classified as domestic robots. A system that clearly stands out and is by no means a domestic robot is *RP-VITA* by InTouch Health and iRobot. *RP-VITA* is a remote healthcare system. *RP-VITA* shall enable doctors to command any clinical, patient or care team management process remotely. *RP-VITA* has a full-fledged autonomous navigation system that allows the personnel to focus on the patient care task rather than on remote navigation. This feature has been awarded clearance by the US Food and Drug Association (FDA).

RP-VITA further provides access to important clinical data to support physicians, nurses and other care personnel in their diagnosis and other medical workflow. For example *RP-VITA* connects with diagnostic devices such as ultrasound and comes equipped with the latest electronic stethoscope. So *RP-VITA* is in a class of its own, which may also justify the higher price.

Apart from *RP-VITA*, the tele-presence robots shown in Fig. 65.13, can all be classified as semiprofessional or domestic service robots. The functionalities and services they offer do not necessarily vary on the same scale as their prices. This can be seen by a comparison of two of the above systems: Double and Beam RPD (see also Table 65.4).

Double is not much more than a mobile iPad equipped with the video-conferencing system Opentok. The mobile base uses a Segway-like dual-wheeled drive system that can balance a pole, which holds the iPad. When *Double* stands still, two retractable kickstands are deployed and allow the system to put the control system in an idle mode and save energy. *Double* can

be remotely controlled and driven around a remote site through an app installed on a second iPad that enables communication with all known Doubles over the web. The height of the iPad holder can be remotely adjusted to enable communication at eye-level. Double Robotics list a number of potential services and applications for which Double could be used: Companies with sites at various locations can use Double to improve communication and collaboration between remote teams. Families can use Double to communicate with family members living abroad. Museums and art galleries can use Doubles to offer remote tours through their exhibitions.

Beam RPD uses two HD cameras with custom wide-angle lenses instead of the plain iPad camera. This gives Beam RPD peripheral vision that is comparable to a human's field of view. A digital zoom lets the op-

erator further focus on details in the remote site. Beam uses an array of six microphones and audio processing algorithms, background noise reduction and echo cancellation. This equipment gives Beam an audio-quality, which obviously goes far beyond that of an iPad. Beam uses a 17-inch screen mounted at a height of 1.58 m that allows the display of a human face at its natural size and height. Another feature that goes beyond iPad standards is the WiFi connectivity. To provide reliable and seamless WiFi connectivity, Beam uses two dual-band radios and proprietary roaming algorithms. Altogether it is obvious that Beam RPD is far more than a movable iPad. It is left to the customers to decide whether this is worth a price which is an order of magnitude higher.  VIDEO 741,  VIDEO 742,  VIDEO 744, and  VIDEO 745 introduce several of the tele-presence robots available on the market today.

65.2 Enabling Technologies

The mass consumer market is very price-sensitive, so the price of the robot is key for the success of the product among consumers. Certain guidelines used in the consumer electronics market are relevant for the domestic robot market to provide a rough estimate of cost of the robot. Let's say that you want to develop a floor-care robot that would retail at \$ 300, the *rule of thumb* is that your bill of materials (BOM) should be between 1/3–1/5 of the retail price. In other words, your BOM should be within \$ 60–\$ 100! And the BOM must include all mechanical parts, electrical parts, battery, processor, memory, motors, assembly, packaging, user manuals, etc.

Given the extreme cost constraints outlined above, this chapter focuses on enabling technologies that are viable, from the cost point of view, to be included in a mobile domestic robot with a price lower than \$ 1000 (or ideally below \$ 500). These technologies are required to have a reliability level in line with the expected life time (and warranty) of the product; otherwise, no matter how good the technology is, if it stops working in an unreasonable period of time, the robot will be returned to the retailer. Special emphasis should be placed in the ease of manufacture of the technology. Difficult to manufacture components create delays in the production line, decreasing the yield of the product and eventually increasing the overall cost of production, leading to either eroding profit margins or a rise in the retail price.

Mobile robots need to sense and understand the environment in which they operate. The first key enabler is the capability of detecting obstacles and hazards

to safely and accurately navigate around them. Section 65.2.1 describes the different available technologies for obstacle and hazard detection. The second key enabler is the ability to localize and create a map of the environment to intelligently plan actions and motions that allow the robot achieve its goal. Section 65.2.2 presents the technologies available for localization and mapping using a number of low-cost, yet powerful sensors. Section 65.2.3 discusses alternative approaches to coverage of the space implemented in commercial products.

65.2.1 Sensing and Obstacle Avoidance

Domestic robots aim to take care of tedious chores, interacting with a household that includes owners, children, babies, pets, and stationary objects such as chairs, tables, walls, etc. Domestic robots must be safe in order to gain acceptance in our daily life: it is not tolerable to have a robot falling down the stairs or hurting a household member. Thus, robots must be equipped with drop/cliff sensors and proximity sensors that ensure proper operation while still satisfying the mentioned cost constraints.

Cliff Sensors

A number of solutions are present in current robots in the market. Off-the-shelf solutions are IR sensors from Sharp that consists of an emitter (light emitting diode LED) and a receiver (photodetector or position-sensitive-device PSD) that provides an output proportional to the distance to the object. The Roomba uses

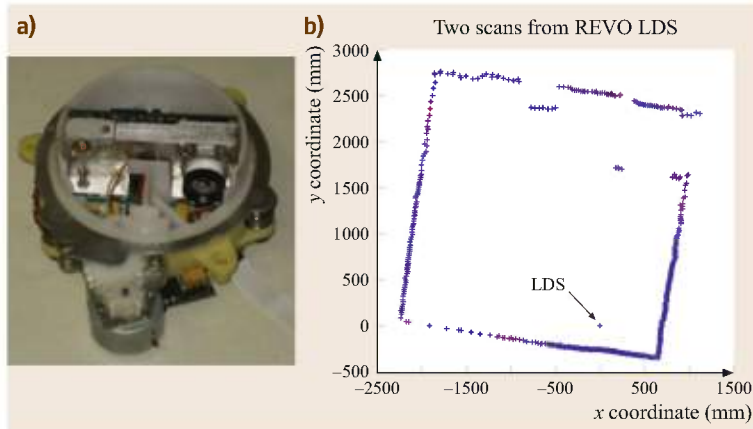


Fig.65.14a,b Laser Distance Sensor. (a) Prototype and (b) occupancy map generated with the sensor

a custom IR cliff sensor based on a similar principle that trades generic distance measurement with sensor cost. The Mint robot from Evolution Robotics employs a factory-calibrated mechanical hammer that triggers upon cliff detection. Solid-state sensors are usually more reliable than mechanical sensors, since they do not have moving parts, but have the drawback of a response dependent on the reflectivity of the surface in the IR spectrum and a dead-band in the response.

Contact and Proximity Sensors

Mechanical switches, called *bump sensors*, are commonly used for detecting when the robot gets in contact with obstacles. Bump sensors are cost-effective solutions providing the ability to stop the robot without damaging the obstacle. Touching obstacles is not desired unless performed very gently to ensure that the robot goes under curtains and bed skirts. IR and sonar sensors are frequently employed as touchless alternatives to the bump sensors by measuring the distance to obstacles. Both types of sensors are composed of an emitter and a receptor with an output proportional to the measured distance. IR sensors are usually more focused than sonars and less sensitive to multiple reflections on walls and other obstacles, but might lose thin obstacles such as chair legs. This type of sensors provides a point-wise measurement of distance to obstacles, so a robot needs a number of these sensors to obtain a dense representation of the obstacles in the environment. The information on obstacles and hazards (cliffs) is collected in occupancy grid maps and used for decision-making in systematic cleaning robots. A number of cost-effective dense distance measuring sensors have recently appeared in the market and will be discussed in the next sections. These dense distance measuring sensors have a cost on the order of tens of dollars, while the point distance measuring sensors have cost only a few dollars, so most of the domestic

robots currently available on the market have yet to incorporate dense distance measuring sensors. The only exception is the Neato XV-21 that uses a low-cost laser distance measuring system.

Laser Distance Sensor

A low-cost Laser Distance Sensor was developed by Konolige and colleagues [65.5] using a laser point beam and a global shutter CMOS imaging sensor separated by a small baseline. The system operates by triangulation and achieves full 360 planar scan by rotating the optical assembly on a full circle. The sensor has a range of 0.2 to 6 m with an error < 3 cm at 6 m and an angular resolution of 1 degree, providing 4000 readings per second (up to 10 Hz) with a small size (approximate width of 10 cm shown in Fig. 65.14a) and low power (< 2 W). The sensor is eye safe and provides measurements that enable laser-based SLAM as shown by Fig. 65.14b.

Structured-Light Distance Sensors

Structured-light distance sensors consist of an emitter that projects a known pattern on the environment and a receptor that computes depth based on the deformation of the received pattern. The Kinect [65.6] interface to the Xbox game system uses a structured-light sensor from PrimeSense [65.7] thus showcasing the readiness of this sensor for consumer applications. The emitter consists of a laser with optics that projects a known pattern (Speckles [65.8, 9]) in near-IR light and a complementary metal-oxide-semiconductor (CMOS) IR camera that observes the pattern to estimate depth using triangulation. The emitter and the camera are calibrated during manufacturing assuming a rigid configuration. The speckles can be further shaped into ellipsis using optics with different focal lengths in x and y so that the orientation of the observed ellipsis is proportional to depth. Speckles of different sizes are used to obtain different depth accuracy depending on size. Figure 65.15a

shows the speckles pattern projected by the sensor and an image taken in the dark by photographer Audrey Penven showing the IR speckles. Figure 65.15b shows the components of the Kinect sensor and the corresponding RGB and depth images of a scene.

Time of Flight (TOF) Distance Sensors

Time of flight (TOF) sensors consist of a light source (usually a laser) that emits a continuous waveform and a special imaging sensor that measures the phase shift of the received signal in each pixel. The depth at each pixel is proportional to the phase shift. TOF sensors have been in the market for quite some time, but the need for allocating a large portion of the sensor to the decoding electronics has made it challenging to produce low-cost sensors at reasonable resolution. Some of the companies offering TOF sensor have been Mesa Imaging AG that produces the SwissRanger [65.10] sensors, Softkinetics [65.11], PMDvision.

Stereo Vision

Stereo vision is a well-known computer vision solution to the extraction of three-dimensional (3-D) depth maps in areas with sufficient texture to find image correspondences. As opposed to structured light or TOF sensors, stereo vision systems are totally passive, but require a calibrated stereo rig, and their performance depends on the level of external illumination and on the amount of texture present in images.

The selection of the optimal dense mapping sensor depends on the application. The laser distance sensor provides reasonable information for laser-based SLAM and obstacle detection and avoidance; however, it only provides range information on a plane as opposed to the dense 3-D range offered by the structured light, TOF, or stereo systems. The structured light sensor uses a simple imaging sensor but requires additional computation to estimate depth in each pixel while the TOF sensor computes depth in each pixel at the expense of a sensor with lower fill factor. The stereo system does not require additional lighting, but requires an additional camera and a computation module to extract depth. Other parameters to consider are the maximum and minimum range that the sensor provides to ensure that it fits the requirements of the application in terms of mapping and obstacle detection.

65.2.2 Localization and Mapping

A robot that knows its location and understands its surroundings is able to plan intelligent maneuvers to achieve its goals. Localization and mapping are basic primitives that enable smart and efficient behavior. Early successful robots like the Roomba chose to sac-

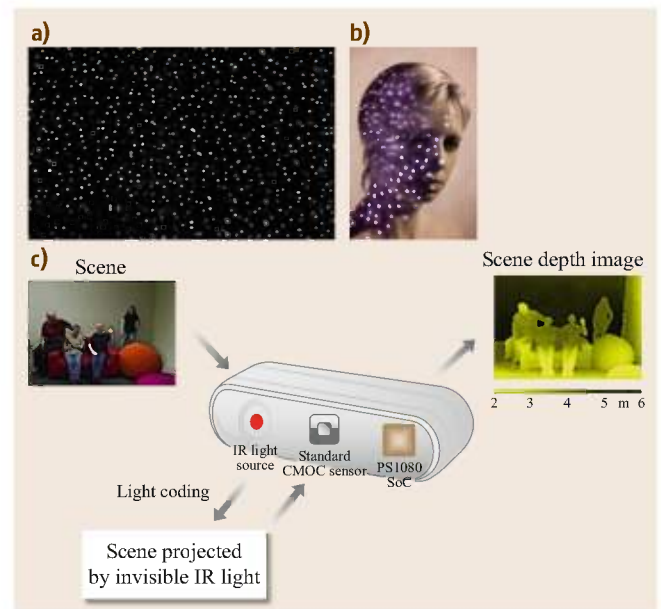


Fig.65.15a–c Primesense sensor: (a,b) Speckles pattern projected by the laser emitter. (c) Block diagram of the Kinect sensor

rifice localization and mapping in order to achieve an appealing retail price since the localization and mapping technology was either too expensive or not existent at the time (many of the dense sensors presented in the previous section were developed after the Roomba). In recent years, a number of low-cost, yet powerful simultaneous localization and mapping (SLAM) technologies have been developed and integrated in floor-care products. These technologies are described in the following sections.

Vector Field SLAM

The Mint robot by Evolution Robotics uses active beacons for localization. The Northstar beacon projects two IR spots onto the ceiling that are modulated to simplify the detection of the spots with the sensor. The spots are invisible to the human eye and thus do not produce visual clutter. While placing the beacon can still be regarded as a modification to the environment, surveys among our customers suggest that the vast majority largely accepts setting up the beacon prior to running the robot [65.12]. The Northstar sensor uses 3 photodiodes to compute the direction to the spots from the measured current through the photodiodes. The photodiodes can be sampled at high frequency to detect the modulated spot frequencies for data association. The sensor is quite inexpensive and suitable for cost-sensitive applications. However, the sensor suffers from multipath since light not only reaches the sensor directly but also through reflections from walls

and other furniture, making the position computed by the sensor inadequate to be directly used for localization and mapping. The sensor is augmented with a localization method that learns the light distribution in the room through a simultaneous localization and mapping (SLAM) approach [65.13]. Figure 65.16 shows the Mint robot cleaning the environment depicting the operation of the Northstar localization system and the problem of multipath. The figure also shows the Mint robot and the Northstar cube and the Northstar sensor.

In vector field SLAM the spatial variation of continuous signals is learned and simultaneously used for localizing the robot by fusing information from dead-reckoning (odometry and gyro) and Northstar. In the following, this method is introduced and tailored toward measurements obtained from Northstar. The signal field is represented as a regular grid of fixed node positions $b_i = (b_{i,x}, b_{i,y})^T$, $i = 1 \dots N$, where each node $m_i \in R^4$ holds the expected Northstar positions of both spots when placing the robot at b_i and pointing it in a fixed orientation $\theta_0 = 0$. Vector field and robot pose are then estimated through the application of SLAM.

Let the robot path be a time series of poses $x_0 \dots x_T$, $x_t \in SE(2)$, i.e., the set of rigid transformations in the horizontal plane, and let $x_0 = (0, 0, 0)^T$. At each time step $t = 1 \dots T$ the robot receives a motion input u_t

with covariance R_t and a measurement $z_t = (z_{x1}, z_{y1}, z_{x2}, z_{y2})^T$ of the two Northstar spot positions with covariance Q_t . The spot positions are also each affected by the rotational variability denoted $c = (c_x, c_y)^T$. The rotational variability models errors in measuring the direction to the spots caused by not having the sensor perfectly level.

The SLAM problem is solved with the ESEIF-SLAM that is constant time and requires memory linear in the size of the area explored by the robot. The method has been implemented in the processor of Mint, an ARM 7 processor with 64kByte RAM [65.13]. Vector field SLAM has been extended to address covering larger areas by using more Northstar beacons [65.15] and to provide a solution for the re-localization of the robot after it has been kidnapped or paused and resumed [65.16].

Figure 65.18 presents the map obtained by the robot after a cleaning run in a home environment of 125 m² covered by three Northstar beacons marked with the pink and green disks (units are in meters). The robot navigated in the home by following a cleaning strategy based on systematically covering sectors of the environment. As long as at least one beacon is visible to the robot, the strategy moves the robot onto a neighboring region until no space is left to clean. At the end the robot follows along the perimeter of detected obsta-

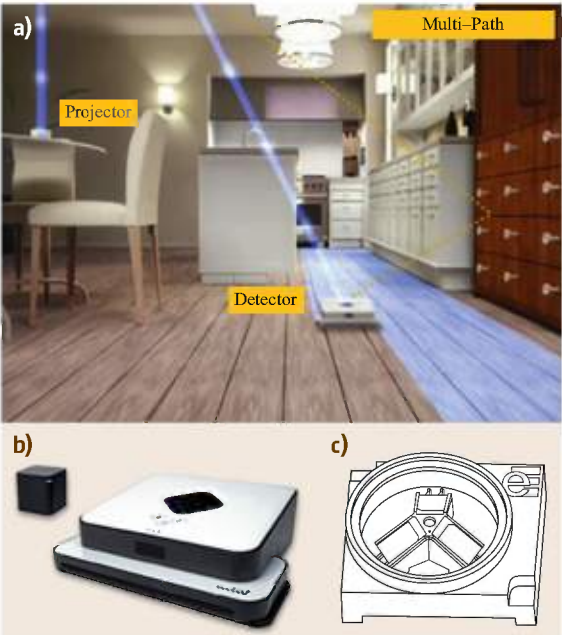


Fig.65.16a–c Mint robot. (a) Normal operation of the Mint robot using Northstar (the yellow path indicates multipath). (b) Mint robot and Northstar cube. (c) Northstar sensor

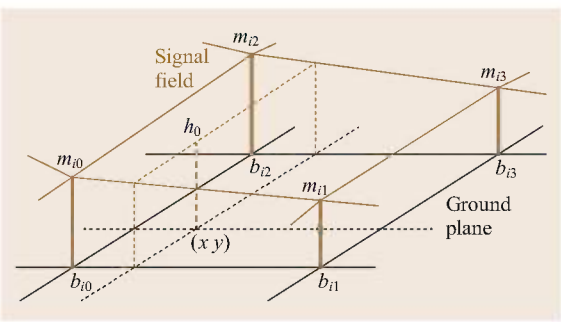


Fig. 65.17 Vector field SLAM with Northstar

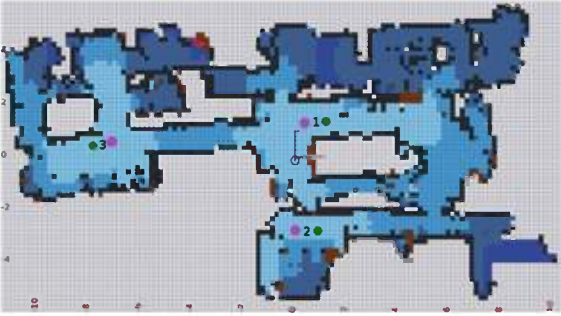


Fig. 65.18 Robot map obtained with vector field SLAM in a 125 m² home environment

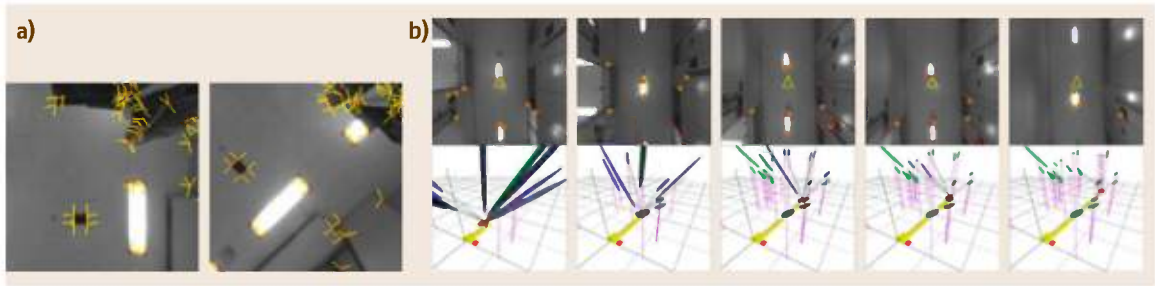


Fig. 65.19 (a) cv-SLAM (after [65.14]). (b) Features extracted by the vision front-end. Map obtained with the EKF

cles for a thorough cleaning around walls and furniture. As the robot moves through the environment it creates an occupancy grid map using the position information from localization. Each visited cell is classified into one of the following categories: obstacles (drawn in black), floor changes (drawn in brown), hazards (drawn in red), and free space (drawn in shades of blue that indicate visibility of the spots, the lighter the blue, the better the visibility).

Visual SLAM

Visual localization and mapping is attractive for a variety of applications due to the rich input and the low cost and footprint of visual sensors. The difficulties lie in robustly extracting a critical subset of information from the high-rate visual data stream and processing it efficiently to yield useful output. Despite steady increases in the computational power of most platforms, such challenges are nonetheless exacerbated by the limited processing and storage provided by low-cost, embedded systems appropriate for low-power applications or consumer products. Many state-of-the-art approaches to visual SLAM depend on a per-frame processing rate sufficiently high, relative to the speed of camera motion, to permit strong temporal assumptions on the image sequence. This section reviews two approaches to visual SLAM focused on domestic robots. Ceiling vision similar to the work of Jeong and Lee [65.14] can be found in the Navibot from Samsung, the Roboking from LG, and the Icebo/Homerun from Yujin Robotics/Philips. Furthermore, visual SLAM systems have been developed for low-cost and embedded systems [65.17, 18].

cv-SLAM

The ceiling vision SLAM [65.14] system is composed of a camera facing upward, looking toward the ceiling. The system extracts corner features using a Harris detector and matches features using correlation. Feature matching achieves invariance to view point changes by training a set of multiview descriptors of the features as matching proceeds during the run. The main orientation(s) of the feature are further used as descriptors

of the feature to ensure a two-dimensional (2-D) rotation invariant feature matching that can be used for relocalization. The localization and mapping backend of the system is based on an extended Kalman filter (EKF) that fuses visual and dead-reckoning information (odometry and gyro) and that tracks the pose of the robot in a two-and-a-half-dimensional (2.5-D) (x, y, θ) and the 3-D position of the features (landmarks of the map). The orientation(s) of the features are also represented in 3-D and tracked with the EKF to predict changes in the feature patch due to motion of the robot (as the robot moves, features on the ceiling will undergo a rotation while features on the walls will experience a shearing transformation in addition to rotation).

Figure 65.19a shows corner points and their estimated orientations in views with rotation only. Figure 65.19b presents snapshots from a sequential map-building experiment on a corridor.

vSLAM

The vSLAM [65.17, 18] system from Evolution Robotics is designed for a low-cost robotic platform equipped with simple odometry and a single camera. Figure 65.20 shows the block diagram of the system. Operating primarily as a view recognition engine, the visual measurement front-end requires only occasional, weak assumptions on processing rate, and intrinsically provides robust loop closing when previously mapped areas are revisited. The visual measurements and odometry are fused in the back-end in a graph representation and optimized incrementally. The SLAM graph complexity is bounded during operation using variable elimination and constraint pruning, with heuristic schedules in order to keep optimization and storage costs commensurate with explored area, rather than with time of exploration, all while causing minimal loss in mapping and localization accuracy. The system has been evaluated on real datasets with planar ground-truth reference, showing that the system operates successfully even at frame rates below 2 Hz, running on an ARM9 processor with 64 MB of RAM [65.18].

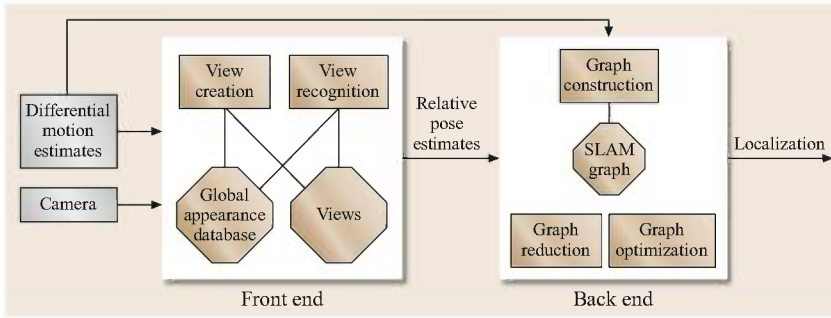


Fig. 65.20 vSLAM block diagram

View recognition engines [65.17–22] have proven attractive components for SLAM systems because they permit robust and flexible loop closing. Instead of making correspondences between individual features or measurements as in the cv-SLAM system, view recognition engines typically match constellations of features or entire images, without requiring tracking. vSLAM [65.17] creates views by first matching SIFT [65.23] features on pairs of incoming images, and second computing structure and motions estimates using bundle adjustment. The SIFT features are stored in a local database for each view and on a global database for the complete map. View recognition is performed by a feature lookup in the global database that provides a set of candidate view matches. Feature lookup in the local view database followed by robust outlier rejection and a local bundle adjustment completes the view recognition and visual pose estimation process.

Commonly used graph SLAM methods for agglomerating sensor information often incur computation and storage costs that grow with time, rather than with space explored. For a robot operating for extended periods within a limited spatial area – typical of practical applications – this is an undesirable trade-off. vSLAM [65.18] instead applies probabilistically sound graph reduction methods that limit the complexity of the graph to a linear factor of the complexity of the explored space. Past poses of the robot that are not used for view recognition can be marginalized out of the estimation, and their incident constraints are collapsed back into the graph.

Figure 65.21 presents the results of running the vSLAM system in two sequences, one collected in a regular household (*right*) (same house as the one shown in Fig. 65.18 for vector field SLAM) and the other collected in a large warehouse environment (*left*). The first set of plots show the ground truth path of

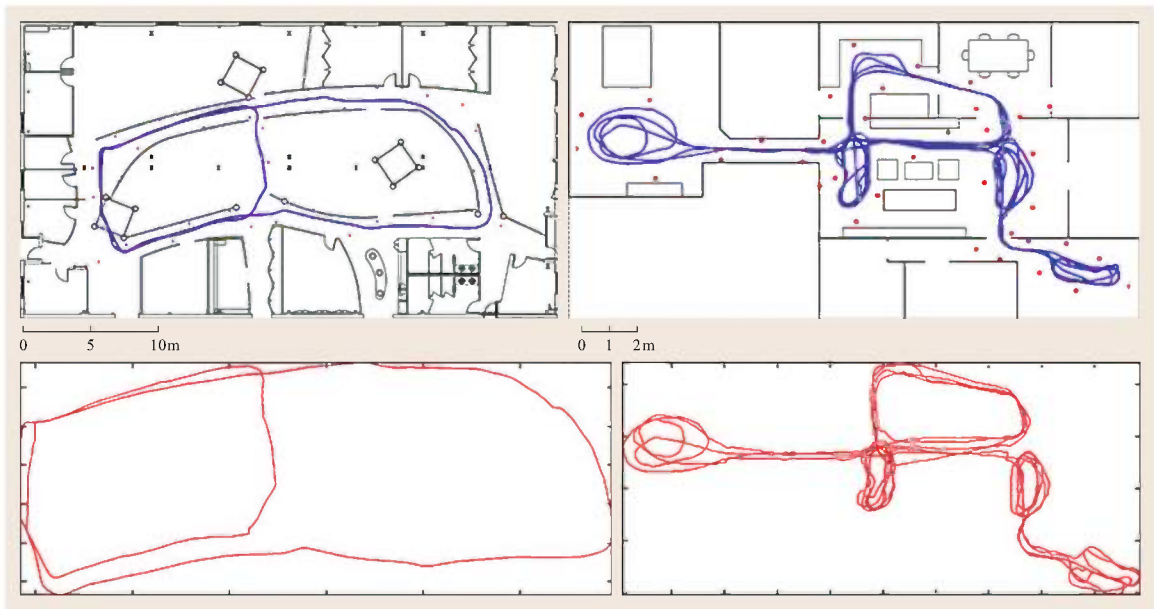


Fig. 65.21 vSLAM results on a warehouse environment and in a household environment



Fig. 65.22 Map created with the Neato XV-11 robot (www.youtube.com/watch?v=zodC8EFvh7g)

the robot and the floor plan of the environments. The second set of plots shows the trajectory of the robot estimated with vSLAM. The warehouse environment extends over an area of 24×12 m and the house over an area of 20×9 m. The estimated trajectory has a root mean square (RMS) error of 44 cm in the warehouse case and of 28 cm in the household case.

Laser-Based SLAM

The previous sections presented a number of low-cost solutions to the SLAM problem in which the data association was (almost) perfect. In the case of Vector-Field SLAM, the modulation of the spots of the Northstar cube ensures a unique identification of the spot. In the case of visual SLAM, the visual front-end incorporates a number of checks to ensure that misrecognitions are very rare. Laser-based SLAM has the characteristic that the measurements acquired with the laser are not unique since it is possible to obtain similar measurements in a variety of places in a household (just aiming the laser to a wall will provide measurements that are indistinguishable from a measurement to a different wall). In addition to the data association problem, laser distance measurement sensors, such as the one presented in Sect. 65.2.1, provide both range and bearing to the landmarks. Both Northstar and cameras give only bearing measurements to the spots or features.

The laser-based SLAM literature is quite extensive [65.25]. The estimation back-end could be an

extended Kalman filter (EKF), a Particle Filter, or a GraphSLAM system. Several data association algorithms are available, some of them are proactive (or even greedy) [65.26, 27] and some are lazy [65.28] in assigning correspondences between measurements and landmarks. The selection of the algorithm to implement in the domestic robot would be guided by the trade-off between computational resources and the performance requirements. Figure 65.22 presents a map obtained with the Neato XV-11 robot.

65.2.3 Navigation and Coverage

The Trilobite robot vacuum cleaner by Electrolux was one of the pioneer robots to be commercialized. The trilobite was equipped with a sophisticated custom sonar sensor system that allowed the robot to navigate without touching obstacles (or touching them very gently). The coverage strategy consisted of two stages: first, explore the perimeter of the room to estimate the area to clean and second, cover the room with wall-to-wall diagonal passes. The perimeter exploration stage assumes that the robot would eventually traverse through the complete boundary of the area to clean and return to the charging station (Fig. 65.23a).

As mentioned previously, the cost constraints forced early robots like the Roomba to eschew advanced features like localization and mapping in order to offer a product at a price point accepted by consumers. Nevertheless, the navigation strategy implemented in the Roomba was quite effective in covering the space, especially when running in single rooms or in presence of large amount of clutter. The Roomba uses a spiral pattern in open area to optimally cover the space without localization until it hits an obstacle (Fig. 65.23b). Then it selects random orientation and continues moving in straight line until reaching the next obstacle where it selects another random orientation to continue with the same behavior. The spiral can be triggered when the robot travels for a certain distance without having found any obstacles. Many of the other random robots present in the market have a strategy that follows similar principles to the Roomba one.

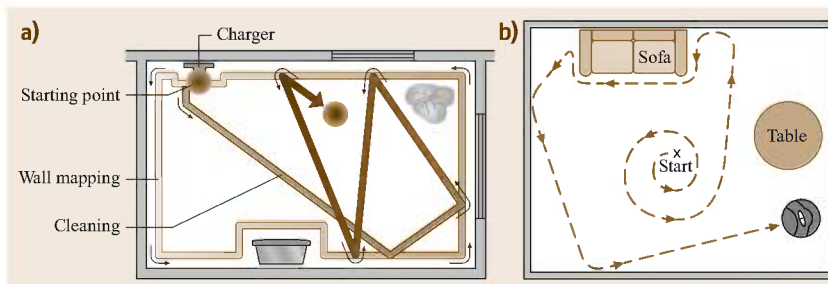


Fig. 65.23a,b Coverage strategies. Trilobite strategy. Roomba strategy [65.24]

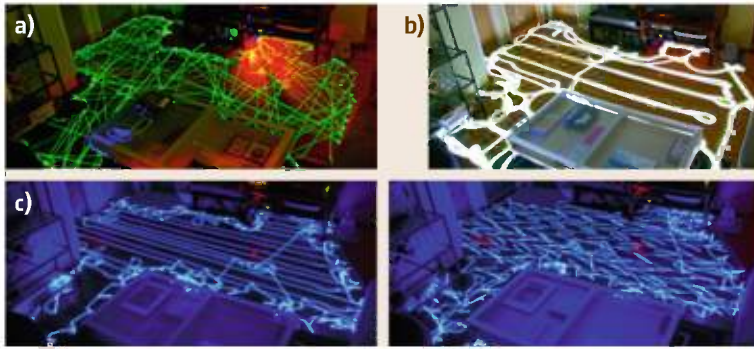


Fig.65.24a–c Long exposure images [65.29] showing the coverage strategies of the (a) Roomba, (b) Neato, and (c) Mint

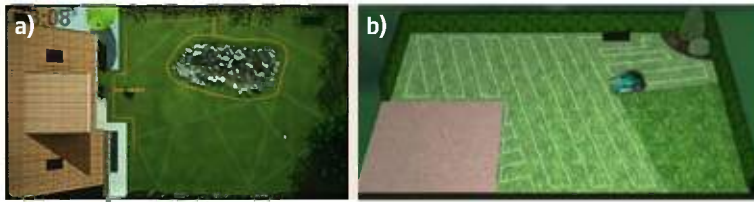


Fig.65.25a,b Coverage strategies for lawnmowers. (a) Tango [65.30] by John Deere and (b) Indego [65.31] by Bosch

The systematic robots on the other hand take advantage of the localization system to plan efficient cleaning paths to maximize coverage in the least amount of time, pause for charging on the docking station and resume cleaning from the last cleaned spot, and intelligently navigate from room to room. Mint takes the strategy of focusing first on open area cleaning, covering the space in parallel straight passes to traverse as much open space as fast as possible, and then performing a final cleaning on the perimeter and around the obstacles. This perimeter cleaning step enables Mint to uncover portals to new areas that were not encountered during open area cleaning. Other robots like the Navibot from Samsung, the Roboking from LG and the iClebo from Yujin/Philips use a similar strategy without the final perimeter cleaning. The Neato XV-11, on the other hand, performs first an exploration and cleaning of the perimeter of the environment in order to create a good localization map and then completes the coverage of the open area. Figure 65.24 shows long exposure pictures of different robots: Roomba, Neato, Mint sweeping and

Mint mopping. These long exposure pictures show the trajectory of the robots during normal cleaning operation.

The lawnmower robots use similar coverage strategies to the ones of vacuum cleaner robots. One element common to all lawnmower robots is the usage of an embedded wire to define the perimeter of the lawn. Within the boundaries defined by the wire, some robots like the Automower from Husqvarna or the Robomow from Friendly Robotics use a complete random coverage strategy. The Tango from John Deere combines straight motions in random directions with spiral motions (a-la-Roomba). The Indego from Bosch is the only robot that attempts to systematically cover the space by first exploring the perimeter to estimate the size of the lawn and then traversing the interior with parallel straight passes. Systematic coverage is enabled by fusing sensory information from wheel odometry and an inertial measurement unit (IMU) by means of probabilistic reasoning with prior knowledge. Figure 65.25 depicts the coverage strategies of the Tango and the Indego.

65.3 Smart Homes

Several attempts have been made in the literature to define the term smart home, for example, in [65.32] the term is defined as *the latest expression of the various ways in which technology in the home has developed*. In [65.33], the notion of a smart home is defined more explicitly as:

a residence equipped with computing and information technology which anticipates and responds to the needs of the occupants, working to promote their comfort, convenience, security and entertainment through management of technology within the home and connections to the world beyond.

Smart homes typically comprise elements such as network of sensors and actuators, and also entire robotic systems.

In 1999, a company called e2 Home was established by Ericsson and Electrolux to explore the possibilities of smart homes [65.34]. E2 Home built several houses with smart home devices using IT technologies. Through the exploration, several issues relating to smart homes from business point of view have been unveiled, which include the difficulties relating to the complexity of the system, starting up the new business model, handling of intellectual property rights (IPR) for the third parties contents, and the consumer's needs. The company was liquidated in 2004.

Current smart home technology includes video monitoring, motion detectors, fall detectors, pressure mats, environment control, health monitoring such as blood pressure, pulse rate, body temperature, weight, and human computer interaction (HCI) technology, for example, to recognize gestures. Smart homes also often refer to houses connected to a smart grid, which is defined as [65.35]:

a developing network of new technologies, equipment, and controls working together to respond immediately to our 21st century demand for electricity.

In this case, the smart home is defined as [65.35]:

a residence with the capability of efficiently controlling generated solar energy and power consumption making it ideal for vehicle power supply and management.

The development of smart homes requires a number of technical questions and challenges to be addressed [65.32]: how to convert current home structures and architectures into smart homes, how to standardize smart home components, for example, sensor networks, how to keep the equipment cost at a reasonable level, and how to deal with security and privacy issues.

In the following sections we will describe a number of prominent smart home developments, the Aware Home at Georgia Institute of Technology, the Gator Tech Smart House at the University of Florida, and the sensorized environment for life (SELF) at AIST (the Japan National Institute of Advanced Industrial Science and Technology) (Figs. 65.26, 65.27). These developments show that current smart home technology goes significantly beyond existing home automation. Accounting for the significant increase in the elderly population in the near future, many of today's smart home developments pay special attention to improving the quality of life of elderly people. The

descriptions below will also address the question on how to integrate robotics systems into smart home concepts.

65.3.1 Gator Tech Smart House

The Gator Tech Smart House (Fig. 65.26) was built at the University of Florida [65.36]. It addresses the needs of elderly people to live independently and maintain dignity and quality of life at older ages. The house is equipped with many smart devices such as smart floors, tracking the motion of the occupants of the house, smart blinds, automatically adjusting ambient light, smart display, smart cameras, smart phones that can act as a remote control to other appliances, location tracking, smart leak detectors, and smart beds. The exterior of the house has a smart mailbox which alerts residence if mail is delivered and a smart front door which can sense home owners using an radio frequency identification (RFID) tag, allowing keyless entry to home owners.

The kitchen of the Gator Tech Smart House includes a smart microwave, which uses RFID on food packages. This allows the microwave to adjust the settings for cooking the meal. It also informs the resident about the readiness of the meal. The kitchen further comprises a smart refrigerator that monitors food availability and consumption, and detects expired food items. The smart refrigerator can create shopping lists automatically and has an integrated meal preparation advisor based on items in the refrigerator and pantry.

The implementation of such a complex system has raised a number of technical issues and questions, including the development of the smart devices, data handling of networks of sensors, and interconnecting smart devices to other devices in the environment. These questions led to some new research tracks on smart houses, primarily grouped into pervasive computing and mobile computing network research.



Fig. 65.26 Georgia Institute of Technology: Aware Home

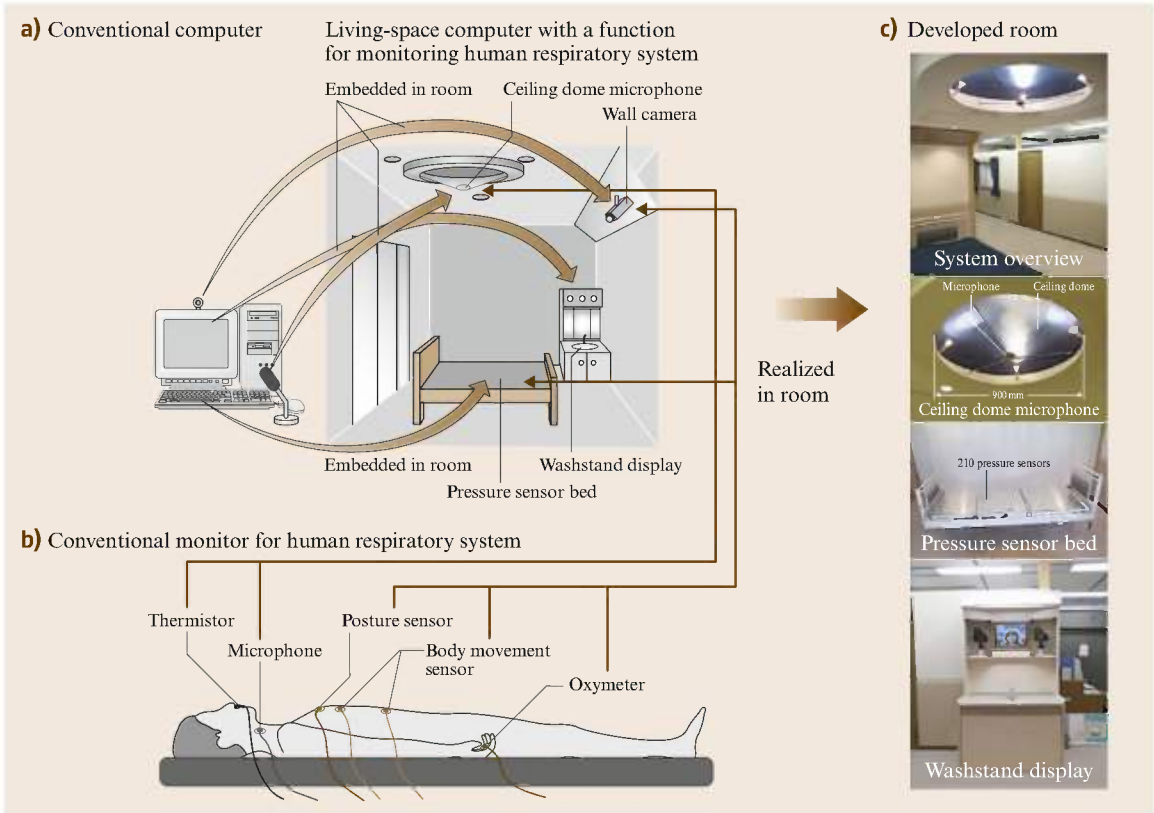


Fig. 65.27 AIST, the sensorized environment for life

65.3.2 Aware Home

Aware Home is a living laboratory for research in ubiquitous computing for everyday activities. This project is conducted at Georgia Institute of Technology [65.37]. The major objective of the Aware Home project is to build an environment that is capable of being aware and keeping track of the states and activities of its inhabitants. Aware Home creates a partnership between the resident and the surrounding sensing and computing technologies. This opens several fields of research, not only from the technology point of view, but also in terms of the social aspects of the inhabitants. The main research agenda of Aware Home spans human-centered and technology-centered research, software engineering, and social implications.

Technology and application-centered research focuses on sensor networks, distributed computing, context awareness and ubiquitous sensing, individual interaction with the home, smart floors, and finding lost objects. Research on context awareness is inspired by the fact that humans communicate with each other very success-fully by referring to what is called shared

context. For communication between humans and computer systems, this shared context must be made explicitly. Sensor systems, which facilitate the extraction of context, need to be developed.

This human-centered research focuses on support for the elderly and other social issues. A key concept in sup-orting the elderly is *aging in place*. Aware Home is designed to support the elderly and allow them to be independent instead of moving to elderly care facilities. Supporting the elderly leads to the study on cognitive support such as reminding them when to take medication, guiding them when they lose their way, and locating lost items.

65.3.3 SELF – Sensorized Environment for LiFe

SELF stands for Sensorized Environment for LiFe [65.1]. The objectives of SELF are to develop a network of sensors which are embedded in the environment, information gathering using the networked sensors, storing and analyzing the information, and the reporting of useful information to assist and maintain good health. The basic advantages

of SELF, due to the embedded nature of the sensors are:

1. No limit on size, weight, or power source,
2. It does not disturb the human,
3. It does not impose physical restrictions, and
4. Sensors are rarely broken since they are fixed to the environment.

SELF can be viewed as a system that monitors a person's behavior or activity and represents the data objectively in an approach known as self-externalization. SELF is motivated by the fact that humans sometimes cannot notice a change in their condition which affects their health without a medical doctor. Therefore, the use of network sensors to monitor human behavior and re-

port useful information that greatly affects the health status will further improve quality of life.

The SELF study considers behavior as a means of communication, and sensors embedded into the environment as one way to observe a person's behavior. The SELF implementation consists of a bed with a sensor, a ceiling with microphones, a washstand with a display, etc. The bed with sensors can determine the time the subject sleeps and wakes up, their posture during sleeping, and their breathing pattern. The microphones attached to the ceiling can detect snoring or normal breathing sounds. Based on the monitored data, the washstand display is used as an output device to provide the subject's health status and thus create feedback to the subject.

Video-References

-  VIDEO 727 Robotic Vacuum Cleaners Reviewed by Click – Spring 2014
available from <http://handbookofrobotics.org/view-chapter/65/videodetails/727>
-  VIDEO 729 How would you choose the best Robotic Vacuum Cleaner?
available from <http://handbookofrobotics.org/view-chapter/65/videodetails/729>
-  VIDEO 731 Husqvarna Automower versus competitors
available from <http://handbookofrobotics.org/view-chapter/65/videodetails/731>
-  VIDEO 734 Windoro window cleaning robot review
available from <http://handbookofrobotics.org/view-chapter/65/videodetails/734>
-  VIDEO 735 WINBOT W710 versus HOBOT 168
available from <http://handbookofrobotics.org/view-chapter/65/videodetails/735>
-  VIDEO 736 Winbot window cleaning robot
available from <http://handbookofrobotics.org/view-chapter/65/videodetails/736>
-  VIDEO 737 Serbot Robot Clean Ant Profi
available from <http://handbookofrobotics.org/view-chapter/65/videodetails/737>
-  VIDEO 739 Home Pool Cleaner Review – 5 types of robotic cleaners
available from <http://handbookofrobotics.org/view-chapter/65/videodetails/739>
-  VIDEO 740 Automatic pool cleaner reviews
available from <http://handbookofrobotics.org/view-chapter/65/videodetails/740>
-  VIDEO 741 Telepresence robot in action
available from <http://handbookofrobotics.org/view-chapter/65/videodetails/741>
-  VIDEO 742 Double robotics – overview
available from <http://handbookofrobotics.org/view-chapter/65/videodetails/742>
-  VIDEO 744 Test-Driving Beam, the telepresence robot
available from <http://handbookofrobotics.org/view-chapter/65/videodetails/744>
-  VIDEO 745 Beam's new Palo Alto store lets telepresence robots sell themselves. Literally
available from <http://handbookofrobotics.org/view-chapter/65/videodetails/745>
-  VIDEO 746 This robot is your running coach – Joggobot
available from <http://handbookofrobotics.org/view-chapter/65/videodetails/746>
-  VIDEO 747 RUFUS – your personal running coach
available from <http://handbookofrobotics.org/view-chapter/65/videodetails/747>
-  VIDEO 748 PhillieBot robot gives first pitch at a Phillies game
available from <http://handbookofrobotics.org/view-chapter/65/videodetails/748>

References

- | | |
|---|---|
| <p>65.1 P. Ljunggren: Intelligent machines, Personal Communication</p> <p>65.2 R.D. Schraft, M. Hägele, K. Wegener: <i>Service-Roboter-Visionen</i> (Hanser, Munich 2004)</p> | <p>65.3 A. Albert: Robert-Bosch GmbH, Personal Communication</p> <p>65.4 Howstuffworks: http://electronics.howstuffworks.com/gadgets/home/robotic-vacuum2.htm</p> |
|---|---|

- 65.5 K. Konolige, J. Augenbraun, N. Donaldson, C. Fiebig, P. Shah: A low-cost laser distance sensor, Proc. Int. Conf. Robotics Autom. (ICRA), Pasadena (2008)
- 65.6 Gearfuse: <http://www.gearfuse.com/robotic-vacuum-paths-mapped-and-compared-with-long-exposure-pictures/>
- 65.7 P. Ridden: Joggobot turns a quadrocopter into a running companion, Gizmag, June 11 <http://www.gizmag.com/joggobot-autonomous-quadrocopter-running-partner/22899/> (2012)
- 65.8 B. Freedman, A. Shpunt, J. Arieli: Distance-varying illumination and imaging techniques for depth mapping, US Patent 29 0698 (2010)
- 65.9 A. Shpunt, A. Zlesky: Depth-varying light fields for three dimensional sensing, US Patent 2008/10 6746 (2008)
- 65.10 Microsoft (Xbox): <http://www.xbox.com/en-US/KINECT>
- 65.11 Apple (Primesense): <http://www.primesense.com/>
- 65.12 J.-S. Gutmann, K. Culp, M. Munich, P. Pirjanian: The social impact of a systematic floor cleaner, ARSO Munich (2012)
- 65.13 J.-S. Gutmann, E. Eade, P. Fong, M. Munich: Vector field SLAM – localization by learning the spatial variation of continuous signals, Trans. Robotics **28**(3), 650–667 (2012)
- 65.14 W. Jeong, K. Lee: CV-SLAM: A new ceiling vision-based SLAM technique, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2005)
- 65.15 J.-S. Gutmann, D. Goel, M. Munich: Scaling vector field SLAM to large environments, Proc. IAS, Jeju (2012)
- 65.16 J.-S. Gutmann, P. Fong, M. Munich: Localization in a vector field map, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2012)
- 65.17 N. Karlsson, E.D. Bernardo, J. Ostrowski, L. Gonçalves, P. Pirjanian, M.E. Munich: The vSLAM algorithm for robust localization and mapping, Proc. Int. Conf. Robotics Autom. (ICRA) (2005)
- 65.18 E. Eade, P. Fong, M. Munich: Monocular graph SLAM with complexity reduction, IROS (2010)
- 65.19 M. Cummins, P. Newman: Accelerated appearance-only SLAM, Proc. Int. Conf. Robotics Autom. (ICRA), Pasadena (2008)
- 65.20 E. Eade, T. Drummond: Monocular slam as a graph of coalesced observations, Proc. 11th IEEE Int. Conf. Comput. Vis. (ICCV'07), Rio de Janeiro (2007)
- 65.21 E. Eade, T. Drummond: Unified loop closing and recovery for real time monocular slam, Proc. Br. Mach. Vis. Conf. (BMVC'08) Leeds (2008) pp. 53–62
- 65.22 K. Konolige, J. Bowman, J.D. Chen, P. Mihelich, M. Calonder, V. Lepetit, P. Fua: View-based maps, Proc. Robotics Sci. Syst. Seattle (2009)
- 65.23 D.G. Lowe: Distinctive image features from scale-invariant keypoints, Proceedings IJCV (2004)
- 65.24 Pmdtechnologies GmbH: <http://www.pmdtec.com/>
- 65.25 S. Thrun, W. Burgard, D. Fox: *Probabilistic Robotics* (MIT Press, Cambridge 2005)
- 65.26 J. Neira, J. D. Tardos: Data association in stochastic mapping using the joint compatibility test, IEEE Trans. Robotics Autom. **17**(6), 890–897 (2001)
- 65.27 S. Thrun, D. Fox, W. Burgard: A probabilistic approach to concurrent mapping and localization for mobile robots, Mach. Learn. **31**, 29–53 (1998)
- 65.28 D. Hähnel, S. Thrun, B. Wegbreit, W. Burgard: Towards lazy data association in SLAM, Proc. 11th Int. Symp. Robotics Res. (ISRR), Sienna (2003) pp. 421–431
- 65.29 John Deere: http://www.deere.com/wps/dcom/en_INT/products/equipment/autonomous_mower/autonomous_mower.page
- 65.30 Robert Bosch GmbH: <http://www.bosch-indego.com/>
- 65.31 E. Prassler, K. Kosuge: Domestic robotics. In: *Springer Handbook of Robotics*, ed. by B. Siciliano, O. Khatib (Springer, Berlin, Heidelberg 2008) pp. 1253–1281
- 65.32 D. Gann, J. Barlow, T. Venables: *Digital Futures: Making Homes Smarter. Report published by Chartered Institute of Housing* (Chartered Institute of Housing, Coventry 1999)
- 65.33 R. Harper: *Inside the Smart Home* (Springer, London 2003)
- 65.34 Japan Electronics and Information Technology Industries Association: <http://www.eclipse-jp.com/jeita/>
- 65.35 Gator Tech Smart House: <http://www.icta.ufl.edu/gt.htm>
- 65.36 Aware Home Research Initiative: <http://www.awarehome.gatech.edu/>
- 65.37 T. Hori, Y. Nishida, T. Suehiro, S. Hirai: SELF-Network: Design and implementation of network for distributed embedded sensors, IEEE/RSJ Int. Conf. Intell. Robots Syst. IROS (2000)

Attachment N

[Home](#) > [Springer Handbook of Robotics](#) > Chapter

Domestic Robotics

| Chapter | First Online: 01 January 2016

| pp 1729–1758 | [Cite this chapter](#)

✓ Access provided by University of Nebraska–Lincoln

Download book PDF

Download book EPUB 



[Springer Handbook of Robotics](#)

[Erwin Prassler](#) , [Mario E. Munich](#), [Paolo Pirjanian](#) & [Kazuhiro Kosuge](#)

 Part of the book series: [Springer Handbooks](#) ((SHB))

 100k Accesses  19 Citations

Abstract

When the first edition of this book was published domestic robots were spoken of as a dream that was slowly becoming reality. At that time, in 2008, we looked back on more than twenty years of research and development in domestic robotics, especially in cleaning robotics.

Although everybody expected cleaning to be the killer app for domestic robotics in the first half of these twenty years nothing big really happened. About ten years before the first edition of this book appeared, all of a sudden things started moving. Several small, but also some larger enterprises announced that they would soon launch domestic cleaning robots. The robotics community was anxiously awaiting these first cleaning robots and so were consumers. The big burst, however, was yet to come. The price tag of those cleaning robots was far beyond what people were willing to pay for a vacuum cleaner. It took another four years until, in 2002, a small and inexpensive device, which was not even called a cleaning robot, brought the first breakthrough: *Roomba*. Sales of the Roomba quickly passed the first million robots and increased rapidly. While for the first years after Roomba's release, the big players remained on the sidelines, possibly to revise their own designs and, in particular their business models and price tags, some other small players followed quickly and came out with their own products. We reported about these devices and their creators in the first edition. Since then the momentum in the field of domestic robotics has steadily increased. Nowadays most big appliance manufacturers have domestic cleaning robots in their portfolio. We are not only seeing more and more domestic cleaning robots and lawn mowers on the market, but we are also seeing new types of domestic robots, window cleaners, plant watering robots, tele-presence robots, domestic surveillance robots, and robotic sports devices. Some of these new types of domestic robots are still prototypes or concept studies. Others have already crossed the threshold to becoming commercial products.

For the second edition of this chapter, we have decided to not only enumerate the devices that have emerged and survived in the past five years, but also to take a look back at how it all began, contrasting this retrospection with the burst of progress in the past five years in domestic cleaning robotics. We will not describe and discuss in detail every single cleaning robot that has seen the light of the day, but select those that are representative for the evolution of the technology as well as the market. We will also reserve some space for new types of mobile domestic robots, which will be the success stories or failures for the next edition of this chapter. Further we will look into nonmobile domestic robots, also called smart appliances, and examine their fate. Last but not least, we will look at the recent developments in the area of intelligent homes that surround and, at times, also control the mobile domestic robots and smart appliances described in the preceding sections.

Multimedia Contents



 Access provided by University of Nebraska–Lincoln. [Download chapter PDF](#) 

Similar content being viewed by others



Onboard Robust Person Detection and Tracking for Domestic Service Robots

Chapter | © 2016



How User Presence Impacts Perceptions and Operation Routines of Robotic Vacuum Clean...

Chapter | © 2022



Application of Human-Robot Interaction Features to Design and Purchase Processes of...

Chapter | © 2021

Explore related subjects

Discover the latest articles, books and news in related subjects, suggested using machine learning.

[Bioinspired Robotics](#)

[History of Technology](#)

[Rehabilitation Robotics](#)

65.1 Mobile Domestic Robotics

The first mention of a domestic cleaning robot dates back to 1985. The device, nicknamed *Robby*, was developed by Hitachi starting in 1983 and officially carried the name HCR-00 (Fig. 65.1). Robby was equipped with a gyroscope to keep track of its position and a rotating sonar scanner for obstacle detection.

Fig. 65.1



First generation of domestic floor cleaning robots (1985–1999)

65.1.1 Domestic Floor Cleaning

Very similar to today's cleaning robots described further below, Robby already mapped its environment and used the map for path planning. HRC-00 remained a prototype like its successors HRC-01 to HRC-03.

65.1.1.1 First Generation of Domestic Cleaning Robotics (1985–1999)

Also in 1985, the Swedish appliance manufacturer Electrolux started the development of a concept vacuum cleaner *Stardust*. The device was equipped with eight fixed sonar sensors and one rotating sensors for obstacle detection. To maintain its orientation Stardust used an infrared sensor that tracked an infrared light bulb mounted to the ceiling. In 1988 Stardust was presented to the public at Domotechnica in Cologne, Germany, one of the largest fairs worldwide for domestic appliances [[65.1](#)].

About five years later, between 1989 and 1991, Panasonic undertook an effort to develop a domestic cleaning robot that led to two prototypes, one of them, *Brownie*, is shown in Fig. [65.1](#). Brownie was a battery-powered vacuum cleaner with a diameter of 40 cm, and at 18 kg, a super heavy weight compared to today's domestic cleaning robots. It was equipped with a gyroscope, a ring of sonar sensors, and a dust sensor. Likely, it was limitations in battery technology for driving such a heavy robot, as well as the total cost, which prevented Brownie from become anything more than a prototype.

Six more years went by before Minolta presented its *Minolta Cleaning Robot* and the first European player, Electrolux, its Robot Vacuum Cleaner, later called Trilobite, which was presented to the public in BBC's TV show *Tomorrows World* in 1996.

The devices had already become significantly smaller and lighter in these six years. The Minolta robot had dimensions of 321 mm × 320 mm × 170 mm and weighted about 8 kg. It was powered by nickel metal hybrid battery and used sonar and tactile sensors for obstacle detection, a cliff-sensor to discover staircases, and a gyroscope to track its position and orientation. Electrolux' robot also had a sophisticated sonar system, which let it follow contours and even return to a homing position. It already had the size and shape of later floor cleaning robots.

Five more domestic cleaning robots made their first appearance during this time, which we call the infancy of domestic cleaning robotics. Two of them only reached the proof of concept level: *AutoCleaner* from InMach Intelligent Machines – the German startup later developed low-cost navigation systems for the professional cleaning robot Robo40, which became commercially available in 2007 – and Koala form EPFL-Lami. AutoCleaner was the first domestic wet-cleaner proof of concept, which cleaned the floor using a rotating microfiber

towel that was pulled through a water tank. Koala used a suction spout to reach corners. Two more robots reached at least the status of industrial prototypes, but were never commercialized: Dyson's DC06 and Cye from *Probotics*. DC06 was a unique domestic cleaning robot. It differed not only in terms of the announced list price of approximately US\$ 4000, but also in many other aspects: Dyson claimed that it had three onboard

CPU

s, more than 50 sensors for obstacle avoidance, cliff-detection, localization and more. On top of that came the unique Dual Cyclone cleaning technology. The DC06 never made it to the market. Neither did Cye. Cye was a small mobile robot that could pull and push a (semi-)regular vacuum cleaner. It was announced as the *first personal robot* which could not only vacuum the floor but also serve coffee or deliver mail. It is unclear if Cye ever delivered coffee or mail or vacuumed a real living room. It disappeared as quickly as it appeared. The last out of the nine first robots that we call first generation was the Kärcher RoboCleaner. Its successor RC3000 was amongst the first cleaning robots that had a docking station at which the robot could not only recharge its battery but also unload the dust, which it had collected. We will come back to the Kärcher RC3000 in the next section.

The first generation domestic floor cleaning robots shown in Fig. [65.1](#) were the ones presented to the public. However, they were by far not the only ones that were developed in the period between 1985 and 1999. By browsing over the patents in the field of cleaning robots during that period it becomes obvious that many of the big appliance and electronic device manufacturers worldwide performed research and development on domestic cleaning robots. Besides the ones mentioned above, one can find such big names as Nintendo, Matsushita, Sanyo, Samsung, Honda, Procter & Gamble, Electrolux, Philips, Henkel and more. So, although only a few players dared to go public, domestic cleaning was already on the radar of many international enterprises. What kept them away from taking the next steps is not hard to guess. It was the risk of getting into a rather conservative and tight market – the cleaning business is extremely conservative – with a semimature technology carrying a price tag that looked quite differently than the one on well-established technology, such as traditional vacuum cleaners.

65.1.1.2 The Second Generation of Domestic Cleaning Robotics (2000–2008)

Figure [65.2](#) shows the second generation of domestic cleaning robots. What made the difference between the first and the second generation? The amount of experimentation regarding the overall design decreased, with the majority of manufacturers settling on the disc shape. The number of sensors decreased, as well as their level of sophistication. Sophisticated and expensive sensors such as sonar largely disappeared. Most designs were limited to very few, simple, and inexpensive sensors such as bumpers, simple one-dimensional range sensors, typically based on infrared (

IR

), and cliff sensors. This was most likely a rather painful lesson learned from the designs in the first generation. What roboticists considered the state of the art in mobile robotics: environmental sensing, map building, range sensors with high quality angular and range resolution, and combined localization and mapping, were far too expensive to be built into a domestic cleaning robot. Those robots had to compete with regular vacuum cleaners in a price range of a few hundred dollars. This competition imposed very harsh cost limits for the robotic components that could be built into a domestic cleaning robot: 50 to 100 USD and often less. These cost constraints had a significant impact on a very fundamental expectation of what cleaning robots should achieve: systematic coverage of the area to be cleaned. Cleaning was understood to be synonymous with covering an area with a systematic and intuitive motion while applying some type of cleaning operation. Most of the patents filed for the first generation of domestic cleaning robots mentioned above described cleaning robots that were supposed to cover their workspace in a systematic fashion.

Fig. 65.2



Second generation of domestic floor cleaning robots (2000–2008)

Systematic coverage, however, is impossible without absolute positioning and without decent knowledge of the environment. The second generation of domestic cleaning robots had to provide a solution for this dilemma of systematic coverage with cheap and simple sensors, which must not cost more than several tens of dollars.

The solution to this dilemma was already proposed in some early prototypes. The idea was to waive the requirement of systematic coverage, which would also involve intuitive motion patterns, and instead produce *semisystematic* coverage and *semi-intuitive* motion patterns. This semisystematic coverage was achieved by a combination of random motion, hard-coded motion patterns such as spiral- or meander-shaped motions, which – considered alone – reflects some systematic coverage, and some other systematic and intuitive motion patterns, for example, following contours of walls or other objects in the workspace. The theory of stochastic process states that the mean squared distance of a particle that performs a random walk with respect to the origin of its motion increases proportionally with time. That implies that the particle moving in confined space will cover that space in a finite amount of time.

First-generation domestic cleaning robots like the Electrolux Robotics Vacuum Cleaner (later Trilobite) or the Kärcher Robot Cleaner (later Kärcher RC 3000) already used random motion combined with hard-coded motion patterns to achieve a certain degree of coverage. But apparently the insight that the attribute *robotic* in front of *vacuum cleaner* was not

enough of a sales argument to justify a price three to five times higher than that of a regular vacuum cleaner was not so easy to digest. Some of the second-generation cleaning robots were offered for a price on the order of 1500 USD or even more. In the end these robots shared the same fate: they become shelf warmers.

This is also due to the fact that in 2002 a robot was launched which was to herald a breakthrough in domestic cleaning robotics: Roomba. Its creator, iRobot Inc., had learned the lesson that others were still struggling with: if you want to sell a domestic appliance, better sell it for a price that is known for domestic appliances. For the sake of fairness one has to mention that the traditional vacuum cleaner manufacturers had to design the robots so they matched the quality and performance level that could be expected from the brand name. They could not take the risk of making their product too poor. Therefore it easily became over specified compared to what the market required at the time. New manufacturers such as iRobot in contrast had no brand name to defend [65.1].

If it is a totally new device, which may not only cause excitement but also concerns and reservations, sell it for less and not for more. When Roomba entered the market it was sold for 199 USD. That was a price, which did not cause customers to think about whether they really needed it, whether the quality was good enough, or whether the device would get into every corner of a room or underneath every couch or bed. The creators of Roomba were also smart enough not to call it a robot. This that prevented many customers from developing wrong expectation of what robots could or should do.

The robot technology that was built into the Roomba was everything but new or revolutionary. Roomba used a suspended front-shield for contact-sensing, a low-cost infrared range-sensor for contour-following, a cliff-sensor to prevent it from falling down stair-cases, and it could detect photoelectric barriers, so-called *virtual walls*, which kept it from leaving a room or a certain part of its workspace. Roomba further used the combination of random motion and hardcoded area-covering motion patterns to achieve a certain level of coverage. All this technology was known before Roomba.

Still Roomba can be seen as a milestone not only in domestic robotics but also in robotics at large. Why is that? It was the first time in robotics history that robotics was no longer

synonymous with high-tech, high-price. Roomba showed that automation of everyday service tasks, such as domestic cleaning, could be achieved with a moderate effort in terms of hardware and at a decent price, given that one accepts some graceful degradation in the overall performance. There is no such thing as a free lunch, not even in robotics and this degradation in the performance is the price to be paid. What turned Roomba into a milestone was the cost-effective design, where the limitations of low-cost hardware – in Roomba’s case the sensors – were balanced out by smart heuristics for problem-solving. This design of a commercially viable product, at a price point competitive with nonrobotized solutions, turned Roomba into the first successful domestic cleaning robot, and the most frequently sold robot in the past 50 years.

In Table [65.1](#) we give an overview of the technical specifications of some of these second generation robots. A more complete overview is included in 065-bib43

Table 65.1 Technical specification of a selection of second-generation domestic cleaning robots

The idea to develop and commercialize a domestic cleaning robot at a price level of 199 USD, comparable to the price of traditional domestic appliances, was very appealing to consumers. However, it was not an overly profitable business model, even if Roomba was manufactured in a low-wage country at a cost of significantly less than 100 USD. Extreme cost pressure, however, often compromises the quality of a product. So a decision to adhere to this low price of below 200 USD may also have been a decision against the quality of the product. Today nearly a dozen models of Roomba are on the market, which differ mostly in terms of their cleaning technology and extra features. Their prices are in a range from 250 to 900 USD.

65.1.1.3 The Third Generation of Domestic Cleaning Robotics (2009–2012)

The second generation of domestic cleaning robots also provided their developers and distributors with a number of painful and partly contradictory insights:

- A domestic cleaning robot sold at a price that is significantly beyond the price of comparable nonrobotic devices, runs a high risk of failing, because many customers may not be willing to pay extra money for a cleaning robot just because it is a robot.
- A domestic cleaning robot sold at the same price level as an inexpensive comparable appliance, runs a high risk of failing, because the low price may compromise the quality and functionality of the product, and many customers may not be willing to pay even a rather low price for a product that is known to be of poor quality.
- For most customers – traditional, as well as technophiles – efficient cleaning requires systematic and efficient coverage of the workspace. A domestic cleaning robot that uses sophisticated sensors to achieve systematic and efficient coverage may easily end up with a price tag that is significantly above the price of a comparable nonrobotic device and hence runs a high risk of failure.
- A domestic cleaning robot, which refrains from using expensive sensors to achieve systematic coverage but instead uses low-cost sensors for collision avoidance, fall protection, and confinement as well as using random motion combined with some hard-coded motion patterns to achieve a certain level of coverage, runs a risk of not satisfying those customers who expect systematic cleaning and coverage.

From a developers point of view these lessons sound as if customers expect not less than to square the circle. The truth is that customers do not care about squares and circles. They expect value for money.

With the insights above one could classify potential customers of domestic cleaning robots into three categories:

- Customers who only care about the price and not so much about quality and functionality. The criterion for this class of customers was obvious: reduce the price as much as possible without ignoring that there may be a bottom line for what customers might expect in terms of quality and functionality.
- Customers who care about the price and quality, but are willing to adjust their expectations of functionality and efficiency. The criterion for such customers was to

reduce the price but only to a certain extent, which does not compromise the quality of the product too much. Expectations of such customers in terms of efficiency (i. e., systematic coverage) can be compensated by auxiliary equipment such as automatic charging stations.

- Customers, who care about price, quality, and efficiency. Apparently these customers are the most demanding. None of the second-generation domestic cleaning robots has really managed to satisfy them. The criterion to serve them would be to develop a low-cost navigation system based on low-cost sensors that provides systematic coverage.

As satisfying the last group of customers, which is also likely the largest of the three groups, seemed nearly impossible, the third generation of domestic cleaning robots have diverged somewhat. Some manufacturers have focused on products that serve the first category of customers, some serve the second group and some even made an effort to square the circle.

May they be successful in the end or not, what cannot be overlooked is the explosion of the number of manufacturers and distributors of domestic cleaning robots in the past five years. At the end of the 2012, twenty-seven years after the first mention of a domestic cleaning robot to the public, Amazon alone listed more than 131 results under the key word *robot vacuum cleaner*, with 14 manufacturers and suppliers. The business to business (

B2B

) portal www.made-in-china.com lists more than 71 companies for *robot vacuum cleaner* and a total of 875 products. Some of these 875 robot vacuum cleaners look surprisingly similar to the products sold over B2C platforms under very-well known brand names.

In Fig. [65.3](#) we show some of the products, which primarily intend to serve customers of the first category and therefore stayed under a price of 200 EUR. Figure [65.3a](#) shows a model series of six cleaning robots made by XRobot of the Chinese manufacturer Shenzhen Silver Star Intelligent Electronics Ltd. In B2B trading these robots cost between 64 and 102 USD per piece at a minimum order of 500 pieces. Most models of this series are also sold under the brand name of European and American enterprises. Figure [65.3b](#) shows another series of still rather cheap robotic vacuum cleaners. The attentive reader may notice that the robots in the

left and the right figure are not totally unique. Some just have different names and different prices. This is not entirely unintended. It just illustrates that in domestic cleaning robotics the value creation chain is no longer limited to the developers and manufacturers, just as in every other business.

Fig. 65.3



Third generation of domestic floor cleaning robots: low-cost, low-tech

Apart from a few minor details, all robots in Fig. 65.3 use very similar, though not to say the same, technology. They use very few and very cheap (contact, cliff, sometimes dirt) sensors and random motion combined with preprogrammed motion patterns. The vacuum technology consists of a rotating brush, sometimes combined with a small fan. Not surprisingly, robotic vacuum cleaners like the ones in Fig. 65.3 are often called Roomba-clones. The collection of robots in Fig. 65.3 is neither representative nor comprehensive.

Roomba has undoubtedly written robot history. It is the merit of Roomba and its developers that domestic cleaning robots are no longer considered as gadgets but as real appliances. Roomba has opened the door for all the cleaning robots mentioned above, be they Roomba-clones or not. And Roomba is the by far the best-selling robot ever.

In Fig. 65.4 we show some of the grandchildren and great-grandchildren of the very first Roomba, which started in 2002. The Roomba has gone through several facelifts in these ten years. The latest series, the Roomba 700 is the sixth generation of Roombas. It has matured quite a bit in many respects. It has matured in terms of handling, cleaning technology, obstacle avoidance and navigation technology. At the same time Roomba is delivered in more than a dozen versions, ranging from 250 to 900 USD. These versions differ in the design, in the sophistication of the user interface, in the navigation and coverage strategies, and in the details of the cleaning technology.

Fig. 65.4



The seventh generation of Roombas

According to iRobot more than 6 million units have been sold in the ten years of its existence. Roomba has certainly satisfied quite a number of customers. It is built with good quality, and comes with a self-charging home base and other auxiliary equipment. What has not changed, however, is the basic strategy for covering the workspace. The sixth generation just like the first uses random motion with some precoded motion patterns and heuristics.

This raises one rather fundamental question. Will Roomba's success story continue? Will it maintain its market position? Or has Roomba possibly reached or even passed its summit? These questions will not be answered before the third edition of this book, as any statement in this direction would be pure speculation.

In any case, there are a few recent domestic cleaning robots, which demonstrate that covering a work space in a systematic fashion does not necessarily require expensive sensing, that would push the price for a domestic cleaning robot to a level which customers would not be willing to pay.

Four different key technologies enable the devices shown in Table 65.2 to localize themselves and navigate in an unknown domestic workspace and systematically cover that space:

- Visual odometry, with a camera pointing to the ceiling, combined with simultaneous localization and mapping (SLAM) used by LG Homebot, Samsung Navibot SR 8xxx, iRobot Roomba and Philips Homerun,
- Localization, using infra red patterns projected to the ceiling, and mapping based on contact information used by Mint,
- Proprioceptive motion estimation, using inexpensive inertial measurement units (including inexpensive gyroscopes and accelerometers)
- Simultaneous localization and mapping (SLAM), using, for example a one-dimensional laser rangefinder like Neato XV-xx.

Table 65.2 Systematic cleaning at low-cost. The information in this table was partly collected from the websites and the technical documentation of the manufactures and partly from public websites such as www.botroom.com, staubsaugerroboter-test.org, www.robotreviews.com, www.geek.com, gizmodo.com, www.engadget.com

These technologies by themselves are not entirely new. As a matter of fact methods such as simultaneous localization and mapping, visual odometry, and virtual landmark based navigation are not state of the technologies from a scientific point of view.

The true achievement of the developers of the above systems, which cannot be assessed high enough, is that they managed to reduce the cost of these technologies and at the same time make them robust enough for 24/7 operation. These key technologies will be described in more detail in Sect. 65.1.2 Enabling Technologies. A review of domestic vacuum cleaning

robots and some criteria to evaluate their performance are presented in [Fig. 65.5](#) and [Fig. 65.6](#).

65.1.2 Domestic Window Cleaning

Window cleaning does not seem to be significantly more pleasant housework as floor cleaning. Notwithstanding, robotic window cleaners have not experienced similar attention or progress as floor cleaners. As a matter of fact, today there are only three commercial robotic window cleaners on the market, shown in Fig. 65.5. www.made-in-china.com does not even list one entry for *window cleaning robot*, although two of the three commercial products are Chinese brands. Why is that so, given the fact that domestic floor cleaning has become a billion-dollar business? The answer to this question has two parts: an economic and a technical one.

Fig. 65.5a–c



Commercial robotics window cleaners (a) Windoro, (b) Winbot, (c) Hobot

The economic one is that windows in private homes are cleaned far less often than floors. Customers may hesitate to buy expensive equipment for a task that needs to be done once a month or even less. So the market for domestic window cleaners is presumably much smaller than that for floor cleaners.

The technical one refers to the technical hurdles that have to be overcome. The technical problems that robotic window cleaners have to face are:

- Adhesion to a vertical, fragile surface, which possibly needs to be moistened in order to be cleaned, and related, the power supply necessary to produce that adhesion.

- Requirements for cleaning performance; people may tolerate if the floor is not 100 % clean, but nobody would buy a window cleaner that leaves streaks on the window.

While floor cleaners do not bother about gravity and falling down unless they are near staircases or ledges, gravity is an essential problem for window-cleaning robots, and the solutions are usually not very cheap. Special mechanisms have to be designed for secure motion. Typically special tether mechanisms prevent the robots from falling. Special locomotion mechanisms have to create enough adhesion force to hold a robot attached to a flat, vertical, damageable surface such as glass and at the same time move the body up and down and sideways. These mechanisms have to be small and light and create enough adhesion forces as well as low energy and resource consumption.

The two robotic window cleaners shown in Fig. [65.6](#), RACOON and QUIRL, are research prototypes, which were developed by Fraunhofer Institute for Manufacturing Engineering and Automation (

IPA

) in Germany. In these prototypes the adhesion problem was solved by means of suction cups. RACOON used caterpillar drives that were equipped with passive suction cups. Passive means that the system does not actively create a vacuum in the cup. Rather a small valve aerates or seals the suction cup depending on the position of the cup along the drive. Drives with passive suction cups have the advantage of moderate energy consumption. They have, however, one severe disadvantage. They tend to lose their adhesion after a while. The reason for this is the torque that acts on the center of gravity of the system. Due to this torque there is a traction force acting on the upper cups while at the same time pressure is exerted on the lower cups. Without any attractive force acting on the upper cups the adhesion there gets weaker and eventually the system falls. Therefore passive suction cups are rarely practical.

Fig. 65.6



(a) RACOON and (b) QUIRL, two early research prototypes of domestic window cleaning robots

An apparent solution to this problem is the use of active suction pumps, which generate a vacuum under the upper suction cups. This solution prevents the system from falling. However, supplying the vacuum to the cups makes the system significantly more complex, heavier, and larger. Researchers at Fraunhofer IPA [65.2] have therefore invented a smart solution, which gets by with passive cups, but gets around the problem of decreasing adhesion. The solution uses a spacer at the rear of the vehicle. This spacer neutralizes the torque around the center of gravity which is typical for a systems with passive suction cups. The spacer causes a traction force which acts on the lower suction cups. This traction force creates a torque around the spacer, which counteracts the torque around the center of gravity and also causes a pressure on the upper suction cups. RACOON was presented at the Hannover Fair in 2002 and got quite some attention. But it never became a product. Neither became its successor QUIRL.

In QUIRL, the number of components, the weight, and the size of the system were significantly optimized. The main functions *cleaning*, *holding*, and *moving* were unified in one single component. QUIRL consisted of two vacuum cups that were attached to a common frame which were driven by two separate motors, which rotated independently of each other. The overall motion of QUIRL could be controlled by selecting the velocities and rotational directions of the vacuum cups. If the motor of one vacuum cup was turned off and the cup did not rotate, QUIRL rotated around this fixed cup. If both motors and cups rotated in the same direction this led to an overall rotation of QUIRL about its vertical axis. If both drives rotated in the opposite direction at exactly the same velocity then QUIRL made a linear motion. If both drives rotated in the opposite direction but their velocities were not identical then the translational motion was superimposed by a rotational motion and QUIRL moved along a curved trajectory. In order to clean the surface some cleaning mechanism or tool needed to

be fixed to the vacuum cups. By attaching, for example, specific cleaning towels in the cups the abrasion effect was increased and a very good cleaning performance could be achieved.

The field of robotic window cleaning has not made as much progress as domestic floor cleaning, but it has still made progress. In Fig. [65.5](#) we show three commercially available window cleaners – Windoro WCR I001, Winbot 7, and Hobot 168 – available today.

In 2010 the South-Korean enterprise Ilshim Global Co. Ltd. introduced its commercial window cleaner Windoro (WCR). Windoro consists of two modules, a navigator module and a cleaning module that are held together by two strong neodymium permanent magnets, whose distance can be adjusted. The navigator module and the cleaning module are placed on the inner and outer side of the window, respectively, like a sandwich with the glass pane in between. The two modules operate as a tandem. The navigator module has a differential drive system with two wide-based rubber wheels. Also the cleaning unit moves on two rubber wheels, which serve as spacers for the four spinning cleaning pads. When the navigator module moves, the cleaning module on the other side of the window moves with it. The linkage by permanent magnets obviously has one rather fundamental advantage. As long as the window does not exceed a certain thickness, namely 28 mm, the permanent magnets hold the robot safely attached to the window. Windoro cannot fall, even if its battery runs out of power. For that reason Windoro does not need any safety mechanism like a safety rope. Unfortunately there is a price to pay for this: it is very difficult and unhandy if not impossible to use Windoro if you cannot open the window that you want to clean.

Windoro first explores the width and height of the window before it then starts to clean the window from the top to the bottom in a zig-zag motion, with a velocity of 8 cm/s. While the robot moves over the window the cleaning module sprays a cleaning solution onto the window surface. With four spinning microfiber pads the cleaning module removes dirt and sprayed solution from the surface.

The second commercial window-cleaning robot is Winbot, a product of the Chinese company Ecovacs, which has established its own brand. Winbot was first introduced in 2011. The most recent version Winbot 7 was presented at the electronics fair

CES

(Consumer Electronics Show) 2013 in Las Vegas. Ecovacs also manufactures and distributes the robotic vacuum cleaner family Deebot.

Winbot uses two suction rings and a vacuum pump for the adhesion at the window. The outer ring also serves as a safety mechanism. If the air pressure in the ring increases, that means Winbot has reached the edge of a window plane, it backs up, turns around and moves in a different direction. A second suction cup serves as a safety anchor for Winbot while it moves. It is connected to Winbot by a safety rope and catches the robot if it falls. Winbot can move at a velocity of approximately 15 cm/s. Its drive system consists of two differentially driven anti-slip rubber tracks.

After it has been turned on, Winbot, like Windoro, first explores the height and width of the window, then calculates a zig-zag path that covers the window area and finally executes this path. Winbot has no *active* cleaning technology such as spinning pads. It uses two micro-fiber towels that are attached to two plates at the front and at the rear of the robot. The micro-fiber towel in the front has to be moistened before Winbot starts moving. It resolves and removes the dirt. A rubber blade behind the front towel removes the remaining moisture. The micro-fiber towel in the rear of the robot finally dries the window.

The third, and most recent, system, Hobot 168, is a product of the Taiwanese company Hobot Technology Inc. It was first presented at

IFA

2012 (Internationale Funk Ausstellung) in Berlin and launched in summer 2013. It looks surprisingly similar to Quirl in Fig. [65.5](#) and apparently uses a similar adhesion and locomotion mechanism, namely two rotating vacuum cups, to move across the window.

The preceding paragraphs read as if the times are over where windows had to be cleaned by hand. Regrettably this is not the case. Although all three devices reasonably solved the adhesion problem their overall performance is modest. Several tests by housewives and

magazines came to the same sobering conclusion: There is a lot of noise and very little cleaning. As a matter of fact, although the navigation problem on a vertical rectangular surface, which is free of obstacles apart from the window frame, seems to be a solvable one, Hobot 168 and Winbot showed a rather poor performance in terms of systematic coverage. After a semisuccessful effort to explore the width and height of the window, Winbot, in one of the tests, moved more or less erratically for several minutes, before it gave up somewhere along the road. Several of the currently available commercial window cleaning robots are reviewed in [\[45\]](#), [\[46\]](#), [\[47\]](#), and [\[48\]](#).

65.1.3 Pool Cleaning

While robotic floor cleaner and window cleaner were still struggling to get rid of the image of only being the crazy ideas of engineers and researchers at the beginning of the millennium, pool-cleaning robots were already well-established products. This may be due to the fact that the challenge of cleaning a rectangular pool is rather modest and so is the *robotic technology* used in robotic pool cleaners. It may also be due to the fact that pool owners belong to a class of customers who did not create the same price pressure as the ordinary housewife.

First patents on self-propelled pool cleaning devices date back to 1965, three years after the first industrial robot was installed. Ferdinand Chauvier, a South African engineer, could possibly be considered the father of automated pool cleaning. He developed several generations of devices for pool cleaning before he finally marketed *Kreepy Krauly*, the first automated pool cleaner in 1974. *Kreepy Krauly* was not only the first one of its kind but also the very first domestic service robot ever, 15 years before Joe Engelberg published his book *Robots in Service* at MIT Press in 1989 and coined the term *service robot*. Since then the technology of pool cleaning robots hasn't changed much.

As one can see in Fig. [65.7](#), most pool cleaners have track drives, which are operated differentially. Typically the motors that drive the tracks are also connected to the front and rear scrubbing brushes, which clean the pool surface, while the robot moves. While tracked drive system were rather common for the early generations of pool cleaners, newer pool cleaners also use wheel drives, for example, the Polaris 9400 from Zodiac Pool Systems. A wheel drive can be advantageous because the space between the bottom of the robot and

the ground allows a better water flow and a higher throughput. Also Zodiac claims that Polaris 9400 has higher maneuverability.

Fig. 65.7



Robotics pool cleaners

Since most sensors, which are used in mobile robots or in aerial robots do not work under water, the sensor modalities that are used by pool cleaners are manageable. Unfortunately the manufacturers do not disclose much about the sensor technology and technology in general used in their system. So we need to speculate a bit about how the behaviors shown in the commercials are internally implemented.

Most pool cleaners claim to be capable of avoiding obstacles. When they sense a collision, the back up for a certain distance, reverse the direction of motion and then continue, possibly on a track parallel to the one the led to the obstacle. They can *recognize* the walls of a pool, which form the borders of their workspace. They can even climb up the walls of a pool, float along the perimeter and then submerge again. They can *explore* the length and width of a pool. They can drive a certain distance before they change direction.

All these behaviors require a combination of several of the following sensors: *odometry* to measure traveled distance, an *inclinometer* to sense if the robot starts moving upward, e. g., when it keeps pushing against a wall and the front wheels start moving in a vertical direction, *contact sensors* to detect when the robot collides with an obstacle – be it a wall of

the pool or a real obstacle on the ground of the pool – *sensor to measure the motor current*, which can be used in addition or instead of a collision sensor to detect if the robot pushes against an object, and possibly an *inertial measurement unit* to correct the heading when driving. Polaris and possibly other systems as well use an accelerometer to constantly determine its position in the pool. A sensor modality, which might be used underwater for obstacle detection and avoidance is laser, with wavelengths in the lower nano-meter range (e. g., 405 nm blue laser). But it is unknown if this principle has been considered for pool-cleaners.

In terms of *navigation and coverage* pool cleaners follow similar strategies to those floor cleaners, which we roughly classified as Roomba-clones. Earlier pool cleaners used random motion pattern. The newer ones shown in Fig. [65.7](#) use certain heuristics and strategies to perform some form of localization and exploration of the pool. The Maytronic's Dolphin, for example, first explores the length and the width of a pool. After being dropped into the pool and floating to the bottom of the floor it crosses the pool until it hits the first wall. With the support of a thruster it climbs up the wall until it reaches the surface. There it hovers to the side before it submerges again and moves back to the ground of the pool. Next it moves to the opposite side of the pool, climbs up the wall, hovers to the side and glides back to the ground. While moving on the ground from one wall to the other Dolphin measures the distance from wall to wall. Once it glides back from the second wall it moves halfway back toward the other wall. It then makes a 90 ° turn and repeats the exploration for the second set of opposite walls. After the exploration is completed, Dolphin knows the length and width of the pool and plans a pattern of parallel and orthogonal tracks as shown in Fig. [65.8](#) that in the end covers the entire pool.

Fig. 65.8



Example of the navigation and cleaning strategy of a robotic pool cleaner

Removing dirt and debris from the ground and the walls of a pool, requires loosening the dirt – if it is not loose already – and to soak it into a container, otherwise the dirt would only be circulated in the pool. The container is typically a jet pipe with a filter at one end that holds the dirt back while the water flows through the pipe back into the pool. Soaking in water and debris from the floor into a filter and pumping it back into the pool requires significant suction power. This is the reason why pool cleaners typically have an external power supply. This motor serves as a pump and as a thruster at the same time. Both effects together allow pool cleaners to easily climb up vertical walls. Using inflation at the bottom of the robot, and a water jet ejected at the top, enough traction power is created that both tracked and wheeled pool cleaners can drive up vertical surfaces.

As mentioned earlier, the cleaning technology, besides the water pump and the filter to keep back the debris, consists of a system of counter-rotating rubber brushes which brush the debris underneath the pool cleaner, where it ends up in the intake socket of the water pump.  and  compare a selection of home pool cleaning robots.

65.1.4 Lawn Mowing

Together with robotic pool cleaners and domestic floor cleaners, robotic lawnmowers today count as regular everyday products. People no longer consider them mystical pieces of technology, which have their own life and which at times become so autonomous that the user no longer knows what they are after.

It is not surprising that robotic lawnmowers have a lot in common with domestic floor cleaners. They have to cover a workspace of a certain size with as little interaction with the owner as possible. They have to perform a certain operation to the surface such as cleaning the floor or mowing the lawn. They must not collide with any obstacles and if they do, they should at least not cause any damage. They must not get stuck anywhere in the environment and they should not leave the workspace without authorization.

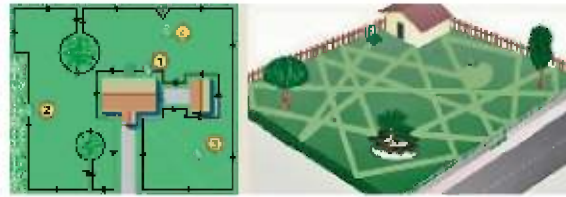
The challenge for lawnmowers, much the same for domestic floor cleaners, is the systematic coverage of the workspace, which in turn requires precise positioning and mapping of the workspace. Given the fact that the price pressure is not as back-breaking as for robotic floor cleaners – there are not that many robotic lawnmowers which costs less than 1000 EUR – why not invest a little more in sensing and especially in position sensing and obtain a decent solution for the localization and coverage problem?

The answer to this question is: things are not that simple. Lawnmowers operate outdoors and none of the solution developed for the floor-cleaners in Fig. [65.2](#) will work. Regular

GPS

(global positioning system) has an accuracy of several meters and has a tendency to deliver erratic readings, which would lead to equally erratic motions of the lawnmowers. As a matter of fact Automower 220 from Husquana uses GPS, but only as an antitheft protection device. Differential GPS, which would provide accuracy below one meter, would be too expensive for an affordable robotics lawnmower. In a nutshell, absolute positioning or SLAM, which would be necessary to cover a large outdoor area like a garden in a systematic fashion, is not practical.

In order to cover their workspace robotics lawnmowers use similar strategies to the Roomba-like floor cleaners. They refer to heuristics, which do not provide an optimal performance but still show a decent result. Using sensors such as gyroscopes, digital compasses, or inertial measurement units, robotic lawnmowers follow a certain heading and cover the workspace as much as possible by parallel tracks. They move along a straight line until they hit an obstacle or reach the border of the workspace. There they back up to become clear from the obstacle, make a U-turn by 180° and drive back the way they came. Another heuristic that is frequently applied by robotic lawnmowers is the random motion shown also in Fig. [65.9](#). The mower moves along a straight line until it hits an obstacle or reaches the border of the workspace but then it does not just reverse but chooses a new direction randomly.

Fig. 65.9

Example for virtual fence and coverage strategy (random motion) of robotics lawnmowers

Since robotic lawnmowers move in open space, there is a danger that they leave their workspace and travel to areas where they are not supposed to be. For floor cleaners so-called virtual walls or fences solve this problem. Virtual walls are realized by infra red light beams emitted by some projectors, which can be placed in the workspace. The robots can sense these infrared light beams and consider them as obstacles, which evoke the typical obstacle avoidance behaviors. Lawnmowers use a similar technique, which is based on induction rather than light. To mark the border of the robot's workspace the owner has to place a wire around the area, which the robot must not leave (Fig. 65.9). This wire is connected to a low-voltage alternating current source. When the robot approaches the wire an inductivity sensor senses the current in the wire and causes the robot to reverse. Today most lawn mowers use a more sophisticated so-called *true in/out systems*, where the position is permanently tracked, not only when the robot approaches or passes the virtual fence [65.3].

The very first robotic lawnmowers had to be recharged manually. To avoid an all too frequent involvement of the human into the operation, Husqvarna – one of the pioneers in robotic lawn mowing – equipped its first mower with solar panels and called it *SolarMower*. SolarMower was released in 1995, and was one of the first robotics lawnmowers. Nowadays, all lawnmowers come with a base station where they can recharge their batteries without human intervention. Also solar panels are coming back as power source, however only as auxiliary power to increase the performance and runtime between two charges and not as main power supply.

Given the fact, that robotic lawnmowers do have poor positioning capabilities, it is somewhat tricky to guide them back to the docking station once the battery gets low and needs to be recharged. Some of the lawnmowers shown in Fig. 65.10 use special wires which radiate from the position of the docking station. This way a robot only needs to follow such a radiating wire to return to the docking station on the fastest way. Another strategy is to follow the wire at the workspace border. That will eventually lead the robot back to the docking station.

Fig. 65.10



Robotic lawnmower

An exception is the Bosch Indego, which is equipped with mapping and localization capabilities. This allows Indego to plan a path that leads to a point close to the docking station. Then the robot can follow a wire for the docking maneuver [65.1].

To comply with the safety regulations and to avoid any injuries of humans or animals the cutting mechanism has to be very lightweight and designed such that it is guaranteed to stop if the device tilts or is lifted. The cutting mechanism of robotic lawnmowers typically consists of a rotating disc with three razor-like blades, which automatically retract into their mountings if the robot is stopped unscheduled, for example, if it hits an obstacle or is lifted.

Naturally the lightweight design limits the thickness of the grass that can be cut. Also it is important for the proper functioning of robotic lawnmowers that the lawn, which is to be cut, is not too high. This in turn requires a rather regular if not continuous operation of the

lawnmowers. With regular use of the lawnmowers the grass cuttings are short enough to quickly decompose into nutritious compost, so there is no need to remove the cuttings after the lawn is mowed.

Figure [65.10](#) shows some of the better-known robotic lawnmowers on the market today. Besides the pioneers in robotic lawn mowing Husqvarna, Friendly Robotics and Zuchetti also a number of new players have entered the market, most noticeably the German automotive supplier Bosch and the Japanese car manufacturer Honda, which has set a milestone in nonindustrial robotics with its humanoid robot Asimo.

We certainly do not claim that the collection in Fig. [65.10](#) is complete. Like for domestic floor cleaning, Chinese B2B portals such as www.made-in-china.com or www.alibaba.com list around a hundred products under the category of robotic lawnmowers and 35 suppliers; the market development for robotic lawnmowers in the past five years was not quite as overwhelming as for domestic floor cleaners but was still remarkable. In Table [65.3](#) we show the technical specifications of a selection of lawnmowers from Fig. [65.10](#).

Table 65.3 Technical specification of a selection of robotic lawnmowers

65.1.5 Sports Robotics

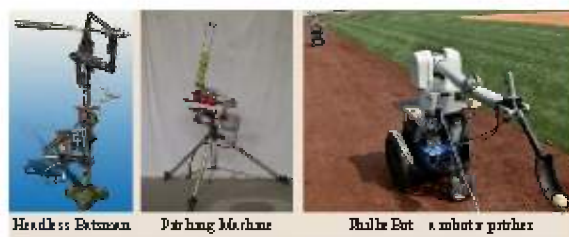
A subdomain of domestic robotics that was not included in the first edition of this handbook, because it virtually did not exist, or it was not visible at that time, is *sports robotics*. What is a sports robot? Since an official definition of this term does not yet exist – at least we haven’t found one – we take the liberty and provide such a definition here. We define a sports robot as a *robotic device*, which either *supports* the human user in their physical exercises as a *coach or a companion*, or acts as an opponent in a game. An important aspect of this definition is the physical exercise of the human, which is supported or challenged.

We would like to emphasize that the above definition does not include any form of entertainment robots, which play games such as soccer against each other but do not involve

any human activity other than watching.

In Fig. 65.11 we show three examples of *robotic baseball players*. As the name says the *Headless Batsman* can act as a batter at least for exercising. It rather successfully hits baseballs thrown at it in many ways. The kinematic structure consists of *two arms, one leg and no head*, as its developer, an industrial designer from Robocross, likes to call it, and is made from auto parts, steel pipes, and pneumatic hoses. The pneumatic hoses are parts of its pneumatic actuation by an air compressor. The *Headless Batsman* is fully controlled by a human operator via a remote control, which has three buttons: one to control the robot's hip, a second one that actuates the arms and a third one to lift and drop the inside shoulder, changing the trajectory of its swing. Similar mechanisms like *Headless Batsman* were developed at Hiroshima University and Tokyo University.

Fig. 65.11



Robotic baseball players

The counterpart of *Headless Batsman* is a pitching machine, which is operated with the same remote control. The barrel of the machine is sawn-off fire extinguisher whose other end is directly connected to a one-inch port poppet valve, which in turn is connected to an 8 bar air pressure tank. The air pressure tank is supplied from a screw compressor. The charging mechanism consists of a double rod actuator with a ring welded horizontally on to the end. Mounted above the ring is a magazine that holds up to 10 balls, which directly drop into the ring. For a pitch the actuator moves the ring with the ball over the edge of the barrel, where it falls on to a small guide. From there the balls rolls backward into the barrel. The actuated ring is covered by a steel strip, which holds back the other balls in the magazine until the actuated ring returns to its initial position. Since both mechanisms are remotely controlled,

the human, who controls them, can actually sit in a chair next to the playground and let the robots play against each other. Such a use would clearly violate the above definition of a sports robot but does not seem to be the one that is primarily intended.

Another robot that can throw baseballs is Philliebot ([video](#)) developed in University of Pennsylvania's GRASP laboratory. Philliebot was developed in a couple of weeks using only spare parts in the GRASP lab. It uses a Segway as mobile base, a Barret arm, and pneumatically actuated wrist to create the necessary dynamics for the pitch. When the button is pressed the arm moves to the back of the robot and then accelerates its motion toward the target of its pitch. When the arm reaches the highest point of motion the pneumatic wrist cylinder delivers a burst of compressed carbon dioxide to snap the wrist forward and release the ball. What remains is the question of why use robotic equipment worth several tens of thousands of dollars to throw a baseball, given that pitching machines have existed for many years. According to the developers, the fact that Philliebot is mobile and its software can be tweaked to vary pitch velocity and trajectory was enough to justify the experimentation.

The two sports robots shown in Fig. 65.12 do not really seduce their users to sit in a chair and relax. They both serve as so-called *robotic running coaches* or *running companions*.

Fig. 65.12



(a) JoggoBot and (b) RUFUS two robotic sports companions

Researchers at the Royal Melbourne Institute of Technology in Australia have redesigned a commercially available Parrot AR Drone quadcopter and turned it into an autonomous,

flying running partner for joggers, called Joggobot [65.4]. Joggobot uses an integrated, front-mounted camera to detect and track a special pattern printed on the T-shirt worn by the jogger. Joggobot takes off when the camera registers the pattern and rises to about the same height as the pattern on the t-shirt. An internal sensor determines Joggobot's altitude. Joggobot can be set into a *companion mode*, in which it flies at a steady pace at a relative distance of about three meters to the jogger or in a *coach mode* in which it flies at a slightly more challenging speed.

There are two features of Joggobot that make the device somewhat limited: first the capacity of the battery limits the flight time to 20 min, which in turn limits the time for exercise; for a short run, this is certainly ok, but for serious training this is insufficient. Second, Joggobot can only fly in a straight line, to let Joggobot follow an arbitrary path the jogger needs to remotely control Joggobot's flight path.

A slightly different concept of a jogging companion is pursued in RUFUS (RUFUS ROBOT), which is developed by runfun (www.runfun.com), a German startup company. RUFUS is an electrically driven, automatically guided, ground vehicle that supports and guides a runner during his/her training. RUFUS plays the role of a *personal running coach*. It fulfills a similar function to a treadmill, which exposes its user to a varying strain by varying its velocity and inclination and thereby improves the fitness, endurance, and resilience of the cardiovascular system of the user. Unlike a treadmill, RUFUS is not a stationary device, however. It drives ahead of the runner like a pace maker in a marathon or a fake rabbit in a dog race, and sets the runner's speed. RUFUS's velocity is set either manually or automatically via a training program.

If operated in *manual mode* the velocity is either set directly as a velocity set point or indirectly as a heartbeat set point. If the training guidance is based on the heartbeat, then RUFUS controls its velocity such that the runner exercises optimally and continuously within a certain heartbeat interval under a moderate strain of the cardiovascular system.

This has a twofold use: on the one hand this prevents users from overstressing themselves through overambitious and intensive training modules, possibly from even injuring him or herself. Such a protection function is beneficial for unfit or untrained runners. On the other

hand RUFUS facilitates optimal training effect and progress through a careful guidance of the training.

The training effect can be further improved if RUFUS is operated in the *program mode* instead of the manual mode. In this mode RUFUS executes complete training modules, for example pyramid speed interval workouts, which are customized to the user. Such training modules are typically elaborated on by physiotherapists or sports physicians. They can be downloaded to the RUFUS embedded PC like an app from an app store.

RUFUS has a major advantage over JoggoBot. It has a battery capacity that allows it driving for about six hours on a flat road without recharging.

65.1.6 Tele-Presence

In a world in which not only large, but even small and medium-sized enterprises operate globally, in which families are scattered over continents, in which ubiquitous presence seems to become an essential requirement for professional progress, and in which professional services are more and more delivered over the internet, tele-presence has become a fast growing market in the past years.

Robotics adds a very important aspect to plain *tele-vision* by turning it into *tele-presence*: embodiment and remotely controlled motion. As tele-presence is nothing but the combination of tele-vision and tele-operation using a robotic device, which are often called *tele-presence robots* or *robotic avatars*.

Tele-presence robots offer a whole spectrum of services and applications ranging from plain mobile video-conferencing systems to tele-surveillance, tele-diagnosing, and tele-care, to tele-teaching and tele-commuting. The term *tele-commuting* was coined by Scott Hassan, a Google developer of the early days, nowadays entrepreneur and investor, and founder of WillowGarage and Suitable Technologies, the manufacturer of Beam-RPD (see also Table [65.4](#)).

Table 65.4 Technical specification of a selection of tele-presence robots

A *tele-presence robot* typically consists of a mobile robotic platform that:

- Can be tele-operated through some user-interface,
- Carries a camera, which often can be actuated separately (via a pan-tilt unit) and allow the operator to actively explore the remote environment, and
- Carries a display, which allows those at the remote site to see the operator of the tele-presence system and communicate and interact with it.

Figure [65.13](#) shows a collection of such tele-presence robots. The devices range from a price of some US\$ 1500 for TeleMe from MantaroBot to more than ten-fold that amount for Beam RPD from Sutable Technologies. The system

RP-VITA

(Remote Presence Virtual+Independent Telemedicine Assistant) emerging from a cooperation between InTouch Health and iRobot is available only for lease, at a monthly fee of \$ 4700. Giraff is the result of a European research project lead by Giraff Technologies funded by the European Commission and is not commercially available yet.

Fig. 65.13



Tele-presence robots and robotic avatars

Not all tele-presence systems shown in Fig. [65.13](#) can be classified as domestic robots. A system that clearly stands out and is by no means a domestic robot is *RP-VITA* by InTouch Health and iRobot. *RP-VITA* is a remote healthcare system. *RP-VITA* shall enable doctors to command any clinical, patient or care team management process remotely. *RP-VITA* has a full-fledged autonomous navigation system that allows the personnel to focus on the patient care task rather than on remote navigation. This feature has been awarded clearance by the US Food and Drug Association (

FDA

). *RP-VITA* further provides access to important clinical data to support physicians, nurses and other care personnel in their diagnosis and other medical workflow. For example *RP-VITA* connects with diagnostic devices such as ultrasound and comes equipped with the latest electronic stethoscope. So *RP-VITA* is in a class of its own, which may also justify the higher price.

Apart from *RP-VITA*, the tele-presence robots shown in Fig. [65.13](#), can all be classified as semiprofessional or domestic service robots. The functionalities and services they offer do not necessarily vary on the same scale as their prices. This can be seen by a comparison of two of the above systems: Double and Beam RPD (see also Table [65.4](#)).

Double is not much more than a mobile iPad equipped with the video-conferencing system Opentok. The mobile base uses a Segway-like dual-wheeled drive system that can balance a pole, which holds the iPad. When *Double* stands still, two retractable kickstands are deployed and allow the system to put the control system in an idle mode and save energy. *Double* can be remotely controlled and driven around a remote site through an app installed on a second iPad that enables communication with all known Doubles over the web. The height of the iPad holder can be remotely adjusted to enable communication at eye-level. Double Robotics list a number of potential services and applications for which *Double* could be used: Companies with sites at various locations can use *Double* to improve communication and collaboration between remote teams. Families can use *Double* to communicate with family members living abroad. Museums and art galleries can use Doubles to offer remote tours through their exhibitions.

Beam RPD uses two HD cameras with custom wide-angle lenses instead of the plain iPad camera. This gives *Beam RPD* peripheral vision that is comparable to a human's field of view. A digital zoom lets the operator further focus on details in the remote site. *Beam* uses an array of six microphones and audio processing algorithms, background noise reduction and echo cancellation. This equipment gives *Beam* an audio-quality, which obviously goes far beyond that of an iPad. *Beam* uses a 17-inch screen mounted at a height of 1.58 m that allows the display of a human face at its natural size and height. Another feature that goes beyond iPad standards is the WiFi connectivity. To provide reliable and seamless WiFi connectivity, *Beam* uses two dual-band radios and proprietary roaming algorithms. Altogether it is obvious that *Beam RPD* is far more than a movable iPad. It is left to the customers to decide whether this is worth a price which is an order of magnitude higher. [\[45\] NDBOT41](#), [\[45\] NDBOT42](#), [\[45\] NDBOT44](#), and [\[45\] NDBOT45](#) introduce several of the tele-presence robots available on the market today.

65.2 Enabling Technologies

The mass consumer market is very price-sensitive, so the price of the robot is key for the success of the product among consumers. Certain guidelines used in the consumer electronics market are relevant for the domestic robot market to provide a rough estimate of cost of the robot. Let's say that you want to develop a floor-care robot that would retail at \$ 300, the *rule of thumb* is that your bill of materials (

BOM

) should be between $1/3 - 1/5$ of the retail price. In other words, your BOM should be within \$ 60–\$ 100! And the BOM must include all mechanical parts, electrical parts, battery, processor, memory, motors, assembly, packaging, user manuals, etc.

Given the extreme cost constraints outlined above, this chapter focuses on enabling technologies that are viable, from the cost point of view, to be included in a mobile domestic robot with a price lower than \$ 1000 (or ideally below \$ 500). These technologies are required to have a reliability level in line with the expected life time (and warranty) of the product; otherwise, no matter how good the technology is, if it stops working in an unreasonable period of time, the robot will be returned to the retailer. Special emphasis should be placed in the ease of manufacture of the technology. Difficult to manufacture components create delays in the production line, decreasing the yield of the product and eventually increasing the overall cost of production, leading to either eroding profit margins or a rise in the retail price.

Mobile robots need to sense and understand the environment in which they operate. The first key enabler is the capability of detecting obstacles and hazards to safely and accurately navigate around them. Section [65.2.1](#) describes the different available technologies for obstacle and hazard detection. The second key enabler is the ability to localize and create a map of the environment to intelligently plan actions and motions that allow the robot achieve its goal. Section [65.2.2](#) presents the technologies available for localization and mapping using a number of low-cost, yet powerful sensors. Section [65.2.3](#) discusses alternative approaches to coverage of the space implemented in commercial products.

65.2.1 Sensing and Obstacle Avoidance

Domestic robots aim to take care of tedious chores, interacting with a household that includes owners, children, babies, pets, and stationary objects such as chairs, tables, walls, etc. Domestic robots must be safe in order to gain acceptance in our daily life: it is not tolerable to have a robot falling down the stairs or hurting a household member. Thus, robots must be equipped with drop/cliff sensors and proximity sensors that ensure proper operation while still satisfying the mentioned cost constraints.

65.2.1.1 Cliff Sensors

A number of solutions are present in current robots in the market. Off-the-shelf solutions are IR sensors from Sharp that consists of an emitter (light emitting diode

LED

) and a receiver (photodetector or position-sensitive-device

PSD

) that provides an output proportional to the distance to the object. The Roomba uses a custom IR cliff sensor based on a similar principle that trades generic distance measurement with sensor cost. The Mint robot from Evolution Robotics employs a factory-calibrated mechanical hammer that triggers upon cliff detection. Solid-state sensors are usually more reliable than mechanical sensors, since they do not have moving parts, but have the drawback of a response dependent on the reflectivity of the surface in the IR spectrum and a dead-band in the response.

65.2.1.2 Contact and Proximity Sensors

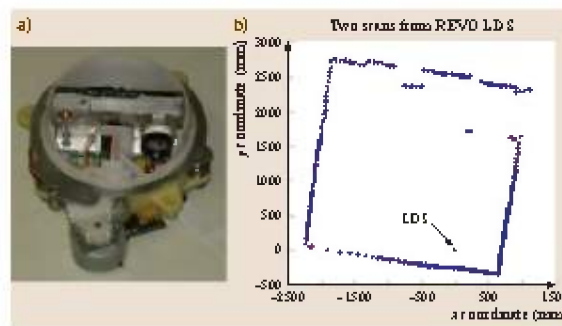
Mechanical switches, called *bump sensors*, are commonly used for detecting when the robot gets in contact with obstacles. Bump sensors are cost-effective solutions providing the ability to stop the robot without damaging the obstacle. Touching obstacles is not desired unless performed very gently to ensure that the robot goes under curtains and bed skirts. IR and sonar sensors are frequently employed as touchless alternatives to the bump sensors by measuring the distance to obstacles. Both types of sensors are composed of an emitter and a receptor with an output proportional to the measured distance. IR sensors are usually more focused than sonars and less sensitive to multiple reflections on walls and other obstacles, but might lose thin obstacles such as chair legs. This type of sensors provides a point-wise measurement of distance to obstacles, so a robot needs a number of these sensors to obtain a dense representation of the obstacles in the environment. The information on obstacles and hazards (cliffs) is collected in occupancy grid maps and used for decision-making in systematic cleaning robots. A number of cost-effective dense distance measuring sensors have recently appeared in the market and will be discussed in the next sections. These dense distance measuring sensors have a cost on the order of tens of dollars, while the point

distance measuring sensors have cost only a few dollars, so most of the domestic robots currently available on the market have yet to incorporate dense distance measuring sensors. The only exception is the Neato XV-21 that uses a low-cost laser distance measuring system.

65.2.1.3 Laser Distance Sensor

A low-cost Laser Distance Sensor was developed by Konolige and colleagues [65.5] using a laser point beam and a global shutter CMOS imaging sensor separated by a small baseline. The system operates by triangulation and achieves full 360 planar scan by rotating the optical assembly on a full circle. The sensor has a range of 0.2 to 6 m with an error < 3 cm at 6 m and an angular resolution of 1 degree, providing 4000 readings per second (up to 10 Hz) with a small size (approximate width of 10 cm shown in Fig. 65.14a) and low power (< 2 W). The sensor is eye safe and provides measurements that enable laser-based SLAM as shown by Fig. 65.14b.

Fig. 65.14a,b



Laser Distance Sensor. (a) Prototype and (b) occupancy map generated with the sensor

65.2.1.4 Structured-Light Distance Sensors

Structured-light distance sensors consist of an emitter that projects a known pattern on the environment and a receptor that computes depth based on the deformation of the received pattern. The Kinect [65.6] interface to the Xbox game system uses a structured-light sensor from PrimeSense [65.7] thus showcasing the readiness of this sensor for consumer applications. The emitter consists of a laser with optics that projects a known pattern (Speckles [65.8, 65.9]) in near-IR light and a complementary metal-oxide-semiconductor (

CMOS

) IR camera that observes the pattern to estimate depth using triangulation. The emitter and the camera are calibrated during manufacturing assuming a rigid configuration. The speckles can be further shaped into ellipsis using optics with different focal lengths in x and y so that the orientation of the observed ellipsis is proportional to depth. Speckles of different sizes are used to obtain different depth accuracy depending on size. Figure 65.15a shows the speckles pattern projected by the sensor and an image taken in the dark by photographer Audrey Penven showing the IR speckles. Figure 65.15b shows the components of the Kinect sensor and the corresponding RGB and depth images of a scene.

Fig. 65.15a–c



Primesense sensor: (a,b) Speckles pattern projected by the laser emitter. (c) Block diagram of the Kinect sensor

65.2.1.5 Time of Flight (TOF) Distance Sensors

Time of flight (

TOF

) sensors consist of a light source (usually a laser) that emits a continuous waveform and a special imaging sensor that measures the phase shift of the received signal in each pixel.

The depth at each pixel is proportional to the phase shift. TOF sensors have been in the market for quite some time, but the need for allocating a large portion of the sensor to the decoding electronics has made it challenging to produce low-cost sensors at reasonable resolution. Some of the companies offering TOF sensor have been Mesa Imaging AG that produces the SwissRanger [65.10] sensors, Softkinetics [65.11], PMD Vision.

65.2.1.6 Stereo Vision

Stereo vision is a well-known computer vision solution to the extraction of three-dimensional (

3-D

) depth maps in areas with sufficient texture to find image correspondences. As opposed to structured light or TOF sensors, stereo vision systems are totally passive, but require a calibrated stereo rig, and their performance depends on the level of external illumination and on the amount of texture present in images.

The selection of the optimal dense mapping sensor depends on the application. The laser distance sensor provides reasonable information for laser-based SLAM and obstacle detection and avoidance; however, it only provides range information on a plane as opposed to the dense 3-D range offered by the structured light, TOF, or stereo systems. The structured light sensor uses a simple imaging sensor but requires additional computation to estimate depth in each pixel while the TOF sensor computes depth in each pixel at the expense of a sensor with lower fill factor. The stereo system does not require additional lighting, but requires an additional camera and a computation module to extract depth. Other parameters to consider are the maximum and minimum range that the sensor provides to ensure that it fits the requirements of the application in terms of mapping and obstacle detection.

65.2.2 Localization and Mapping

A robot that knows its location and understands its surroundings is able to plan intelligent maneuvers to achieve its goals. Localization and mapping are basic primitives that enable smart and efficient behavior. Early successful robots like the Roomba chose to sacrifice

localization and mapping in order to achieve an appealing retail price since the localization and mapping technology was either too expensive or not existent at the time (many of the dense sensors presented in the previous section were developed after the Roomba). In recent years, a number of low-cost, yet powerful simultaneous localization and mapping (SLAM) technologies have been developed and integrated in floor-care products. These technologies are described in the following sections.

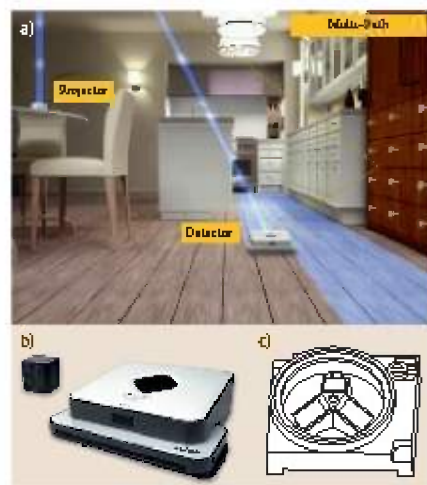
65.2.2.1 Vector Field SLAM

The Mint robot by Evolution Robotics uses active beacons for localization. The Northstar beacon projects two IR spots onto the ceiling that are modulated to simplify the detection of the spots with the sensor. The spots are invisible to the human eye and thus do not produce visual clutter. While placing the beacon can still be regarded as a modification to the environment, surveys among our customers suggest that the vast majority largely accepts setting up the beacon prior to running the robot [65.12]. The Northstar sensor uses 3 photodiodes to compute the direction to the spots from the measured current through the photodiodes. The photodiodes can be sampled at high frequency to detect the modulated spot frequencies for data association. The sensor is quite inexpensive and suitable for cost-sensitive applications. However, the sensor suffers from multipath since light not only reaches the sensor directly but also through reflections from walls and other furniture, making the position computed by the sensor inadequate to be directly used for localization and mapping. The sensor is augmented with a localization method that learns the light distribution in the room through a simultaneous localization and mapping (

SLAM

) approach [65.13]. Figure 65.16 shows the Mint robot cleaning the environment depicting the operation of the Northstar localization system and the problem of multipath. The figure also shows the Mint robot and the Northstar cube and the Northstar sensor.

Fig. 65.16a–c



Mint robot. (a) Normal operation of the Mint robot using Northstar (the *yellow path* indicates multipath). (b) Mint robot and Northstar cube. (c) Northstar sensor

In vector field SLAM the spatial variation of continuous signals is learned and simultaneously used for localizing the robot by fusing information from dead-reckoning (odometry and gyro) and Northstar. In the following, this method is introduced and tailored toward measurements obtained from Northstar. The signal field is represented as a regular grid of fixed node positions $b_i = (b_{i,x}, b_{i,y})^T$, $i = 1 \dots N$, where each node $m_i \in R^4$ holds the expected Northstar positions of both spots when placing the robot at b_i and pointing it in a fixed orientation $\theta_0 = 0$. Vector field and robot pose are then estimated through the application of SLAM.

Let the robot path be a time series of poses $x_0 \dots x_T$, $x_t \in SE(2)$, i.e., the set of rigid transformations in the horizontal plane, and let $x_0 = (0, 0, 0)^T$. At each time step $t = 1 \dots T$ the robot receives a motion input u_t with covariance R_t and a measurement $z_t = (z_{x1}, z_{y1}, z_{x2}, z_{y2})^T$ of the two Northstar spot positions with covariance Q_t . The spot positions are also each affected by the rotational variability denoted $c = (c_x, c_y)^T$. The rotational variability models errors in measuring the direction to the spots caused by not having the sensor perfectly level.

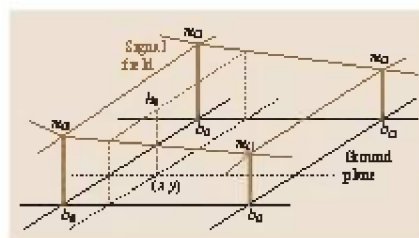
The SLAM problem is solved with the ESEIF-SLAM that is constant time and requires memory linear in the size of the area explored by the robot. The method has been implemented in the processor of Mint, an ARM 7 processor with 64 kByte

RAM

[65.13]. Vector field SLAM has been extended to address covering larger areas by using more Northstar beacons [65.15] and to provide a solution for the re-localization of the robot after it has been kidnapped or paused and resumed [65.16].

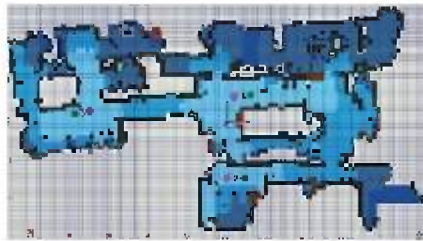
Figure 65.18 presents the map obtained by the robot after a cleaning run in a home environment of 125 m^2 covered by three Northstar beacons marked with the pink and green disks (units are in meters). The robot navigated in the home by following a cleaning strategy based on systematically covering sectors of the environment. As long as at least one beacon is visible to the robot, the strategy moves the robot onto a neighboring region until no space is left to clean. At the end the robot follows along the perimeter of detected obstacles for a thorough cleaning around walls and furniture. As the robot moves through the environment it creates an occupancy grid map using the position information from localization. Each visited cell is classified into one of the following categories: obstacles (drawn in black), floor changes (drawn in brown), hazards (drawn in red), and free space (drawn in shades of blue that indicate visibility of the spots, the lighter the blue, the better the visibility).

Fig. 65.17



Vector field SLAM with Northstar

Fig. 65.18



Robot map obtained with vector field SLAM in a 125 m² home environment

65.2.2.2 Visual SLAM

Visual localization and mapping is attractive for a variety of applications due to the rich input and the low cost and footprint of visual sensors. The difficulties lie in robustly extracting a critical subset of information from the high-rate visual data stream and processing it efficiently to yield useful output. Despite steady increases in the computational power of most platforms, such challenges are nonetheless exacerbated by the limited processing and storage provided by low-cost, embedded systems appropriate for low-power applications or consumer products. Many state-of-the-art approaches to visual SLAM depend on a per-frame processing rate sufficiently high, relative to the speed of camera motion, to permit strong temporal assumptions on the image sequence. This section reviews two approaches to visual SLAM focused on domestic robots. Ceiling vision similar to the work of Jeong and Lee [65.14] can be found in the Navibot from Samsung, the Roboking from LG, and the Iclebo/Homerun from Yujin Robotics/Philips. Furthermore, visual SLAM systems have been developed for low-cost and embedded systems [65.17, 65.18].

65.2.2.3 cv-SLAM

The ceiling vision SLAM [65.14] system is composed of a camera facing upward, looking toward the ceiling. The system extracts corner features using a Harris detector and matches features using correlation. Feature matching achieves invariance to view point changes by training a set of multiview descriptors of the features as matching proceeds during the run. The main orientation(s) of the feature are further used as descriptors of the feature to ensure a two-dimensional (

2-D

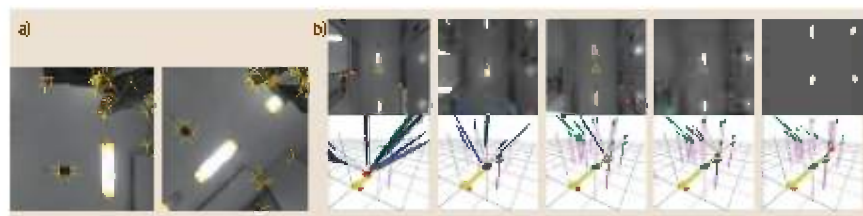
) rotation invariant feature matching that can be used for relocalization. The localization and mapping backend of the system is based on an extended Kalman filter (EKF) that fuses visual and dead-reckoning information (odometry and gyro) and that tracks the pose of the robot in a two-and-a-half-dimensional (

2.5-D

) (x, y, θ) and the 3-D position of the features (landmarks of the map). The orientation(s) of the features are also represented in 3-D and tracked with the EKF to predict changes in the feature patch due to motion of the robot (as the robot moves, features on the ceiling will undergo a rotation while features on the walls will experience a shearing transformation in addition to rotation).

Figure [65.19a](#) shows corner points and their estimated orientations in views with rotation only. Figure [65.19b](#) presents snapshots from a sequential map-building experiment on a corridor.

Fig. 65.19



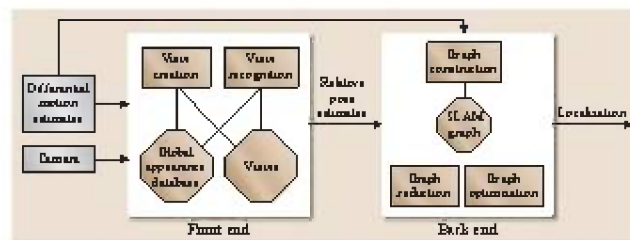
(a) cv-SLAM (after [\[65.14\]](#)). (b) Features extracted by the vision front-end. Map obtained with the EKF

65.2.2.4 vSLAM

The vSLAM [\[65.17, 65.18\]](#) system from Evolution Robotics is designed for a low-cost robotic platform equipped with simple odometry and a single camera. Figure [65.20](#) shows the block diagram of the system. Operating primarily as a view recognition engine, the visual measurement front-end requires only occasional, weak assumptions on processing rate, and intrinsically provides robust loop closing when previously mapped areas are revisited. The

visual measurements and odometry are fused in the back-end in a graph representation and optimized incrementally. The SLAM graph complexity is bounded during operation using variable elimination and constraint pruning, with heuristic schedules in order to keep optimization and storage costs commensurate with explored area, rather than with time of exploration, all while causing minimal loss in mapping and localization accuracy. The system has been evaluated on real datasets with planar ground-truth reference, showing that the system operates successfully even at frame rates below 2 Hz, running on an ARM9 processor with 64 MB of RAM [65.18].

Fig. 65.20



vSLAM block diagram

View recognition engines [65.17, 65.18, 65.19, 65.20, 65.21, 65.22] have proven attractive components for SLAM systems because they permit robust and flexible loop closing. Instead of making correspondences between individual features or measurements as in the cv-SLAM system, view recognition engines typically match constellations of features or entire images, without requiring tracking. vSLAM [65.17] creates views by first matching SIFT [65.23] features on pairs of incoming images, and second computing structure and motions estimates using bundle adjustment. The SIFT features are stored in a local database for each view and on a global database for the complete map. View recognition is performed by a feature lookup in the global database that provides a set of candidate view matches. Feature lookup in the local view database followed by robust outlier rejection and a local bundle adjustment completes the view recognition and visual pose estimation process.

Commonly used graph SLAM methods for agglomerating sensor information often incur computation and storage costs that grow with time, rather than with space explored. For

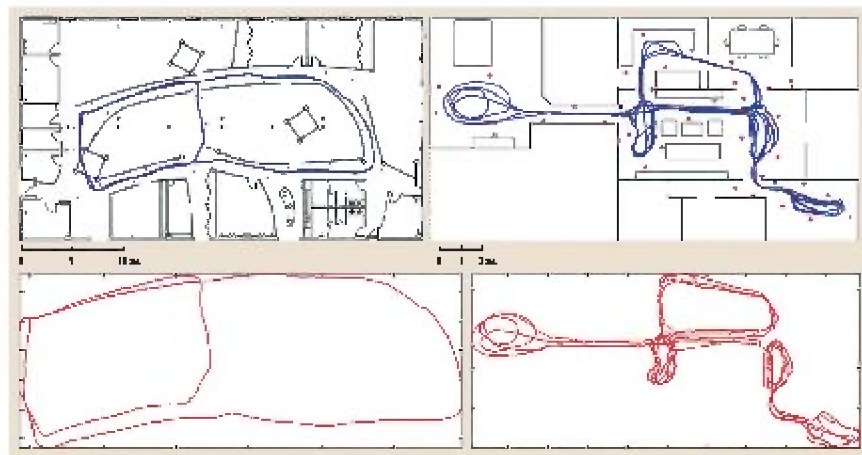
a robot operating for extended periods within a limited spatial area – typical of practical applications – this is an undesirable trade-off. vSLAM [65.18] instead applies probabilistically sound graph reduction methods that limit the complexity of the graph to a linear factor of the complexity of the explored space. Past poses of the robot that are not used for view recognition can be marginalized out of the estimation, and their incident constraints are collapsed back into the graph.

Figure 65.21 presents the results of running the vSLAM system in two sequences, one collected in a regular household (*right*) (same house as the one shown in Fig. 65.18 for vector field SLAM) and the other collected in a large warehouse environment (*left*). The first set of plots show the ground truth path of the robot and the floor plan of the environments. The second set of plots shows the trajectory of the robot estimated with vSLAM. The warehouse environment extends over an area of 24×12 m and the house over an area of 20×9 m. The estimated trajectory has a root mean square (

RMS

) error of 44 cm in the warehouse case and of 28 cm in the household case.

Fig. 65.21



vSLAM results on a warehouse environment and in a household environment

65.2.2.5 Laser-Based SLAM

The previous sections presented a number of low-cost solutions to the SLAM problem in which the data association was (almost) perfect. In the case of Vector-Field SLAM, the modulation of the spots of the Northstar cube ensures a unique identification of the spot. In the case of visual SLAM, the visual front-end incorporates a number of checks to ensure that misrecognitions are very rare. Laser-based SLAM has the characteristic that the measurements acquired with the laser are not unique since it is possible to obtain similar measurements in a variety of places in a household (just aiming the laser to a wall will provide measurements that are indistinguishable from a measurement to a different wall). In addition to the data association problem, laser distance measurement sensors, such as the one presented in Sect. [65.2.1](#), provide both range and bearing to the landmarks. Both Northstar and cameras give only bearing measurements to the spots or features.

The laser-based SLAM literature is quite extensive [[65.25](#)]. The estimation back-end could be an extended Kalman filter (

EKF

), a Particle Filter, or a GraphSLAM system. Several data association algorithms are available, some of them are proactive (or even greedy) [[65.26](#), [65.27](#)] and some are lazy [[65.28](#)] in assigning correspondences between measurements and landmarks. The selection of the algorithm to implement in the domestic robot would be guided by the trade-off between computational resources and the performance requirements. Figure [65.22](#) presents a map obtained with the Neato XV-11 robot.

Fig. 65.22

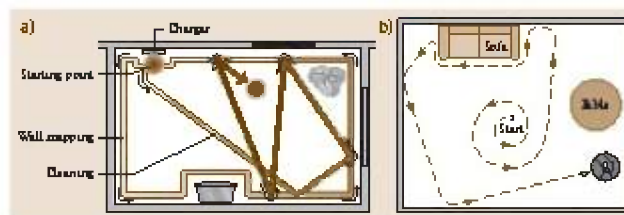


Map created with the Neato XV-11 robot (www.youtube.com/watch?v=zodC8EFvh7g)

65.2.3 Navigation and Coverage

The Trilobite robot vacuum cleaner by Electrolux was one of the pioneer robots to be commercialized. The trilobite was equipped with a sophisticated custom sonar sensor system that allowed the robot to navigate without touching obstacles (or touching them very gently). The coverage strategy consisted of two stages: first, explore the perimeter of the room to estimate the area to clean and second, cover the room with wall-to-wall diagonal passes. The perimeter exploration stage assumes that the robot would eventually traverse through the complete boundary of the area to clean and return to the charging station (Fig. 65.23a).

Fig. 65.23a,b



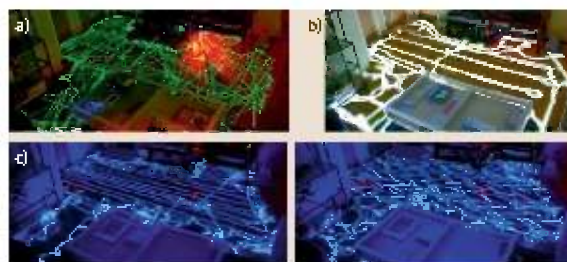
Coverage strategies. Trilobite strategy. Roomba strategy [65.24]

As mentioned previously, the cost constraints forced early robots like the Roomba to eschew advanced features like localization and mapping in order to offer a product at a price point accepted by consumers. Nevertheless, the navigation strategy implemented in the Roomba was quite effective in covering the space, especially when running in single rooms or in presence of large amount of clutter. The Roomba uses a spiral pattern in open area to optimally cover the space without localization until it hits an obstacle (Fig. 65.23b). Then it selects random orientation and continues moving in straight line until reaching the next obstacle where it selects another random orientation to continue with the same behavior. The spiral can be triggered when the robot travels for a certain distance without having

found any obstacles. Many of the other random robots present in the market have a strategy that follows similar principles to the Roomba one.

The systematic robots on the other hand take advantage of the localization system to plan efficient cleaning paths to maximize coverage in the least amount of time, pause for charging on the docking station and resume cleaning from the last cleaned spot, and intelligently navigate from room to room. Mint takes the strategy of focusing first on open area cleaning, covering the space in parallel straight passes to traverse as much open space as fast as possible, and then performing a final cleaning on the perimeter and around the obstacles. This perimeter cleaning step enables Mint to uncover portals to new areas that were not encountered during open area cleaning. Other robots like the Navibot from Samsung, the Roboking from LG and the iClebo from Yujin/Philips use a similar strategy without the final perimeter cleaning. The Neato XV-11, on the other hand, performs first an exploration and cleaning of the perimeter of the environment in order to create a good localization map and then completes the coverage of the open area. Figure 65.24 shows long exposure pictures of different robots: Roomba, Neato, Mint sweeping and Mint mopping. These long exposure pictures show the trajectory of the robots during normal cleaning operation.

Fig. 65.24a–c



Long exposure images [65.29] showing the coverage strategies of the (a) Roomba, (b) Neato, and (c) Mint

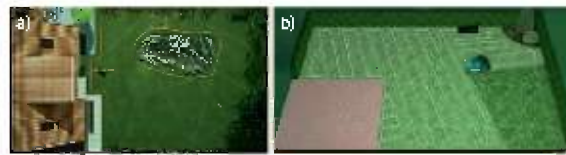
The lawnmower robots use similar coverage strategies to the ones of vacuum cleaner robots. One element common to all lawnmower robots is the usage of an embedded wire to define

the perimeter of the lawn. Within the boundaries defined by the wire, some robots like the Automower from Husqvarna or the Robomow from Friendly Robotics use a complete random coverage strategy. The Tango from John Deere combines straight motions in random directions with spiral motions (a-la-Roomba). The Indego from Bosch is the only robot that attempts to systematically cover the space by first exploring the perimeter to estimate the size of the lawn and then traversing the interior with parallel straight passes. Systematic coverage is enabled by fusing sensory information from wheel odometry and an inertial measurement unit (

IMU

) by means of probabilistic reasoning with prior knowledge. Figure [65.25](#) depicts the coverage strategies of the Tango and the Indego.

Fig. 65.25a,b



Coverage strategies for lawnmowers. (a) Tango [\[65.30\]](#) by John Deere and (b) Indego [\[65.31\]](#) by Bosch

65.3 Smart Homes

Several attempts have been made in the literature to define the term smart home, for example, in [\[65.32\]](#) the term is defined as *the latest expression of the various ways in which technology in the home has developed*. In [\[65.33\]](#), the notion of a smart home is defined more explicitly as:

a residence equipped with computing and information technology which anticipates and responds to the needs of the occupants, working to promote their comfort, convenience,

security and entertainment through management of technology within the home and connections to the world beyond.

Smart homes typically comprise elements such as network of sensors and actuators, and also entire robotic systems.

In 1999, a company called e2 Home was established by Ericsson and Electrolux to explore the possibilities of smart homes [65.34]. E2 Home built several houses with smart home devices using IT technologies. Through the exploration, several issues relating to smart homes from business point of view have been unveiled, which include the difficulties relating to the complexity of the system, starting up the new business model, handling of intellectual property rights (

IPR

) for the third parties contents, and the consumer's needs. The company was liquidated in 2004.

Current smart home technology includes video monitoring, motion detectors, fall detectors, pressure mats, environment control, health monitoring such as blood pressure, pulse rate, body temperature, weight, and human computer interaction (

HCI

) technology, for example, to recognize gestures. Smart homes also often refer to houses connected to a smart grid, which is defined as [65.35]:

a developing network of new technologies, equipment, and controls working together to respond immediately to our 21st century demand for electricity.

In this case, the smart home is defined as [65.35]:

a residence with the capability of efficiently controlling generated solar energy and power consumption making it ideal for vehicle power supply and management.

The development of smart homes requires a number of technical questions and challenges to be addressed [65.32]: how to convert current home structures and architectures into smart homes, how to standardize smart home components, for example, sensor networks, how to keep the equipment cost at a reasonable level, and how to deal with security and privacy issues.

In the following sections we will describe a number of prominent smart home developments, the Aware Home at Georgia Institute of Technology, the Gator Tech Smart House at the University of Florida, and the sensorized environment for life (

SELF

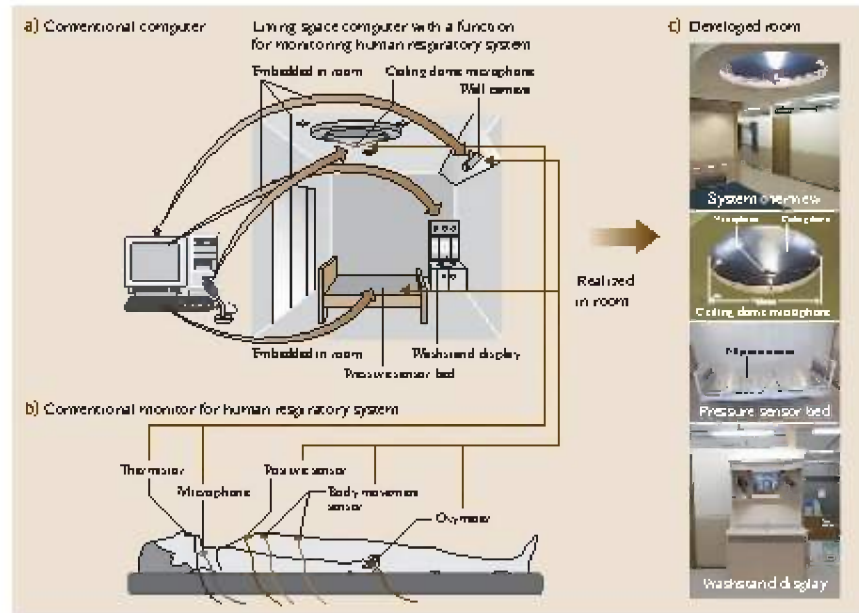
) at

AIST

(the Japan National Institute of Advanced Industrial Science and Technology) (Figs. 65.26, 65.27). These developments show that current smart home technology goes significantly beyond existing home automation. Accounting for the significant increase in the elderly population in the near future, many of today's smart home developments pay special attention to improving the quality of life of elderly people. The descriptions below will also address the question on how to integrate robotics systems into smart home concepts.

Fig. 65.26



Fig. 65.27

AIST, the sensorized environment for life

65.3.1 Gator Tech Smart House

The Gator Tech Smart House (Fig. 65.26) was built at the University of Florida [65.36]. It addresses the needs of elderly people to live independently and maintain dignity and quality of life at older ages. The house is equipped with many smart devices such as smart floors, tracking the motion of the occupants of the house, smart blinds, automatically adjusting ambient light, smart display, smart cameras, smart phones that can act as a remote control to other appliances, location tracking, smart leak detectors, and smart beds. The exterior of the house has a smart mailbox which alerts residence if mail is delivered and a smart front door which can sense home owners using an radio frequency identification (

RFID

) tag, allowing keyless entry to home owners.

The kitchen of the Gator Tech Smart House includes a smart microwave, which uses RFID on food packages. This allows the microwave to adjust the settings for cooking the meal. It also informs the resident about the readiness of the meal. The kitchen further comprises a smart refrigerator that monitors food availability and consumption, and detects expired food items. The smart refrigerator can create shopping lists automatically and has an integrated meal preparation advisor based on items in the refrigerator and pantry.

The implementation of such a complex system has raised a number of technical issues and questions, including the development of the smart devices, data handling of networks of sensors, and interconnecting smart devices to other devices in the environment. These questions led to some new research tracks on smart houses, primarily grouped into pervasive computing and mobile computing network research.

65.3.2 Aware Home

Aware Home is a living laboratory for research in ubiquitous computing for everyday activities. This project is conducted at Georgia Institute of Technology [65.37]. The major objective of the Aware Home project is to build an environment that is capable of being aware and keeping track of the states and activities of its inhabitants. Aware Home creates a partnership between the resident and the surrounding sensing and computing technologies. This opens several fields of research, not only from the technology point of view, but also in terms of the social aspects of the inhabitants. The main research agenda of Aware Home spans human-centered and technology-centered research, software engineering, and social implications.

Technology and application-centered research focuses on sensor networks, distributed computing, context awareness and ubiquitous sensing, individual interaction with the home, smart floors, and finding lost objects. Research on context awareness is inspired by the fact that humans communicate with each other very successfully by referring to what is called shared context. For communication between humans and computer systems, this shared context must be made explicitly. Sensor systems, which facilitate the extraction of context, need to be developed.

This human-centered research focuses on support for the elderly and other social issues. A key concept in supporting the elderly is *aging in place*. Aware Home is designed to support the elderly and allow them to be independent instead of moving to elderly care facilities. Supporting the elderly leads to the study on cognitive support such as reminding them when to take medication, guiding them when they lose their way, and locating lost items.

65.3.3 SELF – Sensorized Environment for LiFe

SELF stands for Sensorized Environment for LiFe [65.1]. The objectives of SELF are to develop a network of sensors which are embedded in the environment, information gathering using the networked sensors, storing and analyzing the information, and the reporting of useful information to assist and maintain good health. The basic advantages of SELF, due to the embedded nature of the sensors are:

1. No limit on size, weight, or power source,
2. It does not disturb the human,
3. It does not impose physical restrictions, and
4. Sensors are rarely broken since they are fixed to the environment.

SELF can be viewed as a system that monitors a person's behavior or activity and represents the data objectively in an approach known as self-externalization. SELF is motivated by the fact that humans sometimes cannot notice a change in their condition which affects their health without a medical doctor. Therefore, the use of network sensors to monitor human behavior and report useful information that greatly affects the health status will further improve quality of life.

The SELF study considers behavior as a means of communication, and sensors embedded into the environment as one way to observe a person's behavior. The SELF implementation consists of a bed with a sensor, a ceiling with microphones, a washstand with a display, etc. The bed with sensors can determine the time the subject sleeps and wakes up, their posture during sleeping, and their breathing pattern. The microphones attached to the ceiling can detect snoring or normal breathing sounds. Based on the monitored data, the washstand

display is used as an output device to provide the subject's health status and thus create feedback to the subject.

Abbreviations

2-D: two-dimensional

2.5-D: two-and-a-half-dimensional

3-D: three-dimensional

AIST: Japan National Institute of Advanced Industrial Science and Technology

B2B: business to business

BOM: bill of material

CES: Consumer Electronics Show

CMOS: complementary metal-oxide-semiconductor

CPU: central processing unit

cv-SLAM: ceiling vision SLAM

EKF: extended Kalman filter

FDA: US Food and Drug Association

GPS: global positioning system

HCI: human-computer interaction

HD: high definition

IFA: Internationale Funk Ausstellung

IMU: inertial measurement unit

IPA: Institute for Manufacturing Engineering and Automation

IPR: intellectual property right

IR: infrared

LED: light-emitting diode

PSD: position-sensitive-device

RAM: random access memory

RFID: radio frequency identification

RMS: root mean square

RP-VITA: remote presence virtual + independent telemedicine assistant

RPS: room positioning system

SELF: sensorized environment for life

SLAM: simultaneous localization and mapping

TOF: time-of-flight

References

65.1 P. Ljunggren: Intelligent machines, Personal Communication

[Google Scholar](#)

65.2 R.D. Schraft, M. Hägele, K. Wegener: *Service-Roboter-Visionen* (Hanser, Munich 2004)

[Google Scholar](#)

65.3 A. Albert: Robert-Bosch GmbH, Personal Communication

[Google Scholar](#)

65.4 Howstuffworks: <http://electronics.howstuffworks.com/gadgets/home/robotic-vacuum2.htm>

65.5 K. Konolige, J. Augenbraun, N. Donaldson, C. Fiebig, P. Shah: A low-cost laser distance sensor, Proc. Int. Conf. Robotics Autom. (ICRA), Pasadena (2008)

[Google Scholar](#)

65.6 Gearfuse: <http://www.gearfuse.com/robotic-vacuum-paths-mapped-and-compared-with-long-exposure-pictures/>

65.7 P. Ridden: Joggobot turns a quadrocopter into a running companion, Gizmag, June 11 <http://www.gizmag.com/joggobot-autonomous-quadrocopter-running-partner/22899/> (2012)

65.8 B. Freedman, A. Shpunt, J. Arieli: Distance-varying illumination and imaging techniques for depth mapping, US Patent 0290698 (2010)

[Google Scholar](#)

65.9 A. Shpunt, A. Zlesky: Depth-varying light fields for three dimensional sensing, US Patent 2008/0106746 (2008)

[Google Scholar](#)

65.10 Microsoft (Xbox): <http://www.xbox.com/en-US/KINECT>

65.11 Apple (Primesense): <http://www.primesense.com/>

65.12 J.-S. Gutmann, K. Culp, M. Munich, P. Pirjanian: The social impact of a systematic floor cleaner, ARSO Munich (2012)

[Google Scholar](#)

65.13 J.-S. Gutmann, E. Eade, P. Fong, M. Munich: Vector field SLAM – localization by learning the spatial variation of continuous signals, Trans. Robotics 28(3), 650–667 (2012)

[Article](#) [Google Scholar](#)

65.14 W. Jeong, K. Lee: CV-SLAM: A new ceiling vision-based SLAM technique, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2005)

[Google Scholar](#)

65.15 J.-S. Gutmann, D. Goel, M. Munich: Scaling vector field SLAM to large environments, Proc. IAS, Jeju (2012)

- 65.16** J.-S. Gutmann, P. Fong, M. Munich: Localization in a vector field map, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2012)

- 65.17** N. Karlsson, E.D. Bernardo, J. Ostrowski, L. Goncalves, P. Pirjanian, M.E. Munich: The vSLAM algorithm for robust localization and mapping, Proc. Int. Conf. Robotics Autom. (ICRA) (2005)

- 65.18** E. Eade, P. Fong, M. Munich: Monocular graph SLAM with complexity reduction, IROS (2010)

- 65.19** M. Cummins, P. Newman: Accelerated appearance-only SLAM, Proc. Int. Conf. Robotics Autom. (ICRA), Pasadena (2008)

- 65.20** E. Eade, T. Drummond: Monocular slam as a graph of coalesced observations, Proc. 11th IEEE Int. Conf. Comput. Vis. (ICCV'07), Rio de Janeiro (2007)

- 65.21** E. Eade, T. Drummond: Unified loop closing and recovery for real time monocular slam, Proc. Br. Mach. Vis. Conf. (BMVC'08) Leeds (2008) pp. 53–62

[Google Scholar](#)

- 65.22 K. Konolige, J. Bowman, J.D. Chen, P. Mihelich, M. Calonder, V. Lepetit, P. Fua: View-based maps, Proc. Robotics Sci. Syst. Seattle (2009)

[Google Scholar](#)

- 65.23 D.G. Lowe: Distinctive image features from scale-invariant keypoints, Proceedings IJCV (2004)

[Google Scholar](#)

- 65.24 Pmdtechnologies GmbH: <http://www.pmdtec.com/>

- 65.25 S. Thrun, W. Burgard, D. Fox: *Probabilistic Robotics* (MIT Press, Cambridge 2005)

[MATH](#) [Google Scholar](#)

- 65.26 J. Neira, J. D. Tard os: Data association in stochastic mapping using the joint compatibility test, IEEE Trans. Robotics Autom. 17(6), 890–897 (2001)

[Article](#) [Google Scholar](#)

- 65.27 S. Thrun, D. Fox, W. Burgard: A probabilistic approach to concurrent mapping and localization for mobile robots, Mach. Learn. 31, 29–53 (1998)

[Article](#) [Google Scholar](#)

- 65.28 D. Hähnel, S. Thrun, B. Wegbreit, W. Burgard: Towards lazy data association in SLAM, Proc. 11th Int. Symp. Robotics Res. (ISRR), Sienna (2003) pp. 421–431

65.29 John Deere:

http://www.deere.com/wps/dcom/en_INT/products/equipment/autonomous_mower/autonomous_mower.page

65.30 Robert Bosch GmbH: <http://www.bosch-indego.com/>

65.31 E. Prassler, K. Kosuge: Domestic robotics. In: *Springer Handbook of Robotics*, ed. by B. Siciliano, O. Khatib (Springer, Berlin, Heidelberg 2008) pp. 1253–1281

[Chapter](#) [Google Scholar](#)

65.32 D. Gann, J. Barlow, T. Venables: *Digital Futures: Making Homes Smarter. Report published by Chartered Institute of Housing* (Chartered Institute of Housing, Coventry 1999)

[Google Scholar](#)

65.33 R. Harper: *Inside the Smart Home* (Springer, London 2003)

[Book](#) [Google Scholar](#)

65.34 Japan Electronics and Information Technology Industries Association:
<http://www.eclipse-jp.com/jeita/>

65.35 Gator Tech Smart House: <http://www.icta.ufl.edu/gt.htm>

65.36 Aware Home Research Initiative: <http://www.awarehome.gatech.edu/>

65.37 T. Hori, Y. Nishida, T. Suehiro, S. Hirai: SELF-Network: Design and implementation of network for distributed embedded sensors, IEEE/RSJ Int. Conf. Intell. Robots Syst. IROS (2000)

[Google Scholar](#)

Author information

Authors and Affiliations

Department of Computer Sciences, Bonn-Rhein-Sieg Univ. of Applied Sciences,
Grantham-Allee 20, 53754, Sankt Augustin, Germany

Erwin Prassler

iRobot Corp., 1055 East Colorado Boulevard, Suite 340, CA 91106, Pasadena, USA

Mario E. Munich

iRobot Corp., 8 Crosby Drive, MA 01730, Bedford, USA

Paolo Pirjanian

System Robotics Laboratory, Tohoku University, Aoba 6-6-01, Aramaki, 980-8579,
Sendai, Japan

Kazuhiro Kosuge

Corresponding author

Correspondence to [Erwin Prassler](#).

Editor information

Editors and Affiliations

Department of Electrical Engineering, University of Naples Federico II, Via Claudio 21,
80125, Naples, Italy

Bruno Siciliano

Department of Computer Sciences, Artificial Intelligence Laboratory, Stanford University,
450 Serra Mall, CA 94305, Stanford, USA

Oussama Khatib

Video-References

 **VIDEO 727**: Robotic Vacuum Cleaners Reviewed by Click – Spring 2014 available from
<http://handbookofrobotics.org/view-chapter/65/videodetails/727>

 **VIDEO 729**: How would you choose the best Robotic Vacuum Cleaner? available from
<http://handbookofrobotics.org/view-chapter/65/videodetails/729>

 **VIDEO 731**: Husqvarna Automower versus competitors available from
<http://handbookofrobotics.org/view-chapter/65/videodetails/731>

 **VIDEO 734**: Windoro window cleaning robot review available from
<http://handbookofrobotics.org/view-chapter/65/videodetails/734>

 **VIDEO 735**: WINBOT W710 versus HOBOT 168 available from
<http://handbookofrobotics.org/view-chapter/65/videodetails/735>

 **VIDEO 736**: Winbot window cleaning robot available from
<http://handbookofrobotics.org/view-chapter/65/videodetails/736>

 **VIDEO 737**: Serbot Robot Clean Ant Profi available from
<http://handbookofrobotics.org/view-chapter/65/videodetails/737>

 **VIDEO 739**: Home Pool Cleaner Review – 5 types of robotic cleaners available from
<http://handbookofrobotics.org/view-chapter/65/videodetails/739>

 **VIDEO 740**: Automatic pool cleaner reviews available from
<http://handbookofrobotics.org/view-chapter/65/videodetails/740>

 VIDEO 741 : Telepresence robot in action available from <http://handbookofrobotics.org/view-chapter/65/videodetails/741>

 VIDEO 742 : Double robotics – overview available from <http://handbookofrobotics.org/view-chapter/65/videodetails/742>

 VIDEO 744 : Test-Driving Beam, the telepresence robot available from <http://handbookofrobotics.org/view-chapter/65/videodetails/744>

 VIDEO 745 : Beam's new Palo Alto store lets telepresence robots sell themselves. Literally available from <http://handbookofrobotics.org/view-chapter/65/videodetails/745>

 VIDEO 746 : This robot is your running coach – Joggerbot available from <http://handbookofrobotics.org/view-chapter/65/videodetails/746>

 VIDEO 747 : RUFUS – your personal running coach available from <http://handbookofrobotics.org/view-chapter/65/videodetails/747>

 VIDEO 748 : PhillieBot robot gives first pitch at a Phillies game available from <http://handbookofrobotics.org/view-chapter/65/videodetails/748>

Rights and permissions

[Reprints and permissions](#)

Copyright information

© 2016 Springer-Verlag Berlin Heidelberg

About this chapter

Cite this chapter

Prassler, E., Munich, M.E., Pirjanian, P., Kosuge, K. (2016). Domestic Robotics. In: Siciliano, B., Khatib, O. (eds) Springer Handbook of Robotics. Springer Handbooks. Springer, Cham.
https://doi.org/10.1007/978-3-319-32552-1_65

[.RIS](#) [.ENW](#) [.BIB](#)

DOI	Published	Publisher Name
https://doi.org/10.1007/978-3-319-32552-1_65	27 July 2016	Springer, Cham
Print ISBN	Online ISBN	eBook Packages
978-3-319-32550-7	978-3-319-32552-1	Engineering
		Engineering (RO)

Share this chapter

Anyone you share the following link with will be able to read this content:

[Get shareable link](#)

Provided by the Springer Nature SharedIt content-sharing initiative

Keywords

[Inertial Measurement Unit](#)

[Smart Home](#)

[Docking Station](#)

[Cleaning Robot](#)

[Domestic Robot](#)

These keywords were added by machine and not by the authors. This process is experimental and the keywords may be updated as the learning algorithm improves.

Publish with us

[Policies and ethics](#) [↗](#)

Attachment O

HomeToolsOn-line path plan...Jim2025Doc3ExM... x

☆↻🖨🔍

⬆⬇1729 (1 of 30)⬆⬇

🔍👤⌂⊕202%🔍📄🗑🔄

This file claims compliance with the PDF/A standard and has been opened read-only to prevent modification.

Enable Editing

Bookmarks

65 Domestic Robotics

65.1 Mobile Domestic Robotics

65.2 Enabling Technologies

65.3 Smart Homes

Video-References

References

Multimedia Contents



Erwin Prassler, Mario E. Mur

When the first edition of this book was published, most types of domestic robots were spoken of as a dream that was slowly becoming reality. At that time, in 2008, we looked back on more than twenty years of research and development in domestic robotics, especially in cleaning robotics. Although everybody expected cleaning to be the killer app for domestic robotics in the first half of these twenty years nothing big really happened. About ten years

prototypes or concept studies. Others have already crossed the threshold to becoming commercial products.

For the second edition of this chapter, we have decided to not only enumerate the devices that have emerged and survived in the past five years, but also to take a look back at how it all began, rasting this retrospection with the burst of

Document Properties

DescriptionSecurityFontsInitial ViewCustomAdvanced

Description

File: Jim2025Doc3ExMSpringerPDF.pdf

Title:

Author:

Subject:

Keywords:

Created: 7/21/16, 7:39:11 AM

Modified: 10/16/25, 5:55:41 PM

Application: pdfsam-console (Ver. 2.4.1e)

Additional Metadata...

Advanced

PDF Producer: Acrobat Distiller 11.0 (Windows)

PDF Version: 1.4 (Acrobat 5.x)

Location: /Users/Mary/Documents/consult/Jim2025 /

File Size: 5.39 MB (5,650,841 Bytes)

Page Size: 7.60 x 10.24 in

Number of Pages: 30

Tagged PDF: No

Fast Web View: Yes

HelpCancelOK

Lightshot Screenshot

Part F | 6

Avidbots Ex1008
Page 540 of 545

Attachment P

Bibliographic Record - OCLC Number: 954461883

LDR	Record Status	'c' - Corrected or revised
	Type of Record	'a' - Language material
	Bibliographic Level	'm' - Monograph/Item
	Type of Control	' ' - No specified type of control
	Encoding Level	' ' - Full level
	Descriptive Cataloging Form	'i' - ISBD punctuation included
001	954461883	
005	20250804215546.9	
008	Date Entered on File	'160802'
	Type of Date/Publication Status	's' - Single date
	Date1/Beginning Date of Publication	'2016'
	Date2/Ending Date of Publication	' '
	Place of Publication, Production, or Execution Code	'gw ' - Germany
	Illustrations	'a '
	Target Audience	' ' - Unknown or unspecified
	Form of Item	'o' - Online
	Nature of Contents Code	'bf '
	Government Publication	' ' - Not a government publication
	Conference Publication	'0' - Not a conference publication
	Festschrift	'0' - Not a festschrift
	Index	'1' - Index present
	Literary Form	'0' - Not fiction
	Biography	' ' - No biographical material
	Language Code	'eng' - English
	Modified Record	' ' - Record is not modified
	Cataloging Source Code	'd' - Other
006	Form of Material	'm' - Computer file
	Target Audience	' ' - Unknown or unspecified
	Form of Item	'o' - Online
	Type of Computer File	'd' - Document
	Government Publication	' ' - Not a government publication
007	Category of Resource	'c' - Electronic Resource
	Specific Material Designation (c)	'r' - Remote
	Color (c)	'c' - Multicolored
	Dimensions (c)	'n' - Not applicable
	Sound (c)	'u' - Unknown
	Image Bit Depth (c)	'----
	File Formats (c)	'u' - Unknown
	Quality Assurance Target(s) (c)	'n' - Not applicable
	Antecedent/Source (c)	'u' - Unknown
	Level of Compression (c)	'u' - Unknown
	Reformatting Quality (c)	'u' - Unknown

040 GW5XE \$b eng \$e rda \$e pn \$c GW5XE \$d YDXCP \$d AZU \$d COO \$d VT2 \$d MUB \$d S
OI \$d NST \$d ISM \$d OCLCF \$d QGK \$d UAB \$d IAD \$d JBG \$d ICW \$d Z5A \$d ILO \$d
ICN \$d EBLCP \$d ESU \$d CSAIL \$d IOG \$d REB \$d U3W \$d OCLCQ \$d AU@ \$d OCLCQ
\$d WYU \$d UWO \$d UKMGB \$d AUD \$d UWW \$d UX1 \$d OCLCQ \$d DCT \$d ERF \$d U
KBTH \$d LEATE \$d OCLCQ \$d U9X \$d UBY \$d OCLCQ \$d OCLCO \$d COM \$d OCLCO
\$d OCLCQ \$d OCLCO \$d OCLCL \$d OCLCQ \$d UKKRT \$d OCLCL

015 GBB8K6642 \$2 bnb

016 7 019105073 \$2 Uk

019 959235085 \$a 961871055 \$a 985042917 \$a 1005755666 \$a 1012062741 \$a 1036288222 \$a 1048
230911 \$a 1058224569 \$a 1066679801 \$a 1079209254 \$a 1086472846 \$a 1110968101 \$a 11125
81416 \$a 1113382365 \$a 1113754110 \$a 1122810025 \$a 1127188081 \$a 1136470448 \$a 1140745
464 \$a 1154172068 \$a 1237457558 \$a 1259253666 \$a 1480160416

020 9783319325521 \$q (electronic bk.)

020 3319325523 \$q (electronic bk.)

020 \$z 9783319325507 \$q (print)

020 \$z 3319325507

020 \$z 9783319325507

024 7 10.1007/978-3-319-32552-1 \$2 doi

024 8 99968724117

037 com.springer.onix.9783319325521 \$b Springer Nature

050 4 TJ211

072 7 TJFM \$2 bicssc

072 7 TJFD \$2 bicssc

072 7 TEC004000 \$2 bisacsh

072 7 TEC037000 \$2 bisacsh

072 7 TEC \$x 009000 \$2 bisacsh

072 7 TJFM \$2 thema

072 7 TJFD \$2 thema

082 0 4 629.8/92 \$2 23 \$1 <https://id.oclc.org/worldcat/ddc/E366dwMq7tD7fTMtMkt8K9cdJh>

090 \$b

049 LDLL

245 0 0 Springer handbook of robotics / \$c Bruno Siciliano, Oussama Khatib (eds.).

246 3 0 Handbook of robotics

250 2nd edition.

264 1 Berlin : \$b Springer, \$c 2016.

300 1 online resource (lxxvi, 2227 pages) : \$b illustrations (some color)

336 text \$b txt \$2 rdacontent

337 computer \$b c \$2 rdamedia

338 online resource \$b cr \$2 rdacarrier

347 text file

347 \$b PDF

490 1 Springer Handbooks, \$x 2522-8692

504 Includes bibliographical references and index.

588 0 Online resource; title from PDF title page (SpringerLink, viewed August 2, 2016).

505 0 Part A -- Robotics Foundations -- Part B -- Design -- Part C -- Sensing and Perception -- Part D -- Man
ipulation and Interfaces -- Part E -- Moving in the Environment -- Part F -- Robots at Work -- Part G --

Robots and Humans -- Acknowledgements -- About the Authors -- Subject Index.

520 The second edition of this handbook provides a state-of-the-art cover view on the various aspects in the rapidly developing field of robotics. Reaching for the human frontier, robotics is vigorously engaged in the growing challenges of new emerging domains. Interacting, exploring, and working with humans, the new generation of robots will increasingly touch people and their lives. The credible prospect of practical robots among humans is the result of the scientific endeavour of a half a century of robotic developments that established robotics as a modern scientific discipline. The ongoing vibrant expansion and strong growth of the field during the last decade has fueled this second edition of the Springer Handbook of Robotics. The first edition of the handbook soon became a landmark in robotics publishing and won the American Association of Publishers PROSE Award for Excellence in Physical Sciences & Mathematics as well as the organization's Award for Engineering & Technology. The second edition of the handbook, edited by two internationally renowned scientists with the support of an outstanding team of seven part editors and more than 200 authors, continues to be an authoritative reference for robotics researchers, newcomers to the field, and scholars from related disciplines. The contents have been restructured to achieve four main objectives: the enlargement of foundational topics for robotics, the enlightenment of design of various types of robotic systems, the extension of the treatment on robots moving in the environment, and the enrichment of advanced robotics applications. Further to an extensive update, fifteen new chapters have been introduced on emerging topics, and a new generation of authors have joined the handbook's team. A novel addition to the second edition is a comprehensive collection of multimedia references to more than 700 videos, which bring valuable insight into the contents. The videos can be viewed directly augmented into the text with a smartphone or tablet using a unique and specially designed app.

588 0 Print version record.

650 0 Robotics \$v Handbooks, manuals, etc.

650 6 Robotique \$v Guides, manuels, etc.

650 7 Artificial intelligence. \$2 bicsc

650 7 Tool making. \$2 bicsc

650 7 Technical design. \$2 bicsc

650 7 Automatic control engineering. \$2 bicsc

650 7 TECHNOLOGY & ENGINEERING \$x Engineering (General) \$2 bisacsh

650 7 Robotics \$2 fast

655 2 Handbook

655 7 handbooks. \$2 aat

655 7 Handbooks and manuals \$2 fast

655 7 Handbooks and manuals. \$2 lcgft

655 7 Guides et manuels. \$2 rvmgf

700 1 Siciliano, Bruno, \$d 1959- \$e editor. \$1 <https://id.oclc.org/worldcat/entity/E39PBjd3JpTC9Xphdmd9WYp9Dq>

700 1 Khatib, Oussama, \$e editor.

758 \$i has work: \$a Springer handbook of robotics (Text) \$1 <https://id.oclc.org/worldcat/entity/E39PCFQvkJrch8qpXCRpxyGg83> \$4 <https://id.oclc.org/worldcat/ontology/hasWork>

776 0 8 \$i Print version: \$t Springer handbook of robotics. \$b 2nd edition. \$d Berlin : Springer, 2016 \$z 9783319325507

830 0 Springer Handbooks, \$x 2522-8692

856 4 0 \$3 EBSCOhost \$u <https://search.ebscohost.com/login.aspx?direct=true&scope=site&db=nlebk&db=nlabk&AN=1260895>

856 4 0 \$3 ProQuest Ebook Central \$u <https://public.ebookcentral.proquest.com/choice/publicfullrecord.aspx?p=4614797>

856 4 0 \$3 SpringerLink \$u <https://doi.org/10.1007/978-3-319-32552-1>

856 4 0 **\$3** SpringerLink **\$u** <https://link.springer.com/book/10.1007%2F978-3-319-32550-7>
856 4 0 **\$3** SpringerLink **\$u** <https://link.springer.com/book/10.1007%2F978-3-319-32552-1>
856 4 0 **\$u** <https://sites.smu.edu/cul/apps/EBL/Default.aspx?id=4614797> **\$z** Connect to resource
856 4 **\$u** https://discover.gcu.ac.uk/discovery/openurl?institution=44GLCU_INST&vid=44GLCU_INST:44GLCU_VU2&?u.ignore_date_coverage=true&rft.mms_id=991002484373503836 **\$p** 5370974000003836 **\$x** StGIGCU
856 4 **\$u** https://openurl.ac.uk/ukfed:bat?u.ignore_date_coverage=true&rft.mms_id=991003561026602761 **\$p** 5345589550002761
856 4 0 **\$3** Kortext **\$u** <https://app.kortext.com/borrow/344568>
938 EBSCOhost **\$b** EBSC **\$n** 1260895
938 YBP Library Services **\$b** YANK **\$n** 13098417
938 Kortext **\$b** KTXT **\$n** 344568
029 1 AU@ **\$b** 000058477689
029 1 UKMGB **\$b** 019105073
029 1 DKDLA **\$b** 820120-katalog:999887638805765
029 1 AU@ **\$b** 000075317058