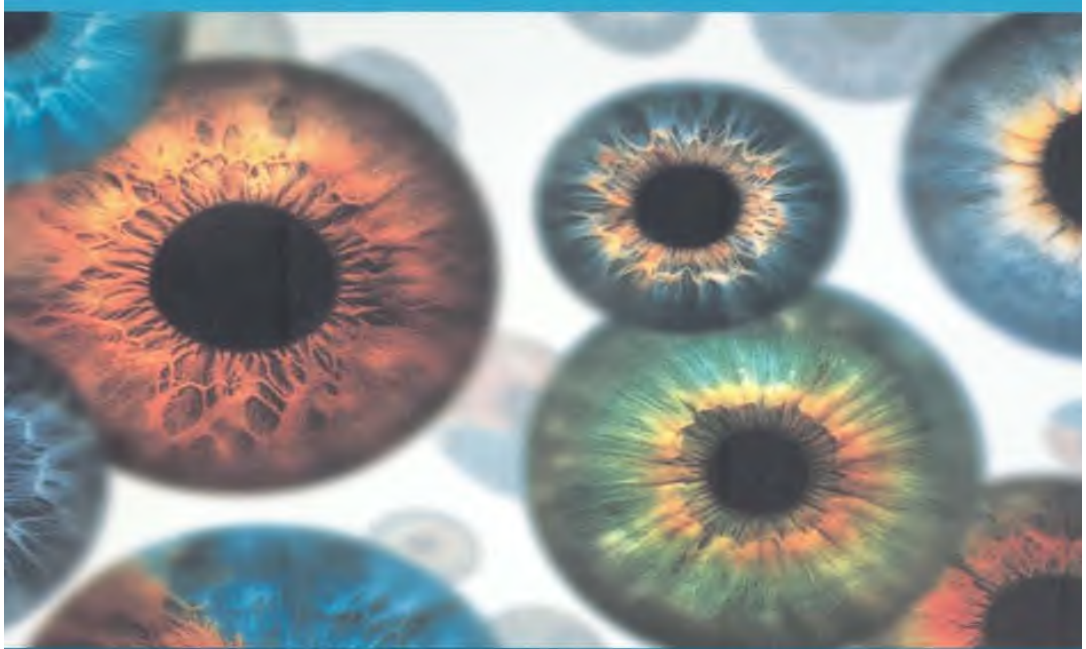
SECOND EDITION

Multiple View Geometry in computer vision



Richard Hartley and Andrew Zisserman

CAMBRIDGE

Samsung/Google, EX1010, p. 1

Multiple View Geometry in Computer Vision

Second Edition

Richard Hartley
Australian National University,
Canberra, Australia

Andrew Zisserman
University of Oxford, UK

CAMBRIDGE
UNIVERSITY PRESS

Samsung/Google, EX1010, p. 2

PUBLISHED BY THE PRESS SYNDICATE OF THE UNIVERSITY OF CAMBRIDGE
The Pitt Building, Trumpington Street, Cambridge, United Kingdom

CAMBRIDGE UNIVERSITY PRESS
The Edinburgh Building, Cambridge, CB2 2RU, UK
40 West 20th Street, New York, NY 10011-4211, USA
477 Williamstown Road, Port Melbourne, VIC 3207, Australia
Ruiz de Alarcón 13, 28014 Madrid, Spain
Dock House, The Waterfront, Cape Town 8001, South Africa

http://www.cambridge.org

© Cambridge University Press 2000, 2003

This book is in copyright. Subject to statutory exception
and to the provisions of relevant collective licensing agreements,
no reproduction of any part may take place without the
written permission of Cambridge University Press.

First Published 2000
Reprinted 2001, 2002
Second Edition 2003
Reprinted 2004

Printed in the United Kingdom at the University Press, Cambridge

A catalogue record for this book is available from the British Library

Library of Congress Cataloguing in Publication data

ISBN 0521 54051 8 hardback

General Library System
University of Wisconsin - Madison
728 State Street
Madison, WI 53706-1494
U.S.A.

Dedication

This book is dedicated to Joe Mundy whose vision and constant search for new ideas led us into this field.

Contents

Foreword *page* xi
Preface xiii

1 Introduction – a Tour of Multiple View Geometry 1
1.1 Introduction – the ubiquitous projective geometry 1
1.2 Camera projections 6
1.3 Reconstruction from more than one view 10
1.4 Three-view geometry 12
1.5 Four view geometry and n-view reconstruction 13
1.6 Transfer 14
1.7 Euclidean reconstruction 16
1.8 Auto-calibration 17
1.9 The reward I : 3D graphical models 18
1.10 The reward II: video augmentation 19

PART 0: The Background: Projective Geometry, Transformations and Estimation 23
Outline 24

2 Projective Geometry and Transformations of 2D 25
2.1 Planar geometry 25
2.2 The 2D projective plane 26
2.3 Projective transformations 32
2.4 A hierarchy of transformations 37
2.5 The projective geometry of 1D 44
2.6 Topology of the projective plane 46
2.7 Recovery of affine and metric properties from images 47
2.8 More properties of conics 58
2.9 Fixed points and lines 61
2.10 Closure 62

3 Projective Geometry and Transformations of 3D 65
3.1 Points and projective transformations 65
3.2 Representing and transforming planes, lines and quadrics 66

Samsung/Google, EX1010, p. 5

3.3 Twisted cubics 75
3.4 The hierarchy of transformations 77
3.5 The plane at infinity 79
3.6 The absolute conic 81
3.7 The absolute dual quadric 83
3.8 Closure 85

4 Estimation – 2D Projective Transformations 87
4.1 The Direct Linear Transformation (DLT) algorithm 88
4.2 Different cost functions 93
4.3 Statistical cost functions and Maximum Likelihood estimation 102
4.4 Transformation invariance and normalization 104
4.5 Iterative minimization methods 110
4.6 Experimental comparison of the algorithms 115
4.7 Robust estimation 116
4.8 Automatic computation of a homography 123
4.9 Closure 127

5 Algorithm Evaluation and Error Analysis 132
5.1 Bounds on performance 132
5.2 Covariance of the estimated transformation 138
5.3 Monte Carlo estimation of covariance 149
5.4 Closure 150

PART I: Camera Geometry and Single View Geometry 151
Outline 152

6 Camera Models 153
6.1 Finite cameras 153
6.2 The projective camera 158
6.3 Cameras at infinity 166
6.4 Other camera models 174
6.5 Closure 176

7 Computation of the Camera Matrix P 178
7.1 Basic equations 178
7.2 Geometric error 180
7.3 Restricted camera estimation 184
7.4 Radial distortion 189
7.5 Closure 193

8 More Single View Geometry 195
8.1 Action of a projective camera on planes, lines, and conics 195
8.2 Images of smooth surfaces 200
8.3 Action of a projective camera on quadrics 201
8.4 The importance of the camera centre 202
8.5 Camera calibration and the image of the absolute conic 208

Samsung/Google, EX1010, p. 6

8.6 Vanishing points and vanishing lines 213
8.7 Affine 3D measurements and reconstruction 220
8.8 Determining camera calibration K from a single view 223
8.9 Single view reconstruction 229
8.10 The calibrating conic 231
8.11 Closure 233

PART II: Two-View Geometry 237
Outline 238

9 Epipolar Geometry and the Fundamental Matrix 239
9.1 Epipolar geometry 239
9.2 The fundamental matrix F 241
9.3 Fundamental matrices arising from special motions 247
9.4 Geometric representation of the fundamental matrix 250
9.5 Retrieving the camera matrices 253
9.6 The essential matrix 257
9.7 Closure 259

10 3D Reconstruction of Cameras and Structure 262
10.1 Outline of reconstruction method 262
10.2 Reconstruction ambiguity 264
10.3 The projective reconstruction theorem 266
10.4 Stratified reconstruction 267
10.5 Direct reconstruction – using ground truth 275
10.6 Closure 276

11 Computation of the Fundamental Matrix F 279
11.1 Basic equations 279
11.2 The normalized 8-point algorithm 281
11.3 The algebraic minimization algorithm 282
11.4 Geometric distance 284
11.5 Experimental evaluation of the algorithms 288
11.6 Automatic computation of F 290
11.7 Special cases of F-computation 293
11.8 Correspondence of other entities 294
11.9 Degeneracies 295
11.10 A geometric interpretation of F-computation 297
11.11 The envelope of epipolar lines 298
11.12 Image rectification 302
11.13 Closure 308

12 Structure Computation 310
12.1 Problem statement 310
12.2 Linear triangulation methods 312
12.3 Geometric error cost function 313
12.4 Sampson approximation (first-order geometric correction) 314

Samsung/Google, EX1010, p. 7

12.5 An optimal solution 315
12.6 Probability distribution of the estimated 3D point 321
12.7 Line reconstruction 321
12.8 Closure 323

13 Scene planes and homographies 325
13.1 Homographies given the plane and vice versa 326
13.2 Plane induced homographies given F and image correspondences 329
13.3 Computing F given the homography induced by a plane 334
13.4 The infinite homography H_∞ 338
13.5 Closure 340

14 Affine Epipolar Geometry 344
14.1 Affine epipolar geometry 344
14.2 The affine fundamental matrix 345
14.3 Estimating F_A from image point correspondences 347
14.4 Triangulation 353
14.5 Affine reconstruction 353
14.6 Necker reversal and the bas-relief ambiguity 355
14.7 Computing the motion 357
14.8 Closure 360

PART III: Three-View Geometry 363
Outline 364

15 The Trifocal Tensor 365
15.1 The geometric basis for the trifocal tensor 365
15.2 The trifocal tensor and tensor notation 376
15.3 Transfer 379
15.4 The fundamental matrices for three views 383
15.5 Closure 387

16 Computation of the Trifocal Tensor $\mathcal{T}$ 391
16.1 Basic equations 391
16.2 The normalized linear algorithm 393
16.3 The algebraic minimization algorithm 395
16.4 Geometric distance 396
16.5 Experimental evaluation of the algorithms 399
16.6 Automatic computation of $\mathcal{T}$ 400
16.7 Special cases of $\mathcal{T}$-computation 404
16.8 Closure 406

PART IV: N-View Geometry 409
Outline 410

17 N-Linearities and Multiple View Tensors 411
17.1 Bilinear relations 411
17.2 Trilinear relations 414

Samsung/Google, EX1010, p. 8

17.3 Quadrilinear relations 418
17.4 Intersections of four planes 421
17.5 Counting arguments 422
17.6 Number of independent equations 428
17.7 Choosing equations 431
17.8 Closure 432

18 *N*-View Computational Methods 434
18.1 Projective reconstruction – bundle adjustment 434
18.2 Affine reconstruction – the factorization algorithm 436
18.3 Non-rigid factorization 440
18.4 Projective factorization 444
18.5 Projective reconstruction using planes 447
18.6 Reconstruction from sequences 452
18.7 Closure 456

19 Auto-Calibration 458
19.1 Introduction 458
19.2 Algebraic framework and problem statement 459
19.3 Calibration using the absolute dual quadric 462
19.4 The Kruppa equations 469
19.5 A stratified solution 473
19.6 Calibration from rotating cameras 481
19.7 Auto-calibration from planes 485
19.8 Planar motion 486
19.9 Single axis rotation – turntable motion 490
19.10 Auto-calibration of a stereo rig 493
19.11 Closure 497

20 Duality 502
20.1 Carlsson–Weinshall duality 502
20.2 Reduced reconstruction 508
20.3 Closure 513

21 Cheirality 515
21.1 Quasi-affine transformations 515
21.2 Front and back of a camera 518
21.3 Three-dimensional point sets 519
21.4 Obtaining a quasi-affine reconstruction 520
21.5 Effect of transformations on cheirality 521
21.6 Orientation 523
21.7 The cheiral inequalities 525
21.8 Which points are visible in a third view 528
21.9 Which points are in front of which 530
21.10 Closure 531

Samsung/Google, EX1010, p. 9

22 Degenerate Configurations 533
22.1 Camera resectioning 533
22.2 Degeneracies in two views 539
22.3 Carlsson–Weinshall duality 546
22.4 Three-view critical configurations 553
22.5 Closure 558

PART V : Appendices 561
Appendix 1 **Tensor Notation** 562
Appendix 2 **Gaussian (Normal) and χ^2 Distributions** 565
Appendix 3 **Parameter Estimation** 568
Appendix 4 **Matrix Properties and Decompositions** 578
Appendix 5 **Least-squares Minimization** 588
Appendix 6 **Iterative Estimation Methods** 597
Appendix 7 **Some Special Plane Projective Transformations** 628
Bibliography 634
Index 646

Samsung/Google, EX1010, p. 10

Foreword

By Olivier Faugeras

Making a computer see was something that leading experts in the field of Artificial Intelligence thought to be at the level of difficulty of a summer student's project back in the sixties. Forty years later the task is still unsolved and seems formidable. A whole field, called Computer Vision, has emerged as a discipline in itself with strong connections to mathematics and computer science and looser connections to physics, the psychology of perception and the neuro sciences.

One of the likely reasons for this half-failure is the fact that researchers had overlooked the fact, perhaps because of this plague called naive introspection, that perception in general and visual perception in particular are far more complex in animals and humans than was initially thought. There is of course no reason why we should pattern Computer Vision algorithms after biological ones, but the fact of the matter is that

(i) the way biological vision works is still largely unknown and therefore hard to emulate on computers, and
(ii) attempts to ignore biological vision and reinvent a sort of silicon-based vision have not been so successful as initially expected.

Despite these negative remarks, Computer Vision researchers have obtained some outstanding successes, both practical and theoretical.

On the side of practice, and to single out one example, the possibility of guiding vehicles such as cars and trucks on regular roads or on rough terrain using computer vision technology was demonstrated many years ago in Europe, the USA and Japan. This requires capabilities for real-time three-dimensional dynamic scene analysis which are quite elaborate. Today, car manufacturers are slowly incorporating some of these functions in their products.

On the theoretical side some remarkable progress has been achieved in the area of what one could call geometric Computer Vision. This includes the description of the way the appearance of objects changes when viewed from different viewpoints as a function of the objects' shape and the cameras parameters. This endeavour would not have been achieved without the use of fairly sophisticated mathematical techniques encompassing many areas of geometry, ancient and novel. This book deals in particular with the intricate and beautiful geometric relations that exist between the images of objects in the world. These relations are important to analyze for their own sake because

Samsung/Google, EX1010, p. 11

this is one of the goals of science to provide explanations for appearances; they are also important to analyze because of the range of applications their understanding opens up.

The book has been written by two pioneers and leading experts in geometric Computer Vision. They have succeeded in what was something of a challenge, namely to convey in a simple and easily accessible way the mathematics that is necessary for understanding the underlying geometric concepts, to be quite exhaustive in the coverage of the results that have been obtained by them and other researchers worldwide, to analyze the interplay between the geometry and the fact that the image measurements are necessarily noisy, to express many of these theoretical results in algorithmic form so that they can readily be transformed into computer code, and to present many real examples that illustrate the concepts and show the range of applicability of the theory.

Returning to the original holy grail of making a computer see we may wonder whether this kind of work is a step in the right direction. I must leave the readers of the book to answer this question, and be content with saying that no designer of systems using cameras hooked to computers that will be built in the foreseeable future can ignore this work. This is perhaps a step in the direction of defining what it means for a computer to see.

Samsung/Google, EX1010, p. 12

this is one of the goals of science to provide explanations for appearances; they are also important to analyze because of the range of applications their understanding opens up.

The book has been written by two pioneers and leading experts in geometric Computer Vision. They have succeeded in what was something of a challenge, namely to convey in a simple and easily accessible way the mathematics that is necessary for understanding the underlying geometric concepts, to be quite exhaustive in the coverage of the results that have been obtained by them and other researchers worldwide, to analyze the interplay between the geometry and the fact that the image measurements are necessarily noisy, to express many of these theoretical results in algorithmic form so that they can readily be transformed into computer code, and to present many real examples that illustrate the concepts and show the range of applicability of the theory.

Returning to the original holy grail of making a computer see we may wonder whether this kind of work is a step in the right direction. I must leave the readers of the book to answer this question, and be content with saying that no designer of systems using cameras hooked to computers that will be built in the foreseeable future can ignore this work. This is perhaps a step in the direction of defining what it means for a computer to see.

Samsung/Google, EX1010, p. 13

Preface

Over the past decade there has been a rapid development in the understanding and modelling of the geometry of multiple views in computer vision. The theory and practice have now reached a level of maturity where excellent results can be achieved for problems that were certainly unsolved a decade ago, and often thought unsolvable. These tasks and algorithms include:

- Given two images, and no other information, compute matches between the images, and the 3D position of the points that generate these matches and the cameras that generate the images.
- Given three images, and no other information, similarly compute the matches between images of points and lines, and the position in 3D of these points and lines and the cameras.
- Compute the epipolar geometry of a stereo rig, and trifocal geometry of a trinocular rig, without requiring a calibration object.
- Compute the internal calibration of a camera from a sequence of images of natural scenes (i.e. calibration "on the fly").

The distinctive flavour of these algorithms is that they are *uncalibrated* — it is not necessary to know or first need to compute the camera internal parameters (such as the focal length).

Underpinning these algorithms is a new and more complete theoretical understanding of the geometry of multiple uncalibrated views: the number of parameters involved, the constraints between points and lines imaged in the views; and the retrieval of cameras and 3-space points from image correspondences. For example, to determine the epipolar geometry of a stereo rig requires specifying only seven parameters, the camera calibration is not required. These parameters are determined from the correspondence of seven or more image point correspondences. Contrast this uncalibrated route, with the previous calibrated route of a decade ago: each camera would first be calibrated from the image of a carefully engineered calibration object with known geometry. The calibration involves determining 11 parameters for each camera. The epipolar geometry would then have been computed from these two sets of 11 parameters.

This example illustrates the importance of the uncalibrated (projective) approach – using the appropriate representation of the geometry makes explicit the parameters

Samsung/Google, EX1010, p. 14

that are required at each stage of a computation. This avoids computing parameters that have no effect on the final result, and results in simpler algorithms. It is also worth correcting a possible misconception. In the uncalibrated framework, entities (for instance point positions in 3-space) are often recovered to within a precisely defined ambiguity. This ambiguity does not mean that the points are poorly estimated.

More practically, it is often not possible to calibrate cameras once-and-for-all; for instance where cameras are moved (on a mobile vehicle) or internal parameters are changed (a surveillance camera with zoom). Furthermore, calibration information is simply not available in some circumstances. Imagine computing the motion of a camera from a video sequence, or building a virtual reality model from archive film footage where both motion and internal calibration information are unknown.

The achievements in multiple view geometry have been possible because of developments in our theoretical understanding, but also because of improvements in estimating mathematical objects from images. The first improvement has been an attention to the error that should be minimized in over-determined systems – whether it be algebraic, geometric or statistical. The second improvement has been the use of robust estimation algorithms (such as RANSAC), so that the estimate is unaffected by "outliers" in the data. Also these techniques have generated powerful search and matching algorithms.

Many of the problems of reconstruction have now reached a level where we may claim that they are solved. Such problems include:

(i) Estimation of the multifocal tensors from image point correspondences, particularly the fundamental matrix and trifocal tensors (the quadrifocal tensor having not received so much attention).
(ii) Extraction of the camera matrices from these tensors, and subsequent projective reconstruction from two, three and four views.

Other significant successes have been achieved, though there may be more to learn about these problems. Examples include:

(i) Application of bundle adjustment to solve more general reconstruction problems.
(ii) Metric (Euclidean) reconstruction given minimal assumptions on the camera matrices.
(iii) Automatic detection of correspondences in image sequences, and elimination of outliers and false matches using the multifocal tensor relationships.

Roadplan. The book is divided into six parts and there are seven short appendices. Each part introduces a new geometric relation: the homography for background, the camera matrix for single view, the fundamental matrix for two views, the trifocal tensor for three views, and the quadrifocal tensor for four views. In each case there is a chapter describing the relation, its properties and applications, and a companion chapter describing algorithms for its estimation from image measurements. The estimation algorithms described range from cheap, simple, approaches through to the optimal algorithms which are currently believed to be the best available.

Samsung/Google, EX1010, p. 15

Part 0: Background. This part is more tutorial than the others. It introduces the central ideas in the projective geometry of 2-space and 3-space (for example ideal points, and the absolute conic); how this geometry may be represented, manipulated, and estimated; and how the geometry relates to various objectives in computer vision such as rectifying images of planes to remove perspective distortion.

Part 1: Single view geometry. Here the various cameras that model the perspective projection from 3-space to an image are defined and their anatomy explored. Their estimation using traditional techniques of calibration objects is described, as well as camera calibration from vanishing points and vanishing lines.

Part 2: Two view geometry. This part describes the epipolar geometry of two cameras, projective reconstruction from image point correspondences, methods of resolving the projective ambiguity, optimal triangulation, transfer between views via planes.

Part 3: Three view geometry. Here the trifocal geometry of three cameras is described, including transfer of a point correspondence from two views to a third, and similarly transfer for a line correspondence; computation of the geometry from point and line correspondences, retrieval of the camera matrices.

Part 4: N-views. This part has two purposes. First, it extends three view geometry to four views (a minor extension) and describes estimation methods applicable to N-views, such as the factorization algorithm of Tomasi and Kanade for computing structure and motion simultaneously from multiple images. Second, it covers themes that have been touched on in earlier chapters, but can be understood more fully and uniformly by emphasising their commonality. Examples include deriving multi-linear view constraints on correspondences, auto-calibration, and ambiguous solutions.

Appendices. These describe further background material on tensors, statistics, parameter estimation, linear and matrix algebra, iterative estimation, the solution of sparse matrix systems, and special projective transformations.

Acknowledgements. We have benefited enormously from ideas and discussions with our colleagues: Paul Beardsley, Stefan Carlsson, Olivier Faugeras, Andrew Fitzgibbon, Jitendra Malik, Steve Maybank, Amnon Shashua, Phil Torr, Bill Triggs.

If there are only a countable number of errors in this book then it is due to Antonio Criminisi, David Liebowitz and Frederik Schaffalitzky who have with great energy and devotion read most of it, and made numerous suggestions for improvements. Similarly both Peter Sturm and Bill Triggs have suggested many improvements to various chapters. We are grateful to other colleagues who have read individual chapters: David Capel, Lourdes de Agapito Vicente, Bob Kaucic, Steve Maybank, Peter Tu.

We are particularly grateful to those who have provided multiple figures: Paul Beardsley, Antonio Criminisi, Andrew Fitzgibbon, David Liebowitz, and Larry Shapiro; and for individual figures from: Martin Armstrong, David Capel, Lourdes de Agapito Vicente, Eric Hayman, Phil Pritchett, Luc Robert, Cordelia Schmid, and others who are explicitly acknowledged in figure captions.

Samsung/Google, EX1010, p. 16

At Cambridge University Press we thank David Tranah for his constant source of advice and patience, and Michael Behrend for excellent copy editing.

A small number of minor errors have been corrected in the reprinted editions, and we thank the following readers for pointing these out: Luis Baumela, Niclas Borlin, Mike Brooks, Jun ho. Choi, Wojciech Chojnacki, Carlo Colombo, Nicolas Dano, Andrew Fitzgibbon, Bogdan Georgescu, Fredrik Kahl, Bob Kaucic, Jae-Hak Kim, Hansung Lee, Dennis Maier, Karsten Muelhmann, David Nister, Andreas Olsson, Stéphane Paris, Frederik Schaffalitzky, Bill Severson, Pedro Lopez de Teruel Alcolea, Bernard Thiesse, Ken Thornton, Magdalena Urbanek, Gergely Vass, Eugene Vendrovsky, Sui Wei, and Tomáš Werner.

The second edition. This new paperback edition has been expanded to include some of the developments since the original version of July 2000. For example, the book now covers the discovery of a closed form factorization solution in the projective case when a plane is visible in the scene, and the extension of affine factorization to non-rigid scenes. We have also extended the discussion of single view geometry (chapter 8) and three view geometry (chapter 15), and added an appendix on parameter estimation.

In preparing this second edition we are very grateful to colleagues who have made suggestion for improvements and additions. These include Marc Pollefeys, Bill Triggs and in particular Tomáš Werner who provided excellent and comprehensive comments. We also thank Antonio Criminisi, Andrew Fitzgibbon, Rob Fergus, David Liebowitz, and particularly Josef Šivic, for proof reading and very helpful comments on parts of the new material. As always we are grateful to David Tranah of CUP.

The figures appearing in this book can be downloaded from

`http://www.robots.ox.ac.uk/~vgg/hzbook.html`

This site also includes Matlab code for several of the algorithms, and lists the errata of earlier printings.

I am never forget the day my first book is published. Every chapter I stole from somewhere else. Index I copy from old Vladivostok telephone directory. This book, this book was sensational!

Excerpts from "Nikolai Ivanovich Lobachevsky" by Tom Lehrer.

Samsung/Google, EX1010, p. 17

1

Introduction – a Tour of Multiple View Geometry

This chapter is an introduction to the principal ideas covered in this book. It gives an *informal* treatment of these topics. Precise, unambiguous definitions, careful algebra, and the description of well honed estimation algorithms is postponed until chapter 2 and the following chapters in the book. Throughout this introduction we will generally not give specific forward pointers to these later chapters. The material referred to can be located by use of the index or table of contents.

1.1 Introduction – the ubiquitous projective geometry

We are all familiar with projective transformations.When we look at a picture, we see squares that are not squares, or circles that are not circles. The transformation that maps these planar objects onto the picture is an example of a projective transformation.

So what properties of geometry are preserved by projective transformations? Certainly, shape is not, since a circle may appear as an ellipse. Neither are lengths since two perpendicular radii of a circle are stretched by different amounts by the projective transformation. Angles, distance, ratios of distances – none of these are preserved, and it may appear that very little geometry is preserved by a projective transformation. However, a property that is preserved is that of straightness. It turns out that this is the most general requirement on the mapping, and we may define a projective transformation of a plane as any mapping of the points on the plane that preserves straight lines.

To see why we will require projective geometry we start from the familiar Euclidean geometry. This is the geometry that describes angles and shapes of objects. Euclidean geometry is troublesome in one major respect – we need to keep making an exception to reason about some of the basic concepts of the geometry – such as intersection of lines. Two lines (we are thinking here of 2-dimensional geometry) almost always meet in a point, but there are some pairs of lines that do not do so – those that we call parallel. A common linguistic device for getting around this is to say that parallel lines meet "at infinity". However this is not altogether convincing, and conflicts with another dictum, that infinity does not exist, and is only a convenient fiction. We can get around this by

Samsung/Google, EX1010, p. 18

enhancing the Euclidean plane by the addition of these points at infinity where parallel lines meet, and resolving the difficulty with infinity by calling them "ideal points."

By adding these points at infinity, the familiar Euclidean space is transformed into a new type of geometric object, projective space. This is a very useful way of thinking, since we are familiar with the properties of Euclidean space, involving concepts such as distances, angles, points, lines and incidence. There is nothing very mysterious about projective space – it is just an extension of Euclidean space in which two lines always meet in a point, though sometimes at mysterious points at infinity.

Coordinates. A point in Euclidean 2-space is represented by an ordered pair of real numbers, (x, y). We may add an extra coordinate to this pair, giving a triple $(x, y, 1)$, that we declare to represent the same point. This seems harmless enough, since we can go back and forward from one representation of the point to the other, simply by adding or removing the last coordinate. We now take the important conceptual step of asking why the last coordinate needs to be 1 – after all, the others two coordinates are not so constrained. What about a coordinate triple $(x, y, 2)$. It is here that we make a definition and say that $(x, y, 1)$ and $(2x, 2y, 2)$ represent the same point, and furthermore, (kx, ky, k) represents the same point as well, for any non-zero value k. Formally, points are represented by *equivalence classes* of coordinate triples, where two triples are equivalent when they differ by a common multiple. These are called the *homogeneous coordinates* of the point. Given a coordinate triple (kx, ky, k), we can get the original coordinates back by dividing by k to get (x, y).

The reader will observe that although $(x, y, 1)$ represents the same point as the coordinate pair (x, y), there is no point that corresponds to the triple $(x, y, 0)$. If we try to divide by the last coordinate, we get the point $(x/0, y/0)$ which is infinite. This is how the points at infinity arise then. They are the points represented by homogeneous coordinates in which the last coordinate is zero.

Once we have seen how to do this for 2-dimensional Euclidean space, extending it to a projective space by representing points as homogeneous vectors, it is clear that we can do the same thing in any dimension. The Euclidean space $\mathbb{R}^n$ can be extended to a projective space $\mathbb{P}^n$ by representing points as homogeneous vectors. It turns out that the points at infinity in the two-dimensional projective space form a line, usually called the *line at infinity*. In three-dimensions they form the *plane at infinity*.

Homogeneity. In classical Euclidean geometry all points are the same. There is no distinguished point. The whole of the space is homogeneous. When coordinates are added, one point is seemingly picked out as the origin. However, it is important to realize that this is just an accident of the particular coordinate frame chosen. We could just as well find a different way of coordinatizing the plane in which a different point is considered to be the origin. In fact, we can consider a change of coordinates for the Euclidean space in which the axes are shifted and rotated to a different position. We may think of this in another way as the space itself translating and rotating to a different position. The resulting operation is known as a Euclidean transform.

A more general type of transformation is that of applying a linear transformation

Samsung/Google, EX1010, p. 19

to $\mathbb{R}^n$, followed by a Euclidean transformation moving the origin of the space. We may think of this as the space moving, rotating and finally *stretching* linearly possibly by different ratios in different directions. The resulting transformation is known as an *affine* transformation.

The result of either a Euclidean or an affine transformation is that points at infinity remain at infinity. Such points are in some way preserved, at least as a set, by such transformations. They are in some way distinguished, or special in the context of Euclidean or *affine* geometry.

From the point of view of projective geometry, points at infinity are not any different from other points. Just as Euclidean space is uniform, so is projective space. The property that points at infinity have final coordinate zero in a homogeneous coordinate representation is nothing other than an accident of the choice of coordinate frame. By analogy with Euclidean or affine transformations, we may define a *projective transformation* of projective space. A linear transformation of Euclidean space $\mathbb{R}^n$ is represented by matrix multiplication applied to the coordinates of the point. In just the same way a projective transformation of projective space $\mathbb{P}^n$ is a mapping of the homogeneous coordinates representing a point (an $(n+1)$-vector), in which the coordinate vector is multiplied by a non-singular matrix. Under such a mapping, points at infinity (with final coordinate zero) are mapped to arbitrary other points. The points at infinity are not preserved. Thus, a projective transformation of projective space $\mathbb{P}^n$ is represented by a linear transformation of homogeneous coordinates

$$\mathbf{X}' = \mathtt{H}_{(n+1)\times(n+1)}\mathbf{X}.$$

In computer vision problems, projective space is used as a convenient way of representing the real 3D world, by extending it to the 3-dimensional (3D) projective space. Similarly images, usually formed by projecting the world onto a 2-dimensional representation, are for convenience extended to be thought of as lying in the 2-dimensional projective space. In reality, the real world, and images of it do not contain points at infinity, and we need to keep our finger on which are the fictitious points, namely the line at infinity in the image and the plane at infinity in the world. For this reason, although we usually work with the projective spaces, we are aware that the line and plane at infinity are in some way special. This goes against the spirit of pure projective geometry, but makes it useful for our practical problems. Generally we try to have it both ways by treating all points in projective space as equals when it suits us, and singling out the line at infinity in space or the plane at infinity in the image when that becomes necessary.

1.1.1 Affine and Euclidean Geometry

We have seen that projective space can be obtained from Euclidean space by adding a line (or plane) at infinity. We now consider the reverse process of going backwards. This discussion is mainly concerned with two and three-dimensional projective space.

Affine geometry. We will take the point of view that the projective space is initially homogeneous, with no particular coordinate frame being preferred. In such a space,

Samsung/Google, EX1010, p. 20

there is no concept of parallelism of lines, since parallel lines (or planes in the three-dimensional case) are ones that meet at infinity. However, in projective space, there is no concept of which points are at infinity – all points are created equal. We say that parallelism is not a concept of projective geometry. It is simply meaningless to talk about it.

In order for such a concept to make sense, we need to pick out some particular line, and decide that this is the line at infinity. This results in a situation where although all points are created equal, some are more equal than others. Thus, start with a blank sheet of paper, and imagine that it extends to infinity and forms a projective space $\mathbb{P}^2$. What we see is just a small part of the space, that looks a lot like a piece of the ordinary Euclidean plane. Now, let us draw a straight line on the paper, and declare that this is the line at infinity. Next, we draw two other lines that intersect at this distinguished line. Since they meet at the "line at infinity" we define them as being parallel. The situation is similar to what one sees by looking at an infinite plane. Think of a photograph taken in a very flat region of the earth. The points at infinity in the plane show up in the image as the horizon line. Lines, such as railway tracks show up in the image as lines meeting at the horizon. Points in the image lying above the horizon (the image of the sky) apparently do not correspond to points on the world plane. However, if we think of extending the corresponding ray backwards behind the camera, it will meet the plane at a point behind the camera. Thus there is a one-to-one relationship between points in the image and points in the world plane. The points at infinity in the world plane correspond to a real horizon line in the image, and parallel lines in the world correspond to lines meeting at the horizon. From our point of view, the world plane and its image are just alternative ways of viewing the geometry of a projective plane, plus a distinguished line. The geometry of the projective plane and a distinguished line is known as *affine geometry* and any projective transformation that maps the distinguished line in one space to the distinguished line of the other space is known as an *affine transformation*.

By identifying a special line as the "line at infinity" we are able to define parallelism of straight lines in the plane. However, certain other concepts make sense as well, as soon as we can define parallelism. For instance, we may define equalities of intervals between two points on parallel lines. For instance, if A, B, C and D are points, and the lines AB and CD are parallel, then we define the two intervals AB and CD to have *equal length* if the lines AC and BD are also parallel. Similarly, two intervals on the same line are equal if there exists another interval on a parallel line that is equal to both.

Euclidean geometry. By distinguishing a special line in a projective plane, we gain the concept of parallelism and with it *affine geometry*. Affine geometry is seen as specialization of projective geometry, in which we single out a particular line (or plane – according to the dimension) and call it the line at infinity.

Next, we turn to Euclidean geometry and show that by singling out some special feature of the line or plane at infinity affine geometry becomes Euclidean geometry. In

Samsung/Google, EX1010, p. 21

doing so, we introduce one of the most important concepts of this book, the *absolute conic*.

We begin by considering two-dimensional geometry, and start with circles. Note that a circle is not a concept of affine geometry, since arbitrary stretching of the plane, which preserves the line at infinity, turns the circle into an ellipse. Thus, affine geometry does not distinguish between circles and ellipses.

In Euclidean geometry however, they are distinct, and have an important difference. Algebraically, an ellipse is described by a second-degree equation. It is therefore expected, and true that two ellipses will most generally intersect in four points. However, it is geometrically evident that two distinct circles can not intersect in more than two points. Algebraically, we are intersecting two second-degree curves here, or equivalently solving two quadratic equations. We should expect to get four solutions. The question is, what is special about circles that they only intersect in two points.

The answer to this question is of course that there exist two other solutions, the two circles meeting in two other *complex* points. We do not have to look very far to find these two points.

The equation for a circle in homogeneous coordinates (x, y, w) is of the form

$$(x - aw)^2 + (y - bw)^2 = r^2 w^2$$

This represents the circle with centre represented in homogeneous coordinates as $(x_0, y_0, w_0)^\mathsf{T} = (a, b, 1)^\mathsf{T}$. It is quickly verified that the points $(x, y, w)^\mathsf{T} = (1, \pm i, 0)^\mathsf{T}$ lie on every such circle. To repeat this interesting fact, every circle passes through the points $(1, \pm i, 0)^\mathsf{T}$, and therefore they lie in the intersection of any two circles. Since their final coordinate is zero, these two points lie on the line at infinity. For obvious reasons, they are called the *circular points* of the plane. Note that although the two circular points are complex, they satisfy a pair of real equations: $x^2 + y^2 = 0$; $w = 0$.

This observation gives the clue of how we may define Euclidean geometry. Euclidean geometry arises from projective geometry by singling out first a line at infinity and subsequently, two points called *circular points* lying on this line. Of course the circular points are complex points, but for the most part we do not worry too much about this. Now, we may define a circle as being any conic (a curve defined by a second-degree equation) that passes through the two circular points. Note that in the standard Euclidean coordinate system, the circular points have the coordinates $(1, \pm i, 0)^\mathsf{T}$. In assigning a Euclidean structure to a projective plane, however, we may designate any line and any two (complex) points on that line as being the line at infinity and the circular points.

As an example of applying this viewpoint, we note that a general conic may be found passing through five arbitrary points in the plane, as may be seen by counting the number of coefficients of a general quadratic equation $ax^2 + by^2 + \ldots + fw^2 = 0$. A circle on the other hand is defined by only three points. Another way of looking at this is that it is a conic passing through two special points, the circular points, as well as three other points, and hence as any other conic, it requires five points to specify it uniquely.

It should not be a surprise that as a result of singling out two circular points one

Samsung/Google, EX1010, p. 22

obtains the whole of the familiar Euclidean geometry. In particular, concepts such as angle and length ratios may be defined in terms of the circular points. However, these concepts are most easily defined in terms of some coordinate system for the Euclidean plane, as will be seen in later chapters.

3D Euclidean geometry. We saw how the Euclidean plane is defined in terms of the projective plane by specifying a line at infinity and a pair of circular points. The same idea may be applied to 3D geometry. As in the two-dimensional case, one may look carefully at spheres, and how they intersect. Two spheres intersect in a circle, and not in a general fourth-degree curve, as the algebra suggests, and as two general ellipsoids (or other *quadric* surfaces) do. This line of thought leads to the discovery that in homogeneous coordinates $(X, Y, Z, T)^\top$ all spheres intersect the plane at infinity in a curve with the equations: $X^2 + Y^2 + Z^2 = 0$; $T = 0$. This is a second-degree curve (a conic) lying on the plane at infinity, and consisting only of complex points. It is known as the *absolute conic* and is one of the key geometric entities in this book, most particularly because of its connection to camera calibration, as will be seen later.

The absolute conic is defined by the above equations only in the Euclidean coordinate system. In general we may consider 3D Euclidean space to be derived from projective space by singling out a particular plane as the plane at infinity and specifying a particular conic lying in this plane to be the absolute conic. These entities may have quite general descriptions in terms of a coordinate system for the projective space.

We will not here go into details of how the absolute conic determines the complete Euclidean 3D geometry. A single example will serve. Perpendicularity of lines in space is not a valid concept in affine geometry, but belongs to Euclidean geometry. The perpendicularity of lines may be defined in terms of the absolute conic, as follows. By extending the lines until they meet the plane at infinity, we obtain two points called the directions of the two lines. Perpendicularity of the lines is defined in terms of the relationship of the two directions to the absolute conic. The lines are perpendicular if the two directions are conjugate points with respect to the absolute conic (see figure 3.8(p83)). The geometry and algebraic representation of conjugate points are defined in section 2.8.1(p58). Briefly, if the absolute conic is represented by a 3×3 symmetric matrix Ω_∞, and the directions are the points $\mathbf{d}_1$ and $\mathbf{d}_2$, then they are conjugate with respect to Ω_∞ if $\mathbf{d}_1^\top \Omega_\infty \mathbf{d}_2 = 0$. More generally, angles may be defined in terms of the absolute conic in any arbitrary coordinate system, as expressed by (3.23–p82).

1.2 Camera projections

One of the principal topics of this book is the process of image formation, namely the formation of a two-dimensional representation of a three-dimensional world, and what we may deduce about the 3D structure of what appears in the images.

The drop from three-dimensional world to a two-dimensional image is a projection process in which we lose one dimension. The usual way of modelling this process is by *central projection* in which a ray from a point in space is drawn from a 3D world point through a fixed point in space, the *centre of projection*. This ray will intersect a specific plane in space chosen as the *image plane*. The intersection of the ray with the

Samsung/Google, EX1010, p. 23

image plane represents the image of the point. If the 3D structure lies on a plane then there is no drop in dimension.

This model is in accord with a simple model of a camera, in which a ray of light from a point in the world passes through the lens of a camera and impinges on a film or digital device, producing an image of the point. Ignoring such effects as focus and lens thickness, a reasonable approximation is that all the rays pass through a single point, the centre of the lens.

In applying projective geometry to the imaging process, it is customary to model the world as a 3D projective space, equal to $\mathbb{R}^3$ along with points at infinity. Similarly the model for the image is the 2D projective plane $\mathbb{P}^2$. Central projection is simply a map from $\mathbb{P}^3$ to $\mathbb{P}^2$. If we consider points in $\mathbb{P}^3$ written in terms of homogeneous coordinates $(\mathtt{X}, \mathtt{Y}, \mathtt{Z}, \mathtt{T})^\top$ and let the centre of projection be the origin $(0, 0, 0, 1)^\top$, then we see that the set of all points $(\mathtt{X}, \mathtt{Y}, \mathtt{Z}, \mathtt{T})^\top$ for fixed $\mathtt{X}$, $\mathtt{Y}$ and $\mathtt{Z}$, but varying $\mathtt{T}$ form a single ray passing through the point centre of projection, and hence all mapping to the same point. Thus, the final coordinate of $(\mathtt{X}, \mathtt{Y}, \mathtt{Z}, \mathtt{T})$ is irrelevant to where the point is imaged. In fact, the image point is the point in $\mathbb{P}^2$ with homogeneous coordinates $(\mathtt{X}, \mathtt{Y}, \mathtt{Z})^\top$. Thus, the mapping may be represented by a mapping of 3D homogeneous coordinates, represented by a 3×4 matrix $\mathtt{P}$ with the block structure $\mathtt{P} = [\mathtt{I}_{3\times3}|\mathtt{0}_3]$, where $\mathtt{I}_{3\times3}$ is the 3×3 identity matrix and $\mathtt{0}_3$ a zero 3-vector. Making allowance for a different centre of projection, and a different projective coordinate frame in the image, it turns out that the most general imaging projection is represented by an arbitrary 3×4 matrix of rank 3, acting on the homogeneous coordinates of the point in $\mathbb{P}^3$ mapping it to the imaged point in $\mathbb{P}^2$. This matrix $\mathtt{P}$ is known as the camera matrix.

In summary, the action of a projective camera on a point in space may be expressed in terms of a linear mapping of homogeneous coordinates as

$$\begin{pmatrix} x \\ y \\ w \end{pmatrix} = \mathtt{P}_{3\times4} \begin{pmatrix} \mathtt{X} \\ \mathtt{Y} \\ \mathtt{Z} \\ \mathtt{T} \end{pmatrix}$$

Furthermore, if all the points lie on a plane (we may choose this as the plane $\mathtt{Z} = 0$) then the linear mapping reduces to

$$\begin{pmatrix} x \\ y \\ w \end{pmatrix} = \mathtt{H}_{3\times3} \begin{pmatrix} \mathtt{X} \\ \mathtt{Y} \\ \mathtt{T} \end{pmatrix}$$

which is a projective transformation.

Cameras as points. In a central projection, points in $\mathbb{P}^3$ are mapped to points in $\mathbb{P}^2$, all points in a ray passing through the centre of projection projecting to the same point in an image. For the purposes of image projection, it is possible to consider all points along such a ray as being equal. We can go one step further, and think of the ray through the projection centre as representing the image point. Thus, the set of all image points is the same as the set of rays through the camera centre. If we represent

Samsung/Google, EX1010, p. 24

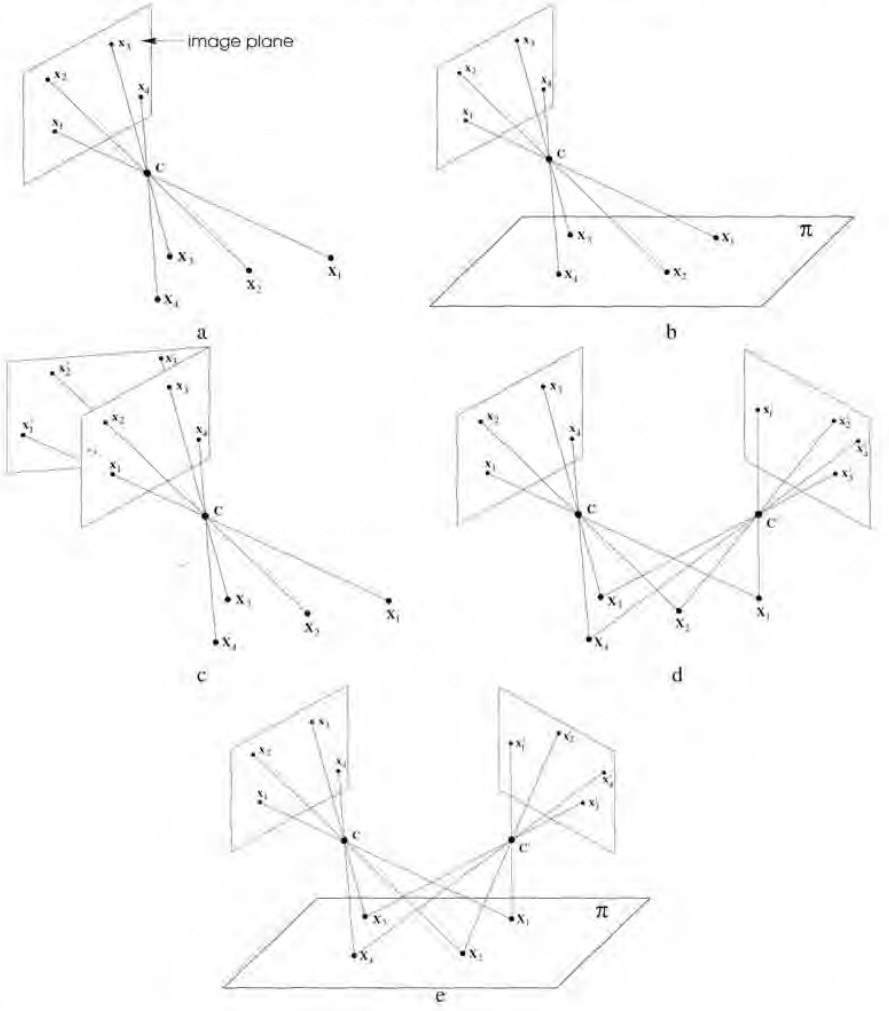


Fig. 1.1. **The camera centre is the essence.** *(a) Image formation: the image points $\mathbf{x}_i$ are the intersection of a plane with rays from the space points $\mathbf{X}_i$ through the camera centre $\mathbf{C}$. (b) If the space points are coplanar then there is a projective transformation between the world and image planes, $\mathbf{x}_i = \mathtt{H}_{3\times3}\mathtt{X}_i$. (c) All images with the same camera centre are related by a projective transformation, $\mathbf{x}'_i = \mathtt{H}'_{3\times3}\mathbf{x}_i$. Compare (b) and (c) – in both cases planes are mapped to one another by rays through a centre. In (b) the mapping is between a scene and image plane, in (c) between two image planes. (d) If the camera centre moves, then the images are in general not related by a projective transformation, unless (e) all the space points are coplanar.*

the ray from $(0,0,0,1)^\mathsf{T}$ through the point $(\mathtt{X},\mathtt{Y},\mathtt{Z},\mathtt{T})^\mathsf{T}$ by its first three coordinates $(\mathtt{X},\mathtt{Y},\mathtt{Z})^\mathsf{T}$, it is easily seen that for any constant k, the ray $k(\mathtt{X},\mathtt{Y},\mathtt{Z})^\mathsf{T}$ represents the same ray. Thus the rays themselves are represented by homogeneous coordinates. In

Samsung/Google, EX1010, p. 25

fact they make up a 2-dimensional space of rays. The set of rays themselves may be thought of as a representation of the image space $\mathbb{P}^2$. In this representation of the image, all that is important is the camera centre, for this alone determines the set of rays forming the image. Different camera matrices representing the image formation from the same centre of projection reflect only different coordinate frames for the set of rays forming the image. Thus two images taken from the same point in space are projectively equivalent. It is only when we start to measure points in an image, that a particular coordinate frame for the image needs to be specified. Only then does it become necessary to specify a particular camera matrix. In short, modulo field-of-view which we ignore for now, all images acquired with the same camera centre are equivalent – they can be mapped onto each other by a projective transformation without any information about the 3D points or position of the camera centre. These issues are illustrated in figure 1.1.

Calibrated cameras. To understand fully the Euclidean relationship between the image and the world, it is necessary to express their relative Euclidean geometry. As we have seen, the Euclidean geometry of the 3D world is determined by specifying a particular plane in $\mathbb{P}^3$ as being the plane at infinity, and a specific conic Ω in that plane as being the absolute conic. For a camera not located on the plane at infinity, the plane at infinity in the world maps one-to-one onto the image plane. This is because any point in the image defines a ray in space that meets the plane at infinity in a single point. Thus, the plane at infinity in the world does not tell us anything new about the image. The absolute conic, however being a conic in the plane at infinity must project to a conic in the image. The resulting image curve is called the Image of the Absolute Conic, or IAC. If the location of the IAC is known in an image, then we say that the camera is *calibrated*.

In a calibrated camera, it is possible to determine the angle between the two rays back-projected from two points in the image. We have seen that the angle between two lines in space is determined by where they meet the plane at infinity, relative to the absolute conic. In a calibrated camera, the plane at infinity and the absolute conic Ω_∞ are projected one-to-one onto the image plane and the IAC, denoted ω. The projective relationship between the two image points and ω is exactly equal to the relationship between the intersections of the back-projected rays with the plane at infinity, and Ω_∞. Consequently, knowing the IAC, one can measure the angle between rays by direct measurements in the image. Thus, for a calibrated camera, one can measure angles between rays, compute the field of view represented by an image patch or determine whether an ellipse in the image back-projects to a circular cone. Later on, we will see that it helps us to determine the *Euclidean structure* of a reconstructed scene.

Example 1.1. 3D reconstructions from paintings
Using techniques of projective geometry, it is possible in many instances to reconstruct scenes from a single image. This cannot be done without some assumptions being made about the imaged scene. Typical techniques involve the analysis of features such as parallel lines and vanishing points to determine the affine structure of the scene, for

Samsung/Google, EX1010, p. 26



Fig. 1.2. **Single view reconstruction.** *(a) Original painting – St. Jerome in his study, 1630, Hendrick van Steenwijck (1580-1649), Joseph R. Ritman Private Collection, Amsterdam, The Netherlands. (b) (c)(d) Views of the 3D model created from the painting. Figures courtesy of Antonio Criminisi.*

example by determining the line at infinity for observed planes in the image. Knowledge (or assumptions) about angles observed in the scene, most particularly orthogonal lines or planes, can be used to upgrade the affine reconstruction to Euclidean.

It is not yet possible for such techniques to be fully automatic. However, projective geometric knowledge may be built into a system that allows user-guided single-view reconstruction of the scene.

Such techniques have been used to reconstruct 3D texture mapped graphical models derived from old-master paintings. Starting in the Renaissance, paintings with extremely accurate perspective were produced. In figure 1.2 a reconstruction carried out from such a painting is shown. △

1.3 Reconstruction from more than one view

We now turn to one of the major topics in the book – that of reconstructing a scene from several images. The simplest case is that of two images, which we will consider first. As a mathematical abstraction, we restrict the discussion to "scenes" consisting of points only.

The usual input to many of the algorithms given in this book is a set of point correspondences. In the two-view case, therefore, we consider a set of correspondences

Samsung/Google, EX1010, p. 27



a



b



c



d

Fig. 1.2. **Single view reconstruction.** *(a) Original painting – St. Jerome in his study, 1630, Hendrick van Steenwijck (1580-1649), Joseph R. Ritman Private Collection, Amsterdam, The Netherlands. (b) (c)(d) Views of the 3D model created from the painting. Figures courtesy of Antonio Criminisi.*

example by determining the line at infinity for observed planes in the image. Knowledge (or assumptions) about angles observed in the scene, most particularly orthogonal lines or planes, can be used to upgrade the affine reconstruction to Euclidean.

It is not yet possible for such techniques to be fully automatic. However, projective geometric knowledge may be built into a system that allows user-guided single-view reconstruction of the scene.

Such techniques have been used to reconstruct 3D texture mapped graphical models derived from old-master paintings. Starting in the Renaissance, paintings with extremely accurate perspective were produced. In figure 1.2 a reconstruction carried out from such a painting is shown. △

1.3 Reconstruction from more than one view

We now turn to one of the major topics in the book – that of reconstructing a scene from several images. The simplest case is that of two images, which we will consider first. As a mathematical abstraction, we restrict the discussion to "scenes" consisting of points only.

The usual input to many of the algorithms given in this book is a set of point correspondences. In the two-view case, therefore, we consider a set of correspondences

Samsung/Google, EX1010, p. 28

$\mathbf{x}_i \leftrightarrow \mathbf{x}'_i$ in two images. It is assumed that there exist some camera matrices, $\mathtt{P}$ and $\mathtt{P}'$ and a set of 3D points $\mathbf{X}_i$ that give rise to these image correspondences in the sense that $\mathtt{P}\mathbf{X}_i = \mathbf{x}_i$ and $\mathtt{P}'\mathbf{X}_i = \mathbf{x}'_i$. Thus, the point $\mathbf{X}_i$ projects to the two given data points. However, neither the cameras (represented by projection matrices $\mathtt{P}$ and $\mathtt{P}'$), nor the points $\mathbf{X}_i$ are known. It is our task to determine them.

It is clear from the outset that it is impossible to determine the positions of the points uniquely. This is a general ambiguity that holds however many images we are given, and even if we have more than just point correspondence data. For instance, given several images of a cube, it is impossible to tell its absolute position (is it located in a night-club in Addis Ababa, or the British Museum), its orientation (which face is facing north) or its scale. We express this by saying that the reconstruction is possible at best up to a similarity transformation of the world. However, it turns out that unless something is known about the calibration of the two cameras, the ambiguity in the reconstruction is expressed by a more general class of transformations – projective transformations.

This ambiguity arises because it is possible to apply a projective transformation (represented by a 4×4 matrix $\mathtt{H}$) to each point $\mathbf{X}_i$, and on the right of each camera matrix $\mathtt{P}_j$, without changing the projected image points, thus:

$$\mathtt{P}_j\mathbf{X}_i = (\mathtt{P}_j\mathtt{H}^{-1})(\mathtt{H}\mathbf{X}_i). \tag{1.1}$$

There is no compelling reason to choose one set of points and camera matrices over the other. The choice of $\mathtt{H}$ is essentially arbitrary, and we say that the reconstruction has a projective ambiguity, or is a *projective reconstruction*.

However, the good news is that this is the worst that can happen. It is possible to reconstruct a set of points from two views, up to an unavoidable projective ambiguity. Well, to be able to say this, we need to make a few qualifications; there must be sufficiently many points, at least seven, and they must not lie in one of various well-defined *critical configurations*.

The basic tool in the reconstruction of point sets from two views is the *fundamental matrix*, which represents the constraint obeyed by image points $\mathbf{x}$ and $\mathbf{x}'$ if they are to be images of the same 3D point. This constraint arises from the coplanarity of the camera centres of the two views, the images points and the space point. Given the fundamental matrix $\mathtt{F}$, a pair of matching points $\mathbf{x}_i \leftrightarrow \mathbf{x}'_i$ must satisfy

$$\mathbf{x}_i'^{\mathsf{T}}\mathtt{F}\mathbf{x}_i = 0$$

where $\mathtt{F}$ is a 3×3 matrix of rank 2. These equations are linear in the entries of the matrix $\mathtt{F}$, which means that if $\mathtt{F}$ is unknown, then it can be computed from a set of point correspondences.

A pair of camera matrices $\mathtt{P}$ and $\mathtt{P}'$ uniquely determine a fundamental matrix $\mathtt{F}$, and conversely, the fundamental matrix determines the pair of camera matrices, up to a 3D projective ambiguity. Thus, the fundamental matrix encapsulates the complete projective geometry of the pair of cameras, and is unchanged by projective transformation of 3D.

Samsung/Google, EX1010, p. 29

The fundamental-matrix method for reconstructing the scene is very simple, consisting of the following steps:

(i) Given several point correspondences $\mathbf{x}_i \leftrightarrow \mathbf{x}'_i$ across two views, form linear equations in the entries of F based on the coplanarity equations $\mathbf{x}'^{\mathsf{T}}_i \mathbf{F} \mathbf{x}_i = 0$.
(ii) Find F as the solution to a set of linear equations.
(iii) Compute a pair of camera matrices from F according to the simple formula given in section 9.5(p253).
(iv) Given the two cameras $(\mathbf{P}, \mathbf{P}')$ and the corresponding image point pairs $\mathbf{x}_i \leftrightarrow \mathbf{x}'_i$, find the 3D point $\mathbf{X}_i$ that projects to the given image points. Solving for X in this way is known as *triangulation*.

The algorithm given here is an outline only, and each part of it is examined in detail in this book. The algorithm should not be implemented directly from this brief description.

1.4 Three-view geometry

In the last section it was discussed how reconstruction of a set of points, and the relative placement of the cameras, is possible from two views of a set of points. The reconstruction is possible only up to a projective transformation of space, and the corresponding adjustment to the camera matrices.

In this section, we consider the case of three views. Whereas for two views, the basic algebraic entity is the fundamental matrix, for three views this role is played by the trifocal tensor. The trifocal tensor is a $3 \times 3 \times 3$ array of numbers that relate the coordinates of corresponding points or lines in three views. Just as the fundamental matrix is determined by the two camera matrices, and determines them up to projective transformation, so in three views, the trifocal tensor is determined by the three camera matrices, and in turn determines them, again up to projective transformation. Thus, the trifocal tensor encapsulates the relative projective geometry of the three cameras.

For reasons that will be explained in chapter 15 it is usual to write some of the indices of a tensor as lower and some as upper indices. These are referred to as the covariant and contravariant indices. The trifocal tensor is of the form $\mathcal{T}_i^{jk}$, having two upper and one lower index.

The most basic relationship between image entities in three views concerns a correspondence between two lines and a point. We consider a correspondence $\mathbf{x} \leftrightarrow \mathbf{l}' \leftrightarrow \mathbf{l}''$ between a point x in one image and two lines $\mathbf{l}'$ and $\mathbf{l}''$ in the other two images. This relationship means that there is a point X in space that maps to x in the first image, and to points $\mathbf{x}'$ and $\mathbf{x}''$ lying on the lines $\mathbf{l}'$ and $\mathbf{l}''$ in the other two images. The coordinates of these three images are then related via the trifocal tensor relationship:

$$\sum_{ijk} x^i l'_j l''_k \mathcal{T}_i^{jk} = 0. \tag{1.2}$$

This relationship gives a single linear relationship between the elements of the tensor. With sufficiently many such correspondences, it is possible to solve linearly for the

Samsung/Google, EX1010, p. 30

elements of the tensor. Fortunately, one can obtain more equations from a point correspondence $\mathbf{x} \leftrightarrow \mathbf{x}' \leftrightarrow \mathbf{x}''$. In fact, in this situation, one can choose any lines $\mathbf{l}'$ and $\mathbf{l}''$ passing through the points $\mathbf{x}'$ and $\mathbf{x}''$ and generate a relation of the sort (1.2). Since it is possible to choose two independent lines passing through $\mathbf{x}'$, and two others passing through $\mathbf{x}''$, one can obtain four independent equations in this way. A total of seven point correspondences are sufficient to compute the trifocal tensor linearly in this way. It can be computed from a minimum of six point correspondences using a non-linear method.

The 27 elements of the tensor are not independent, however, but are related by a set of so called *internal constraints*. These constraints are quite complicated, but tensors satisfying the constraints can be computed in various ways, for instance by using the 6 point non-linear method. The fundamental matrix (which is a 2-view tensor) also satisfies an internal constraint but a relatively simple one: the elements obey $\det \mathbf{F} = 0$.

As with the fundamental matrix, once the trifocal tensor is known, it is possible to extract the three camera matrices from it, and thereby obtain a reconstruction of the scene points and lines. As ever, this reconstruction is unique only up to a 3D projective transformation; it is a projective reconstruction.

Thus, we are able to generalize the method for two views to three views. There are several advantages to using such a three-view method for reconstruction.

(i) It is possible to use a mixture of line and point correspondences to compute the projective reconstruction. With two views, only point correspondences can be used.

(ii) Using three views gives greater stability to the reconstruction, and avoids unstable configurations that may occur using only two views for the reconstruction.

1.5 Four view geometry and n-view reconstruction

It is possible to go one more step with tensor-based methods and define a quadrifocal tensor relating entities visible in four views. This method is seldom used, however, because of the relative difficulty of computing a quadrifocal tensor that obey its internal constraints. Nevertheless, it does provide a non-iterative method for computing a projective reconstruction based on four views. The tensor method does not extend to more than four views, however, and so reconstruction from more than four views becomes more difficult.

Many methods have been considered for reconstruction from several views, and we consider a few of these in the book. One way to proceed is to reconstruct the scene bit by bit, using three-view or two-view techniques. Such a method may be applied to any image sequence, and with care in selecting the right triples to use, it will generally succeed.

There are methods that can be used in specific circumstances. The task of reconstruction becomes easier if we are able to apply a simpler camera model, known as the *affine camera*. This camera model is a fair approximation to perspective projection whenever the distance to the scene is large compared with the difference in depth between the back and front of the scene. If a set of points are visible in all of a set of n views

Samsung/Google, EX1010, p. 31

involving an affine camera, then a well-known algorithm, the *factorization algorithm*, can be used to compute both the structure of the scene, and the specific camera models in one step using the Singular Value Decomposition. This algorithm is very reliable and simple to implement. Its main difficulties are the use of the affine camera model, rather than a full projective model, and the requirement that all the points be visible in all views.

This method has been extended to projective cameras in a method known as projective factorization. Although this method is generally satisfactory, it can not be proven to converge to the correct solution in all cases. Besides, it also requires all points to be visible in all images.

Other methods for n-view reconstruction involve various assumptions, such as knowledge of four coplanar points in the world visible in all views, or six or seven points that are visible in all images in the sequence. Methods that apply to specific motion sequences, such as linear motion, planar motion or single axis (turntable) motion have also been developed.

The dominant methodology for the general reconstruction problem is *bundle adjustment*. This is an iterative method, in which one attempts to fit a non-linear model to the measured data (the point correspondences). The advantage of bundle-adjustment is that it is a very general method that may be applied to a wide range of reconstruction and optimization problems. It may be implemented in such a way that the discovered solution is the Maximum Likelihood solution to the problem, that is a solution that is in some sense optimal in terms of a model for the inaccuracies of image measurements.

Unfortunately, bundle adjustment is an iterative process, which can not be guaranteed to converge to the optimal solution from an arbitrary starting point. Much research in reconstruction methods seeks easily computable non-optimal solutions that can be used as a starting point for bundle adjustment. An initialization step followed by bundle adjustment is the generally preferred technique for reconstruction. A common impression is that bundle-adjustment is necessarily a slow technique. The truth is that it is quite efficient when implemented carefully. A lengthy appendix in this book deals with efficient methods of bundle adjustment.

Using n-view reconstruction techniques, it is possible to carry out reconstructions automatically from quite long sequences of images. An example is given in figure 1.3, showing a reconstruction from 700 frames.

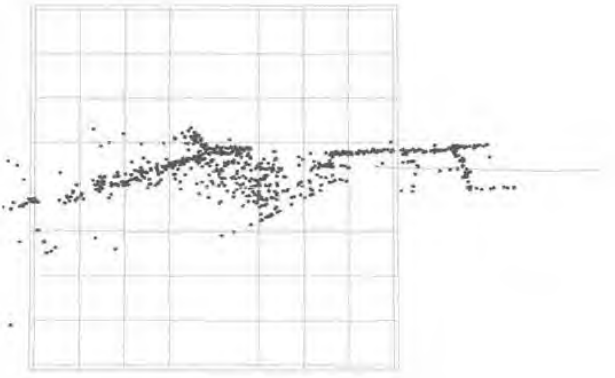
1.6 Transfer

We have discussed 3D reconstruction from a set of images. Another useful application of projective geometry is that of *transfer*: given the position of a point in one (or more) image(s), determine where it will appear in all other images of the set. To do this, we must first establish the relationship between the cameras using (for instance) a set of auxiliary point correspondences. Conceptually transfer is straightforward given that a reconstruction is possible. For instance, suppose the point is identified in two views (at $\mathbf{x}$ and $\mathbf{x}'$) and we wish to know its position $\mathbf{x}''$ in a third, then this may be computed by the following steps:

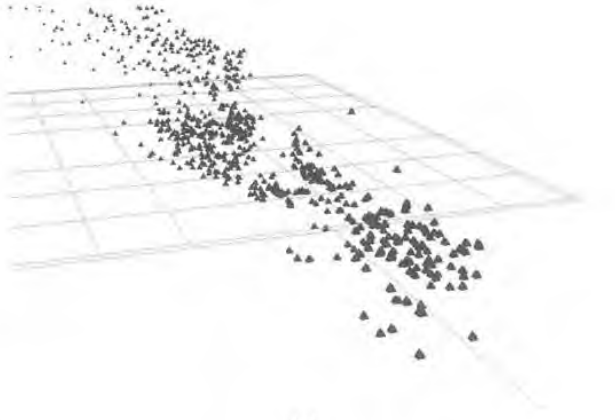
Samsung/Google, EX1010, p. 32



(a)



(b)



(c)

Fig. 1.3. **Reconstruction.** *(a) Seven frames of a 700 frame sequence acquired by a hand held camera whilst walking down a street in Oxford. (b)(c) Two views of the reconstructed point cloud and camera path (the red curve). Figures courtesy of David Capel and 2d3 (*`www.2d3.com`*).*

Samsung/Google, EX1010, p. 33



Fig. 1.4. **Projective ambiguity:** *Reconstructions of a mug (shown with the true shape in the centre) under 3D projective transformations in the Z direction. Five examples of the cup with different degrees of projective distortion are shown. The shapes are quite different from the original.*

(i) Compute the camera matrices of the three views $\mathtt{P}, \mathtt{P}', \mathtt{P}''$ from other point correspondences $\mathbf{x}_i \leftrightarrow \mathbf{x}'_i \leftrightarrow \mathbf{x}''_i$.
(ii) Triangulate the 3D point $\mathbf{X}$ from $\mathbf{x}$ and $\mathbf{x}'$ using $\mathtt{P}$ and $\mathtt{P}'$.
(iii) Project the 3D point into the third view as $\mathbf{x}'' = \mathtt{P}''\mathbf{X}$.

This procedure only requires projective information. An alternative procedure is to use the multi-view tensors (the fundamental matrix and trifocal tensor) to transfer the point directly without an explicit 3D reconstruction. Both methods have their advantages.

Suppose the camera rotates about its centre or that all the scene points of interest lie on a plane. Then the appropriate multiple view relations are the planar projective transformations between the images. In this case, a point seen in just one image can be transferred to any other image.

1.7 Euclidean reconstruction

So far we have considered the reconstruction of a scene, or transfer, for images taken with a set of uncalibrated cameras. For such cameras, important parameters such as the focal length, the geometric centre of the image (the principal point) and possibly the aspect ratio of the pixels in the image are unknown. If a complete calibration of each of the cameras is known then it is possible to remove some of the ambiguity of the reconstructed scene.

So far, we have discussed projective reconstruction, which is all that is possible without knowing something about the calibration of the cameras or the scene. Projective reconstruction is insufficient for many purposes, such as application to computer graphics, since it involves distortions of the model that appear strange to a human used to viewing a Euclidean world. For instance, the distortions that projective transformations induce in a simple object are shown in figure 1.4. Using the technique of projective reconstruction, there is no way to choose between any of the possible shapes of the mug in figure 1.4, and a projective reconstruction algorithm is as likely to come up with any one of the reconstructions shown there as any other. Even more severely distorted models may arise from projective reconstruction.

In order to obtain a reconstruction of the model in which objects have their correct (Euclidean) shape, it is necessary to determine the calibration of the cameras. It is easy to see that this is sufficient to determine the Euclidean structure of the scene. As we have seen, determining the Euclidean structure of the world is equivalent to specifying the plane at infinity and the absolute conic. In fact, since the absolute conic

Samsung/Google, EX1010, p. 34

lies in a plane, the plane at infinity, it is enough to find the absolute conic in space. Now, suppose that we have computed a projective reconstruction of the world, using calibrated cameras. By definition, this means that the IAC is known in each of the images; let it be denoted by ω_i in the i-th image. The back-projection of each ω_i is a cone in space, and the absolute conic must lie in the intersection of all the cones. Two cones in general intersect in a fourth-degree curve, but given that they must intersect in a conic, this curve must split into two conics. Thus, reconstruction of the absolute conic from two images is not unique – rather, there are two possible solutions in general. However, from three or more images, the intersection of the cones is unique in general. Thus the absolute conic is determined and with it the Euclidean structure of the scene.

Of course, if the Euclidean structure of the scene is known, then so is the position of the absolute conic. In this case we may project it back into each of the images, producing the IAC in each image, and hence calibrating the cameras. Thus knowledge of the camera calibration is equivalent to being able to determine the Euclidean structure of the scene.

1.8 Auto-calibration

Without any knowledge of the calibration of the cameras, it is impossible to do better than projective reconstruction. There is no information in a set of feature correspondences across any number of views that can help us find the image of the absolute conic, or equivalently the calibration of the cameras. However, if we know just a little about the calibration of the cameras then we may be able to determine the position of the absolute conic.

Suppose, for instance that it is known that the calibration is the same for each of the cameras used in reconstructing a scene from an image sequence. By this we mean the following. In each image a coordinate system is defined, in which we have measured the image coordinates of corresponding features used to do projective reconstruction. Suppose that in all these image coordinate systems, the IAC is the same, but just where it is located is unknown. From this knowledge, we wish to compute the position of the absolute conic.

One way to find the absolute conic is to hypothesize the position of the IAC in one image; by hypothesis, its position in the other images will be the same. The back-projection of each of the conics will be a cone in space. If the three cones all meet in a single conic, then this must be a possible solution for the position of the absolute conic, consistent with the reconstruction.

Note that this is a conceptual description only. The IAC is of course a conic containing only complex points, and its back-projection will be a complex cone. However, algebraically, the problem is more tractable. Although it is complex, the IAC may be described by a real quadratic form (represented by a real symmetric matrix). The back-projected cone is also represented by a real quadratic form. For some value of the IAC, the three back-projected cones will meet in a conic curve in space.

Generally given three cameras known to have the same calibration, it is possible to determine the absolute conic, and hence the calibration of the cameras. However,

Samsung/Google, EX1010, p. 35

although various methods have been proposed for this, it remains quite a difficult problem.

Knowing the plane at infinity. One method of auto-calibration is to proceed in steps by first determining the plane on which it lies. This is equivalent to identifying the plane at infinity in the world, and hence to determining the affine geometry of the world. In a second step, one locates the position of the absolute conic on the plane to determine the Euclidean geometry of space. Assuming one knows the plane at infinity, one can back-project a hypothesised IAC from each of a sequence of images and intersect the resulting cones with the plane at infinity. If the IAC is chosen correctly, the intersection curve is the absolute conic. Thus, from each pair of images one has a condition that the back-projected cones meet in the same conic curve on the plane at infinity. It turns out that this gives a linear constraint on the entries of the matrix representing the IAC. From a set of linear equations, one can determine the IAC, and hence the absolute conic. Thus, auto-calibration is relatively simple, once the plane at infinity has been identified. The identification of the plane at infinity itself is substantially more difficult.

Auto-calibration given square pixels in the image. If the cameras are partially calibrated, then it is possible to complete the calibration starting from a projective reconstruction. One can make do with quite minimal conditions on the calibration of the cameras, represented by the IAC. One interesting example is the square-pixel constraint on the cameras. What this means is that a Euclidean coordinate system is known in each image. In this case, the absolute conic, lying in the plane at infinity in the world must meet the image plane in its two circular points. The circular points in a plane are the two points where the absolute conic meets that plane. The back-projected rays through the circular points of the image plane must intersect the absolute conic. Thus, each image with square pixels determines two rays that must meet the absolute conic. Given n images, the autocalibration task then becomes that of determining a space conic (the absolute conic) that meets a set of $2n$ rays in space. An equivalent geometric picture is to intersect the set of rays with a plane and require that the set of intersection points lie on a conic. By a simple counting argument one may see that there are only a finite number of conics that meet eight prescribed rays in space. Therefore, from four images one may determine the calibration, albeit up to a finite number of possibilities.

1.9 The reward I : 3D graphical models

We have now described all the ingredients necessary to compute realistic graphics models from image sequences. From point matches between images, it is possible to carry out first a projective reconstruction of the point set, and determine the motion of the camera in the chosen projective coordinate frame.

Using auto-calibration techniques, assuming some restrictions on the calibration of the camera that captured the image sequence, the camera may be calibrated, and the scene subsequently transformed to its true Euclidean structure.

Samsung/Google, EX1010, p. 36

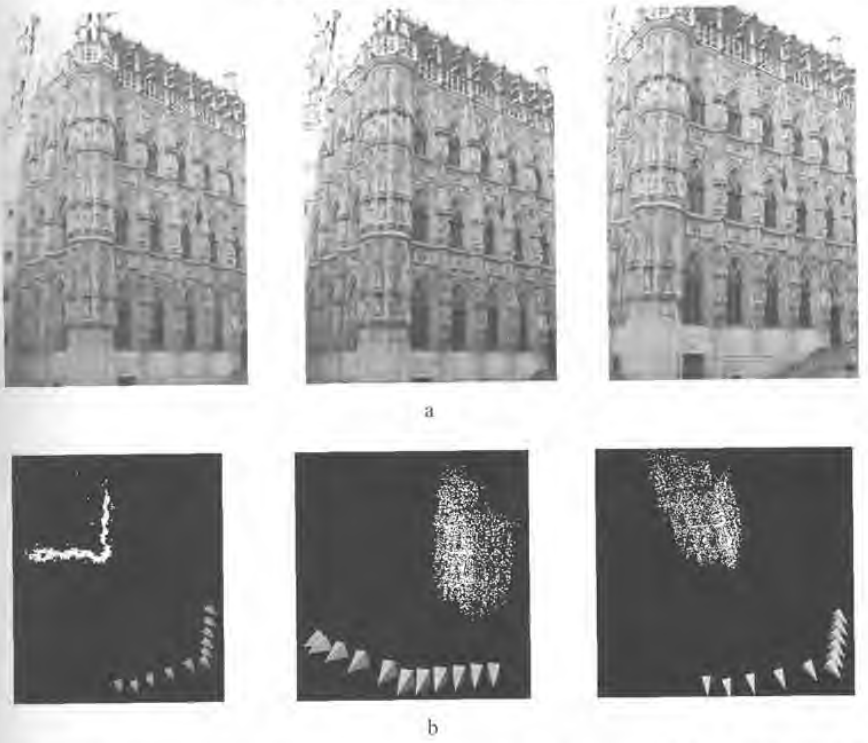


Fig. 1.5. *(a) Three high resolution images (3000 × 2000 pixels) from a set of eleven of the cityhall in Leuven, Belgium. (b) Three views of a Euclidean reconstruction computed from the image set showing the 11 camera positions and point cloud.*

Knowing the projective structure of the scene, it is possible to find the epipolar geometry relating pairs of images and this restricts the correspondence search for further matches to a line – a point in one image defines a line in the other image on which the (as yet unknown) corresponding point must lie. In fact for suitable scenes, it is possible to carry out a dense point match between images and create a dense 3D model of the imaged scene. This takes the form of a triangulated shape model that is subsequently shaded or texture-mapped from the supplied images and used to generate novel views. The steps of this process are illustrated in figure 1.5 and figure 1.6.

1.10 The reward II: video augmentation

We finish this introduction with a further application of reconstruction methods to computer graphics. Automatic reconstruction techniques have recently become widely used in the film industry as a means for adding artificial graphics objects in real video sequences. Computer analysis of the motion of the camera is replacing the previously used manual methods for correctly aligning the artificial inserted object.

The most important requirement for realistic insertion of an artificial object in a video

Samsung/Google, EX1010, p. 37

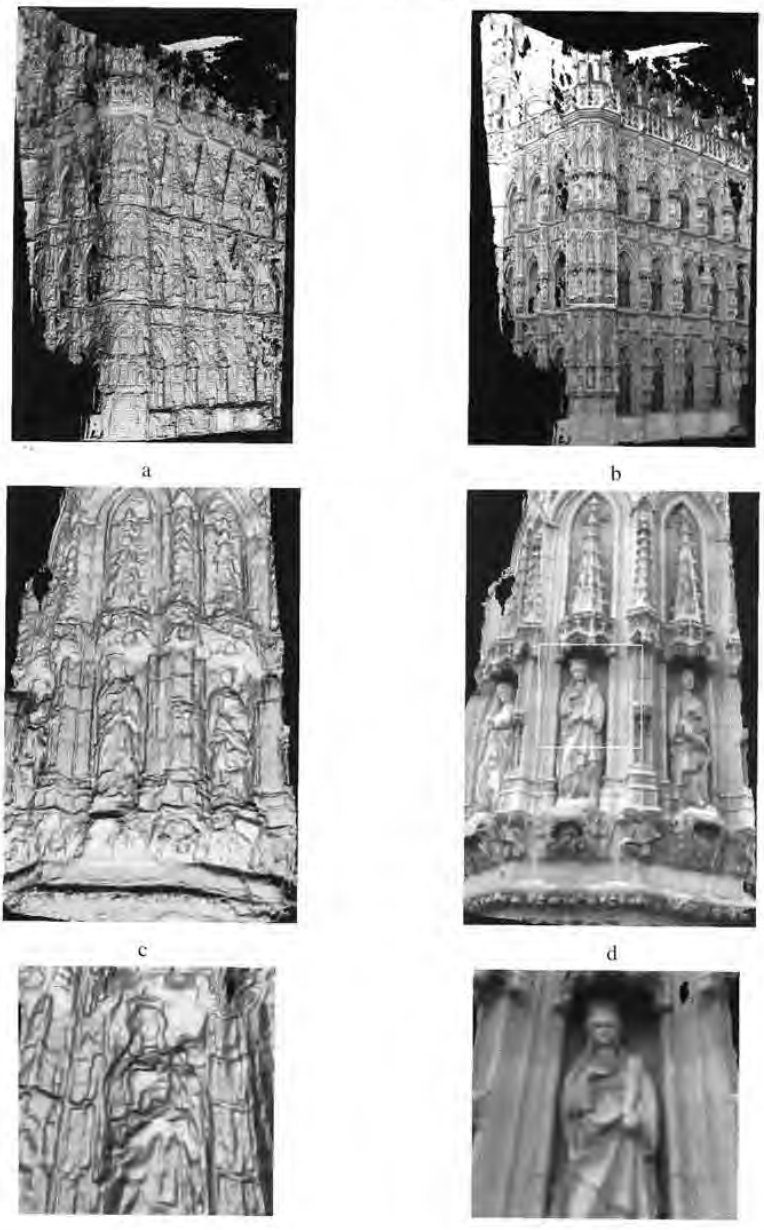


Fig. 1.6. **Dense reconstructions.** *These are computed from the cameras and image of figure 1.5. (a) Untextured and (b) textured reconstruction of the full scene. (c) Untextured and (d) textured close up of the area shown in the white rectangle of (b). (e) Untextured and (f) textured close up of the area shown in the white rectangle of (d). The dense surface is computed using the three-view stereo algorithm described in [Strecha-02]. Figures courtesy of Christoph Strecha, Frank Verbiest, and Luc Van Gool.*

Samsung/Google, EX1010, p. 38



Fig. 1.7. **Augmented video.** *The animated robot is inserted into the scene and rendered using the computed cameras of figure 1.3. (a)-(c) Original frames from the sequence. (d)-(f) The augmented frames. Figures courtesy of 2d3 (*www.2d3.com*).*

sequence is to compute the correct motion of the camera. Unless the camera motion is correctly determined, it is impossible to generate the correct sequences of views of the graphics model in a way that will appear consistent with the background video. Generally, it is only the motion of the camera that is important here; we do not need to reconstruct the scene, since it is already present in the existing video, and novel views of the scene visible in the video are not required. The only requirement is to be able to generate correct perspective views of the graphics model.

It is essential to compute the motion of the camera in a Euclidean frame. It is not enough merely to know the projective motion of the camera. This is because a Euclidean object is to be placed in the scene. Unless this graphics object and the cameras are known in the same coordinate frame, then generated views of the inserted object will be seen to distort with respect to the perceived structure of the scene seen in the existing video.

Once the correct motion of the camera, and its calibration are known the inserted object may be rendered into the scene in a realistic manner. If the change of the camera calibration from frame to frame is correctly determined, then the camera may change focal length (zoom) during the sequence. It is even possible for the principal point to vary during the sequence through cropping.

In inserting the rendered model into the video, the task is relatively straight-forward if it lies in front of all the existing scene. Otherwise the possibility of occlusions arises, in which the scene may obscure parts of the model. An example of video augmentation is shown in figure 1.7.

Samsung/Google, EX1010, p. 39

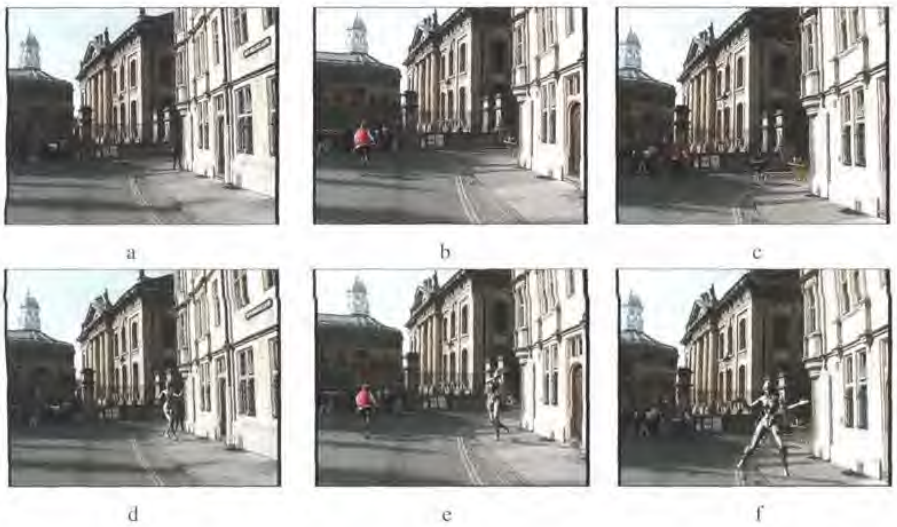


Fig. 1.7. **Augmented video.** *The animated robot is inserted into the scene and rendered using the computed cameras of figure 1.3. (a)-(c) Original frames from the sequence. (d)-(f) The augmented frames. Figures courtesy of 2d3* (www.2d3.com).

sequence is to compute the correct motion of the camera. Unless the camera motion is correctly determined, it is impossible to generate the correct sequences of views of the graphics model in a way that will appear consistent with the background video. Generally, it is only the motion of the camera that is important here; we do not need to reconstruct the scene, since it is already present in the existing video, and novel views of the scene visible in the video are not required. The only requirement is to be able to generate correct perspective views of the graphics model.

It is essential to compute the motion of the camera in a Euclidean frame. It is not enough merely to know the projective motion of the camera. This is because a Euclidean object is to be placed in the scene. Unless this graphics object and the cameras are known in the same coordinate frame, then generated views of the inserted object will be seen to distort with respect to the perceived structure of the scene seen in the existing video.

Once the correct motion of the camera, and its calibration are known the inserted object may be rendered into the scene in a realistic manner. If the change of the camera calibration from frame to frame is correctly determined, then the camera may change focal length (zoom) during the sequence. It is even possible for the principal point to vary during the sequence through cropping.

In inserting the rendered model into the video, the task is relatively straight-forward if it lies in front of all the existing scene. Otherwise the possibility of occlusions arises, in which the scene may obscure parts of the model. An example of video augmentation is shown in figure 1.7.

Samsung/Google, EX1010, p. 40

Part 0

The Background: Projective Geometry, Transformations and Estimation



La reproduction interdite (The Forbidden Reproduction), 1937, René Magritte.
Courtesy of Museum Boijmans van Beuningen, Rotterdam.
© ADAGP, Paris, and DACS, London 2000.

Samsung/Google, EX1010, p. 41

Outline

The four chapters in this part lay the foundation for the representations, terminology, and notation that will be used in the subsequent parts of the book. The ideas and notation of projective geometry are central to an analysis of multiple view geometry. For example, the use of homogeneous coordinates enables non-linear mappings (such as perspective projection) to be represented by linear matrix equations, and points at infinity to be represented quite naturally avoiding the awkward necessity of taking limits.

Chapter 2 introduces projective transformations of 2-space. These are the transformations that arise when a plane is imaged by a perspective camera. This chapter is more introductory and sets the scene for the geometry of 3-space. Most of the concepts can be more easily understood and visualized in 2D than in 3D. Specializations of projective transformations are introduced, including affine and similarity transformations. Particular attention is focussed on the recovery of affine properties (e.g. parallel lines) and metric properties (e.g. angles between lines) from a perspective image.

Chapter 3 covers the projective geometry of 3-space. This geometry develops in much the same manner as that of 2-space, though of course there are extra properties arising from the additional dimension. The main new geometry here is the plane at infinity and the absolute conic.

Chapter 4 introduces *estimation* of geometry from image measurements, which is one of the main topics of this book. The example of estimating a projective transformation from point correspondences is used to illustrate the basis and motivation for the algorithms that will be used throughout the book. The important issue of what should be minimized in a cost function, e.g. algebraic or geometric or statistical measures, is described at length. The chapter also introduces the idea of robust estimation, and the use of such techniques in the automatic estimation of transformations.

Chapter 5 describes how the results of estimation algorithms may be evaluated. In particular how the covariance of an estimation may be computed.

Samsung/Google, EX1010, p. 42

2

Projective Geometry and Transformations of 2D

This chapter introduces the main geometric ideas and notation that are required to understand the material covered in this book. Some of these ideas are relatively familiar, such as vanishing point formation or representing conics, whilst others are more esoteric, such as using circular points to remove perspective distortion from an image. These ideas can be understood more easily in the planar (2D) case because they are more easily visualized here. The geometry of 3-space, which is the subject of the later parts of this book, is only a simple generalization of this planar case.

In particular, the chapter covers the geometry of projective transformations of the plane. These transformations model the geometric distortion which arises when a plane is imaged by a perspective camera. Under perspective imaging certain geometric properties are preserved, such as collinearity (a straight line is imaged as a straight line), whilst others are not, for example parallel lines are not imaged as parallel lines in general. Projective geometry models this imaging and also provides a mathematical representation appropriate for computations.

We begin by describing the representation of points, lines and conics in homogeneous notation, and how these entities map under projective transformations. The line at infinity and the circular points are introduced, and it is shown that these capture the affine and metric properties of the plane. Algorithms for rectifying planes are then given which enable affine and metric properties to be computed from images. We end with a description of fixed points under projective transformations.

2.1 Planar geometry

The basic concepts of planar geometry are familiar to anyone who has studied mathematics even at an elementary level. In fact, they are so much a part of our everyday experience that we take them for granted. At an elementary level, geometry is the study of points and lines and their relationships.

To the purist, the study of geometry ought properly to be carried out from a "geometric" or coordinate-free viewpoint. In this approach, theorems are stated and proved in terms of geometric primitives only, without the use of algebra. The classical approach of Euclid is an example of this method. Since Descartes, however, it has been seen that geometry may be algebraicized, and indeed the theory of geometry may be developed

Samsung/Google, EX1010, p. 43

from an algebraic viewpoint. Our approach in this book will be a hybrid approach, sometimes using geometric, and sometimes algebraic methods. In the algebraic approach, geometric entities are described in terms of coordinates and algebraic entities. Thus, for instance a point is identified with a vector in terms of some coordinate basis. A line is also identified with a vector, and a conic section (more briefly, a conic) is represented by a symmetric matrix. In fact, we often carry this identification so far as to consider that the vector actually *is* a point, or the symmetric matrix *is* a conic, at least for convenience of language. A significant advantage of the algebraic approach to geometry is that results derived in this way may more easily be used to derive algorithms and practical computational methods. Computation and algorithms are a major concern in this book, which justifies the use of the algebraic method.

2.2 The 2D projective plane

As we all know, a point in the plane may be represented by the pair of coordinates (x, y) in $\mathbb{R}^2$. Thus, it is common to identify the plane with $\mathbb{R}^2$. Considering $\mathbb{R}^2$ as a vector space, the coordinate pair (x, y) is a vector – a point is identified as a vector. In this section we introduce the *homogeneous* notation for points and lines on a plane.

Row and column vectors. Later on, we will want to consider linear mappings between vector spaces, and represent such mappings as matrices. In the usual manner, the product of a matrix and a vector is another vector, the image under the mapping. This brings up the distinction between "column" and "row" vectors, since a matrix may be multiplied on the right by a column and on the left by a row vector. Geometric entities will by default be represented by column vectors. A bold-face symbol such as $\mathbf{x}$ always represents a column vector, and its transpose is the row vector $\mathbf{x}^\top$. In accordance with this convention, a point in the plane will be represented by the column vector $(x, y)^\top$, rather than its transpose, the row vector (x, y). We write $\mathbf{x} = (x, y)^\top$, both sides of this equation representing column vectors.

2.2.1 Points and lines

Homogeneous representation of lines. A line in the plane is represented by an equation such as $ax+by+c = 0$, different choices of a, b and c giving rise to different lines. Thus, a line may naturally be represented by the vector $(a, b, c)^\top$. The correspondence between lines and vectors $(a, b, c)^\top$ is not one-to-one, since the lines $ax + by + c = 0$ and $(ka)x + (kb)y + (kc) = 0$ are the same, for any non-zero constant k. Thus, the vectors $(a, b, c)^\top$ and $k(a, b, c)^\top$ represent the same line, for any non-zero k. In fact, two such vectors related by an overall scaling are considered as being equivalent. An equivalence class of vectors under this equivalence relationship is known as a *homogeneous* vector. Any particular vector $(a, b, c)^\top$ is a representative of the equivalence class. The set of equivalence classes of vectors in $\mathbb{R}^3 - (0, 0, 0)^\top$ forms the *projective space* $\mathbb{P}^2$. The notation $-(0, 0, 0)^\top$ indicates that the vector $(0, 0, 0)^\top$, which does not correspond to any line, is excluded.

Samsung/Google, EX1010, p. 44

Homogeneous representation of points. A point $\mathbf{x} = (x, y)^\top$ lies on the line $\mathbf{l} = (a, b, c)^\top$ if and only if $ax + by + c = 0$. This may be written in terms of an inner product of vectors representing the point as $(x, y, 1)(a, b, c)^\top = (x, y, 1)\mathbf{l} = 0$; that is the point $(x, y)^\top$ in $\mathbb{R}^2$ is represented as a 3-vector by adding a final coordinate of 1. Note that for any non-zero constant k and line $\mathbf{l}$ the equation $(kx, ky, k)\mathbf{l} = 0$ if and only if $(x, y, 1)\mathbf{l} = 0$. It is natural, therefore, to consider the set of vectors $(kx, ky, k)^\top$ for varying values of k to be a representation of the point $(x, y)^\top$ in $\mathbb{R}^2$. Thus, just as with lines, points are represented by homogeneous vectors. An arbitrary homogeneous vector representative of a point is of the form $\mathbf{x} = (x_1, x_2, x_3)^\top$, representing the point $(x_1/x_3, x_2/x_3)^\top$ in $\mathbb{R}^2$. Points, then, as homogeneous vectors are also elements of $\mathbb{P}^2$.

One has a simple equation to determine when a point lies on a line, namely

Result 2.1. *The point* $\mathbf{x}$ *lies on the line* $\mathbf{l}$ *if and only if* $\mathbf{x}^\top\mathbf{l} = 0$.

Note that the expression $\mathbf{x}^\top\mathbf{l}$ is just the inner or scalar product of the two vectors $\mathbf{l}$ and $\mathbf{x}$. The scalar product $\mathbf{x}^\top\mathbf{l} = \mathbf{l}^\top\mathbf{x} = \mathbf{x}.\mathbf{l}$. In general, the transpose notation $\mathbf{l}^\top\mathbf{x}$ will be preferred, but occasionally, we will use a . to denote the inner product. We distinguish between the *homogeneous coordinates* $\mathbf{x} = (x_1, x_2, x_3)^\top$ of a point, which is a 3-vector, and the *inhomogeneous* coordinates $(x, y)^\top$, which is a 2-vector.

Degrees of freedom (dof). It is clear that in order to specify a point two values must be provided, namely its x- and y-coordinates. In a similar manner a line is specified by two parameters (the two independent ratios $\{a : b : c\}$) and so has two degrees of freedom. For example, in an inhomogeneous representation, these two parameters could be chosen as the gradient and y intercept of the line.

Intersection of lines. Given two lines $\mathbf{l} = (a, b, c)^\top$ and $\mathbf{l}' = (a', b', c')^\top$, we wish to find their intersection. Define the vector $\mathbf{x} = \mathbf{l} \times \mathbf{l}'$, where $\times$ represents the vector or cross product. From the triple scalar product identity $\mathbf{l}.(\mathbf{l} \times \mathbf{l}') = \mathbf{l}'.(\mathbf{l} \times \mathbf{l}') = 0$, we see that $\mathbf{l}^\top\mathbf{x} = \mathbf{l}'^\top\mathbf{x} = 0$. Thus, if $\mathbf{x}$ is thought of as representing a point, then $\mathbf{x}$ lies on both lines $\mathbf{l}$ and $\mathbf{l}'$, and hence is the intersection of the two lines. This shows:

Result 2.2. *The intersection of two lines* $\mathbf{l}$ *and* $\mathbf{l}'$ *is the point* $\mathbf{x} = \mathbf{l} \times \mathbf{l}'$.

Note that the simplicity of this expression for the intersection of the two lines is a direct consequence of the use of homogeneous vector representations of lines and points.

Example 2.3. Consider the simple problem of determining the intersection of the lines $x = 1$ and $y = 1$. The line $x = 1$ is equivalent to $-1x + 1 = 0$, and thus has homogeneous representation $\mathbf{l} = (-1, 0, 1)^\top$. The line $y = 1$ is equivalent to $-1y + 1 = 0$, and thus has homogeneous representation $\mathbf{l}' = (0, -1, 1)^\top$. From result 2.2 the intersection point is

$$\mathbf{x} = \mathbf{l} \times \mathbf{l}' = \begin{vmatrix} \mathbf{i} & \mathbf{j} & \mathbf{k} \\ -1 & 0 & 1 \\ 0 & -1 & 1 \end{vmatrix} = \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix}$$

which is the inhomogeneous point $(1, 1)^\top$ as required. △

Samsung/Google, EX1010, p. 45

Line joining points. An expression for the line passing through two points $\mathbf{x}$ and $\mathbf{x}'$ may be derived by an entirely analogous argument. Defining a line $\mathbf{l}$ by $\mathbf{l} = \mathbf{x} \times \mathbf{x}'$, it may be verified that both points $\mathbf{x}$ and $\mathbf{x}'$ lie on $\mathbf{l}$. Thus

Result 2.4. *The line through two points* $\mathbf{x}$ *and* $\mathbf{x}'$ *is* $\mathbf{l} = \mathbf{x} \times \mathbf{x}'$.

2.2.2 Ideal points and the line at infinity

Intersection of parallel lines. Consider two lines $ax+by+c = 0$ and $ax+by+c' = 0$. These are represented by vectors $\mathbf{l} = (a, b, c)^\top$ and $\mathbf{l}' = (a, b, c')^\top$ for which the first two coordinates are the same. Computing the intersection of these lines gives no difficulty, using result 2.2. The intersection is $\mathbf{l} \times \mathbf{l}' = (c' - c)(b, -a, 0)^\top$, and ignoring the scale factor $(c' - c)$, this is the point $(b, -a, 0)^\top$.

Now if we attempt to find the inhomogeneous representation of this point, we obtain $(b/0, -a/0)^\top$, which makes no sense, except to suggest that the point of intersection has infinitely large coordinates. In general, points with homogeneous coordinates $(x, y, 0)^\top$ do not correspond to any finite point in $\mathbb{R}^2$. This observation agrees with the usual idea that parallel lines meet at infinity.

Example 2.5. Consider the two lines $x = 1$ and $x = 2$. Here the two lines are parallel, and consequently intersect "at infinity". In homogeneous notation the lines are $\mathbf{l} = (-1, 0, 1)^\top$, $\mathbf{l}' = (-1, 0, 2)^\top$, and from result 2.2 their intersection point is

$$\mathbf{x} = \mathbf{l} \times \mathbf{l}' = \begin{vmatrix} \mathbf{i} & \mathbf{j} & \mathbf{k} \\ -1 & 0 & 1 \\ -1 & 0 & 2 \end{vmatrix} = \begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix}$$

which is the point at infinity in the direction of the y-axis. △

Ideal points and the line at infinity. Homogeneous vectors $\mathbf{x} = (x_1, x_2, x_3)^\top$ such that $x_3 \neq 0$ correspond to finite points in $\mathbb{R}^2$. One may augment $\mathbb{R}^2$ by adding points with last coordinate $x_3 = 0$. The resulting space is the set of all homogeneous 3-vectors, namely the projective space $\mathbb{P}^2$. The points with last coordinate $x_3 = 0$ are known as *ideal* points, or points at infinity. The set of all ideal points may be written $(x_1, x_2, 0)^\top$, with a particular point specified by the ratio $x_1 : x_2$. Note that this set lies on a single line, the *line at infinity*, denoted by the vector $\mathbf{l}_\infty = (0, 0, 1)^\top$. Indeed, one verifies that $(0, 0, 1)(x_1, x_2, 0)^\top = 0$.

Using result 2.2 one finds that a line $\mathbf{l} = (a, b, c)^\top$ intersects $\mathbf{l}_\infty$ in the ideal point $(b, -a, 0)^\top$ (since $(b, -a, 0)\mathbf{l} = 0$). A line $\mathbf{l}' = (a, b, c')^\top$ *parallel* to $\mathbf{l}$ intersects $\mathbf{l}_\infty$ in the same ideal point $(b, -a, 0)^\top$ irrespective of the value of c'. In inhomogeneous notation $(b, -a)^\top$ is a vector tangent to the line, and orthogonal to the line normal (a, b), and so represents the line's *direction*. As the line's direction varies the ideal point $(b, -a, 0)^\top$ varies over $\mathbf{l}_\infty$. For these reasons the line at infinity can be thought of as the set of directions of lines in the plane.

Note how the introduction of the concept of points at infinity serves to simplify the intersection properties of points and lines. In the projective plane $\mathbb{P}^2$, one may state without qualification that two distinct lines meet in a single point and two distinct

Samsung/Google, EX1010, p. 46

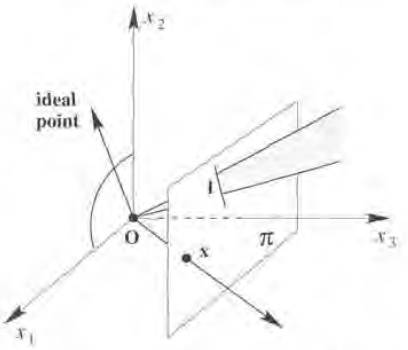


Fig. 2.1. **A model of the projective plane.** *Points and lines of* $\mathbb{P}^2$ *are represented by rays and planes, respectively, through the origin in* $\mathbb{R}^3$. *Lines lying in the* x_1x_2*-plane represent ideal points, and the* x_1x_2*-plane represents* $\mathbf{l}_\infty$.

points lie on a single line. This is not true in the standard Euclidean geometry of $\mathbb{R}^2$, in which parallel lines form a special case.

The study of the geometry of $\mathbb{P}^2$ is known as projective geometry. In a coordinate-free purely geometric study of projective geometry, one does not make any distinction between points at infinity (ideal points) and ordinary points. It will, however, serve our purposes in this book sometimes to distinguish between ideal points and non-ideal points. Thus, the line at infinity will at times be considered as a special line in projective space.

A model for the projective plane. A fruitful way of thinking of $\mathbb{P}^2$ is as a set of rays in $\mathbb{R}^3$. The set of all vectors $k(x_1, x_2, x_3)^\mathsf{T}$ as k varies forms a ray through the origin. Such a ray may be thought of as representing a single point in $\mathbb{P}^2$. In this model, the lines in $\mathbb{P}^2$ are planes passing through the origin. One verifies that two non-identical rays lie on exactly one plane, and any two planes intersect in one ray. This is the analogue of two distinct points uniquely defining a line, and two lines always intersecting in a point.

Points and lines may be obtained by intersecting this set of rays and planes by the plane $x_3 = 1$. As illustrated in figure 2.1 the rays representing ideal points and the plane representing $\mathbf{l}_\infty$ are parallel to the plane $x_3 = 1$.

Duality. The reader has probably noticed how the role of points and lines may be interchanged in statements concerning the properties of lines and points. In particular, the basic incidence equation $\mathbf{l}^\mathsf{T}\mathbf{x} = 0$ for line and point is symmetric, since $\mathbf{l}^\mathsf{T}\mathbf{x} = 0$ implies $\mathbf{x}^\mathsf{T}\mathbf{l} = 0$, in which the positions of line and point are swapped. Similarly, result 2.2 and result 2.4 giving the intersection of two lines and the line through two points are essentially the same, with the roles of points and lines swapped. One may enunciate a general principle, the *duality principle* as follows:

Samsung/Google, EX1010, p. 47

Result 2.6. Duality principle. *To any theorem of 2-dimensional projective geometry there corresponds a dual theorem, which may be derived by interchanging the roles of points and lines in the original theorem.*

In applying this principle, concepts of incidence must be appropriately translated as well. For instance, the line through two points is dual to the point through (that is the point of intersection of) two lines.

Note that is it not necessary to prove the dual of a given theorem once the original theorem has been proved. The proof of the dual theorem will be the dual of the proof of the original theorem.

2.2.3 Conics and dual conics

A conic is a curve described by a second-degree equation in the plane. In Euclidean geometry conics are of three main types: hyperbola, ellipse, and parabola (apart from so-called degenerate conics, to be defined later). Classically these three types of conic arise as conic sections generated by planes of differing orientation (the degenerate conics arise from planes which contain the cone vertex). However, it will be seen that in 2D projective geometry all non-degenerate conics are equivalent under projective transformations.

The equation of a conic in inhomogeneous coordinates is

$$ax^2 + bxy + cy^2 + dx + ey + f = 0$$

i.e. a polynomial of degree 2. "Homogenizing" this by the replacements: $x \mapsto x_1/x_3$, $y \mapsto x_2/x_3$ gives

$$ax_1^2 + bx_1x_2 + cx_2^2 + dx_1x_3 + ex_2x_3 + fx_3^2 = 0 \tag{2.1}$$

or in matrix form

$$\mathbf{x}^\mathsf{T}\mathtt{C}\mathbf{x} = 0 \tag{2.2}$$

where the conic coefficient matrix $\mathtt{C}$ is given by

$$\mathtt{C} = \begin{bmatrix} a & b/2 & d/2 \\ b/2 & c & e/2 \\ d/2 & e/2 & f \end{bmatrix}. \tag{2.3}$$

Note that the conic coefficient matrix is symmetric. As in the case of the homogeneous representation of points and lines, only the ratios of the matrix elements are important, since multiplying $\mathtt{C}$ by a non-zero scalar does not affect the above equations. Thus $\mathtt{C}$ is a homogeneous representation of a conic. The conic has five degrees of freedom which can be thought of as the ratios $\{a : b : c : d : e : f\}$ or equivalently the six elements of a symmetric matrix less one for scale.

Five points define a conic. Suppose we wish to compute the conic which passes through a set of points, $\mathbf{x}_i$. How many points are we free to specify before the conic is determined uniquely? The question can be answered constructively by providing an

Samsung/Google, EX1010, p. 48

algorithm to determine the conic. From (2.1) each point $\mathbf{x}_i$ places one constraint on the conic coefficients, since if the conic passes through (x_i, y_i) then

$$ax_i^2 + bx_iy_i + cy_i^2 + dx_i + ey_i + f = 0.$$

This constraint can be written as

$$\begin{pmatrix} x_i^2 & x_iy_i & y_i^2 & x_i & y_i & 1 \end{pmatrix} \mathbf{c} = 0$$

where $\mathbf{c} = (a, b, c, d, e, f)^\top$ is the conic $\mathtt{C}$ represented as a 6-vector.

Stacking the constraints from five points we obtain

$$\begin{bmatrix} x_1^2 & x_1y_1 & y_1^2 & x_1 & y_1 & 1 \\ x_2^2 & x_2y_2 & y_2^2 & x_2 & y_2 & 1 \\ x_3^2 & x_3y_3 & y_3^2 & x_3 & y_3 & 1 \\ x_4^2 & x_4y_4 & y_4^2 & x_4 & y_4 & 1 \\ x_5^2 & x_5y_5 & y_5^2 & x_5 & y_5 & 1 \end{bmatrix} \mathbf{c} = 0 \tag{2.4}$$

and the conic is the null vector of this 5×6 matrix. This shows that a conic is determined uniquely (up to scale) by five points in general position. The method of fitting a geometric entity (or relation) by determining a null space will be used frequently in the computation chapters throughout this book.

Tangent lines to conics. The line $\mathbf{l}$ tangent to a conic at a point $\mathbf{x}$ has a particularly simple form in homogeneous coordinates:

Result 2.7. *The line* $\mathbf{l}$ *tangent to* $\mathtt{C}$ *at a point* $\mathbf{x}$ *on* $\mathtt{C}$ *is given by* $\mathbf{l} = \mathtt{C}\mathbf{x}$.

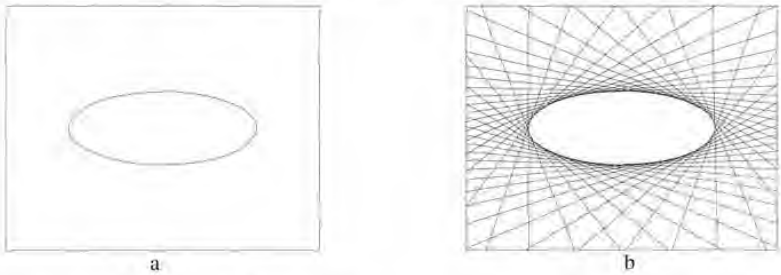
Proof. The line $\mathbf{l} = \mathtt{C}\mathbf{x}$ passes through $\mathbf{x}$, since $\mathbf{l}^\top\mathbf{x} = \mathbf{x}^\top\mathtt{C}\mathbf{x} = 0$. If $\mathbf{l}$ has one-point contact with the conic, then it is a tangent, and we are done. Otherwise suppose that $\mathbf{l}$ meets the conic in another point $\mathbf{y}$. Then $\mathbf{y}^\top\mathtt{C}\mathbf{y} = 0$ and $\mathbf{x}^\top\mathtt{C}\mathbf{y} = \mathbf{l}^\top\mathbf{y} = 0$. From this it follows that $(\mathbf{x} + \alpha\mathbf{y})^\top\mathtt{C}(\mathbf{x} + \alpha\mathbf{y}) = 0$ for all α, which means that the whole line $\mathbf{l} = \mathtt{C}\mathbf{x}$ joining $\mathbf{x}$ and $\mathbf{y}$ lies on the conic $\mathtt{C}$, which is therefore degenerate (see below). □

Dual conics. The conic $\mathtt{C}$ defined above is more properly termed a *point* conic, as it defines an equation on points. Given the duality result 2.6 of $\mathbb{P}^2$ it is not surprising that there is also a conic which defines an equation on lines. This *dual* (or line) conic is also represented by a 3×3 matrix, which we denote as $\mathtt{C}^*$. A line $\mathbf{l}$ *tangent* to the conic $\mathtt{C}$ satisfies $\mathbf{l}^\top\mathtt{C}^*\mathbf{l} = 0$. The notation $\mathtt{C}^*$ indicates that $\mathtt{C}^*$ is the adjoint matrix of $\mathtt{C}$ (the adjoint is defined in section A4.2(p580) of appendix 4(p578)). For a non-singular symmetric matrix $\mathtt{C}^* = \mathtt{C}^{-1}$ (up to scale).

The equation for a dual conic is straightforward to derive in the case that $\mathtt{C}$ has full rank: From result 2.7, at a point $\mathbf{x}$ on $\mathtt{C}$ the tangent is $\mathbf{l} = \mathtt{C}\mathbf{x}$. Inverting, we find the point $\mathbf{x}$ at which the line $\mathbf{l}$ is tangent to $\mathtt{C}$ is $\mathbf{x} = \mathtt{C}^{-1}\mathbf{l}$. Since $\mathbf{x}$ satisfies $\mathbf{x}^\top\mathtt{C}\mathbf{x} = 0$ we obtain $(\mathtt{C}^{-1}\mathbf{l})^\top\mathtt{C}(\mathtt{C}^{-1}\mathbf{l}) = \mathbf{l}^\top\mathtt{C}^{-1}\mathbf{l} = 0$, the last step following from $\mathtt{C}^{-\top} = \mathtt{C}^{-1}$ because $\mathtt{C}$ is symmetric.

Dual conics are also known as conic envelopes, and the reason for this is illustrated

Samsung/Google, EX1010, p. 49



a b

Fig. 2.2. *(a) Points* $\mathbf{x}$ *satisfying* $\mathbf{x}^\top \mathtt{C}\mathbf{x} = 0$ *lie on a point conic. (b) Lines* $\mathbf{l}$ *satisfying* $\mathbf{l}^\top \mathtt{C}^* \mathbf{l} = 0$ *are tangent to the point conic* $\mathtt{C}$*. The conic* $\mathtt{C}$ *is the envelope of the lines* $\mathbf{l}$.

in figure 2.2. A dual conic has five degrees of freedom. In a similar manner to points defining a point conic, it follows that five lines in general position define a dual conic.

Degenerate conics. If the matrix $\mathtt{C}$ is not of full rank, then the conic is termed degenerate. Degenerate point conics include two lines (rank 2), and a repeated line (rank 1).

Example 2.8. The conic

$$\mathtt{C} = \mathbf{l}\mathbf{m}^\top + \mathbf{m}\mathbf{l}^\top$$

is composed of two lines $\mathbf{l}$ and $\mathbf{m}$. Points on $\mathbf{l}$ satisfy $\mathbf{l}^\top \mathbf{x} = 0$, and are on the conic since $\mathbf{x}^\top \mathtt{C}\mathbf{x} = (\mathbf{x}^\top \mathbf{l})(\mathbf{m}^\top \mathbf{x}) + (\mathbf{x}^\top \mathbf{m})(\mathbf{l}^\top \mathbf{x}) = 0$. Similarly, points satisfying $\mathbf{m}^\top \mathbf{x} = 0$ also satisfy $\mathbf{x}^\top \mathtt{C}\mathbf{x} = 0$. The matrix $\mathtt{C}$ is symmetric and has rank 2. The null vector is $\mathbf{x} = \mathbf{l} \times \mathbf{m}$ which is the intersection point of $\mathbf{l}$ and $\mathbf{m}$. △

Degenerate *line* conics include two points (rank 2), and a repeated point (rank 1). For example, the line conic $\mathtt{C}^* = \mathbf{x}\mathbf{y}^\top + \mathbf{y}\mathbf{x}^\top$ has rank 2 and consists of lines passing through either of the two points $\mathbf{x}$ and $\mathbf{y}$. Note that for matrices that are not invertible $(\mathtt{C}^*)^* \neq \mathtt{C}$.

2.3 Projective transformations

In the view of geometry set forth by Felix Klein in his famous "Erlangen Program", [Klein-39], geometry is the study of properties invariant under groups of transformations. From this point of view, 2D projective geometry is the study of properties of the projective plane $\mathbb{P}^2$ that are invariant under a group of transformations known as *projectivities*.

A projectivity is an invertible mapping from points in $\mathbb{P}^2$ (that is homogeneous 3-vectors) to points in $\mathbb{P}^2$ that maps lines to lines. More precisely,

Definition 2.9. A *projectivity* is an invertible mapping h from $\mathbb{P}^2$ to itself such that three points $\mathbf{x}_1$, $\mathbf{x}_2$ and $\mathbf{x}_3$ lie on the same line if and only if $h(\mathbf{x}_1)$, $h(\mathbf{x}_2)$ and $h(\mathbf{x}_3)$ do.

Projectivities form a group since the inverse of a projectivity is also a projectivity, and so is the composition of two projectivities. A projectivity is also called a *collineation*

Samsung/Google, EX1010, p. 50

(a helpful name), a *projective transformation* or a *homography*: the terms are synonymous.

In definition 2.9, a projectivity is defined in terms of a coordinate-free geometric concept of point line incidence. An equivalent algebraic definition of a projectivity is possible, based on the following result.

Theorem 2.10. *A mapping $h : \mathbb{P}^2 \to \mathbb{P}^2$ is a projectivity if and only if there exists a non-singular 3×3 matrix $\mathtt{H}$ such that for any point in $\mathbb{P}^2$ represented by a vector $\mathbf{x}$ it is true that $h(\mathbf{x}) = \mathtt{H}\mathbf{x}$.*

To interpret this theorem, any point in $\mathbb{P}^2$ is represented as a homogeneous 3-vector, $\mathbf{x}$, and $\mathtt{H}\mathbf{x}$ is a linear mapping of homogeneous coordinates. The theorem asserts that any projectivity arises as such a linear transformation in homogeneous coordinates, and that conversely any such mapping is a projectivity. The theorem will not be proved in full here. It will only be shown that any invertible linear transformation of homogeneous coordinates is a projectivity.

Proof. Let $\mathbf{x}_1$, $\mathbf{x}_2$ and $\mathbf{x}_3$ lie on a line $\mathbf{l}$. Thus $\mathbf{l}^\top \mathbf{x}_i = 0$ for $i = 1, \ldots, 3$. Let $\mathtt{H}$ be a non-singular 3×3 matrix. One verifies that $\mathbf{l}^\top \mathtt{H}^{-1} \mathtt{H}\mathbf{x}_i = 0$. Thus, the points $\mathtt{H}\mathbf{x}_i$ all lie on the line $\mathtt{H}^{-\top}\mathbf{l}$, and collinearity is preserved by the transformation.

The converse is considerably harder to prove, namely that each projectivity arises in this way. □

As a result of this theorem, one may give an alternative definition of a projective transformation (or collineation) as follows.

Definition 2.11. Projective transformation. A planar projective transformation is a linear transformation on homogeneous 3-vectors represented by a non-singular 3×3 matrix:

$$\begin{pmatrix} x'_1 \\ x'_2 \\ x'_3 \end{pmatrix} = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{bmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix}. \tag{2.5}$$

or more briefly, $\mathbf{x}' = \mathtt{H}\mathbf{x}$.

Note that the matrix $\mathtt{H}$ occurring in this equation may be changed by multiplication by an arbitrary non-zero scale factor without altering the projective transformation. Consequently we say that $\mathtt{H}$ is a *homogeneous* matrix, since as in the homogeneous representation of a point, only the ratio of the matrix elements is significant. There are eight independent ratios amongst the nine elements of $\mathtt{H}$, and it follows that a projective transformation has eight degrees of freedom.

A projective transformation projects every figure into a projectively equivalent figure, leaving all its projective properties invariant. In the ray model of figure 2.1 a projective transformation is simply a linear transformation of $\mathbb{R}^3$.

Samsung/Google, EX1010, p. 51

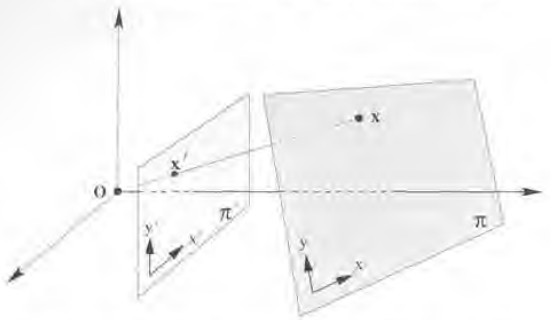


Fig. 2.3. **Central projection maps points on one plane to points on another plane.** *The projection also maps lines to lines as may be seen by considering a plane through the projection centre which intersects with the two planes π and π'. Since lines are mapped to lines, central projection is a projectivity and may be represented by a linear mapping of homogeneous coordinates $\mathbf{x}' = \mathtt{H}\mathbf{x}$.*

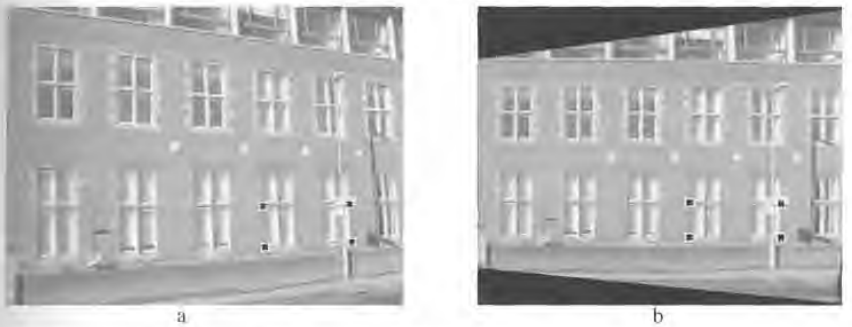
Mappings between planes. As an example of how theorem 2.10 may be applied, consider figure 2.3. Projection along rays through a common point (the centre of projection) defines a mapping from one plane to another. It is evident that this point-to-point mapping preserves lines in that a line in one plane is mapped to a line in the other. If a coordinate system is defined in each plane and points are represented in homogeneous coordinates, then the *central projection* mapping may be expressed by $\mathbf{x}' = \mathtt{H}\mathbf{x}$ where $\mathtt{H}$ is a non-singular 3×3 matrix. Actually, if the two coordinate systems defined in the two planes are both Euclidean (rectilinear) coordinate systems then the mapping defined by central projection is more restricted than an arbitrary projective transformation. It is called a *perspectivity* rather than a full projectivity, and may be represented by a transformation with six degrees of freedom. We return to perspectivities in section A7.4(*p*632).

Example 2.12. Removing the projective distortion from a perspective image of a plane.

Shape is distorted under perspective imaging. For instance, in figure 2.4a the windows are not rectangular in the image, although the originals are. In general parallel lines on a scene plane are not parallel in the image but instead converge to a finite point. We have seen that a central projection image of a plane (or section of a plane) is related to the original plane via a projective transformation, and so the image is a projective distortion of the original. It is possible to "undo" this projective transformation by computing the inverse transformation and applying it to the image. The result will be a new synthesized image in which the objects in the plane are shown with their correct geometric shape. This will be illustrated here for the front of the building of figure 2.4a. Note that since the ground and the front are not in the same plane, the projective transformation that must be applied to rectify the front is not the same as the one used for the ground.

Computation of a projective transformation from point-to-point correspondences will be considered in great detail in chapter 4. For now, a method for computing the trans-

Samsung/Google, EX1010, p. 52



a b

Fig. 2.4. **Removing perspective distortion.** *(a) The original image with perspective distortion – the lines of the windows clearly converge at a finite point. (b) Synthesized frontal orthogonal view of the front wall. The image (a) of the wall is related via a projective transformation to the true geometry of the wall. The inverse transformation is computed by mapping the four imaged window corners to corners of an appropriately sized rectangle. The four point correspondences determine the transformation. The transformation is then applied to the whole image. Note that sections of the image of the ground are subject to a further projective distortion. This can also be removed by a projective transformation.*

formation is briefly indicated. One begins by selecting a section of the image corresponding to a planar section of the world. Local 2D image and world coordinates are selected as shown in figure 2.3. Let the inhomogeneous coordinates of a pair of matching points $\mathbf{x}$ and $\mathbf{x}'$ in the world and image plane be (x, y) and (x', y') respectively. We use inhomogeneous coordinates here instead of the homogeneous coordinates of the points, because it is these inhomogeneous coordinates that are measured directly from the image and from the world plane. The projective transformation of (2.5) can be written in inhomogeneous form as

$$x' = \frac{x'_1}{x'_3} = \frac{h_{11}x + h_{12}y + h_{13}}{h_{31}x + h_{32}y + h_{33}}, \qquad y' = \frac{x'_2}{x'_3} = \frac{h_{21}x + h_{22}y + h_{23}}{h_{31}x + h_{32}y + h_{33}}.$$

Each point correspondence generates two equations for the elements of $\mathtt{H}$, which after multiplying out are

$$\begin{aligned} x'\,(h_{31}x + h_{32}y + h_{33}) &= h_{11}x + h_{12}y + h_{13} \\ y'\,(h_{31}x + h_{32}y + h_{33}) &= h_{21}x + h_{22}y + h_{23}. \end{aligned}$$

These equations are *linear* in the elements of $\mathtt{H}$. Four point correspondences lead to eight such linear equations in the entries of $\mathtt{H}$, which are sufficient to solve for $\mathtt{H}$ up to an insignificant multiplicative factor. The only restriction is that the four points must be in "general position", which means that no three points are collinear. The inverse of the transformation $\mathtt{H}$ computed in this way is then applied to the whole image to undo the effect of perspective distortion on the selected plane. The results are shown in figure 2.4b. △

Three remarks concerning this example are appropriate: first, the computation of the rectifying transformation $\mathtt{H}$ in this way does not require knowledge of *any* of the camera's parameters or the pose of the plane; second, it is not always necessary to

Samsung/Google, EX1010, p. 53

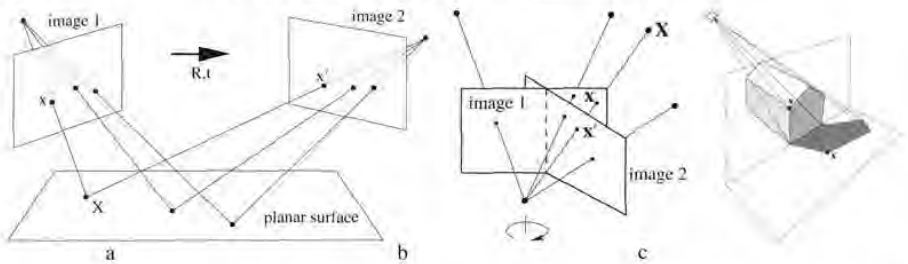


Fig. 2.5. **Examples of a projective transformation, $\mathbf{x}' = \mathtt{H}\mathbf{x}$, arising in perspective images.** *(a) The projective transformation between two images induced by a world plane (the concatenation of two projective transformations is a projective transformation); (b) The projective transformation between two images with the same camera centre (e.g. a camera rotating about its centre or a camera varying its focal length); (c) The projective transformation between the image of a plane (the end of the building) and the image of its shadow onto another plane (the ground plane). Figure (c) courtesy of Luc Van Gool.*

know coordinates for four points in order to remove projective distortion: alternative approaches, which are described in section 2.7, require less, and different types of, information; third, superior (and preferred) methods for computing projective transformations are described in chapter 4.

Projective transformations are important mappings representing many more situations than the perspective imaging of a world plane. A number of other examples are illustrated in figure 2.5. Each of these situations is covered in more detail later in the book.

2.3.1 Transformations of lines and conics

Transformation of lines. It was shown in the proof of theorem 2.10 that if points $\mathbf{x}_i$ lie on a line $\mathbf{l}$, then the transformed points $\mathbf{x}'_i = \mathtt{H}\mathbf{x}_i$ under a projective transformation lie on the line $\mathbf{l}' = \mathtt{H}^{-\mathsf{T}}\mathbf{l}$. In this way, incidence of points on lines is preserved, since $\mathbf{l}'^{\mathsf{T}}\mathbf{x}'_i = \mathbf{l}^{\mathsf{T}}\mathtt{H}^{-1}\mathtt{H}\mathbf{x}_i = 0$. This gives the transformation rule for lines:
Under the point transformation $\mathbf{x}' = \mathtt{H}\mathbf{x}$, a line transforms as

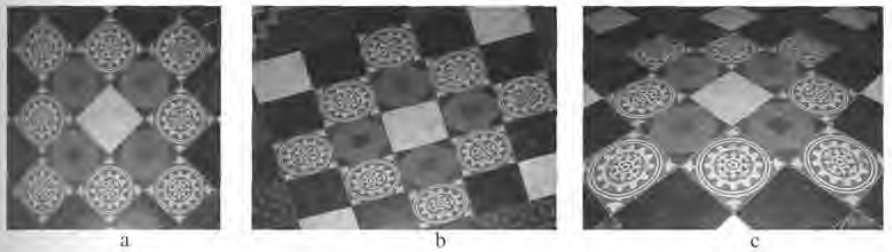
$$\mathbf{l}' = \mathtt{H}^{-\mathsf{T}}\mathbf{l}. \tag{2.6}$$

One may alternatively write $\mathbf{l}'^{\mathsf{T}} = \mathbf{l}^{\mathsf{T}}\mathtt{H}^{-1}$. Note the fundamentally different way in which lines and points transform. Points transform according to $\mathtt{H}$, whereas lines (as rows) transform according to $\mathtt{H}^{-1}$. This may be explained in terms of "covariant" or "contravariant" behaviour. One says that points transform *contravariantly* and lines transform *covariantly*. This distinction will be taken up again, when we discuss tensors in chapter 15 and is fully explained in appendix 1(*p*562).

Transformation of conics. Under a point transformation $\mathbf{x}' = \mathtt{H}\mathbf{x}$, (2.2) becomes

$$\begin{aligned} \mathbf{x}^{\mathsf{T}}\mathtt{C}\mathbf{x} &= \mathbf{x}'^{\mathsf{T}}[\mathtt{H}^{-1}]^{\mathsf{T}}\mathtt{C}\mathtt{H}^{-1}\mathbf{x}' \\ &= \mathbf{x}'^{\mathsf{T}}\mathtt{H}^{-\mathsf{T}}\mathtt{C}\mathtt{H}^{-1}\mathbf{x}' \end{aligned}$$

Samsung/Google, EX1010, p. 54



a b c

Fig. 2.6. **Distortions arising under central projection.** *Images of a tiled floor. (a)* **Similarity**: *the circular pattern is imaged as a circle. A square tile is imaged as a square. Lines which are parallel or perpendicular have the same relative orientation in the image. (b)* **Affine**: *The circle is imaged as an ellipse. Orthogonal world lines are not imaged as orthogonal lines. However, the sides of the square tiles, which are parallel in the world are parallel in the image. (c)* **Projective**: *Parallel world lines are imaged as converging lines. Tiles closer to the camera have a larger image than those further away.*

which is a quadratic form $\mathbf{x}'^{\mathsf{T}}\mathtt{C}'\mathbf{x}'$ with $\mathtt{C}' = \mathtt{H}^{-\mathsf{T}}\mathtt{C}\mathtt{H}^{-1}$. This gives the transformation rule for a conic:

Result 2.13. *Under a point transformation* $\mathbf{x}' = \mathtt{H}\mathbf{x}$, *a conic* $\mathtt{C}$ *transforms to* $\mathtt{C}' = \mathtt{H}^{-\mathsf{T}}\mathtt{C}\mathtt{H}^{-1}$.

The presence of $\mathtt{H}^{-1}$ in this equation may be expressed by saying that a conic transforms *covariantly*. The transformation rule for a dual conic is derived in a similar manner. This gives:

Result 2.14. *Under a point transformation* $\mathbf{x}' = \mathtt{H}\mathbf{x}$, *a dual conic* $\mathtt{C}^*$ *transforms to* $\mathtt{C}^{*\prime} = \mathtt{H}\mathtt{C}^*\mathtt{H}^{\mathsf{T}}$.

2.4 A hierarchy of transformations

In this section we describe the important specializations of a projective transformation and their geometric properties. It was shown in section 2.3 that projective transformations form a group. This group is called the *projective linear group*, and it will be seen that these specializations are *subgroups* of this group.

The group of invertible $n \times n$ matrices with real elements is the (real) general linear group on n dimensions, or $GL(n)$. To obtain the projective linear group the matrices related by a scalar multiplier are identified, giving $PL(n)$ (this is a quotient group of $GL(n)$). In the case of projective transformations of the plane $n = 3$.

The important subgroups of $PL(3)$ include the *affine group*, which is the subgroup of $PL(3)$ consisting of matrices for which the last row is $(0, 0, 1)$, and the *Euclidean group*, which is a subgroup of the affine group for which in addition the upper left hand 2×2 matrix is orthogonal. One may also identify the *oriented Euclidean group* in which the upper left hand 2×2 matrix has determinant 1.

We will introduce these transformations starting from the most specialized, the isometries, and progressively generalizing until projective transformations are reached.

Samsung/Google, EX1010, p. 55

This defines a *hierarchy* of transformations. The distortion effects of various transformations in this hierarchy are shown in figure 2.6.

Some transformations of interest are not groups, for example, perspectivities (because the composition of two perspectivities is a projectivity, not a perspectivity). This point is covered in section A7.4(p632).

Invariants. An alternative to describing the transformation *algebraically*, i.e. as a matrix acting on coordinates of a point or curve, is to describe the transformation in terms of those elements or quantities that are preserved or *invariant*. A (scalar) invariant of a geometric configuration is a function of the configuration whose value is unchanged by a particular transformation. For example, the separation of two points is unchanged by a Euclidean transformation (translation and rotation), but not by a similarity (e.g. translation, rotation and isotropic scaling). Distance is thus a Euclidean, but not similarity invariant. The angle between two lines is both a Euclidean and a similarity invariant.

2.4.1 Class I: Isometries

Isometries are transformations of the plane $\mathbb{R}^2$ that preserve Euclidean distance (from *iso* = same, *metric* = measure). An isometry is represented as

$$\begin{pmatrix} x' \\ y' \\ 1 \end{pmatrix} = \begin{bmatrix} \epsilon\cos\theta & -\sin\theta & t_x \\ \epsilon\sin\theta & \cos\theta & t_y \\ 0 & 0 & 1 \end{bmatrix} \begin{pmatrix} x \\ y \\ 1 \end{pmatrix}$$

where $\epsilon = \pm 1$. If $\epsilon = 1$ then the isometry is *orientation-preserving* and is a *Euclidean* transformation (a composition of a translation and rotation). If $\epsilon = -1$ then the isometry reverses orientation. An example is the composition of a reflection, represented by the matrix $\mathrm{diag}(-1, 1, 1)$, with a Euclidean transformation.

Euclidean transformations model the motion of a rigid object. They are by far the most important isometries in practice, and we will concentrate on these. However, the orientation reversing isometries often arise as ambiguities in structure recovery.

A planar Euclidean transformation can be written more concisely in block form as

$$\mathbf{x}' = \mathtt{H}_{\mathrm{E}}\mathbf{x} = \begin{bmatrix} \mathtt{R} & \mathbf{t} \\ \mathbf{0}^\top & 1 \end{bmatrix} \mathbf{x} \tag{2.7}$$

where $\mathtt{R}$ is a 2×2 rotation matrix (an orthogonal matrix such that $\mathtt{R}^\top\mathtt{R} = \mathtt{R}\mathtt{R}^\top = \mathtt{I}$), $\mathbf{t}$ a translation 2-vector, and $\mathbf{0}$ a null 2-vector. Special cases are a pure rotation (when $\mathbf{t} = \mathbf{0}$) and a pure translation (when $\mathtt{R} = \mathtt{I}$). A Euclidean transformation is also known as a *displacement*.

A planar Euclidean transformation has three degrees of freedom, one for the rotation and two for the translation. Thus three parameters must be specified in order to define the transformation. The transformation can be computed from two point correspondences.

Invariants. The invariants are very familiar, for instance: length (the distance between two points), angle (the angle between two lines), and area.

Samsung/Google, EX1010, p. 56

Groups and orientation. An isometry is orientation-preserving if the upper left hand 2×2 matrix has determinant 1. Orientation-*preserving* isometries form a group, orientation-*reversing* ones do not. This distinction applies also in the case of similarity and affine transformations which now follow.

2.4.2 Class II: Similarity transformations

A similarity transformation (or more simply a *similarity*) is an isometry composed with an isotropic scaling. In the case of a Euclidean transformation composed with a scaling (i.e. no reflection) the similarity has matrix representation

$$\begin{pmatrix} x' \\ y' \\ 1 \end{pmatrix} = \begin{bmatrix} s\cos\theta & -s\sin\theta & t_x \\ s\sin\theta & s\cos\theta & t_y \\ 0 & 0 & 1 \end{bmatrix} \begin{pmatrix} x \\ y \\ 1 \end{pmatrix}. \tag{2.8}$$

This can be written more concisely in block form as

$$\mathbf{x}' = \mathtt{H}_s\mathbf{x} = \begin{bmatrix} s\mathtt{R} & \mathtt{t} \\ \mathbf{0}^\top & 1 \end{bmatrix} \mathbf{x} \tag{2.9}$$

where the scalar s represents the isotropic scaling. A similarity transformation is also known as an *equi-form* transformation, because it preserves "shape" (form). A planar similarity transformation has four degrees of freedom, the scaling accounting for one more degree of freedom than a Euclidean transformation. A similarity can be computed from two point correspondences.

Invariants. The invariants can be constructed from Euclidean invariants with suitable provision being made for the additional scaling degree of freedom. Angles between lines are not affected by rotation, translation or isotropic scaling, and so are similarity invariants. In particular parallel lines are mapped to parallel lines. The length between two points is not a similarity invariant, but the *ratio* of two lengths is an invariant, because the scaling of the lengths cancels out. Similarly a ratio of areas is an invariant because the scaling (squared) cancels out.

Metric structure. A term that will be used frequently in the discussion on reconstruction (chapter 10) is *metric*. The description *metric structure* implies that the structure is defined up to a similarity.

2.4.3 Class III: Affine transformations

An affine transformation (or more simply an *affinity*) is a non-singular linear transformation followed by a translation. It has the matrix representation

$$\begin{pmatrix} x' \\ y' \\ 1 \end{pmatrix} = \begin{bmatrix} a_{11} & a_{12} & t_x \\ a_{21} & a_{22} & t_y \\ 0 & 0 & 1 \end{bmatrix} \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} \tag{2.10}$$

Samsung/Google, EX1010, p. 57

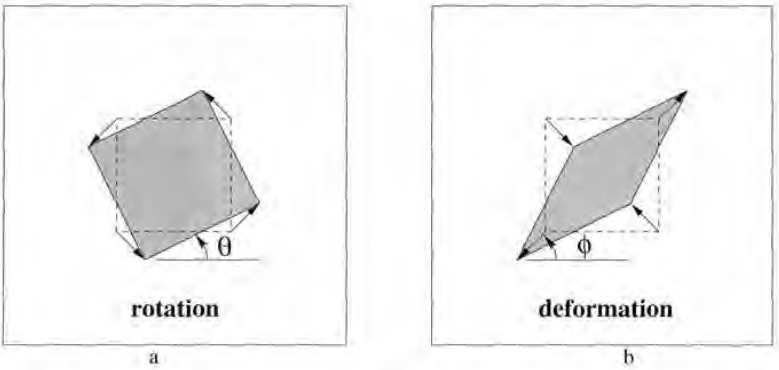


Fig. 2.7. **Distortions arising from a planar affine transformation.** *(a) Rotation by* $\mathtt{R}(\theta)$*. (b) A deformation* $\mathtt{R}(-\phi)\,\mathtt{D}\,\mathtt{R}(\phi)$*. Note, the scaling directions in the deformation are orthogonal.*

or in block form

$$\mathbf{x}' = \mathtt{H}_{\mathtt{A}}\mathbf{x} = \begin{bmatrix} \mathtt{A} & \mathbf{t} \\ \mathbf{0}^{\mathsf{T}} & 1 \end{bmatrix} \mathbf{x} \tag{2.11}$$

with $\mathtt{A}$ a 2×2 non-singular matrix. A planar affine transformation has six degrees of freedom corresponding to the six matrix elements. The transformation can be computed from three point correspondences.

A helpful way to understand the geometric effects of the linear component $\mathtt{A}$ of an affine transformation is as the composition of two fundamental transformations, namely rotations and non-isotropic scalings. The affine matrix $\mathtt{A}$ can always be decomposed as

$$\mathtt{A} \;=\; \mathtt{R}(\theta)\,\mathtt{R}(-\phi)\,\mathtt{D}\,\mathtt{R}(\phi) \tag{2.12}$$

where $\mathtt{R}(\theta)$ and $\mathtt{R}(\phi)$ are rotations by θ and ϕ respectively, and $\mathtt{D}$ is a diagonal matrix:

$$\mathtt{D} = \begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{bmatrix}.$$

This decomposition follows directly from the SVD (section A4.4($p585$)): writing $\mathtt{A} = \mathtt{U}\mathtt{D}\mathtt{V}^{\mathsf{T}} = (\mathtt{U}\mathtt{V}^{\mathsf{T}})(\mathtt{V}\mathtt{D}\mathtt{V}^{\mathsf{T}}) = \mathtt{R}(\theta)\,(\mathtt{R}(-\phi)\,\mathtt{D}\,\mathtt{R}(\phi))$, since $\mathtt{U}$ and $\mathtt{V}$ are orthogonal matrices.

The affine matrix $\mathtt{A}$ is hence seen to be the concatenation of a rotation (by ϕ); a scaling by λ_1 and λ_2 respectively in the (rotated) x and y directions; a rotation back (by $-\phi$); and finally another rotation (by θ). The only "new" geometry, compared to a similarity, is the non-isotropic scaling. This accounts for the two extra degrees of freedom possessed by an affinity over a similarity. They are the angle ϕ specifying the scaling direction, and the ratio of the scaling parameters $\lambda_1 : \lambda_2$. The essence of an affinity is this scaling in orthogonal directions, oriented at a particular angle. Schematic examples are given in figure 2.7.

Samsung/Google, EX1010, p. 58

Invariants. Because an affine transformation includes non-isotropic scaling, the similarity invariants of length ratios and angles between lines are not preserved under an affinity. Three important invariants are:

(i) **Parallel lines.** Consider two parallel lines. These intersect at a point $(x_1, x_2, 0)^\top$ at infinity. Under an affine transformation this point is mapped to another point at infinity. Consequently, the parallel lines are mapped to lines which still intersect at infinity, and so are parallel after the transformation.

(ii) **Ratio of lengths of parallel line segments.** The length scaling of a line segment depends only on the angle between the line direction and scaling directions. Suppose the line is at angle α to the x-axis of the orthogonal scaling direction, then the scaling magnitude is $\sqrt{\lambda_1^2 \cos^2\alpha + \lambda_2^2 \sin^2\alpha}$. This scaling is common to all lines with the same direction, and so cancels out in a ratio of parallel segment lengths.

(iii) **Ratio of areas.** This invariance can be deduced directly from the decomposition (2.12). Rotations and translations do not affect area, so only the scalings by λ_1 and λ_2 matter here. The effect is that area is scaled by $\lambda_1\lambda_2$ which is equal to $\det \mathtt{A}$. Thus the area of any shape is scaled by $\det \mathtt{A}$, and so the scaling cancels out for a ratio of areas. It will be seen that this does not hold for a projective transformation.

An affinity is orientation-preserving or -reversing according to whether $\det \mathtt{A}$ is positive or negative respectively. Since $\det \mathtt{A} = \lambda_1\lambda_2$ the property depends only on the sign of the scalings.

2.4.4 Class IV: Projective transformations

A projective transformation was defined in (2.5). It is a general non-singular linear transformation of *homogeneous* coordinates. This generalizes an affine transformation, which is the composition of a general non-singular linear transformation of *inhomogeneous* coordinates and a translation. We have earlier seen the action of a projective transformation (in section 2.3). Here we examine its block form

$$\mathbf{x}' = \mathtt{H}_\mathrm{P}\mathbf{x} = \begin{bmatrix} \mathtt{A} & \mathbf{t} \\ \mathbf{v}^\top & v \end{bmatrix} \mathbf{x} \tag{2.13}$$

where the vector $\mathbf{v} = (v_1, v_2)^\top$. The matrix has nine elements with only their ratio significant, so the transformation is specified by eight parameters. Note, it is not always possible to scale the matrix such that v is unity since v might be zero. A projective transformation between two planes can be computed from four point correspondences, with no three collinear on either plane. See figure 2.4.

Unlike the case of affinities, it is not possible to distinguish between orientation preserving and orientation reversing projectivities in $\mathbb{P}^2$. We will return to this point in section 2.6.

Samsung/Google, EX1010, p. 59

Invariants. The most fundamental projective invariant is the cross ratio of four collinear points: a ratio of lengths on a line is invariant under affinities, but not under projectivities. However, a ratio of ratios or *cross ratio* of lengths on a line is a projective invariant. We return to properties of this invariant in section 2.5.

2.4.5 Summary and comparison

Affinities (6 dof) occupy the middle ground between similarities (4 dof) and projectivities (8 dof). They generalize similarities in that angles are not preserved, so that shapes are skewed under the transformation. On the other hand their action is homogeneous over the plane: for a given affinity the $\det \mathtt{A}$ scaling in area of an object (e.g. a square) is the same anywhere on the plane; and the orientation of a transformed line depends only on its initial orientation, not on its position on the plane. In contrast, for a given projective transformation, area scaling varies with position (e.g. under perspective a more distant square on the plane has a smaller image than one that is nearer, as in figure 2.6); and the orientation of a transformed line depends on both the orientation and position of the source line (however, it will be seen later in section 8.6($p213$) that a line's vanishing point depends only on line orientation, not position).

The key difference between a projective and affine transformation is that the vector $\mathbf{v}$ is not null for a projectivity. This is responsible for the non-linear effects of the projectivity. Compare the mapping of an ideal point $(x_1, x_2, 0)^\top$ under an affinity and projectivity: First the affine transformation

$$\begin{bmatrix} \mathtt{A} & \mathbf{t} \\ \mathbf{0}^\top & 1 \end{bmatrix} \begin{pmatrix} x_1 \\ x_2 \\ 0 \end{pmatrix} = \begin{pmatrix} \mathtt{A} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} \\ 0 \end{pmatrix}. \tag{2.14}$$

Second the projective transformation

$$\begin{bmatrix} \mathtt{A} & \mathbf{t} \\ \mathbf{v}^\top & v \end{bmatrix} \begin{pmatrix} x_1 \\ x_2 \\ 0 \end{pmatrix} = \begin{pmatrix} \mathtt{A} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} \\ v_1 x_1 + v_2 x_2 \end{pmatrix}. \tag{2.15}$$

In the first case the ideal point remains ideal (i.e. at infinity). In the second it is mapped to a finite point. It is this ability which allows a projective transformation to model vanishing points.

2.4.6 Decomposition of a projective transformation

A projective transformation can be decomposed into a chain of transformations, where each matrix in the chain represents a transformation higher in the hierarchy than the previous one.

$$\mathtt{H} = \mathtt{H}_\mathrm{S}\,\mathtt{H}_\mathrm{A}\,\mathtt{H}_\mathrm{P} = \begin{bmatrix} s\mathtt{R} & \mathbf{t} \\ \mathbf{0}^\top & 1 \end{bmatrix} \begin{bmatrix} \mathtt{K} & \mathbf{0} \\ \mathbf{0}^\top & 1 \end{bmatrix} \begin{bmatrix} \mathtt{I} & \mathbf{0} \\ \mathbf{v}^\top & v \end{bmatrix} = \begin{bmatrix} \mathtt{A} & \mathbf{t} \\ \mathbf{v}^\top & v \end{bmatrix} \tag{2.16}$$

with $\mathtt{A}$ a non-singular matrix given by $\mathtt{A} = s\mathtt{R}\mathtt{K} + \mathbf{t}\mathbf{v}^\top$, and $\mathtt{K}$ an upper-triangular matrix normalized as $\det \mathtt{K} = 1$. This decomposition is valid provided $v \neq 0$, and is unique if s is chosen positive.

Samsung/Google, EX1010, p. 60

Each of the matrices $\mathtt{H}_S, \mathtt{H}_A, \mathtt{H}_P$ is the "essence" of a transformation of that type (as indicated by the subscripts S, A, P). Consider the process of rectifying the perspective image of a plane as in example 2.12: $\mathtt{H}_P$ (2 dof) moves the line at infinity; $\mathtt{H}_A$ (2 dof) affects the affine properties, but does not move the line at infinity; and finally, $\mathtt{H}_S$ is a general similarity transformation (4 dof) which does not affect the affine or projective properties. The transformation $\mathtt{H}_P$ is an *elation*, described in section A7.3(*p*631).

Example 2.15. The projective transformation

$$\mathtt{H} = \begin{bmatrix} 1.707 & 0.586 & 1.0 \\ 2.707 & 8.242 & 2.0 \\ 1.0 & 2.0 & 1.0 \end{bmatrix}$$

may be decomposed as

$$\mathtt{H} = \begin{bmatrix} 2\cos 45^\circ & -2\sin 45^\circ & 1 \\ 2\sin 45^\circ & 2\cos 45^\circ & 2 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 0.5 & 1 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 1 & 2 & 1 \end{bmatrix}.$$

△

This decomposition can be employed when the objective is to only partially determine the transformation. For example, if one wants to measure length ratios from the perspective image of a plane, then it is only necessary to determine (rectify) the transformation up to a similarity. We return to this approach in section 2.7.

Taking the inverse of $\mathtt{H}$ in (2.16) gives $\mathtt{H}^{-1} = \mathtt{H}_P^{-1}\mathtt{H}_A^{-1}\mathtt{H}_S^{-1}$. Since $\mathtt{H}_P^{-1}, \mathtt{H}_A^{-1}$ and $\mathtt{H}_S^{-1}$ are still projective, affine and similarity transformations respectively, a general projective transformation may also be decomposed in the form

$$\mathtt{H} = \mathtt{H}_P\mathtt{H}_A\mathtt{H}_S = \begin{bmatrix} \mathtt{I} & \mathbf{0} \\ \mathbf{v}^\top & 1 \end{bmatrix} \begin{bmatrix} \mathtt{K} & \mathbf{0} \\ \mathbf{0}^\top & 1 \end{bmatrix} \begin{bmatrix} s\mathtt{R} & \mathbf{t} \\ \mathbf{0}^\top & 1 \end{bmatrix} \tag{2.17}$$

Note that the actual values of $\mathtt{K}$, $\mathtt{R}$, $\mathbf{t}$ and $\mathbf{v}$ will be different from those of (2.16).

2.4.7 The number of invariants

The question naturally arises as to how many invariants there are for a given geometric configuration under a particular transformation. First the term "number" needs to be made more precise, for if a quantity is invariant, such as length under Euclidean transformations, then any function of that quantity is invariant. Consequently, we seek a counting argument for the number of functionally independent invariants. By considering the number of transformation parameters that must be eliminated in order to form an invariant, it can be seen that:

Result 2.16. *The number of functionally independent invariants is equal to, or greater than, the number of degrees of freedom of the configuration less the number of degrees of freedom of the transformation.*

Samsung/Google, EX1010, p. 61

Group	Matrix	Distortion	Invariant properties
Projective 8 dof	$\begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{bmatrix}$		Concurrency, collinearity, **order of contact**: intersection (1 pt contact); tangency (2 pt contact); inflections (3 pt contact with line); tangent discontinuities and cusps. cross ratio (ratio of ratio of lengths).
Affine 6 dof	$\begin{bmatrix} a_{11} & a_{12} & t_x \\ a_{21} & a_{22} & t_y \\ 0 & 0 & 1 \end{bmatrix}$		Parallelism, ratio of areas, ratio of lengths on collinear or parallel lines (e.g. midpoints), linear combinations of vectors (e.g. centroids). The line at infinity, $\mathbf{l}_\infty$.
Similarity 4 dof	$\begin{bmatrix} sr_{11} & sr_{12} & t_x \\ sr_{21} & sr_{22} & t_y \\ 0 & 0 & 1 \end{bmatrix}$		Ratio of lengths, angle. The circular points, $\mathbf{I}, \mathbf{J}$ (see section 2.7.3).
Euclidean 3 dof	$\begin{bmatrix} r_{11} & r_{12} & t_x \\ r_{21} & r_{22} & t_y \\ 0 & 0 & 1 \end{bmatrix}$		Length, area

Table 2.1. **Geometric properties invariant to commonly occurring *planar* transformations.** *The matrix $\mathtt{A} = [a_{ij}]$ is an invertible 2×2 matrix, $\mathtt{R} = [r_{ij}]$ is a 2D rotation matrix, and (t_x, t_y) a 2D translation. The distortion column shows typical effects of the transformations on a square. Transformations higher in the table can produce all the actions of the ones below. These range from Euclidean, where only translations and rotations occur, to projective where the square can be transformed to any arbitrary quadrilateral (provided no three points are collinear).*

For example, a configuration of four points in general position has 8 degrees of freedom (2 for each point), and so 4 similarity, 2 affinity and zero projective invariants since these transformations have respectively 4, 6 and 8 degrees of freedom.

Table 2.1 summarizes the 2D transformation groups and their invariant properties. Transformations lower in the table are specializations of those above. A transformation lower in the table inherits the invariants of those above.

2.5 The projective geometry of 1D

The development of the projective geometry of a line, $\mathbb{P}^1$, proceeds in much the same way as that of the plane. A point x on the line is represented by homogeneous coordinates $(x_1, x_2)^\top$, and a point for which $x_2 = 0$ is an ideal point of the line. We will use the notation $\bar{\mathbf{x}}$ to represent the 2-vector $(x_1, x_2)^\top$. A projective transformation of a line is represented by a 2×2 homogeneous matrix,

$$\bar{\mathbf{x}}' = \mathtt{H}_{2\times 2}\bar{\mathbf{x}}$$

and has 3 degrees of freedom corresponding to the four elements of the matrix less one for overall scaling. A projective transformation of a line may be determined from three corresponding points.

Samsung/Google, EX1010, p. 62

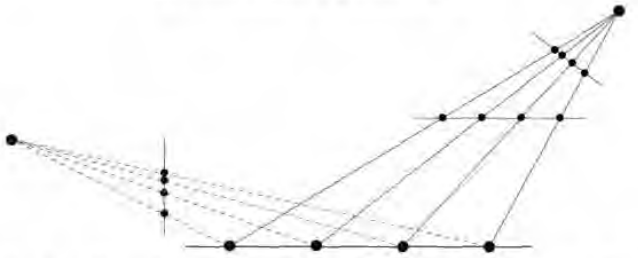


Fig. 2.8. **Projective transformations between lines.** *There are four sets of four collinear points in this figure. Each set is related to the others by a line-to-line projectivity. Since the cross ratio is an invariant under a projectivity, the cross ratio has the same value for all the sets shown.*

The cross ratio. The cross ratio is the basic projective invariant of $\mathbb{P}^1$. Given 4 points $\bar{\mathbf{x}}_i$ the *cross ratio* is defined as

$$\text{Cross}(\bar{\mathbf{x}}_1, \bar{\mathbf{x}}_2, \bar{\mathbf{x}}_3, \bar{\mathbf{x}}_4) = \frac{|\bar{\mathbf{x}}_1\bar{\mathbf{x}}_2||\bar{\mathbf{x}}_3\bar{\mathbf{x}}_4|}{|\bar{\mathbf{x}}_1\bar{\mathbf{x}}_3||\bar{\mathbf{x}}_2\bar{\mathbf{x}}_4|}$$

where

$$|\bar{\mathbf{x}}_i\bar{\mathbf{x}}_j| = \det \begin{bmatrix} x_{i1} & x_{j1} \\ x_{i2} & x_{j2} \end{bmatrix} .$$

A few comments on the cross ratio:

(i) The value of the cross ratio is not dependent on which particular homogeneous representative of a point $\bar{\mathbf{x}}_i$ is used, since the scale cancels between numerator and denominator.

(ii) If each point $\bar{\mathbf{x}}_i$ is a finite point and the homogeneous representative is chosen such that $x_2 = 1$, then $|\bar{\mathbf{x}}_i\bar{\mathbf{x}}_j|$ represents the signed distance from $\bar{\mathbf{x}}_i$ to $\bar{\mathbf{x}}_j$.

(iii) The definition of the cross ratio is also valid if one of the points $\bar{\mathbf{x}}_i$ is an ideal point.

(iv) The value of the cross ratio is invariant under any projective transformation of the line: if $\bar{\mathbf{x}}' = \mathtt{H}_{2\times 2}\bar{\mathbf{x}}$ then

$$\text{Cross}(\bar{\mathbf{x}}'_1, \bar{\mathbf{x}}'_2, \bar{\mathbf{x}}'_3, \bar{\mathbf{x}}'_4) = \text{Cross}(\bar{\mathbf{x}}_1, \bar{\mathbf{x}}_2, \bar{\mathbf{x}}_3, \bar{\mathbf{x}}_4). \tag{2.18}$$

The proof is left as an exercise. Equivalently stated, the cross ratio is invariant to the projective coordinate frame chosen for the line.

Figure 2.8 illustrates a number of projective transformations between lines with equivalent cross ratios.

Under a projective transformation of the plane, a 1D projective transformation is induced on any line in the plane.

Concurrent lines. A configuration of concurrent lines is dual to collinear points on a line. This means that concurrent lines on a plane also have the geometry $\mathbb{P}^1$. In particular four concurrent lines have a cross ratio as illustrated in figure 2.9a.

Samsung/Google, EX1010, p. 63

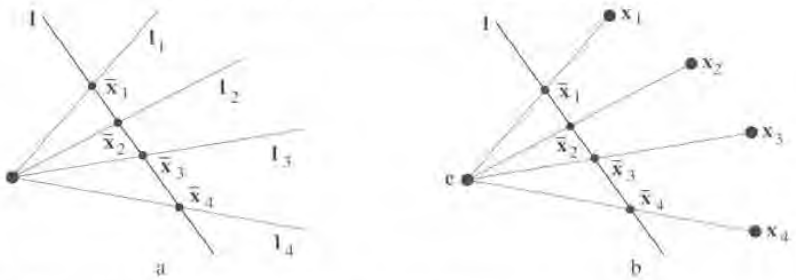


Fig. 2.9. **Concurrent lines.** *(a) Four concurrent lines* $\mathbf{l}_i$ *intersect the line* $\mathbf{l}$ *in the four points* $\bar{\mathbf{x}}_i$*. The cross ratio of these lines is an invariant to projective transformations of the plane. Its value is given by the cross ratio of the points,* $Cross(\bar{\mathbf{x}}_1, \bar{\mathbf{x}}_2, \bar{\mathbf{x}}_3, \bar{\mathbf{x}}_4)$*. (b) Coplanar points* $\mathbf{x}_i$ *are imaged onto a line* $\mathbf{l}$ *(also in the plane) by a projection with centre* $\mathbf{c}$*. The cross ratio of the image points* $\bar{\mathbf{x}}_i$ *is invariant to the position of the image line* $\mathbf{l}$*.*

Note how figure 2.9b may be thought of as representing projection of points in $\mathbb{P}^2$ into a 1-dimensional image. In particular, if $\mathbf{c}$ represents a camera centre, and the line $\mathbf{l}$ represents an image line (1D analogue of the image plane), then the points $\bar{\mathbf{x}}_i$ are the projections of points $\mathbf{x}_i$ into the image. The cross ratio of the points $\bar{\mathbf{x}}_i$ characterizes the projective configuration of the four image points. Note that the actual position of the image line is irrelevant as far as the projective configuration of the four image points is concerned – different choices of image line give rise to projectively equivalent configurations of image points.

The projective geometry of concurrent lines is important to the understanding of the projective geometry of epipolar lines in chapter 9.

2.6 Topology of the projective plane

We make brief mention of the topology of $\mathbb{P}^2$*. Understanding of this section is not required for following the rest of the book.*

We have seen that the projective plane $\mathbb{P}^2$ may be thought of as the set of all homogeneous 3-vectors. A vector of this type $\mathbf{x} = (x_1, x_2, x_3)^\top$ may be normalized by multiplication by a non-zero factor so that $x_1^2 + x_2^2 + x_3^2 = 1$. Such a point lies on the unit sphere in $\mathbb{R}^3$. However, any vector $\mathbf{x}$ and $-\mathbf{x}$ represent the same point in $\mathbb{P}^2$, since they differ by a multiplicative factor, -1. Thus, there is a two-to-one correspondence between the unit sphere S^2 in $\mathbb{R}^3$ and the projective plane $\mathbb{P}^2$. The projective plane may be pictured as the unit sphere with opposite points identified. In this representation, a line in $\mathbb{P}^2$ is modelled as a great circle on the unit sphere (as ever, with opposite points identified). One may verify that any two distinct (non-antipodal) points on the sphere lie on exactly one great circle, and any two great circles intersect in one point (since antipodal points are identified).

In the language of topology, the sphere S^2 is a 2-sheeted covering space of $\mathbb{P}^2$. This implies that $\mathbb{P}^2$ is not *simply-connected*, which means that there are loops in $\mathbb{P}^2$ which cannot be contracted to a point inside $\mathbb{P}^2$. To be technical, the fundamental group of $\mathbb{P}^2$ is the cyclic group of order 2.

Samsung/Google, EX1010, p. 64

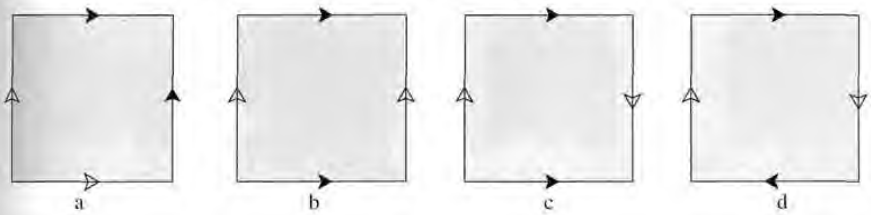


Fig. 2.10. **Topology of surfaces.** *Common surfaces may be constructed from a paper square (topologically a disk) with edges glued together. In each case, the matching arrow edges of the square are to be glued together in such a way that the directions of the arrows match. One obtains (a) a sphere, (b) a torus, (c) a Klein bottle and (d) a projective plane. Only the sphere and torus are actually realizable with a real sheet of paper. The sphere and torus are orientable but the projective plane and Klein bottle are not.*

In the model for the projective plane as a sphere with opposite points identified one may dispense with the lower hemisphere of S^2, since points in this hemisphere are the same as the opposite points in the upper hemisphere. In this case, $\mathbb{P}^2$ may be constructed from the upper hemisphere by identifying opposite points on the equator. Since the upper hemisphere of S^2 is topologically the same as a disk, $\mathbb{P}^2$ is simply a disk with opposite points on its boundary identified, or glued together. This is not physically possible. Constructing topological spaces by gluing the boundary of a disk is a common method in topology, and in fact any 2-manifold may be constructed in this way. This is illustrated in figure 2.10.

A notable feature of the projective plane $\mathbb{P}^2$ is that it is non-orientable. This means that it is impossible to define a local orientation (represented for instance by a pair of oriented coordinate axes) that is consistent over the whole surface. This is illustrated in figure 2.11 in which it is shown that the projective plane contains an orientation-reversing path.

The topology of $\mathbb{P}^1$. In a similar manner, the 1-dimensional projective line may be identified as a 1-sphere S^1 (that is, a circle) with opposite points identified. If we omit the lower half of the circle, as being duplicated by the top half, then the top half of a circle is topologically equivalent to a line segment. Thus $\mathbb{P}^1$ is topologically equivalent to a line segment with the two endpoints identified – namely a circle, S^1.

2.7 Recovery of affine and metric properties from images

We return to the example of projective rectification of example 2.12(p34) where the aim was to remove the projective distortion in the perspective image of a plane to the extent that similarity properties (angles, ratios of lengths) could be measured on the original plane. In that example the projective distortion was completely removed by specifying the position of four reference points on the plane (a total of 8 degrees of freedom), and explicitly computing the transformation mapping the reference points to their images. In fact this overspecifies the geometry – a projective transformation has only 4 degrees of freedom more than a similarity, so it is only necessary to specify 4

Samsung/Google, EX1010, p. 65



a b

Fig. 2.11. **Orientation of surfaces.** *A coordinate frame (represented by an L in the diagram) may be transported along a path in the surface eventually coming back to the point where it started. (a) represents a projective plane. In the path shown, the coordinate frame (represented by a pair of axes) is reversed when it returns to the same point, since the identification at the boundary of the square swaps the direction of one of the axes. Such a path is called an orientation-reversing path, and a surface that contains such a path is called non-orientable. (b) shows the well known example of a Möbius strip obtained by joining two opposite edges of a rectangle (M.C. Escher's "Moebius Strip II [Red Ants]", 1963. ©2000 Cordon Art B.V. – Baarn-Holland. All rights reserved). As can be verified, a path once around the strip is orientation-reversing.*

degrees of freedom (not 8) in order to determine metric properties. In projective geometry these 4 degrees of freedom are given "physical substance" by being associated with geometric objects: the line at infinity $\mathbf{l}_\infty$ (2 dof), and the two *circular points* (2 dof) on $\mathbf{l}_\infty$. This association is often a more intuitive way of reasoning about the problem than the equivalent description in terms of specifying matrices in the decomposition chain (2.16).

In the following it is shown that the projective distortion may be removed once the image of $\mathbf{l}_\infty$ is specified, and the affine distortion removed once the image of the circular points is specified. Then the only remaining distortion is a similarity.

2.7.1 The line at infinity

Under a projective transformation ideal points may be mapped to finite points (2.15), and consequently $\mathbf{l}_\infty$ is mapped to a finite line. However, if the transformation is an affinity, then $\mathbf{l}_\infty$ is not mapped to a finite line, but remains at infinity. This is evident directly from the line transformation (2.6–*p*36):

$$\mathbf{l}'_\infty = \mathtt{H}_{\mathtt{A}}^{-\mathsf{T}}\mathbf{l}_\infty = \begin{bmatrix} \mathtt{A}^{-\mathsf{T}} & \mathbf{0} \\ -\mathbf{t}^{\mathsf{T}}\mathtt{A}^{-\mathsf{T}} & 1 \end{bmatrix} \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix} = \mathbf{l}_\infty.$$

The converse is also true, i.e. an affine transformation is the most general linear transformation that fixes $\mathbf{l}_\infty$, and may be seen as follows. We require that a point at infinity, say $\mathbf{x} = (1, 0, 0)^{\mathsf{T}}$, be mapped to a point at infinity. This requires that $h_{31} = 0$. Similarly, $h_{32} = 0$, so the transformation is an affinity. To summarize,

Result 2.17. *The line at infinity, $\mathbf{l}_\infty$, is a fixed line under the projective transformation* $\mathtt{H}$ *if and only if* $\mathtt{H}$ *is an affinity.*

However, $\mathbf{l}_\infty$ is not fixed pointwise under an affine transformation: (2.14) showed that under an affinity a point on $\mathbf{l}_\infty$ (an ideal point) is mapped to a point on $\mathbf{l}_\infty$, but

Samsung/Google, EX1010, p. 66

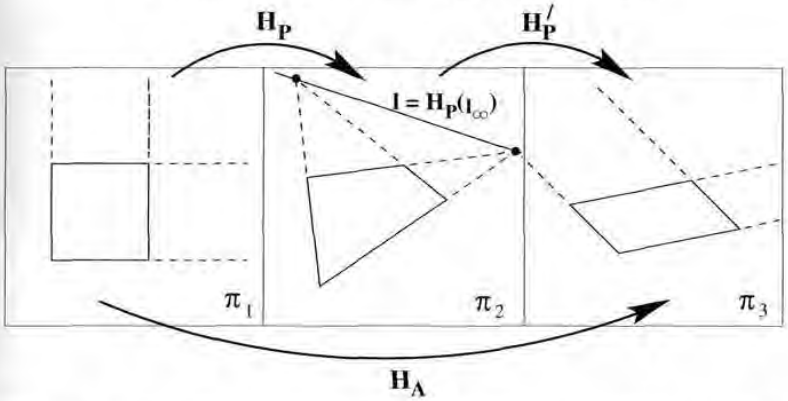


Fig. 2.12. **Affine rectification.** *A projective transformation maps* $\mathbf{l}_\infty$ *from* $(0,0,1)^\mathsf{T}$ *on a Euclidean plane* π_1 *to a finite line* $\mathbf{l}$ *on the plane* π_2. *If a projective transformation is constructed such that* $\mathbf{l}$ *is mapped back to* $(0,0,1)^\mathsf{T}$ *then from result 2.17 the transformation between the first and third planes* must be an affine transformation *since the canonical position of* $\mathbf{l}_\infty$ *is preserved. This means that affine properties of the first plane can be measured from the third, i.e. the third plane is within an affinity of the first.*

it is not the same point unless $\mathtt{A}(x_1, x_2)^\mathsf{T} = k(x_1, x_2)^\mathsf{T}$. It will now be shown that identifying $\mathbf{l}_\infty$ allows the recovery of affine properties (parallelism, ratio of areas).

2.7.2 Recovery of affine properties from images

Once the imaged line at infinity is identified in an image of a plane, it is then possible to make affine measurements on the original plane. For example, lines may be identified as parallel on the original plane if the imaged lines intersect on the imaged $\mathbf{l}_\infty$. This follows because parallel lines on the Euclidean plane intersect on $\mathbf{l}_\infty$, and after a projective transformation the lines still intersect on the imaged $\mathbf{l}_\infty$ since intersections are preserved by projectivities. Similarly, once $\mathbf{l}_\infty$ is identified a length ratio on a line may be computed from the cross ratio of the three points specifying the lengths together with the intersection of the line with $\mathbf{l}_\infty$ (which provides the fourth point for the cross ratio), and so forth.

However, a less tortuous path which is better suited to computational algorithms is simply to transform the identified $\mathbf{l}_\infty$ to its canonical position of $\mathbf{l}_\infty = (0,0,1)^\mathsf{T}$. The (projective) matrix which achieves this transformation can be applied to every point in the image in order to affinely rectify the image, i.e. after the transformation, affine measurements can be made directly from the rectified image. The key idea here is illustrated in figure 2.12.

If the imaged line at infinity is the line $\mathbf{l} = (l_1, l_2, l_3)^\mathsf{T}$, then provided $l_3 \neq 0$ a suitable projective point transformation which will map $\mathbf{l}$ back to $\mathbf{l}_\infty = (0,0,1)^\mathsf{T}$ is

$$\mathtt{H} = \mathtt{H}_\mathtt{A} \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ l_1 & l_2 & l_3 \end{bmatrix} \tag{2.19}$$

Samsung/Google, EX1010, p. 67

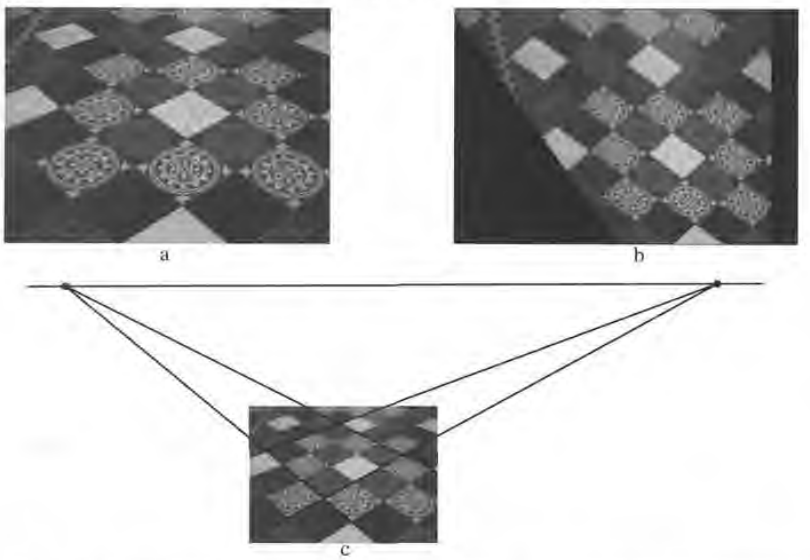


Fig. 2.13. **Affine rectification via the vanishing line.** *The vanishing line of the plane imaged in (a) is computed (c) from the intersection of two sets of imaged parallel lines. The image is then projectively warped to produce the affinely rectified image (b). In the affinely rectified image parallel lines are now parallel. However, angles do not have their veridical world value since they are affinely distorted. See also figure 2.17.*

where $\mathtt{H}_A$ is any affine transformation (the last row of $\mathtt{H}$ is $\mathbf{l}^\mathsf{T}$). One can verify that under the line transformation (2.6–*p*36) $\mathtt{H}^{-\mathsf{T}}(l_1, l_2, l_3)^\mathsf{T} = (0, 0, 1)^\mathsf{T} = \mathbf{l}_\infty$.

Example 2.18. Affine rectification
In a perspective image of a plane, the line at infinity on the world plane is imaged as the vanishing line of the plane. This is discussed in more detail in chapter 8. As illustrated in figure 2.13 the vanishing line $\mathbf{l}$ may be computed by intersecting imaged parallel lines. The image is then rectified by applying a projective warping (2.19) such that $\mathbf{l}$ is mapped to its canonical position $\mathbf{l}_\infty = (0, 0, 1)^\mathsf{T}$. △

This example shows that affine properties may be recovered by simply specifying a line (2 dof). It is equivalent to specifying only the projective component of the transformation decomposition chain (2.16). Conversely if affine properties are known, these may be used to determine points and the line at infinity. This is illustrated in the following example.

Example 2.19. Computing a vanishing point from a length ratio. Given two intervals on a line with a known length ratio, the point at infinity on the line may be determined. A typical case is where three points $\mathbf{a}'$, $\mathbf{b}'$ and $\mathbf{c}'$ are identified on a line in an image. Suppose $\mathbf{a}$, $\mathbf{b}$ and $\mathbf{c}$ are the corresponding collinear points on the world line, and the length ratio $d(\mathbf{a}, \mathbf{b}) : d(\mathbf{b}, \mathbf{c}) = a : b$ is known (where $d(\mathbf{x}, \mathbf{y})$ is the Euclidean

Samsung/Google, EX1010, p. 68

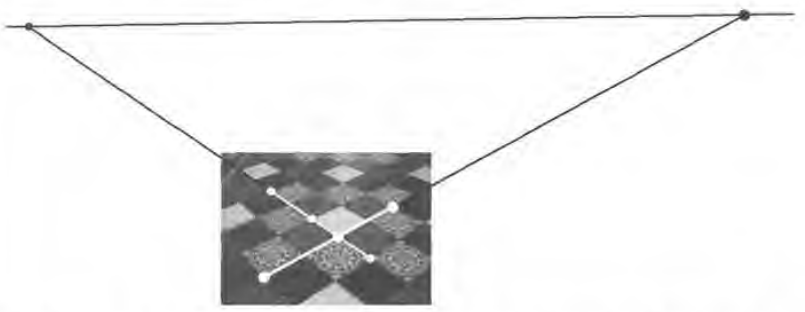


Fig. 2.14. *Two examples of using equal length ratios on a line to determine the point at infinity. The line intervals used are shown as the thin and thick white lines delineated by points. This construction determines the vanishing line of the plane. Compare with figure 2.13c.*

distance between the points $\mathbf{x}$ and $\mathbf{y}$). It is possible to find the vanishing point using the cross ratio. Equivalently, one may proceed as follows:

(i) Measure the distance ratio in the image, $d(\mathbf{a}', \mathbf{b}') : d(\mathbf{b}', \mathbf{c}') = a' : b'$.
(ii) Points $\mathbf{a}$, $\mathbf{b}$ and $\mathbf{c}$ may be represented as coordinates 0, a and $a+b$ in a coordinate frame on the line $\langle \mathbf{a}, \mathbf{b}, \mathbf{c} \rangle$. For computational purposes, these points are represented by homogeneous 2-vectors $(0,1)^\top$, $(a,1)^\top$ and $(a+b,1)^\top$. Similarly, $\mathbf{a}'$, $\mathbf{b}'$ and $\mathbf{c}'$ have coordinates 0, a' and $a'+b'$, which may also be expressed as homogeneous vectors.
(iii) Relative to these coordinate frames, compute the 1D projective transformation $\mathtt{H}_{2\times 2}$ mapping $\mathbf{a} \mapsto \mathbf{a}'$, $\mathbf{b} \mapsto \mathbf{b}'$ and $\mathbf{c} \mapsto \mathbf{c}'$.
(iv) The image of the point at infinity (with coordinates $(1,0)^\top$) under $\mathtt{H}_{2\times 2}$ is the vanishing point on the line $\langle \mathbf{a}', \mathbf{b}', \mathbf{c}' \rangle$.

An example of vanishing points computed in this manner is shown in figure 2.14. △

Example 2.20. Geometric construction of vanishing points from a length ratio. The vanishing points shown in figure 2.14 may also be computed by a purely geometric construction consisting of the following steps:

(i) Given: three collinear points, $\mathbf{a}'$, $\mathbf{b}'$ and $\mathbf{c}'$, in an image corresponding to collinear world points with interval ratio $a : b$.
(ii) Draw any line $\mathbf{l}$ through $\mathbf{a}'$ (not coincident with the line $\mathbf{a}'\mathbf{c}'$), and mark off points $\mathbf{a} = \mathbf{a}'$, $\mathbf{b}$ and $\mathbf{c}$ such that the line segments $\langle \mathbf{ab} \rangle$, $\langle \mathbf{bc} \rangle$ have length ratio $a : b$.
(iii) Join $\mathbf{bb}'$ and $\mathbf{cc}'$ and intersect in $\mathbf{o}$.
(iv) The line through $\mathbf{o}$ parallel to $\mathbf{l}$ meets the line $\mathbf{a}'\mathbf{c}'$ in the vanishing point $\mathbf{v}'$.

This construction is illustrated in figure 2.15. △

Samsung/Google, EX1010, p. 69

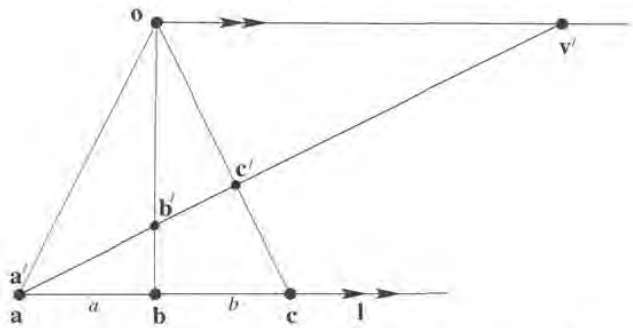


Fig. 2.15. *A geometric construction to determine the image of the point at infinity on a line given a known length ratio. The details are given in the text.*

2.7.3 The circular points and their dual

Under any similarity transformation there are two points on I_∞ which are fixed. These are the *circular points* (also called the *absolute points*) $\mathbf{I}, \mathbf{J}$, with canonical coordinates

$$\mathbf{I} = \begin{pmatrix} 1 \\ i \\ 0 \end{pmatrix} \qquad \mathbf{J} = \begin{pmatrix} 1 \\ -i \\ 0 \end{pmatrix}.$$

The circular points are a pair of complex conjugate ideal points. To see that they are fixed under an orientation-preserving similarity:

$$\begin{aligned} \mathbf{I}' &= \mathtt{H}_S \mathbf{I} \\ &= \begin{bmatrix} s\cos\theta & -s\sin\theta & t_x \\ s\sin\theta & s\cos\theta & t_y \\ 0 & 0 & 1 \end{bmatrix} \begin{pmatrix} 1 \\ i \\ 0 \end{pmatrix} \\ &= se^{-i\theta} \begin{pmatrix} 1 \\ i \\ 0 \end{pmatrix} = \mathbf{I} \end{aligned}$$

with an analogous proof for $\mathbf{J}$. A reflection swaps $\mathbf{I}$ and $\mathbf{J}$. The converse is also true, i.e. if the circular points are fixed then the linear transformation is a similarity. The proof is left as an exercise. To summarize,

Result 2.21. *The circular points, $\mathbf{I}, \mathbf{J}$, are fixed points under the projective transformation $\mathtt{H}$ if and only if $\mathtt{H}$ is a similarity.*

The name "circular points" arises because every circle intersects I_∞ at the circular points. To see this, start from equation (2.1–p30) for a conic. In the case that the conic is a circle: $a = c$ and $b = 0$. Then

$$x_1^2 + x_2^2 + dx_1x_3 + ex_2x_3 + fx_3^2 = 0$$

Samsung/Google, EX1010, p. 70

where a has been set to unity. This conic intersects I_∞ in the (ideal) points for which $x_3 = 0$, namely

$$x_1^2 + x_2^2 = 0$$

with solution $\mathbf{I} = (1, i, 0)^\top$, $\mathbf{J} = (1, -i, 0)^\top$, i.e. any circle intersects I_∞ in the circular points. In Euclidean geometry it is well known that a circle is specified by three points. The circular points enable an alternative computation. A circle can be computed using the general formula for a conic defined by five points (2.4–$p31$), where the five points are the three points augmented with the two circular points.

In section 2.7.5 it will be shown that identifying the circular points (or equivalently their dual, see below) allows the recovery of similarity properties (angles, ratios of lengths). Algebraically, the circular points are the orthogonal directions of Euclidean geometry, $(1, 0, 0)^\top$ and $(0, 1, 0)^\top$, packaged into a single complex conjugate entity, e.g.

$$\mathbf{I} = (1, 0, 0)^\top + i(0, 1, 0)^\top.$$

Consequently, it is not so surprising that once the circular points are identified, orthogonality, and other metric properties, are then determined.

The conic dual to the circular points. The conic

$$\mathrm{C}_\infty^* = \mathbf{I}\mathbf{J}^\top + \mathbf{J}\mathbf{I}^\top \tag{2.20}$$

is dual to the circular points. The conic C_∞^* is a degenerate (rank 2) line conic (see section 2.2.3), which consists of the two circular points. In a Euclidean coordinate system it is given by

$$\mathrm{C}_\infty^* = \begin{pmatrix} 1 \\ i \\ 0 \end{pmatrix} \begin{pmatrix} 1 & -i & 0 \end{pmatrix} + \begin{pmatrix} 1 \\ -i \\ 0 \end{pmatrix} \begin{pmatrix} 1 & i & 0 \end{pmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}.$$

The conic C_∞^* is fixed under similarity transformations in an analogous fashion to the fixed properties of circular points. A conic is fixed if the same matrix results (up to scale) under the transformation rule. Since C_∞^* is a dual conic it transforms according to result 2.14($p37$) ($\mathrm{C}^{*\prime} = \mathrm{H}\mathrm{C}^*\mathrm{H}^\top$), and one can verify that under the point transformation $\mathbf{x}' = \mathrm{H}_\mathrm{S}\mathbf{x}$,

$$\mathrm{C}_\infty^{*\prime} = \mathrm{H}_\mathrm{S}\mathrm{C}_\infty^*\mathrm{H}_\mathrm{S}^\top = \mathrm{C}_\infty^*.$$

The converse is also true, and we have

ılt 2.22. *The dual conic C_∞^* is fixed under the projective transformation H if and if H is a similarity.*

Some properties of C_∞^* in any projective frame:

(i) C_∞^* has 4 degrees of freedom: a 3×3 homogeneous symmetric matrix has 5 degrees of freedom, but the constraint $\det \mathrm{C}_\infty^* = 0$ reduces the degrees of freedom by 1.

Samsung/Google, EX1010, p. 71

(ii) $\mathbf{I}_\infty$ is the null vector of $\mathtt{C}^*_\infty$. This is clear from the definition: the circular points lie on $\mathbf{I}_\infty$, so that $\mathbf{I}^\mathsf{T}\mathbf{I}_\infty = \mathbf{J}^\mathsf{T}\mathbf{I}_\infty = 0$; then

$$\mathtt{C}^*_\infty\mathbf{I}_\infty = (\mathbf{I}\mathbf{J}^\mathsf{T} + \mathbf{J}\mathbf{I}^\mathsf{T})\mathbf{I}_\infty = \mathbf{I}(\mathbf{J}^\mathsf{T}\mathbf{I}_\infty) + \mathbf{J}(\mathbf{I}^\mathsf{T}\mathbf{I}_\infty) = \mathbf{0}.$$

2.7.4 Angles on the projective plane

In Euclidean geometry the angle between two lines is computed from the dot product of their normals. For the lines $\mathbf{l} = (l_1, l_2, l_3)^\mathsf{T}$ and $\mathbf{m} = (m_1, m_2, m_3)^\mathsf{T}$ with normals parallel to $(l_1, l_2)^\mathsf{T}, (m_1, m_2)^\mathsf{T}$ respectively, the angle is

$$\cos\theta = \frac{l_1 m_1 + l_2 m_2}{\sqrt{(l_1^2 + l_2^2)(m_1^2 + m_2^2)}}. \tag{2.21}$$

The problem with this expression is that the first two components of $\mathbf{l}$ and $\mathbf{m}$ do not have well defined transformation properties under projective transformations (they are not tensors), and so (2.21) cannot be applied after an affine or projective transformation of the plane. However, an analogous expression to (2.21) which is invariant to projective transformations is

$$\cos\theta = \frac{\mathbf{l}^\mathsf{T}\mathtt{C}^*_\infty\mathbf{m}}{\sqrt{(\mathbf{l}^\mathsf{T}\mathtt{C}^*_\infty\mathbf{l})(\mathbf{m}^\mathsf{T}\mathtt{C}^*_\infty\mathbf{m})}} \tag{2.22}$$

where $\mathtt{C}^*_\infty$ is the conic dual to the circular points. It is clear that in a Euclidean coordinate system (2.22) reduces to (2.21). It may be verified that (2.22) is invariant to projective transformations by using the transformation rules for lines (2.6–*p*36) ($\mathbf{l}' = \mathtt{H}^{-\mathsf{T}}\mathbf{l}$) and dual conics (result 2.14(*p*37)) ($\mathtt{C}^{*\prime} = \mathtt{H}\mathtt{C}^*\mathtt{H}^\mathsf{T}$) under the point transformation $\mathbf{x}' = \mathtt{H}\mathbf{x}$. For example, the numerator transforms as

$$\mathbf{l}^\mathsf{T}\mathtt{C}^*_\infty\mathbf{m} \mapsto \mathbf{l}^\mathsf{T}\mathtt{H}^{-1}\mathtt{H}\mathtt{C}^*_\infty\mathtt{H}^\mathsf{T}\mathtt{H}^{-\mathsf{T}}\mathbf{m} = \mathbf{l}^\mathsf{T}\mathtt{C}^*_\infty\mathbf{m}.$$

It may also be verified that the scale of the homogeneous objects cancels between the numerator and denominator. Thus (2.22) is indeed invariant to the projective frame. To summarize, we have shown

Result 2.23. *Once the conic $\mathtt{C}^*_\infty$ is identified on the projective plane then Euclidean angles may be measured by (2.22).*

Note, as a corollary,

Result 2.24. *Lines $\mathbf{l}$ and $\mathbf{m}$ are orthogonal if $\mathbf{l}^\mathsf{T}\mathtt{C}^*_\infty\mathbf{m} = 0$.*

Geometrically, if $\mathbf{l}$ and $\mathbf{m}$ satisfy $\mathbf{l}^\mathsf{T}\mathtt{C}^*_\infty\mathbf{m} = 0$, then the lines are conjugate (see section 2.8.1) with respect to the conic $\mathtt{C}^*_\infty$.

Length ratios may also be measured once $\mathtt{C}^*_\infty$ is identified. Consider the triangle shown in figure 2.16 with vertices $\mathbf{a}, \mathbf{b}, \mathbf{c}$. From the standard trigonometric sine rule the ratio of lengths $d(\mathbf{b}, \mathbf{c}) : d(\mathbf{a}, \mathbf{c}) = \sin\alpha : \sin\beta$, where $d(\mathbf{x}, \mathbf{y})$ denotes the Euclidean distance between the points $\mathbf{x}$ and $\mathbf{y}$. Using (2.22), both $\cos\alpha$ and $\cos\beta$ may be computed from the lines $\mathbf{l}' = \mathbf{a}' \times \mathbf{b}'$, $\mathbf{m}' = \mathbf{c}' \times \mathbf{a}'$ and $\mathbf{n}' = \mathbf{b}' \times \mathbf{c}'$ for any

Samsung/Google, EX1010, p. 72

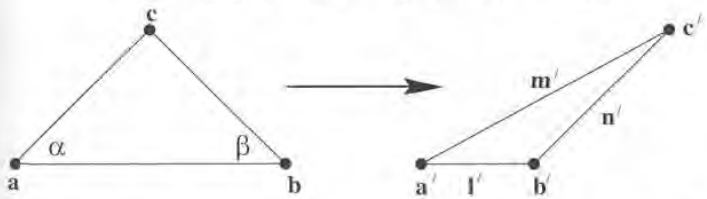


Fig. 2.16. **Length ratios.** *Once C_∞^* is identified the Euclidean length ratio $d(\mathbf{b},\mathbf{c}) : d(\mathbf{a},\mathbf{c})$ may be measured from the projectively distorted figure. See text for details.*

projective frame in which C_∞^* is specified. Consequently both $\sin\alpha$, $\sin\beta$, and thence the ratio $d(\mathbf{a},\mathbf{b}) : d(\mathbf{c},\mathbf{a})$, may be determined from the projectively mapped points.

2.7.5 Recovery of metric properties from images

A completely analogous approach to that of section 2.7.2 and figure 2.12, where affine properties are recovered by specifying l_∞, enables metric properties to be recovered from an image of a plane by transforming the circular points to their canonical position. Suppose the circular points are identified in an image, and the image is then rectified by a projective transformation H that maps the imaged circular points to their canonical position (at $(1, \pm i, 0)^\top$) on l_∞. From result 2.21 the transformation between the world plane and the rectified image is then a similarity since it is projective and the circular points are fixed.

Metric rectification using C_∞^*. The dual conic C_∞^* neatly packages all the information required for a metric rectification. It enables both the projective and affine components of a projective transformation to be determined, leaving only similarity distortions. This is evident from its transformation under a projectivity. If the point transformation is $\mathbf{x}' = \mathtt{H}\mathbf{x}$, where the x-coordinate frame is Euclidean and $\mathbf{x}'$ projective, C_∞^* transforms according to result 2.14($p37$) ($C^{*\prime} = \mathtt{H}C^*\mathtt{H}^\top$). Using the decomposition chain (2.17–$p43$) for H

$$\begin{aligned} C_\infty^{*\prime} &= (\mathtt{H}_\mathrm{P}\mathtt{H}_\mathrm{A}\mathtt{H}_\mathrm{S})\, C_\infty^* \,(\mathtt{H}_\mathrm{P}\mathtt{H}_\mathrm{A}\mathtt{H}_\mathrm{S})^\top = (\mathtt{H}_\mathrm{P}\mathtt{H}_\mathrm{A})\left(\mathtt{H}_\mathrm{S}\, C_\infty^* \mathtt{H}_\mathrm{S}^\top\right)\left(\mathtt{H}_\mathrm{A}^\top \mathtt{H}_\mathrm{P}^\top\right) \\ &= (\mathtt{H}_\mathrm{P}\mathtt{H}_\mathrm{A})\, C_\infty^* \left(\mathtt{H}_\mathrm{A}^\top \mathtt{H}_\mathrm{P}^\top\right) \\ &= \begin{bmatrix} \mathtt{K}\mathtt{K}^\top & \mathtt{K}\mathtt{K}^\top\mathbf{v} \\ \mathbf{v}^\top\mathtt{K}\mathtt{K}^\top & \mathbf{v}^\top\mathtt{K}\mathtt{K}^\top\mathbf{v} \end{bmatrix}. \end{aligned} \tag{2.23}$$

It is clear that the projective (v) and affine (K) components are determined directly from the image of C_∞^*, but (since C_∞^* is invariant to similarity transformation by result 2.22) the similarity component is undetermined. Consequently,

Result 2.25. *Once the conic C_∞^* is identified on the projective plane then projective distortion may be rectified up to a similarity.*

Actually, a suitable rectifying homography may be obtained directly from the identified $C_\infty^{*\prime}$ in an image using the SVD (section A4.4($p585$)): writing the SVD of $C_\infty^{*\prime}$

Samsung/Google, EX1010, p. 73

as

$$\mathtt{C}_\infty^{*\prime} = \mathtt{U}\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}\mathtt{U}^\mathsf{T}$$

then by inspection from (2.23) the rectifying projectivity is $\mathtt{H} = \mathtt{U}$ up to a similarity.

The following two examples show typical situations where $\mathtt{C}_\infty^*$ may be identified in an image, and thence a metric rectification obtained.

Example 2.26. Metric rectification I
Suppose an image has been affinely rectified (as in example 2.18 above), then we require two constraints to specify the 2 degrees of freedom of the circular points in order to determine a metric rectification. These two constraints may be obtained from two imaged right angles on the world plane.

Suppose the lines $\mathbf{l}', \mathbf{m}'$ in the affinely rectified image correspond to an orthogonal line pair $\mathbf{l}, \mathbf{m}$ on the world plane. From result 2.24 $\mathbf{l}'^\mathsf{T}\mathtt{C}_\infty^{*\prime}\mathbf{m}' = 0$, and using (2.23) with $\mathbf{v} = \mathbf{0}$

$$\begin{pmatrix} l_1' & l_2' & l_3' \end{pmatrix}\begin{bmatrix} \mathtt{K}\mathtt{K}^\mathsf{T} & \mathbf{0} \\ \mathbf{0}^\mathsf{T} & 0 \end{bmatrix}\begin{pmatrix} m_1' \\ m_2' \\ m_3' \end{pmatrix} = 0$$

which is a *linear* constraint on the 2×2 matrix $\mathtt{S} = \mathtt{K}\mathtt{K}^\mathsf{T}$. The matrix $\mathtt{S} = \mathtt{K}\mathtt{K}^\mathsf{T}$ is symmetric with three independent elements, and thus 2 degrees of freedom (as the overall scaling is unimportant). The orthogonality condition reduces to the equation $(l_1', l_2')\mathtt{S}(m_1', m_2')^\mathsf{T} = 0$ which may be written as

$$(l_1'm_1', l_1'm_2' + l_2'm_1', l_2'm_2')\,\mathbf{s} = 0,$$

where $\mathbf{s} = (s_{11}, s_{12}, s_{22})^\mathsf{T}$ is S written as a 3-vector. Two such orthogonal line pairs provide two constraints which may be stacked to give a 2×3 matrix with $\mathbf{s}$ determined as the null vector. Thus S, and hence K, is obtained up to scale (by Cholesky decomposition, section A4.2.1(*p*582)). Figure 2.17 shows an example of two orthogonal line pairs being used to metrically rectify the affinely rectified image computed in figure 2.13. △

Alternatively, the two constraints required for metric rectification may be obtained from an imaged circle or two known length ratios. In the case of a circle, the image conic is an ellipse in the affinely rectified image, and the intersection of this ellipse with the (known) $\mathbf{l}_\infty$ directly determines the imaged circular points.

The conic $\mathtt{C}_\infty^*$ can alternatively be identified directly in a perspective image, without first identifying $\mathbf{l}_\infty$, as is illustrated in the following example.

Example 2.27. Metric rectification II
We start here from the original perspective image of the plane (not the affinely rectified image of example 2.26). Suppose lines $\mathbf{l}$ and $\mathbf{m}$ are images of orthogonal lines on the world plane; then from result 2.24 $\mathbf{l}^\mathsf{T}\mathtt{C}_\infty^*\mathbf{m} = 0$, and in a similar manner to constraining

Samsung/Google, EX1010, p. 74

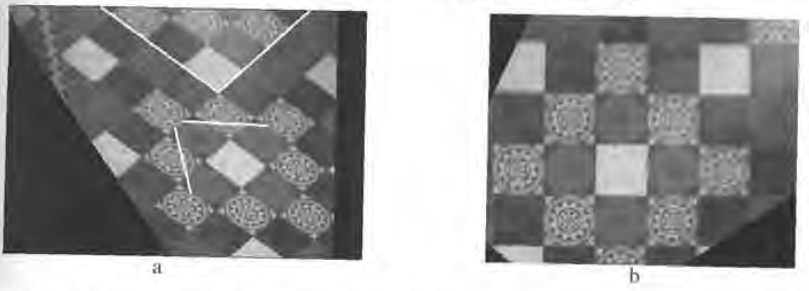


Fig. 2.17. **Metric rectification via orthogonal lines I.** *The affine transformation required to metrically rectify an affine image may be computed from imaged orthogonal lines. (a) Two (non-parallel) line pairs identified on the affinely rectified image (figure 2.13) correspond to orthogonal lines on the world plane. (b) The metrically rectified image. Note that in the metrically rectified image all lines orthogonal in the world are orthogonal, world squares have unit aspect ratio, and world circles are circular.*

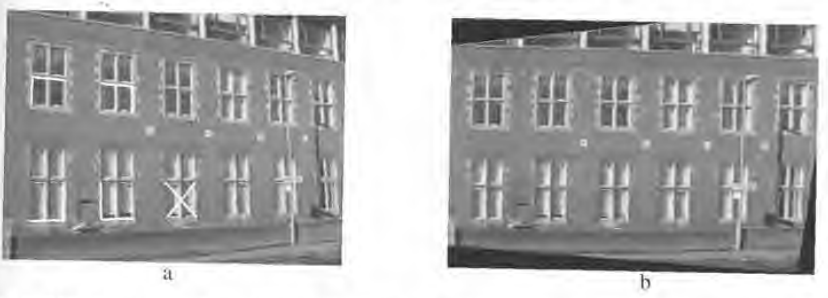


Fig. 2.18. **Metric rectification via orthogonal lines II.** *(a) The conic C^*_∞ is determined on the perspectively imaged plane (the front wall of the building) using the five orthogonal line pairs shown. The conic C^*_∞ determines the circular points, and equivalently the projective transformation necessary to metrically rectify the image (b). The image shown in (a) is the same perspective image as that of figure 2.4(p35), where the perspective distortion was removed by specifying the world position of four image points.*

a conic to contain a point (2.4–$p31$), this provides a linear constraint on the elements of C^*_∞, namely

$$(l_1m_1, (l_1m_2 + l_2m_1)/2, l_2m_2, (l_1m_3 + l_3m_1)/2, (l_2m_3 + l_3m_2)/2, l_3m_3)\,\mathbf{c} = 0$$

where $\mathbf{c} = (a, b, c, d, e, f)^\top$ is the conic matrix (2.3–$p30$) of C^*_∞ written as a 6-vector. Five such constraints can be stacked to form a 5×6 matrix, and $\mathbf{c}$, and hence C^*_∞, is obtained as the null vector. This shows that C^*_∞ can be determined linearly from the images of five line pairs which are orthogonal on the world plane. An example of metric rectification using such line pair constraints is shown in figure 2.18. △

Stratification. Note, in example 2.27 the affine and projective distortions are determined in one step by specifying C^*_∞. In the previous example 2.26 first the projective and subsequently the affine distortions were removed. This two-step approach is termed *stratified*. Analogous approaches apply in 3D, and are employed in chapter 10

Samsung/Google, EX1010, p. 75

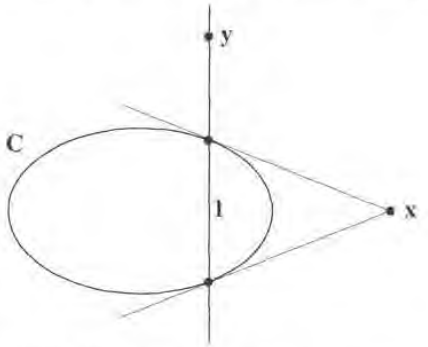


Fig. 2.19. **The pole–polar relationship.** *The line* $\mathbf{l} = \mathtt{C}\mathbf{x}$ *is the* polar *of the point* $\mathbf{x}$ *with respect to the conic* $\mathtt{C}$*, and the point* $\mathbf{x} = \mathtt{C}^{-1}\mathbf{l}$ *is the* pole *of* $\mathbf{l}$ *with respect to* $\mathtt{C}$*. The polar of* $\mathbf{x}$ *intersects the conic at the points of tangency of lines from* $\mathbf{x}$*. If* $\mathbf{y}$ *is on* $\mathbf{l}$ *then* $\mathbf{y}^\top\mathbf{l} = \mathbf{y}^\top\mathtt{C}\mathbf{x} = 0$*. Points* $\mathbf{x}$ *and* $\mathbf{y}$ *which satisfy* $\mathbf{y}^\top\mathtt{C}\mathbf{x} = 0$ *are* conjugate.

on 3D reconstruction and chapter 19 on auto-calibration, when obtaining a metric from a 3D projective reconstruction.

2.8 More properties of conics

We now introduce an important geometric relation between a point, line and conic, which is termed *polarity*. Applications of this relation (to the representation of orthogonality) are given in chapter 8.

2.8.1 The pole–polar relationship

A point $\mathbf{x}$ and conic $\mathtt{C}$ define a line $\mathbf{l} = \mathtt{C}\mathbf{x}$. The line $\mathbf{l}$ is called the *polar* of $\mathbf{x}$ with respect to $\mathtt{C}$, and the point $\mathbf{x}$ is the *pole* of $\mathbf{l}$ with respect to $\mathtt{C}$.

- *The polar line* $\mathbf{l} = \mathtt{C}\mathbf{x}$ *of the point* $\mathbf{x}$ *with respect to a conic* $\mathtt{C}$ *intersects the conic in two points. The two lines tangent to* $\mathtt{C}$ *at these points intersect at* $\mathbf{x}$*.*

This relationship is illustrated in figure 2.19.

Proof. Consider a point $\mathbf{y}$ on $\mathtt{C}$. The tangent line at $\mathbf{y}$ is $\mathtt{C}\mathbf{y}$, and this line contains $\mathbf{x}$ if $\mathbf{x}^\top\mathtt{C}\mathbf{y} = 0$. Using the symmetry of $\mathtt{C}$, the condition $\mathbf{x}^\top\mathtt{C}\mathbf{y} = (\mathtt{C}\mathbf{x})^\top\mathbf{y} = 0$ is that the point $\mathbf{y}$ lies on the line $\mathtt{C}\mathbf{x}$. Thus the polar line $\mathtt{C}\mathbf{x}$ intersects the conic in the point $\mathbf{y}$ at which the tangent line contains $\mathbf{x}$. □

As the point $\mathbf{x}$ approaches the conic the tangent lines become closer to collinear, and their contact points on the conic also become closer. In the limit that $\mathbf{x}$ lies on $\mathtt{C}$, the polar line has two-point contact at $\mathbf{x}$, and we have:

- *If the point* $\mathbf{x}$ *is on* $\mathtt{C}$ *then the polar is the tangent line to the conic at* $\mathbf{x}$*.*

See result 2.7(*p*31).

Samsung/Google, EX1010, p. 76

Example 2.28. A circle of radius r centred on the x-axis at $x = a$ has the equation $(x-a)^2 + y^2 = r^2$, and is represented by the conic matrix

$$\mathtt{C} = \begin{bmatrix} 1 & 0 & -a \\ 0 & 1 & 0 \\ -a & 0 & a^2 - r^2 \end{bmatrix}.$$

The polar line of the origin is given by $\mathbf{l} = \mathtt{C}(0,0,1)^\mathsf{T} = (-a, 0, a^2 - r^2)^\mathsf{T}$. This is a vertical line at $x = (a^2 - r^2)/a$. If $r = a$ the origin lies on the circle. In this case the polar line is the y-axis and is tangent to the circle. △

It is evident that the conic induces a map between points and lines of $\mathbb{P}^2$. This map is a projective construction since it involves only intersections and tangency, both properties that are preserved under projective transformations. A projective map between points and lines is termed a *correlation* (an unfortunate name, given its more common usage).

Definition 2.29. A *correlation* is an invertible mapping from points of $\mathbb{P}^2$ to lines of $\mathbb{P}^2$. It is represented by a 3×3 non-singular matrix $\mathtt{A}$ as $\mathbf{l} = \mathtt{A}\mathbf{x}$.

A correlation provides a systematic way to dualize relations involving points and lines. It need not be represented by a symmetric matrix, but we will only consider symmetric correlations here, because of the association with conics.

- **Conjugate points.** *If the point* $\mathbf{y}$ *is on the line* $\mathbf{l} = \mathtt{C}\mathbf{x}$ *then* $\mathbf{y}^\mathsf{T}\mathbf{l} = \mathbf{y}^\mathsf{T}\mathtt{C}\mathbf{x} = 0$. *Any two points* $\mathbf{x}, \mathbf{y}$ *satisfying* $\mathbf{y}^\mathsf{T}\mathtt{C}\mathbf{x} = 0$ *are* conjugate *with respect to the conic* $\mathtt{C}$.

  The conjugacy relation is symmetric:

- *If* $\mathbf{x}$ *is on the polar of* $\mathbf{y}$ *then* $\mathbf{y}$ *is on the polar of* $\mathbf{x}$.

This follows simply because of the symmetry of the conic matrix – the point $\mathbf{x}$ is on the polar of $\mathbf{y}$ if $\mathbf{x}^\mathsf{T}\mathtt{C}\mathbf{y} = 0$, and the point $\mathbf{y}$ is on the polar of $\mathbf{x}$ if $\mathbf{y}^\mathsf{T}\mathtt{C}\mathbf{x} = 0$. Since $\mathbf{x}^\mathsf{T}\mathtt{C}\mathbf{y} = \mathbf{y}^\mathsf{T}\mathtt{C}\mathbf{x}$, if one form is zero, then so is the other. There is a dual conjugacy relationship for lines: two lines $\mathbf{l}$ and $\mathbf{m}$ are conjugate if $\mathbf{l}^\mathsf{T}\mathtt{C}^*\mathbf{m} = 0$.

2.8.2 Classification of conics

This section describes the projective and affine classification of conics.

Projective normal form for a conic. Since $\mathtt{C}$ is a symmetric matrix it has real eigenvalues, and may be decomposed as a product $\mathtt{C} = \mathtt{U}^\mathsf{T}\mathtt{D}\mathtt{U}$ (see section A4.2(*p*580)), where $\mathtt{U}$ is an orthogonal matrix, and $\mathtt{D}$ is diagonal. Applying the projective transformation represented by $\mathtt{U}$, conic $\mathtt{C}$ is transformed to another conic $\mathtt{C}' = \mathtt{U}^{-\mathsf{T}}\mathtt{C}\mathtt{U}^{-1} = \mathtt{U}^{-\mathsf{T}}\mathtt{U}^\mathsf{T}\mathtt{D}\mathtt{U}\mathtt{U}^{-1} = \mathtt{D}$. This shows that any conic is equivalent under projective transformation to one with a diagonal matrix. Let $\mathtt{D} = \mathrm{diag}(\epsilon_1 d_1, \epsilon_2 d_2, \epsilon_3 d_3)$ where $\epsilon_i = \pm 1$ or 0 and each $d_i > 0$. Thus, $\mathtt{D}$ may be written in the form

$$\mathtt{D} = \mathrm{diag}(s_1, s_2, s_3)^\mathsf{T}\mathrm{diag}(\epsilon_1, \epsilon_2, \epsilon_3)\mathrm{diag}(s_1, s_2, s_3)$$

Samsung/Google, EX1010, p. 77

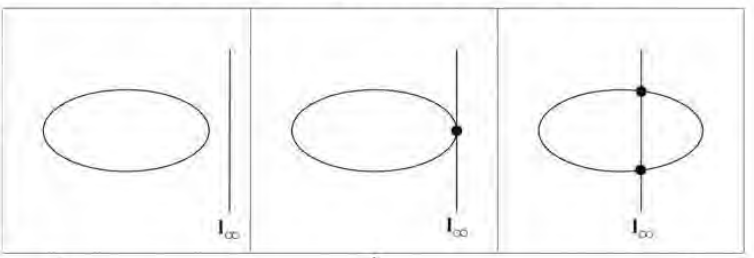


Fig. 2.20. **Affine classification of point conics.** *A conic is an (a) ellipse, (b) parabola, or (c) hyperbola; according to whether it (a) has no real intersection, (b) is tangent to (2-point contact), or (c) has 2 real intersections with l_∞. Under an affine transformation l_∞ is a fixed line, and intersections are preserved. Thus this classification is unaltered by an affinity.*

where $s_i^2 = d_i$. Note that $\mathrm{diag}(s_1, s_2, s_3)^\mathsf{T} = \mathrm{diag}(s_1, s_2, s_3)$. Now, transforming once more by the transformation $\mathrm{diag}(s_1, s_2, s_3)$, the conic D is transformed to a conic with matrix $\mathrm{diag}(\epsilon_1, \epsilon_2, \epsilon_3)$, with each $\epsilon_i = \pm 1$ or 0. Further transformation by permutation matrices may be carried out to ensure that values $\epsilon_i = 1$ occur before values $\epsilon_i = -1$ which in turn precede values $\epsilon_i = 0$. Finally, by multiplying by -1 if necessary, one may ensure that there are at least as many $+1$ entries as -1. The various types of conics may now be enumerated, and are shown in table 2.2.

Diagonal	Equation	Conic type
$(1,1,1)$	$x^2 + y^2 + w^2 = 0$	Improper conic – no real points.
$(1,1,-1)$	$x^2 + y^2 - w^2 = 0$	Circle
$(1,1,0)$	$x^2 + y^2 = 0$	Single real point $(0,0,1)^\mathsf{T}$
$(1,-1,0)$	$x^2 - y^2 = 0$	Two lines $x = \pm y$
$(1,0,0)$	$x^2 = 0$	Single line $x = 0$ counted twice.

Table 2.2. **Projective classification of point conics.** *Any plane conic is projectively equivalent to one of the types shown in this table. Those conics for which $\epsilon_i = 0$ for some i are known as degenerate conics, and are represented by a matrix of rank less than 3. The conic type column only describes the real points of the conics – for example as a complex conic $x^2 + y^2 = 0$ consists of the line pair $x = \pm iy$.*

Affine classification of conics. The classification of (non-degenerate, proper) conics in Euclidean geometry into hyperbola, ellipse and parabola is well known. As shown above in projective geometry these three types of conic are projectively equivalent to a circle. However, in affine geometry the Euclidean classification is still valid because it depends only on the relation of l_∞ to the conic. The relation for the three types of conic is illustrated in figure 2.20.

Samsung/Google, EX1010, p. 78

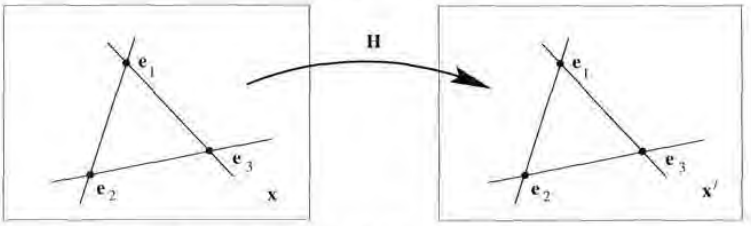


Fig. 2.21. **Fixed points and lines of a plane projective transformation.** *There are three fixed points, and three fixed lines through these points. The fixed lines and points may be complex. Algebraically, the fixed points are the eigenvectors, $\mathbf{e}_i$, of the point transformation ($\mathbf{x}' = \mathtt{H}\mathbf{x}$), and the fixed lines eigenvectors of the line transformation ($\mathbf{l}' = \mathtt{H}^{-\mathsf{T}}\mathbf{l}$). Note, the fixed line is* not *fixed pointwise: under the transformation, points on the line are mapped to other points on the line; only the fixed points are mapped to themselves.*

2.9 Fixed points and lines

We have seen, by the examples of $\mathbf{l}_\infty$ and the circular points, that points and lines may be fixed under a projective transformation. In this section the idea is investigated more thoroughly.

Here, the source and destination planes are identified (the same) so that the transformation maps points $\mathbf{x}$ to points $\mathbf{x}'$ in the same coordinate system. The key idea is that an *eigenvector* corresponds to a *fixed point* of the transformation, since for an eigenvector $\mathbf{e}$ with eigenvalue λ,

$$\mathtt{H}\mathbf{e} = \lambda\mathbf{e}$$

and $\mathbf{e}$ and $\lambda\mathbf{e}$ represent the same point. Often the eigenvector and eigenvalue have physical or geometric significance in computer vision applications.

A 3×3 matrix has three eigenvalues and consequently a plane projective transformation has up to three fixed points, if the eigenvalues are distinct. Since the characteristic equation is a cubic in this case, one or three of the eigenvalues, and corresponding eigenvectors, is real. A similar development can be given for *fixed lines*, which, since lines transform as (2.6–*p*36) $\mathbf{l}' = \mathtt{H}^{-\mathsf{T}}\mathbf{l}$, correspond to the eigenvectors of $\mathtt{H}^{\mathsf{T}}$.

The relationship between the fixed points and fixed lines is shown in figure 2.21. Note the lines are fixed as a set, not fixed pointwise, i.e. a point on the line is mapped to another point on the line, but in general the source and destination points will differ. There is nothing mysterious here: The projective transformation of the plane induces a 1D projective transformation on the line. A 1D projective transformation is represented by a 2×2 homogeneous matrix (section 2.5). This 1D projectivity has two fixed points corresponding to the two eigenvectors of the 2×2 matrix. These fixed points are those of the 2D projective transformation.

A further specialization concerns repeated eigenvalues. Suppose two of the eigenvalues (λ_2, λ_3 say) are identical, and that there are two distinct eigenvectors ($\mathbf{e}_2, \mathbf{e}_3$), corresponding to $\lambda_2 = \lambda_3$. Then the line containing the eigenvectors $\mathbf{e}_2, \mathbf{e}_3$ will be fixed pointwise, i.e. it is a line of fixed points. For suppose $\mathbf{x} = \alpha\mathbf{e}_2 + \beta\mathbf{e}_3$; then

$$\mathtt{H}\mathbf{x} = \lambda_2\alpha\mathbf{e}_2 + \lambda_2\beta\mathbf{e}_3 = \lambda_2\mathbf{x}$$

Samsung/Google, EX1010, p. 79

i.e. a point on the line through two degenerate eigenvectors is mapped to itself (only differing by scale). Another possibility is that $\lambda_2 = \lambda_3$, but that there is only one corresponding eigenvector. In this case, the eigenvector has *algebraic dimension* equal to two, but *geometric dimension* equal to one. Then there is one fewer fixed point (2 instead of 3). Various cases of repeated eigenvalues are discussed further in appendix 7(*p*628).

We now examine the fixed points and lines of the hierarchy of projective transformation subgroups of section 2.4. Affine transformations, and the more specialized forms, have two eigenvectors which are ideal points ($x_3 = 0$), and which correspond to the eigenvectors of the upper left 2×2 matrix. The third eigenvector is finite in general.

A Euclidean matrix. The two ideal fixed points are the complex conjugate pair of circular points $\mathbf{I}, \mathbf{J}$, with corresponding eigenvalues $\{e^{i\theta}, e^{-i\theta}\}$, where θ is the rotation angle. The third eigenvector, which has unit eigenvalue, is called the *pole*. The Euclidean transformation is equal to a pure rotation by θ about this point with no translation.

A special case is that of a pure translation (i.e. where $\theta = 0$). Here the eigenvalues are triply degenerate. The line at infinity is fixed pointwise, and there is a pencil of fixed lines through the point $(t_x, t_y, 0)^\top$ which corresponds to the translation direction. Consequently lines parallel to $\mathbf{t}$ are fixed. This is an example of an elation (see section A7.3(*p*631)).

A similarity matrix. The two ideal fixed points are again the circular points. The eigenvalues are $\{1, se^{i\theta}, se^{-i\theta}\}$. The action can be understood as a rotation and isotropic scaling by s about the finite fixed point. Note that the eigenvalues of the circular points again encode the angle of rotation.

An affine matrix. The two ideal fixed points can be real or complex conjugates, but the fixed line $\mathbf{l}_\infty = (0, 0, 1)^\top$ through these points is real in either case.

2.10 Closure

2.10.1 The literature

A gentle introduction to plane projective geometry, written for computer vision researchers, is given in the appendix of Mundy and Zisserman [Mundy-92]. A more formal approach is that of Semple and Kneebone [Semple-79], but [Springer-64] is more readable.

On the recovery of affine and metric scene properties for an imaged plane, Collins and Beveridge [Collins-93] use the vanishing line to recover affine properties from satellite images, and Liebowitz and Zisserman [Liebowitz-98] use metric information on the plane, such as right angles, to recover the metric geometry.

2.10.2 Notes and exercises

(i) **Affine transformations.**

Samsung/Google, EX1010, p. 80

(a) Show that an affine transformation can map a circle to an ellipse, but cannot map an ellipse to a hyperbola or parabola.

(b) Prove that under an affine transformation the ratio of lengths on parallel line segments is an invariant, but that the ratio of two lengths that are not parallel is not.

(ii) **Projective transformations.** Show that there is a three-parameter family of projective transformations which fix (as a set) a unit circle at the origin, i.e. a unit circle at the origin is mapped to a unit circle at the origin (hint, use result 2.13(*p*37) to compute the transformation). What is the geometric interpretation of this family?

(iii) **Isotropies.** Show that two lines have an invariant under a similarity transformation; and that two lines and two points have an invariant under a projective transformation. In both cases the equality case of the counting argument (result 2.16(*p*43)) is violated. Show that for these two cases the respective transformation cannot be fully determined, although it is partially determined.

(iv) **Invariants.** Using the transformation rules for points, lines and conics show:

(a) Two lines, $\mathbf{l}_1$, $\mathbf{l}_2$, and two points, $\mathbf{x}_1$, $\mathbf{x}_2$, not lying on the lines have the invariant

$$I = \frac{(\mathbf{l}_1^\top \mathbf{x}_1)(\mathbf{l}_2^\top \mathbf{x}_2)}{(\mathbf{l}_1^\top \mathbf{x}_2)(\mathbf{l}_2^\top \mathbf{x}_1)}$$

(see the previous question).

(b) A conic $\mathbf{C}$ and two points, $\mathbf{x}_1$ and $\mathbf{x}_2$, in general position have the invariant

$$I = \frac{(\mathbf{x}_1^\top \mathbf{C} \mathbf{x}_2)^2}{(\mathbf{x}_1^\top \mathbf{C} \mathbf{x}_1)(\mathbf{x}_2^\top \mathbf{C} \mathbf{x}_2)}.$$

(c) Show that the projectively invariant expression for measuring angles (2.22) is equivalent to Laguerre's projectively invariant expression involving a cross ratio with the circular points (see [Springer-64]).

(v) **The cross ratio.** Prove the invariance of the cross ratio of four collinear points under projective transformations of the line (2.18–*p*45). Hint, start with the transformation of two points on the line written as $\bar{\mathbf{x}}_i' = \lambda_i \mathbf{H}_{2\times 2} \bar{\mathbf{x}}_i$ and $\bar{\mathbf{x}}_j' = \lambda_j \mathbf{H}_{2\times 2} \bar{\mathbf{x}}_j$, where equality is *not* up to scale, then from the properties of determinants show that $|\bar{\mathbf{x}}_i' \bar{\mathbf{x}}_j'| = \lambda_i \lambda_j \det \mathbf{H}_{2\times 2} |\bar{\mathbf{x}}_i \bar{\mathbf{x}}_j|$ and continue from here. An alternative derivation method is given in [Semple-79].

(vi) **Polarity.** Figure 2.19 shows the geometric construction of the polar line for a point $\mathbf{x}$ *outside* an ellipse. Give a geometric construction for the polar when the point is inside. Hint, start by choosing any line through $\mathbf{x}$. The pole of this line is a point on the polar of $\mathbf{x}$.

(vii) **Conics.** If the sign of the conic matrix $\mathbf{C}$ is chosen such that two eigenvalues are positive and one negative, then internal and external points may be distinguished according to the sign of $\mathbf{x}^\top \mathbf{C} \mathbf{x}$: the point $\mathbf{x}$ is inside/on/outside the conic

Samsung/Google, EX1010, p. 81

$\mathtt{C}$ if $\mathbf{x}^\top\mathtt{C}\mathbf{x}$ is negative/zero/positive respectively. This can seen by example from a circle $\mathtt{C} = \mathrm{diag}(1, 1, -1)$. Under projective transformations internality is invariant, though its interpretation requires care in the case of an ellipse being transformed to a hyperbola (see figure 2.20).

(viii) **Dual conics.** Show that the matrix $[\mathtt{l}]_\times\mathtt{C}[\mathtt{l}]_\times$ represents a rank 2 dual conic which consists of the two points at which the line l intersects the (point) conic $\mathtt{C}$ (the notation $[\mathtt{l}]_\times$ is defined in (A4.5–*p*581)).

(ix) **Special projective transformations.** Suppose points on a scene plane are related by reflection in a line: for example, a plane object with bilateral symmetry. Show that in a perspective image of the plane the points are related by a projectivity $\mathtt{H}$ satisfying $\mathtt{H}^2 = \mathtt{I}$. Furthermore, show that under $\mathtt{H}$ there is a line of fixed points corresponding to the imaged reflection line, and that $\mathtt{H}$ has an eigenvector, not lying on this line, which is the vanishing point of the reflection direction ($\mathtt{H}$ is a planar harmonic homology, see section A7.2(*p*629)).

Now suppose that the points are related by a finite rotational symmetry: for example, points on a hexagonal bolt head. Show in this case that $\mathtt{H}^n = \mathtt{I}$, where n is the order of rotational symmetry (6 for a hexagonal symmetry), that the eigenvalues of $\mathtt{H}$ determine the rotation angle, and that the eigenvector corresponding to the real eigenvalue is the image of the centre of the rotational symmetry.

Samsung/Google, EX1010, p. 82

3

Projective Geometry and Transformations of 3D

This chapter describes the properties and entities of projective 3-space, or $\mathbb{P}^3$. Many of these are straightforward generalizations of those of the projective plane, described in chapter 2. For example, in $\mathbb{P}^3$ Euclidean 3-space is augmented with a set of ideal points which are on a *plane* at infinity, π_∞. This is the analogue of l_∞ in $\mathbb{P}^2$. Parallel lines, and now parallel *planes*, intersect on π_∞. Not surprisingly, homogeneous coordinates again play an important role, here with all dimensions increased by one. However, additional properties appear by virtue of the extra dimension. For example, two lines always intersect on the projective plane, but they need not intersect in 3-space.

The reader should be familiar with the ideas and notation of chapter 2 before reading this chapter. We will concentrate here on the differences and additional geometry introduced by adding the extra dimension, and will not repeat the bulk of the material of the previous chapter.

3.1 Points and projective transformations

A point $\mathbf{X}$ in 3-space is represented in homogeneous coordinates as a 4-vector. Specifically, the homogeneous vector $\mathbf{X} = (X_1, X_2, X_3, X_4)^\top$ with $X_4 \neq 0$ represents the point $(X, Y, Z)^\top$ of $\mathbb{R}^3$ with inhomogeneous coordinates

$$X = X_1/X_4, \;\; Y = X_2/X_4, \;\; Z = X_3/X_4.$$

For example, a homogeneous representation of $(X, Y, Z)^\top$ is $\mathbf{X} = (X, Y, Z, 1)^\top$. Homogeneous points with $X_4 = 0$ represent points at infinity.

A projective transformation acting on $\mathbb{P}^3$ is a linear transformation on homogeneous 4-vectors represented by a non-singular 4×4 matrix: $\mathbf{X}' = \mathtt{H}\mathbf{X}$. The matrix $\mathtt{H}$ representing the transformation is homogeneous and has 15 degrees of freedom. The degrees of freedom follow from the 16 elements of the matrix less one for overall scaling.

As in the case of planar projective transformations, the map is a collineation (lines are mapped to lines), which preserves incidence relations such as the intersection point of a line with a plane, and order of contact.

Samsung/Google, EX1010, p. 83

3.2 Representing and transforming planes, lines and quadrics

In $\mathbb{P}^3$ points and *planes* are dual, and their representation and development is analogous to the point–line duality in $\mathbb{P}^2$. Lines are self-dual in $\mathbb{P}^3$.

3.2.1 Planes

A plane in 3-space may be written as

$$\pi_1 X + \pi_2 Y + \pi_3 Z + \pi_4 = 0. \tag{3.1}$$

Clearly this equation is unaffected by multiplication by a non-zero scalar, so only the three independent ratios $\{\pi_1 : \pi_2 : \pi_3 : \pi_4\}$ of the plane coefficients are significant. It follows that a plane has 3 degrees of freedom in 3-space. The homogeneous representation of the plane is the 4-vector $\boldsymbol{\pi} = (\pi_1, \pi_2, \pi_3, \pi_4)^\top$.

Homogenizing (3.1) by the replacements $X \mapsto X_1/X_4$, $Y \mapsto X_2/X_4$, $Z \mapsto X_3/X_4$ gives

$$\pi_1 X_1 + \pi_2 X_2 + \pi_3 X_3 + \pi_4 X_4 = 0$$

or more concisely

$$\boldsymbol{\pi}^\top \mathbf{X} = 0 \tag{3.2}$$

which expresses that the point $\mathbf{X}$ is on the plane $\boldsymbol{\pi}$.

The first 3 components of $\boldsymbol{\pi}$ correspond to the plane normal of Euclidean geometry – using inhomogeneous notation (3.2) becomes the familiar plane equation written in 3-vector notation as $\mathbf{n}.\widetilde{\mathbf{X}} + d = 0$, where $\mathbf{n} = (\pi_1, \pi_2, \pi_3)^\top$, $\widetilde{\mathbf{X}} = (X, Y, Z)^\top$, $X_4 = 1$ and $d = \pi_4$. In this form $d/\|\mathbf{n}\|$ is the distance of the plane from the origin.

Join and incidence relations. In $\mathbb{P}^3$ there are numerous geometric relations between planes and points and lines. For example,

(i) A plane is defined uniquely by the join of three points, or the join of a line and point, in general position (i.e. the points are not collinear or incident with the line in the latter case).
(ii) Two distinct planes intersect in a unique line.
(iii) Three distinct planes intersect in a unique point.

These relations have algebraic representations which will now be developed in the case of points and planes. The representations of the relations involving lines are not as simple as those arising from 3D vector algebra of $\mathbb{P}^2$ (e.g. $\mathbf{l} = \mathbf{x} \times \mathbf{y}$), and are postponed until line representations are introduced in section 3.2.2.

Three points define a plane. Suppose three points $\mathbf{X}_i$ are incident with the plane $\boldsymbol{\pi}$. Then each point satisfies (3.2) and thus $\boldsymbol{\pi}^\top \mathbf{X}_i = 0$, $i = 1, \ldots, 3$. Stacking these equations into a matrix gives

$$\begin{bmatrix} \mathbf{X}_1^\top \\ \mathbf{X}_2^\top \\ \mathbf{X}_3^\top \end{bmatrix} \boldsymbol{\pi} = 0. \tag{3.3}$$

Samsung/Google, EX1010, p. 84

Since three points $\mathbf{X}_1, \mathbf{X}_2$ and $\mathbf{X}_3$ in general position are linearly independent, it follows that the 3×4 matrix composed of the points as rows has rank 3. The plane π defined by the points is thus obtained uniquely (up to scale) as the 1-dimensional (right) null-space. If the matrix has only a rank of 2, and consequently the null-space is 2-dimensional, then the points are collinear, and define a pencil of planes with the line of collinear points as axis.

In $\mathbb{P}^2$, where points are dual to lines, a line $\mathbf{l}$ through two points $\mathbf{x}, \mathbf{y}$ can similarly be obtained as the null-space of the 2×3 matrix with $\mathbf{x}^\mathsf{T}$ and $\mathbf{y}^\mathsf{T}$ as rows. However, a more convenient direct formula $\mathbf{l} = \mathbf{x} \times \mathbf{y}$ is also available from vector algebra. In $\mathbb{P}^3$ the analogous expression is obtained from properties of determinants and minors.

We start from the matrix $\mathtt{M} = [\mathbf{X}, \mathbf{X}_1, \mathbf{X}_2, \mathbf{X}_3]$ which is composed of a general point $\mathbf{X}$ and the three points $\mathbf{X}_i$ which define the plane π. The determinant $\det \mathtt{M} = 0$ when $\mathbf{X}$ lies on π since the point $\mathbf{X}$ is then expressible as a linear combination of the points $\mathbf{X}_i, i = 1, \ldots, 3$. Expanding the determinant about the column $\mathbf{X}$ we obtain

$$\det \mathtt{M} = \mathtt{X}_1 D_{234} - \mathtt{X}_2 D_{134} + \mathtt{X}_3 D_{124} - \mathtt{X}_4 D_{123}$$

where D_{jkl} is the determinant formed from the jkl rows of the 4×3 matrix $[\mathbf{X}_1, \mathbf{X}_2, \mathbf{X}_3]$. Since $\det \mathtt{M} = 0$ for points on π we can then read off the plane coefficients as

$$\pi = (D_{234}, -D_{134}, D_{124}, -D_{123})^\mathsf{T}. \tag{3.4}$$

This is the solution vector (the null-space) of (3.3) above.

Example 3.1. Suppose the three points defining the plane are

$$\mathbf{X}_1 = \begin{pmatrix} \widetilde{\mathbf{X}}_1 \\ 1 \end{pmatrix} \quad \mathbf{X}_2 = \begin{pmatrix} \widetilde{\mathbf{X}}_2 \\ 1 \end{pmatrix} \quad \mathbf{X}_3 = \begin{pmatrix} \widetilde{\mathbf{X}}_3 \\ 1 \end{pmatrix}$$

where $\widetilde{\mathbf{X}} = (\mathtt{X}, \mathtt{Y}, \mathtt{Z})^\mathsf{T}$. Then

$$D_{234} = \begin{vmatrix} \mathtt{Y}_1 & \mathtt{Y}_2 & \mathtt{Y}_3 \\ \mathtt{Z}_1 & \mathtt{Z}_2 & \mathtt{Z}_3 \\ 1 & 1 & 1 \end{vmatrix} = \begin{vmatrix} \mathtt{Y}_1 - \mathtt{Y}_3 & \mathtt{Y}_2 - \mathtt{Y}_3 & \mathtt{Y}_3 \\ \mathtt{Z}_1 - \mathtt{Z}_3 & \mathtt{Z}_2 - \mathtt{Z}_3 & \mathtt{Z}_3 \\ 0 & 0 & 1 \end{vmatrix} = \left((\widetilde{\mathbf{X}}_1 - \widetilde{\mathbf{X}}_3) \times (\widetilde{\mathbf{X}}_2 - \widetilde{\mathbf{X}}_3) \right)_1$$

and similarly for the other components, giving

$$\pi = \begin{pmatrix} (\widetilde{\mathbf{X}}_1 - \widetilde{\mathbf{X}}_3) \times (\widetilde{\mathbf{X}}_2 - \widetilde{\mathbf{X}}_3) \\ -\widetilde{\mathbf{X}}_3^\mathsf{T} (\widetilde{\mathbf{X}}_1 \times \widetilde{\mathbf{X}}_2) \end{pmatrix}.$$

This is the familiar result from Euclidean vector geometry where, for example, the plane normal is computed as $(\widetilde{\mathbf{X}}_1 - \widetilde{\mathbf{X}}_3) \times (\widetilde{\mathbf{X}}_2 - \widetilde{\mathbf{X}}_3)$. △

Three planes define a point. The development here is dual to the case of three points defining a plane. The intersection point $\mathbf{X}$ of three planes π_i can be computed straightforwardly as the (right) null-space of the 3×4 matrix composed of the planes as rows:

$$\begin{bmatrix} \pi_1^\mathsf{T} \\ \pi_2^\mathsf{T} \\ \pi_3^\mathsf{T} \end{bmatrix} \mathbf{X} = 0. \tag{3.5}$$

Samsung/Google, EX1010, p. 85

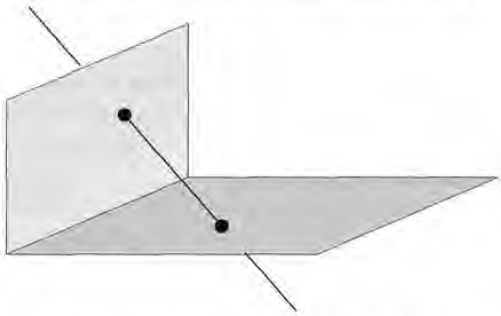


Fig. 3.1. *A line may be specified by its points of intersection with two orthogonal planes. Each intersection point has 2 degrees of freedom, which demonstrates that a line in* $\mathbb{P}^3$ *has a total of 4 degrees of freedom.*

A direct solution for $\mathbf{X}$, in terms of determinants of 3×3 submatrices, is obtained as an analogue of (3.4), though computationally a numerical solution would be obtained by algorithm A5.1($p589$).
The two following results are direct analogues of their 2D counterparts.

Projective transformation. Under the point transformation $\mathbf{X}' = \mathtt{H}\mathbf{X}$, a plane transforms as

$$\boldsymbol{\pi}' = \mathtt{H}^{-\mathsf{T}}\boldsymbol{\pi}. \tag{3.6}$$

Parametrized points on a plane. The points $\mathbf{X}$ on the plane $\boldsymbol{\pi}$ may be written as

$$\mathbf{X} = \mathtt{M}\mathbf{x} \tag{3.7}$$

where the columns of the 4×3 matrix $\mathtt{M}$ generate the rank 3 null-space of $\boldsymbol{\pi}^\mathsf{T}$, i.e. $\boldsymbol{\pi}^\mathsf{T}\mathtt{M} = \mathbf{0}$, and the 3-vector $\mathbf{x}$ (which is a point on the projective plane $\mathbb{P}^2$) parametrizes points on the plane $\boldsymbol{\pi}$. $\mathtt{M}$ is not unique, of course. Suppose the plane is $\boldsymbol{\pi} = (a, b, c, d)^\mathsf{T}$ and a is non-zero, then $\mathtt{M}^\mathsf{T}$ can be written as $\mathtt{M}^\mathsf{T} = [\mathbf{p} \mid \mathtt{I}_{3\times3}]$, where $\mathbf{p} = (-b/a, -c/a, -d/a)^\mathsf{T}$.

This parametrized representation is simply the analogue in 3D of a line $\mathbf{l}$ in $\mathbb{P}^2$ defined as a linear combination of its 2D null-space as $\mathbf{x} = \mu\mathbf{a} + \lambda\mathbf{b}$, where $\mathbf{l}^\mathsf{T}\mathbf{a} = \mathbf{l}^\mathsf{T}\mathbf{b} = 0$.

3.2.2 Lines

A line is defined by the *join* of two points or the intersection of two planes. Lines have 4 degrees of freedom in 3-space. A convincing way to count these degrees of freedom is to think of a line as defined by its intersection with two orthogonal planes, as in figure 3.1. The point of intersection on each plane is specified by two parameters, producing a total of 4 degrees of freedom for the line.

Lines are very awkward to represent in 3-space since a natural representation for an object with 4 degrees of freedom would be a homogeneous 5-vector. The problem is that a homogeneous 5 vector cannot easily be used in mathematical expressions together with the 4-vectors representing points and planes. To overcome this problem

Samsung/Google, EX1010, p. 86

a number of line representations have been proposed, and these differ in their mathematical complexity. We survey three of these representations. In each case the representation provides mechanisms for a line to be defined by: the join of two points, a dual version where the line is defined by the intersection of two planes, and also a map between the two definitions. The representations also enable join and incidence relations to be computed, for example the point at which a line intersects a plane.

I. Null-space and span representation. This representation builds on the intuitive geometric notion that a line is a pencil (one-parameter family) of collinear points, and is defined by any two of these points. Similarly, a line is the axis of a pencil of planes, and is defined by the intersection of any two planes from the pencil. In both cases the actual points or planes are not important (in fact two points have 6 degrees of freedom and are represented by two 4-vectors – far too many parameters). This notion is captured mathematically by representing a line as the *span* of two vectors. Suppose $\mathbf{A}, \mathbf{B}$ are two (non-coincident) space points. Then the line joining these points is represented by the span of the row space of the 2×4 matrix $\mathtt{W}$ composed of $\mathbf{A}^\top$ and $\mathbf{B}^\top$ as rows:

$$\mathtt{W} = \begin{bmatrix} \mathbf{A}^\top \\ \mathbf{B}^\top \end{bmatrix}.$$

Then:

(i) The span of $\mathtt{W}^\top$ is the pencil of points $\lambda\mathbf{A} + \mu\mathbf{B}$ on the line.
(ii) The span of the 2-dimensional right null-space of $\mathtt{W}$ is the pencil of planes with the line as axis.

It is evident that two other points, $\mathbf{A}'^\top$ and $\mathbf{B}'^\top$, on the line will generate a matrix $\mathtt{W}'$ with the same span as $\mathtt{W}$, so that the span, and hence the representation, is independent of the particular points used to define it.

To prove the null-space property, suppose that $\mathbf{P}$ and $\mathbf{Q}$ are a basis for the null-space. Then $\mathtt{W}\mathbf{P} = \mathbf{0}$ and consequently $\mathbf{A}^\top\mathbf{P} = \mathbf{B}^\top\mathbf{P} = 0$, so that $\mathbf{P}$ is a plane containing the points $\mathbf{A}$ and $\mathbf{B}$. Similarly, $\mathbf{Q}$ is a distinct plane also containing the points $\mathbf{A}$ and $\mathbf{B}$. Thus $\mathbf{A}$ and $\mathbf{B}$ lie on both the (linearly independent) planes $\mathbf{P}$ and $\mathbf{Q}$, so the line defined by $\mathtt{W}$ is the plane intersection. Any plane of the pencil, with the line as axis, is given by the span $\lambda'\mathbf{P} + \mu'\mathbf{Q}$.

The dual representation of a line as the intersection of two planes, $\mathbf{P}, \mathbf{Q}$, follows in a similar manner. The line is represented as the span (of the row space) of the 2×4 matrix $\mathtt{W}^*$ composed of $\mathbf{P}^\top$ and $\mathbf{Q}^\top$ as rows:

$$\mathtt{W}^* = \begin{bmatrix} \mathbf{P}^\top \\ \mathbf{Q}^\top \end{bmatrix}$$

with the properties

(i) The span of $\mathtt{W}^{*\top}$ is the pencil of planes $\lambda'\mathbf{P} + \mu'\mathbf{Q}$ with the line as axis.
(ii) The span of the 2-dimensional null-space of $\mathtt{W}^*$ is the pencil of points on the line.

Samsung/Google, EX1010, p. 87

The two representations are related by $\mathtt{W}^*\mathtt{W}^\mathsf{T} = \mathtt{W}\mathtt{W}^{*\mathsf{T}} = \mathtt{0}_{2\times2}$, where $\mathtt{0}_{2\times2}$ is a 2×2 null matrix.

Example 3.2. The X-axis is represented as

$$\mathtt{W} = \begin{bmatrix} 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \end{bmatrix} \qquad \mathtt{W}^* = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix}$$

where the points A and B are here the origin and ideal point in the X-direction, and the planes P and Q are the XY- and XZ-planes respectively. △

Join and incidence relations are also computed from null-spaces.

(i) The plane π defined by the join of the point X and line W is obtained from the null-space of

$$\mathtt{M} = \begin{bmatrix} \mathtt{W} \\ \mathbf{X}^\mathsf{T} \end{bmatrix}.$$

If the null-space of M is 2-dimensional then X is on W, otherwise $\mathtt{M}\pi = \mathbf{0}$.

(ii) The point X defined by the intersection of the line W with the plane π is obtained from the null-space of

$$\mathtt{M} = \begin{bmatrix} \mathtt{W}^* \\ \pi^\mathsf{T} \end{bmatrix}.$$

If the null-space of M is 2-dimensional then the line W is on π, otherwise $\mathtt{M}\mathbf{X} = \mathbf{0}$.

These properties can be derived almost by inspection. For example, the first is equivalent to three points defining a plane (3.3).

The span representation is very useful in practical numerical implementations where null-spaces can be computed simply by using the SVD algorithm (see section A4.4-(p585)) available with most matrix packages. The representation is also useful in estimation problems, where it is often not a problem that the entity being estimated is over-parametrized (see the discussion of section 4.5(p110)).

II. Plücker matrices. Here a line is represented by a 4×4 skew-symmetric homogeneous matrix. In particular, the line joining the two points A, B is represented by the matrix L with elements

$$l_{ij} = A_i B_j - B_i A_j$$

or equivalently in vector notation as

$$\mathtt{L} = \mathbf{A}\mathbf{B}^\mathsf{T} - \mathbf{B}\mathbf{A}^\mathsf{T} \tag{3.8}$$

First a few properties of L:

(i) L has rank 2. Its 2-dimensional null-space is spanned by the pencil of planes with the line as axis (in fact $\mathtt{L}\mathtt{W}^{*\mathsf{T}} = \mathtt{0}$, with 0 a 4×2 null-matrix).

Samsung/Google, EX1010, p. 88

(ii) The representation has the required 4 degrees of freedom for a line. This is accounted as follows: the skew-symmetric matrix has 6 independent non-zero elements, but only their 5 ratios are significant, and furthermore because $\det \mathtt{L} = 0$ the elements satisfy a (quadratic) constraint (see below). The net number of degrees of freedom is then 4.

(iii) The relation $\mathtt{L} = \mathbf{A}\mathbf{B}^\top - \mathbf{B}\mathbf{A}^\top$ is the generalization to 4-space of the vector product formula $\mathbf{l} = \mathbf{x} \times \mathbf{y}$ of $\mathbb{P}^2$ for a line $\mathbf{l}$ defined by two points $\mathbf{x}, \mathbf{y}$ all represented by 3-vectors.

(iv) The matrix $\mathtt{L}$ is independent of the points $\mathbf{A}, \mathbf{B}$ used to define it, since if a different point $\mathbf{C}$ on the line is used, with $\mathbf{C} = \mathbf{A} + \mu\mathbf{B}$, then the resulting matrix is

$$\begin{aligned} \hat{\mathtt{L}} &= \mathbf{A}\mathbf{C}^\top - \mathbf{C}\mathbf{A}^\top = \mathbf{A}(\mathbf{A}^\top + \mu\mathbf{B}^\top) - (\mathbf{A} + \mu\mathbf{B})\mathbf{A}^\top \\ &= \mathbf{A}\mathbf{B}^\top - \mathbf{B}\mathbf{A}^\top = \mathtt{L}. \end{aligned}$$

(v) Under the point transformation $\mathbf{X}' = \mathtt{H}\mathbf{X}$, the matrix transforms as $\mathtt{L}' = \mathtt{H}\mathtt{L}\mathtt{H}^\top$, i.e. it is a valency-2 tensor (see appendix 1(p562)).

Example 3.3. From (3.8) the X-axis is represented as

$$\mathtt{L} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 1 \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 & 0 \end{pmatrix} - \begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \end{pmatrix} \begin{pmatrix} 0 & 0 & 0 & 1 \end{pmatrix} = \begin{bmatrix} 0 & 0 & 0 & -1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix}$$

where the points $\mathbf{A}$ and $\mathbf{B}$ are (as in the previous example) the origin and ideal point in the X-direction respectively. △

A dual Plücker representation $\mathtt{L}^*$ is obtained for a line formed by the intersection of two planes $\mathbf{P}, \mathbf{Q}$,

$$\mathtt{L}^* = \mathbf{P}\mathbf{Q}^\top - \mathbf{Q}\mathbf{P}^\top \tag{3.9}$$

and has similar properties to $\mathtt{L}$. Under the point transformation $\mathbf{X}' = \mathtt{H}\mathbf{X}$, the matrix $\mathtt{L}^*$ transforms as $\mathtt{L}^{*\prime} = \mathtt{H}^{-\top}\mathtt{L}\mathtt{H}^{-1}$. The matrix $\mathtt{L}^*$ can be obtained directly from $\mathtt{L}$ by a simple rewrite rule:

$$l_{12} : l_{13} : l_{14} : l_{23} : l_{42} : l_{34} = l^*_{34} : l^*_{42} : l^*_{23} : l^*_{14} : l^*_{13} : l^*_{12}. \tag{3.10}$$

The correspondence rule is very simple: the indices of the dual and original component always include all the numbers $\{1, 2, 3, 4\}$, so if the original is ij then the dual is those numbers of $\{1, 2, 3, 4\}$ which are not ij. For example $12 \mapsto 34$.

Join and incidence properties are very nicely represented in this notation:

(i) The plane defined by the join of the point $\mathbf{X}$ and line $\mathtt{L}$ is

$$\boldsymbol{\pi} = \mathtt{L}^*\mathbf{X}$$

and $\mathtt{L}^*\mathbf{X} = \mathbf{0}$ if, and only if, $\mathbf{X}$ is on $\mathtt{L}$.

Samsung/Google, EX1010, p. 89

(ii) The point defined by the intersection of the line $\mathtt{L}$ with the plane π is

$$\mathbf{X} = \mathtt{L}\pi$$

and $\mathtt{L}\pi = \mathbf{0}$ if, and only if, $\mathtt{L}$ is on π.

The properties of two (or more) lines $\mathtt{L}_1, \mathtt{L}_2, \ldots$ can be obtained from the null-space of the matrix $\mathtt{M} = [\mathtt{L}_1, \mathtt{L}_2, \ldots]$. For example if the lines are coplanar then $\mathtt{M}^\mathsf{T}$ has a 1-dimensional null-space corresponding to the plane π of the lines.

Example 3.4. The intersection of the X-axis with the plane $\mathtt{X} = 1$ is given by $\mathbf{X} = \mathtt{L}\pi$ as

$$\mathbf{X} = \begin{bmatrix} 0 & 0 & 0 & -1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix} \begin{pmatrix} 1 \\ 0 \\ 0 \\ -1 \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \\ 0 \\ 1 \end{pmatrix}$$

which is the inhomogeneous point $(\mathtt{X}, \mathtt{Y}, \mathtt{Z})^\mathsf{T} = (1, 0, 0)^\mathsf{T}$. △

III. Plücker line coordinates. The Plücker line coordinates are the six non-zero elements of the 4×4 skew-symmetric Plücker matrix (3.8) $\mathtt{L}$, namely[1]

$$\mathcal{L} = \{l_{12}, l_{13}, l_{14}, l_{23}, l_{42}, l_{34}\}. \tag{3.11}$$

This is a homogeneous 6-vector, and thus is an element of $\mathbb{P}^5$. It follows from evaluating $\det \mathtt{L} = 0$ that the coordinates satisfy the equation

$$l_{12}l_{34} + l_{13}l_{42} + l_{14}l_{23} = 0. \tag{3.12}$$

A 6-vector $\mathcal{L}$ only corresponds to a line in 3-space if it satisfies (3.12). The geometric interpretation of this constraint is that the lines of $\mathbb{P}^3$ define a (co-dimension 1) surface in $\mathbb{P}^5$ which is known as the *Klein quadric*, a quadric because the terms of (3.12) are quadratic in the Plücker line coordinates.

Suppose two lines $\mathcal{L}$, $\hat{\mathcal{L}}$ are the joins of the points $\mathbf{A}, \mathbf{B}$ and $\hat{\mathbf{A}}, \hat{\mathbf{B}}$ respectively. The lines intersect if and only if the four points are coplanar. A necessary and sufficient condition for this is that $\det[\mathbf{A}, \mathbf{B}, \hat{\mathbf{A}}, \hat{\mathbf{B}}] = 0$. It can be shown that the determinant expands as

$$\begin{aligned} \det[\mathbf{A}, \mathbf{B}, \hat{\mathbf{A}}, \hat{\mathbf{B}}] &= l_{12}\hat{l}_{34} + \hat{l}_{12}l_{34} + l_{13}\hat{l}_{42} + \hat{l}_{13}l_{42} + l_{14}\hat{l}_{23} + \hat{l}_{14}l_{23} \\ &= (\mathcal{L}|\hat{\mathcal{L}}). \end{aligned} \tag{3.13}$$

Since the Plücker coordinates are independent of the particular points used to define them, the bilinear product $(\mathcal{L}|\hat{\mathcal{L}})$ is independent of the points used in the derivation and only depends on the lines $\mathcal{L}$ and $\hat{\mathcal{L}}$. Then we have

Result 3.5. *Two lines $\mathcal{L}$ and $\hat{\mathcal{L}}$ are coplanar (and thus intersect) if and only if $(\mathcal{L}|\hat{\mathcal{L}}) = 0$.*

This product appears in a number of useful formulae:

[1] The element l_{42} is conventionally used instead of l_{24} as it eliminates negatives in many of the subsequent formulae.

Samsung/Google, EX1010, p. 90

(i) A 6-vector $\mathcal{L}$ only represents a line in $\mathbb{P}^3$ if $(\mathcal{L}|\mathcal{L}) = 0$. This is simply repeating the Klein quadric constraint (3.12) above.

(ii) Suppose two lines $\mathcal{L}$, $\hat{\mathcal{L}}$ are the intersections of the planes $\mathbf{P}$, $\mathbf{Q}$ and $\widehat{\mathbf{P}}$, $\widehat{\mathbf{Q}}$ respectively. Then

$$(\mathcal{L}|\hat{\mathcal{L}}) = \det[\mathbf{P}, \mathbf{Q}, \widehat{\mathbf{P}}, \widehat{\mathbf{Q}}]$$

and again the lines intersect if and only if $(\mathcal{L}|\hat{\mathcal{L}}) = 0$.

(iii) If $\mathcal{L}$ is the intersection of two planes $\mathbf{P}$ and $\mathbf{Q}$ and $\hat{\mathcal{L}}$ is the join of two points $\mathbf{A}$ and $\mathbf{B}$, then

$$(\mathcal{L}|\hat{\mathcal{L}}) = (\mathbf{P}^\top\mathbf{A})(\mathbf{Q}^\top\mathbf{B}) - (\mathbf{Q}^\top\mathbf{A})(\mathbf{P}^\top\mathbf{B}). \tag{3.14}$$

Plücker coordinates are useful in algebraic derivations. They will be used in defining the map from a line in 3-space to its image in chapter 8.

3.2.3 Quadrics and dual quadrics

A quadric is a surface in $\mathbb{P}^3$ defined by the equation

$$\mathbf{X}^\top\mathtt{Q}\mathbf{X} = 0 \tag{3.15}$$

where $\mathtt{Q}$ is a symmetric 4×4 matrix. Often the matrix $\mathtt{Q}$ and the quadric surface it defines are not distinguished, and we will simply refer to the quadric $\mathtt{Q}$.

Many of the properties of quadrics follow directly from those of conics in section 2.2.3(*p*30). To highlight a few:

(i) A quadric has 9 degrees of freedom. These correspond to the ten independent elements of a 4×4 symmetric matrix less one for scale.

(ii) Nine points in general position define a quadric.

(iii) If the matrix $\mathtt{Q}$ is singular, then the quadric is *degenerate*, and may be defined by fewer points.

(iv) A quadric defines a polarity between a point and a plane, in a similar manner to the polarity defined by a conic between a point and a line (section 2.8.1). The plane $\pi = \mathtt{Q}\mathbf{X}$ is the polar plane of $\mathbf{X}$ with respect to $\mathtt{Q}$. In the case that $\mathtt{Q}$ is non-singular and $\mathbf{X}$ is outside the quadric, the polar plane is defined by the points of contact with $\mathtt{Q}$ of the cone of rays through $\mathbf{X}$ tangent to $\mathtt{Q}$. If $\mathbf{X}$ lies on $\mathtt{Q}$, then $\mathtt{Q}\mathbf{X}$ is the tangent plane to $\mathtt{Q}$ at $\mathbf{X}$.

(v) The intersection of a plane π with a quadric $\mathtt{Q}$ is a conic $\mathtt{C}$. Computing the conic can be tricky because it requires a coordinate system for the plane. Recall from (3.7) that a coordinate system for the plane can be defined by the complement space to π as $\mathbf{X} = \mathtt{M}\mathbf{x}$. Points on π are on $\mathtt{Q}$ if $\mathbf{X}^\top\mathtt{Q}\mathbf{X} = \mathbf{x}^\top\mathtt{M}^\top\mathtt{Q}\mathtt{M}\mathbf{x} = 0$. These points lie on a conic $\mathtt{C}$, since $\mathbf{x}^\top\mathtt{C}\mathbf{x} = 0$, with $\mathtt{C} = \mathtt{M}^\top\mathtt{Q}\mathtt{M}$.

(vi) Under the point transformation $\mathbf{X}' = \mathtt{H}\mathbf{X}$, a (point) quadric transforms as

$$\mathtt{Q}' = \mathtt{H}^{-\top}\mathtt{Q}\mathtt{H}^{-1}. \tag{3.16}$$

The dual of a quadric is also a quadric. Dual quadrics are equations on planes: the tangent planes π to the point quadric $\mathtt{Q}$ satisfy $\pi^\top\mathtt{Q}^*\pi = 0$, where $\mathtt{Q}^* = \text{adjoint } \mathtt{Q}$.

Samsung/Google, EX1010, p. 91

or $\mathtt{Q}^{-1}$ if $\mathtt{Q}$ is invertible. Under the point transformation $\mathbf{X}' = \mathtt{H}\mathbf{X}$, a dual quadric transforms as

$$\mathtt{Q}^{*\prime} = \mathtt{H}\mathtt{Q}^{*}\mathtt{H}^{\mathsf{T}}. \tag{3.17}$$

The algebra of imaging a quadric is far simpler for a dual quadric than a point quadric. This is detailed in chapter 8.

3.2.4 Classification of quadrics

Since the matrix, $\mathtt{Q}$, representing a quadric is symmetric, it may be decomposed as $\mathtt{Q} = \mathtt{U}^{\mathsf{T}}\mathtt{D}\mathtt{U}$ where $\mathtt{U}$ is a real orthogonal matrix and $\mathtt{D}$ is a real diagonal matrix. Further, by appropriate scaling of the rows of $\mathtt{U}$, one may write $\mathtt{Q} = \mathtt{H}^{\mathsf{T}}\mathtt{D}\mathtt{H}$ where $\mathtt{D}$ is diagonal with entries equal to $0, 1$, or -1. We may further ensure that the zero entries of $\mathtt{D}$ appear last along the diagonal, and that the $+1$ entries appear first. Now, replacement of $\mathtt{Q} = \mathtt{H}^{\mathsf{T}}\mathtt{D}\mathtt{H}$ by $\mathtt{D}$ is equivalent to a projective transformation effected by the matrix $\mathtt{H}$ (see (3.16)). Thus, up to projective equivalence, we may assume that the quadric is represented by a matrix $\mathtt{D}$ of the given simple form.

The *signature* of a diagonal matrix $\mathtt{D}$, denoted $\sigma(\mathtt{D})$, is defined to be the number of $+1$ entries minus the number of -1 entries. This definition is extended to arbitrary real symmetric matrices $\mathtt{Q}$ by defining $\sigma(\mathtt{Q}) = \sigma(\mathtt{D})$ such that $\mathtt{Q} = \mathtt{H}^{\mathsf{T}}\mathtt{D}\mathtt{H}$, where $\mathtt{H}$ is a real matrix. It may be proved that the signature is well defined, being independent of the particular choice of $\mathtt{H}$. Since the matrix representing a quadric is defined only up to sign, we may assume that its signature is non-negative. Then, the projective type of a quadric is uniquely determined by its rank and signature. This will allow us to enumerate the different projective equivalence classes of quadrics.

A quadric represented by a diagonal matrix $\mathrm{diag}(d_1, d_2, d_3, d_4)$ corresponds to a set of points satisfying an equation $d_1\mathtt{X}^2 + d_2\mathtt{Y}^2 + d_3\mathtt{Z}^2 + d_4\mathtt{T}^2 = 0$. One may set $\mathtt{T} = 1$ to get an equation for the non-infinite points on the quadric. See table 3.1. Examples of quadric surfaces are shown in figure 3.2 – figure 3.4.

Rank	σ	Diagonal	Equation	Realization
4	4	$(1,1,1,1)$	$\mathtt{X}^2+\mathtt{Y}^2+\mathtt{Z}^2+1=0$	No real points
	2	$(1,1,1,-1)$	$\mathtt{X}^2+\mathtt{Y}^2+\mathtt{Z}^2=1$	Sphere
	0	$(1,1,-1,-1)$	$\mathtt{X}^2+\mathtt{Y}^2=\mathtt{Z}^2+1$	Hyperboloid of one sheet
3	3	$(1,1,1,0)$	$\mathtt{X}^2+\mathtt{Y}^2+\mathtt{Z}^2=0$	One point $(0,0,0,1)^{\mathsf{T}}$
	1	$(1,1,-1,0)$	$\mathtt{X}^2+\mathtt{Y}^2=\mathtt{Z}^2$	Cone at the origin
2	2	$(1,1,0,0)$	$\mathtt{X}^2+\mathtt{Y}^2=0$	Single line (Z-axis)
	0	$(1,-1,0,0)$	$\mathtt{X}^2=\mathtt{Y}^2$	Two planes $\mathtt{X}=\pm\mathtt{Y}$
1	1	$(1,0,0,0)$	$\mathtt{X}^2=0$	The plane $\mathtt{X}=0$

Table 3.1. *Categorization of point quadrics.*

Samsung/Google, EX1010, p. 92

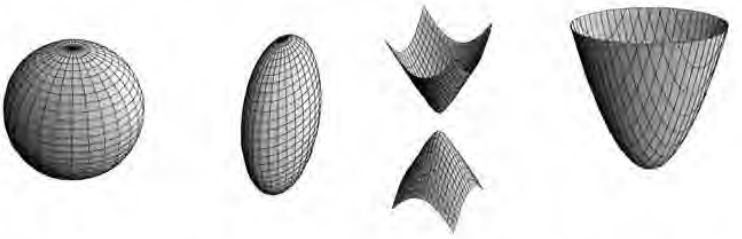


Fig. 3.2. **Non-ruled quadrics.** *This shows plots of a sphere, ellipsoid, hyperboloid of two sheets and paraboloid. They are all projectively equivalent.*

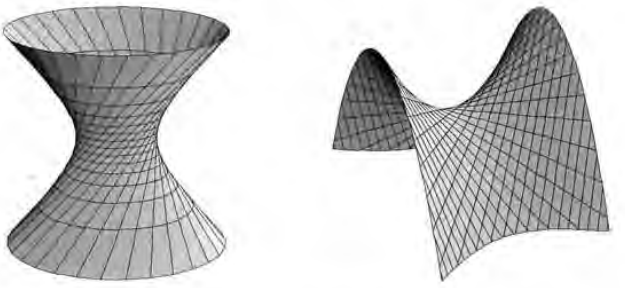


Fig. 3.3. **Ruled quadrics.** *Two examples of a hyperboloid of one sheet are given. These surfaces are given by equations* $X^2 + Y^2 = Z^2 + 1$ *and* $XY = Z$ *respectively, and are projectively equivalent. Note that these two surfaces are made up of two sets of disjoint straight lines, and that each line from one set meets each line from the other set. The two quadrics shown here are projectively (though not affinely) equivalent.*

Ruled quadrics. Quadrics fall into two classes – ruled and unruled quadrics. A ruled quadric is one that contains a straight line. More particularly, as shown in figure 3.3, the non-degenerate ruled quadric (hyperboloid of one sheet) contains two families of straight lines called *generators*. For more properties of ruled quadrics, refer to [Semple-79].

The most interesting of the quadrics are the two quadrics of rank 4. Note that these two quadrics differ even in their topological type. The quadric of signature 2 (the sphere) is (obviously enough) topologically a sphere. On the other hand, the hyperboloid of 1 sheet is *not* topologically equivalent (homeomorphic) to a sphere. In fact, it is topologically a torus (topologically equivalent to $S^1 \times S^1$). This gives the clearest indication that they are not projectively equivalent.

3.3 Twisted cubics

The twisted cubic may be considered to be a 3-dimensional analogue of a 2D conic (although in other ways it is a quadric surface which is the 3-dimensional analogue of a 2D conic.)

Samsung/Google, EX1010, p. 93

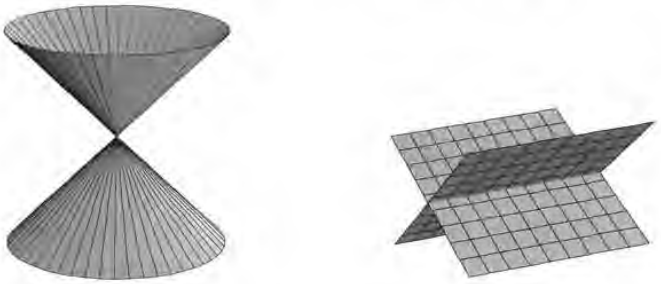


Fig. 3.4. **Degenerate quadrics.** *The two most important degenerate quadrics are shown, the cone and two planes. Both these quadrics are ruled. The matrix representing the cone has rank 3, and the null-vector represents the nodal point of the cone. The matrix representing the two (non-coincident) planes has rank 2, and the two generators of the rank 2 null-space are two points on the intersection line of the planes.*

A conic in the 2-dimensional projective plane may be described as a parametrized curve given by the equation

$$\begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = \mathtt{A} \begin{pmatrix} 1 \\ \theta \\ \theta^2 \end{pmatrix} = \begin{pmatrix} a_{11} + a_{12}\theta + a_{13}\theta^2 \\ a_{21} + a_{22}\theta + a_{23}\theta^2 \\ a_{31} + a_{32}\theta + a_{33}\theta^2 \end{pmatrix} \tag{3.18}$$

where $\mathtt{A}$ is a non-singular 3×3 matrix.

In an analogous manner, a twisted cubic is defined to be a curve in $\mathbb{P}^3$ given in parametric form as

$$\begin{pmatrix} \mathrm{X}_1 \\ \mathrm{X}_2 \\ \mathrm{X}_3 \\ \mathrm{X}_4 \end{pmatrix} = \mathtt{A} \begin{pmatrix} 1 \\ \theta \\ \theta^2 \\ \theta^3 \end{pmatrix} = \begin{pmatrix} a_{11} + a_{12}\theta + a_{13}\theta^2 + a_{14}\theta^3 \\ a_{21} + a_{22}\theta + a_{23}\theta^2 + a_{24}\theta^3 \\ a_{31} + a_{32}\theta + a_{33}\theta^2 + a_{34}\theta^3 \\ a_{41} + a_{42}\theta + a_{43}\theta^2 + a_{44}\theta^3 \end{pmatrix} \tag{3.19}$$

where $\mathtt{A}$ is a non-singular 4×4 matrix.

Since a twisted cubic is perhaps an unfamiliar object, various views of the curve are shown in figure 3.5. In fact, a twisted cubic is a quite benign space curve.

Properties of a twisted cubic. Let $\mathbf{c}$ be a non-singular twisted cubic. Then $\mathbf{c}$ is not contained within any plane of $\mathbb{P}^3$; it intersects a general plane at three distinct points. A twisted cubic has 12 degrees of freedom (counted as 15 for the matrix $\mathtt{A}$, less 3 for a 1D projectivity on the parametrization θ, which leaves the curve unaltered). Requiring the curve to pass through a point $\mathbf{X}$ places two constraints on $\mathbf{c}$, since $\mathbf{X} = \mathtt{A}(1, \theta, \theta^2, \theta^3)^\mathsf{T}$ is three independent ratios, but only two constraints once θ is eliminated. Thus, there is a unique $\mathbf{c}$ through six points in general position. Finally, all non-degenerate twisted cubics are projectively equivalent. This is clear from the definition (3.19): a projective transformation $\mathtt{A}^{-1}$ maps $\mathbf{c}$ to the standard form $\mathbf{c}(\theta') = (1, \theta', \theta'^2, \theta'^3)^\mathsf{T}$, and since

Samsung/Google, EX1010, p. 94



Fig. 3.5. *Various views of the twisted cubic* $(t^3, t^2, t,)^\mathsf{T}$. *The curve is thickened to a tube to aid in visualization.*

all twisted cubics can be mapped to this curve, it follows that all twisted cubics are projectively equivalent.

A classification of the various special cases of a twisted cubic, such as a conic and coincident line, are given in [Semple-79]. The twisted cubic makes an appearance as the horopter for two-view geometry (chapter 9), and plays the central role in defining the degenerate set for camera resectioning (chapter 22).

3.4 The hierarchy of transformations

There are a number of specializations of a projective transformation of 3-space which will appear frequently throughout this book. The specializations are analogous to the strata of section 2.4(p37) for planar transformations. Each specialization is a subgroup, and is identified by its matrix form, or equivalently by its invariants. These are summarized in table 3.2. This table lists only the *additional* properties of the 3-space transformations over their 2-space counterparts – the transformations of 3-space also have the invariants listed in table 2.1(p44) for the corresponding 2-space transformations.

The 15 degrees of freedom of a projective transformation are accounted for as seven for a similarity (three for rotation, three for translation, one for isotropic scaling), five for affine scalings, and three for the projective part of the transformation.

Two of the most important characterizations of these transformations are parallelism and angles. For example, after an affine transformation lines which were originally parallel remain parallel, but angles are skewed; and after a projective transformation parallelism is lost.

In the following we briefly describe a decomposition of a Euclidean transformation that will be useful when discussing special motions later in this book.

3.4.1 The screw decomposition

A Euclidean transformation on the plane may be considered as a specialization of a Euclidean transformation of 3-space with the restrictions that the translation vector $\mathbf{t}$ lies in the plane, and the rotation axis is perpendicular to the plane. However, Euclidean actions on 3-space are more general than this because the rotation axis and translation are not perpendicular in general. The screw decomposition enables any Euclidean

Samsung/Google, EX1010, p. 95

Group	Matrix	Distortion	Invariant properties
Projective 15 dof	$\begin{bmatrix} \mathtt{A} & \mathbf{t} \\ \mathbf{v}^\top & v \end{bmatrix}$		Intersection and tangency of surfaces in contact. Sign of Gaussian curvature.
Affine 12 dof	$\begin{bmatrix} \mathtt{A} & \mathbf{t} \\ \mathbf{0}^\top & 1 \end{bmatrix}$		Parallelism of planes, volume ratios, centroids. The plane at infinity, π_∞, (see section 3.5).
Similarity 7 dof	$\begin{bmatrix} s\mathtt{R} & \mathbf{t} \\ \mathbf{0}^\top & 1 \end{bmatrix}$		The absolute conic, Ω_∞, (see section 3.6).
Euclidean 6 dof	$\begin{bmatrix} \mathtt{R} & \mathbf{t} \\ \mathbf{0}^\top & 1 \end{bmatrix}$		Volume.

Table 3.2. **Geometric properties invariant to commonly occurring transformations of 3-space.** *The matrix* $\mathtt{A}$ *is an invertible* 3×3 *matrix,* $\mathtt{R}$ *is a 3D rotation matrix,* $\mathbf{t} = (t_X, t_Y, t_Z)^\top$ *a 3D translation,* $\mathbf{v}$ *a general 3-vector,* v *a scalar, and* $\mathbf{0} = (0,0,0)^\top$ *a null 3-vector. The distortion column shows typical effects of the transformations on a cube. Transformations higher in the table can produce all the actions of the ones below. These range from Euclidean, where only translations and rotations occur, to projective where five points can be transformed to any other five points (provided no three points are collinear, or four coplanar).*

action (a rotation composed with a translation) to be reduced to a situation almost as simple as the 2D case. The screw decomposition is that

Result 3.6. *Any particular translation and rotation is equivalent to a rotation about a* screw axis *together with a translation along the screw axis. The screw axis is parallel to the rotation axis.*

In the case of a translation and an *orthogonal* rotation axis (termed *planar motion*), the motion is equivalent to a rotation *alone* about the screw axis.

Proof. We will sketch a constructive geometric proof that can easily be visualized. Consider first the 2D case – a Euclidean transformation on the plane. It is evident from figure 3.6 that a screw axis exists for such 2D transformations. For the 3D case, decompose the translation $\mathbf{t}$ into two components $\mathbf{t} = \mathbf{t}_\parallel + \mathbf{t}_\perp$, parallel and orthogonal respectively to the rotation axis direction ($\mathbf{t}_\parallel = (\mathbf{t}.\mathbf{a})\mathbf{a}$, $\mathbf{t}_\perp = \mathbf{t} - (\mathbf{t}.\mathbf{a})\mathbf{a}$).
Then the Euclidean motion is partitioned into two parts: first a rotation about the screw

Samsung/Google, EX1010, p. 96

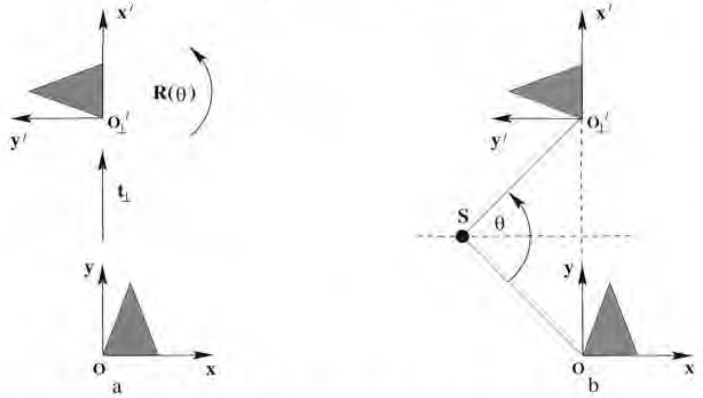


Fig. 3.6. **2D Euclidean motion and a "screw" axis.** *(a) The frame $\{x, y\}$ undergoes a translation $\mathbf{t}_\perp$ and a rotation by θ to reach the frame $\{x', y'\}$. The motion is in the plane orthogonal to the rotation axis. (b) This motion is equivalent to a single rotation about the screw axis $\mathbf{S}$. The screw axis lies on the perpendicular bisector of the line joining corresponding points, such that the angle between the lines joining $\mathbf{S}$ to the corresponding points is θ. In the figure the corresponding points are the two frame origins and θ has the value $90°$.*

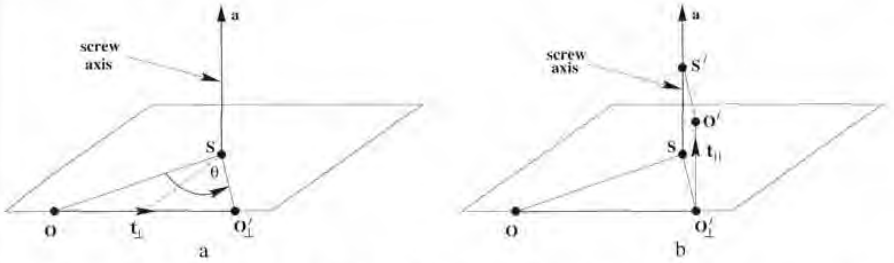


Fig. 3.7. **3D Euclidean motion and the screw decomposition.** *Any Euclidean rotation $\mathtt{R}$ and translation $\mathbf{t}$ may be achieved by (a) a rotation about the screw axis, followed by (b) a translation along the screw axis by $\mathbf{t}_\parallel$. Here $\mathbf{a}$ is the (unit) direction of the rotation axis (so that $\mathtt{R}\mathbf{a} = \mathbf{a}$), and $\mathbf{t}$ is decomposed as $\mathbf{t} = \mathbf{t}_\parallel + \mathbf{t}_\perp$, which are vector components parallel and orthogonal respectively to the rotation axis direction The point $\mathbf{S}$ is closest to $\mathbf{O}$ on the screw axis (so that the line from $\mathbf{S}$ to $\mathbf{O}$ is perpendicular to the direction of $\mathbf{a}$). Similarly $\mathbf{S}'$ is the point on the screw axis closest to $\mathbf{O}'$.*

axis, which covers the rotation and $\mathbf{t}_\perp$; second a translation by $\mathbf{t}_\parallel$ along the screw axis. The complete motion is illustrated in figure 3.7. □

The screw decomposition can be determined from the fixed points of the 4×4 matrix representing the Euclidean transformation. This idea is examined in the exercises at the end of the chapter.

3.5 The plane at infinity

In planar projective geometry identifying the line at infinity, $\mathbf{l}_\infty$, allowed affine properties of the plane to be measured. Identifying the circular points on $\mathbf{l}_\infty$ then allowed

Samsung/Google, EX1010, p. 97

the measurement of metric properties. In the projective geometry of 3-space the corresponding geometric entities are the plane at infinity, π_∞, and the absolute conic, Ω_∞.

The plane at infinity has the canonical position $\pi_\infty = (0, 0, 0, 1)^\mathsf{T}$ in affine 3-space. It contains the directions $\mathtt{D} = (\mathtt{X}_1, \mathtt{X}_2, \mathtt{X}_3, 0)^\mathsf{T}$, and enables the identification of affine properties such as parallelism. In particular:

- Two planes are parallel if, and only if, their line of intersection is on π_∞.
- A line is parallel to another line, or to a plane, if the point of intersection is on π_∞.

We then have in $\mathbb{P}^3$ that any pair of planes intersect in a line, with parallel planes intersecting in a line on the plane at infinity.

The plane π_∞ is a geometric representation of the 3 degrees of freedom required to specify affine properties in a projective coordinate frame. In loose terms, the plane at infinity is a fixed plane under any affine transformation, but "sees" (is moved by) a projective transformation. The 3 degrees of freedom of π_∞ thus measure the projective component of a general homography – they account for the 15 degrees of freedom of this general transformation compared to an affinity (12 dof). More formally:

Result 3.7. *The plane at infinity, π_∞, is a fixed plane under the projective transformation* $\mathtt{H}$ *if, and only if,* $\mathtt{H}$ *is an affinity.*

The proof is the analogue of the derivation of result 2.17(*p*48). It is worth clarifying two points:

(i) The plane π_∞ is, in general, only fixed as a set under an affinity; it is not fixed pointwise.

(ii) Under a particular affinity (for example a Euclidean motion) there may be planes in addition to π_∞ which are fixed. However, only π_∞ is fixed under any affinity.

These points are illustrated in more detail by the following example.

Example 3.8. Consider the Euclidean transformation represented by the matrix

$$\mathtt{H}_\mathrm{E} = \begin{bmatrix} \mathtt{R} & \mathbf{0} \\ \mathbf{0}^\mathsf{T} & 1 \end{bmatrix} = \begin{bmatrix} \cos\theta & -\sin\theta & 0 & 0 \\ \sin\theta & \cos\theta & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}. \tag{3.20}$$

This is a rotation by θ about the Z-axis with a zero translation (it is a planar screw motion, see section 3.4.1). Geometrically it is evident that the family of XY-planes orthogonal to the rotation axis are simply rotated about the Z-axis by this transformation. This means that there is a pencil of fixed planes orthogonal to the Z-axis. The planes are fixed as sets, but not pointwise as any (finite) point (not on the axis) is rotated in horizontal circles by this Euclidean action. Algebraically, the fixed planes of $\mathtt{H}$ are the eigenvectors of $\mathtt{H}^\mathsf{T}$ (refer to section 2.9). In this case the eigenvalues are $\{e^{i\theta}, e^{-i\theta}, 1, 1\}$

Samsung/Google, EX1010, p. 98

and the corresponding eigenvectors of $\mathtt{H}_E^{\mathsf{T}}$ are

$$\mathbf{E}_1 = \begin{pmatrix} 1 \\ i \\ 0 \\ 0 \end{pmatrix} \quad \mathbf{E}_2 = \begin{pmatrix} 1 \\ -i \\ 0 \\ 0 \end{pmatrix} \quad \mathbf{E}_3 = \begin{pmatrix} 0 \\ 0 \\ 1 \\ 0 \end{pmatrix} \quad \mathbf{E}_4 = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 1 \end{pmatrix}.$$

The eigenvectors $\mathbf{E}_1$ and $\mathbf{E}_2$ do not correspond to real planes, and will not be discussed further here. The eigenvectors $\mathbf{E}_3$ and $\mathbf{E}_4$ are degenerate. Thus there is a pencil of fixed planes which is spanned by these eigenvectors. The axis of this pencil is the line of intersection of the the planes (perpendicular to the Z-axis) with π_∞, and the pencil includes π_∞. △

The example also illustrates the connection between the geometry of the projective plane, $\mathbb{P}^2$, and projective 3-space, $\mathbb{P}^3$. A plane π intersects π_∞ in a line which is the line at infinity, $\mathtt{l}_\infty$, of the plane π. A projective transformation of $\mathbb{P}^3$ induces a *subordinate* plane projective transformation on π.

Affine properties of a reconstruction. In later chapters on reconstruction, for example chapter 10, it will be seen that the projective coordinates of the (Euclidean) scene can be reconstructed from multiple views. Once π_∞ is identified in projective 3-space, i.e. its projective coordinates are known, it is then possible to determine affine properties of the reconstruction such as whether geometric entities are parallel – they are parallel if they intersect on π_∞.

A more algorithmic approach is to transform $\mathbb{P}^3$ so that the identified π_∞ is moved to its canonical position at $\pi_\infty = (0,0,0,1)^{\mathsf{T}}$. After this mapping we then have the situation that the Euclidean scene, where π_∞ has the coordinates $(0,0,0,1)^{\mathsf{T}}$, and our reconstruction are related by a projective transformation that fixes π_∞ at $(0,0,0,1)^{\mathsf{T}}$. It follows from result 3.7 that the scene and reconstruction are related by an affine transformation. Thus affine properties can now be measured directly from the coordinates of the entities.

3.6 The absolute conic

The absolute conic, Ω_∞, is a (point) conic on π_∞. In a metric frame $\pi_\infty = (0,0,0,1)^{\mathsf{T}}$, and points on Ω_∞ satisfy

$$\left.\begin{aligned} \mathtt{X}_1^2 + \mathtt{X}_2^2 + \mathtt{X}_3^2 \\ \mathtt{X}_4 \end{aligned}\right\} = 0. \tag{3.21}$$

Note that two equations are required to define Ω_∞.

For directions on π_∞ (i.e. points with $\mathtt{X}_4 = 0$) the defining equation can be written

$$(\mathtt{X}_1, \mathtt{X}_2, \mathtt{X}_3)\mathtt{I}(\mathtt{X}_1, \mathtt{X}_2, \mathtt{X}_3)^{\mathsf{T}} = 0$$

so that Ω_∞ corresponds to a conic $\mathtt{C}$ with matrix $\mathtt{C} = \mathtt{I}$. It is thus a conic of purely imaginary points on π_∞.

Samsung/Google, EX1010, p. 99

The conic Ω_∞ is a geometric representation of the 5 additional degrees of freedom that are required to specify metric properties in an affine coordinate frame. A key property of Ω_∞ is that it is a fixed conic under any similarity transformation. More formally:

Result 3.9. *The absolute conic, Ω_∞, is a fixed conic under the projective transformation* $\mathtt{H}$ *if, and only if,* $\mathtt{H}$ *is a similarity transformation.*

Proof. Since the absolute conic lies in the plane at infinity, a transformation fixing it must fix the plane at infinity, and hence must be affine. Such a transformation is of the form

$$\mathtt{H}_\mathtt{A} = \begin{bmatrix} \mathtt{A} & \mathbf{t} \\ \mathbf{0}^\mathsf{T} & 1 \end{bmatrix}.$$

Restricting to the plane at infinity, the absolute conic is represented by the matrix $\mathtt{I}_{3\times3}$, and since it is fixed by $\mathtt{H}_\mathtt{A}$, one has $\mathtt{A}^{-\mathsf{T}}\mathtt{I}\mathtt{A}^{-1} = \mathtt{I}$ (up to scale), and taking inverses gives $\mathtt{A}\mathtt{A}^\mathsf{T} = \mathtt{I}$. This means that $\mathtt{A}$ is orthogonal, hence a scaled rotation, or scaled rotation with reflection. This completes the proof. □

Even though Ω_∞ does not have any real points, it shares the properties of any conic – such as that a line intersects a conic in two points; the pole–polar relationship etc. Here are a few particular properties of Ω_∞:

(i) Ω_∞ is only fixed as a set by a general similarity; it is not fixed pointwise. This means that under a similarity a point on Ω_∞ may travel to another point on Ω_∞, but it is not mapped to a point off the conic.
(ii) All circles intersect Ω_∞ in two points. Suppose the support plane of the circle is π. Then π intersects π_∞ in a line, and this line intersects Ω_∞ in two points. These two points are the circular points of π.
(iii) All spheres intersect π_∞ in Ω_∞.

Metric properties. Once Ω_∞ (and its support plane π_∞) have been identified in projective 3-space then metric properties, such as angles and relative lengths, can be measured.

Consider two lines with directions (3-vectors) $\mathbf{d}_1$ and $\mathbf{d}_2$. The angle between these directions in a Euclidean world frame is given by

$$\cos\theta = \frac{(\mathbf{d}_1^\mathsf{T}\mathbf{d}_2)}{\sqrt{(\mathbf{d}_1^\mathsf{T}\mathbf{d}_1)(\mathbf{d}_2^\mathsf{T}\mathbf{d}_2)}}. \tag{3.22}$$

This may be written as

$$\cos\theta = \frac{(\mathbf{d}_1^\mathsf{T}\Omega_\infty\mathbf{d}_2)}{\sqrt{(\mathbf{d}_1^\mathsf{T}\Omega_\infty\mathbf{d}_1)(\mathbf{d}_2^\mathsf{T}\Omega_\infty\mathbf{d}_2)}} \tag{3.23}$$

where $\mathbf{d}_1$ and $\mathbf{d}_2$ are the points of intersection of the lines with the plane π_∞ containing the conic Ω_∞, and Ω_∞ is the matrix representation of the absolute conic in that plane.

Samsung/Google, EX1010, p. 100

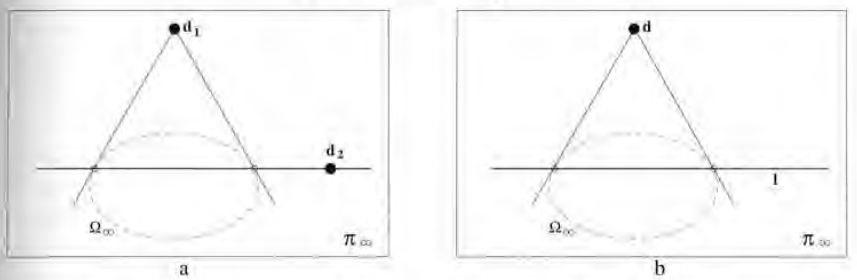


Fig. 3.8. **Orthogonality and Ω_∞.** *(a) On π_∞ orthogonal directions $\mathbf{d}_1$, $\mathbf{d}_2$ are conjugate with respect to Ω_∞. (b) A plane normal direction $\mathbf{d}$ and the intersection line $\mathbf{l}$ of the plane with π_∞ are in pole–polar relation with respect to Ω_∞.*

The expression (3.23) reduces to (3.22) in a Euclidean world frame where $\Omega_\infty = \mathtt{I}$. However, the expression is valid in any projective coordinate frame as may be verified from the transformation properties of points and conics (see (iv)(b) on page 63).

There is no simple formula for the angle between two planes computed from the directions of their surface normals.

Orthogonality and polarity. We now give a geometric representation of orthogonality in a projective space based on the absolute conic. The main device will be the pole–polar relationship between a point and line induced by a conic.

An immediate consequence of (3.23) is that two directions $\mathbf{d}_1$ and $\mathbf{d}_2$ are orthogonal if $\mathbf{d}_1^\top \Omega_\infty \mathbf{d}_2 = 0$. Thus orthogonality is encoded by *conjugacy* with respect to Ω_∞. The great advantage of this is that conjugacy is a projective relation, so that in a projective frame (obtained by a projective transformation of Euclidean 3-space) directions can be identified as orthogonal if they are conjugate with respect to Ω_∞ in that frame (in general the matrix of Ω_∞ is not $\mathtt{I}$ in a projective frame). The geometric representation of orthogonality is shown in figure 3.8.

This representation is helpful when considering orthogonality between rays in a camera, for example in determining the normal to a plane through the camera centre (see section 8.6(p213)). If image points are conjugate with respect to the *image* of Ω_∞ then the corresponding rays are orthogonal.

Again, a more algorithmic approach is to projectively transform the coordinates so that Ω_∞ is mapped to its canonical position (3.21), and then metric properties can be determined directly from coordinates.

3.7 The absolute dual quadric

Recall that Ω_∞ is defined by *two* equations – it is a conic on the plane at infinity. The dual of the absolute conic Ω_∞ is a degenerate dual *quadric* in 3-space called the *absolute dual quadric*, and denoted $\mathtt{Q}^*_\infty$. Geometrically $\mathtt{Q}^*_\infty$ consists of the planes tangent to Ω_∞, so that Ω_∞ is the "rim" of $\mathtt{Q}^*_\infty$. This is called a *rim quadric*. Think of the set of planes tangent to an ellipsoid, and then squash the ellipsoid to a pancake.

Algebraically $\mathtt{Q}^*_\infty$ is represented by a 4×4 homogeneous matrix of rank 3, which in

Samsung/Google, EX1010, p. 101

metric 3-space has the canonical form

$$\mathtt{Q}_\infty^* = \begin{bmatrix} \mathtt{I} & \mathbf{0} \\ \mathbf{0}^\mathsf{T} & 0 \end{bmatrix}. \tag{3.24}$$

We will show that any plane in the dual absolute quadric envelope is indeed tangent to Ω_∞, so the $\mathtt{Q}_\infty^*$ is truly a dual of Ω_∞. Consider a plane represented by $\boldsymbol{\pi} = (\mathbf{v}^\mathsf{T}, k)^\mathsf{T}$. This plane is in the envelope defined by $\mathtt{Q}_\infty^*$ if and only if $\boldsymbol{\pi}^\mathsf{T}\mathtt{Q}_\infty^*\boldsymbol{\pi} = 0$, which given the form (3.24) is equivalent to $\mathbf{v}^\mathsf{T}\mathbf{v} = 0$. Now, (see section 8.6($p213$)) $\mathbf{v}$ represents the line in which the plane $(\mathbf{v}^\mathsf{T}, k)^\mathsf{T}$ meets the plane at infinity. This line is tangent to the absolute conic if and only if $\mathbf{v}^\mathsf{T}\mathtt{I}\mathbf{v} = 0$. Thus, the envelope of $\mathtt{Q}_\infty^*$ is made up of just those planes tangent to the absolute conic.

Since this is an important fact, we consider it from another angle. Consider the absolute conic as the limit of a series of squashed ellipsoids, namely quadrics represented by the matrix $\mathtt{Q} = \mathrm{diag}(1, 1, 1, k)$. As $k \to \infty$, these quadrics become increasingly close to the plane at infinity, and in the limit the only points they contain are the points $(\mathtt{X}_1, \mathtt{X}_2, \mathtt{X}_3, 0)^\mathsf{T}$ with $\mathtt{X}_1^2 + \mathtt{X}_2^2 + \mathtt{X}_3^2 = 0$, that is points on the absolute conic. However, the dual of $\mathtt{Q}$ is the quadric $\mathtt{Q}^* = \mathtt{Q}^{-1} = \mathrm{diag}(1, 1, 1, k^{-1})$, which in the limit becomes the absolute dual quadric $\mathtt{Q}_\infty^* = \mathrm{diag}(1, 1, 1, 0)$.

The dual quadric $\mathtt{Q}_\infty^*$ is a degenerate quadric and has 8 degrees of freedom (a symmetric matrix has 10 independent elements, but the irrelevant scale and zero determinant condition each reduce the degrees of freedom by 1). It is a geometric representation of the 8 degrees of freedom that are required to specify metric properties in a projective coordinate frame. $\mathtt{Q}_\infty^*$ has a significant advantage over Ω_∞ in algebraic manipulations because both $\boldsymbol{\pi}_\infty$ and Ω_∞ are contained in a single geometric object (unlike Ω_∞ which requires two equations (3.21) in order to specify it). In the following we give its three most important properties.

Result 3.10. *The absolute dual quadric, $\mathtt{Q}_\infty^*$, is fixed under the projective transformation $\mathtt{H}$ if, and only if, $\mathtt{H}$ is a similarity.*

Proof. This follows directly from the invariance of the absolute conic under a similarity transform, since the planar tangency relationship between $\mathtt{Q}_\infty^*$ and Ω_∞ is transformation invariant. Nevertheless, we give an independent direct proof.
Since $\mathtt{Q}_\infty^*$ is a dual quadric, it transforms according to (3.17–$p74$), so it is fixed under $\mathtt{H}$ if and only if $\mathtt{Q}_\infty^* = \mathtt{H}\mathtt{Q}_\infty^*\mathtt{H}^\mathsf{T}$. Applying this with an arbitrary transform

$$\mathtt{H} = \begin{bmatrix} \mathtt{A} & \mathbf{t} \\ \mathbf{v}^\mathsf{T} & k \end{bmatrix}$$

we find

$$\begin{aligned} \begin{bmatrix} \mathtt{I} & \mathbf{0} \\ \mathbf{0}^\mathsf{T} & 0 \end{bmatrix} &= \begin{bmatrix} \mathtt{A} & \mathbf{t} \\ \mathbf{v}^\mathsf{T} & k \end{bmatrix} \begin{bmatrix} \mathtt{I} & \mathbf{0} \\ \mathbf{0}^\mathsf{T} & 0 \end{bmatrix} \begin{bmatrix} \mathtt{A}^\mathsf{T} & \mathbf{v} \\ \mathbf{t}^\mathsf{T} & k \end{bmatrix} \\ &= \begin{bmatrix} \mathtt{A}\mathtt{A}^\mathsf{T} & \mathtt{A}\mathbf{v} \\ \mathbf{v}^\mathsf{T}\mathtt{A}^\mathsf{T} & \mathbf{v}^\mathsf{T}\mathbf{v} \end{bmatrix} \end{aligned}$$

Samsung/Google, EX1010, p. 102

which must be true up to scale. By inspection, this equation holds if and only if $\mathbf{v} = \mathbf{0}$ and $\mathtt{A}$ is a scaled orthogonal matrix (scaling, rotation and possible reflection). In other words, $\mathtt{H}$ is a similarity transform. □

Result 3.11. *The plane at infinity $\boldsymbol{\pi}_\infty$ is the null-vector of $\mathtt{Q}^*_\infty$.*

This is easily verified when $\mathtt{Q}^*_\infty$ has its canonical form (3.24) in a metric frame since then, with $\boldsymbol{\pi}_\infty = (0, 0, 0, 1)^\mathsf{T}$, $\mathtt{Q}^*_\infty \boldsymbol{\pi}_\infty = \mathbf{0}$. This property holds in any frame as may be readily seen algebraically from the transformation properties of planes and dual quadrics: if $\mathbf{X}' = \mathtt{H}\mathbf{X}$, then $\mathtt{Q}^*_\infty{}' = \mathtt{H}\,\mathtt{Q}^*_\infty\,\mathtt{H}^\mathsf{T}$, $\boldsymbol{\pi}'_\infty = \mathtt{H}^{-\mathsf{T}}\boldsymbol{\pi}_\infty$, and

$$\mathtt{Q}^*_\infty{}'\boldsymbol{\pi}'_\infty = (\mathtt{H}\,\mathtt{Q}^*_\infty\,\mathtt{H}^\mathsf{T})\mathtt{H}^{-\mathsf{T}}\boldsymbol{\pi}_\infty = \mathtt{H}\mathtt{Q}^*_\infty\boldsymbol{\pi}_\infty = \mathbf{0}.$$

Result 3.12. *The angle between two planes $\boldsymbol{\pi}_1$ and $\boldsymbol{\pi}_2$ is given by*

$$\cos\theta = \frac{\boldsymbol{\pi}_1^\mathsf{T}\mathtt{Q}^*_\infty\boldsymbol{\pi}_2}{\sqrt{(\boldsymbol{\pi}_1^\mathsf{T}\mathtt{Q}^*_\infty\boldsymbol{\pi}_1)\,(\boldsymbol{\pi}_2^\mathsf{T}\mathtt{Q}^*_\infty\boldsymbol{\pi}_2)}}. \tag{3.25}$$

Proof. Consider two planes with Euclidean coordinates $\boldsymbol{\pi}_1 = (\mathbf{n}_1^\mathsf{T}, d_1)^\mathsf{T}$, $\boldsymbol{\pi}_2 = (\mathbf{n}_2^\mathsf{T}, d_2)^\mathsf{T}$. In a Euclidean frame, $\mathtt{Q}^*_\infty$ has the form (3.24), and (3.25) reduces to

$$\cos\theta = \frac{\mathbf{n}_1^\mathsf{T}\mathbf{n}_2}{\sqrt{(\mathbf{n}_1^\mathsf{T}\mathbf{n}_1)\,(\mathbf{n}_2^\mathsf{T}\mathbf{n}_2)}}$$

which is the angle between the planes expressed in terms of a scalar product of their normals.

If the planes and $\mathtt{Q}^*_\infty$ are projectively transformed, (3.25) will still determine the angle between planes due to the (covariant) transformation properties of planes and dual quadrics. □

The details of the last part of the proof are left as an exercise, but are a direct 3D analogue of the derivation of result 2.23(*p*54) on the angle between two lines in $\mathbb{P}^2$ computed using the dual of the circular points. Planes in $\mathbb{P}^3$ are the analogue of lines in $\mathbb{P}^2$, and the absolute dual quadric is the analogue of the dual of the circular points.

3.8 Closure

3.8.1 The literature

The textbooks cited in chapter 2 are also relevant here. See also [Boehm-94] for a general background from the perspective of descriptive geometry, and Hilbert and Cohn-Vossen [Hilbert-56] for many clearly explained properties of curves and surfaces.

An important representation for points, lines and planes in $\mathbb{P}^3$, which is omitted in this chapter, is the Grassmann–Cayley algebra. In this representation geometric operations such as incidence and joins are represented by a "bracket algebra" based on matrix determinants. A good introduction to this area is given by [Carlsson-94], and its application to multiple view tensors is illustrated in [Triggs-95].

Samsung/Google, EX1010, p. 103

Faugeras and Maybank [Faugeras-90] introduced Ω_∞ into the computer vision literature (in order to determine the multiplicity of solutions for relative orientation), and Triggs introduced $\mathtt{Q}^*_\infty$ in [Triggs-97] for use in auto-calibration.

3.8.2 Notes and exercises

(i) **Plücker coordinates.**

(a) Using Plücker line coordinates, $\mathcal{L}$, write an expression for the point of intersection of a line with a plane, and the plane defined by a point and a line.

(b) Now derive the condition for a point to be on a line, and a line to be on a plane.

(c) Show that parallel planes intersect in a line on π_∞. Hint, start from (3.9–$p71$) to determine the line of intersection of two parallel planes $\mathtt{L}^*$.

(d) Show that parallel lines intersect on π_∞.

(ii) **Projective transformations.** Show that a (real) projective transformation of 3-space can map an ellipsoid to a paraboloid or hyperboloid of two sheets, but cannot map an ellipsoid to a hyperboloid of one sheet (i.e. a surface with real rulings).

(iii) **Screw decomposition.** Show that the 4×4 matrix representing the Euclidean transformation $\{\mathtt{R}, \mathtt{t}\}$ (with $\mathbf{a}$ the direction of the rotation axis, i.e. $\mathtt{R}\mathbf{a} = \mathbf{a}$) has two complex conjugate eigenvalues, and two equal real eigenvalues, and the following eigenvector structure:

(a) if $\mathbf{a}$ is perpendicular to $\mathbf{t}$, then the eigenvectors corresponding to the real eigenvalues are distinct;

(b) otherwise, the eigenvectors corresponding to the real eigenvalues are coincident, and on π_∞.

(E.g. choose simple cases such as (3.20), another case is given on page 495). In the first case the two real points corresponding to the real eigenvalues define a line of fixed points. This is the screw axis for planar motion. In the second case, the direction of the screw axis is defined, but it is not a line of fixed points. What do the eigenvectors corresponding to the complex eigenvalues represent?

Samsung/Google, EX1010, p. 104

4

Estimation – 2D Projective Transformations

In this chapter, we consider the problem of estimation. In the present context this will be taken to mean the computation of some transformation or other mathematical quantity, based on measurements of some nature. This definition is somewhat vague, so to make it more concrete, here are a number of estimation problems of the type that we would like to consider.

(i) **2D homography.** Given a set of points $\mathbf{x}_i$ in $\mathbb{P}^2$ and a corresponding set of points $\mathbf{x}'_i$ likewise in $\mathbb{P}^2$, compute the projective transformation that takes each $\mathbf{x}_i$ to $\mathbf{x}'_i$. In a practical situation, the points $\mathbf{x}_i$ and $\mathbf{x}'_i$ are points in two images (or the same image), each image being considered as a projective plane $\mathbb{P}^2$.

(ii) **3D to 2D camera projection.** Given a set of points $\mathbf{X}_i$ in 3D space, and a set of corresponding points $\mathbf{x}_i$ in an image, find the 3D to 2D projective mapping that maps $\mathbf{X}_i$ to $\mathbf{x}_i$. Such a 3D to 2D projection is the mapping carried out by a projective camera, as discussed in chapter 6.

(iii) **Fundamental matrix computation.** Given a set of points $\mathbf{x}_i$ in one image, and corresponding points $\mathbf{x}'_i$ in another image, compute the fundamental matrix $\mathtt{F}$ consistent with these correspondences. The fundamental matrix, discussed in chapter 9, is a singular 3×3 matrix $\mathtt{F}$ satisfying $\mathbf{x}'^{\mathsf{T}}_i \mathtt{F} \mathbf{x}_i = 0$ for all i.

(iv) **Trifocal tensor computation.** Given a set of point correspondences $\mathbf{x}_i \leftrightarrow \mathbf{x}'_i \leftrightarrow \mathbf{x}''_i$ across three images, compute the trifocal tensor. The trifocal tensor, discussed in chapter 15, is a tensor $\mathcal{T}_i^{jk}$ relating points or lines in three views.

These problems have many features in common, and the considerations that relate to one of the problems are also relevant to each of the others. Therefore, in this chapter, the first of these problems will be considered in detail. What we learn about ways of solving this problem will teach us how to proceed in solving each of the other problems as well.

Apart from being important for illustrative purposes, the problem of estimating 2D projective transformations is of importance in its own right. We consider a set of point correspondences $\mathbf{x}_i \leftrightarrow \mathbf{x}'_i$ between two images. Our problem is to compute a 3×3 matrix $\mathtt{H}$ such that $\mathtt{H}\mathbf{x}_i = \mathbf{x}'_i$ for each i.

Samsung/Google, EX1010, p. 105

Number of measurements required. The first question to consider is how many corresponding points $\mathbf{x}_i \leftrightarrow \mathbf{x}'_i$ are required to compute the projective transformation $\mathtt{H}$. A lower bound is available by a consideration of the number of degrees of freedom and number of constraints. On the one hand, the matrix $\mathtt{H}$ contains 9 entries, but is defined only up to scale. Thus, the total number of degrees of freedom in a 2D projective transformation is 8. On the other hand, each point-to-point correspondence accounts for two constraints, since for each point $\mathbf{x}_i$ in the first image the two degrees of freedom of the point in the second image must correspond to the mapped point $\mathtt{H}\mathbf{x}_i$. A 2D point has two degrees of freedom corresponding to the x and y components, each of which may be specified separately. Alternatively, the point is specified as a homogeneous 3-vector, which also has two degrees of freedom since scale is arbitrary. As a consequence, it is necessary to specify four point correspondences in order to constrain $\mathtt{H}$ fully.

Approximate solutions. It will be seen that if exactly four correspondences are given, then an exact solution for the matrix $\mathtt{H}$ is possible. This is the *minimal* solution. Such solutions are important as they define the size of the subsets required in robust estimation algorithms, such as RANSAC, described in section 4.7. However, since points are measured inexactly ("noise"), if more than four such correspondences are given, then these correspondences may not be fully compatible with any projective transformation, and one will be faced with the task of determining the "best" transformation given the data. This will generally be done by finding the transformation $\mathtt{H}$ that minimizes some cost function. Different cost functions will be discussed during this chapter, together with methods for minimizing them. There are two main categories of cost function: those based on minimizing an algebraic error; and those based on minimizing a geometric or statistical image distance. These two categories are described in section 4.2.

The Gold Standard algorithm. There will usually be one cost function which is optimal in the sense that the $\mathtt{H}$ that minimizes it gives the best possible estimate of the transformation under certain assumptions. The computational algorithm that enables this cost function to be minimized is called the "Gold Standard" algorithm. The results of other algorithms are assessed by how well they compare to this Gold Standard. In the case of estimating a homography between two views the cost function is (4.8), the assumptions for optimality are given in section 4.3, and the Gold Standard is algorithm 4.3(p114).

4.1 The Direct Linear Transformation (DLT) algorithm

We begin with a simple linear algorithm for determining $\mathtt{H}$ given a set of four 2D to 2D point correspondences, $\mathbf{x}_i \leftrightarrow \mathbf{x}'_i$. The transformation is given by the equation $\mathbf{x}'_i = \mathtt{H}\mathbf{x}_i$. Note that this is an equation involving homogeneous vectors; thus the 3-vectors $\mathbf{x}'_i$ and $\mathtt{H}\mathbf{x}_i$ are not equal, they have the same direction but may differ in magnitude by a non-zero scale factor. The equation may be expressed in terms of the vector cross product as $\mathbf{x}'_i \times \mathtt{H}\mathbf{x}_i = \mathbf{0}$. This form will enable a simple linear solution for $\mathtt{H}$ to be derived.

Samsung/Google, EX1010, p. 106

If the j-th row of the matrix $\mathtt{H}$ is denoted by $\mathbf{h}^{j\top}$, then we may write

$$\mathtt{H}\mathbf{x}_i = \begin{pmatrix} \mathbf{h}^{1\top}\mathbf{x}_i \\ \mathbf{h}^{2\top}\mathbf{x}_i \\ \mathbf{h}^{3\top}\mathbf{x}_i \end{pmatrix}.$$

Writing $\mathbf{x}'_i = (x'_i, y'_i, w'_i)^\top$, the cross product may then be given explicitly as

$$\mathbf{x}'_i \times \mathtt{H}\mathbf{x}_i = \begin{pmatrix} y'_i\mathbf{h}^{3\top}\mathbf{x}_i - w'_i\mathbf{h}^{2\top}\mathbf{x}_i \\ w'_i\mathbf{h}^{1\top}\mathbf{x}_i - x'_i\mathbf{h}^{3\top}\mathbf{x}_i \\ x'_i\mathbf{h}^{2\top}\mathbf{x}_i - y'_i\mathbf{h}^{1\top}\mathbf{x}_i \end{pmatrix}.$$

Since $\mathbf{h}^{j\top}\mathbf{x}_i = \mathbf{x}_i^\top\mathbf{h}^j$ for $j = 1, \ldots, 3$, this gives a set of three equations in the entries of $\mathtt{H}$, which may be written in the form

$$\begin{bmatrix} \mathbf{0}^\top & -w'_i\mathbf{x}_i^\top & y'_i\mathbf{x}_i^\top \\ w'_i\mathbf{x}_i^\top & \mathbf{0}^\top & -x'_i\mathbf{x}_i^\top \\ -y'_i\mathbf{x}_i^\top & x'_i\mathbf{x}_i^\top & \mathbf{0}^\top \end{bmatrix} \begin{pmatrix} \mathbf{h}^1 \\ \mathbf{h}^2 \\ \mathbf{h}^3 \end{pmatrix} = \mathbf{0}. \tag{4.1}$$

These equations have the form $\mathtt{A}_i\mathbf{h} = \mathbf{0}$, where $\mathtt{A}_i$ is a 3×9 matrix, and $\mathbf{h}$ is a 9-vector made up of the entries of the matrix $\mathtt{H}$,

$$\mathbf{h} = \begin{pmatrix} \mathbf{h}^1 \\ \mathbf{h}^2 \\ \mathbf{h}^3 \end{pmatrix}, \qquad \mathtt{H} = \begin{bmatrix} h_1 & h_2 & h_3 \\ h_4 & h_5 & h_6 \\ h_7 & h_8 & h_9 \end{bmatrix} \tag{4.2}$$

with h_i the $i-$th element of $\mathbf{h}$. Three remarks regarding these equations are in order here.

(i) The equation $\mathtt{A}_i\mathbf{h} = \mathbf{0}$ is an equation *linear* in the unknown $\mathbf{h}$. The matrix elements of $\mathtt{A}_i$ are quadratic in the known coordinates of the points.

(ii) Although there are three equations in (4.1), only two of them are linearly independent (since the third row is obtained, up to scale, from the sum of x'_i times the first row and y'_i times the second). Thus each point correspondence gives two equations in the entries of $\mathtt{H}$. It is usual to omit the third equation in solving for $\mathtt{H}$ ([Sutherland-63]). Then (for future reference) the set of equations becomes

$$\begin{bmatrix} \mathbf{0}^\top & -w'_i\mathbf{x}_i^\top & y'_i\mathbf{x}_i^\top \\ w'_i\mathbf{x}_i^\top & \mathbf{0}^\top & -x'_i\mathbf{x}_i^\top \end{bmatrix} \begin{pmatrix} \mathbf{h}^1 \\ \mathbf{h}^2 \\ \mathbf{h}^3 \end{pmatrix} = \mathbf{0}. \tag{4.3}$$

This will be written

$$\mathtt{A}_i\mathbf{h} = \mathbf{0}$$

where $\mathtt{A}_i$ is now the 2×9 matrix of (4.3).

(iii) The equations hold for any homogeneous coordinate representation $(x'_i, y'_i, w'_i)^\top$ of the point $\mathbf{x}'_i$. One may choose $w'_i = 1$, which means that (x'_i, y'_i) are the coordinates measured in the image. Other choices are possible, however, as will be seen later.

Samsung/Google, EX1010, p. 107

Solving for H

Each point correspondence gives rise to two independent equations in the entries of H. Given a set of four such point correspondences, we obtain a set of equations $\mathtt{A}\mathbf{h} = \mathbf{0}$, where A is the matrix of equation coefficients built from the matrix rows $\mathtt{A}_i$ contributed from each correspondence, and $\mathbf{h}$ is the vector of unknown entries of H. We seek a non-zero solution $\mathbf{h}$, since the obvious solution $\mathbf{h} = \mathbf{0}$ is of no interest to us. If (4.1) is used then A has dimension 12×9, and if (4.3) the dimension is 8×9. In either case A has rank 8, and thus has a 1-dimensional null-space which provides a solution for $\mathbf{h}$. Such a solution $\mathbf{h}$ can only be determined up to a non-zero scale factor. However, H is in general only determined up to scale, so the solution $\mathbf{h}$ gives the required H. A scale may be arbitrarily chosen for $\mathbf{h}$ by a requirement on its norm such as $\|\mathbf{h}\| = 1$.

4.1.1 Over-determined solution

If more than four point correspondences $\mathbf{x}_i \leftrightarrow \mathbf{x}'_i$ are given, then the set of equations $\mathtt{A}\mathbf{h} = \mathbf{0}$ derived from (4.3) is over-determined. If the position of the points is exact then the matrix A will still have rank 8, a one dimensional null-space, and there is an exact solution for $\mathbf{h}$. This will not be the case if the measurement of image coordinates is inexact (generally termed *noise*) – there will not be an exact solution to the over-determined system $\mathtt{A}\mathbf{h} = \mathbf{0}$ apart from the zero solution. Instead of demanding an exact solution, one attempts to find an approximate solution, namely a vector $\mathbf{h}$ that minimizes a suitable cost function. The question that naturally arises then is: what should be minimized? Clearly, to avoid the solution $\mathbf{h} = \mathbf{0}$ an additional constraint is required. Generally, a condition on the norm is used, such as $\|\mathbf{h}\| = 1$. The value of the norm is unimportant since H is only defined up to scale. Given that there is no exact solution to $\mathtt{A}\mathbf{h} = \mathbf{0}$, it seems natural to attempt to minimize the norm $\|\mathtt{A}\mathbf{h}\|$ instead, subject to the usual constraint, $\|\mathbf{h}\| = 1$. This is identical to the problem of finding the minimum of the quotient $\|\mathtt{A}\mathbf{h}\|/\|\mathbf{h}\|$. As shown in section A5.3(*p*592) the solution is the (unit) eigenvector of $\mathtt{A}^\mathsf{T}\mathtt{A}$ with least eigenvalue. Equivalently, the solution is the unit singular vector corresponding to the smallest singular value of A. The resulting algorithm, known as the basic DLT algorithm, is summarized in algorithm 4.1.

4.1.2 Inhomogeneous solution

An alternative to solving for $\mathbf{h}$ directly as a homogeneous vector is to turn the set of equations (4.3) into a inhomogeneous set of linear equations by imposing a condition $h_j = 1$ for some entry of the vector $\mathbf{h}$. Imposing the condition $h_j = 1$ is justified by the observation that the solution is determined only up to scale, and this scale can be chosen such that $h_j = 1$. For example, if the last element of $\mathbf{h}$, which corresponds to $\mathtt{H}_{33}$, is chosen as unity then the resulting equations derived from (4.3) are

$$\begin{bmatrix} 0 & 0 & 0 & -x_i w'_i & -y_i w'_i & -w_i w'_i & x_i y'_i & y_i y'_i \\ x_i w'_i & y_i w'_i & w_i w'_i & 0 & 0 & 0 & -x_i x'_i & -y_i x'_i \end{bmatrix} \tilde{\mathbf{h}} = \begin{pmatrix} -w_i y'_i \\ w_i x'_i \end{pmatrix}$$

where $\tilde{\mathbf{h}}$ is an 8-vector consisting of the first 8 components of $\mathbf{h}$. Concatenating the equations from four correspondences then generates a matrix equation of the form

Samsung/Google, EX1010, p. 108

Objective

Given $n \geq 4$ 2D to 2D point correspondences $\{\mathbf{x}_i \leftrightarrow \mathbf{x}'_i\}$, determine the 2D homography matrix $\mathtt{H}$ such that $\mathbf{x}'_i = \mathtt{H}\mathbf{x}_i$.

Algorithm

(i) For each correspondence $\mathbf{x}_i \leftrightarrow \mathbf{x}'_i$ compute the matrix $\mathtt{A}_i$ from (4.1). Only the first two rows need be used in general.

(ii) Assemble the n 2×9 matrices $\mathtt{A}_i$ into a single $2n \times 9$ matrix $\mathtt{A}$.

(iii) Obtain the SVD of $\mathtt{A}$ (section A4.4($p585$)). The unit singular vector corresponding to the smallest singular value is the solution $\mathbf{h}$. Specifically, if $\mathtt{A} = \mathtt{U}\mathtt{D}\mathtt{V}^\top$ with $\mathtt{D}$ diagonal with positive diagonal entries, arranged in descending order down the diagonal, then $\mathbf{h}$ is the last column of $\mathtt{V}$.

(iv) The matrix $\mathtt{H}$ is determined from $\mathbf{h}$ as in (4.2).

Algorithm 4.1. *The basic DLT for* $\mathtt{H}$ *(but see algorithm 4.2(*p*109) which includes normalization).*

$\mathtt{M}\tilde{\mathbf{h}} = \mathbf{b}$, where $\mathtt{M}$ has 8 columns and $\mathbf{b}$ is an 8-vector. Such an equation may be solved for $\tilde{\mathbf{h}}$ using standard techniques for solving linear equations (such as Gaussian elimination) in the case where $\mathtt{M}$ contains just 8 rows (the minimum case), or by least-squares techniques (section A5.1($p588$)) in the case of an over-determined set of equations.

However, if in fact $h_j = 0$ is the true solution, then no multiplicative scale k can exist such that $kh_j = 1$. This means that the true solution cannot be reached. For this reason, this method can be expected to lead to unstable results in the case where the chosen h_j is close to zero. Consequently, this method is *not* recommended in general.

Example 4.1. It will be shown that $h_9 = \mathtt{H}_{33}$ is zero if the coordinate origin is mapped to a point at infinity by $\mathtt{H}$. Since $(0,0,1)^\top$ represents the coordinate origin $\mathbf{x}_0$, and also $(0,0,1)^\top$ represents the line at infinity $\mathbf{l}$, this condition may be written as $\mathbf{l}^\top\mathtt{H}\mathbf{x}_0 = (0,0,1)\mathtt{H}(0,0,1)^\top = 0$, thus $\mathtt{H}_{33} = 0$. In a perspective image of a scene plane the line at infinity is imaged as the vanishing line of the plane (see chapter 8), for example the horizon is the vanishing line of the ground plane. It is not uncommon for the horizon to pass through the image centre, and for the coordinate origin to coincide with the image centre. In this case the mapping that takes the image to the world plane maps the origin to the line at infinity, so that the true solution has $\mathtt{H}_{33} = h_9 = 0$. Consequently, an $h_9 = 1$ normalization can be a serious failing in practical situations. △

4.1.3 Degenerate configurations

Consider a minimal solution in which a homography is computed using four point correspondences, and suppose that three of the points $\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3$ are collinear. The question is whether this is significant. If the corresponding points $\mathbf{x}'_1, \mathbf{x}'_2, \mathbf{x}'_3$ are also collinear then one might suspect that the homography is not sufficiently constrained, and there will exist a family of homographies mapping $\mathbf{x}_i$ to $\mathbf{x}'_i$. On the other hand, if the corresponding points $\mathbf{x}'_1, \mathbf{x}'_2, \mathbf{x}'_3$ are not collinear then clearly there can be no transformation $\mathtt{H}$ taking $\mathbf{x}_i$ to $\mathbf{x}'_i$, since a projective transformation must preserve collinearity. Never-

Samsung/Google, EX1010, p. 109

theless the set of eight homogeneous equations derived from (4.3) must have a non-zero solution, giving rise to a matrix H. How is this apparent contradiction to be resolved?

The equations (4.3) express the condition that $\mathbf{x}'_i \times \mathtt{H}\mathbf{x}_i = \mathbf{0}$ for $i = 1, \ldots, 4$, and so the matrix H found by solving the system of 8 equations will satisfy this condition. Suppose that $\mathbf{x}_1, \ldots, \mathbf{x}_3$ are collinear and let $\mathbf{l}$ be the line that they lie on, so that $\mathbf{l}^\mathsf{T}\mathbf{x}_i = 0$ for $i = 1, \ldots, 3$. Now define $\mathtt{H}^* = \mathbf{x}'_4\mathbf{l}^\mathsf{T}$, which is a 3×3 matrix of rank 1. In this case, one verifies that $\mathtt{H}^*\mathbf{x}_i = \mathbf{x}'_4(\mathbf{l}^\mathsf{T}\mathbf{x}_i) = \mathbf{0}$ for $i = 1, \ldots, 3$, since $\mathbf{l}^\mathsf{T}\mathbf{x}_i = 0$. On the other hand, $\mathtt{H}^*\mathbf{x}_4 = \mathbf{x}'_4(\mathbf{l}^\mathsf{T}\mathbf{x}_4) = k\mathbf{x}'_4$. Therefore the condition $\mathbf{x}'_i \times \mathtt{H}^*\mathbf{x}_i = \mathbf{0}$ is satisfied for all i. Note that the vector $\mathbf{h}^*$ corresponding to $\mathtt{H}^*$ is given by $\mathbf{h}^{*\mathsf{T}} = (x_4\mathbf{l}^\mathsf{T}, y_4\mathbf{l}^\mathsf{T}, w_4\mathbf{l}^\mathsf{T})$, and one easily verifies that this vector satisfies (4.3) for all i. The problem with this solution for $\mathtt{H}^*$ is that $\mathtt{H}^*$ is a rank 1 matrix and hence does not represent a projective transformation. As a consequence the points $\mathtt{H}^*\mathbf{x}_i = \mathbf{0}$ for $i = 1, \ldots, 3$ are not well defined.

We showed that if $\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3$ are collinear then $\mathtt{H}^* = \mathbf{x}'_4\mathbf{l}^\mathsf{T}$ is a solution to (4.1). There are two cases: either $\mathtt{H}^*$ is the unique solution (up to scale) or there is a further solution H. In the first case, since $\mathtt{H}^*$ is a singular matrix, there exists no transformation taking each $\mathbf{x}_i$ to $\mathbf{x}'_i$. This occurs when $\mathbf{x}_1, \ldots, \mathbf{x}_3$ are collinear but $\mathbf{x}'_1, \ldots, \mathbf{x}'_3$ are not. In the second case, where a further solution H exists, then any matrix of the form $\alpha\,\mathtt{H}^* + \beta\,\mathtt{H}$ is a solution. Thus a 2-dimensional family of transformations exist, and it follows that the 8 equations derived from (4.3) are not independent.

A situation where a configuration does not determine a unique solution for a particular class of transformation is termed *degenerate*. Note that the definition of degeneracy involves both the configuration and the type of transformation. The degeneracy problem is not limited to a minimal solution, however. If additional (perfect, i.e. error-free) correspondences are supplied which are also collinear (lie on l), then the degeneracy is not resolved.

4.1.4 Solutions from lines and other entities

The development to this point, and for the rest of the chapter, is exclusively in terms of computing homographies from point correspondences. However, an identical development can be given for computing homographies from line correspondences. Starting from the line transformation $\mathbf{l}_i = \mathtt{H}^{-\mathsf{T}}\mathbf{l}'_i$, a matrix equation of the form $\mathtt{A}\mathbf{h} = \mathbf{0}$ can be derived, with a minimal solution requiring four lines in general position. Similarly, a homography may be computed from conic correspondences and so forth.

There is the question then of how many correspondences are required to compute the homography (or any other relation). The general rule is that the number of constraints must equal or exceed the number of degrees of freedom of the transformation. For example, in 2D each corresponding point or line generates two constraints on H, in 3D each corresponding point or plane generates three constraints. Thus in 2D the correspondence of four points or four lines is sufficient to compute H, since $4 \times 2 = 8$, with 8 the number of degrees of freedom of the homography. In 3D a homography has 15 degrees of freedom, and five points or five planes are required. For a planar affine transformation (6 dof) only three corresponding points or lines are required, and so on. A conic provides five constraints on a 2D homography.

Samsung/Google, EX1010, p. 110

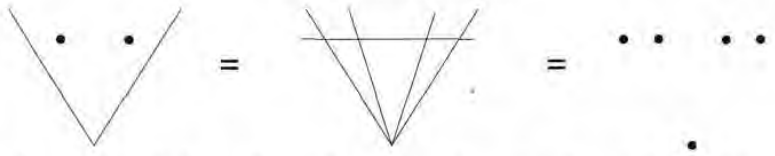


Fig. 4.1. **Geometric equivalence of point–line configurations.** *A configuration of two points and two lines is equivalent to five lines with four concurrent, or five points with four collinear.*

Care has to be taken when computing H from correspondences of mixed type. For example, a 2D homography cannot be determined uniquely from the correspondences of two points and two lines, but can from three points and one line or one point and three lines, even though in each case the configuration has 8 degrees of freedom. The case of three lines and one point is geometrically equivalent to four points, since the three lines define a triangle and the vertices of the triangle uniquely define three points. We have seen that the correspondence of four points in general position uniquely determines a homography, which means that the correspondence of three lines and one point also uniquely determines a homography. Similarly the case of three points and a line is equivalent to four lines, and again the correspondence of four lines in general position (i.e. no three concurrent) uniquely determines a homography. However, as a quick sketch shows (figure 4.1), the case of two points and two lines is equivalent to five lines with four concurrent, or five points with four collinear. As shown in the previous section, this configuration is degenerate and a one-parameter family of homographies map the two-point and two-line configuration to the corresponding configuration.

4.2 Different cost functions

We will now describe a number of cost functions which may be minimized in order to determine H for over-determined solutions. Methods of minimizing these functions are described later in the chapter.

4.2.1 Algebraic distance

The DLT algorithm minimizes the norm $\|\mathtt{A}\mathbf{h}\|$. The vector $\boldsymbol{\epsilon} = \mathtt{A}\mathbf{h}$ is called the residual vector and it is the norm of this error vector that is minimized. The components of this vector arise from the individual correspondences that generate each row of the matrix A. Each correspondence $\mathbf{x}_i \leftrightarrow \mathbf{x}'_i$ contributes a partial error vector $\boldsymbol{\epsilon}_i$ from (4.1) or (4.3) towards the full error vector $\boldsymbol{\epsilon}$. This vector $\boldsymbol{\epsilon}_i$ is the *algebraic error vector* associated with the point correspondence $\mathbf{x}_i \leftrightarrow \mathbf{x}'_i$ and the homography H. The norm of this vector is a scalar which is called the *algebraic distance*:

$$d_{\text{alg}}(\mathbf{x}'_i, \mathtt{H}\mathbf{x}_i)^2 = \|\boldsymbol{\epsilon}_i\|^2 = \left\| \begin{bmatrix} \mathbf{0}^\top & -w'_i\mathbf{x}_i^\top & y'_i\mathbf{x}_i^\top \\ w'_i\mathbf{x}_i^\top & \mathbf{0}^\top & -x'_i\mathbf{x}_i^\top \end{bmatrix} \mathbf{h} \right\|^2. \tag{4.4}$$

More generally, and briefly, for any two vectors $\mathbf{x}_1$ and $\mathbf{x}_2$ we may write

$$d_{\text{alg}}(\mathbf{x}_1, \mathbf{x}_2)^2 = a_1^2 + a_2^2 \text{ where } \mathbf{a} = (a_1, a_2, a_3)^\top = \mathbf{x}_1 \times \mathbf{x}_2.$$

Samsung/Google, EX1010, p. 111

The relation of this distance to a geometric distance is described in section 4.2.4.

Given a set of correspondences, the quantity $\epsilon = \mathbf{A}\mathbf{h}$ is the algebraic error vector for the complete set, and one sees that

$$\sum_i d_{\text{alg}}(\mathbf{x}'_i, \mathbf{H}\mathbf{x}_i)^2 = \sum_i \|\epsilon_i\|^2 = \|\mathbf{A}\mathbf{h}\|^2 = \|\epsilon\|^2. \tag{4.5}$$

The concept of algebraic distance originated in the conic-fitting work of Bookstein [Bookstein-79]. Its disadvantage is that the quantity that is minimized is not geometrically or statistically meaningful. As Bookstein demonstrated, the solutions that minimize algebraic distance may not be those expected intuitively. Nevertheless, with a good choice of normalization (as will be discussed in section 4.4) methods which minimize algebraic distance do give very good results. Their particular advantages are a linear (and thus a unique) solution, and computational cheapness. Often solutions based on algebraic distance are used as a starting point for a non-linear minimization of a geometric or statistical cost function. The non-linear minimization gives the solution a final "polish".

4.2.2 Geometric distance

Next we discuss alternative error functions based on the measurement of geometric distance in the image, and minimization of the difference between the measured and estimated image coordinates.

Notation. Vectors $\mathbf{x}$ represent the *measured* image coordinates; $\hat{\mathbf{x}}$ represent estimated values of the points and $\bar{\mathbf{x}}$ represent true values of the points.

Error in one image. We start by considering error only in the second image, with points in the first measured perfectly. Clearly, this will not be true in most practical situations with images. An example where the assumption is more reasonable is in estimating the projective transformation between a calibration pattern or a world plane, where points are measured to a very high accuracy, and its image. The appropriate quantity to be minimized is the *transfer* error. This is the Euclidean image distance in the second image between the measured point $\mathbf{x}'$ and the point $\mathbf{H}\bar{\mathbf{x}}$ at which the corresponding point $\bar{\mathbf{x}}$ is mapped from the first image. We use the notation $d(\mathbf{x}, \mathbf{y})$ to represent the Euclidean distance between the inhomogeneous points represented by $\mathbf{x}$ and $\mathbf{y}$. Then the transfer error for the set of correspondences is

$$\sum_i d(\mathbf{x}'_i, \mathbf{H}\bar{\mathbf{x}}_i)^2. \tag{4.6}$$

The estimated homography $\hat{\mathbf{H}}$ is the one for which the error (4.6) is minimized.

Symmetric transfer error. In the more realistic case where image measurement errors occur in both the images, it is preferable that errors be minimized in both images, and not solely in the one. One way of constructing a more satisfactory error function is to

Samsung/Google, EX1010, p. 112

consider the forward ($\mathtt{H}$) and backward ($\mathtt{H}^{-1}$) transformation, and sum the geometric errors corresponding to each of these two transformations. Thus, the error is

$$\sum_i d(\mathbf{x}_i, \mathtt{H}^{-1}\mathbf{x}'_i)^2 + d(\mathbf{x}'_i, \mathtt{H}\mathbf{x}_i)^2. \tag{4.7}$$

The first term in this sum is the transfer error in the first image, and the second term is the transfer error in the second image. Again the estimated homography $\hat{\mathtt{H}}$ is the one for which (4.7) is minimized.

4.2.3 Reprojection error – both images

An alternative method of quantifying error in each of the two images involves estimating a "correction" for each correspondence. One asks how much it is necessary to correct the measurements in each of the two images in order to obtain a perfectly matched set of image points. One should compare this with the geometric one-image transfer error (4.6) which measures the correction that it is necessary to make to the measurements in one image (the second image) in order to get a set of perfectly matching points.

In the present case, we are seeking a homography $\hat{\mathtt{H}}$ and pairs of *perfectly* matched points $\hat{\mathbf{x}}_i$ and $\hat{\mathbf{x}}'_i$ that minimize the total error function

$$\sum_i d(\mathbf{x}_i, \hat{\mathbf{x}}_i)^2 + d(\mathbf{x}'_i, \hat{\mathbf{x}}'_i)^2 \quad \text{subject to } \hat{\mathbf{x}}'_i = \hat{\mathtt{H}}\hat{\mathbf{x}}_i \ \ \forall i. \tag{4.8}$$

Minimizing this cost function involves determining both $\hat{\mathtt{H}}$ *and* a set of subsidiary correspondences $\{\hat{\mathbf{x}}_i\}$ and $\{\hat{\mathbf{x}}'_i\}$. This estimation models, for example, the situation that measured correspondences $\mathbf{x}_i \leftrightarrow \mathbf{x}'_i$ arise from images of points on a world plane. We wish to estimate a point on the world plane $\widehat{\mathbf{X}}_i$ from $\mathbf{x}_i \leftrightarrow \mathbf{x}'_i$ which is then *reprojected* to the estimated perfectly matched correspondence $\hat{\mathbf{x}}_i \leftrightarrow \hat{\mathbf{x}}'_i$.

This reprojection error function is compared with the symmetric error function in figure 4.2. It will be seen in section 4.3 that (4.8) is related to the Maximum Likelihood estimation of the homography and correspondences.

4.2.4 Comparison of geometric and algebraic distance

We return to the case of errors only in the second image. Let $\mathbf{x}'_i = (x'_i, y'_i, w'_i)^\mathsf{T}$ and define a vector $(\bar{x}'_i, \bar{y}'_i, \bar{w}'_i)^\mathsf{T} = \hat{\mathbf{x}}'_i = \mathtt{H}\bar{\mathbf{x}}_i$. Using this notation, the left hand side of (4.3) becomes

$$\mathtt{A}_i\mathbf{h} = \boldsymbol{\epsilon}_i = \begin{pmatrix} y'_i\bar{w}'_i - w'_i\bar{y}'_i \\ w'_i\bar{x}'_i - x'_i\bar{w}'_i \end{pmatrix}.$$

This vector is the *algebraic error vector* associated with the point correspondence $\mathbf{x}_i \leftrightarrow \mathbf{x}'_i$ and the camera mapping $\mathtt{H}$. Thus,

$$d_{\mathrm{alg}}(\mathbf{x}'_i, \hat{\mathbf{x}}'_i)^2 = (y'_i\bar{w}'_i - w'_i\bar{y}'_i)^2 + (w'_i\bar{x}'_i - x'_i\bar{w}'_i)^2.$$

For points $\mathbf{x}'_i$ and $\hat{\mathbf{x}}'_i$ the geometric distance is

$$d(\mathbf{x}'_i, \hat{\mathbf{x}}'_i) \ = \ \left((x'_i/w'_i - \hat{x}'_i/\hat{w}'_i)^2 + (y'_i/w'_i - \hat{y}'_i/\hat{w}'_i)^2\right)^{1/2}$$

Samsung/Google, EX1010, p. 113

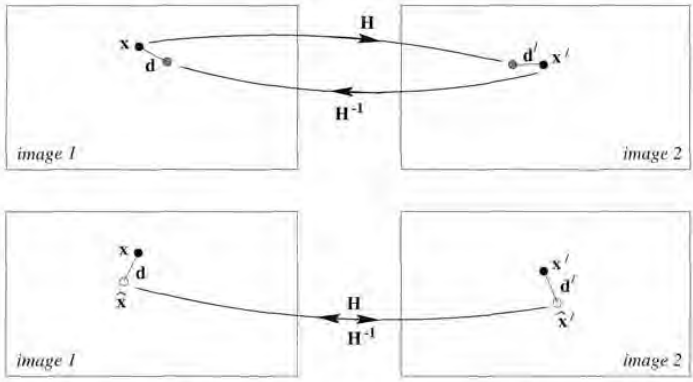


Fig. 4.2. *A comparison between symmetric transfer error (upper) and reprojection error (lower) when estimating a homography. The points* $\mathbf{x}$ *and* $\mathbf{x}'$ *are the measured (noisy) points. Under the estimated homography the points* $\hat{\mathbf{x}}'$ *and* $\mathtt{H}\mathbf{x}$ *do* not *correspond perfectly (and neither do the points* $\mathbf{x}$ *and* $\mathtt{H}^{-1}\mathbf{x}'$*). However, the estimated points,* $\hat{\mathbf{x}}$ *and* $\hat{\mathbf{x}}'$*, do correspond perfectly by the homography* $\hat{\mathbf{x}}' = \mathtt{H}\hat{\mathbf{x}}$*. Using the notation* $d(\mathbf{x}, \mathbf{y})$ *for the Euclidean image distance between* $\mathbf{x}$ *and* $\mathbf{y}$*, the symmetric transfer error is* $d(\mathbf{x}, \mathtt{H}^{-1}\mathbf{x}')^2 + d(\mathbf{x}', \mathtt{H}\mathbf{x})^2$*; the reprojection error is* $d(\mathbf{x}, \hat{\mathbf{x}})^2 + d(\mathbf{x}', \hat{\mathbf{x}}')^2$.

$$= \quad d_{\text{alg}}(\mathbf{x}'_i, \hat{\mathbf{x}}'_i)/\hat{w}'_i w'_i.$$

Thus, geometric distance is related to, but not quite the same as, algebraic distance. Note, though, that if $\hat{w}'_i = w'_i = 1$, then the two distances are identical.

One can always assume that $w_i = 1$, thus expressing the points $\mathbf{x}_i$ in the usual form $\mathbf{x}_i = (x_i, y_i, 1)^\mathsf{T}$. For one important class of 2D homographies, the values of $\hat{w}'_i$ will always be 1 as well. A 2D *affine* transformation is represented by a matrix of the form (2.10–*p*39)

$$\mathtt{H}_\mathrm{A} = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ 0 & 0 & 1 \end{bmatrix}. \tag{4.9}$$

One verifies immediately from $\hat{\mathbf{x}}'_i = \mathtt{H}_\mathrm{A}\hat{\mathbf{x}}_i$ that $\hat{w}'_i = 1$ if $w_i = 1$. This demonstrates that in the case of an affine transformation geometric distance and algebraic distance are identical. The DLT algorithm is easily adapted to enforce the condition that the last row of $\mathtt{H}$ has the form $(0, 0, 1)$ by setting $h_7 = h_8 = 0$. Hence, for affine transformations, geometric distance can be minimized by the linear DLT algorithm based on algebraic distance.

4.2.5 Geometric interpretation of reprojection error

The estimation of a homography between two planes can be thought of as fitting a "surface" to points in a 4D space, $\mathbb{R}^4$. Each pair of image points $\mathbf{x}, \mathbf{x}'$ defines a single point denoted $\mathbf{X}$ in a measurement space $\mathbb{R}^4$, formed by concatenating the inhomogeneous coordinates of $\mathbf{x}$ and $\mathbf{x}'$. For a given specific homography $\mathtt{H}$, the image correspondences $\mathbf{x} \leftrightarrow \mathbf{x}'$ that satisfy $\mathbf{x}' \times (\mathtt{H}\mathbf{x}) = \mathbf{0}$ define an algebraic variety[1] $\mathcal{V}_\mathtt{H}$ in $\mathbb{R}^4$ which is the

[1] A *variety* is the simultaneous zero-set of one or more multivariate polynomials defined in $\mathbb{R}^N$.

Samsung/Google, EX1010, p. 114

intersection of two quadric hypersurfaces. The surface is a quadric in $\mathbb{R}^4$ because each row of (4.1) is a degree 2 polynomial in x, y, x', y'. The elements of $\mathtt{H}$ determine the coefficient of each term of the polynomial, and so $\mathtt{H}$ specifies the particular quadric. The two independent equations of (4.1) define two such quadrics.

Given points $\mathbf{X}_i = (x_i, y_i, x_i', y_i')^\mathsf{T}$ in $\mathbb{R}^4$, the task of estimating a homography becomes the task of finding a variety $\mathcal{V}_\mathtt{H}$ that passes (or most nearly passes) through the points $\mathbf{X}_i$. In general, of course, it will not be possible to fit a variety precisely. In this case, let $\mathcal{V}_\mathtt{H}$ be some variety corresponding to a transformation $\mathtt{H}$, and for each point $\mathbf{X}_i$, let $\widehat{\mathbf{X}}_i = (\hat{x}_i, \hat{y}_i, \hat{x}_i', \hat{y}_i')^\mathsf{T}$ be the closest point to $\mathbf{X}_i$ lying on the variety $\mathcal{V}_\mathtt{H}$. One sees immediately that

$$\begin{aligned} \|\mathbf{X}_i - \widehat{\mathbf{X}}_i\|^2 &= (x_i - \hat{x}_i)^2 + (y_i - \hat{y}_i)^2 + (x_i' - \hat{x}_i')^2 + (y_i' - \hat{y}_i')^2 \\ &= d(\mathbf{x}_i, \hat{\mathbf{x}}_i)^2 + d(\mathbf{x}_i', \hat{\mathbf{x}}_i')^2. \end{aligned}$$

Thus geometric distance in $\mathbb{R}^4$ is equivalent to the reprojection error measured in both the images, and finding the variety $\mathcal{V}_\mathtt{H}$ and points $\widehat{\mathbf{X}}_i$ on $\mathcal{V}_\mathtt{H}$ that minimize the squared sum of distances to the measured points $\mathbf{X}_i$ is equivalent to finding the homography $\hat{\mathtt{H}}$ and the estimated points $\hat{\mathbf{x}}_i$ and $\hat{\mathbf{x}}_i'$ that minimize the reprojection error function (4.8).

The point $\widehat{\mathbf{X}}$ on $\mathcal{V}_\mathtt{H}$ that lies closest to a measured point $\mathbf{X}$ is a point where the line between $\mathbf{X}$ and $\widehat{\mathbf{X}}$ is perpendicular to the tangent plane to $\mathcal{V}_\mathtt{H}$ at $\widehat{\mathbf{X}}$. Thus

$$d(\mathbf{x}_i, \hat{\mathbf{x}}_i)^2 + d(\mathbf{x}_i', \hat{\mathbf{x}}_i')^2 = d_\perp(\mathbf{X}_i, \mathcal{V}_\mathtt{H})^2$$

where $d_\perp(\mathbf{X}, \mathcal{V}_\mathtt{H})$ is the perpendicular distance of the point $\mathbf{X}$ to the variety $\mathcal{V}_\mathtt{H}$. As may be seen from the conic-fitting analogue discussed below, there may be more than one such perpendicular from $\mathbf{X}$ to $\mathcal{V}_\mathtt{H}$.

The distance $d_\perp(\mathbf{X}, \mathcal{V}_\mathtt{H})$ is invariant to rigid transformations of $\mathbb{R}^4$, and this includes as a special case rigid transformations of the coordinates (x, y), (x', y') of each image individually. This point is returned to in section 4.4.3.

Conic analogue. Before proceeding further we will first sketch an analogous estimation problem that can be visualized more easily. The problem is fitting a conic to 2D points, which occupies a useful intermediate position between fitting a straight line (no curvature, too simple) and fitting a homography (four dimensions, with non-zero curvature).

Consider the problem of fitting a conic to a set of $n > 5$ points $(x_i, y_i)^\mathsf{T}$ on the plane such that an error based on geometric distance is minimized. The points may be thought of as "correspondences" $x_i \leftrightarrow y_i$. The transfer distance and reprojection (perpendicular) distance are illustrated in figure 4.3. It is clear from this figure that $d_\perp$ is less than or equal to the transfer error.

The algebraic distance of a point $\mathbf{x}$ from a conic $\mathtt{C}$ is defined as $d_{\text{alg}}(\mathbf{x}, \mathtt{C})^2 = \mathbf{x}^\mathsf{T}\mathtt{C}\mathbf{x}$. A linear solution for $\mathtt{C}$ can be obtained by minimizing $\sum_i d_{\text{alg}}(\mathbf{x}_i, \mathtt{C})^2$ with a suitable normalization on $\mathtt{C}$. There is no linear expression for the perpendicular distance of a point (x, y) to a conic $\mathtt{C}$, since through each point in $\mathbb{R}^2$ there are up to 4 lines perpendicular to $\mathtt{C}$. The solution can be obtained from the roots of a quartic. However, a function $d_\perp(\mathbf{x}, \mathtt{C})$ may be defined which returns the shortest distance between a conic

Samsung/Google, EX1010, p. 115

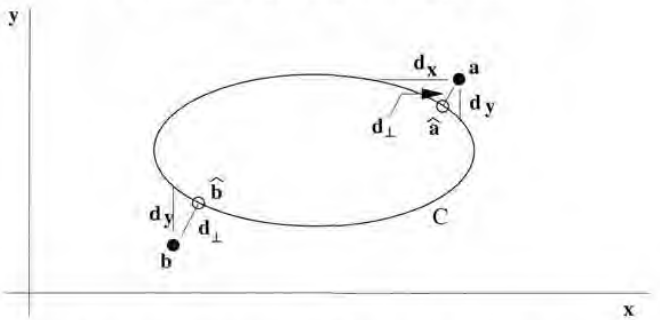


Fig. 4.3. *A conic may be estimated from a set of 2D points by minimizing "symmetric transfer error" $d_x^2 + d_y^2$ or the sum of squared perpendicular distances $d_\perp^2$. The analogue of transfer error is to consider x as perfect and measure the distance d_y to the conic in the y direction, and similarly for d_x. For point* a *it is clear that $d_\perp \leq d_x$ and $d_\perp \leq d_y$. Also $d_\perp$ is more stable than d_x or d_y as illustrated by point* b *where d_x cannot be defined.*

and a point. A conic can then be estimated by minimizing $\sum_i d_\perp(\mathbf{x}_i, \mathtt{C})^2$ over the five parameters of $\mathtt{C}$, though this cannot be achieved by a linear solution. Given a conic $\mathtt{C}$ and a measured point $\mathbf{x}$, a corrected point $\hat{\mathbf{x}}$ is obtained simply by choosing the closest point on $\mathtt{C}$.

We return now to estimating a homography. In the case of an affine transformation the variety is the intersection of two hyperplanes, i.e. it is a linear subspace of dimension 2. This follows from the form (4.9) of the affine matrix which for $\mathbf{x}' = \mathtt{H}_\mathrm{A}\mathbf{x}$ yields one linear constraint between x, x', y and another between x, y, y', each of which defines a hyperplane in $\mathbb{R}^4$. An analogue of this situation is line fitting to points on the plane. In both cases the relation (affine transformation or line) may be estimated by minimizing the perpendicular distance of points to the variety. In both cases there is a closed form solution as discussed in the following section.

4.2.6 Sampson error

The geometric error (4.8) is quite complex in nature, and minimizing it requires the simultaneous estimation of both the homography matrix and the points $\hat{\mathbf{x}}_i, \hat{\mathbf{x}}'_i$. This non-linear estimation problem will be discussed further in section 4.5. Its complexity contrasts with the simplicity of minimizing the algebraic error (4.4). The geometric interpretation of geometric error given in section 4.2.5 leads to a further cost function that lies between the algebraic and geometric cost functions in terms of complexity, but gives a close approximation to geometric error. We will refer to this cost function as *Sampson error* since Sampson [Sampson-82] used this approximation for conic fitting.

As described in section 4.2.5, the vector $\widehat{\mathbf{x}}$ that minimizes the geometric error $\|\mathbf{x} - \widehat{\mathbf{x}}\|^2$ is the closest point on the variety $\mathcal{V}_\mathtt{H}$ to the measurement $\mathbf{x}$. This point can not be estimated directly except via iteration, because of the non-linear nature of the variety $\mathcal{V}_\mathtt{H}$. The idea of the Sampson error function is to estimate a first-order approximation to the point $\widehat{\mathbf{x}}$, assuming that the cost function is well approximated linearly in the neighbourhood of the estimated point. The discussion to follow is related directly to

Samsung/Google, EX1010, p. 116

the 2D homography estimation problem, but applies substantially unchanged to the other estimation problems discussed in this book.

For a given homography $\mathtt{H}$, any point $\mathbf{x} = (x, y, x', y')^\mathsf{T}$ that lies on $\mathcal{V}_\mathtt{H}$ will satisfy the equation (4.3–*p*89), or $\mathtt{A}\mathbf{h} = \mathbf{0}$. To emphasize the dependency on $\mathbf{x}$ we will write this instead as $\mathcal{C}_\mathtt{H}(\mathbf{x}) = \mathbf{0}$, where $\mathcal{C}_\mathtt{H}(\mathbf{x})$ is in this case a 2-vector. To first order, this cost function may be approximated by a Taylor expansion

$$\mathcal{C}_\mathtt{H}(\mathbf{x} + \boldsymbol{\delta}_\mathbf{x}) = \mathcal{C}_\mathtt{H}(\mathbf{x}) + \frac{\partial \mathcal{C}_\mathtt{H}}{\partial \mathbf{x}} \boldsymbol{\delta}_\mathbf{x}. \tag{4.10}$$

If we write $\boldsymbol{\delta}_\mathbf{x} = \widehat{\mathbf{x}} - \mathbf{x}$ and desire $\widehat{\mathbf{x}}$ to lie on the variety $\mathcal{V}_\mathtt{H}$ so that $\mathcal{C}_\mathtt{H}(\widehat{\mathbf{x}}) = \mathbf{0}$, then the result is $\mathcal{C}_\mathtt{H}(\mathbf{x}) + (\partial \mathcal{C}_\mathtt{H}/\partial \mathbf{x})\boldsymbol{\delta}_\mathbf{x} = \mathbf{0}$, which we will henceforth write as $\mathtt{J}\boldsymbol{\delta}_\mathbf{x} = -\boldsymbol{\epsilon}$ where $\mathtt{J}$ is the partial-derivative matrix, and $\boldsymbol{\epsilon}$ is the cost $\mathcal{C}_\mathtt{H}(\mathbf{x})$ associated with $\mathbf{x}$. The minimization problem that we now face is to find the smallest $\boldsymbol{\delta}_\mathbf{x}$ that satisfies this equation, namely:

- *Find the vector* $\boldsymbol{\delta}_\mathbf{x}$ *that minimizes* $\|\boldsymbol{\delta}_\mathbf{x}\|$ *subject to* $\mathtt{J}\boldsymbol{\delta}_\mathbf{x} = -\boldsymbol{\epsilon}$.

The standard way to solve problems of this type is to use Lagrange multipliers. A vector $\boldsymbol{\lambda}$ of Lagrange multipliers is introduced, and the problem reduces to that of finding the extrema of $\boldsymbol{\delta}_\mathbf{x}^\mathsf{T}\boldsymbol{\delta}_\mathbf{x} - 2\boldsymbol{\lambda}^\mathsf{T}(\mathtt{J}\boldsymbol{\delta}_\mathbf{x} + \boldsymbol{\epsilon})$, where the factor 2 is simply introduced for convenience. Taking derivatives with respect to $\boldsymbol{\delta}_\mathbf{x}$ and equating to zero gives

$$2\boldsymbol{\delta}_\mathbf{x}^\mathsf{T} - 2\boldsymbol{\lambda}^\mathsf{T}\mathtt{J} = \mathbf{0}^\mathsf{T}$$

from which we obtain $\boldsymbol{\delta}_\mathbf{x} = \mathtt{J}^\mathsf{T}\boldsymbol{\lambda}$. The derivative with respect to $\boldsymbol{\lambda}$ gives $\mathtt{J}\boldsymbol{\delta}_\mathbf{x} + \boldsymbol{\epsilon} = \mathbf{0}$, the original constraint. Substituting for $\boldsymbol{\delta}_\mathbf{x}$ leads to

$$\mathtt{J}\mathtt{J}^\mathsf{T}\boldsymbol{\lambda} = -\boldsymbol{\epsilon}$$

which may be solved for $\boldsymbol{\lambda}$ giving $\boldsymbol{\lambda} = -(\mathtt{J}\mathtt{J}^\mathsf{T})^{-1}\boldsymbol{\epsilon}$, and so finally

$$\boldsymbol{\delta}_\mathbf{x} = -\mathtt{J}^\mathsf{T}(\mathtt{J}\mathtt{J}^\mathsf{T})^{-1}\boldsymbol{\epsilon}, \tag{4.11}$$

and $\widehat{\mathbf{x}} = \mathbf{x} + \boldsymbol{\delta}_\mathbf{x}$. The norm $\|\boldsymbol{\delta}_\mathbf{x}\|^2$ is the Sampson error:

$$\|\boldsymbol{\delta}_\mathbf{x}\|^2 = \boldsymbol{\delta}_\mathbf{x}^\mathsf{T}\boldsymbol{\delta}_\mathbf{x} = \boldsymbol{\epsilon}^\mathsf{T}(\mathtt{J}\mathtt{J}^\mathsf{T})^{-1}\boldsymbol{\epsilon}. \tag{4.12}$$

Example 4.2. Sampson approximation for a conic

We will compute the Sampson approximation to the geometric distance $d_\perp(\mathbf{x}, \mathtt{C})$ between a point $\mathbf{x}$ and conic $\mathtt{C}$ shown in figure 4.3. In this case the conic variety $\mathcal{V}_\mathtt{C}$ is defined by the equation $\mathbf{x}^\mathsf{T}\mathtt{C}\mathbf{x} = 0$, so that $\mathbf{x} = (x, y)^\mathsf{T}$ is a 2-vector, $\boldsymbol{\epsilon} = \mathbf{x}^\mathsf{T}\mathtt{C}\mathbf{x}$ is a scalar, and $\mathtt{J}$ is the 1×2 matrix given by

$$\mathtt{J} = \left[\frac{\partial(\mathbf{x}^\mathsf{T}\mathtt{C}\mathbf{x})}{\partial x}, \frac{\partial(\mathbf{x}^\mathsf{T}\mathtt{C}\mathbf{x})}{\partial y}\right].$$

This means that $\mathtt{J}\mathtt{J}^\mathsf{T}$ is a scalar. The elements of $\mathtt{J}$ may be computed by the chain rule as

$$\frac{\partial(\mathbf{x}^\mathsf{T}\mathtt{C}\mathbf{x})}{\partial x} = \frac{\partial(\mathbf{x}^\mathsf{T}\mathtt{C}\mathbf{x})}{\partial \mathbf{x}}\frac{\partial \mathbf{x}}{\partial x} = 2\mathbf{x}^\mathsf{T}\mathtt{C}(1, 0, 0)^\mathsf{T} = 2(\mathtt{C}\mathbf{x})_1$$

Samsung/Google, EX1010, p. 117

where $(\mathtt{Cx})_i$ denotes the i-th component of the 3-vector $\mathtt{Cx}$. Then from (4.12)

$$d_\perp^2 = \|\boldsymbol{\delta}_\mathbf{X}\|^2 = \boldsymbol{\epsilon}^\mathsf{T}(\mathtt{J}\mathtt{J}^\mathsf{T})^{-1}\boldsymbol{\epsilon} = \frac{\boldsymbol{\epsilon}^\mathsf{T}\boldsymbol{\epsilon}}{\mathtt{J}\mathtt{J}^\mathsf{T}} = \frac{(\mathbf{x}^\mathsf{T}\mathtt{C}\mathbf{x})^2}{4((\mathtt{C}\mathbf{x})_1^2 + (\mathtt{C}\mathbf{x})_2^2)}$$

△

A few points to note:

(i) For the 2D homography estimation problem, $\mathbf{X} = (x, y, x', y')^\mathsf{T}$ where the 2D measurements are $\mathbf{x} = (x, y, 1)^\mathsf{T}$ and $\mathbf{x}' = (x', y', 1)^\mathsf{T}$.

(ii) $\boldsymbol{\epsilon} = \mathcal{C}_\mathtt{H}(\mathbf{X})$ is the algebraic error vector $\mathtt{A}_i\mathbf{h}$ – a 2-vector – and $\mathtt{A}_i$ is defined in (4.3–p89).

(iii) $\mathtt{J} = \partial\mathcal{C}_\mathtt{H}(\mathbf{X})/\partial\mathbf{X}$ is a 2×4 matrix. For example

$$J_{11} = \partial(-w_i'\mathbf{x}_i^\mathsf{T}\mathbf{h}^2 + y_i'\mathbf{x}_i^\mathsf{T}\mathbf{h}^3)/\partial x = -w_i'h_{21} + y_i'h_{31}.$$

(iv) Note the similarity of (4.12) to the algebraic error $\|\boldsymbol{\epsilon}\| = \boldsymbol{\epsilon}^\mathsf{T}\boldsymbol{\epsilon}$. The Sampson error may be interpreted as being the Mahalanobis norm (see section A2.1-(p565)), $\|\boldsymbol{\epsilon}\|_{\mathtt{J}\mathtt{J}^\mathsf{T}}$.

(v) One could alternatively use $\mathtt{A}$ defined by (4.1–p89), in which case $\mathtt{J}$ has dimension 3×4 and $\boldsymbol{\epsilon}$ is a 3-vector. However, in general the Sampson error, and consequently the solution $\boldsymbol{\delta}_\mathbf{X}$, will be independent of whether (4.1–p89) or (4.3–p89) is used.

The Sampson error (4.12) is derived here for a single point pair. In applying this to the estimation of a 2D homography $\mathtt{H}$ from several point correspondences $\mathbf{x}_i \leftrightarrow \mathbf{x}_i'$, the errors corresponding to all the point correspondences must be summed, giving

$$\mathcal{D}_\perp = \sum_i \boldsymbol{\epsilon}_i^\mathsf{T}(\mathtt{J}_i\mathtt{J}_i^\mathsf{T})^{-1}\boldsymbol{\epsilon}_i \tag{4.13}$$

where $\boldsymbol{\epsilon}$ and $\mathtt{J}$ both depend on $\mathtt{H}$. To estimate $\mathtt{H}$, this expression must be minimized over all values of $\mathtt{H}$. This is a simple minimization problem in which the set of variable parameters consists only of the entries (or some other parametrization) of $\mathtt{H}$.

This derivation of the Sampson error assumed that each point had isotropic (circular) error distribution, the same in each image. The appropriate formulae for more general Gaussian error distributions are given in the exercises at the end of this chapter.

Linear cost function

The algebraic error vector $\mathcal{C}_\mathtt{H}(\mathbf{X}) = \mathtt{A}(\mathbf{X})\mathbf{h}$ is typically multilinear in the entries of $\mathbf{X}$. The case where $\mathtt{A}(\mathbf{X})\mathbf{h}$ is *linear* is, however, important in its own right. The first point to note is that in this case, the first-order approximation to geometric error given by the Taylor expansion in (4.10) is exact (the higher order terms are zero), which means that *the Sampson error is identical to geometric error.*

In addition, the variety $\mathcal{V}_\mathtt{H}$ defined by the equation $\mathcal{C}_\mathtt{H}(\mathbf{X}) = \mathbf{0}$, a set of linear equations, is a hyperplane depending on $\mathtt{H}$. The problem of finding $\mathtt{H}$ now becomes a hyperplane fitting problem – find the best fit to the data $\mathbf{X}_i$ among the hyperplanes parametrized by $\mathtt{H}$.

Samsung/Google, EX1010, p. 118

As an example of this idea a *linear* algorithm which minimizes geometric error (4.8) for an affine transformation is developed in the exercises at the end of this chapter.

4.2.7 Another geometric interpretation

It was shown in section 4.2.5 that finding a homography that takes a set of points $\mathbf{x}_i$ to another set $\mathbf{x}'_i$ is equivalent to the problem of fitting a variety of a given type to a set of points in $\mathbb{R}^4$. We now consider a different interpretation in which the set of all measurements is represented by a single point in a measurement space $\mathbb{R}^N$.

The estimation problems we consider may all be fitted into a common framework. In abstract terms the estimation problem has two components,

- a *measurement space* $\mathbb{R}^N$ consisting of *measurement vectors* $\mathbf{X}$, and
- a *model*, which in abstract terms may be thought of simply as a subset S of points in $\mathbb{R}^N$. A measurement vector $\mathbf{X}$ that lies inside this subset is said to *satisfy the model*. Typically the subspace that satisfies the model is a submanifold, or variety in $\mathbb{R}^N$.

Now, given a measurement vector $\mathbf{X}$ in $\mathbb{R}^N$, the estimation problem is to find the vector $\hat{\mathbf{X}}$, closest to $\mathbf{X}$, that satisfies the model.

It will now be pointed out how the 2D homography estimation problem fits into this framework.

Error in both images. Let $\{\mathbf{x}_i \leftrightarrow \mathbf{x}'_i\}$ be a set of measured matched points for $i = 1, \ldots, n$. In all, there are $4n$ measurements, namely two coordinates in each of two images for n points. Thus, the set of matched points represents a point in $\mathbb{R}^N$, where $N = 4n$. The vector made up of the coordinates of all the matched points in both images will be denoted $\mathbf{X}$.

Of course, not all sets of point pairs $\mathbf{x}_i \leftrightarrow \mathbf{x}'_i$ are related via a homography $\mathtt{H}$. A set of point correspondences $\{\mathbf{x}_i \leftrightarrow \mathbf{x}'_i\}$ for which there exists a projective transformation $\mathtt{H}$ satisfying $\mathbf{x}'_i = \mathtt{H}\mathbf{x}_i$ for all i constitutes the subset of $\mathbb{R}^N$ satisfying the model. In general, this set of points will form a submanifold S in $\mathbb{R}^N$ (in fact a variety) of some dimension. The dimension of this submanifold is equal to the minimal number of parameters that may be used to parametrize the submanifold.

One may arbitrarily choose n points $\hat{\mathbf{x}}_i$ in the first image. In addition, a homography $\mathtt{H}$ may be chosen arbitrarily. Once these choices have been made, the points $\hat{\mathbf{x}}'_i$ in the second image are determined by $\hat{\mathbf{x}}'_i = \mathtt{H}\hat{\mathbf{x}}_i$. Thus, a feasible choice of points is determined by a set of $2n + 8$ parameters: the $2n$ coordinates of the points $\hat{\mathbf{x}}_i$, plus the 8 independent parameters (degrees of freedom) of the transformation $\mathtt{H}$. Thus, the submanifold $S \subset \mathbb{R}^N$ has dimension $2n + 8$, and hence codimension $2n - 8$.

Given a set of measured point pairs $\{\mathbf{x}_i \leftrightarrow \mathbf{x}'_i\}$, corresponding to a point $\mathbf{X}$ in $\mathbb{R}^N$, and an estimated point $\widehat{\mathbf{X}} \in \mathbb{R}^N$ lying on S, one easily verifies that

$$\|\mathbf{X} - \widehat{\mathbf{X}}\|^2 = \sum_i d(\mathbf{x}_i, \hat{\mathbf{x}}_i)^2 + d(\mathbf{x}'_i, \hat{\mathbf{x}}'_i)^2.$$

Thus, finding the point $\widehat{\mathbf{X}}$ on S lying closest to $\mathbf{X}$ in $\mathbb{R}^N$ is equivalent to minimizing the cost function given by (4.8). The estimated correct correspondences $\hat{\mathbf{x}}_i \leftrightarrow \hat{\mathbf{x}}'_i$ are

Samsung/Google, EX1010, p. 119

those corresponding to the closest surface point $\hat{\mathbf{X}}$ in $\mathbb{R}^N$. Once $\hat{\mathbf{X}}$ is known $\mathtt{H}$ may be computed.

Error in one image only. In the case of error in one image, one has a set of correspondences $\{\bar{\mathbf{x}}_i \leftrightarrow \mathbf{x}'_i\}$. The points $\bar{\mathbf{x}}_i$ are assumed perfect. The inhomogeneous coordinates of the $\mathbf{x}'_i$ constitute the measurement vector $\mathbf{X}$. Hence, in this case the measurement space has dimension $N = 2n$. The vector $\hat{\mathbf{X}}$ consists of the inhomogeneous coordinates of the mapped perfect points $\{\mathtt{H}\bar{\mathbf{x}}_1, \mathtt{H}\bar{\mathbf{x}}_2, \ldots, \mathtt{H}\bar{\mathbf{x}}_n\}$. The set of measurement vectors satisfying the model is the set $\hat{\mathbf{X}}$ as $\mathtt{H}$ varies over the set of all homography matrices. Once again this subspace is a variety. Its dimension is 8, since this is the total number of degrees of freedom of the homography matrix $\mathtt{H}$. As with the previous case, the codimension is $2n - 8$. One verifies that

$$\|\mathbf{X} - \hat{\mathbf{X}}\|^2 = \sum_i d(\mathbf{x}'_i, \mathtt{H}\bar{\mathbf{x}}_i)^2.$$

Thus, finding the closest point on S to the measurement vector $\mathbf{X}$ is equivalent to minimizing the cost function (4.6).

4.3 Statistical cost functions and Maximum Likelihood estimation

In section 4.2, various cost functions were considered that were related to geometric distance between estimated and measured points in an image. The use of such cost functions is now justified and then generalized by a consideration of error statistics of the point measurements in an image.

In order to obtain a best (optimal) estimate of $\mathtt{H}$ it is necessary to have a model for the measurement error (the "noise"). We are assuming here that in the absence of measurement error the true points exactly satisfy a homography, i.e. $\bar{\mathbf{x}}'_i = \mathtt{H}\bar{\mathbf{x}}_i$. A common assumption is that image coordinate measurement errors obey a Gaussian (or normal) probability distribution. This assumption is surely not justified in general, and takes no account of the presence of outliers (grossly erroneous measurements) in the measured data. Methods for detecting and removing outliers will be discussed later in section 4.7. Once outliers have been removed, the assumption of a Gaussian error model, if still not strictly justified, becomes more tenable. Therefore, for the present, we assume that image measurement errors obey a zero-mean isotropic Gaussian distribution. This distribution is described in section A2.1(p565).

Specifically we assume that the noise is Gaussian on each image coordinate with zero mean and uniform standard deviation σ. This means that $x = \bar{x} + \Delta x$, with Δx obeying a Gaussian distribution with variance σ^2. If it is further assumed that the noise on each measurement is independent, then, if the true point is $\bar{\mathbf{x}}$, the probability density function (PDF) of each measured point $\mathbf{x}$ is

$$\Pr(\mathbf{x}) = \left(\frac{1}{2\pi\sigma^2}\right) e^{-d(\mathbf{x},\bar{\mathbf{x}})^2/(2\sigma^2)}. \tag{4.14}$$

Error in one image. First we consider the case where the errors are only in the second image. The probability of obtaining the set of correspondences $\{\bar{\mathbf{x}}_i \leftrightarrow \mathbf{x}'_i\}$ is

Samsung/Google, EX1010, p. 120

simply the product of their individual PDFs, since the errors on each point are assumed independent. Then the PDF of the noise-perturbed data is

$$\Pr(\{\mathbf{x}'_i\}|\mathtt{H}) = \prod_i \left(\frac{1}{2\pi\sigma^2}\right) e^{-d(\mathbf{x}'_i, \mathtt{H}\bar{\mathbf{x}}_i)^2/(2\sigma^2)} \quad . \tag{4.15}$$

The symbol $\Pr(\{\mathbf{x}'_i\}|\mathtt{H})$ is to be interpreted as meaning the probability of obtaining the measurements $\{\mathbf{x}'_i\}$ given that the true homography is $\mathtt{H}$. The *log-likelihood* of the set of correspondences is

$$\log \Pr(\{\mathbf{x}'_i\}|\mathtt{H}) = -\frac{1}{2\sigma^2}\sum_i d(\mathbf{x}'_i, \mathtt{H}\bar{\mathbf{x}}_i)^2 + \text{ constant.}$$

The *Maximum Likelihood estimate* (MLE) of the homography, $\hat{\mathtt{H}}$, maximizes this log-likelihood, i.e. minimizes

$$\sum_i d(\mathbf{x}'_i, \mathtt{H}\bar{\mathbf{x}}_i)^2.$$

Thus, we note that ML estimation is equivalent to minimizing the geometric error function (4.6).

Error in both images. Following a similar development to the above, if the true correspondences are $\{\bar{\mathbf{x}}_i \leftrightarrow \mathtt{H}\bar{\mathbf{x}}_i = \bar{\mathbf{x}}'_i\}$, then the PDF of the noise-perturbed data is

$$\Pr(\{\mathbf{x}_i, \mathbf{x}'_i\}|\mathtt{H}, \{\bar{\mathbf{x}}_i\}) = \prod_i \left(\frac{1}{2\pi\sigma^2}\right) e^{-\left(d(\mathbf{x}_i, \bar{\mathbf{x}}_i)^2 + d(\mathbf{x}'_i, \mathtt{H}\bar{\mathbf{x}}_i)^2\right)/(2\sigma^2)}.$$

The additional complication here is that we have to seek "corrected" image measurements that play the role of the true measurements ($\mathtt{H}\bar{\mathbf{x}}$ above). Thus the ML estimate of the projective transformation $\mathtt{H}$ *and* the correspondences $\{\mathbf{x}_i \leftrightarrow \mathbf{x}'_i\}$, is the homography $\hat{\mathtt{H}}$ and corrected correspondences $\{\hat{\mathbf{x}}_i \leftrightarrow \hat{\mathbf{x}}'_i\}$ that minimize

$$\sum_i d(\mathbf{x}_i, \hat{\mathbf{x}}_i)^2 + d(\mathbf{x}'_i, \hat{\mathbf{x}}'_i)^2$$

with $\hat{\mathbf{x}}'_i = \hat{\mathtt{H}}\hat{\mathbf{x}}_i$. Note that in this case, the ML estimate is identical with minimizing the reprojection error function (4.8).

Mahalanobis distance. In the general Gaussian case, one may assume a vector of measurements $\mathbf{X}$ satisfying a Gaussian distribution function with covariance matrix Σ. The cases above are equivalent to a covariance matrix which is a multiple of the identity.

Maximizing the log-likelihood is then equivalent to minimizing the Mahalanobis distance (see section A2.1(p565))

$$\|\mathbf{X} - \bar{\mathbf{X}}\|^2_\Sigma = (\mathbf{X} - \bar{\mathbf{X}})^\top \Sigma^{-1} (\mathbf{X} - \bar{\mathbf{X}}).$$

In the case where there is error in each image, but assuming that errors in one image are independent of the error in the other image, the appropriate cost function is

$$\|\mathbf{X} - \bar{\mathbf{X}}\|^2_\Sigma + \|\mathbf{X}' - \bar{\mathbf{X}}'\|^2_{\Sigma'}$$

Samsung/Google, EX1010, p. 121

where Σ and Σ' are the covariance matrices of the measurements in the two images.

Finally, if we assume that the errors for all the points $\mathbf{x}_i$ and $\mathbf{x}'_i$ are independent, with individual covariance matrices Σ_i and Σ'_i respectively, then the above expression expands to

$$\sum \|\mathbf{x}_i - \bar{\mathbf{x}}_i\|^2_{\Sigma_i} + \sum \|\mathbf{x}'_i - \bar{\mathbf{x}}'_i\|^2_{\Sigma'_i} \tag{4.16}$$

This equation allows the incorporation of the type of anisotropic covariance matrices that arise for point locations computed as the intersection of two non-perpendicular lines. In the case where the points are known exactly in one of the two images, errors being confined to the other image, one of the two summation terms in (4.16) disappears.

4.4 Transformation invariance and normalization

We now start to discuss the properties and performance of the DLT algorithm of section 4.1 and how it compares with algorithms minimizing geometric error. The first topic is the invariance of the algorithm to different choices of coordinates in the image. It is clear that it would generally be undesirable for the result of an algorithm to be dependent on such arbitrary choices as the origin and scale, or even orientation, of the coordinate system in an image.

4.4.1 Invariance to image coordinate transformations

Image coordinates are sometimes given with the origin at the top-left of the image, and sometimes with the origin at the centre. The question immediately occurs whether this makes a difference to the results of computing the transformation. Similarly, if the units used to express image coordinates are changed by multiplication by some factor, then is it possible that the result of the algorithm changes also? More generally, to what extent is the result of an algorithm that minimizes a cost function to estimate a homography dependent on the choice of coordinates in the image? Suppose, for instance, that the image coordinates are changed by some similarity, affine or even projective transformation before running the algorithm. Will this materially change the result?

Formally, suppose that coordinates $\mathbf{x}$ in one image are replaced by $\tilde{\mathbf{x}} = \mathtt{T}\mathbf{x}$, and coordinates $\mathbf{x}'$ in the other image are replaced by $\tilde{\mathbf{x}}' = \mathtt{T}'\mathbf{x}'$, where $\mathtt{T}$ and $\mathtt{T}'$ are 3×3 homographies. Substituting in the equation $\mathbf{x}' = \mathtt{H}\mathbf{x}$, we derive the equation $\tilde{\mathbf{x}}' = \mathtt{T}'\mathtt{H}\mathtt{T}^{-1}\tilde{\mathbf{x}}$. This relation implies that $\tilde{\mathtt{H}} = \mathtt{T}'\mathtt{H}\mathtt{T}^{-1}$ is the transformation matrix for the point correspondences $\tilde{\mathbf{x}} \leftrightarrow \tilde{\mathbf{x}}'$. An alternative method of finding the transformation taking $\mathbf{x}_i$ to $\mathbf{x}'_i$ is therefore suggested, as follows.

(i) Transform the image coordinates according to transformations $\tilde{\mathbf{x}}_i = \mathtt{T}\mathbf{x}_i$ and $\tilde{\mathbf{x}}'_i = \mathtt{T}'\mathbf{x}'_i$.
(ii) Find the transformation $\tilde{\mathtt{H}}$ from the correspondences $\tilde{\mathbf{x}}_i \leftrightarrow \tilde{\mathbf{x}}'_i$.
(iii) Set $\mathtt{H} = \mathtt{T}'^{-1}\tilde{\mathtt{H}}\mathtt{T}$.

The transformation matrix $\mathtt{H}$ found in this way applies to the original untransformed point correspondences $\mathbf{x}_i \leftrightarrow \mathbf{x}'_i$. What choice should be made for the transformations $\mathtt{T}$ and $\mathtt{T}'$ will be left unspecified for now. The question to be decided now is whether the

Samsung/Google, EX1010, p. 122

outcome of this algorithm is independent of the transformations $\mathtt{T}$ and $\mathtt{T}'$ being applied. Ideally it ought to be, at least when $\mathtt{T}$ and $\mathtt{T}'$ are similarity transformations, since the choice of a different scale, orientation or coordinate origin in the images should not materially affect the outcome of the algorithm.

In the subsequent sections it will be shown that an algorithm that minimizes geometric error is invariant to similarity transformations. On the other hand, for the DLT algorithm as described in section 4.1, the result unfortunately is not invariant to similarity transformations. The solution is to apply a normalizing transformation to the data before applying the DLT algorithm. This normalizing transformation will nullify the effect of the arbitrary selection of origin and scale in the coordinate frame of the image, and will mean that the combined algorithm is invariant to a similarity transformation of the image. Appropriate normalizing transformations will be discussed later.

4.4.2 Non-invariance of the DLT algorithm

Consider a set of correspondences $\mathbf{x}_i \leftrightarrow \mathbf{x}'_i$ and a matrix $\mathtt{H}$ that is the result of the DLT algorithm applied to this set of corresponding points. Consider further a related set of correspondences $\tilde{\mathbf{x}}_i \leftrightarrow \tilde{\mathbf{x}}'_i$ where $\tilde{\mathbf{x}}_i = \mathtt{T}\mathbf{x}_i$ and $\tilde{\mathbf{x}}'_i = \mathtt{T}'\mathbf{x}'_i$, and let $\tilde{\mathtt{H}}$ be defined by $\tilde{\mathtt{H}} = \mathtt{T}'\mathtt{H}\mathtt{T}^{-1}$. Following section 4.4.1, the question to be decided here is the following:

- *Does the DLT algorithm applied to the correspondence set $\tilde{\mathbf{x}}_i \leftrightarrow \tilde{\mathbf{x}}'_i$ yield the transformation $\tilde{\mathtt{H}}$?*

We will use the following notation: Matrix $\mathtt{A}_i$ is the DLT equation matrix (4.3–*p*89) derived from a point correspondence $\mathbf{x}_i \leftrightarrow \mathbf{x}'_i$, and $\mathtt{A}$ is the $2n \times 9$ matrix formed by stacking the $\mathtt{A}_i$. Matrix $\tilde{\mathtt{A}}_i$ is similarly defined in terms of the correspondences $\tilde{\mathbf{x}}_i \leftrightarrow \tilde{\mathbf{x}}'_i$, where $\tilde{\mathbf{x}}_i = \mathtt{T}\mathbf{x}_i$ and $\tilde{\mathbf{x}}'_i = \mathtt{T}'\mathbf{x}'_i$ for some projective transformations $\mathtt{T}$ and $\mathtt{T}'$.

Result 4.3. *Let $\mathtt{T}'$ be a similarity transformation with scale factor s, and let $\mathtt{T}$ be an arbitrary projective transformation. Further, suppose $\mathtt{H}$ is any 2D homography and let $\tilde{\mathtt{H}}$ be defined by $\tilde{\mathtt{H}} = \mathtt{T}'\mathtt{H}\mathtt{T}^{-1}$. Then $\|\tilde{\mathtt{A}}\tilde{\mathbf{h}}\| = s\|\mathtt{A}\mathbf{h}\|$ where $\mathbf{h}$ and $\tilde{\mathbf{h}}$ are the vectors of entries of $\mathtt{H}$ and $\tilde{\mathtt{H}}$.*

Proof. Define the vector $\boldsymbol{\epsilon}_i = \mathbf{x}'_i \times \mathtt{H}\mathbf{x}_i$. Note that $\mathtt{A}_i\mathbf{h}$ is the vector consisting of the first two entries of $\boldsymbol{\epsilon}_i$. Let $\tilde{\boldsymbol{\epsilon}}_i$ be similarly defined in terms of the transformed quantities as $\tilde{\boldsymbol{\epsilon}}_i = \tilde{\mathbf{x}}'_i \times \tilde{\mathtt{H}}\tilde{\mathbf{x}}_i$. One computes:

$$\begin{aligned}\tilde{\boldsymbol{\epsilon}}_i &= \tilde{\mathbf{x}}'_i \times \tilde{\mathtt{H}}\tilde{\mathbf{x}}_i = \mathtt{T}'\mathbf{x}'_i \times (\mathtt{T}'\mathtt{H}\mathtt{T}^{-1})\mathtt{T}\mathbf{x}_i \\ &= \mathtt{T}'\mathbf{x}'_i \times \mathtt{T}'\mathtt{H}\mathbf{x}_i = \mathtt{T}'^{*}(\mathbf{x}'_i \times \mathtt{H}\mathbf{x}_i) \\ &= \mathtt{T}'^{*}\boldsymbol{\epsilon}_i\end{aligned}$$

where $\mathtt{T}'^{*}$ represents the cofactor matrix of $\mathtt{T}'$ and the second-last equality follows from lemma A4.2(*p*581). For a general transformation $\mathtt{T}$, the error vectors $\mathtt{A}_i\mathbf{h}$ and $\tilde{\mathtt{A}}_i\tilde{\mathbf{h}}$ (namely the first two components of $\boldsymbol{\epsilon}_i$ and $\tilde{\boldsymbol{\epsilon}}_i$) are not simply related. However, in the special case where $\mathtt{T}'$ is a similarity transformation, one may write $\mathtt{T}' = \begin{bmatrix} s\mathtt{R} & \mathbf{t} \\ \mathbf{0}^\mathsf{T} & 1 \end{bmatrix}$ where $\mathtt{R}$ is a rotation matrix, $\mathbf{t}$ is a translation and s is a scaling factor. In this case, we

Samsung/Google, EX1010, p. 123

see that $\mathtt{T}'^{*} = s\begin{bmatrix} \mathtt{R} & \mathbf{0} \\ -\mathbf{t}^{\mathsf{T}}\mathtt{R} & s \end{bmatrix}$. Applying $\mathtt{T}'^{*}$ just to the first two components of $\boldsymbol{\epsilon}_i$, one sees that

$$\tilde{\mathtt{A}}_i\tilde{\mathbf{h}} = (\tilde{\epsilon}_{i1}, \tilde{\epsilon}_{i2})^{\mathsf{T}} = s\mathtt{R}(\epsilon_{i1}, \epsilon_{i2})^{\mathsf{T}} = s\mathtt{R}\mathtt{A}_i\mathbf{h}.$$

Since rotation does not affect vector norms, one sees that $\|\tilde{\mathtt{A}}\tilde{\mathbf{h}}\| = s\|\mathtt{A}\mathbf{h}\|$, as required. This result may be expressed in terms of algebraic error as

$$d_{\text{alg}}(\tilde{\mathbf{x}}'_i, \tilde{\mathtt{H}}\tilde{\mathbf{x}}_i) = sd_{\text{alg}}(\mathbf{x}'_i, \mathtt{H}\mathbf{x}_i).$$

□

Thus, there is a one-to-one correspondence between $\mathtt{H}$ and $\tilde{\mathtt{H}}$ giving rise to the same error, except for constant scale. It may appear therefore that the matrices $\mathtt{H}$ and $\tilde{\mathtt{H}}$ minimizing the algebraic error will be related by the formula $\tilde{\mathtt{H}} = \mathtt{T}'\mathtt{H}\mathtt{T}^{-1}$, and hence one may retrieve $\mathtt{H}$ as the product $\mathtt{T}'^{-1}\tilde{\mathtt{H}}\mathtt{T}$. This conclusion is **false** however. For, although $\mathtt{H}$ and $\tilde{\mathtt{H}}$ so defined give rise to the same error ϵ, the condition $\|\mathtt{H}\| = 1$, imposed as a constraint on the solution, is not equivalent to the condition $\|\tilde{\mathtt{H}}\| = 1$. Specifically, $\|\mathtt{H}\|$ and $\|\tilde{\mathtt{H}}\|$ are not related in any simple manner. Thus, there is no one-to-one correspondence between $\mathtt{H}$ and $\tilde{\mathtt{H}}$ giving rise to the same error ϵ, subject to the constraint $\|\mathtt{H}\| = \|\tilde{\mathtt{H}}\| = 1$. Specifically,

$$\begin{aligned} & \text{minimize } \sum_i d_{\text{alg}}(\mathbf{x}'_i, \mathtt{H}\mathbf{x}_i)^2 \text{ subject to } \|\mathtt{H}\| = 1 \\ \Leftrightarrow\ & \text{minimize } \sum_i d_{\text{alg}}(\tilde{\mathbf{x}}'_i, \tilde{\mathtt{H}}\tilde{\mathbf{x}}_i)^2 \text{ subject to } \|\mathtt{H}\| = 1 \\ \not\Leftrightarrow\ & \text{minimize } \sum_i d_{\text{alg}}(\tilde{\mathbf{x}}'_i, \tilde{\mathtt{H}}\tilde{\mathbf{x}}_i)^2 \text{ subject to } \|\tilde{\mathtt{H}}\| = 1. \end{aligned}$$

Thus, the method of transformation leads to a different solution for the computed transformation matrix. This is a rather undesirable feature of the DLT algorithm as it stands, that the result is changed by a change of coordinates, or even simply a change of the origin of coordinates. If the constraint under which the norm $\|\mathtt{A}\mathbf{h}\|$ is minimized is invariant under the transformation, however, then one sees that the computed matrices $\mathtt{H}$ and $\tilde{\mathtt{H}}$ are related in the right way. Examples of minimization conditions for which $\mathtt{H}$ is transformation-invariant are discussed in the exercises at the end of this chapter.

4.4.3 Invariance of geometric error

It will be shown now that minimizing geometric error to find $\mathtt{H}$ is invariant under similarity (scaled Euclidean) transformations. As before, consider a point correspondence $\mathbf{x} \leftrightarrow \mathbf{x}'$ and a transformation matrix $\mathtt{H}$. Also, define a related set of correspondences $\tilde{\mathbf{x}} \leftrightarrow \tilde{\mathbf{x}}'$ where $\tilde{\mathbf{x}} = \mathtt{T}\mathbf{x}$ and $\tilde{\mathbf{x}}' = \mathtt{T}'\mathbf{x}'$, and let $\tilde{\mathtt{H}}$ be defined by $\tilde{\mathtt{H}} = \mathtt{T}'\mathtt{H}\mathtt{T}^{-1}$. Suppose that $\mathtt{T}$ and $\mathtt{T}'$ represent Euclidean transformations of $\mathbb{P}^2$. One verifies that

$$d(\tilde{\mathbf{x}}', \tilde{\mathtt{H}}\tilde{\mathbf{x}}) = d(\mathtt{T}'\mathbf{x}', \mathtt{T}'\mathtt{H}\mathtt{T}^{-1}\mathtt{T}\mathbf{x}) = d(\mathtt{T}'\mathbf{x}', \mathtt{T}'\mathtt{H}\mathbf{x}) = d(\mathbf{x}', \mathtt{H}\mathbf{x})$$

where the last equality holds because Euclidean distance is unchanged under a Euclidean transformation such as $\mathtt{T}'$. This shows that if $\mathtt{H}$ minimizes the geometric error

Samsung/Google, EX1010, p. 124

for a set of correspondences, then $\widetilde{\mathtt{H}}$ minimizes the geometric error for the transformed set of correspondences, and so minimizing geometric error is invariant under Euclidean transformations.

For similarity transformations, geometric error is multiplied by the scale factor of the transformation, hence the minimizing transformations correspond in the same way as in the Euclidean transformation case. Minimizing geometric error is invariant to similarity transformations.

4.4.4 Normalizing transformations

As was shown in section 4.4.2, the result of the DLT algorithm for computing 2D homographies depends on the coordinate frame in which points are expressed. In fact the result is not invariant to similarity transformations of the image. This suggests the question whether some coordinate systems are in some way better than others for computing a 2D homography. The answer to this is an emphatic *yes*. In this section a method of normalization of the data is described, consisting of translation and scaling of image coordinates. This normalization should be carried out before applying the DLT algorithm. Subsequently an appropriate correction to the result expresses the computed $\mathtt{H}$ with respect to the original coordinate system.

Apart from improved accuracy of results, data normalization provides a second desirable benefit, namely that an algorithm that incorporates an initial data normalization step will be invariant with respect to arbitrary choices of the scale and coordinate origin. This is because the normalization step undoes the effect of coordinate changes, by effectively choosing a canonical coordinate frame for the measurement data. Thus, algebraic minimization is carried out in a fixed canonical frame, and the DLT algorithm is in practice invariant to similarity transformations.

Isotropic scaling. As a first step of normalization, the coordinates in each image are translated (by a different translation for each image) so as to bring the centroid of the set of all points to the origin. The coordinates are also scaled so that on the average a point $\mathbf{x}$ is of the form $\mathbf{x} = (x, y, w)^\mathsf{T}$, with each of x, y and w having the same average magnitude. Rather than choose different scale factors for each coordinate direction, an isotropic scaling factor is chosen so that the x and y-coordinates of a point are scaled equally. To this end, we choose to scale the coordinates so that the average distance of a point $\mathbf{x}$ from the origin is equal to $\sqrt{2}$. This means that the "average" point is equal to $(1, 1, 1)^\mathsf{T}$. In summary the transformation is as follows:

(i) The points are translated so that their centroid is at the origin.
(ii) The points are then scaled so that the average distance from the origin is equal to $\sqrt{2}$.
(iii) This transformation is applied to each of the two images independently.

Why is normalization essential? The recommended version of the DLT algorithm with data normalization is given in algorithm 4.2. We will now motivate why this

Samsung/Google, EX1010, p. 125

version of the algorithm, incorporating data normalization, should be used in preference to the basic DLT of algorithm 4.1($p91$). Note that normalization is also called *pre-conditioning* in the numerical literature.

The DLT method of algorithm 4.1 uses the SVD of $\mathtt{A} = \mathtt{U}\mathtt{D}\mathtt{V}^\mathsf{T}$ to obtain a solution to the overdetermined set of equations $\mathtt{A}\mathbf{h} = \mathbf{0}$. These equations do not have an exact solution (since the $2n \times 9$ matrix $\mathtt{A}$ will not have rank 8 for noisy data), but the vector $\mathbf{h}$, given by the last column of $\mathtt{V}$, provides a solution which minimizes $\|\mathtt{A}\mathbf{h}\|$ (subject to $\|\mathbf{h}\| = 1$). This is equivalent to finding the rank 8 matrix $\hat{\mathtt{A}}$ which is closest to $\mathtt{A}$ in Frobenius norm and obtaining $\mathbf{h}$ as the exact solution of $\hat{\mathtt{A}}\mathbf{h} = \mathbf{0}$. The matrix $\hat{\mathtt{A}}$ is given by $\hat{\mathtt{A}} = \mathtt{U}\hat{\mathtt{D}}\mathtt{V}^\mathsf{T}$ where $\hat{\mathtt{D}}$ is $\mathtt{D}$ with the smallest singular value set to zero. The matrix $\hat{\mathtt{A}}$ has rank 8 and minimizes the difference to $\mathtt{A}$ in Frobenius norm because

$$\|\mathtt{A} - \hat{\mathtt{A}}\|_\mathrm{F} = \|\mathtt{U}\mathtt{D}\mathtt{V}^\mathsf{T} - \mathtt{U}\hat{\mathtt{D}}\mathtt{V}^\mathsf{T}\|_\mathrm{F} = \|\mathtt{D} - \hat{\mathtt{D}}\|_\mathrm{F}.$$

where $\|.\|_\mathrm{F}$ is the Frobenius norm, i.e. the square root of the sum of squares of all entries.

Without normalization typical image points $\mathbf{x}_i, \mathbf{x}_i'$ are of the order $(x, y, w)^\mathsf{T} = (100, 100, 1)^\mathsf{T}$, i.e., x, y are much larger than w. In $\mathtt{A}$ the entries xx', xy', yx', yy' will be of order 10^4, entries xw', yw' etc. of order 10^2, and entries ww' will be unity. Replacing $\mathtt{A}$ by $\hat{\mathtt{A}}$ means that some entries are increased and others decreased such that the square sum of differences of these changes is minimal (and the resulting matrix has rank 8). However, and this is the key point, increasing the term ww' by 100 means a huge change in the image points, whereas increasing the term xx' by 100 means only a slight change. This is the reason why all entries in $\mathtt{A}$ must have similar magnitude and why normalization is essential.

The effect of normalization is related to the condition number of the set of DLT equations, or more precisely the ratio d_1/d_{n-1} of the first to the second-last singular value of the equation matrix $\mathtt{A}$. This point is investigated in more detail in [Hartley-97c]. For the present it is sufficient to say that for exact data and infinite precision arithmetic the results will be independent of the normalizing transformation. However, in the presence of noise the solution will diverge from the correct result. The effect of a large condition number is to amplify this divergence. This is true even for infinite-precision arithmetic – this is not a round-off error effect.

The effect that this data normalization has on the results of the DLT algorithm is shown graphically in figure 4.4. The conclusion to be drawn here is that data normalization gives dramatically better results. The examples shown in the figure are chosen to make the effect easily visible. However, a marked advantage remains even in cases of computation from larger numbers of point correspondences, with points more widely distributed. To emphasize this point we remark:

- *Data normalization is an* essential *step in the DLT algorithm. It must* not *be considered optional.*

Data normalization becomes even more important for less well conditioned problems, such as the DLT computation of the fundamental matrix or the trifocal tensor, which will be considered in later chapters.

Samsung/Google, EX1010, p. 126

Objective

Given $n \geq 4$ 2D to 2D point correspondences $\{\mathbf{x}_i \leftrightarrow \mathbf{x}'_i\}$, determine the 2D homography matrix $\mathtt{H}$ such that $\mathbf{x}'_i = \mathtt{H}\mathbf{x}_i$.

Algorithm

(i) **Normalization of $\mathbf{x}$:** Compute a similarity transformation $\mathtt{T}$, consisting of a translation and scaling, that takes points $\mathbf{x}_i$ to a new set of points $\tilde{\mathbf{x}}_i$ such that the centroid of the points $\tilde{\mathbf{x}}_i$ is the coordinate origin $(0, 0)^\mathsf{T}$, and their average distance from the origin is $\sqrt{2}$.

(ii) **Normalization of $\mathbf{x}'$:** Compute a similar transformation $\mathtt{T}'$ for the points in the second image, transforming points $\mathbf{x}'_i$ to $\tilde{\mathbf{x}}'_i$.

(iii) **DLT:** Apply algorithm 4.1(*p*91) to the correspondences $\tilde{\mathbf{x}}_i \leftrightarrow \tilde{\mathbf{x}}'_i$ to obtain a homography $\tilde{\mathtt{H}}$.

(iv) **Denormalization:** Set $\mathtt{H} = \mathtt{T}'^{-1}\tilde{\mathtt{H}}\mathtt{T}$.

Algorithm 4.2. *The normalized DLT for 2D homographies.*

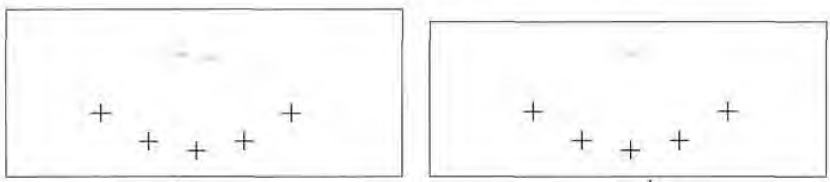


Fig. 4.4. *Results of Monte Carlo simulation (see section 5.3(*p*149) of computation of 2D homographies). A set of 5 points (denoted by large crosses) was used to compute a 2D homography. Each of the 5 points is mapped (in the noise-free case) to the point with the same coordinates, so that homography* $\mathtt{H}$ *is the identity mapping. Now, 100 trials were made with each point being subject to 0.1 pixel Gaussian noise in one image. (For reference, the large crosses are 4 pixels across.) The mapping* $\mathtt{H}$ *computed using the DLT algorithm was then applied to transfer a further point into the second image. The 100 projections of this point are shown with small crosses and the 95% ellipse computed from their scatter matrix is also shown. (a) are the results without data normalization, and (b) the results with normalization. The left- and rightmost reference points have (unnormalized) coordinates* $(130, 108)$ *and* $(170, 108)$.

Non-isotropic scaling. Other methods of scaling are also possible. In non-isotropic scaling, the centroid of the points is translated to the origin as before. After this translation the points form a cloud about the origin. Scaling is then carried out so that the two principal moments of the set of points are both equal to unity. Thus, the set of points will form an approximately symmetric circular cloud of points of radius 1 about the origin. Experimental results given in [Hartley-97c] suggest that the extra effort required for non-isotropic scaling does not lead to significantly better results than isotropic scaling.

A further variant on scaling was discussed in [Muehlich-98], based on a statistical analysis of the estimator, its bias and variance. In that paper it was observed that some columns of $\mathtt{A}$ are not affected by noise. This applies to the third and sixth columns in (4.3–*p*89), corresponding to the entry $w_i w'_i = 1$. Such error-free entries in $\mathtt{A}$ should not be varied in finding $\hat{\mathtt{A}}$, the closest rank-deficient approximation to $\mathtt{A}$. A method known

Samsung/Google, EX1010, p. 127

as Total Least Squares - Fixed Columns is used to find the best solution. For estimation of the fundamental matrix (see chapter 11), [Muehlich-98] reports slightly improved results compared with non-isotropic scaling.

Scaling with points near infinity. Consider the case of estimation of a homography between an infinite plane and an image. If the viewing direction is sufficiently oblique, then very distant points in the plane may be visible in the image – even points at infinity (vanishing points) if the horizon is visible. In this case it makes no sense to normalize the coordinates of points in the infinite plane by setting the centroid at the origin, since the centroid may have very large coordinates, or be undefined. An approach to normalization in this case is considered in exercise (iii) on page 128.

4.5 Iterative minimization methods

This section describes methods for minimizing the various geometric cost functions developed in section 4.2 and section 4.3. Minimizing such cost functions requires the use of iterative techniques. This is unfortunate, because iterative techniques tend to have certain disadvantages compared to linear algorithms such as the normalized DLT algorithm 4.2:

(i) They are slower.
(ii) They generally need an initial estimate at which to start the iteration.
(iii) They risk not converging, or converging to a local minimum instead of the global minimum.
(iv) Selection of a stopping criterion for iteration may be tricky.

Consequently, iterative techniques generally require more careful implementation.

The technique of iterative minimization generally consists of five steps:

(i) **Cost function.** A cost function is chosen as the basis for minimization. Different possible cost functions were discussed in section 4.2.
(ii) **Parametrization.** The transformation (or other entity) to be computed is expressed in terms of a finite number of parameters. It is not in general necessary that this be a minimum set of parameters, and there are in fact often advantages to over-parametrization. (See the discussion below.)
(iii) **Function specification.** A function must be specified that expresses the cost in terms of the set of parameters.
(iv) **Initialization.** A suitable initial parameter estimate is computed. This will generally be done using a linear algorithm such as the DLT algorithm.
(v) **Iteration.** Starting from the initial solution, the parameters are iteratively refined with the goal of minimizing the cost function.

A word about parametrization

For a given cost function, there are often several choices of parametrization. The general strategy that guides parametrization is to select a set of parameters that cover the complete space over which one is minimizing, while at the same time allowing one to

Samsung/Google, EX1010, p. 128

compute the cost function in a convenient manner. For example, H may be parametrized by 9 parameters – that is, it is over-parametrized, since there are really only 8 degrees of freedom, overall scale not being significant. A *minimal* parametrization (i.e. the same number of parameters as degrees of freedom) would involve only 8 parameters.

In general no bad effects are likely to occur if a minimization problem of this type is over-parametrized, as long as for all choices of parameters the corresponding object is of the desired type. In particular for homogeneous objects, such as the 3×3 projection matrix encountered here, it is usually not necessary or advisable to attempt to use a minimal parametrization by removing the scale-factor ambiguity.

The reasoning is the following: it is not *necessary* to use minimal parametrization because a well-performing non-linear minimization algorithm will "notice" that it is not necessary to move in redundant directions, such as the matrix scaling direction. The algorithm described in Gill and Murray [Gill-78], which is a modification of the Gauss–Newton method, has an effective strategy for discarding redundant combinations of the parameters. Similarly, the Levenberg-Marquardt algorithm (see section A6.2(p600)) handles redundant parametrizations easily. It is not *advisable* because it is found empirically that the cost function surface is more complicated when minimal parametrizations are used. There is then a greater possibility of becoming stuck in a local minimum.

One other issue that arises in choosing a parametrization is that of restricting the transformation to a particular class. For example, suppose H is known to be a homology, then as described in section A7.2(p629) it may be parametrized as

$$\mathtt{H} = \mathtt{I} + (\mu - 1)\frac{\mathbf{v}\mathbf{a}^\top}{\mathbf{v}^\top\mathbf{a}}$$

where μ is a scalar, and $\mathbf{v}$ and $\mathbf{a}$ 3-vectors. A homology has 5 degrees of freedom which correspond here to the scalar μ and the directions of $\mathbf{v}$ and $\mathbf{a}$. If H is parametrized by its 9 matrix entries, then the estimated H is unlikely to exactly be a homology. However, if H is parametrized by μ, $\mathbf{v}$ and $\mathbf{a}$ (a total of 7 parameters) then the estimated H is guaranteed to be a homology. This parametrization is *consistent* with a homology (it is also an over-parametrization). We will return to the issues of consistent, local, minimal and over-parametrization in later chapters. The issues are also discussed further in appendix A6.9(p623).

Function specification

It has been seen in section 4.2.7 that a general class of estimation problems is concerned with a measurement space $\mathbb{R}^N$ containing a model surface S. Given a measurement $\mathbf{x} \in \mathbb{R}^N$ the estimation task is to find the point $\widehat{\mathbf{x}}$ lying on S closest to $\mathbf{x}$. In the case where a non-isotropic Gaussian error distribution is imposed on $\mathbb{R}^N$, the word *closest* is to be interpreted in terms of Mahalanobis distance. Iterative minimization methods will now be described in terms of this estimation model. In iterative estimation through parameter fitting, the model surface S is locally parametrized, and the parameters are allowed to vary to minimize the distance to the measured point. More specifically,

(i) One has a *measurement vector* $\mathbf{x} \in \mathbb{R}^N$ with covariance matrix Σ.

Samsung/Google, EX1010, p. 129

(ii) A set of parameters are represented as a vector $\mathbf{P} \in \mathbb{R}^M$.

(iii) A mapping $f : \mathbb{R}^M \to \mathbb{R}^N$ is defined. The range of this mapping is (at least locally) the model surface S in $\mathbb{R}^N$ representing the set of allowable measurements.

(iv) The cost function to be minimized is the squared Mahalanobis distance

$$\|\mathbf{X} - f(\mathbf{P})\|^2_\Sigma = (\mathbf{X} - f(\mathbf{P}))^\mathsf{T} \Sigma^{-1} (\mathbf{X} - f(\mathbf{P})).$$

In effect, we are attempting to find a set of parameters $\mathbf{P}$ such that $f(\mathbf{P}) = \mathbf{X}$, or failing that, to bring $f(\mathbf{P})$ as close to $\mathbf{X}$ as possible, with respect to Mahalanobis distance. The Levenberg–Marquardt algorithm is a general tool for iterative minimization, when the cost function to be minimized is of this type. We will now show how the various different types of cost functions described in this chapter fit into this format.

Error in one image. Here one fixes the coordinates of points $\mathbf{x}_i$ in the first image, and varies $\mathtt{H}$ so as to minimize cost function (4.6–*p*94), namely

$$\sum_i d(\mathbf{x}'_i, \mathtt{H}\bar{\mathbf{x}}_i)^2.$$

The measurement vector $\mathbf{X}$ is made up of the $2n$ inhomogeneous coordinates of the points $\mathbf{x}'_i$. One may choose as parameters the vector $\mathbf{h}$ of entries of the homography matrix $\mathtt{H}$. The function f is defined by

$$f : \mathbf{h} \mapsto (\mathtt{H}\mathbf{x}_1, \mathtt{H}\mathbf{x}_2, \ldots, \mathtt{H}\mathbf{x}_n)$$

where it is understood that here, and in the functions below, $\mathtt{H}\mathbf{x}_i$ indicates the inhomogeneous coordinates. One verifies that $\|\mathbf{X} - f(\mathbf{h})\|^2$ is equal to (4.6–*p*94).

Symmetric transfer error. In the case of the symmetric cost function (4.7–*p*95)

$$\sum_i d(\mathbf{x}_i, \mathtt{H}^{-1}\mathbf{x}'_i)^2 + d(\mathbf{x}'_i, \mathtt{H}\mathbf{x}_i)^2$$

one chooses as measurement vector $\mathbf{X}$ the $4n$-vector made up of the inhomogeneous coordinates of the points $\mathbf{x}_i$ followed by the inhomogeneous coordinates of the points $\mathbf{x}'_i$. The parameter vector as before is the vector $\mathbf{h}$ of entries of $\mathtt{H}$, and the function f is defined by

$$f : \mathbf{h} \mapsto (\mathtt{H}^{-1}\mathbf{x}'_1, \ldots, \mathtt{H}^{-1}\mathbf{x}'_n, \mathtt{H}\mathbf{x}_1, \ldots, \mathtt{H}\mathbf{x}_n).$$

As before, we find that $\|\mathbf{X} - f(\mathbf{h})\|^2$ is equal to (4.7–*p*95).

Reprojection error. Minimizing the cost function (4.8–*p*95) is more complex. The difficulty is that it requires a simultaneous minimization over all choices of points $\hat{\mathbf{x}}_i$ as well as the entries of the transformation matrix $\mathtt{H}$. If there are many point correspondences, then this becomes a very large minimization problem. Thus, the problem may be parametrized by the coordinates of the points $\hat{\mathbf{x}}_i$ and the entries of the matrix $\hat{\mathtt{H}}$ – a total of $2n + 9$ parameters. The coordinates of $\hat{\mathbf{x}}'_i$ are not required, since they are related to the other parameters by $\hat{\mathbf{x}}'_i = \hat{\mathtt{H}}\hat{\mathbf{x}}_i$. The parameter vector is therefore

Samsung/Google, EX1010, p. 130

$\mathtt{P} = (\mathbf{h}, \hat{\mathbf{x}}_1, \ldots, \hat{\mathbf{x}}_n)$. The measurement vector contains the inhomogeneous coordinates of all the points $\mathbf{x}_i$ and $\mathbf{x}'_i$. The function f is defined by

$$f : (\mathbf{h}, \hat{\mathbf{x}}_1, \ldots, \hat{\mathbf{x}}_n) \mapsto (\hat{\mathbf{x}}_1, \hat{\mathbf{x}}'_1, \ldots, \hat{\mathbf{x}}_n, \hat{\mathbf{x}}'_n)$$

where $\hat{\mathbf{x}}'_i = \mathtt{H}\hat{\mathbf{x}}_i$. One verifies that $\|\mathbf{X} - f(\mathbf{P})\|^2$, with $\mathbf{X}$ a $4n$-vector, is equal to the cost function (4.8–p95). This cost function must be minimized over all $2n + 9$ parameters.

Sampson approximation. In contrast with $2n + 9$ parameters of reprojection error, minimizing the error in one image (4.6–p94) or symmetric transfer error (4.7–p95) requires a minimization over the 9 entries of the matrix $\mathtt{H}$ only – in general a more tractable problem. The Sampson approximation to reprojection error enables reprojection error also to be minimized with only 9 parameters.

This is an important consideration, since the iterative solution of an m-parameter non-linear minimization problem using a method such as Levenberg–Marquardt involves the solution of an $m \times m$ set of linear equations at each iteration step. This is a problem with complexity $O(m^3)$. Hence, it is appropriate to keep the size of m low.

The Sampson error avoids minimizing over the $2n + 9$ parameters of reprojection error because effectively it determines the $2n$ variables $\{\hat{\mathbf{x}}_i\}$ for each particular choice of $\mathbf{h}$. Consequently the minimization then only requires the 9 parameters of $\mathbf{h}$. In practice this approximation gives excellent results provided the errors are small compared to the measurements.

Initialization

An initial estimate for the parametrization may be found by employing a linear technique. For example, the normalized DLT algorithm 4.2 directly provides $\mathtt{H}$ and thence the 9-vector $\mathbf{h}$ used to parametrize the iterative minimization. In general if there are $n \geq 4$ correspondences, then all will be used in the linear solution. However, as will be seen in section 4.7 on robust estimation, when the correspondences contain outliers it may be advisable to use a carefully selected minimal set of correspondences (i.e. four correspondences). Linear techniques or minimal solutions are the two initialization techniques recommended in this book.

An alternative method that is sometimes used (for instance see [Horn-90, Horn-91]) is to carry out a sufficiently dense sampling of parameter space, iterating from each sampled starting point and retaining the best result. This is only possible if the dimension of the parameter space is sufficiently small. Sampling of parameter space may be done either randomly, or else according to some pattern. Another initialization method is simply to do without any effective initialization at all, starting the iteration at a given fixed point in parameter space. This method is not often viable. Iteration is very likely to fall into a false minimum or not converge. Even in the best case, the number of iteration steps required will increase the further one starts from the final solution. For this reason using a good initialization method is the best plan.

Samsung/Google, EX1010, p. 131

Objective

Given $n > 4$ image point correspondences $\{\mathbf{x}_i \leftrightarrow \mathbf{x}'_i\}$, determine the Maximum Likelihood estimate $\hat{\mathtt{H}}$ of the homography mapping between the images.

The MLE involves also solving for a set of subsidiary points $\{\hat{\mathbf{x}}_i\}$, which minimize

$$\sum_i d(\mathbf{x}_i, \hat{\mathbf{x}}_i)^2 + d(\mathbf{x}'_i, \hat{\mathbf{x}}'_i)^2$$

where $\hat{\mathbf{x}}'_i = \hat{\mathtt{H}}\hat{\mathbf{x}}_i$.

Algorithm

(i) **Initialization**: Compute an initial estimate of $\hat{\mathtt{H}}$ to provide a starting point for the geometric minimization. For example, use the linear normalized DLT algorithm 4.2, or use RANSAC (section 4.7.1) to compute $\hat{\mathtt{H}}$ from four point correspondences.

(ii) **Geometric minimization of – either Sampson error:**

- Minimize the Sampson approximation to the geometric error (4.12–p99).
- The cost is minimized using the Newton algorithm of section A6.1(p597) or Levenberg–Marquardt algorithm of section A6.2(p600) over a suitable parametrization of $\hat{\mathtt{H}}$. For example the matrix may be parametrized by its 9 entries.

or Gold Standard error:

- Compute an initial estimate of the subsidiary variables $\{\hat{\mathbf{x}}_i\}$ using the measured points $\{\mathbf{x}_i\}$ or (better) the Sampson correction to these points given by (4.11–p99).
- Minimize the cost

$$\sum_i d(\mathbf{x}_i, \hat{\mathbf{x}}_i)^2 + d(\mathbf{x}'_i, \hat{\mathbf{x}}'_i)^2$$

over $\hat{\mathtt{H}}$ and $\hat{\mathbf{x}}_i$, $i = 1, \ldots, n$. The cost is minimized using the Levenberg–Marquardt algorithm over $2n+9$ variables: $2n$ for the n 2D points $\hat{\mathbf{x}}_i$, and 9 for the homography matrix $\hat{\mathtt{H}}$.

- If the number of points is large then the sparse method of minimizing this cost function given in section A6.4(p607) is the recommended approach.

Algorithm 4.3. *The Gold Standard algorithm and variations for estimating* $\mathtt{H}$ *from image correspondences. The Gold Standard algorithm is preferred to the Sampson method for 2D homography computation.*

Iteration methods

There are various iterative methods for minimizing the chosen cost function, of which the most popular are Newton iteration and the Levenberg–Marquardt method. These methods are described in appendix 6(p597). Other general methods for minimizing a cost function are available, such as Powell's method and the simplex method both described in [Press-88].

Summary. The ideas in this section are collected together in algorithm 4.3, which describes the Gold Standard and Sampson methods for estimating the homography mapping between point correspondences in two images.

Samsung/Google, EX1010, p. 132



a b c

Fig. 4.5. *Three images of a plane which are used to compare methods of computing projective transformations from corresponding points.*

Method	Pair 1 figure 4.5 a & b	Pair 2 figure 4.5 a & c
Linear normalized	0.4078	0.6602
Gold Standard	0.4078	0.6602
Linear unnormalized	0.4080	26.2056
Homogeneous scaling	0.5708	0.7421
Sampson	0.4077	0.6602
Error in 1 view	0.4077	0.6602
Affine	6.0095	2.8481
Theoretical optimal	0.5477	0.6582

Table 4.1. *Residual errors in pixels for the various algorithms.*

4.6 Experimental comparison of the algorithms

The algorithms are compared for the images shown in figure 4.5. Table 4.1 shows the results of testing several of the algorithms described in this chapter. Residual error is shown for two pairs of images. The methods used are fairly self-explanatory, with a few exceptions. The method "affine" was an attempt to fit the projective transformation with an optimal affine mapping. The "optimal" is the ML estimate assuming a noise level of one pixel.

The first pair of images are (a) and (b) of figure 4.5, with 55 point correspondences. It appears that all methods work almost equally well (except the affine method). The optimal residual is greater than the achieved results, because the noise level (unknown) is less than one pixel.

Image (c) of figure 4.5 was produced synthetically by resampling (a), and the second pair consists of (a) and (c) with 20 point correspondences. In this case, almost all methods perform almost optimally, as shown in the table 4.1. The exception is the affine method (expected to perform badly, since it is not an affine transformation) and the unnormalized linear method. The unnormalized method is expected to perform badly (though maybe not this badly). Just why it performs well in the first pair and very badly for the second pair is not understood. In any case, it is best to avoid this method and use a normalized linear or Gold Standard method.

A further evaluation is presented in figure 4.6. The transformation to be estimated is the one that maps the chessboard image shown here to a square grid aligned with the

Samsung/Google, EX1010, p. 133

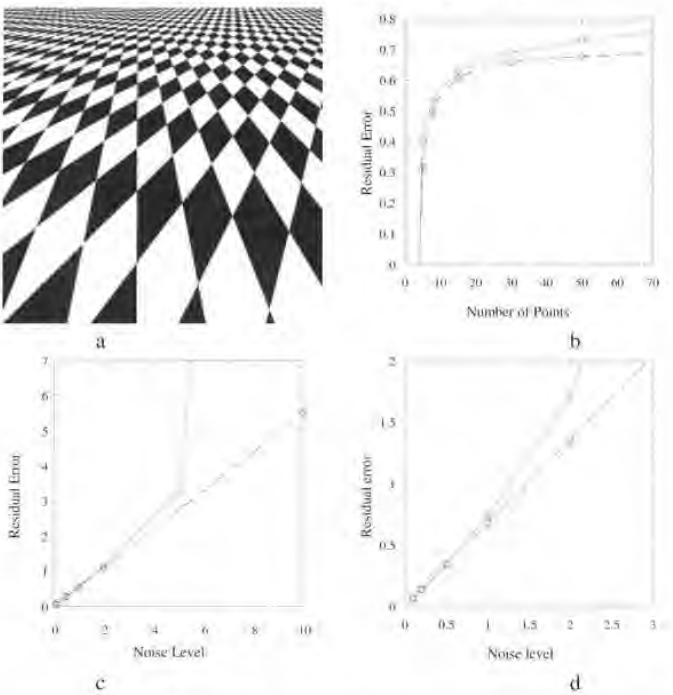


Fig. 4.6. **Comparison of the DLT and Gold Standard algorithms to the theoretically optimal residual error.** *(a) The homography is computed between a chessboard and this image. In all three graphs, the result for the Gold Standard algorithm overlap and are indistinguishable from the theoretical minimum. (b) Residual error as a function of the number of points. (c) The effect of varying noise level for 10 points, and (d) 50 points.*

axes. As may be seen, the image is substantially distorted, with respect to a square grid. For the experiments, randomly selected points in the image were chosen and matched with the corresponding point on the square grid. The (normalized) DLT algorithm and the Gold Standard algorithm are compared to the theoretical minimum or residual error (see chapter 5). Note that for noise up to 5 pixels, the DLT algorithm performs adequately. However, for a noise level of 10 pixels it fails. Note however that in a 200-pixel image, an error of 10 pixels is extremely high. For less severe homographies, closer to the identity map, the DLT performs almost as well as the Gold Standard algorithm.

4.7 Robust estimation

Up to this point it has been assumed that we have been presented with a set of correspondences, $\{\mathbf{x}_i \leftrightarrow \mathbf{x}'_i\}$, where the only source of error is in the measurement of the point's position, which follows a Gaussian distribution. In many practical situations this assumption is not valid because points are mismatched. The mismatched points are *outliers* to the Gaussian error distribution. These outliers can severely disturb the

Samsung/Google, EX1010, p. 134

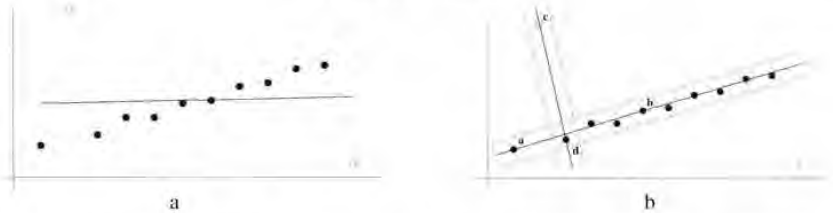


Fig. 4.7. **Robust line estimation.** *The solid points are inliers, the open points outliers. (a) A least-squares (orthogonal regression) fit to the point data is severely affected by the outliers. (b) In the RANSAC algorithm the support for lines through randomly selected point pairs is measured by the number of points within a threshold distance of the lines. The dotted lines indicate the threshold distance. For the lines shown the support is 10 for line* $\langle \mathbf{a}, \mathbf{b} \rangle$ *(where both of the points* $\mathbf{a}$ *and* $\mathbf{b}$ *are inliers); and 2 for line* $\langle \mathbf{c}, \mathbf{d} \rangle$ *where the point* $\mathbf{c}$ *is an outlier.*

estimated homography, and consequently should be identified. The goal then is to determine a set of *inliers* from the presented "correspondences" so that the homography can then be estimated in an optimal manner from these inliers using the algorithms described in the previous sections. This is *robust estimation* since the estimation is robust (tolerant) to outliers (measurements following a different, and possibly unmodelled, error distribution).

4.7.1 RANSAC

We start with a simple example that can easily be visualized – estimating a straight line fit to a set of 2-dimensional points. This can be thought of as estimating a 1-dimensional affine transformation, $x' = ax + b$, between corresponding points lying on two lines.

The problem, which is illustrated in figure 4.7a, is the following: given a set of 2D data points, find the line which minimizes the sum of squared perpendicular distances (orthogonal regression), subject to the condition that none of the valid points deviates from this line by more than t units. This is actually two problems: a line fit to the data; and a classification of the data into inliers (valid points) and outliers. The threshold t is set according to the measurement noise (for example $t = 3\sigma$), and is discussed below. There are many types of robust algorithms and which one to use depends to some extent on the proportion of outliers. For example, if it is known that there is only one outlier, then each point can be deleted in turn and the line estimated from the remainder. Here we describe in detail a general and very successful robust estimator – the RANdom SAmple Consensus (RANSAC) algorithm of Fischler and Bolles [Fischler-81]. The RANSAC algorithm is able to cope with a large proportion of outliers.

The idea is very simple: two of the points are selected randomly; these points define a line. The *support* for this line is measured by the number of points that lie within a distance threshold. This random selection is repeated a number of times and the line with most support is deemed the robust fit. The points within the threshold distance are the inliers (and constitute the eponymous *consensus* set). The intuition is that if one of the points is an outlier then the line will not gain much support, see figure 4.7b.

Samsung/Google, EX1010, p. 135

Furthermore, scoring a line by its support has the additional advantage of favouring better fits. For example, the line $\langle \mathbf{a}, \mathbf{b} \rangle$ in figure 4.7b has a support of 10, whereas the line $\langle \mathbf{a}, \mathbf{d} \rangle$, where the sample points are neighbours, has a support of only 4. Consequently, even though both samples contain no outliers, the line $\langle \mathbf{a}, \mathbf{b} \rangle$ will be selected.

More generally, we wish to fit a *model*, in this case a line, to data, and the random *sample* consists of a minimal subset of the data, in this case two points, sufficient to determine the model. If the model is a planar homography, and the data a set of 2D point correspondences, then the minimal subset consists of four correspondences. The application of RANSAC to the estimation of a homography is described below.

As stated by Fischler and Bolles [Fischler-81] "The RANSAC procedure is opposite to that of conventional smoothing techniques: Rather than using as much of the data as possible to obtain an initial solution and then attempting to eliminate the invalid data points, RANSAC uses as small an initial data set as feasible and enlarges this set with consistent data when possible".

The RANSAC algorithm is summarized in algorithm 4.4. Three questions immediately arise:

Objective

Robust fit of a model to a data set S which contains outliers.

Algorithm

(i) Randomly select a sample of s data points from S and instantiate the model from this subset.
(ii) Determine the set of data points S_i which are within a distance threshold t of the model. The set S_i is the consensus set of the sample and defines the inliers of S.
(iii) If the size of S_i (the number of inliers) is greater than some threshold T, re-estimate the model using all the points in S_i and terminate.
(iv) If the size of S_i is less than T, select a new subset and repeat the above.
(v) After N trials the largest consensus set S_i is selected, and the model is re-estimated using all the points in the subset S_i.

Algorithm 4.4. *The RANSAC robust estimation algorithm, adapted from [Fischler-81]. A minimum of s data points are required to instantiate the free parameters of the model. The three algorithm thresholds t, T, and N are discussed in the text.*

1. What is the distance threshold? We would like to choose the distance threshold, t, such that with a probability α the point is an inlier. This calculation requires the probability distribution for the distance of an inlier from the model. In practice the distance threshold is usually chosen empirically. However, if it is assumed that the measurement error is Gaussian with zero mean and standard deviation σ, then a value for t may be computed. In this case the square of the point distance, $d_\perp^2$, is a sum of squared Gaussian variables and follows a χ_m^2 distribution with m degrees of freedom, where m equals the codimension of the model. For a line the codimension is 1 – only the perpendicular distance to the line is measured. If the model is a point the codimension is 2, and the square of the distance is the sum of squared x and y measurement errors.

Samsung/Google, EX1010, p. 136

The probability that the value of a χ^2_m random variable is less than k^2 is given by the cumulative chi-squared distribution, $F_m(k^2) = \int_0^{k^2} \chi^2_m(\xi)d\xi$. Both of these distributions are described in section A2.2(*p*566). From the cumulative distribution

$$\begin{cases} \text{inlier} & d_\perp^2 < t^2 \\ \text{outlier} & d_\perp^2 \geq t^2 \end{cases} \quad \text{with } t^2 = F_m^{-1}(\alpha)\sigma^2. \tag{4.17}$$

Usually α is chosen as 0.95, so that there is a 95% probability that the point is an inlier. This means that an inlier will only be incorrectly rejected 5% of the time. Values of t for $\alpha = 0.95$ and for the models of interest in this book are tabulated in table 4.2.

Codimension m	Model	t^2
1	line, fundamental matrix	$3.84\,\sigma^2$
2	homography, camera matrix	$5.99\,\sigma^2$
3	trifocal tensor	$7.81\,\sigma^2$

Table 4.2. *The distance threshold $t^2 = F_m^{-1}(\alpha)\sigma^2$ for a probability of $\alpha = 0.95$ that the point (correspondence) is an inlier.*

2. How many samples? It is often computationally infeasible and unnecessary to try every possible sample. Instead the number of samples N is chosen sufficiently high to ensure with a probability, p, that at least one of the random samples of s points is free from outliers. Usually p is chosen at 0.99. Suppose w is the probability that any selected data point is an inlier, and thus $\epsilon = 1 - w$ is the probability that it is an outlier. Then at least N selections (each of s points) are required, where $(1 - w^s)^N = 1 - p$, so that

$$N = \log(1 - p)/\log(1 - (1 - \epsilon)^s). \tag{4.18}$$

Table 4.3 gives examples of N for $p = 0.99$ for a given s and ϵ.

Sample size	Proportion of outliers ϵ						
s	5%	10%	20%	25%	30%	40%	50%
2	2	3	5	6	7	11	17
3	3	4	7	9	11	19	35
4	3	5	9	13	17	34	72
5	4	6	12	17	26	57	146
6	4	7	16	24	37	97	293
7	4	8	20	33	54	163	588
8	5	9	26	44	78	272	1177

Table 4.3. *The number N of samples required to ensure, with a probability $p = 0.99$, that at least one sample has no outliers for a given size of sample, s, and proportion of outliers, ϵ.*

Example 4.4. For the line-fitting problem of figure 4.7 there are $n = 12$ data points, of

Samsung/Google, EX1010, p. 137

which two are outliers so that $\epsilon = 2/12 = 1/6$. From table 4.3 for a minimal subset of size $s = 2$, at least $N = 5$ samples are required. This should be compared with the cost of exhaustively trying every point pair, in which case $\binom{12}{2} = 66$ samples are required (the notation $\binom{n}{2}$ means the number of choices of 2 among n, specifically, $\binom{n}{2} = n(n-1)/2$). △

Note

(i) The number of samples is linked to the proportion rather than number of outliers. This means that the number of samples required may be smaller than the number of outliers. Consequently the computational cost of the sampling can be acceptable even when the number of outliers is large.

(ii) The number of samples increases with the size of the minimal subset (for a given ϵ and p). It might be thought that it would be advantageous to use more than the minimal subset, three or more points in the case of a line, because then a better estimate of the line would be obtained, and the measured support would more accurately reflect the true support. However, this possible advantage in measuring support is generally outweighed by the severe increase in computational cost incurred by the increase in the number of samples.

3. How large is an acceptable consensus set? A rule of thumb is to terminate if the size of the consensus set is similar to the number of inliers believed to be in the data set, given the assumed proportion of outliers, i.e. for n data points $T = (1-\epsilon)n$. For the line-fitting example of figure 4.7 a conservative estimate of ϵ is $\epsilon = 0.2$, so that $T = (1.0 - 0.2)12 = 10$.

Determining the number of samples adaptively. It is often the case that ϵ, the fraction of data consisting of outliers, is unknown. In such cases the algorithm is initialized using a worst case estimate of ϵ, and this estimate can then be updated as larger consistent sets are found. For example, if the worst case guess is $\epsilon = 0.5$ and a consensus set with 80% of the data is found as inliers, then the updated estimate is $\epsilon = 0.2$.

This idea of "probing" the data via the consensus sets can be applied repeatedly in order to adaptively determine the number of samples, N. To continue the example above, the worst case estimate of $\epsilon = 0.5$ determines an initial N according to (4.18). When a consensus set containing more than 50% of the data is found, we then know that there is at least that proportion of inliers. This updated estimate of ϵ determines a reduced N from (4.18). This update is repeated at each sample, and whenever a consensus set with ϵ lower than the current estimate is found, then N is again reduced. The algorithm terminates as soon as N samples have been performed. It may occur that a sample is found for which ϵ determines an N less than the number of samples that have already been performed. In such a case sufficient samples have been performed and the algorithm terminates. In pseudo-code the adaptive computation of N is summarized in algorithm 4.5.

This adaptive approach works very well and in practice covers the questions of both

Samsung/Google, EX1010, p. 138

- $N = \infty$, sample_count$= 0$.
- While $N >$ sample_count Repeat
  - Choose a sample and count the number of inliers.
  - Set $\epsilon = 1 - (\text{number of inliers})/(\text{total number of points})$
  - Set N from ϵ and (4.18) with $p = 0.99$.
  - Increment the sample_count by 1.
- Terminate.

Algorithm 4.5. *Adaptive algorithm for determining the number of RANSAC samples.*

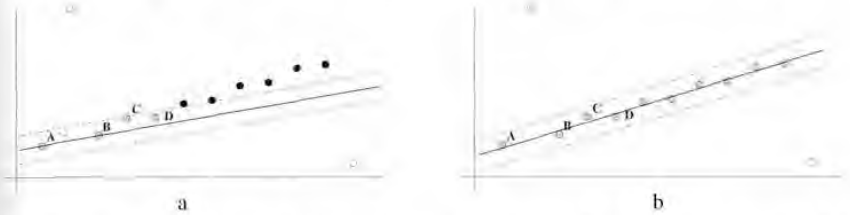


Fig. 4.8. **Robust ML estimation.** *The grey points are classified as inliers to the line. (a) A line defined by points $\langle \mathbf{A}, \mathbf{B} \rangle$ has a support of four (from points $\{\mathbf{A}, \mathbf{B}, \mathbf{C}, \mathbf{D}\}$). (b) The ML line fit (orthogonal least-squares) to the four points. This is a much improved fit over that defined by $\langle \mathbf{A}, \mathbf{B} \rangle$. 10 points are classified as inliers.*

the number of samples and terminating the algorithm. The initial ϵ can be chosen as 1.0, in which case the initial N will be infinite. It is wise to use a conservative probability p such as 0.99 in (4.18). Table 4.4 on page 127 gives example ϵ's and N's when computing a homography.

4.7.2 Robust Maximum Likelihood estimation

The RANSAC algorithm partitions the data set into inliers (the largest consensus set) and outliers (the rest of the data set), and also delivers an estimate of the model, M_0, computed from the minimal set with greatest support. The final step of the RANSAC algorithm is to re-estimate the model using all the inliers. This re-estimation should be optimal and will involve minimizing a ML cost function, as described in section 4.3. In the case of a line, ML estimation is equivalent to orthogonal regression, and a closed form solution is available. In general, though, the ML estimation involves iterative minimization, and the minimal set estimate, M_0, provides the starting point.

The only drawback with this procedure, which is often the one adopted, is that the inlier–outlier classification is irrevocable. After the model has been optimally fitted to the consensus set, there may well be additional points which would now be classified as inliers if the distance threshold was applied to the new model. For example, suppose the line $\langle \mathbf{A}, \mathbf{B} \rangle$ in figure 4.8 was selected by RANSAC. This line has a support of four points, all inliers. After the optimal fit to these four points, there are now 10 points which would correctly be classified as inliers. These two steps: optimal fit to inliers; re-classify inliers using (4.17); can then be iterated until the number of inliers converges.

Samsung/Google, EX1010, p. 139

A least-squares fit with inliers weighted by their distance to the model is often used at this stage.

Robust cost function. An alternative to minimizing $\mathcal{C} = \sum_i d_{\perp i}^2$ over the inliers is to minimize a robust version including all data. A suitable robust cost function is

$$\mathcal{D} = \sum_i \gamma\left(d_{\perp i}\right) \quad \text{with } \gamma(e) = \begin{cases} e^2 & e^2 < t^2 \quad \text{inlier} \\ t^2 & e^2 \geq t^2 \quad \text{outlier} \end{cases} \tag{4.19}$$

Here $d_{\perp i}$ are point errors and $\gamma(e)$ is a robust cost function [Huber-81] where outliers are given a fixed cost. The χ^2 motivation for the threshold is the same as that of (4.17), where t^2 is defined. The quadratic cost for inliers arises from the Gaussian error model, as described in section 4.3. The constant cost for outliers in the robust cost function arises from the assumption that outliers follow a diffuse or uniform distribution, the log-likelihood of which is a constant. It might be thought that outliers could be excluded from the cost function by simply thresholding on $d_{\perp i}$. The problem with thresholding alone is that it would result in only outliers being included because they would incur no cost.

The cost function $\mathcal{D}$ allows the minimization to be conducted on all points whether they are outliers or inliers. At the start of the iterative minimization $\mathcal{D}$ differs from $\mathcal{C}$ only by a constant (given by 4 times the number of outliers). However, as the minimization progresses outliers can be redesignated inliers, and this typically occurs in practice. A discussion and comparison of cost functions is given in appendix A6.8-(*p*616).

4.7.3 Other robust algorithms

In RANSAC a model instantiated from a minimal set is scored by the number of data points within a threshold distance. An alternative is to score the model by the median of the distances to all points in the data. The model with least median is then selected. This is Least Median of Squares (LMS) estimation, where, as in RANSAC, minimum size subset samples are selected randomly with the number of samples obtained from (4.18). The advantage of LMS is that it requires *no* setting of thresholds or *a priori* knowledge of the variance of the error. The disadvantage of LMS is that it fails if more than half the data is outlying, for then the median distance will be to an outlier. The solution is to use the proportion of outliers to determine the selection distance. For example if there are 50% outliers then a distance below the median value (the quartile say) should be used.

Both the RANSAC and LMS algorithms are able to cope with a large proportion of outliers. If the number of outliers is small, then other robust methods may well be more efficient. These include case deletion, where each point in turn is deleted and the model fitted to the remaining data; and iterative weighted least-squares, where a data point's influence on the fit is weighted inversely by its residual. Generally these methods are **not** recommended. Both Torr [Torr-95b] and Xu and Zhang [Xu-96] describe and compare various robust estimators for estimating the fundamental matrix.

Samsung/Google, EX1010, p. 140

Objective

Compute the 2D homography between two images.

Algorithm

(i) **Interest points:** Compute interest points in each image.
(ii) **Putative correspondences:** Compute a set of interest point matches based on proximity and similarity of their intensity neighbourhood.
(iii) **RANSAC robust estimation:** Repeat for N samples, where N is determined adaptively as in algorithm 4.5:
  (a) Select a random sample of 4 correspondences and compute the homography H.
  (b) Calculate the distance $d_\perp$ for each putative correspondence.
  (c) Compute the number of inliers consistent with H by the number of correspondences for which $d_\perp < t = \sqrt{5.99}\sigma$ pixels.

  Choose the H with the largest number of inliers. In the case of ties choose the solution that has the lowest standard deviation of inliers.
(iv) **Optimal estimation:** re-estimate H from all correspondences classified as inliers, by minimizing the ML cost function (4.8–p95) using the Levenberg–Marquardt algorithm of section A6.2(p600).
(v) **Guided matching:** Further interest point correspondences are now determined using the estimated H to define a search region about the transferred point position.

The last two steps can be iterated until the number of correspondences is stable.

Algorithm 4.6. *Automatic estimation of a homography between two images using RANSAC.*

4.8 Automatic computation of a homography

This section describes an algorithm to automatically compute a homography between two images. The input to the algorithm is simply the images, with no other *a priori* information required; and the output is the estimated homography together with a set of interest points in correspondence. The algorithm might be applied, for example, to two images of a planar surface or two images acquired by rotating a camera about its centre.

The first step of the algorithm is to compute interest points in each image. We are then faced with a "chicken and egg" problem: once the correspondence between the interest points is established the homography can be computed; conversely, given the homography the correspondence between the interest points can easily be established. This problem is resolved by using robust estimation, here RANSAC, as a "search engine". The idea is first to obtain by some means a set of putative point correspondences. It is expected that a proportion of these correspondences will in fact be mismatches. RANSAC is designed to deal with exactly this situation – estimate the homography and also a set of inliers consistent with this estimate (the true correspondences), and outliers (the mismatches).

The algorithm is summarized in algorithm 4.6, with an example of its use shown in figure 4.9, and the steps described in more detail below. Algorithms with essentially the same methodology enable the automatic computation of the fundamental matrix and trifocal tensor directly from image pairs and triplets respectively. This computation is described in chapter 11 and chapter 16.

Samsung/Google, EX1010, p. 141

Determining putative correspondences. The aim, in the absence of any knowledge of the homography, is to provide an initial point correspondence set. A good proportion of these correspondences should be correct, but the aim is not perfect matching, since RANSAC will later be used to eliminate the mismatches. Think of these as "seed" correspondences. These putative correspondences are obtained by detecting interest points independently in each image, and then matching these interest points using a combination of proximity and similarity of intensity neighbourhoods as follows. For brevity, the interest points will be referred to as 'corners'. However, these corners need not be images of physical corners in the scene. The corners are defined by a minimum of the image auto-correlation function.

For each corner at (x, y) in image 1 the match with highest neighbourhood cross-correlation in image 2 is selected within a square search region centred on (x, y). Symmetrically, for each corner in image 2 the match is sought in image 1. Occasionally there will be a conflict where a corner in one image is "claimed" by more than one corner in the other. In such cases a "winner takes all" scheme is applied and only the match with highest cross-correlation is retained.

A variation on the similarity measure is to use Squared Sum of intensity Differences (SSD) instead of (normalized) Cross-Correlation (CC). CC is invariant to the affine mapping of the intensity values (i.e. $I \mapsto \alpha I + \beta$, scaling plus offset) which often occurs in practice between images. SSD is not invariant to this mapping. However, SSD is often preferred when there is small variation in intensity between images, because it is a more sensitive measure than CC and is computationally cheaper.

RANSAC for a homography. The RANSAC algorithm is applied to the putative correspondence set to estimate the homography and the (inlier) correspondences which are consistent with this estimate. The sample size is four, since four correspondences determine a homography. The number of samples is set adaptively as the proportion of outliers is determined from each consensus set, as described in algorithm 4.5.

There are two issues: what is the "distance" in this case? and how should the samples be selected?

(i) **Distance measure**: The simplest method of assessing the error of a correspondence from a homography $\mathtt{H}$ is to use the symmetric transfer error, i.e. $d^2_{\text{transfer}} = d(\mathbf{x}, \mathtt{H}^{-1}\mathbf{x}')^2 + d(\mathbf{x}', \mathtt{H}\mathbf{x})^2$, where $\mathbf{x} \leftrightarrow \mathbf{x}'$ is the point correspondence. A better, though more expensive, distance measure is the reprojection error, $d^2_{\perp} = d(\mathbf{x}, \hat{\mathbf{x}})^2 + d(\mathbf{x}', \hat{\mathbf{x}}')^2$, where $\hat{\mathbf{x}}' = \mathtt{H}\hat{\mathbf{x}}$ is the perfect correspondence. This measure is more expensive because $\hat{\mathbf{x}}$ must also be computed. A further alternative is Sampson error.

(ii) **Sample selection**: There are two issues here. First, degenerate samples should be disregarded. For example, if three of the four points are collinear then a homography cannot be computed; second, the sample should consist of points with a good spatial distribution over the image. This is because of the extrapolation problem – an estimated homography will accurately map the region straddled by the computation points, but the accuracy generally deteriorates

Samsung/Google, EX1010, p. 142

with distance from this region (think of four points in the very top corner of the image). Distributed spatial sampling can be implemented by tiling the image and ensuring, by a suitable weighting of the random sampler, that samples with points lying in different tiles are the more likely.

Robust ML estimation and guided matching. The aim of this final stage is two-fold: first, to obtain an improved estimate of the homography by using all the inliers in the estimation (rather than only the four points of the sample); second, to obtain more inlying matches from the putative correspondence set because a more accurate homography is available. An improved estimate of the homography is then computed from the inliers by minimizing an ML cost function. This final stage can be implemented in two ways. One way is to carry out an ML estimation on the inliers, then recompute the inliers using the new estimated $\mathtt{H}$, and repeat this cycle until the number of inliers converges. The ML cost function minimization is carried out using the Levenberg–Marquardt algorithm described in section A6.2($p600$). The alternative is to estimate the homography and inliers simultaneously by minimizing a robust ML cost function of (4.19) as described in section 4.7.2. The disadvantage of the simultaneous approach is the computational effort incurred in the minimization of the cost function. For this reason the cycle approach is usually the more attractive.

4.8.1 Application domain

The algorithm requires that interest points can be recovered fairly uniformly across the image, and this in turn requires scenes and resolutions which support this requirement. Scenes should be lightly textured – images of blank walls are not ideal.

The search window proximity constraint places an upper limit on the image motion of corners (the *disparity*) between views. However, the algorithm is not defeated if this constraint is not applied, and in practice the main role of the proximity constraint is to reduce computational complexity, as a smaller search window means that fewer corner matches must be evaluated.

Ultimately the scope of the algorithm is limited by the success of the corner neighbourhood similarity measure (SSD or CC) in providing disambiguation between correspondences. Failure generally results from lack of spatial invariance: the measures are only invariant to image translation, and are severely degraded by transformations outside this class such as image rotation or significant differences in foreshortening between images. One solution is to use measures with a greater invariance to the homography mapping between images, for example measures which are rotationally invariant. An alternative solution is to use an initial estimate of the homography to map between intensity neighbourhoods. Details are beyond the scope of this discussion, but are provided in [Pritchett-98, Schmid-98]. The use of robust estimation confers moderate immunity to independent motion, changes in shadows, partial occlusions etc.

Samsung/Google, EX1010, p. 143



Fig. 4.9. **Automatic computation of a homography between two images using RANSAC.** *The motion between views is a rotation about the camera centre so the images are exactly related by a homography. (a) (b) left and right images of Keble College, Oxford. The images are* 640×480 *pixels. (c) (d) detected corners superimposed on the images. There are approximately 500 corners on each image. The following results are superimposed on the left image: (e) 268 putative matches shown by the line linking corners, note the clear mismatches; (f) outliers – 117 of the putative matches; (g) inliers – 151 correspondences consistent with the estimated* $\mathbf{H}$*; (h) final set of 262 correspondences after guided matching and MLE.*

Samsung/Google, EX1010, p. 144

4.8.2 Implementation and run details

Interest points are obtained using the Harris [Harris-88] corner detector. This detector localizes corners to sub-pixel accuracy, and it has been found empirically that the correspondence error is usually less than a pixel [Schmid-98].

When obtaining seed correspondences, in the putative correspondence stage of the algorithm, the threshold on the neighbourhood similarity measure for match acceptance is deliberately conservative to minimize incorrect matches (the SSD threshold is 20). For the guided matching stage this threshold is relaxed (it is doubled) so that additional putative correspondences are available.

Number of inliers	$1-\epsilon$	Adaptive N
6	2%	20,028,244
10	3%	2,595,658
44	16%	6,922
58	21%	2,291
73	26%	911
151	56%	43

Table 4.4. *The results of the adaptive algorithm 4.5 used during RANSAC to compute the homography for figure 4.9. N is the total number of samples required as the algorithm runs for $p = 0.99$ probability of no outliers in the sample. The algorithm terminated after 43 samples.*

For the example of figure 4.9 the images are 640×480 pixels, and the search window ± 320 pixels, i.e. the entire image. Of course a much smaller search window could have been used given the actual point disparities in this case. Often in video sequences a search window of ± 40 pixels suffices (i.e. a square of side 80 centred on the current position). The inlier threshold was $t = 1.25$ pixels.

A total of 43 samples were required, with the sampling run as shown in table 4.4. The guided matching required two iterations of the MLE–inlier classification cycle. The RMS values for $d_\perp$ pixel error were 0.23 before the MLE and 0.19 after. The Levenberg–Marquardt algorithm required 10 iterations.

4.9 Closure

This chapter has illustrated the issues and techniques that apply to estimating the tensors representing multiple view relations. These ideas will reoccur in each of the computation chapters throughout the book. In each case there are a minimal number of correspondences required; degenerate configurations that should be avoided; algebraic and geometric errors that can be minimized when more than the minimal number of correspondences are available; parametrizations that enforce internal constraints on the tensor etc.

4.9.1 The literature

The DLT algorithm dates back at least to Sutherland [Sutherland-63]. Sampson's classic paper on conic fitting (an improvement on the equally classic Bookstein algorithm)

Samsung/Google, EX1010, p. 145

appeared in [Sampson-82]. Normalization was made public in the Computer Vision literature by Hartley [Hartley-97c].

Related reading on numerical methods may be found in the excellent *Numerical Recipes in C* [Press-88], and also Gill and Murray [Gill-78] for iterative minimization.

Fischler and Bolles' [Fischler-81] RANSAC was one of the earliest robust algorithms, and in fact was developed to solve a Computer Vision problem (pose from 3 points). The original paper is very clearly argued and well worth reading. Other background material on robustness may be found in Rousseeuw [Rousseeuw-87]. The primary application of robust estimation in computer vision was to estimating the fundamental matrix (chapter 11), by Torr and Murray [Torr-93] using RANSAC, and, Zhang *et al.* [Zhang-95] using LMS. The automatic ML estimation of a homography was described by Torr and Zisserman [Torr-98].

4.9.2 Notes and exercises

(i) **Computing homographies of $\mathbb{P}^n$.** The derivation of (4.1–*p*89) and (4.3–*p*89) assumed that the dimension of $\mathbf{x}'_i$ is three, so that the cross-product is defined. However, (4.3) may be derived in a way that generalizes to all dimensions. Assuming that $w'_i = 1$, we may solve for the unknown scale factor explicitly by writing $\mathtt{H}\mathbf{x}_i = k(x_i, y_i, 1)^\mathsf{T}$. From the third coordinate we obtain $k = \mathbf{h}^{3\mathsf{T}}\mathbf{x}_i$, and substituting this into the original equation gives

$$\begin{pmatrix} \mathbf{h}^{1\mathsf{T}}\mathbf{x}_i \\ \mathbf{h}^{2\mathsf{T}}\mathbf{x}_i \end{pmatrix} = \begin{pmatrix} x'_i\mathbf{h}^{3\mathsf{T}}\mathbf{x}_i \\ y'_i\mathbf{h}^{3\mathsf{T}}\mathbf{x}_i \end{pmatrix}$$

which leads directly to (4.3).

(ii) **Computing homographies for ideal points.** If one of the points $\mathbf{x}'_i$ is an ideal point, so that $w'_i = 0$, then the pair of equations (4.3) collapses to a single equation although (4.1) does contain two independent equations. To avoid such degeneracy, while including only the minimum number of equations, a good way to proceed is as follows. We may rewrite the equation $\mathbf{x}'_i = \mathtt{H}\mathbf{x}_i$ as

$$[\mathbf{x}'_i]^\perp \mathtt{H}\mathbf{x}_i = \mathbf{0}$$

where $[\mathbf{x}'_i]^\perp$ is a matrix with rows orthogonal to $\mathbf{x}'_i$ so that $[\mathbf{x}'_i]^\perp\mathbf{x}'_i = \mathbf{0}$. Each row of $[\mathbf{x}'_i]^\perp$ leads to a separate linear equation in the entries of $\mathtt{H}$. The matrix $[\mathbf{x}'_i]^\perp$ may be obtained by deleting the first row of an orthogonal matrix $\mathtt{M}$ satisfying $\mathtt{M}\mathbf{x}'_i = (1, 0, \ldots, 0)^\mathsf{T}$. A Householder matrix (see section A4.1.2(*p*580)) is an easily constructed matrix with the desired property.

(iii) **Scaling unbounded point sets.** In the case of points at or near infinity in a plane, it is neither reasonable nor feasible to normalize coordinates using the isotropic (or non-isotropic) scaling schemes presented in this chapter, since the centroid and scale are infinite or near infinite. A method that seems to give good results is to normalize the set of points $\mathbf{x}_i = (x_i, y_i, w_i)^\mathsf{T}$ such that

$$\sum_i x_i = \sum_i y_i = 0 \;:\; \sum_i x_i^2 + y_i^2 = 2\sum_i w_i^2 \;:\; x_i^2 + y_i^2 + w_i^2 = 1 \forall i$$

Samsung/Google, EX1010, p. 146

Note that the coordinates x_i and y_i appearing here are the homogeneous coordinates, and the conditions no longer imply that the centroid is at the origin. Investigate methods of achieving this normalization, and evaluate its properties.

(iv) **Transformation invariance of DLT.** We consider computation of a 2D homography by minimizing algebraic error $\|\mathtt{A}\mathbf{h}\|$ (see (4.5–p94)) subject to various constraints. Prove the following cases:

(a) If $\|\mathtt{A}\mathbf{h}\|$ is minimized subject to the constraint $h_9 = \mathtt{H}_{33} = 1$, then the result is invariant under change of scale but *not* translation of coordinates.

(b) If instead the constraint is $\mathtt{H}_{31}^2 + \mathtt{H}_{32}^2 = 1$ then the result is similarity invariant.

(c) **Affine case:** The same is true for the constraint $\mathtt{H}_{31} = \mathtt{H}_{32} = 0$: $\mathtt{H}_{33} = 1$.

(v) **Expressions for image coordinate derivatives.** For the map $\mathbf{x}' = (x', y', w')^\mathsf{T} = \mathtt{H}\mathbf{x}$, derive the following expressions (where $\tilde{\mathbf{x}}' = (\tilde{x}', \tilde{y}')^\mathsf{T} = (x'/w', y'/w')^\mathsf{T}$ are the inhomogeneous coordinates of the image point):

(a) Derivative wrt $\mathbf{x}$

$$\partial\tilde{\mathbf{x}}'/\partial\mathbf{x} = \frac{1}{w'}\begin{bmatrix} \mathbf{h}^{1\mathsf{T}} - \tilde{x}'\mathbf{h}^{3\mathsf{T}} \\ \mathbf{h}^{2\mathsf{T}} - \tilde{y}'\mathbf{h}^{3\mathsf{T}} \end{bmatrix} \tag{4.20}$$

where $\mathbf{h}^{j\mathsf{T}}$ is the $j-$th row of $\mathtt{H}$.

(b) Derivative wrt $\mathtt{H}$

$$\partial\tilde{\mathbf{x}}'/\partial\mathbf{h} = \frac{1}{w'}\begin{bmatrix} \mathbf{x}^\mathsf{T} & 0 & -\tilde{x}'\mathbf{x}^\mathsf{T} \\ 0 & \mathbf{x}^\mathsf{T} & -\tilde{y}'\mathbf{x}^\mathsf{T} \end{bmatrix} \tag{4.21}$$

with $\mathbf{h}$ as defined in (4.2–p89).

(vi) **Sampson error with non-isotropic error distributions.** The derivation of Sampson error in section 4.2.6(p98) assumed that points were measured with circular error distributions. In the case where the point $\mathbf{X} = (x, y, x', y')$ is measured with covariance matrix $\Sigma_\mathbf{X}$ it is appropriate instead to minimize the Mahalanobis norm $\|\boldsymbol{\delta}_\mathbf{X}\|^2_{\Sigma_\mathbf{X}} = \boldsymbol{\delta}_\mathbf{X}^\mathsf{T}\Sigma_\mathbf{X}^{-1}\boldsymbol{\delta}_\mathbf{X}$. Show that in this case the formulae corresponding to (4.11–p99) and (4.12–p99) are

$$\boldsymbol{\delta}_\mathbf{X} = -\Sigma_\mathbf{X}\mathtt{J}^\mathsf{T}(\mathtt{J}\Sigma_\mathbf{X}\mathtt{J}^\mathsf{T})^{-1}\boldsymbol{\epsilon} \tag{4.22}$$

and

$$\|\boldsymbol{\delta}_\mathbf{X}\|^2_{\Sigma_\mathbf{X}} = \boldsymbol{\epsilon}^\mathsf{T}(\mathtt{J}\Sigma_\mathbf{X}J^\mathsf{T})^{-1}\boldsymbol{\epsilon}. \tag{4.23}$$

Note that if the measurements in the two images are independent, then the covariance matrix $\Sigma_\mathbf{X}$ will be block-diagonal with two 2×2 diagonal blocks corresponding to the two images.

(vii) **Sampson error programming hint.** In the case of 2D homography estimation, and in fact every other similar problem considered in this book, the cost function $C_\mathtt{H}(\mathbf{X}) = \mathtt{A}(\mathbf{X})\mathbf{h}$ of section 4.2.6(p98) is multilinear in the coordinates

Samsung/Google, EX1010, p. 147

Objective

Given $n \geq 4$ image point correspondences $\{\mathbf{x}_i \leftrightarrow \mathbf{x}'_i\}$, determine the affine homography $\mathtt{H}_{\mathtt{A}}$ which minimizes reprojection error in both images (4.8–*p*95).

Algorithm

(a) Express points as inhomogeneous 2-vectors. Translate the points $\mathbf{x}_i$ by a translation $\mathbf{t}$ so that their centroid is at the origin. Do the same to the points $\mathbf{x}'_i$ by a translation $\mathbf{t}'$. Henceforth work with the translated coordinates.

(b) Form the $n \times 4$ matrix $\mathtt{A}$ whose rows are the vectors

$$\mathbf{X}_i^\top = (\mathbf{x}_i^\top, \mathbf{x}_i'^\top) = (x_i, y_i, x'_i, y'_i).$$

(c) Let $\mathbf{V}_1$ and $\mathbf{V}_2$ be the right singular-vectors of $\mathtt{A}$ corresponding to the two largest (sic) singular values.

(d) Let $\mathtt{H}_{2\times 2} = \mathtt{C}\mathtt{B}^{-1}$, where $\mathtt{B}$ and $\mathtt{C}$ are the 2×2 blocks such that

$$[\mathbf{V}_1 \mathbf{V}_2] = \begin{bmatrix} \mathtt{B} \\ \mathtt{C} \end{bmatrix}.$$

(e) The required homography is

$$\mathtt{H}_{\mathtt{A}} = \begin{bmatrix} \mathtt{H}_{2\times 2} & \mathtt{H}_{2\times 2}\mathbf{t} - \mathbf{t}' \\ \mathbf{0}^\top & 1 \end{bmatrix},$$

and the corresponding estimate of the image points is given by

$$\widehat{\mathbf{X}}_i = (\mathbf{V}_1 \mathbf{V}_1^\top + \mathbf{V}_2 \mathbf{V}_2^\top)\mathbf{X}_i$$

Algorithm 4.7. *The Gold Standard Algorithm for estimating an* affine *homography* $\mathtt{H}_{\mathtt{A}}$ *from image correspondences.*

of $\mathbf{X}$. This means that the partial derivative $\partial \mathcal{C}_{\mathtt{H}}(\mathbf{X})/\partial \mathbf{X}$ may be very simply computed. For instance, the derivative

$$\partial \mathcal{C}_{\mathtt{H}}(x, y, x', y')/\partial x = \mathcal{C}_{\mathtt{H}}(x+1, y, x', y') - \mathcal{C}_{\mathtt{H}}(x, y, x', y')$$

is *exact*, not a finite difference approximation. This means that for programming purposes, one does not need to code a special routine for taking derivatives – the routine for computing $\mathcal{C}_{\mathtt{H}}(\mathbf{X})$ will suffice. Denoting by $\mathbf{E}_i$ the vector containing 1 in the i-th position, and otherwise 0, one sees that $\partial \mathcal{C}_{\mathtt{H}}(\mathbf{X})/\partial X_i = \mathcal{C}_{\mathtt{H}}(\mathbf{X} + \mathbf{E}_i) - \mathcal{C}_{\mathtt{H}}(\mathbf{X})$, and further

$$\mathtt{J}\mathtt{J}^\top = \sum_i \left(\mathcal{C}_{\mathtt{H}}(\mathbf{X} + \mathbf{E}_i) - \mathcal{C}_{\mathtt{H}}(\mathbf{X})\right)\left(\mathcal{C}_{\mathtt{H}}(\mathbf{X} + \mathbf{E}_i) - \mathcal{C}_{\mathtt{H}}(\mathbf{X})\right)^\top.$$

Also note that computationally it is more efficient to solve $\mathtt{J}\mathtt{J}^\top \boldsymbol{\lambda} = -\boldsymbol{\epsilon}$ directly for $\boldsymbol{\lambda}$, rather than take the inverse as $\boldsymbol{\lambda} = -(\mathtt{J}\mathtt{J}^\top)^{-1}\boldsymbol{\epsilon}$.

(viii) **Minimizing geometric error for affine transformations.** Given a set of correspondences $(x_i, y_i) \leftrightarrow (x'_i, y'_i)$, find an affine transformation $\mathtt{H}_{\mathtt{A}}$ that minimizes geometric error (4.8–*p*95). We will step through the derivation of a linear algorithm based on Sampson's approximation which is exact in this case. The complete method is summarized in algorithm 4.7.

Samsung/Google, EX1010, p. 148

(a) Show that the optimum affine transformation takes the centroid of the $\mathbf{x}_i$ to the centroid of $\mathbf{x}'_i$, so by translating the points to have their centroid at the origin, the translation part of the transformation is determined. It is only necessary then to determine the upper-left 2×2 submatrix $\mathtt{H}_{2\times 2}$ of $\mathtt{H}_A$, which represents the linear part of the transformation.

(b) The point $\mathbf{X}_i = (\mathbf{x}_i^\mathsf{T}, \mathbf{x}_i'^\mathsf{T})^\mathsf{T}$ lies on $\mathcal{V}_\mathtt{H}$ if and only if $[\mathtt{H}_{2\times 2} | -\mathtt{I}_{2\times 2}]\mathbf{X} = \mathbf{0}$. So $\mathcal{V}_\mathtt{H}$ is a codimension-2 subspace of $\mathbb{R}^4$.

(c) Any codimension-2 subspace may be expressed as $[\mathtt{H}_{2\times 2} | -\mathtt{I}]\mathbf{X} = \mathbf{0}$ for suitable $\mathtt{H}_{2\times 2}$. Thus given measurements $\mathbf{X}_i$, the estimation task is equivalent to finding the best-fitting codimension-2 subspace.

(d) Given a matrix $\mathtt{M}$ with rows $\mathbf{X}_i^\mathsf{T}$, the best-fitting subspace to the $\mathbf{X}_i$ is spanned by the singular vectors $\mathbf{V}_1$ and $\mathbf{V}_2$ corresponding to the two largest singular values of $\mathtt{M}$.

(e) The $\mathtt{H}_{2\times 2}$ corresponding to the subspace spanned by $\mathbf{V}_1$ and $\mathbf{V}_2$ is found by solving the equations $[\mathtt{H}_{2\times 2} | -\mathtt{I}][\mathbf{V}_1 \mathbf{V}_2] = \mathtt{0}$.

(ix) **Computing homographies of $\mathbb{P}^3$ from line correspondences.** Consider computing a 4×4 homography $\mathtt{H}$ from lines correspondences alone, assuming the lines are in general position in $\mathbb{P}^3$. There are two questions: how many correspondences are required?, and how to formulate the algebraic constraints to obtain a solution for $\mathtt{H}$? It might be thought that four line correspondences would be sufficient because each line in $\mathbb{P}^3$ has four degrees of freedom, and thus four lines should provide $4 \times 4 = 16$ constraints on the 15 degrees of freedom of $\mathtt{H}$. However, a configuration of four lines is degenerate (see section 4.1.3(*p*91)) for computing the transformation, as there is a 2D isotropy subgroup. This is discussed further in [Hartley-94c]. Equations linear in $\mathtt{H}$ can be obtained in the following way:

$$\boldsymbol{\pi}_i^\mathsf{T} \mathtt{H} \mathbf{X}_j = 0\,, \quad i = 1, 2,\ j = 1, 2\,,$$

where $\mathtt{H}$ transfers a line defined by the two points $(\mathbf{X}_1, \mathbf{X}_2)$ to a line defined by the intersection of the two planes $(\boldsymbol{\pi}_1, \boldsymbol{\pi}_2)$. This method was derived in [Oskarsson-02], where more details are to be found.

Samsung/Google, EX1010, p. 149

5

Algorithm Evaluation and Error Analysis

This chapter describes methods for assessing and quantifying the results of estimation algorithms. Often it is not sufficient to simply have an estimate of a variable or transformation. Instead some measure of confidence or uncertainty is also required.

Two methods for computing this uncertainty (covariance) are outlined here. The first is based on linear approximations and involves concatenating various Jacobian expressions. The second is the easier to implement Monte Carlo method.

5.1 Bounds on performance

Once an algorithm has been developed for the estimation of a certain type of transformation it is time to test its performance. This may be done by testing it on real or on synthetic data. In this section, testing on synthetic data will be considered, and a methodology for testing will be sketched.

We recall the notational convention:

- A quantity such as $\mathbf{x}$ represents a measured image point.
- Estimated quantities are represented by a hat, such as $\hat{\mathbf{x}}$ or $\hat{\mathtt{H}}$.
- True values are represented by a bar, such as $\bar{\mathbf{x}}$ or $\bar{\mathtt{H}}$.

Typically, testing will begin with the synthetic generation of a set of image correspondences $\bar{\mathbf{x}}_i \leftrightarrow \bar{\mathbf{x}}'_i$ between two images. The number of such correspondences will vary. Corresponding points will be chosen in such a way that they correspond via a given fixed projective transformation $\bar{\mathtt{H}}$, and the correspondence is exact, in the sense that $\bar{\mathbf{x}}'_i = \bar{\mathtt{H}}\bar{\mathbf{x}}_i$ precisely, up to machine accuracy.

Next, artificial Gaussian noise will be added to the image measurements by perturbing both the x- and y-coordinates of the point by a zero-mean Gaussian random variable with known variance. The resulting noisy points are denoted $\mathbf{x}_i$ and $\mathbf{x}'_i$. A suitable Gaussian random number generator is given in [Press-88]. The estimation algorithm is then run to compute the estimated quantity. For the 2D projective transformation problem considered in chapter 4, this means the projective transformation itself, and also perhaps estimates of the correct original noise-free image points. The algorithm is then evaluated according to how closely the computed model matches the (noisy) input data, or alternatively, how closely the estimated model agrees with the original

Samsung/Google, EX1010, p. 150

noise-free data. This procedure should be carried out many times with different noise (i.e. a different seed for the random number generator, though each time with the same noise variance) in order to obtain a statistically meaningful performance evaluation.

5.1.1 Error in one image

To illustrate this, we continue our investigation of the problem of 2D homography estimation. For simplicity we consider the case where noise is added to the coordinates of the second image only. Thus, $\mathbf{x}_i = \bar{\mathbf{x}}_i$ for all i. Let $\mathbf{x}_i \leftrightarrow \mathbf{x}'_i$ be a set of noisy matched points between two images, generated from a perfectly matched set of data by injection of Gaussian noise with variance σ^2 in each of the two coordinates of the second (primed) image. Let there be n such matched points. From this data, a projective transformation $\hat{\mathtt{H}}$ is estimated using any one of the algorithms described in chapter 4. Obviously, the estimated transformation $\hat{\mathtt{H}}$ will not generally map $\mathbf{x}_i$ to $\mathbf{x}'_i$, nor $\bar{\mathbf{x}}_i$ to $\bar{\mathbf{x}}'_i$ precisely, because of the injected noise in the coordinates of $\mathbf{x}'_i$. The RMS (root-mean-squared) residual error

$$\epsilon_{\text{res}} = \left(\frac{1}{2n} \sum_{i=1}^{n} d(\mathbf{x}'_i, \hat{\mathbf{x}}'_i)^2 \right)^{1/2} \tag{5.1}$$

measures the average difference between the noisy input data ($\mathbf{x}'_i$) and the estimated points $\hat{\mathbf{x}}'_i = \hat{\mathtt{H}}\mathbf{x}_i$. It is therefore appropriately called *residual error*. It measures how well the computed transformation matches the input data, and as such is a suitable quality measure for the estimation procedure.

The value of the residual error is *not* in itself an absolute measure of the quality of the solution obtained. For instance, consider the 2D projectivity problem in the case where the input data consists of just 4 matched points. Since a projective transformation is defined uniquely and exactly by 4 point correspondences, any reasonable algorithm will compute an $\hat{\mathtt{H}}$ that matches the points exactly, in the sense that $\mathbf{x}'_i = \hat{\mathtt{H}}\mathbf{x}_i$. This means that the residual error is zero. One cannot expect any better performance from an algorithm than this.

Note that $\hat{\mathtt{H}}$ matches the projected points to the input data $\mathbf{x}'_i$, and not to the original noise-free data, $\bar{\mathbf{x}}'_i$. In fact, since the difference between the noise-free and the noisy coordinates has variance σ^2, in the minimal four-point case the residual difference between projected points $\hat{\mathtt{H}}\mathbf{x}_i$ and the noise-free data $\bar{\mathbf{x}}'_i$ also has variance σ^2. Thus, in the case of 4 points, the model fits the noisy input data perfectly (i.e. the residual is zero), but does not give a very close approximation to the true noise-free values.

With more than 4 point matches, the value of the residual error will increase. In fact, intuitively, one expects that as the number of measurements (matched points) increases, the estimated model should agree more and more closely with the noise-free true values. Asymptotically, the variance should decrease in inverse proportion to the number of point matches. At the same time, the residual error will increase.

Samsung/Google, EX1010, p. 151

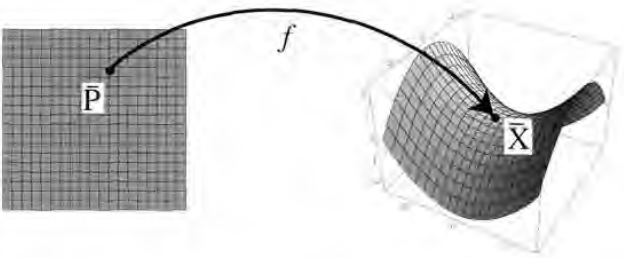


Fig. 5.1. *As the values of the parameters* $\mathbf{P}$ *vary, the function image traces out a surface* S_M *through the true value* $\overline{\mathbf{X}}$.

5.1.2 Error in both images

In the case of error in both images, the residual error is

$$\epsilon_{\text{res}} = \frac{1}{\sqrt{4n}} \left(\sum_{i=1}^{n} d(\mathbf{x}_i, \hat{\mathbf{x}}_i)^2 + \sum_{i=1}^{n} d(\mathbf{x}'_i, \hat{\mathbf{x}}'_i)^2 \right)^{1/2} \tag{5.2}$$

where $\hat{\mathbf{x}}_i$ and $\hat{\mathbf{x}}'_i$ are estimated points such that $\hat{\mathbf{x}}'_i = \overline{\mathbf{H}}\hat{\mathbf{x}}_i$.

5.1.3 Optimal estimators (MLE)

Bounds for estimation performance will be considered in a general framework, and then specialized to the two cases of error in one or both images. The goal is to derive formulae for the expected residual error of the Maximum Likelihood Estimate (MLE). As described previously, minimization of geometric error is equivalent to MLE, and so the goal of any algorithm implementing minimization of geometric error should be to achieve the theoretical bound given for the MLE. Another algorithm minimizing a different cost function (such as algebraic error) can be judged according to how close it gets to the bound given by the MLE.

A general estimation problem is concerned with a function f from $\mathbb{R}^M$ to $\mathbb{R}^N$ as described in section 4.2.7(*p*101), where $\mathbb{R}^M$ is a parameter space, and $\mathbb{R}^N$ is a space of measurements. Consider now a point $\overline{\mathbf{X}} \in \mathbb{R}^N$ for which there exists a vector of parameters $\overline{\mathbf{P}} \in \mathbb{R}^M$ such that $f(\overline{\mathbf{P}}) = \overline{\mathbf{X}}$ (i.e. a point $\overline{\mathbf{X}}$ in the range of f with preimage $\overline{\mathbf{P}}$). In the context of 2D projectivities with measurements in the second image only, this corresponds to a noise-free set of points $\bar{\mathbf{x}}'_i = \overline{\mathbf{H}}\bar{\mathbf{x}}_i$. The x- and y-components of the n points $\bar{\mathbf{x}}'_i, i = 1, \ldots, n$ constitute the N-vector $\overline{\mathbf{X}}$ with $N = 2n$, and the parameters of the homography constitute the vector $\overline{\mathbf{P}}$ which may be an 8- or 9-vector depending on the parametrization of $\overline{\mathbf{H}}$.

Let $\mathbf{X}$ be a measurement vector chosen according to an isotropic Gaussian distribution with mean the true measurement $\overline{\mathbf{X}}$ and variance $N\sigma^2$ (this notation means that each of the N components has variance σ^2). As the value of the parameter vector $\mathbf{P}$ varies in a neighbourhood of the point $\overline{\mathbf{P}}$, the value of the function $f(\mathbf{P})$ traces out a surface S_M in $\mathbb{R}^N$ through the point $\overline{\mathbf{X}}$. This is illustrated in figure 5.1. The surface S_M

Samsung/Google, EX1010, p. 152

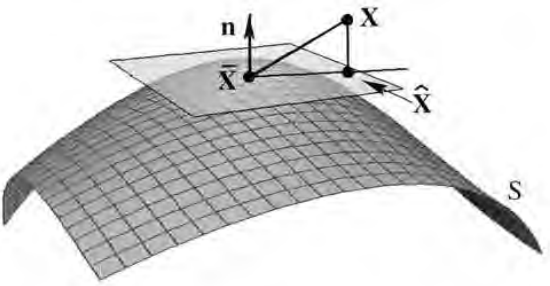


Fig. 5.2. *Geometry of the errors in measurement space using the tangent plane approximation to S_M. The estimated point $\hat{\mathbf{X}}$ is the closest point on S_M to the measured point $\mathbf{X}$. The residual error is the distance between the measured point $\mathbf{X}$ and $\hat{\mathbf{X}}$. The estimation error is the distance from $\hat{\mathbf{X}}$ to the true point $\bar{\mathbf{X}}$.*

is given by the range of f. The dimension of the surface as a submanifold of $\mathbb{R}^N$ is equal to d, where d is the number of essential parameters (that is the number of degrees of freedom, or minimum number of parameters). In the single-image error case, this equals 8, since the mapping determined by the matrix H is independent of scale.

Now, given a measurement vector $\mathbf{x}$, the maximum likelihood (ML) estimate $\hat{\mathbf{x}}$ is the point on S_M closest to $\mathbf{x}$. The ML estimator is the one that returns this closest point to $\mathbf{x}$ that lies on this surface. Denote this ML estimate by $\hat{\mathbf{x}}$.

We now assume that in the neighbourhood of $\bar{\mathbf{x}}$, the surface is essentially planar and is well approximated by the tangent surface – at least for neighbourhoods around $\bar{\mathbf{x}}$ of the order of magnitude of noise variance. In this linear approximation, the ML estimate $\hat{\mathbf{x}}$ is the foot of the perpendicular from $\mathbf{x}$ onto the tangent plane. The residual error is the distance from the point $\mathbf{x}$ to the estimated value $\hat{\mathbf{x}}$. Furthermore, the distance from $\hat{\mathbf{x}}$ to (the unknown) $\bar{\mathbf{x}}$ is the distance from the optimally estimated value to the true value as seen in figure 5.2. Our task is to compute the expected value of these errors.

Computing the expected ML residual error has now been abstracted to a geometric problem as follows. The *total variance* of an N-dimensional Gaussian distribution is the trace of the covariance matrix, that is the sum of variances in each of the axial directions. This is, of course, unchanged by a change of orthogonal coordinate frame. For an N-dimensional isotropic Gaussian distribution with independent variances σ^2 in each variable, the total variance is $N\sigma^2$. Now, given an isotropic Gaussian random variable defined on $\mathbb{R}^N$ with total variance $N\sigma^2$ and mean the true point $\bar{\mathbf{x}}$, we wish to compute the expected distance of the random variable from a dimension d hyperplane passing through $\bar{\mathbf{x}}$. The projection of a Gaussian random variable in $\mathbb{R}^N$ onto the d-dimensional tangent plane gives the distribution of the *estimation error* (the difference between the estimated value and the true result). Projection onto the

Samsung/Google, EX1010, p. 153

$(N-d)$-dimensional normal to the tangent surface gives the distribution of the residual error.

By a rotation of axes if necessary, one may assume, without loss of generality, that the tangent surface coincides with the first d coordinate axes. Integration over the remaining axial directions provides the following result.

Result 5.1. *The projection of an isotropic Gaussian distribution defined on $\mathbb{R}^N$ with total variance $N\sigma^2$ onto a subspace of dimension s is an isotropic Gaussian distribution with total variance $s\sigma^2$.*

The proof of this is straightforward, and is omitted. We apply this in the two cases where $s = d$ and $s = N - d$ to obtain the following results.

Result 5.2. *Consider an estimation problem where N measurements are to be modelled by a function depending on a set of d essential parameters. Suppose the measurements are subject to independent Gaussian noise with standard deviation σ in each measurement variable.*

(i) *The RMS* **residual error** *(distance of the measured from the estimated value) for the ML estimator is*

$$\epsilon_{\text{res}} = E[\|\hat{\mathbf{X}} - \mathbf{X}\|^2/N]^{1/2} = \sigma(1 - d/N)^{1/2} \tag{5.3}$$

(ii) *The RMS* **estimation error** *(distance of the estimated from the true value) for the ML estimator is*

$$\epsilon_{\text{est}} = E[\|\hat{\mathbf{X}} - \overline{\mathbf{X}}\|^2/N]^{1/2} = \sigma(d/N)^{1/2} \tag{5.4}$$

where $\mathbf{X}$, $\hat{\mathbf{X}}$ and $\overline{\mathbf{X}}$ are respectively the measured, estimated and true values of the measurement vector.

Result 5.2 follows directly from result 5.1 by dividing by N to get the variance per measurement, then taking a square root to get standard deviation, instead of variance.

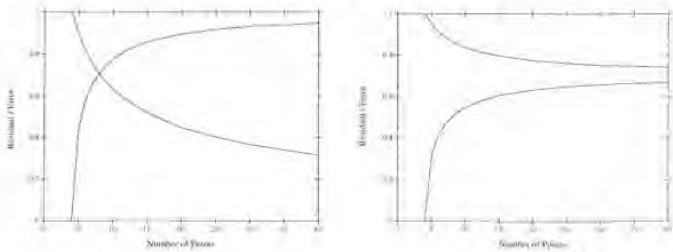
These values give lower bounds for residual error against which a particular estimation algorithm may be measured.

2D homography – error in one image. For the 2D projectivity estimation problem considered in this chapter, assuming error in the second image only, we have $d = 8$ and $N = 2n$, where n is the number of point matches. Thus, we have for this problem

$$\begin{aligned} \epsilon_{\text{res}} &= \sigma\,(1 - 4/n)^{1/2} \\ \epsilon_{\text{est}} &= \sigma\,(4/n)^{1/2}\,. \end{aligned} \tag{5.5}$$

Graphs of these errors as n varies are shown in figure 5.3.

Samsung/Google, EX1010, p. 154



a b

Fig. 5.3. *Optimal error when noise is present in (a) one image, and in (b) both images as the number of points varies. An error level of one pixel is assumed. The descending curve shows the estimation error* ϵ_{est} *and the ascending curve shows the residual error* ϵ_{res}.

Error in both images. In this case, $N = 4n$ and $d = 2n + 8$. As before, assuming linearity of the tangent surface in the neighbourhood of the true measurement vector $\widehat{\mathbf{x}}$, result 5.2 gives the following expected errors.

$$\begin{aligned} \epsilon_{\text{res}} &= \sigma \left(\frac{n-4}{2n} \right)^{1/2} \\ \epsilon_{\text{est}} &= \sigma \left(\frac{n+4}{2n} \right)^{1/2} . \end{aligned} \tag{5.6}$$

Graphs of these errors as n varies are also shown in Figure 5.3.

An interesting observation to be made from this graph is that the asymptotic error with respect to the true values is $\sigma/\sqrt{2}$, and not 0 as in the case of error in one image. This result is expected, since in effect, one has two measurements of the position of each point, one in each image, related by the projective transformation. With two measurements of a point the variance in the estimate of the point position decreases by a factor of $\sqrt{2}$. By contrast, in the previous case where errors occur in one image only, one has one exact measurement for each point (i.e. in the first image). Thus, as the transformation H is estimated with greater and greater accuracy, the exact position of the point in the second image becomes known with uncertainty asymptotically approaching 0.

Mahalanobis distance. The formulae quoted above were derived under the assumption that the error distribution in measurement space was an isotropic Gaussian distribution, meaning that errors in each coordinate were independent. This assumption is not essential. We may assume any Gaussian distribution of error, with covariance matrix Σ. The formulae of result 5.2 remain true with ϵ being replaced with the expected Mahalanobis distance $E[\|\widehat{\mathbf{x}} - \mathbf{x}\|_{\Sigma}^2/N]^{1/2}$. The standard deviation σ also disappears, since it is taken care of by the Mahalanobis distance.

This follows from a simple change of coordinates in the measurement space $\mathbb{R}^N$ to make the covariance matrix equal to the identity. In this new coordinate frame, Mahalanobis distance becomes the same as Euclidean distance.

Samsung/Google, EX1010, p. 155

5.1.4 Determining the correct convergence of an algorithm

The relations given in (5.3) and (5.4) give a simple way of determining correct convergence of an estimation algorithm, without the need to determine the number of degrees of freedom of the problem. As seen in figure 5.2, the measurement space corresponding to the model specified by the parameter vector $\mathbf{P}$ forms a surface S_M. If near the noise-free data $\overline{\mathbf{X}}$ the surface is nearly planar, then it may be approximated by its tangent plane, and the three points $\hat{\mathbf{X}}$, $\mathbf{X}$ and $\overline{\mathbf{X}}$ form a right-angled triangle. In most estimation problems this assumption of planarity will be very close to correct at the scale set by typical noise magnitude. In this case, the Pythagorean equality may be written as

$$\|\mathbf{X} - \overline{\mathbf{X}}\|^2 = \|\mathbf{X} - \hat{\mathbf{X}}\|^2 + \|\overline{\mathbf{X}} - \hat{\mathbf{X}}\|^2 \tag{5.7}$$

In evaluating an algorithm with synthetic data, this equality allows a simple test to see whether the algorithm has converged to the optimal value. If the estimated value $\hat{\mathbf{X}}$ satisfies this equality, then it is a strong indication that the algorithm has found the true global minimum. Note that it is unnecessary in applying this test to determine the number of degrees of freedom of the problem. A few more properties are listed:

- This test can be used to determine on a run-by-run basis whether the algorithm has succeeded. Thus, with repeated runs, it allows an estimate of the percentage success rate for the algorithm.
- This test can only be used for synthetic data, or at least data for which the true measurements $\overline{\mathbf{X}}$ are known.
- The equality (5.7) depends on the assumption that the surface S_M consisting of valid measurements is locally planar. If the equality is not satisfied for a particular run of the estimation algorithm, then this is because the surface is not planar, or (far more likely) because the algorithm is failing to find the best solution.
- The test (5.7) is a test for the algorithm finding the global, not a local solution. If $\hat{\mathbf{X}}$ settles to a local cost minimum, then the right-hand-side of (5.7) is likely to be much larger than the left-hand-side. The condition is unlikely to be satisfied entirely by chance if the algorithm finds the incorrect point $\hat{\mathbf{X}}$.

5.2 Covariance of the estimated transformation

In the previous section the ML estimate was considered, and how its expected average error may be computed. Comparing the achieved residual error or estimation error of an algorithm against the ML error is a good way of evaluating the performance of a particular estimation algorithm, since it compares the results of the algorithm against the best that may be achieved (the optimum estimate) in the absence of any other prior information.

Nevertheless, the chief concern is how accurately the transformation itself has been computed. The uncertainty of the estimated transformation depends on many factors, including the number of points used to compute it, the accuracy of the given point matches, as well as the configuration of the points in question. To illustrate the importance of the configuration suppose the points used to compute the transformation are

Samsung/Google, EX1010, p. 156

close to a degenerate configuration; then the transformation may not be computed with great accuracy. For instance, if the transformation is computed from a set of points that lie close to a straight line, then the behaviour of the transformation in the dimension perpendicular to that line is not accurately determined. Thus, whereas the achievable residual error and estimation error were seen to be dependent only on the number of point correspondences and their accuracy, by contrast, the accuracy of the computed transformation is dependent on the particular points. The uncertainty of the computed transformation is conveniently captured in the *covariance matrix* of the transformation. Since $\mathtt{H}$ is a matrix with 9 entries, its covariance matrix will be a 9×9 matrix. In this section it will be seen how this covariance matrix may be computed.

5.2.1 Forward propagation of covariance

The covariance matrix behaves in a pleasantly simple manner under affine transformations, as described in the following theorem.

Result 5.3. *Let* $\mathbf{v}$ *be a random vector in* $\mathbb{R}^M$ *with mean* $\bar{\mathbf{v}}$ *and covariance matrix* Σ*, and suppose that* $f : \mathbb{R}^M \to \mathbb{R}^N$ *is an affine mapping defined by* $f(\mathbf{v}) = f(\bar{\mathbf{v}}) + \mathtt{A}(\mathbf{v} - \bar{\mathbf{v}})$*. Then* $f(\mathbf{v})$ *is a random variable with mean* $f(\bar{\mathbf{v}})$ *and covariance matrix* $\mathtt{A}\Sigma\mathtt{A}^\mathsf{T}$*.*

Note that it is not assumed that $\mathtt{A}$ is a square matrix. Instead of giving a proof of this theorem, we give an example.

Example 5.4. Let x and y be independent random variables with mean 0 and standard deviations of 1 and 2 respectively. What are the mean and standard deviation of $x' = f(x, y) = 3x + 2y - 7$?

The mean is $\bar{x}' = f(0, 0) = -7$. Next we compute the variance of x'. In this case, Σ is the matrix $\begin{bmatrix} 1 & 0 \\ 0 & 4 \end{bmatrix}$ and $\mathtt{A}$ is the matrix $[3\ 2]$. Thus, the variance of x' is $\mathtt{A}\Sigma\mathtt{A}^\mathsf{T} = 25$. Thus $3x + 2y - 7$ has standard deviation 5. △

Example 5.5. Let $x' = 3x + 2y$ and $y' = 3x - 2y$. Find the covariance matrix of (x', y'), given that x and y have the same distribution as before.

In this case, the matrix $\mathtt{A} = \begin{bmatrix} 3 & 2 \\ 3 & -2 \end{bmatrix}$. One computes $\mathtt{A}\Sigma\mathtt{A}^\mathsf{T} = \begin{bmatrix} 25 & -7 \\ -7 & 25 \end{bmatrix}$. Thus, one sees that both x' and y' have variance 25 (standard deviation 5), whereas x' and y' are negatively correlated, with covariance $E[x'y'] = -7$. △

Non-linear propagation. If $\mathbf{v}$ is a random vector in $\mathbb{R}^M$ and $f : \mathbb{R}^M \to \mathbb{R}^N$ is a non-linear function acting on $\mathbf{v}$, then we may compute an approximation to the mean and covariance of $f(\mathbf{v})$ by assuming that f is approximately affine in the vicinity of the mean of the distribution. The affine approximation to f is $f(\mathbf{v}) \approx f(\bar{\mathbf{v}}) + \mathtt{J}(\mathbf{v} - \bar{\mathbf{v}})$, where $\mathtt{J}$ is the partial derivative (Jacobian) matrix $\partial f / \partial \mathbf{v}$ evaluated at $\bar{\mathbf{v}}$. Note that $\mathtt{J}$ has dimension $N \times M$. Then we have the following result.

Result 5.6. *Let* $\mathbf{v}$ *be a random vector in* $\mathbb{R}^M$ *with mean* $\bar{\mathbf{v}}$ *and covariance matrix* Σ*,*

Samsung/Google, EX1010, p. 157

and let $f : \mathbb{R}^M \to \mathbb{R}^N$ be differentiable in a neighbourhood of $\bar{\mathbf{v}}$. Then up to a first-order approximation, $f(\mathbf{v})$ is a random variable with mean $f(\bar{\mathbf{v}})$ and covariance $\mathtt{J}\Sigma\mathtt{J}^\mathsf{T}$, where $\mathtt{J}$ is the Jacobian matrix of f, evaluated at $\bar{\mathbf{v}}$.

The extent to which this result gives a good approximation to the actual mean and variance of $f(\bar{\mathbf{v}})$ depends on how closely the function f is approximated by a linear function in a region about $\bar{\mathbf{v}}$ commensurate in size with the support of the probability distribution of $\mathbf{v}$.

Example 5.7. Let $\mathbf{x} = (x, y)^\mathsf{T}$ be a Gaussian random vector with mean $(0, 0)^\mathsf{T}$ and covariance matrix $\sigma^2\mathrm{diag}(1, 4)$. Let $x' = f(x, y) = x^2 + 3x - 2y + 5$. Then one may compute the true values of the mean and standard deviation of $f(x, y)$ according to the formulae

$$\begin{aligned} \bar{x}' &= \int\int_{-\infty}^{\infty} P(x, y) f(x, y) dx dy \\ \sigma_{x'}^2 &= \int\int_{-\infty}^{\infty} P(x, y)(f(x, y) - \bar{x}')^2 dx dy \end{aligned}$$

where

$$P(x, y) = \frac{1}{4\pi\sigma^2} e^{-(x^2 + y^2/4)/2\sigma^2}$$

is the Gaussian probability distribution (A2.1–*p*565). One obtains

$$\begin{aligned} \bar{x}' &= 5 + \sigma^2 \\ \sigma_{x'}^2 &= 25\sigma^2 + 2\sigma^4. \end{aligned}$$

Applying the approximation given by result 5.6, and noting that $\mathtt{J} = [3 \;\; -2]$, we find that the estimated values are

$$\begin{aligned} \bar{x}' &= 5 \\ \sigma_{x'}^2 &= \sigma^2 [3 \;\; -2] \begin{bmatrix} 1 & \\ & 4 \end{bmatrix} [3 \;\; -2]^\mathsf{T} = 25\sigma^2. \end{aligned}$$

Thus, as long as σ is small, this is a close approximation to the correct values of the mean and variance of x'. The following table shows the true and approximated values for the mean and standard deviation of $f(x, y)$ for two different values of σ.

	$\sigma = 0.25$		$\sigma = 0.5$	
	$\bar{x}'$	$\sigma_{x'}$	$\bar{x}'$	$\sigma_{x'}$
estimate	5.0000	1.25000	5.00	2.5000
true	5.0625	1.25312	5.25	2.5249

For reference, in the case $\sigma = 0.25$, one sees that as long as $|x| < 2\sigma$ (about 95% of the total distribution) the value $f(x, y) = x^2 + 3x - 2y + 5$ differs from its linear approximation $3x - 2y + 5$ by no more than $x^2 < 0.25$. △

Samsung/Google, EX1010, p. 158

Example 5.8. More generally, assuming that x and y are independent zero-mean Gaussian random variables, one may compute that for the function $f(x, y) = ax^2 + bxy + cy^2 + dx + ey + f$,

$$\begin{aligned} \text{mean} &= a\sigma_x^2 + c\sigma_y^2 + f \\ \text{variance} &= 2a^2\sigma_x^4 + b^2\sigma_x^2\sigma_y^2 + 2c^2\sigma_y^4 + d^2\sigma_x^2 + e^2\sigma_y^2 \end{aligned}$$

which are close to the estimated values mean $= f$, variance $= d^2\sigma_x^2 + e^2\sigma_y^2$ as long as σ_x and σ_y are small. △

5.2.2 Backward propagation of covariance

The material in this and the following section 5.2.3 is more advanced. The examples in section 5.2.4 show the straightforward application of the results of these sections, and can be read first.

Consider a differentiable mapping f from a "parameter space", $\mathbb{R}^M$ to a "measurement space" $\mathbb{R}^N$, and let a Gaussian probability distribution be defined on $\mathbb{R}^N$ with covariance matrix Σ. Let S_{M} be the image of the mapping f. We assume that $M < N$ and that S_{M} has the same dimension M as the parameter space $\mathbb{R}^M$. Thus we are not considering the over-parametrized case at present. A vector $\mathbf{P}$ in $\mathbb{R}^M$ represents a parametrization of the point $f(\mathbf{P})$ on S_{M}. Finding the point on S_{M} closest in Mahalanobis distance to a given point $\mathbf{X}$ in $\mathbb{R}^N$ defines a map from $\mathbb{R}^N$ to the surface S_{M}. We call this mapping $\eta : \mathbb{R}^N \to S_{\text{M}}$. Now, f is by assumption invertible on the surface S_{M}, and we define $f^{-1} : S_{\text{M}} \to \mathbb{R}^M$ to be the inverse function.

By composing the map $\eta : \mathbb{R}^N \to S_{\text{M}}$ and $f^{-1} : S_{\text{M}} \to \mathbb{R}^M$ we have a mapping $f^{-1} \circ \eta : \mathbb{R}^N \to \mathbb{R}^M$. This mapping assigns to a measurement vector $\mathbf{X}$, the set of parameters $\mathbf{P}$ corresponding to the ML estimate $\widehat{\mathbf{X}}$. In principle we may propagate the covariance of the probability distribution in the measurement space $\mathbb{R}^N$ to compute a covariance matrix for the set of parameters $\mathbf{P}$ corresponding to ML estimation. Our goal is to apply result 5.3 or result 5.6.

We consider first the case where the mapping f is an affine mapping from $\mathbb{R}^M$ into $\mathbb{R}^N$. We will show next that the mapping $f^{-1} \circ \eta$ is also an affine mapping, and a specific form will be given for $f^{-1} \circ \eta$, thereby allowing us to apply result 5.3 to compute the covariance of the estimated parameters $\widehat{\mathbf{P}} = f^{-1} \circ \eta(\mathbf{X})$.

Since f is affine, we may write $f(\mathbf{P}) = f(\overline{\mathbf{P}}) + \mathrm{J}(\mathbf{P} - \overline{\mathbf{P}})$, where $f(\overline{\mathbf{P}}) = \overline{\mathbf{X}}$ is the mean of the probability distribution on $\mathbb{R}^N$. Since we are assuming that the surface $S_{\text{M}} = f(\mathbb{R}^M)$ has dimension M, the rank of J is equal to its column dimension. Given a measurement vector $\mathbf{X}$, the ML estimate $\widehat{\mathbf{X}}$ minimizes $\|\mathbf{X} - \widehat{\mathbf{X}}\|_\Sigma = \|\mathbf{X} - f(\widehat{\mathbf{P}})\|_\Sigma$. Thus, we seek $\widehat{\mathbf{P}}$ to minimize this latter quantity. However,

$$\|\mathbf{X} - f(\widehat{\mathbf{P}})\|_\Sigma = \|(\mathbf{X} - \overline{\mathbf{X}}) - \mathrm{J}(\widehat{\mathbf{P}} - \overline{\mathbf{P}})\|_\Sigma$$

and this is minimized (see (A5.2–p591) in section A5.2.1(p591)) when

$$(\widehat{\mathbf{P}} - \overline{\mathbf{P}}) = (\mathrm{J}^\mathsf{T}\Sigma^{-1}\mathrm{J})^{-1}\mathrm{J}^\mathsf{T}\Sigma^{-1}(\mathbf{X} - \overline{\mathbf{X}}) \ .$$

Samsung/Google, EX1010, p. 159

Writing $\overline{\mathbf{P}} = f^{-1}\overline{\mathbf{X}}$ and $\widehat{\mathbf{P}} = f^{-1}\widehat{\mathbf{X}}$, we see that

$$\begin{aligned} f^{-1} \circ \eta(\mathbf{X}) &= \widehat{\mathbf{P}} \\ &= (\mathtt{J}^\mathsf{T}\Sigma^{-1}\mathtt{J})^{-1}\mathtt{J}^\mathsf{T}\Sigma^{-1}(\mathbf{X} - \overline{\mathbf{X}}) + f^{-1}(\overline{\mathbf{X}}) \\ &= (\mathtt{J}^\mathsf{T}\Sigma^{-1}\mathtt{J})^{-1}\mathtt{J}^\mathsf{T}\Sigma^{-1}(\mathbf{X} - \overline{\mathbf{X}}) + f^{-1} \circ \eta(\overline{\mathbf{X}}) \ . \end{aligned}$$

This shows that $f^{-1} \circ \eta$ is affine and $(\mathtt{J}^\mathsf{T}\Sigma^{-1}\mathtt{J})^{-1}\mathtt{J}^\mathsf{T}\Sigma^{-1}$ is its linear part. Applying result 5.3, we see that the covariance matrix for $\widehat{\mathbf{P}}$ is

$$\begin{aligned} \left[(\mathtt{J}^\mathsf{T}\Sigma^{-1}\mathtt{J})^{-1}\mathtt{J}^\mathsf{T}\Sigma^{-1}\right]\Sigma\left[(\mathtt{J}^\mathsf{T}\Sigma^{-1}\mathtt{J})^{-1}\mathtt{J}^\mathsf{T}\Sigma^{-1}\right]^\mathsf{T} &= (\mathtt{J}^\mathsf{T}\Sigma^{-1}\mathtt{J})^{-1}\mathtt{J}^\mathsf{T}\Sigma^{-1}\Sigma\Sigma^{-1}\mathtt{J}(\mathtt{J}^\mathsf{T}\Sigma^{-1}\mathtt{J})^{-1} \\ &= (\mathtt{J}^\mathsf{T}\Sigma^{-1}\mathtt{J})^{-1}, \end{aligned}$$

recalling that Σ is symmetric. We have proved the following theorem.

Result 5.9 Backward transport of covariance – affine case. *Let $f : \mathbb{R}^M \to \mathbb{R}^N$ be an affine mapping of the form $f(\mathbf{P}) = f(\overline{\mathbf{P}}) + \mathtt{J}(\mathbf{P} - \overline{\mathbf{P}})$, where $\mathtt{J}$ has rank M. Let $\mathbf{X}$ be a random variable in $\mathbb{R}^N$ with mean $\overline{\mathbf{X}} = f(\overline{\mathbf{P}})$ and covariance matrix Σ. Let $f^{-1} \circ \eta : \mathbb{R}^N \to \mathbb{R}^M$ be the mapping that maps a measurement $\mathbf{X}$ to the set of parameters corresponding to the ML estimate $\widehat{\mathbf{X}}$. Then $\widehat{\mathbf{P}} = f^{-1} \circ \eta(\mathbf{X})$ is a random variable with mean $\overline{\mathbf{P}}$ and covariance matrix*

$$\Sigma_\mathbf{P} = (\mathtt{J}^\mathsf{T}\Sigma_\mathbf{X}^{-1}\mathtt{J})^{-1}. \tag{5.8}$$

In the case where f is not affine, an approximation to the mean and covariance may be obtained by approximating f by an affine function in the usual way, as follows.

Result 5.10 Backward transport of covariance – non-linear case. *Let $f : \mathbb{R}^M \to \mathbb{R}^N$ be a differentiable mapping and let $\mathtt{J}$ be its Jacobian matrix evaluated at a point $\overline{\mathbf{P}}$. Suppose that $\mathtt{J}$ has rank M. Then f is one-to-one in a neighbourhood of $\overline{\mathbf{P}}$. Let $\mathbf{X}$ be a random variable in $\mathbb{R}^N$ with mean $\overline{\mathbf{X}} = f(\overline{\mathbf{P}})$ and covariance matrix $\Sigma_\mathbf{X}$. Let $f^{-1} \circ \eta : \mathbb{R}^N \to \mathbb{R}^M$ be the mapping that maps a measurement $\mathbf{X}$ to the set of parameters corresponding to the ML estimate $\widehat{\mathbf{X}}$. Then to first-order, $\widehat{\mathbf{P}} = f^{-1} \circ \eta(\mathbf{X})$ is a random variable with mean $\overline{\mathbf{P}}$ and covariance matrix $(\mathtt{J}^\mathsf{T}\Sigma_\mathbf{X}^{-1}\mathtt{J})^{-1}$.*

5.2.3 Over-parametrization

One may generalize result 5.9 and result 5.10 to the case of redundant sets of parameters – the over-parametrized case. In this case, the mapping f from the parameter space $\mathbb{R}^M$ to measurement space $\mathbb{R}^N$ is not locally one-to-one. For instance, in the case of estimating a 2D homography as discussed in section 4.5(p110) there is a mapping $f(\mathbf{P})$ where $\mathbf{P}$ is a 9-vector representing the entries of the homography matrix $\mathtt{H}$. Since the homography has only 8 degrees of freedom, the mapping f is not one-to-one. In particular, for any constant k, the matrix $k\mathtt{H}$ represents the same map, and so the image coordinate vectors $f(\mathbf{P})$ and $f(k\mathbf{P})$ are equal.

In the general case of a mapping $f : \mathbb{R}^M \to \mathbb{R}^N$ the Jacobian matrix $\mathtt{J}$ does not have full rank M, but rather a smaller rank $d < M$. This rank d is called the number of *essential parameters*. The matrix $\mathtt{J}^\mathsf{T}\Sigma_\mathbf{X}^{-1}\mathtt{J}$ in this case has dimension M but rank

Samsung/Google, EX1010, p. 160

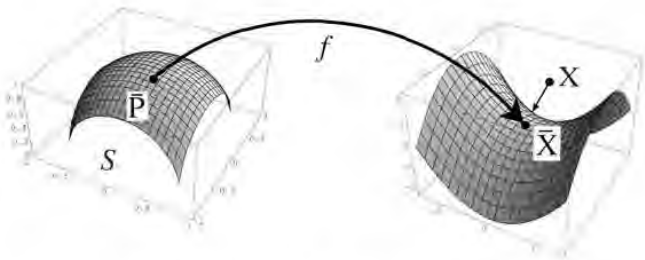


Fig. 5.4. **Back propagation (over-parametrized).** *Mapping f maps constrained parameter surface to measurement space. A measurement $\mathbf{X}$ is mapped (by a mapping η) to the closest point on the surface $f(S_{\mathrm{P}})$ and then back via f^{-1} to the parameter space, providing the ML estimate of the parameters. The covariance of $\mathbf{X}$ is transferred via $f^{-1} \circ \eta$ to a covariance of the parameters.*

$d < M$. The formula (5.8), $\Sigma_{\mathbf{P}} = (\mathtt{J}^{\mathsf{T}}\Sigma_{\mathbf{X}}^{-1}\mathtt{J})^{-1}$, clearly does not hold, since the matrix on the right side is not invertible.

In fact, it is clear that without any further restriction, the elements of the estimated vector $\widehat{\mathbf{P}}$ may vary without bound, namely through multiplication by an arbitrary constant k. Hence the elements have infinite variance. It is usual to restrict the estimated homography matrix $\mathtt{H}$ or more generally the parameter vector $\mathbf{P}$ by imposing some constraint. The usual constraint is that $\|\mathbf{P}\| = 1$ though other constraints are possible, such as demanding that the last parameter should equal 1 (see section 4.4.2(p105)). Thus, the parameter vector $\mathbf{P}$ is constrained to lie on a surface in the parameter space $\mathbb{R}^9$, or generally $\mathbb{R}^M$. In the first case the surface $\|\mathbf{P}\| = 1$ is the unit sphere in $\mathbb{R}^M$. The constraint $P_m = 1$ represents a plane in $\mathbb{R}^M$. In the general case we may assume that the estimated vector $\mathbf{P}$ lies on some submanifold of $\mathbb{R}^M$ as in the following theorem.

Result 5.11. Backward transport of covariance – over-parametrized case. *Let $f : \mathbb{R}^M \to \mathbb{R}^N$ be a differentiable mapping taking a parameter vector $\overline{\mathbf{P}}$ to a measurement vector $\overline{\mathbf{X}}$. Let S_{P} be a smooth manifold of dimension d embedded in $\mathbb{R}^M$ passing through point $\overline{\mathbf{P}}$, and such that the map f is one-to-one on the manifold S_{P} in a neighbourhood of $\overline{\mathbf{P}}$, mapping S_{P} locally to a manifold $f(S_{\mathrm{P}})$ in $\mathbb{R}^N$. The function f has a local inverse, denoted f^{-1}, restricted to the surface $f(S_{\mathrm{P}})$ in a neighbourhood of $\overline{\mathbf{X}}$. Let a Gaussian distribution on $\mathbb{R}^N$ be defined with mean $\overline{\mathbf{X}}$ and covariance matrix $\Sigma_{\mathbf{X}}$ and let $\eta : \mathbb{R}^N \to f(S_{\mathrm{P}})$ be the mapping that takes a point in $\mathbb{R}^N$ to the closest point on $f(S_{\mathrm{P}})$ with respect to Mahalanobis norm $\|\cdot\|_{\Sigma_{\mathbf{X}}}$. Via $f^{-1} \circ \eta$ the probability distribution on $\mathbb{R}^N$ with covariance matrix $\Sigma_{\mathbf{X}}$ induces a probability distribution on $\mathbb{R}^M$ with covariance matrix, to first-order equal to*

$$\Sigma_{\mathbf{P}} = (\mathtt{J}^{\mathsf{T}}\Sigma_{\mathbf{X}}^{-1}\mathtt{J})^{+\mathtt{A}} = \mathtt{A}(\mathtt{A}^{\mathsf{T}}\mathtt{J}^{\mathsf{T}}\Sigma_{\mathbf{X}}^{-1}\mathtt{J}\mathtt{A})^{-1}\mathtt{A}^{\mathsf{T}} \tag{5.9}$$

where $\mathtt{A}$ is any $m \times d$ matrix whose column vectors span the tangent space to S_{P} at $\overline{\mathbf{P}}$.

This is illustrated in figure 5.4. The notation $(\mathtt{J}^{\mathsf{T}}\Sigma_{\mathbf{X}}^{-1}\mathtt{J})^{+\mathtt{A}}$, defined by (5.9), is discussed further in section A5.2(p590).

Samsung/Google, EX1010, p. 161

Proof. The proof of result 5.11 is straightforward. Let d be the number of essential parameters. One defines a map $g : \mathbb{R}^d \rightarrow \mathbb{R}^M$ mapping an open neighbourhood U in $\mathbb{R}^d$ to an open set of S_{P} containing the point $\overline{\mathtt{P}}$. Then the combined mapping $f \circ g : \mathbb{R}^d \rightarrow \mathbb{R}^N$ is one-to-one on the neighbourhood U. Let us denote the partial derivative matrices of f by $\mathtt{J}$ and of g by $\mathtt{A}$. The matrix of partial derivatives of $f \circ g$ is then $\mathtt{JA}$. Result 5.10 now applies, and one sees that the probability distribution function with covariance matrix Σ on $\mathbb{R}^N$ may be transported backwards to a covariance matrix $(\mathtt{A}^\mathsf{T}\mathtt{J}^\mathsf{T}\Sigma^{-1}\mathtt{JA})^{-1}$ on $\mathbb{R}^d$. Transporting this forwards again to $\mathbb{R}^M$, applying result 5.6, we arrive at the covariance matrix $\mathtt{A}(\mathtt{A}^\mathsf{T}\mathtt{J}^\mathsf{T}\Sigma^{-1}\mathtt{JA})^{-1}\mathtt{A}^\mathsf{T}$ on S_{P}. This matrix, which will be denoted here by $(\mathtt{J}^\mathsf{T}\Sigma^{-1}\mathtt{J})^{+\mathtt{A}}$, is related to the pseudo-inverse of $(\mathtt{J}^\mathsf{T}\Sigma^{-1}\mathtt{J})$ as defined in section A5.2(p590). The expression (5.9) is not dependent on the particular choice of the matrix $\mathtt{A}$ as long as the column span of $\mathtt{A}$ is unchanged. In particular, if $\mathtt{A}$ is replaced by $\mathtt{AB}$ for any invertible $d \times d$ matrix $\mathtt{B}$, then the value of (5.9) does not change. Thus, any matrix $\mathtt{A}$ whose columns span the tangent space of S_{P} at $\overline{\mathtt{P}}$ will do. □

Note that the proof gives a specific way of computing a matrix $\mathtt{A}$ spanning the tangent space – namely the Jacobian matrix of g. In many instances, as we will see, there are easier ways of finding $\mathtt{A}$. Note that the covariance matrix (5.9) is singular. In particular, it has dimension M and rank $d < M$. This is because the variance of the estimated parameter set in directions orthogonal to the constraint surface S_{P} is zero – there can be no variation in that direction. Note that whereas $\mathtt{J}^\mathsf{T}\Sigma^{-1}\mathtt{J}$ is non-invertible, the $d \times d$ matrix $\mathtt{A}^\mathsf{T}\mathtt{J}^\mathsf{T}\Sigma^{-1}\mathtt{JA}$ has rank d and is invertible.

An important case occurs when the constraint surface is locally orthogonal to the null-space of the Jacobian matrix. Denote by $N_L(\mathtt{X})$ the left null-space of matrix $\mathtt{X}$, namely the space of all vectors $\mathbf{x}$ such that $\mathbf{x}^\mathsf{T}\mathtt{X} = \mathbf{0}$. Then (as shown in section A5.2-(p590)), the *pseudo-inverse* $\mathtt{X}^+$ is given by

$$\mathtt{X}^+ = \mathtt{X}^{+\mathtt{A}} = \mathtt{A}(\mathtt{A}^\mathsf{T}\mathtt{X}\mathtt{A})^{-1}\mathtt{A}^\mathsf{T}$$

if and only if $N_L(\mathtt{A}) = N_L(\mathtt{X})$. The following result then derives directly from result 5.11.

Result 5.12. *Let $f : \mathbb{R}^M \rightarrow \mathbb{R}^N$ be a differentiable mapping taking $\overline{\mathtt{P}}$ to $\overline{\mathtt{X}}$, and let $\mathtt{J}$ be the Jacobian matrix of f. Let a Gaussian distribution on $\mathbb{R}^N$ be defined at $\overline{\mathtt{X}}$ with covariance matrix $\Sigma_{\mathbf{X}}$ and let $f^{-1} \circ \eta : \mathbb{R}^N \rightarrow \mathbb{R}^M$ as in result 5.11 be the mapping taking a measurement $\mathbf{X}$ to the MLE parameter vector $\mathtt{P}$ constrained to lie on a surface S_{P} locally orthogonal to the null-space of $\mathtt{J}$. Then $f^{-1} \circ \eta$ induces a distribution on $\mathbb{R}^M$ with covariance matrix, to first-order equal to*

$$\Sigma_{\mathbf{P}} = (\mathtt{J}^\mathsf{T}\Sigma_{\mathbf{X}}^{-1}\mathtt{J})^+. \tag{5.10}$$

Note that the restriction that $\mathbf{P}$ be constrained to lie on a surface locally orthogonal to the null-space of $\mathtt{J}$ is in many cases the natural constraint. For instance, if $\mathbf{P}$ is a *homogeneous* parameter vector (such as the entries of a homogeneous matrix), the restriction is satisfied for the usual constraint $\|\mathbf{P}\| = 1$. In such a case, the constraint surface is the unit sphere, and the tangent plane at any point is perpendicular to the parameter vector. On the other hand, since $\mathtt{P}$ is a homogeneous vector, the function

Samsung/Google, EX1010, p. 162

$f(\mathtt{P})$ is invariant to changes of scale, and so $\mathtt{J}$ has a null-vector in the radial direction, thus perpendicular to the constraint surface.

In other cases, it is often not critical what restriction we place on the parameter set for the purpose of computing the covariance matrix of the parameters. In addition, since the pseudo-inversion operation is its own inverse, we can retrieve the original matrix from its pseudo-inverse, according to $\mathtt{J}^\mathsf{T}\Sigma_\mathbf{X}^{-1}\mathtt{J} = \Sigma_\mathtt{P}^+$. One can then compute the covariance matrix corresponding to any other subspace, according to

$$(\mathtt{J}^\mathsf{T}\Sigma_\mathbf{X}^{-1}\mathtt{J})^{+\mathtt{A}} = (\Sigma_\mathtt{P}^+)^{+\mathtt{A}}$$

where the columns of $\mathtt{A}$ span the constrained subspace of parameter space.

5.2.4 Application and examples

Error in one image. Let us consider the application of this theory to the problem of finding the covariance of an estimated 2D homography $\mathtt{H}$. First, we look at the case where the error is limited to the second image. The 3×3 matrix $\mathtt{H}$ is represented by a 9-dimensional parameter vector which will be denoted by $\mathbf{h}$ instead of $\mathbf{P}$ so as to remind us that it is made up of the entries of $\mathtt{H}$. The covariance of the estimated $\hat{\mathbf{h}}$ is a 9×9 symmetric matrix. We are given a set of matched points $\bar{\mathbf{x}}_i \leftrightarrow \mathbf{x}'_i$. The points $\bar{\mathbf{x}}_i$ are fixed true values, and the points $\mathbf{x}'_i$ are considered as random variables subject to Gaussian noise with variance σ^2 in each component, or if desired, with a more general covariance. The function $f : \mathbb{R}^9 \to \mathbb{R}^{2n}$ is defined as mapping a 9-vector $\mathbf{h}$ representing a matrix $\mathtt{H}$ to the $2n$-vector made up of the coordinates of the points $\mathbf{x}'_i = \mathtt{H}\bar{\mathbf{x}}_i$. The coordinates of $\mathbf{x}'_i$ make up a composite vector in $\mathbb{R}^N$, which we denote by $\mathbf{X}'$. As we have seen, as $\mathbf{h}$ varies, the point $f(\mathbf{h})$ traces out an 8-dimensional surface S_P in $\mathbb{R}^{2n}$. Each point $\mathbf{X}'$ on the surface represents a set of points $\mathbf{x}'_i$ consistent with the first-image points $\bar{\mathbf{x}}_i$. Given a vector of measurements $\mathbf{X}'$, one selects the closest point $\hat{\mathbf{X}}'$ on the surface S_P with respect to Mahalanobis distance. The pre-image $\hat{\mathbf{h}} = f^{-1}(\hat{\mathbf{X}}')$, subject to constraint $\|\mathbf{h}\| = 1$, represents the estimated homography matrix $\hat{\mathtt{H}}$, estimated using the ML estimator. From the probability distribution of values of $\mathbf{X}'$ one wishes to derive the distribution of the estimated $\hat{\mathbf{h}}$. The covariance matrix $\Sigma_\mathbf{h}$ is given by result 5.12. This covariance matrix corresponds to the constraint $\|\mathbf{h}\| = 1$.

Thus, a procedure for computing the covariance matrix of the estimated transformation is as follows.

(i) Estimate the transformation $\hat{\mathtt{H}}$ from the given data.
(ii) Compute the Jacobian matrix $\mathtt{J}_f = \partial\mathbf{X}'/\partial\mathbf{h}$, evaluated at $\hat{\mathbf{h}}$.
(iii) The covariance matrix of the estimated $\mathbf{h}$ is given by (5.10): $\Sigma_\mathbf{h} = (\mathtt{J}_f^\mathsf{T}\Sigma_{\mathbf{X}'}^{-1}\mathtt{J}_f)^+$.

We investigate the two last steps of this method in slightly more detail.

Computation of the derivative matrix. Consider first the Jacobian matrix $\mathtt{J} = \partial\mathbf{X}'/\partial\mathbf{h}$. This matrix has a natural decomposition into blocks so that $\mathtt{J} = (\mathtt{J}_1^\mathsf{T}, \mathtt{J}_2^\mathsf{T}, \ldots, \mathtt{J}_i^\mathsf{T}, \ldots, \mathtt{J}_n^\mathsf{T})^\mathsf{T}$ where $\mathtt{J}_i = \partial\mathbf{x}'_i/\partial\mathbf{h}$. A formula for $\partial\mathbf{x}'_i/\partial\mathbf{h}$ is given in

Samsung/Google, EX1010, p. 163

(4.21–*p*129):

$$\mathtt{J}_i = \partial \mathbf{x}_i'/\partial \mathbf{h} = \frac{1}{w_i'}\begin{bmatrix} \tilde{\mathbf{x}}_i^\mathsf{T} & \mathbf{0}^\mathsf{T} & -x_i'\tilde{\mathbf{x}}_i^\mathsf{T} \\ \mathbf{0}^\mathsf{T} & \tilde{\mathbf{x}}_i^\mathsf{T} & -y_i'\tilde{\mathbf{x}}_i^\mathsf{T} \end{bmatrix} \tag{5.11}$$

where $\tilde{\mathbf{x}}_i^\mathsf{T}$ represents the vector $(x_i, y_i, 1)$.

Stacking these matrices on top of each other for all points $\mathbf{x}_i$ gives the derivative matrix $\partial \mathbf{X}'/\partial \mathbf{h}$. An important case is when the image measurements $\mathbf{x}_i'$ are independent random vectors. In this case $\Sigma = \mathrm{diag}(\Sigma_1, \ldots, \Sigma_n)$ where each Σ_i is the 2×2 covariance matrix of the i-th measured point $\mathbf{x}_i'$. Then one computes

$$\Sigma_\mathbf{h} = (\mathtt{J}^\mathsf{T}\Sigma_{\mathbf{X}'}^{-1}\mathtt{J})^+ = \left(\sum_i \mathtt{J}_i^\mathsf{T}\Sigma_i^{-1}\mathtt{J}_i\right)^+. \tag{5.12}$$

Example 5.13. We consider the simple numerical example of a point correspondence containing just 4 points as follows:

$$\begin{aligned} \mathbf{x}_1 = (1,0)^\mathsf{T} &\leftrightarrow (1,0)^\mathsf{T} = \mathbf{x}_1' \\ \mathbf{x}_2 = (0,1)^\mathsf{T} &\leftrightarrow (0,1)^\mathsf{T} = \mathbf{x}_2' \\ \mathbf{x}_3 = (-1,0)^\mathsf{T} &\leftrightarrow (-1,0)^\mathsf{T} = \mathbf{x}_3' \\ \mathbf{x}_4 = (0,-1)^\mathsf{T} &\leftrightarrow (0,-1)^\mathsf{T} = \mathbf{x}_4' \end{aligned}$$

namely, the identity map on the points of a projective basis. We assume that points $\mathbf{x}_i$ are known exactly, and points $\mathbf{x}_i'$ have one pixel standard deviation in each coordinate direction. This means that the covariance matrix $\Sigma_{\mathbf{x}_i'}$ is the identity.

Obviously, the computed homography will be the identity map. For simplicity we normalize (scale it) so that it is indeed the identity matrix, and hence $\|\mathtt{H}\|^2 = 3$ instead of the usual normalization $\|\mathtt{H}\| = 1$. In this case, all the w_i' in (5.11) are equal to 1. The matrix J is easily computed from (5.11) to equal

$$\mathtt{J} = \left[\begin{array}{ccc|ccc|ccc} 1 & 0 & 1 & 0 & 0 & 0 & -1 & 0 & -1 \\ 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 \\ \hline 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 0 & -1 & -1 \\ \hline -1 & 0 & 1 & 0 & 0 & 0 & -1 & 0 & 1 \\ 0 & 0 & 0 & -1 & 0 & 1 & 0 & 0 & 0 \\ \hline 0 & -1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & -1 & 1 & 0 & -1 & 1 \end{array}\right].$$

Then

$$\mathtt{J}^\mathsf{T}\mathtt{J} = \left[\begin{array}{ccc|ccc|ccc} 2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -2 \\ 0 & 2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 4 & 0 & 0 & 0 & -2 & 0 & 0 \\ \hline 0 & 0 & 0 & 2 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 2 & 0 & 0 & 0 & -2 \\ 0 & 0 & 0 & 0 & 0 & 4 & 0 & -2 & 0 \\ \hline 0 & 0 & -2 & 0 & 0 & 0 & 2 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & -2 & 0 & 2 & 0 \\ -2 & 0 & 0 & 0 & -2 & 0 & 0 & 0 & 4 \end{array}\right]. \tag{5.13}$$

Samsung/Google, EX1010, p. 164

To take the pseudo-inverse of this matrix, we may use (5.9) where A is a matrix with columns spanning the tangent plane to the constraint surface. Since H is computed subject to the condition $\|\mathtt{H}\|^2 = 3$, which represents a hypersphere, the constraint surface is perpendicular to the vector $\mathbf{h}$ corresponding to the computed homography H. A Householder matrix A (see section A4.1.2(p580)) corresponding to the vector $\mathbf{h}$ has the property that $\mathtt{A}\mathbf{h} = (0, \ldots, 0, 1)^\mathsf{T}$, so the first 8 columns of A (denoted $\mathtt{A}_1$)are perpendicular to $\mathbf{h}$ as required. This allows the pseudo-inverse to be computed exactly without using SVD. Applying (5.9) the pseudo-inverse is computed to be

$$\Sigma_\mathbf{h} = (\mathtt{J}^\mathsf{T}\mathtt{J})^{+\mathtt{A}_1} = \mathtt{A}_1(\mathtt{A}_1^\mathsf{T}(\mathtt{J}^\mathsf{T}\mathtt{J})\mathtt{A}_1)^{-1}\mathtt{A}_1^\mathsf{T} = \frac{1}{18}\left[\begin{array}{ccc|ccc|ccc} 5 & 0 & 0 & 0 & -4 & 0 & 0 & 0 & -1 \\ 0 & 9 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 9 & 0 & 0 & 0 & 9 & 0 & 0 \\ \hline 0 & 0 & 0 & 9 & 0 & 0 & 0 & 0 & 0 \\ -4 & 0 & 0 & 0 & 5 & 0 & 0 & 0 & -1 \\ 0 & 0 & 0 & 0 & 0 & 9 & 0 & 9 & 0 \\ \hline 0 & 0 & 9 & 0 & 0 & 0 & 18 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 9 & 0 & 18 & 0 \\ -1 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 2 \end{array}\right]. \tag{5.14}$$

The diagonals give the individual variances of the entries of H. △

This computed covariance is used to assess the accuracy of point transfer in example 5.14.

5.2.5 Error in both images

In the case of error in both images, computation of the covariance of the transformation is a bit more complicated. As seen in section 4.2.7(p101), one may define a set of $2n+8$ parameters, where 8 parameters describe the transformation matrix and $2n$ parameters $\hat{\mathbf{x}}_i$ represent estimates of the points in the first image. More conveniently, one may over-parametrize by using 9 parameters for the transformation H. The Jacobian matrix naturally splits up into two parts as $\mathtt{J} = [\mathtt{A} \mid \mathtt{B}]$ where A and B are the derivatives with respect to the camera parameters and the points $\mathbf{x}_i$ respectively. Applying (5.10) one computes

$$\mathtt{J}^\mathsf{T}\Sigma_\mathbf{x}^{-1}\mathtt{J} = \left[\begin{array}{cc} \mathtt{A}^\mathsf{T}\Sigma_\mathbf{x}^{-1}\mathtt{A} & \mathtt{A}^\mathsf{T}\Sigma_\mathbf{x}^{-1}\mathtt{B} \\ \mathtt{B}^\mathsf{T}\Sigma_\mathbf{x}^{-1}\mathtt{A} & \mathtt{B}^\mathsf{T}\Sigma_\mathbf{x}^{-1}\mathtt{B} \end{array}\right].$$

The pseudo-inverse of this matrix is the covariance of the parameter set and the top-left block of this pseudo-inverse is the covariance of the entries of H. A detailed discussion of this is given in section A6.4.1(p608), where it is shown how one can make use of the block structure of the Jacobian to simplify the computation.

In example 5.13 on estimating the covariance of H from four points in the previous section, the covariance turns out to be $\Sigma_\mathbf{h} = 2(\mathtt{J}^\mathsf{T}\Sigma_\mathbf{x}^{-1}\mathtt{J})^+$, namely twice the covariance computed for noise in one image only. This assumes that points are measured with the same covariance in both images. This simple relationship between the covariances in the one and two-image cases does not generally hold.

Samsung/Google, EX1010, p. 165

5.2.6 Using the covariance matrix in point transfer

Once one has the covariance, one may compute the uncertainty associated with a given point transfer. Consider a new point $\mathbf{x}$ in the first image, not used in the computation of the transformation, $\mathtt{H}$. The corresponding point in the second image is $\mathbf{x}' = \mathtt{H}\mathbf{x}$. However, because of the uncertainty in the estimation of $\mathtt{H}$, the correct location of the point $\mathbf{x}'$ will also have associated uncertainty. One may compute this uncertainty from the covariance matrix of $\mathtt{H}$.

The covariance matrix for the point $\mathbf{x}'$ is given by the formula

$$\Sigma_{\mathbf{x}'} = \mathtt{J}_{\mathbf{h}} \Sigma_{\mathbf{h}} \mathtt{J}_{\mathbf{h}}^{\mathsf{T}} \tag{5.15}$$

where $\mathtt{J}_{\mathbf{h}} = \partial \mathbf{x}'/\partial \mathbf{h}$. A formula for $\partial \mathbf{x}'/\partial \mathbf{h}$ is given in (4.21–p129).

If in addition, the point $\mathbf{x}$ itself is measured with some uncertainty, then one has instead

$$\Sigma_{\mathbf{x}'} = \mathtt{J}_{\mathbf{h}} \Sigma_{\mathbf{h}} \mathtt{J}_{\mathbf{h}}^{\mathsf{T}} + \mathtt{J}_{\mathbf{x}} \Sigma_{\mathbf{x}} \mathtt{J}_{\mathbf{x}}^{\mathsf{T}} \tag{5.16}$$

assuming that there is no cross-correlation between $\mathbf{x}$ and $\mathbf{h}$, which is reasonable, since point $\mathbf{x}$ is assumed to be a new point not used in the computation of the transformation $\mathtt{H}$. A formula for the Jacobian matrix $\mathtt{J}_{\mathbf{x}} = \partial \mathbf{x}'/\partial \mathbf{x}$ is given in (4.20–p129).

The covariance matrix $\Sigma_{\mathbf{x}'}$ given by (5.15) is expressed in terms of the covariance matrix $\Sigma_{\mathbf{h}}$ of the transformation $\mathtt{H}$. We have seen that this covariance matrix $\Sigma_{\mathbf{h}}$ depends on the particular constraint used in estimating $\mathtt{H}$, according to (5.9). It may therefore appear that $\Sigma_{\mathbf{x}'}$ also depends on the particular method used to constrain $\mathtt{H}$. It may however be verified that these formulae are independent of the particular constraint $\mathtt{A}$ used to compute the covariance matrix $\Sigma_{\mathbf{P}} = (\mathtt{J}^{\mathsf{T}} \Sigma_{\mathbf{X}}^{-1} \mathtt{J})^{+\mathtt{A}}$.

Example 5.14. To continue example 5.13, let the computed 2D homography $\mathtt{H}$ be given by the identity matrix, with covariance matrix $\Sigma_{\mathbf{h}}$ as in (5.14). Consider an arbitrary point $(\mathbf{x}, \mathbf{y})$ mapped to the point $\mathbf{x}' = \mathtt{H}\mathbf{x}$. In this case the covariance matrix $\Sigma_{\mathbf{x}'} = \mathtt{J}_{\mathbf{h}} \Sigma_{\mathbf{h}} \mathtt{J}_{\mathbf{h}}^{\mathsf{T}}$ may be computed symbolically to equal

$$\Sigma_{\mathbf{x}'} = \begin{bmatrix} \sigma_{x'x'} & \sigma_{x'y'} \\ \sigma_{x'y'} & \sigma_{y'y'} \end{bmatrix} = \frac{1}{4} \begin{bmatrix} 2 - x^2 + x^4 + y^2 + x^2 y^2 & xy(x^2 + y^2 - 2) \\ xy(x^2 + y^2 - 2) & 2 - y^2 + y^4 + x^2 + x^2 y^2 \end{bmatrix}.$$

Note that $\sigma_{x'x'}$ and $\sigma_{y'y'}$ are even functions of x and y, whereas $\sigma_{x'y'}$ is an odd function. This is a consequence of the symmetry about the x and y axes of the point set used to compute $\mathtt{H}$. Also note that $\sigma_{x'x'}$ and $\sigma_{y'y'}$ differ by swapping x and y, which is a further consequence of the symmetry of the defining point set.

As may be seen, the variance $\sigma_{x'x'}$ varies as the fourth power of x, and hence the standard deviation varies as the square. This shows that extrapolating the values of transformed points $\mathbf{x}' = \mathtt{H}\mathbf{x}$ far beyond the set of points used to compute $\mathtt{H}$ is not reliable. More specifically, the RMS uncertainty of the position of $\mathbf{x}'$ is equal to $(\sigma_{x'x'} + \sigma_{y'y'})^{1/2} = \sqrt{\mathrm{trace}(\Sigma_{\mathbf{x}'})}$ which one finds is equal to $(1 + (x^2 + y^2)^2)^{1/2} = (1 + r^4)^{1/2}$, where r is the radial distance from the origin. Note the interesting fact that the RMS error is only dependent on the radial distance. In fact, one may verify that the probability distribution for point $\mathbf{x}'$ depends only on the radial distance of $\mathbf{x}'$, its

Samsung/Google, EX1010, p. 166



Fig. 5.5. *RMS error in the position of a projected point* $\mathbf{x}'$ *as a function of radial distance of* $\mathbf{x}'$ *from the origin. The homography* H *is computed from 4 evenly spaced points on a unit circle around the origin with errors in the second image only. The RMS error is proportional to the assumed error in the points used to compute* H*, and the vertical axis is calibrated in terms of this assumed error.*

two principal axes pointing radially and tangentially. Figure 5.5 shows the graph of the RMS error in $\mathbf{x}'$ as a function of r. △

This example has computed the covariance of a transferred point in the minimal case of four point correspondences. For more than four correspondences, the situation is not substantially different. Extrapolation beyond the set of points used to compute the homography is unreliable. In fact, one may show that if H is computed from n points evenly spaced around a unit circle (instead of 4 as in the computation above) then the RMS error is equal to $\sigma_{x'x'} + \sigma_{y'y'} = 4(1 + r^4)/n$, so the error exhibits the same quadratic growth.

5.3 Monte Carlo estimation of covariance

The method of estimating covariance discussed in the previous sections has relied on an assumption of linearity. In other words, it has been assumed that the surface $f(\mathbf{h})$ is locally flat in the vicinity of the estimated point, at least over a region corresponding to the approximate extent of the noise distribution. It has also been assumed that the method of estimation of the transformation H was the Maximum Likelihood Estimate. If the surface is not entirely flat then the estimate of covariance may be incorrect. In addition, a particular estimation method may be inferior to the ML estimate, thereby introducing additional uncertainty in the values of the estimated transformation H.

A general (though expensive) method of getting an estimate of the covariance is by exhaustive simulation. Assuming that the noise is drawn from a given noise distribution, one starts with a set of point matches corresponding perfectly to a given transformation. One then adds noise to the points and computes the corresponding transformation using the chosen estimation procedure. The covariance of the transformation H or a further transferred point is then computed statistically from multiple trials with noise drawn from the assumed noise distribution. This is illustrated for the case of the identity mapping in figure 5.6.

Both the analytical and the Monte Carlo methods of estimating covariance of the transformation H may be applied to the estimation of covariance from real data for which one does not know the true value of H. From the given data, an estimate of H and the corresponding true values of the points $\mathbf{x}'_i$ and $\mathbf{x}_i$ are computed. Then the

Samsung/Google, EX1010, p. 167

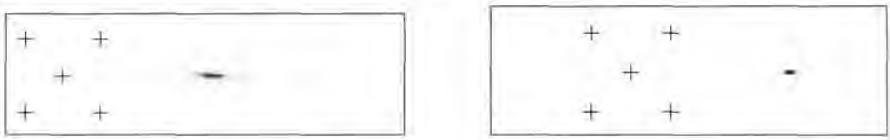


Fig. 5.6. *Transfer of a point under the identity mapping for the normalized and unnormalized DLT algorithm. See also figure 4.4(p109) for further explanation.*

covariance is computed as if the estimated values were the true values of the matched data points and the transformation. The resulting covariance matrix is assumed to be the covariance of the true transformation. This identification is based on the assumption that the true values of the data points are sufficiently close to the estimated values that the covariance matrix is essentially unaffected.

5.4 Closure

An extended discussion of bias and variance of estimated parameters is given in appendix 3(*p*568).

5.4.1 The literature

The derivations throughout this chapter have been considerably simplified by only using first-order Taylor expansions, and assuming Gaussian error distributions. Similar ideas (ML, covariance ...) can be developed for other distributions by using the Fisher Information matrix. Related reading may be found in Kanatani [Kanatani-96], Press *et al.* [Press-88], and other statistical textbooks.

Criminisi *et al.* [Criminisi-99b] give many examples of the computed covariances in point transfer as the correspondences used to determine the homography vary in number and position.

5.4.2 Notes and exercises

(i) Consider the problem of computing a best line fit to a set of 2D points in the plane using orthogonal regression. Suppose that N points are measured with independent standard deviations of σ in each coordinate. What is the expected RMS distance of each point from a fitted line? **Answer :** $\sigma\left((n-2)/n\right)^{1/2}$.

(ii) (Harder) : In section 18.5.2(*p*450) a method is given for computing a projective reconstruction from a set of $n+4$ point correspondences across m views, where 4 of the point correspondences are presumed to be known to be from a plane. Suppose the 4 planar correspondences are known exactly, and the other n image points are measured with 1 pixel error (each coordinate in each image). What is the expected residual error of $\|\mathbf{x}_j^i - \widehat{\mathrm{P}}^i\mathbf{X}_j\|$?

Samsung/Google, EX1010, p. 168

Part I

Camera Geometry and Single View Geometry



The Cyclops, c. 1914 (oil on canvas) by Odilon Redon (1840-1916)
Rijksmuseum Kroller-Muller, Otterlo, Netherlands /Bridgeman Art Library

Samsung/Google, EX1010, p. 169

Outline

This part of the book concentrates on the geometry of a single perspective camera. It contains three chapters.

Chapter 6 describes the projection of 3D scene space onto a 2D image plane. The camera mapping is represented by a matrix, and in the case of mapping points it is a 3×4 matrix $\mathtt{P}$ which maps from homogeneous coordinates of a world point in 3-space to homogeneous coordinates of the imaged point on the image plane. This matrix has in general 11 degrees of freedom, and the properties of the camera, such as its centre and focal length, may be extracted from it. In particular the internal camera parameters, such as the focal length and aspect ratio, are packaged in a 3×3 matrix $\mathtt{K}$ which is obtained from $\mathtt{P}$ by a simple decomposition. There are two particularly important classes of camera matrix: finite cameras, and cameras with their centre at infinity such as the *affine camera* which represents parallel projection.

Chapter 7 describes the estimation of the camera matrix $\mathtt{P}$, given the coordinates of a set of corresponding world and image points. The chapter also describes how constraints on the camera may be efficiently incorporated into the estimation, and a method of correcting for radial lens distortion.

Chapter 8 has three main topics. First, it covers the action of a camera on geometric objects other than finite points. These include lines, conics, quadrics and points at infinity. The image of points/lines at infinity are vanishing points/lines. The second topic is camera calibration, in which the internal parameters $\mathtt{K}$ of the camera matrix are computed, without computing the entire matrix $\mathtt{P}$. In particular the relation of the internal parameters to the image of the absolute conic is described, and the calibration of a camera from vanishing points and vanishing lines. The final topic is the *calibrating conic*. This is a simple geometric device for visualizing camera calibration.

Samsung/Google, EX1010, p. 170

6

Camera Models

A camera is a mapping between the 3D world (object space) and a 2D image. The principal camera of interest in this book is *central projection*. This chapter develops a number of camera *models* which are matrices with particular properties that represent the camera mapping.

It will be seen that all cameras modelling central projection are specializations of the *general projective camera*. The anatomy of this most general camera model is examined using the tools of projective geometry. It will be seen that geometric entities of the camera, such as the projection centre and image plane, can be computed quite simply from its matrix representation. Specializations of the general projective camera inherit its properties, for example their geometry is computed using the same algebraic expressions.

The specialized models fall into two major classes – those that model cameras with a finite centre, and those that model cameras with centre "at infinity". Of the cameras at infinity the *affine camera* is of particular importance because it is the natural generalization of parallel projection.

This chapter is principally concerned with the projection of points. The action of a camera on other geometric entities, such as lines, is deferred until chapter 8.

6.1 Finite cameras

In this section we start with the most specialized and simplest camera model, which is the basic pinhole camera, and then progressively generalize this model through a series of gradations.

The models we develop are principally designed for CCD like sensors, but are also applicable to other cameras, for example X-ray images, scanned photographic negatives, scanned photographs from enlarged negatives, etc.

The basic pinhole model. We consider the central projection of points in space onto a plane. Let the centre of projection be the origin of a Euclidean coordinate system, and consider the plane $Z = f$, which is called the *image plane* or *focal plane*. Under the pinhole camera model, a point in space with coordinates $\mathbf{X} = (X, Y, Z)^\top$ is mapped to the point on the image plane where a line joining the point $\mathbf{X}$ to the centre of projection meets the image plane. This is shown in figure 6.1. By similar triangles, one quickly

Samsung/Google, EX1010, p. 171

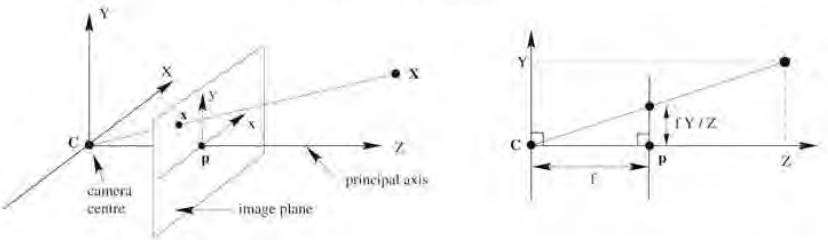


Fig. 6.1. **Pinhole camera geometry.** C *is the camera centre and* p *the principal point. The camera centre is here placed at the coordinate origin. Note the image plane is placed in front of the camera centre.*

computes that the point $(X, Y, Z)^\top$ is mapped to the point $(fX/Z, fY/Z, f)^\top$ on the image plane. Ignoring the final image coordinate, we see that

$$(X, Y, Z)^\top \mapsto (fX/Z, fY/Z)^\top \tag{6.1}$$

describes the central projection mapping from world to image coordinates. This is a mapping from Euclidean 3-space $\mathbb{R}^3$ to Euclidean 2-space $\mathbb{R}^2$.

The centre of projection is called the *camera centre*. It is also known as the *optical centre*. The line from the camera centre perpendicular to the image plane is called the *principal axis* or *principal ray* of the camera, and the point where the principal axis meets the image plane is called the *principal point*. The plane through the camera centre parallel to the image plane is called the *principal plane* of the camera.

Central projection using homogeneous coordinates. If the world and image points are represented by homogeneous vectors, then central projection is very simply expressed as a linear mapping between their homogeneous coordinates. In particular, (6.1) may be written in terms of matrix multiplication as

$$\begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix} \mapsto \begin{pmatrix} fX \\ fY \\ Z \end{pmatrix} = \begin{bmatrix} f & & & 0 \\ & f & & 0 \\ & & 1 & 0 \end{bmatrix} \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix}. \tag{6.2}$$

The matrix in this expression may be written as $\mathrm{diag}(f, f, 1)[\mathtt{I} \mid \mathbf{0}]$ where $\mathrm{diag}(f, f, 1)$ is a diagonal matrix and $[\mathtt{I} \mid \mathbf{0}]$ represents a matrix divided up into a 3×3 block (the identity matrix) plus a column vector, here the zero vector.

We now introduce the notation $\mathbf{X}$ for the world point represented by the homogeneous 4-vector $(X, Y, Z, 1)^\top$, $\mathbf{x}$ for the image point represented by a homogeneous 3-vector, and $\mathtt{P}$ for the 3×4 homogeneous *camera projection matrix*. Then (6.2) is written compactly as

$$\mathbf{x} = \mathtt{P}\mathbf{X}$$

which defines the camera matrix for the pinhole model of central projection as

$$\mathtt{P} = \mathrm{diag}(f, f, 1)\,[\mathtt{I} \mid \mathbf{0}].$$

Samsung/Google, EX1010, p. 172

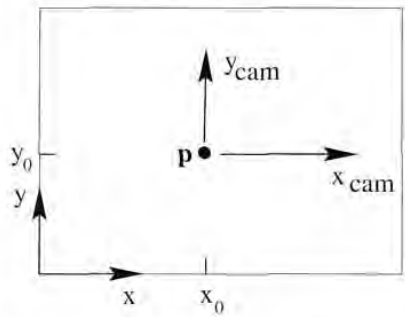


Fig. 6.2. *Image* (x, y) *and camera* $(x_{\text{cam}}, y_{\text{cam}})$ *coordinate systems.*

Principal point offset. The expression (6.1) assumed that the origin of coordinates in the image plane is at the principal point. In practice, it may not be, so that in general there is a mapping

$$(\mathrm{X}, \mathrm{Y}, \mathrm{Z})^\mathsf{T} \mapsto (f\mathrm{X}/\mathrm{Z} + p_x, f\mathrm{Y}/\mathrm{Z} + p_y)^\mathsf{T}$$

where $(p_x, p_y)^\mathsf{T}$ are the coordinates of the principal point. See figure 6.2. This equation may be expressed conveniently in homogeneous coordinates as

$$\begin{pmatrix} \mathrm{X} \\ \mathrm{Y} \\ \mathrm{Z} \\ 1 \end{pmatrix} \mapsto \begin{pmatrix} f\mathrm{X} + \mathrm{Z}p_x \\ f\mathrm{Y} + \mathrm{Z}p_y \\ \mathrm{Z} \end{pmatrix} = \begin{bmatrix} f & & p_x & 0 \\ & f & p_y & 0 \\ & & 1 & 0 \end{bmatrix} \begin{pmatrix} \mathrm{X} \\ \mathrm{Y} \\ \mathrm{Z} \\ 1 \end{pmatrix}. \tag{6.3}$$

Now, writing

$$\mathtt{K} = \begin{bmatrix} f & & p_x \\ & f & p_y \\ & & 1 \end{bmatrix} \tag{6.4}$$

then (6.3) has the concise form

$$\mathbf{x} = \mathtt{K}[\mathtt{I} \mid \mathtt{0}]\mathbf{X}_{\text{cam}}. \tag{6.5}$$

The matrix $\mathtt{K}$ is called the *camera calibration matrix*. In (6.5) we have written $(\mathrm{X}, \mathrm{Y}, \mathrm{Z}, 1)^\mathsf{T}$ as $\mathbf{X}_{\text{cam}}$ to emphasize that the camera is assumed to be located at the origin of a Euclidean coordinate system with the principal axis of the camera pointing straight down the Z-axis, and the point $\mathbf{X}_{\text{cam}}$ is expressed in this coordinate system. Such a coordinate system may be called the *camera coordinate frame*.

Camera rotation and translation. In general, points in space will be expressed in terms of a different Euclidean coordinate frame, known as the *world coordinate frame*. The two coordinate frames are related via a rotation and a translation. See figure 6.3. If $\widetilde{\mathbf{X}}$ is an inhomogeneous 3-vector representing the coordinates of a point in the world coordinate frame, and $\widetilde{\mathbf{X}}_{\text{cam}}$ represents the same point in the camera coordinate frame, then we may write $\widetilde{\mathbf{X}}_{\text{cam}} = \mathtt{R}(\widetilde{\mathbf{X}} - \widetilde{\mathbf{C}})$, where $\widetilde{\mathbf{C}}$ represents the coordinates of the camera

Samsung/Google, EX1010, p. 173

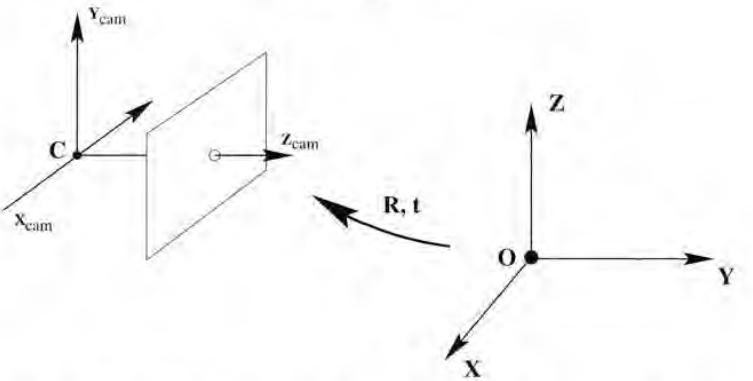


Fig. 6.3. *The Euclidean transformation between the world and camera coordinate frames.*

centre in the world coordinate frame, and R is a 3×3 rotation matrix representing the orientation of the camera coordinate frame. This equation may be written in homogeneous coordinates as

$$\mathbf{X}_{\text{cam}} = \begin{bmatrix} \mathtt{R} & -\mathtt{R}\widetilde{\mathbf{C}} \\ 0 & 1 \end{bmatrix} \begin{pmatrix} \mathrm{X} \\ \mathrm{Y} \\ \mathrm{Z} \\ 1 \end{pmatrix} = \begin{bmatrix} \mathtt{R} & -\mathtt{R}\widetilde{\mathbf{C}} \\ 0 & 1 \end{bmatrix} \mathbf{X}. \tag{6.6}$$

Putting this together with (6.5) leads to the formula

$$\mathbf{x} = \mathtt{K}\mathtt{R}[\mathtt{I} \mid -\widetilde{\mathbf{C}}]\mathbf{X} \tag{6.7}$$

where X is now in a world coordinate frame. This is the general mapping given by a pinhole camera. One sees that a general pinhole camera, $\mathtt{P} = \mathtt{K}\mathtt{R}[\mathtt{I} \mid -\widetilde{\mathbf{C}}]$, has 9 degrees of freedom: 3 for K (the elements f, p_x, p_y), 3 for R, and 3 for $\widetilde{\mathbf{C}}$. The parameters contained in K are called the *internal* camera parameters, or the *internal orientation* of the camera. The parameters of R and $\widetilde{\mathbf{C}}$ which relate the camera orientation and position to a world coordinate system are called the *external* parameters or the *exterior orientation*.

It is often convenient not to make the camera centre explicit, and instead to represent the world to image transformation as $\widetilde{\mathbf{X}}_{\text{cam}} = \mathtt{R}\widetilde{\mathbf{X}} + \mathbf{t}$. In this case the camera matrix is simply

$$\mathtt{P} = \mathtt{K}[\mathtt{R} \mid \mathbf{t}] \tag{6.8}$$

where from (6.7) $\mathbf{t} = -\mathtt{R}\widetilde{\mathbf{C}}$.

CCD cameras. The pinhole camera model just derived assumes that the image coordinates are Euclidean coordinates having equal scales in both axial directions. In the case of CCD cameras, there is the additional possibility of having non-square pixels. If image coordinates are measured in pixels, then this has the extra effect of introducing unequal scale factors in each direction. In particular if the number of pixels per unit

Samsung/Google, EX1010, p. 174

distance in image coordinates are m_x and m_y in the x and y directions, then the transformation from world coordinates to pixel coordinates is obtained by multiplying (6.4) on the left by an extra factor $\mathrm{diag}(m_x, m_y, 1)$. Thus, the general form of the calibration matrix of a CCD camera is

$$\mathtt{K} = \begin{bmatrix} \alpha_x & & x_0 \\ & \alpha_y & y_0 \\ & & 1 \end{bmatrix} \tag{6.9}$$

where $\alpha_x = f m_x$ and $\alpha_y = f m_y$ represent the focal length of the camera in terms of pixel dimensions in the x and y direction respectively. Similarly, $\tilde{\mathbf{x}}_0 = (x_0, y_0)$ is the principal point in terms of pixel dimensions, with coordinates $x_0 = m_x p_x$ and $y_0 = m_y p_y$. A CCD camera thus has 10 degrees of freedom.

Finite projective camera. For added generality, we can consider a calibration matrix of the form

$$\mathtt{K} = \begin{bmatrix} \alpha_x & s & x_0 \\ & \alpha_y & y_0 \\ & & 1 \end{bmatrix}. \tag{6.10}$$

The added parameter s is referred to as the *skew* parameter. The skew parameter will be zero for most normal cameras. However, in certain unusual instances which are described in section 6.2.4, it can take non-zero values.

A camera

$$\mathtt{P} = \mathtt{KR}[\mathtt{I} \mid -\widetilde{\mathbf{C}}] \tag{6.11}$$

for which the calibration matrix K is of the form (6.10) will be called a *finite projective* camera. A finite projective camera has 11 degrees of freedom. This is the same number of degrees of freedom as a 3×4 matrix, defined up to an arbitrary scale.

Note that the left hand 3×3 submatrix of P, equal to KR, is non-singular. Conversely, any 3×4 matrix P for which the left hand 3×3 submatrix is non-singular is the camera matrix of some finite projective camera, because P can be decomposed as $\mathtt{P} = \mathtt{KR}[\mathtt{I} \mid -\widetilde{\mathbf{C}}]$. Indeed, letting M be the left 3×3 submatrix of P, one decomposes M as a product $\mathtt{M} = \mathtt{KR}$ where K is upper-triangular of the form (6.10) and R is a rotation matrix. This decomposition is essentially the RQ matrix decomposition, described in section A4.1.1(*p*579), of which more will be said in section 6.2.4. The matrix P can therefore be written $\mathtt{P} = \mathtt{M}[\mathtt{I} \mid \mathtt{M}^{-1}\mathbf{p}_4] = \mathtt{KR}[\mathtt{I} \mid -\widetilde{\mathbf{C}}]$ where $\mathbf{p}_4$ is the last column of P. In short

- *The set of camera matrices of finite projective cameras is identical with the set of homogeneous 3×4 matrices for which the left hand 3×3 submatrix is non-singular.*

General projective cameras. The final step in our hierarchy of projective cameras is to remove the non-singularity restriction on the left hand 3×3 submatrix. A *general projective* camera is one represented by an arbitrary homogeneous 3×4 matrix of rank 3. It has 11 degrees of freedom. The rank 3 requirement arises because if the rank is

Samsung/Google, EX1010, p. 175

Camera centre. The camera centre is the 1-dimensional right null-space $\mathbf{C}$ of $\mathtt{P}$, i.e. $\mathtt{P}\mathbf{C} = \mathbf{0}$.

- **Finite camera** ($\mathtt{M}$ is not singular) $\mathbf{C} = \begin{pmatrix} -\mathtt{M}^{-1}\mathbf{p}_4 \\ 1 \end{pmatrix}$
- **Camera at infinity** ($\mathtt{M}$ is singular) $\mathbf{C} = \begin{pmatrix} \mathbf{d} \\ 0 \end{pmatrix}$ where $\mathbf{d}$ is the null 3-vector of $\mathtt{M}$, i.e. $\mathtt{M}\mathbf{d} = \mathbf{0}$.

Column points. For $i = 1, \ldots, 3$, the column vectors $\mathbf{p}_i$ are vanishing points in the image corresponding to the X, Y and Z axes respectively. Column $\mathbf{p}_4$ is the image of the coordinate origin.

Principal plane. The principal plane of the camera is $\mathbf{P}^3$, the last row of $\mathtt{P}$.

Axis planes. The planes $\mathbf{P}^1$ and $\mathbf{P}^2$ (the first and second rows of $\mathtt{P}$) represent planes in space through the camera centre, corresponding to points that map to the image lines $x = 0$ and $y = 0$ respectively.

Principal point. The image point $\mathbf{x}_0 = \mathtt{M}\mathbf{m}^3$ is the principal point of the camera, where $\mathbf{m}^{3\mathsf{T}}$ is the third row of $\mathtt{M}$.

Principal ray. The principal ray (axis) of the camera is the ray passing through the camera centre $\mathbf{C}$ with direction vector $\mathbf{m}^{3\mathsf{T}}$. The principal axis vector $\mathbf{v} = \det(\mathtt{M})\mathbf{m}^3$ is directed towards the front of the camera.

Table 6.1. *Summary of the properties of a projective camera* $\mathtt{P}$. *The matrix is represented by the block form* $\mathtt{P} = [\mathtt{M} \mid \mathbf{p}_4]$.

less than this then the range of the matrix mapping will be a line or point and not the whole plane; in other words not a 2D image.

6.2 The projective camera

A general projective camera $\mathtt{P}$ maps world points $\mathbf{X}$ to image points $\mathbf{x}$ according to $\mathbf{x} = \mathtt{P}\mathbf{X}$. Building on this mapping we will now dissect the camera model to reveal how geometric entities, such as the camera centre, are encoded. Some of the properties that we consider will apply only to finite projective cameras and their specializations, whilst others will apply to general cameras. The distinction will be evident from the context. The derived properties of the camera are summarized in table 6.1.

6.2.1 Camera anatomy

A general projective camera may be decomposed into blocks according to $\mathtt{P} = [\mathtt{M} \mid \mathbf{p}_4]$, where $\mathtt{M}$ is a 3×3 matrix. It will be seen that if $\mathtt{M}$ is non-singular, then this is a finite camera, otherwise it is not.

Camera centre. The matrix $\mathtt{P}$ has a 1-dimensional right null-space because its rank is 3, whereas it has 4 columns. Suppose the null-space is generated by the 4-vector $\mathbf{C}$, that is $\mathtt{P}\mathbf{C} = \mathbf{0}$. It will now be shown that $\mathbf{C}$ is the camera centre, represented as a homogeneous 4-vector.

Consider the line containing $\mathbf{C}$ and any other point $\mathbf{A}$ in 3-space. Points on this line may be represented by the join

$$\mathbf{X}(\lambda) = \lambda\mathbf{A} + (1 - \lambda)\mathbf{C} \ .$$

Samsung/Google, EX1010, p. 176

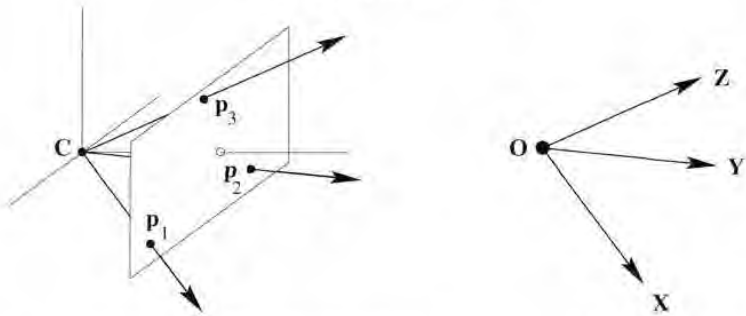


Fig. 6.4. *The three image points defined by the columns* $\mathbf{p}_i, i = 1, \ldots, 3$, *of the projection matrix are the vanishing points of the directions of the world axes.*

Under the mapping $\mathbf{x} = \mathtt{P}\mathbf{X}$ points on this line are projected to

$$\mathbf{x} = \mathtt{P}\mathbf{X}(\lambda) = \lambda\mathtt{P}\mathbf{A} + (1-\lambda)\mathtt{P}\mathbf{C} = \lambda\mathtt{P}\mathbf{A}$$

since $\mathtt{P}\mathbf{C} = \mathbf{0}$. That is all points on the line are mapped to the same image point $\mathtt{P}\mathbf{A}$, which means that the line must be a ray through the camera centre. It follows that $\mathbf{C}$ is the homogeneous representation of the camera centre, since for all choices of $\mathbf{A}$ the line $\mathbf{X}(\lambda)$ is a ray through the camera centre.

This result is not unexpected since the image point $(0,0,0)^\mathsf{T} = \mathtt{P}\mathbf{C}$ is not defined, and the camera centre is the unique point in space for which the image is undefined. In the case of finite cameras the result may be established directly, since $\mathbf{C} = (\widetilde{\mathbf{C}}^\mathsf{T}, 1)^\mathsf{T}$ is clearly the null-vector of $\mathtt{P} = \mathtt{K}\mathtt{R}[\mathtt{I} \mid -\widetilde{\mathbf{C}}]$. The result is true even in the case where the first 3×3 submatrix $\mathtt{M}$ of $\mathtt{P}$ is singular. In this singular case, though, the null-vector has the form $\mathbf{C} = (\mathbf{d}^\mathsf{T}, 0)^\mathsf{T}$ where $\mathtt{M}\mathbf{d} = \mathbf{0}$. The camera centre is then a point at infinity. Camera models of this class are discussed in section 6.3.

Column vectors. The columns of the projective camera are 3-vectors which have a geometric meaning as particular image points. With the notation that the columns of $\mathtt{P}$ are $\mathbf{p}_i, i = 1, \ldots, 4$, then $\mathbf{p}_1, \mathbf{p}_2, \mathbf{p}_3$ are the vanishing points of the world coordinate X, Y and Z axes respectively. This follows because these points are the images of the axes' directions. For example the x-axis has direction $\mathbf{D} = (1,0,0,0)^\mathsf{T}$, which is imaged at $\mathbf{p}_1 = \mathtt{P}\mathbf{D}$. See figure 6.4. The column $\mathbf{p}_4$ is the image of the world origin.

Row vectors. The rows of the projective camera (6.12) are 4-vectors which may be interpreted geometrically as particular world planes. These planes are examined next. We introduce the notation that the rows of $\mathtt{P}$ are $\mathbf{P}^{i\mathsf{T}}$ so that

$$\mathtt{P} = \begin{bmatrix} p_{11} & p_{12} & p_{13} & p_{14} \\ p_{21} & p_{22} & p_{23} & p_{24} \\ p_{31} & p_{32} & p_{33} & p_{34} \end{bmatrix} = \begin{bmatrix} \mathbf{P}^{1\mathsf{T}} \\ \mathbf{P}^{2\mathsf{T}} \\ \mathbf{P}^{3\mathsf{T}} \end{bmatrix}. \tag{6.12}$$

Samsung/Google, EX1010, p. 177

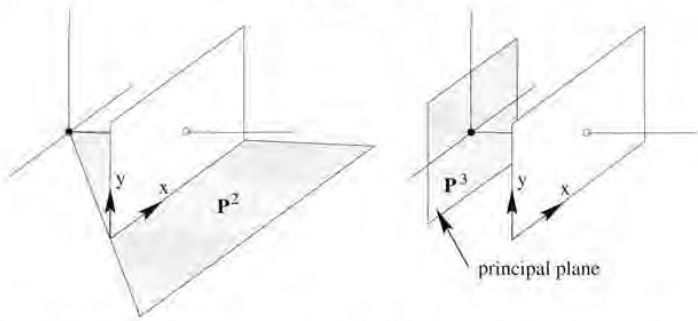


Fig. 6.5. *Two of the three planes defined by the rows of the projection matrix.*

The principal plane. The principal plane is the plane through the camera centre parallel to the image plane. It consists of the set of points $\mathbf{X}$ which are imaged on the line at infinity of the image. Explicitly, $\mathtt{P}\mathbf{X} = (x, y, 0)^\mathsf{T}$. Thus a point lies on the principal plane of the camera if and only if $\mathbf{P}^{3\mathsf{T}}\mathbf{X} = 0$. In other words, $\mathbf{P}^3$ is the vector representing the principal plane of the camera. If $\mathbf{C}$ is the camera centre, then $\mathtt{P}\mathbf{C} = \mathbf{0}$, and so in particular $\mathbf{P}^{3\mathsf{T}}\mathbf{C} = 0$. That is $\mathbf{C}$ lies on the principal plane of the camera.

Axis planes. Consider the set of points $\mathbf{X}$ on the plane $\mathbf{P}^1$. This set satisfies $\mathbf{P}^{1\mathsf{T}}\mathbf{X} = 0$, and so is imaged at $\mathtt{P}\mathbf{X} = (0, y, w)^\mathsf{T}$ which are points on the image y-axis. Again it follows from $\mathtt{P}\mathbf{C} = \mathbf{0}$ that $\mathbf{P}^{1\mathsf{T}}\mathbf{C} = 0$ and so $\mathbf{C}$ lies also on the plane $\mathbf{P}^1$. Consequently the plane $\mathbf{P}^1$ is defined by the camera centre and the line $x = 0$ in the image. Similarly the plane $\mathbf{P}^2$ is defined by the camera centre and the line $y = 0$.

Unlike the principal plane $\mathbf{P}^3$, the axis planes $\mathbf{P}^1$ and $\mathbf{P}^2$ are dependent on the image x- and y-axes, i.e. on the choice of the image coordinate system. Thus they are less tightly coupled to the natural camera geometry than the principal plane. In particular the line of intersection of the planes $\mathbf{P}^1$ and $\mathbf{P}^2$ is a line joining the camera centre and image origin, i.e. the back-projection of the image origin. This line will not coincide in general with the camera principal axis. The planes arising from $\mathbf{P}^i$ are illustrated in figure 6.5.

The camera centre $\mathbf{C}$ lies on all three planes, and since these planes are distinct (as the $\mathtt{P}$ matrix has rank 3) it must lie on their intersection. Algebraically, the condition for the centre to lie on all three planes is $\mathtt{P}\mathbf{C} = \mathbf{0}$ which is the original equation for the camera centre given above.

The principal point. The principal axis is the line passing through the camera centre $\mathbf{C}$, with direction perpendicular to the principal plane $\mathbf{P}^3$. The axis intersects the image plane at the principal point. We may determine this point as follows. In general, the normal to a plane $\boldsymbol{\pi} = (\pi_1, \pi_2, \pi_3, \pi_4)^\mathsf{T}$ is the vector $(\pi_1, \pi_2, \pi_3)^\mathsf{T}$. This may alternatively be represented by a point $(\pi_1, \pi_2, \pi_3, 0)^\mathsf{T}$ on the plane at infinity. In the case of the principal plane $\mathbf{P}^3$ of the camera, this point is $(p_{31}, p_{32}, p_{33}, 0)^\mathsf{T}$, which we denote $\widehat{\mathbf{P}}^3$. Projecting that point using the camera matrix $\mathtt{P}$ gives the principal point of the

Samsung/Google, EX1010, p. 178

camera $\mathtt{P}\widehat{\mathbf{P}}^3$. Note that only the left hand 3×3 part of $\mathtt{P} = [\mathtt{M} \mid \mathbf{p}_4]$ is involved in this formula. In fact the principal point is computed as $\mathbf{x}_0 = \mathtt{M}\mathbf{m}^3$ where $\mathbf{m}^{3\mathsf{T}}$ is the third row of $\mathtt{M}$.

The principal axis vector. Although any point $\mathbf{X}$ not on the principal plane may be mapped to an image point according to $\mathbf{x} = \mathtt{P}\mathbf{X}$, in reality only half the points in space, those that lie in front of the camera, may be seen in an image. Let $\mathtt{P}$ be written as $\mathtt{P} = [\mathtt{M} \mid \mathbf{p}_4]$. It has just been seen that the vector $\mathbf{m}^3$ points in the direction of the principal axis. We would like to define this vector in such a way that it points in the direction towards the front of the camera (the *positive* direction). Note however that $\mathtt{P}$ is only defined up to sign. This leaves an ambiguity as to whether $\mathbf{m}^3$ or $-\mathbf{m}^3$ points in the positive direction. We now proceed to resolve this ambiguity.

We start by considering coordinates with respect to the camera coordinate frame. According to (6.5), the equation for projection of a 3D point to a point in the image is given by $\mathbf{x} = \mathtt{P}_{\text{cam}}\mathbf{X}_{\text{cam}} = \mathtt{K}[\mathtt{I} \mid 0]\mathbf{X}_{\text{cam}}$, where $\mathbf{X}_{\text{cam}}$ is the 3D point expressed in camera coordinates. In this case observe that the vector $\mathbf{v} = \det(\mathtt{M})\mathbf{m}^3 = (0, 0, 1)^\mathsf{T}$ points *towards the front of the camera* in the direction of the principal axis, irrespective of the scaling of $\mathtt{P}_{\text{cam}}$. For example, if $\mathtt{P}_{\text{cam}} \rightarrow k\mathtt{P}_{\text{cam}}$ then $\mathbf{v} \rightarrow k^4\mathbf{v}$ which has the same direction.

If the 3D point is expressed in world coordinates then $\mathtt{P} = k\mathtt{K}[\mathtt{R} \mid -\mathtt{R}\tilde{\mathbf{C}}] = [\mathtt{M} \mid \mathbf{p}_4]$, where $\mathtt{M} = k\mathtt{K}\mathtt{R}$. Since $\det(\mathtt{R}) > 0$ the vector $\mathbf{v} = \det(\mathtt{M})\mathbf{m}^3$ is again unaffected by scaling. In summary,

- $\mathbf{v} = \det(\mathtt{M})\mathbf{m}^3$ *is a vector in the direction of the principal axis, directed towards the front of the camera.*

6.2.2 Action of a projective camera on points

Forward projection. As we have already seen, a general projective camera maps a point in space $\mathbf{X}$ to an image point according to the mapping $\mathbf{x} = \mathtt{P}\mathbf{X}$. Points $\mathbf{D} = (\mathbf{d}^\mathsf{T}, 0)^\mathsf{T}$ on the plane at infinity represent vanishing points. Such points map to

$$\mathbf{x} = \mathtt{P}\mathbf{D} = [\mathtt{M} \mid \mathbf{p}_4]\mathbf{D} = \mathtt{M}\mathbf{d}$$

and thus are only affected by $\mathtt{M}$, the first 3×3 submatrix of $\mathtt{P}$.

Back-projection of points to rays. Given a point $\mathbf{x}$ in an image, we next determine the set of points in space that map to this point. This set will constitute a ray in space passing through the camera centre. The form of the ray may be specified in several ways, depending on how one wishes to represent a line in 3-space. A Plücker representation is postponed until section 8.1.2(p196). Here the line is represented as the join of two points.

We know two points on the ray. These are the camera centre $\mathbf{C}$ (where $\mathtt{P}\mathbf{C} = \mathbf{0}$) and the point $\mathtt{P}^+\mathbf{x}$, where $\mathtt{P}^+$ is the pseudo-inverse of $\mathtt{P}$. The pseudo-inverse of $\mathtt{P}$ is the matrix $\mathtt{P}^+ = \mathtt{P}^\mathsf{T}(\mathtt{P}\mathtt{P}^\mathsf{T})^{-1}$, for which $\mathtt{P}\mathtt{P}^+ = \mathtt{I}$ (see section A5.2(p590)). Point $\mathtt{P}^+\mathbf{x}$ lies

Samsung/Google, EX1010, p. 179

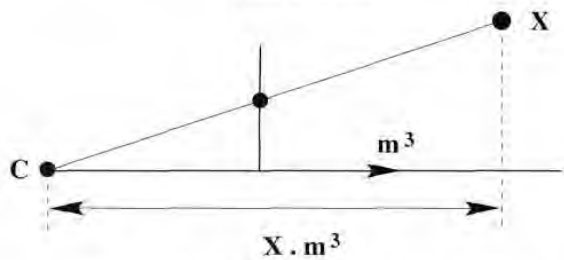


Fig. 6.6. *If the camera matrix* $\mathtt{P} = [\mathtt{M} \mid \mathbf{p}_4]$ *is normalized so that* $\|\mathbf{m}^3\| = 1$ *and* $\det \mathtt{M} > 0$, *and* $\mathbf{x} = w(x, y, 1)^\mathsf{T} = \mathtt{P}\mathbf{X}$, *where* $\mathbf{X} = (\mathtt{X}, \mathtt{Y}, \mathtt{Z}, 1)^\mathsf{T}$, *then* w *is the depth of the point* $\mathbf{X}$ *from the camera centre in the direction of the principal ray of the camera.*

on the ray because it projects to $\mathbf{x}$, since $\mathtt{P}(\mathtt{P}^+\mathbf{x}) = \mathtt{I}\mathbf{x} = \mathbf{x}$. Then the ray is the line formed by the join of these two points

$$\mathbf{X}(\lambda) = \mathtt{P}^+\mathbf{x} + \lambda \mathbf{C}. \tag{6.13}$$

In the case of finite cameras an alternative expression can be developed. Writing $\mathtt{P} = [\mathtt{M} \mid \mathbf{p}_4]$, the camera centre is given by $\widetilde{\mathbf{C}} = -\mathtt{M}^{-1}\mathbf{p}_4$. An image point $\mathbf{x}$ back-projects to a ray intersecting the plane at infinity at the point $\mathbf{D} = ((\mathtt{M}^{-1}\mathbf{x})^\mathsf{T}, 0)^\mathsf{T}$, and $\mathbf{D}$ provides a second point on the ray. Again writing the line as the join of two points on the ray,

$$\mathbf{X}(\mu) = \mu \begin{pmatrix} \mathtt{M}^{-1}\mathbf{x} \\ 0 \end{pmatrix} + \begin{pmatrix} -\mathtt{M}^{-1}\mathbf{p}_4 \\ 1 \end{pmatrix} = \begin{pmatrix} \mathtt{M}^{-1}(\mu\mathbf{x} - \mathbf{p}_4) \\ 1 \end{pmatrix}. \tag{6.14}$$

6.2.3 Depth of points

Next, we consider the distance a point lies in front of or behind the principal plane of the camera. Consider a camera matrix $\mathtt{P} = [\mathtt{M} \mid \mathbf{p}_4]$, projecting a point $\mathbf{X} = (\mathtt{X}, \mathtt{Y}, \mathtt{Z}, 1)^\mathsf{T} = (\widetilde{\mathbf{X}}^\mathsf{T}, 1)^\mathsf{T}$ in 3-space to the image point $\mathbf{x} = w(x, y, 1)^\mathsf{T} = \mathtt{P}\mathbf{X}$. Let $\mathbf{C} = (\widetilde{\mathbf{C}}, 1)^\mathsf{T}$ be the camera centre. Then $w = \mathbf{P}^{3\mathsf{T}}\mathbf{X} = \mathbf{P}^{3\mathsf{T}}(\mathbf{X} - \mathbf{C})$ since $\mathtt{P}\mathbf{C} = \mathbf{0}$ for the camera centre $\mathbf{C}$. However, $\mathbf{P}^{3\mathsf{T}}(\mathbf{X} - \mathbf{C}) = \mathbf{m}^{3\mathsf{T}}(\widetilde{\mathbf{X}} - \widetilde{\mathbf{C}})$ where $\mathbf{m}^3$ is the principal ray direction, so $w = \mathbf{m}^{3\mathsf{T}}(\widetilde{\mathbf{X}} - \widetilde{\mathbf{C}})$ can be interpreted as the dot product of the ray from the camera centre to the point $\mathbf{X}$, with the principal ray direction. If the camera matrix is normalized so that $\det \mathtt{M} > 0$ and $\|\mathbf{m}^3\| = 1$, then $\mathbf{m}^3$ is a unit vector pointing in the *positive* axial direction. Then w may be interpreted as the depth of the point $\mathbf{X}$ from the camera centre $\mathbf{C}$ in the direction of the principal ray. This is illustrated in figure 6.6.

Any camera matrix may be normalized by multiplying it by an appropriate factor. However, to avoid having always to deal with normalized camera matrices, the depth of a point may be computed as follows:

Result 6.1. *Let* $\mathbf{X} = (\mathtt{X}, \mathtt{Y}, \mathtt{Z}, \mathtt{T})^\mathsf{T}$ *be a 3D point and* $\mathtt{P} = [\mathtt{M} \mid \mathbf{p}_4]$ *be a camera matrix for a finite camera. Suppose* $\mathtt{P}(\mathtt{X}, \mathtt{Y}, \mathtt{Z}, \mathtt{T})^\mathsf{T} = w(x, y, 1)^\mathsf{T}$. *Then*

$$\text{depth}(\mathbf{X}; \mathtt{P}) = \frac{\text{sign}(\det \mathtt{M})w}{\mathtt{T}\|\mathbf{m}^3\|} \tag{6.15}$$

Samsung/Google, EX1010, p. 180

is the depth of the point $\mathbf{X}$ *in front of the principal plane of the camera.*

This formula is an effective way to determine if a point $\mathbf{X}$ is in front of the camera. One verifies that the value of $\text{depth}(\mathbf{X};\mathtt{P})$ is unchanged if either the point $\mathbf{X}$ or the camera matrix $\mathtt{P}$ is multiplied by a constant factor k. Thus, $\text{depth}(\mathbf{X};\mathtt{P})$ is independent of the particular homogeneous representation of $\mathbf{X}$ and $\mathtt{P}$.

6.2.4 Decomposition of the camera matrix

Let $\mathtt{P}$ be a camera matrix representing a general projective camera. We wish to find the camera centre, the orientation of the camera and the internal parameters of the camera from $\mathtt{P}$.

Finding the camera centre. The camera centre $\mathtt{C}$ is the point for which $\mathtt{PC} = \mathbf{0}$. Numerically this right null-vector may be obtained from the SVD of $\mathtt{P}$, see section A4.4(*p*585). Algebraically, the centre $\mathtt{C} = (\mathtt{X}, \mathtt{Y}, \mathtt{Z}, \mathtt{T})^\mathsf{T}$ may be obtained as (see (3.5–*p*67))

$$\begin{aligned} \mathtt{X} &= \det([\mathbf{p}_2, \mathbf{p}_3, \mathbf{p}_4]) & \mathtt{Y} &= -\det([\mathbf{p}_1, \mathbf{p}_3, \mathbf{p}_4]) \\ \mathtt{Z} &= \det([\mathbf{p}_1, \mathbf{p}_2, \mathbf{p}_4]) & \mathtt{T} &= -\det([\mathbf{p}_1, \mathbf{p}_2, \mathbf{p}_3]). \end{aligned}$$

Finding the camera orientation and internal parameters. In the case of a finite camera, according to (6.11),

$$\mathtt{P} = [\mathtt{M} \mid -\mathtt{M}\tilde{\mathtt{C}}] = \mathtt{K}[\mathtt{R} \mid -\mathtt{R}\tilde{\mathtt{C}}].$$

We may easily find both $\mathtt{K}$ and $\mathtt{R}$ by decomposing $\mathtt{M}$ as $\mathtt{M} = \mathtt{KR}$ using the RQ-decomposition. This decomposition into the product of an upper-triangular and orthogonal matrix is described in section A4.1.1(*p*579). The matrix $\mathtt{R}$ gives the orientation of the camera, whereas $\mathtt{K}$ is the calibration matrix. The ambiguity in the decomposition is removed by requiring that $\mathtt{K}$ have positive diagonal entries.

The matrix $\mathtt{K}$ has the form (6.10)

$$\mathtt{K} = \begin{bmatrix} \alpha_x & s & x_0 \\ 0 & \alpha_y & y_0 \\ 0 & 0 & 1 \end{bmatrix}$$

where

- α_x is the scale factor in the x-coordinate direction,
- α_y is the scale factor in the y-coordinate direction,
- s is the skew,
- $(x_0, y_0)^\mathsf{T}$ are the coordinates of the principal point.

The *aspect ratio* is α_y/α_x.

Example 6.2. The camera matrix

$$\mathtt{P} = \begin{bmatrix} 3.53553\,e{+}2 & 3.39645\,e{+}2 & 2.77744\,e{+}2 & -1.44946\,e{+}6 \\ -1.03528\,e{+}2 & 2.33212\,e{+}1 & 4.59607\,e{+}2 & -6.32525\,e{+}5 \\ 7.07107\,e{-}1 & -3.53553\,e{-}1 & 6.12372\,e{-}1 & -9.18559\,e{+}2 \end{bmatrix}$$

Samsung/Google, EX1010, p. 181

with $\mathtt{P} = [\mathtt{M} \mid -\mathtt{M}\widetilde{\mathbf{C}}]$, has centre $\widetilde{\mathbf{C}} = (1000.0, 2000.0, 1500.0)^\mathsf{T}$, and the matrix M decomposes as

$$\mathtt{M} = \mathtt{KR} = \begin{bmatrix} 468.2 & 91.2 & 300.0 \\ & 427.2 & 200.0 \\ & & 1.0 \end{bmatrix} \begin{bmatrix} 0.41380 & 0.90915 & 0.04708 \\ -0.57338 & 0.22011 & 0.78917 \\ 0.70711 & -0.35355 & 0.61237 \end{bmatrix}.$$

△

When is $s \neq 0$? As was shown in section 6.1 a true CCD camera has only four internal camera parameters, since generally $s = 0$. If $s \neq 0$ then this can be interpreted as a skewing of the pixel elements in the CCD array so that the x- and y-axes are not perpendicular. This is admittedly very unlikely to happen.

In realistic circumstances a non-zero skew might arise as a result of taking an image of an image, for example if a photograph is re-photographed, or a negative is enlarged. Consider enlarging an image taken by a pinhole camera (such as an ordinary film camera) where the axis of the magnifying lens is not perpendicular to the film plane or the enlarged image plane.

The most severe distortion that can arise from this "picture of a picture" process is a planar homography. Suppose the original (finite) camera is represented by the matrix P, then the camera representing the picture of a picture is HP, where H is the homography matrix. Since H is non-singular, the left 3×3 submatrix of HP is non-singular and can be decomposed as the product KR – and K need not have $s = 0$. Note however that the K and R are no longer the calibration matrix and orientation of the original camera.

On the other hand, one verifies that the process of taking a picture of a picture does not change the apparent camera centre. Indeed, since H is non-singular, $\mathtt{HPC} = \mathbf{0}$ if and only if $\mathtt{PC} = \mathbf{0}$.

Where is the decomposition required? If the camera P is constructed from (6.11) then the parameters are known and a decomposition is clearly unnecessary. So the question arises – where would one obtain a camera for which the decomposition is not known? In fact cameras will be computed in myriad ways throughout this book and decomposing an unknown camera is a frequently used tool in practice. For example cameras can be computed directly by *calibration* – where the camera is computed from a set of world to image correspondences (chapter 7) – and indirectly by computing a multiple view relation (such as the fundamental matrix or trifocal tensor) and subsequently computing projection matrices from this relation.

A note on coordinate orientation. In the derivation of the camera model and its parametrization (6.10) it is assumed that the coordinate systems used in both the image and the 3D world are right handed systems, as shown in figure 6.1(*p*154). However, a common practice in measuring image coordinates is that the y-coordinate increases in the downwards direction, thus defining a left handed coordinate system, contrary to figure 6.1(*p*154). A recommended practice in this case is to negate the y-coordinate of the image point so that the coordinate system again becomes right handed. However, if

Samsung/Google, EX1010, p. 182

the image coordinate system is left handed, then the consequences are not grave. The relationship between world and image coordinates is still expressed by a 3×4 camera matrix. Decomposition of this camera matrix according to (6.11) with $\mathtt{K}$ of the form (6.10) is still possible with α_x and α_y positive. The difference is that $\mathtt{R}$ now represents the orientation of the camera with respect to the negative Z-axis. In addition, the depth of points given by (6.15) will be *negative* instead of positive for points in front of the camera. If this is borne in mind then it is permissible to use left handed coordinates in the image.

6.2.5 Euclidean vs projective spaces

The development of the sections to this point has implicitly assumed that the world and image coordinate systems are Euclidean. Ideas have been borrowed from projective geometry (such as directions corresponding to points on π_∞) and the convenient notation of homogeneous coordinates has allowed central projection to be represented linearly.

In subsequent chapters of the book we will go further and use a projective coordinate frame. This is easily achieved, for suppose the world coordinate frame is projective; then the transformation between the camera and world coordinate frame (6.6) is again represented by a 4×4 homogeneous matrix, $\mathbf{X}_{\text{cam}} = \mathtt{H}\mathbf{X}$, and the resulting map from projective 3-space $\mathbb{P}^3$ to the image is still represented by a 3×4 matrix $\mathtt{P}$ with rank 3. In fact, at its most general the projective camera is a map from $\mathbb{P}^3$ to $\mathbb{P}^2$, and covers the composed effects of a projective transformation of 3-space, a projection from 3-space to an image, and a projective transformation of the image. This follows simply by concatenating the matrices representing these mappings:

$$\mathtt{P} = [3 \times 3 \text{ homography}] \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} [4 \times 4 \text{ homography}]$$

which results in a 3×4 matrix.

However, it is important to remember that cameras are Euclidean devices and simply because we have a projective model of a camera it does not mean that we should eschew notions of Euclidean geometry.

Euclidean and affine interpretations. Although a (finite) 3×4 matrix can always be decomposed as in section 6.2.4 to obtain a rotation matrix, a calibration matrix $\mathtt{K}$, and so forth, Euclidean interpretations of the parameters so obtained are only meaningful if the image and space coordinates are in an appropriate frame. In the decomposition case a Euclidean frame is required for both image and 3-space. On the other hand, the interpretation of the null-vector of $\mathtt{P}$ as the camera centre is valid even if both frames are projective – the interpretation requires only collinearity, which is a projective notion. The interpretation of $\mathbf{P}^3$ as the principal plane requires at least affine frames for the image and 3-space. Finally, the interpretation of $\mathbf{m}^3$ as the principal ray requires an affine image frame but a Euclidean world frame in order for the concept of orthogonality (to the principal plane) to be meaningful.

Samsung/Google, EX1010, p. 183

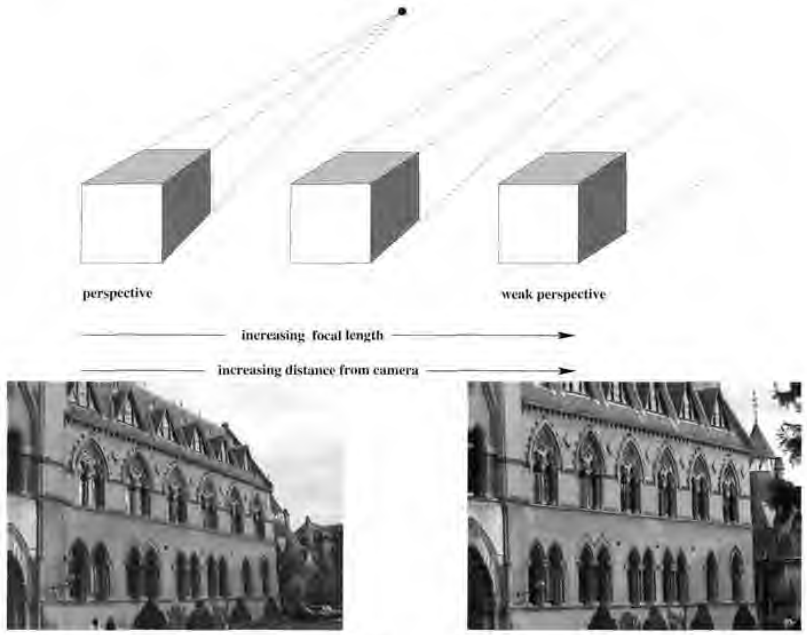


Fig. 6.7. *As the focal length increases and the distance between the camera and object also increases, the image remains the same size but perspective effects diminish.*

6.3 Cameras at infinity

We now turn to consider cameras with centre lying on the plane at infinity. This means that the left hand 3×3 block of the camera matrix $\mathtt{P}$ is singular. The camera centre may be found from $\mathtt{PC} = \mathbf{0}$ just as with finite cameras.

Cameras at infinity may be broadly classified into two different types, *affine cameras* and *non-affine cameras*. We consider first of all the affine class of cameras which are the most important in practice.

Definition 6.3. An *affine* camera is one that has a camera matrix $\mathtt{P}$ in which the last row $\mathbf{P}^{3\mathsf{T}}$ is of the form $(0, 0, 0, 1)$.

It is called an affine camera because points at infinity are mapped to points at infinity.

6.3.1 Affine cameras

Consider what happens as we apply a cinematographic technique of tracking back while zooming in, in such a way as to keep objects of interest the same size[1]. This is illustrated in figure 6.7. We are going to model this process by taking the limit as both the focal length and principal axis distance of the camera from the object increase.

In analyzing this technique, we start with a finite projective camera (6.11). The

[1] See 'Vertigo' (Dir. Hitchcock, 1958) and 'Mishima' (Dir. Schrader, 1985).

Samsung/Google, EX1010, p. 184

camera matrix may be written as

$$\mathtt{P}_0 = \mathtt{K}\mathtt{R}[\mathtt{I} \mid -\widetilde{\mathbf{C}}] = \mathtt{K}\begin{bmatrix} \mathbf{r}^{1\mathsf{T}} & -\mathbf{r}^{1\mathsf{T}}\widetilde{\mathbf{C}} \\ \mathbf{r}^{2\mathsf{T}} & -\mathbf{r}^{2\mathsf{T}}\widetilde{\mathbf{C}} \\ \mathbf{r}^{3\mathsf{T}} & -\mathbf{r}^{3\mathsf{T}}\widetilde{\mathbf{C}} \end{bmatrix} \tag{6.16}$$

where $\mathbf{r}^{i\mathsf{T}}$ is the i-th row of the rotation matrix. This camera is located at position $\widetilde{\mathbf{C}}$ and has orientation denoted by matrix $\mathtt{R}$ and internal parameters matrix $\mathtt{K}$ of the form given in (6.10–p157). From section 6.2.1 the principal ray of the camera is in the direction of the vector $\mathbf{r}^3$, and the value $d_0 = -\mathbf{r}^{3\mathsf{T}}\widetilde{\mathbf{C}}$ is the distance of the world origin from the camera centre in the direction of the principal ray.

Now, we consider what happens if the camera centre is moved backwards along the principal ray at unit speed for a time t, so that the centre of the camera is moved to $\widetilde{\mathbf{C}} - t\mathbf{r}^3$. Replacing $\widetilde{\mathbf{C}}$ in (6.16) by $\widetilde{\mathbf{C}} - t\mathbf{r}^3$ gives the camera matrix at time t:

$$\mathtt{P}_t = \mathtt{K}\begin{bmatrix} \mathbf{r}^{1\mathsf{T}} & -\mathbf{r}^{1\mathsf{T}}(\widetilde{\mathbf{C}} - t\mathbf{r}^3) \\ \mathbf{r}^{2\mathsf{T}} & -\mathbf{r}^{2\mathsf{T}}(\widetilde{\mathbf{C}} - t\mathbf{r}^3) \\ \mathbf{r}^{3\mathsf{T}} & -\mathbf{r}^{3\mathsf{T}}(\widetilde{\mathbf{C}} - t\mathbf{r}^3) \end{bmatrix} = \mathtt{K}\begin{bmatrix} \mathbf{r}^{1\mathsf{T}} & -\mathbf{r}^{1\mathsf{T}}\widetilde{\mathbf{C}} \\ \mathbf{r}^{2\mathsf{T}} & -\mathbf{r}^{2\mathsf{T}}\widetilde{\mathbf{C}} \\ \mathbf{r}^{3\mathsf{T}} & d_t \end{bmatrix} \tag{6.17}$$

where the terms $\mathbf{r}^{i\mathsf{T}}\mathbf{r}^3$ are zero for $i = 1, 2$ because $\mathtt{R}$ is a rotation matrix. The scalar $d_t = -\mathbf{r}^{3\mathsf{T}}\widetilde{\mathbf{C}} + t$ is the depth of the world origin with respect to the camera centre in the direction of the principal ray $\mathbf{r}^3$ of the camera. Thus

- *The effect of tracking along the principal ray is to replace the (3,4) entry of the matrix by the depth d_t of the camera centre from the world origin.*

Next, we consider zooming such that the camera focal length is increased by a factor k. This magnifies the image by a factor k. It is shown in section 8.4.1(p203) that the effect of zooming by a factor k is to multiply the calibration matrix $\mathtt{K}$ on the right by $\mathrm{diag}(k, k, 1)$.

Now, we combine the effects of tracking and zooming. We suppose that the magnification factor is $k = d_t/d_0$ so that the image size remains fixed. The resulting camera matrix at time t, derived from (6.17), is

$$\mathtt{P}_t = \mathtt{K}\begin{bmatrix} d_t/d_0 & & \\ & d_t/d_0 & \\ & & 1 \end{bmatrix}\begin{bmatrix} \mathbf{r}^{1\mathsf{T}} & -\mathbf{r}^{1\mathsf{T}}\widetilde{\mathbf{C}} \\ \mathbf{r}^{2\mathsf{T}} & -\mathbf{r}^{2\mathsf{T}}\widetilde{\mathbf{C}} \\ \mathbf{r}^{3\mathsf{T}} & d_t \end{bmatrix} = \frac{d_t}{d_0}\mathtt{K}\begin{bmatrix} \mathbf{r}^{1\mathsf{T}} & -\mathbf{r}^{1\mathsf{T}}\widetilde{\mathbf{C}} \\ \mathbf{r}^{2\mathsf{T}} & -\mathbf{r}^{2\mathsf{T}}\widetilde{\mathbf{C}} \\ \mathbf{r}^{3\mathsf{T}}d_0/d_t & d_0 \end{bmatrix}$$

and one can ignore the factor d_t/d_0. When $t = 0$, the camera matrix $\mathtt{P}_t$ corresponds with (6.16). Now, in the limit as d_t tends to infinity, this matrix becomes

$$\mathtt{P}_\infty = \lim_{t\to\infty} \mathtt{P}_t = \mathtt{K}\begin{bmatrix} \mathbf{r}^{1\mathsf{T}} & -\mathbf{r}^{1\mathsf{T}}\widetilde{\mathbf{C}} \\ \mathbf{r}^{2\mathsf{T}} & -\mathbf{r}^{2\mathsf{T}}\widetilde{\mathbf{C}} \\ \mathbf{0}^{\mathsf{T}} & d_0 \end{bmatrix} \tag{6.18}$$

which is just the original camera matrix (6.16) with the first three entries of the last row set to zero. From definition 6.3 $\mathtt{P}_\infty$ is an instance of an affine camera.

Samsung/Google, EX1010, p. 185

6.3.2 Error in employing an affine camera

It may be noted that the image of any point on the plane through the world origin perpendicular to the principal axis direction $\mathbf{r}^3$ is unchanged by this combined zooming and motion. Indeed, such a point may be written as

$$\mathbf{X} = \begin{pmatrix} \alpha\mathbf{r}^1 + \beta\mathbf{r}^2 \\ 1 \end{pmatrix}.$$

One then verifies that $\mathtt{P}_0\mathbf{X} = \mathtt{P}_t\mathbf{X} = \mathtt{P}_\infty\mathbf{X}$ for all t, since $\mathbf{r}^{3\mathsf{T}}(\alpha\mathbf{r}^1 + \beta\mathbf{r}^2) = 0$.

For points not on this plane the images under $\mathtt{P}_0$ and $\mathtt{P}_\infty$ differ, and we will now investigate the extent of this error. Consider a point $\mathbf{X}$ which is at a perpendicular distance Δ from this plane. The 3D point can be represented as

$$\mathbf{X} = \begin{pmatrix} \alpha\mathbf{r}^1 + \beta\mathbf{r}^2 + \Delta\mathbf{r}^3 \\ 1 \end{pmatrix}$$

and is imaged by the cameras $\mathtt{P}_0$ and $\mathtt{P}_\infty$ at

$$\mathbf{x}_{\text{proj}} = \mathtt{P}_0\mathbf{X} = \mathtt{K}\begin{pmatrix} \tilde{x} \\ \tilde{y} \\ d_0 + \Delta \end{pmatrix} \qquad \mathbf{x}_{\text{affine}} = \mathtt{P}_\infty\mathbf{X} = \mathtt{K}\begin{pmatrix} \tilde{x} \\ \tilde{y} \\ d_0 \end{pmatrix}$$

where $\tilde{x} = \alpha - \mathbf{r}^{1\mathsf{T}}\widetilde{\mathbf{C}}$, $\tilde{y} = \beta - \mathbf{r}^{2\mathsf{T}}\widetilde{\mathbf{C}}$. Now, writing the calibration matrix as

$$\mathtt{K} = \begin{bmatrix} \mathtt{K}_{2\times 2} & \tilde{\mathbf{x}}_0 \\ \tilde{\mathbf{0}}^{\mathsf{T}} & 1 \end{bmatrix},$$

where $\mathtt{K}_{2\times 2}$ is an upper-triangular 2×2 matrix, gives

$$\mathbf{x}_{\text{proj}} = \begin{pmatrix} \mathtt{K}_{2\times 2}\tilde{\mathbf{x}} + (d_0 + \Delta)\tilde{\mathbf{x}}_0 \\ d_0 + \Delta \end{pmatrix} \qquad \mathbf{x}_{\text{affine}} = \begin{pmatrix} \mathtt{K}_{2\times 2}\tilde{\mathbf{x}} + d_0\tilde{\mathbf{x}}_0 \\ d_0 \end{pmatrix}$$

The image point for $\mathtt{P}_0$ is obtained by dehomogenizing, by dividing by the third element, as $\tilde{\mathbf{x}}_{\text{proj}} = \tilde{\mathbf{x}}_0 + \mathtt{K}_{2\times 2}\tilde{\mathbf{x}}/(d_0 + \Delta)$, and for $\mathtt{P}_\infty$ the inhomogeneous image point is $\tilde{\mathbf{x}}_{\text{affine}} = \tilde{\mathbf{x}}_0 + \mathtt{K}_{2\times 2}\tilde{\mathbf{x}}/d_0$. The relationship between the two points is therefore

$$\tilde{\mathbf{x}}_{\text{affine}} - \tilde{\mathbf{x}}_0 = \frac{d_0 + \Delta}{d_0}(\tilde{\mathbf{x}}_{\text{proj}} - \tilde{\mathbf{x}}_0)$$

which shows that

- *The effect of the affine approximation $\mathtt{P}_\infty$ to the true camera matrix $\mathtt{P}_0$ is to move the image of a point $\mathbf{X}$ radially towards or away from the principal point $\tilde{\mathbf{x}}_0$ by a factor equal to $(d_0 + \Delta)/d_0 = 1 + \Delta/d_0$.*

This is illustrated in figure 6.8.

Samsung/Google, EX1010, p. 186

Affine imaging conditions. From the expressions for $\tilde{\mathbf{x}}_{\text{proj}}$ and $\tilde{\mathbf{x}}_{\text{affine}}$ we can deduce that

$$\tilde{\mathbf{x}}_{\text{affine}} - \tilde{\mathbf{x}}_{\text{proj}} = \frac{\Delta}{d_0}(\tilde{\mathbf{x}}_{\text{proj}} - \tilde{\mathbf{x}}_0) \tag{6.19}$$

which shows that the distance between the true perspective image position and the position obtained using the affine camera approximation $\mathtt{P}_\infty$ will be small provided:

(i) The depth relief (Δ) is small compared to the average depth (d_0), and
(ii) The distance of the point from the principal ray is small.

The latter condition is satisfied by a small field of view. In general, images acquired using a lens with a longer focal length tend to satisfy these conditions as both the field of view and the depth of field are smaller than those obtained by a short focal length lens with the same CCD array.

For scenes at which there are many points at different depths, the affine camera is not a good approximation. For instance where the scene contains close foreground as well as background objects, an affine camera model should not be used. However, a different affine model can be used for each region in these circumstances.

6.3.3 Decomposition of $\mathtt{P}_\infty$

The camera matrix (6.18) may be written as

$$\mathtt{P}_\infty = \begin{bmatrix} \mathtt{K}_{2\times 2} & \tilde{\mathbf{x}}_0 \\ \hat{\mathbf{0}}^\top & 1 \end{bmatrix} \begin{bmatrix} \hat{\mathtt{R}} & \hat{\mathbf{t}} \\ \mathbf{0}^\top & d_0 \end{bmatrix}$$

where $\hat{\mathtt{R}}$ consists of the two first rows of a rotation matrix, $\hat{\mathbf{t}}$ is the vector $(-\mathbf{r}^{1\top}\tilde{\mathbf{C}}, -\mathbf{r}^{2\top}\tilde{\mathbf{C}})^\top$, and $\hat{\mathbf{0}}$ the vector $(0,0)^\top$. The 2×2 matrix $\mathtt{K}_{2\times 2}$ is upper-triangular. One quickly verifies that

$$\mathtt{P}_\infty = \begin{bmatrix} \mathtt{K}_{2\times 2} & \tilde{\mathbf{x}}_0 \\ \hat{\mathbf{0}}^\top & 1 \end{bmatrix} \begin{bmatrix} \hat{\mathtt{R}} & \hat{\mathbf{t}} \\ \mathbf{0}^\top & d_0 \end{bmatrix} = \begin{bmatrix} d_0^{-1}\mathtt{K}_{2\times 2} & \tilde{\mathbf{x}}_0 \\ \hat{\mathbf{0}}^\top & 1 \end{bmatrix} \begin{bmatrix} \hat{\mathtt{R}} & \hat{\mathbf{t}} \\ \mathbf{0}^\top & 1 \end{bmatrix}$$

so we may replace $\mathtt{K}_{2\times 2}$ by $d_0^{-1}\mathtt{K}_{2\times 2}$ and assume that $d_0 = 1$. Multiplying out this product gives

$$\begin{aligned} \mathtt{P}_\infty &= \begin{bmatrix} \mathtt{K}_{2\times 2}\hat{\mathtt{R}} & \mathtt{K}_{2\times 2}\hat{\mathbf{t}} + \tilde{\mathbf{x}}_0 \\ \hat{\mathbf{0}}^\top & 1 \end{bmatrix} = \begin{bmatrix} \mathtt{K}_{2\times 2} & \hat{\mathbf{0}} \\ \hat{\mathbf{0}}^\top & 1 \end{bmatrix} \begin{bmatrix} \hat{\mathtt{R}} & \hat{\mathbf{t}} + \mathtt{K}_{2\times 2}^{-1}\tilde{\mathbf{x}}_0 \\ \mathbf{0}^\top & 1 \end{bmatrix} \\ &= \begin{bmatrix} \mathtt{K}_{2\times 2} & \mathtt{K}_{2\times 2}\hat{\mathbf{t}} + \tilde{\mathbf{x}}_0 \\ \hat{\mathbf{0}}^\top & 1 \end{bmatrix} \begin{bmatrix} \hat{\mathtt{R}} & \hat{\mathbf{0}} \\ \mathbf{0}^\top & 1 \end{bmatrix}. \end{aligned}$$

Thus, making appropriate substitutions for $\hat{\mathbf{t}}$ or $\tilde{\mathbf{x}}_0$, we can write the affine camera matrix in one of the two forms

$$\mathtt{P}_\infty = \begin{bmatrix} \mathtt{K}_{2\times 2} & \hat{\mathbf{0}} \\ \hat{\mathbf{0}}^\top & 1 \end{bmatrix} \begin{bmatrix} \hat{\mathtt{R}} & \hat{\mathbf{t}} \\ \mathbf{0}^\top & 1 \end{bmatrix} = \begin{bmatrix} \mathtt{K}_{2\times 2} & \tilde{\mathbf{x}}_0 \\ \hat{\mathbf{0}}^\top & 1 \end{bmatrix} \begin{bmatrix} \hat{\mathtt{R}} & \hat{\mathbf{0}} \\ \mathbf{0}^\top & 1 \end{bmatrix}. \tag{6.20}$$

Consequently, the camera $\mathtt{P}_\infty$ can be interpreted in terms of these decompositions in

Samsung/Google, EX1010, p. 187

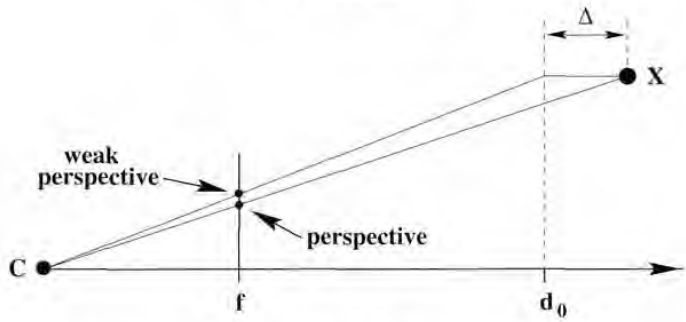


Fig. 6.8. **Perspective vs weak perspective projection.** *The action of the weak perspective camera is equivalent to orthographic projection onto a plane (at $Z = d_0$), followed by perspective projection from the plane. The difference between the perspective and weak perspective image point depends both on the distance Δ of the point $\mathbf{X}$ from the plane, and the distance of the point from the principal ray.*

one of two ways, either with $\tilde{\mathbf{x}}_0 = \mathbf{0}$ or with $\tilde{\mathbf{t}} = \hat{\mathbf{0}}$. Using the second decomposition of (6.20), the image of the world origin is $\mathtt{P}_\infty(0,0,0,1)^\top = (\tilde{\mathbf{x}}_0^\top, 1)^\top$. Consequently, the value of $\tilde{\mathbf{x}}_0$ is dependent on the particular choice of world coordinates, and hence is not an intrinsic property of the camera itself. This means that the camera matrix $\mathtt{P}_\infty$ does not have a principal point. Therefore, it is preferable to use the first decomposition of $\mathtt{P}_\infty$ in (6.20), and write

$$\mathtt{P}_\infty = \begin{bmatrix} \mathtt{K}_{2\times 2} & \hat{\mathbf{0}} \\ \hat{\mathbf{0}}^\top & 1 \end{bmatrix} \begin{bmatrix} \tilde{\mathtt{R}} & \tilde{\mathbf{t}} \\ \mathbf{0}^\top & 1 \end{bmatrix} \tag{6.21}$$

where the two matrices represent the internal camera parameters and external camera parameters of $\mathtt{P}_\infty$.

Parallel projection. In summary the essential differences between $\mathtt{P}_\infty$ and a finite camera are:

- *The parallel projection matrix* $\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$ *replaces the canonical projection matrix* $[\mathtt{I} \mid \mathbf{0}]$ *of a finite camera (6.5–p155).*
- *The calibration matrix* $\begin{bmatrix} \mathtt{K}_{2\times 2} & \hat{\mathbf{0}} \\ \hat{\mathbf{0}}^\top & 1 \end{bmatrix}$ *replaces* $\mathtt{K}$ *of a finite camera (6.10–p157).*
- *The principal point is not defined.*

6.3.4 A hierarchy of affine cameras

In a similar manner to the development of the finite projection camera taxonomy in section 6.1 we can start with the basic operation of parallel projection and build a hierarchy of camera models representing progressively more general cases of parallel projection.

Samsung/Google, EX1010, p. 188

Orthographic projection. Consider projection along the Z-axis. This is represented by a matrix of the form

$$\mathtt{P} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}. \tag{6.22}$$

This mapping takes a point $(\mathtt{X}, \mathtt{Y}, \mathtt{Z}, 1)^\mathsf{T}$ to the image point $(\mathtt{X}, \mathtt{Y}, 1)^\mathsf{T}$, dropping the Z-coordinate.

For a general orthographic projection mapping, we precede this map by a 3D Euclidean coordinate change of the form

$$\mathtt{H} = \begin{bmatrix} \mathtt{R} & \mathbf{t} \\ \mathbf{0}^\mathsf{T} & 1 \end{bmatrix}.$$

Writing $\mathbf{t} = (t_1, t_2, t_3)^\mathsf{T}$, we see that a general orthographic camera is of the form

$$\mathtt{P} = \begin{bmatrix} \mathbf{r}^{1\mathsf{T}} & t_1 \\ \mathbf{r}^{2\mathsf{T}} & t_2 \\ \mathbf{0}^\mathsf{T} & 1 \end{bmatrix}. \tag{6.23}$$

An orthographic camera has five degrees of freedom, namely the three parameters describing the rotation matrix R, plus the two offset parameters t_1 and t_2. An orthographic projection matrix $\mathtt{P} = [\mathtt{M} \mid \mathbf{t}]$ is characterized by a matrix M with last row zero, with the first two rows orthogonal and of unit norm, and $t_3 = 1$.

Scaled orthographic projection. A scaled orthographic projection is an orthographic projection followed by isotropic scaling. Thus, in general, its matrix may be written in the form

$$\mathtt{P} = \begin{bmatrix} k & & \\ & k & \\ & & 1 \end{bmatrix} \begin{bmatrix} \mathbf{r}^{1\mathsf{T}} & t_1 \\ \mathbf{r}^{2\mathsf{T}} & t_2 \\ \mathbf{0}^\mathsf{T} & 1 \end{bmatrix} = \begin{bmatrix} \mathbf{r}^{1\mathsf{T}} & t_1 \\ \mathbf{r}^{2\mathsf{T}} & t_2 \\ \mathbf{0}^\mathsf{T} & 1/k \end{bmatrix}. \tag{6.24}$$

It has six degrees of freedom. A scaled orthographic projection matrix $\mathtt{P} = [\mathtt{M} \mid \mathbf{t}]$ is characterized by a matrix M with last row zero, and the first two rows orthogonal and of equal norm.

Weak perspective projection. Analogous to a finite CCD camera, we may consider the case of a camera at infinity for which the scale factors in the two axial image directions are not equal. Such a camera has a projection matrix of the form

$$\mathtt{P} = \begin{bmatrix} \alpha_x & & \\ & \alpha_y & \\ & & 1 \end{bmatrix} \begin{bmatrix} \mathbf{r}^{1\mathsf{T}} & t_1 \\ \mathbf{r}^{2\mathsf{T}} & t_2 \\ \mathbf{0}^\mathsf{T} & 1 \end{bmatrix}. \tag{6.25}$$

It has seven degrees of freedom. A weak perspective projection matrix $\mathtt{P} = [\mathtt{M} \mid \mathbf{t}]$ is characterized by a matrix M with last row zero, and first two rows orthogonal (but they need not have equal norm as is required in the scaled orthographic case). The geometric action of this camera is illustrated in figure 6.8.

Samsung/Google, EX1010, p. 189

The affine camera, $\mathtt{P}_A$. As has already been seen in the case of $\mathtt{P}_\infty$, a general camera matrix of the affine form, and with no restrictions on its elements, may be decomposed as

$$\mathtt{P}_A = \begin{bmatrix} \alpha_x & s & \\ & \alpha_y & \\ & & 1 \end{bmatrix} \begin{bmatrix} \mathbf{r}^{1\mathsf{T}} & t_1 \\ \mathbf{r}^{2\mathsf{T}} & t_2 \\ \mathbf{0}^{\mathsf{T}} & 1 \end{bmatrix}.$$

It has eight degrees of freedom, and may be thought of as the parallel projection version of the finite projective camera (6.11–*p*157).

In full generality an affine camera has the form

$$\mathtt{P}_A = \begin{bmatrix} m_{11} & m_{12} & m_{13} & t_1 \\ m_{21} & m_{22} & m_{23} & t_2 \\ 0 & 0 & 0 & 1 \end{bmatrix}.$$

It has eight degrees of freedom corresponding to the eight non-zero and non-unit matrix elements. We denote the top left 2×3 submatrix by $\mathtt{M}_{2\times3}$. The sole restriction on the affine camera is that $\mathtt{M}_{2\times3}$ has rank 2. This arises from the requirement that the rank of $\mathtt{P}$ is 3.

The affine camera covers the composed effects of an affine transformation of 3-space, an orthographic projection from 3-space to an image, and an affine transformation of the image. This follows simply by concatenating the matrices representing these mappings:

$$\mathtt{P}_A = [3 \times 3 \text{ affine}] \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} [4 \times 4 \text{ affine}]$$

which results in a 3×4 matrix of the affine form.

Projection under an affine camera is a linear mapping on *inhomogeneous* coordinates composed with a translation:

$$\begin{pmatrix} x \\ y \end{pmatrix} = \begin{bmatrix} m_{11} & m_{12} & m_{13} \\ m_{21} & m_{22} & m_{23} \end{bmatrix} \begin{pmatrix} \mathrm{X} \\ \mathrm{Y} \\ \mathrm{Z} \end{pmatrix} + \begin{pmatrix} t_1 \\ t_2 \end{pmatrix}$$

which is written more concisely as

$$\tilde{\mathbf{x}} = \mathtt{M}_{2\times3}\tilde{\mathbf{X}} + \tilde{\mathbf{t}} \ . \tag{6.26}$$

The point $\tilde{\mathbf{t}} = (t_1, t_2)^{\mathsf{T}}$ is the image of the world origin.

The camera models of this section are seen to be affine cameras satisfying additional constraints, thus the affine camera is an abstraction of this hierarchy. For example, in the case of the weak perspective camera the rows of $\mathtt{M}_{2\times3}$ are scalings of rows of a rotation matrix, and thus are orthogonal.

6.3.5 More properties of the affine camera

The plane at infinity in space is mapped to points at infinity in the image. This is easily seen by computing $\mathtt{P}_A(\mathrm{X}, \mathrm{Y}, \mathrm{Z}, 0)^{\mathsf{T}} = (\mathrm{X}, \mathrm{Y}, 0)^{\mathsf{T}}$. Extending the terminology of finite

Samsung/Google, EX1010, p. 190

projective cameras, we interpret this by saying that the principal plane of the camera is the plane at infinity. The optical centre, since it lies on the principal plane, must also lie on the plane at infinity. From this we have

(i) Conversely, any projective camera matrix for which the principal plane is the plane at infinity is an affine camera matrix.

(ii) Parallel world lines are projected to parallel image lines. This follows because parallel world lines intersect at the plane at infinity, and this intersection point is mapped to a point at infinity in the image. Hence the image lines are parallel.

(iii) The vector $\mathbf{d}$ satisfying $\mathtt{M}_{2\times 3}\mathbf{d} = \mathbf{0}$ is the *direction* of parallel projection, and $(\mathbf{d}^\top, 0)^\top$ the camera centre since $\mathtt{P}_A \begin{pmatrix} \mathbf{d} \\ 0 \end{pmatrix} = \mathbf{0}$.

Any camera which consists of the composed effects of affine transformations (either of space, or of the image) with parallel projection will have the affine form. For example, *para-perspective* projection consists of two such mappings: the first is parallel projection onto a plane π through the centroid and parallel to the image plane. The direction of parallel projection is the ray joining the centroid to the camera centre. This parallel projection is followed by an affine transformation (actually a similarity) between π and the image. Thus a para-perspective camera is an affine camera.

6.3.6 General cameras at infinity

An affine camera is one for which the principal plane is the plane at infinity. As such, its camera centre lies on the plane at infinity. However, it is possible for the camera centre to lie on the plane at infinity without the whole principal plane being the plane at infinity.

A camera centre lies at infinity if $\mathtt{P} = [\mathtt{M} \mid \mathbf{p}_4]$ with $\mathtt{M}$ a singular matrix. This is clearly a weaker condition than insisting that the last row of $\mathtt{M}$ is zero, as is the case for affine cameras. If $\mathtt{M}$ is singular, but the last row of $\mathtt{M}$ is not zero, then the camera is not affine, but not a finite projective camera either. Such a camera is rather a strange object, however, and will not be treated in detail in this book. We may compare the properties of affine and non-affine infinite cameras:

	Affine camera	Non-affine camera
Camera centre on π_∞	yes	yes
Principal plane is π_∞	yes	no
Image of points on π_∞ on $\mathbf{l}_\infty$	yes	no in general

In both cases the camera centre is the direction of projection. Furthermore, in the case of an affine camera all non-infinite points are in front of the camera. For a non-affine camera space is partitioned into two sets of points by the principal plane.

A general camera at infinity could arise from a perspective image of an image produced by an affine camera. This imaging process corresponds to left-multiplying the

Samsung/Google, EX1010, p. 191

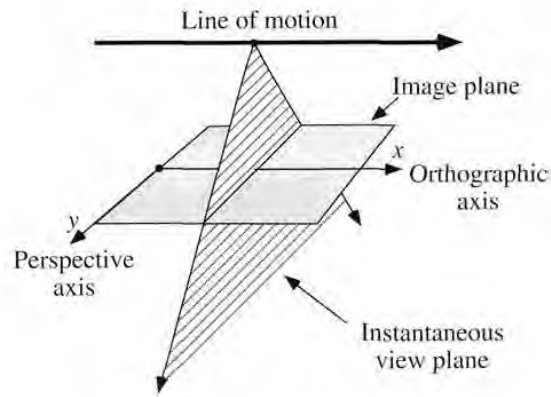


Fig. 6.9. *Acquisition geometry of a pushbroom camera.*

affine camera by a general 3×3 matrix representing the planar homography. The resulting 3×4 matrix is still a camera at infinity, but it does not have the affine form, since parallel lines in the world will in general appear as converging lines in the image.

6.4 Other camera models

6.4.1 Pushbroom cameras

The Linear Pushbroom (LP) camera is an abstraction of a type of sensor common in satellites, for instance the SPOT sensor. In such a camera, a linear sensor array is used to capture a single line of imagery at a time. As the sensor moves the sensor plane sweeps out a region of space (hence the name pushbroom), capturing the image a single line at a time. The second dimension of the image is provided by the motion of the sensor. In the linear pushbroom model, the sensor is assumed to move in a straight line at constant velocity with respect to the ground. In addition, one assumes that the orientation of the sensor array with respect to the direction of travel is constant. In the direction of the sensor, the image is effectively a perspective image, whereas in the direction of the sensor motion it is an orthographic projection. The geometry of the LP camera is illustrated in figure 6.9. It turns out that the mapping from object space into the image may be described by a 3×4 camera matrix, just as with a general projective camera. However, the way in which this matrix is used is somewhat different.

- *Let* $\mathbf{X} = (X, Y, Z, 1)^\top$ *be an object point, and let* $\mathtt{P}$ *be the camera matrix of the LP camera. Suppose that* $\mathtt{P}\mathbf{X} = (x, y, w)^\top$. *Then the corresponding image point (represented as an inhomogeneous 2-vector) is* $(x, y/w)^\top$.

One must compare this with the projective camera mapping. In that case the point represented by $(x, y, w)^\top$ is $(x/w, y/w)^\top$. Note the difference that in the LP case, the coordinate x is not divided by the factor w to get the image coordinate. In this formula, the x-axis in the image is the direction of the sensor motion, whereas the y-axis is in the direction of the linear sensor array. The camera has 11 degrees of freedom.

Samsung/Google, EX1010, p. 192

Another way of writing the formula for LP projection is

$$\tilde{x} = x = \mathtt{P}^{1\mathsf{T}}\mathbf{X} \quad \tilde{y} = y/z = \frac{\mathtt{P}^{2\mathsf{T}}\mathbf{X}}{\mathtt{P}^{3\mathsf{T}}\mathbf{X}} \tag{6.27}$$

where $(\tilde{x}, \tilde{y})$ is the image point.

Note that the $\tilde{y}$-coordinate behaves projectively, whereas the $\tilde{x}$ is obtained by orthogonal projection of the point $\mathbf{X}$ on the direction perpendicular to the plane $\mathtt{P}^1$. The vector $\mathtt{P}^1$ represents the sweep plane of the camera at time $t = 0$ – that is the moment when the line with coordinates $\tilde{x} = 0$ is captured.

Mapping of lines. One of the novel features of the LP camera is that straight lines in space are not mapped to straight lines in the image (they are mapped to straight lines in the case of a projective camera – see section 8.1.2). The set of points $\mathbf{X}$ lying on a 3D line may be written as $\mathbf{X}_0 + \alpha\mathbf{D}$, where $\mathbf{X}_0 = (\mathrm{X}, \mathrm{Y}, \mathrm{Z}, 1)^\mathsf{T}$ is a point on the line and $\mathbf{D} = (D_\mathrm{X}, D_\mathrm{Y}, D_\mathrm{Z}, 0)^\mathsf{T}$ is the intersection of this line with the plane at infinity. In this case, we compute from (6.27)

$$\begin{aligned} \tilde{x} &= \mathbf{P}^{1\mathsf{T}}(\mathbf{X}_0 + t\mathbf{D}) \\ \tilde{y} &= \frac{\mathbf{P}^{2\mathsf{T}}(\mathbf{X}_0 + t\mathbf{D})}{\mathbf{P}^{3\mathsf{T}}(\mathbf{X}_0 + t\mathbf{D})}. \end{aligned}$$

This may be written as a pair of equations $\tilde{x} = a+bt$ and $(c+dt)\tilde{y} = e+ft$. Eliminating t from these equations leads to an equation of the form $\alpha\tilde{x}\tilde{y} + \beta\tilde{x} + \gamma\tilde{y} + \delta = 0$, which is the equation of a hyperbola in the image plane, asymptotic in one direction to the line $\alpha\tilde{x} + \gamma = 0$, and in the other direction to the line $\alpha\tilde{y} + \beta = 0$. A hyperbola is made up of two curves. However, only one of the curves making up the image of a line actually appears in the image – the other part of the hyperbola corresponds to points lying behind the camera.

6.4.2 Line cameras

This chapter has dealt with the central projection of 3-space onto a 2D image. An analogous development can be given for the central projection of a plane onto a 1D line contained in the plane. See figure 22.1(*p*535). The camera model for this geometry is

$$\begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} p_{11} & p_{12} & p_{13} \\ p_{21} & p_{22} & p_{23} \end{bmatrix} \begin{bmatrix} \mathrm{X} \\ \mathrm{Y} \\ \mathrm{Z} \end{bmatrix} = \mathtt{P}_{2\times3}\mathbf{X}$$

which is a linear mapping from homogeneous representation of the plane to a homogeneous representation of the line. The camera has 5 degrees of freedom. Again the null-space, $\mathbf{c}$, of the $\mathtt{P}_{2\times3}$ projection matrix is the camera centre, and the matrix can be decomposed in a similar manner to the finite projective camera (6.11–*p*157) as

$$\mathtt{P}_{2\times3} = \mathtt{K}_{2\times2}\mathtt{R}_{2\times2}[\mathtt{I}_{2\times2} \mid -\tilde{\mathbf{c}}]$$

Samsung/Google, EX1010, p. 193

where $\tilde{\mathbf{c}}$ is the inhomogeneous 2-vector representing the centre (2 dof), $\mathtt{R}_{2\times2}$ is a rotation matrix (1 dof), and

$$\mathtt{K}_{2\times2} = \begin{bmatrix} \alpha_x & x_0 \\ & 1 \end{bmatrix}$$

the internal calibration matrix (2 dof).

6.5 Closure

This chapter has covered camera models, their taxonomy and anatomy. The subsequent chapters cover the estimation of cameras from a set of world to image correspondences, and the action of a camera on various geometric objects such as lines and quadrics. Vanishing points and vanishing lines are also described in more detail in chapter 8.

6.5.1 The literature

[Aloimonos-90] defined a hierarchy of camera models including para-perspective. Mundy and Zisserman [Mundy-92] generalized this with the affine camera. Faugeras developed properties of the projective camera in his textbook [Faugeras-93]. Further details on the linear pushbroom camera are given in [Gupta-97], and on the 2D camera in [Quan-97b].

6.5.2 Notes and exercises

(i) Let I_0 be a projective image, and I_1 be an image of I_0 (an image of an image). Let the composite image be denoted by I'. Show that the apparent camera centre of I' is the same as that of I_0. Speculate on how this explains why a portrait's eyes "follow you round the room." Verify on the other hand that all other parameters of I' and I_0 may be different.

(ii) Show that the ray back-projected from an image point $\mathbf{x}$ under a projective camera $\mathtt{P}$ (as in (6.14–p162)) may be written as

$$\mathtt{L}^* = \mathtt{P}^\top [\mathbf{x}]_\times \mathtt{P} \tag{6.28}$$

where $\mathtt{L}^*$ is the dual Plücker representation of a line (3.9–p71).

(iii) **The affine camera.**

(a) Show that the affine camera is the most general linear mapping on homogeneous coordinates that maps parallel world lines to parallel image lines. To do this consider the projection of points on π_∞, and show that only if $\mathtt{P}$ has the affine form will they map to points at infinity in the image.

(b) Show that for parallel lines mapped by an affine camera the ratio of lengths on line segments is an invariant. What other invariants are there under an affine camera?

(iv) **The rational polynomial camera** is a general camera model, used extensively

Samsung/Google, EX1010, p. 194

in the satellite surveillance community. Image coordinates are defined by the ratios

$$x = N_x(\mathbf{X})/D_x(\mathbf{X}) \quad y = N_y(\mathbf{X})/D_y(\mathbf{X})$$

where the functions N_x, D_x, N_y, D_y are homogeneous *cubic* polynomials in the 3-space point $\mathbf{X}$. Each cubic has 20 coefficients, so that overall the camera has 78 degrees of freedom. All of the cameras surveyed in this chapter (projective, affine, pushbroom) are special cases of the rational polynomial camera. Its disadvantage is that it is severely over-parametrized for these cases. More details are given in Hartley and Saxena [Hartley-97e].

(v) A finite projective camera (6.11–$p157$) $\mathtt{P}$ may be transformed to an orthographic camera (6.22) by applying a 4×4 homography $\mathtt{H}$ on the right such that

$$\mathtt{PH} = \mathtt{KR}[\mathtt{I} \mid -\widetilde{\mathbf{C}}]\mathtt{H} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} = \mathtt{P}_{\mathrm{orthog}} \ .$$

(the last row of $\mathtt{H}$ is chosen so that $\mathtt{H}$ has rank 4). Then since

$$\mathbf{x} = \mathtt{P}(\mathtt{H}\mathtt{H}^{-1})\mathbf{X} = (\mathtt{P}\mathtt{H})(\mathtt{H}^{-1}\mathbf{X}) = \mathtt{P}_{\mathrm{orthog}}\mathbf{X}'$$

imaging under $\mathtt{P}$ is equivalent to first transforming the 3-space points $\mathbf{X}$ to $\mathbf{X}' = \mathtt{H}^{-1}\mathbf{X}$ and then applying an orthographic projection. Thus the action of any camera may be considered as a projective transformation of 3-space followed by orthographic projection.

Samsung/Google, EX1010, p. 195

7

Computation of the Camera Matrix P

This chapter describes numerical methods for estimating the camera projection matrix from corresponding 3-space and image entities. This computation of the camera matrix is known as *resectioning*. The simplest such correspondence is that between a 3D point $\mathbf{X}$ and its image $\mathbf{x}$ under the unknown camera mapping. Given sufficiently many correspondences $\mathbf{X}_i \leftrightarrow \mathbf{x}_i$ the camera matrix $\mathtt{P}$ may be determined. Similarly, $\mathtt{P}$ may be determined from sufficiently many corresponding world and image lines.

If additional constraints apply to the matrix $\mathtt{P}$, such as that the pixels are square, then a *restricted* camera matrix subject to these constraints may be estimated from world to image correspondences.

Throughout this book it is assumed that the map from 3-space to the image is linear. This assumption is invalid if there is lens distortion. The topic of radial lens distortion correction is dealt with in this chapter.

The internal parameters $\mathtt{K}$ of the camera may be extracted from the matrix $\mathtt{P}$ by the decomposition of section 6.2.4. Alternatively, the internal parameters can be computed directly, without necessitating estimating $\mathtt{P}$, by the methods of chapter 8.

7.1 Basic equations

We assume a number of point correspondences $\mathbf{X}_i \leftrightarrow \mathbf{x}_i$ between 3D points $\mathbf{X}_i$ and 2D image points $\mathbf{x}_i$ are given. We are required to find a camera matrix $\mathtt{P}$, namely a 3×4 matrix such that $\mathbf{x}_i = \mathtt{P}\mathbf{X}_i$ for all i. The similarity of this problem with that of computing a 2D projective transformation $\mathtt{H}$, treated in chapter 4, is evident. The only difference is the dimension of the problem. In the 2D case the matrix $\mathtt{H}$ has dimension 3×3, whereas in the present case, $\mathtt{P}$ is a 3×4 matrix. As one may expect, much of the material from chapter 4 applies almost unchanged to the present case.

As in section 4.1(*p*88) for each correspondence $\mathbf{X}_i \leftrightarrow \mathbf{x}_i$ we derive a relationship

$$\begin{bmatrix} \mathbf{0}^\mathsf{T} & -w_i\mathbf{X}_i^\mathsf{T} & y_i\mathbf{X}_i^\mathsf{T} \\ w_i\mathbf{X}_i^\mathsf{T} & \mathbf{0}^\mathsf{T} & -x_i\mathbf{X}_i^\mathsf{T} \\ -y_i\mathbf{X}_i^\mathsf{T} & x_i\mathbf{X}_i^\mathsf{T} & \mathbf{0}^\mathsf{T} \end{bmatrix} \begin{pmatrix} \mathbf{P}^1 \\ \mathbf{P}^2 \\ \mathbf{P}^3 \end{pmatrix} = \mathbf{0}. \tag{7.1}$$

where each $\mathbf{P}^{i\mathsf{T}}$ is a 4-vector, the i-th row of $\mathtt{P}$. Alternatively, one may choose to use

Samsung/Google, EX1010, p. 196

only the first two equations:

$$\begin{bmatrix} \mathbf{0}^\top & -w_i\mathbf{X}_i^\top & y_i\mathbf{X}_i^\top \\ w_i\mathbf{X}_i^\top & \mathbf{0}^\top & -x_i\mathbf{X}_i^\top \end{bmatrix} \begin{pmatrix} \mathbf{P}^1 \\ \mathbf{P}^2 \\ \mathbf{P}^3 \end{pmatrix} = \mathbf{0} \tag{7.2}$$

since the three equations of (7.1) are linearly dependent. From a set of n point correspondences, we obtain a $2n \times 12$ matrix $\mathtt{A}$ by stacking up the equations (7.2) for each correspondence. The projection matrix $\mathtt{P}$ is computed by solving the set of equations $\mathtt{A}\mathbf{p} = \mathbf{0}$, where $\mathbf{p}$ is the vector containing the entries of the matrix $\mathtt{P}$.

Minimal solution. Since the matrix $\mathtt{P}$ has 12 entries, and (ignoring scale) 11 degrees of freedom, it is necessary to have 11 equations to solve for $\mathtt{P}$. Since each point correspondence leads to two equations, at a minimum $5\frac{1}{2}$ such correspondences are required to solve for $\mathtt{P}$. The $\frac{1}{2}$ indicates that only one of the equations is used from the sixth point, so one needs only to know the x-coordinate (or alternatively the y-coordinate) of the sixth image point.

Given this minimum number of correspondences, the solution is exact, i.e. the space points are projected exactly onto their measured images. The solution is obtained by solving $\mathtt{A}\mathbf{p} = \mathbf{0}$ where $\mathtt{A}$ is an 11×12 matrix in this case. In general $\mathtt{A}$ will have rank 11, and the solution vector $\mathbf{p}$ is the 1-dimensional right null-space of $\mathtt{A}$.

Over-determined solution. If the data is not exact, because of noise in the point coordinates, and $n \geq 6$ point correspondences are given, then there will not be an exact solution to the equations $\mathtt{A}\mathbf{p} = \mathbf{0}$. As in the estimation of a homography a solution for $\mathtt{P}$ may be obtained by minimizing an algebraic or geometric error.

In the case of algebraic error the approach is to minimize $\|\mathtt{A}\mathbf{p}\|$ subject to some normalization constraint. Possible constraints are

(i) $\|\mathbf{p}\| = 1$;
(ii) $\|\hat{\mathbf{p}}^3\| = 1$, where $\hat{\mathbf{p}}^3$ is the vector $(p_{31}, p_{32}, p_{33})^\top$, namely the first three entries in the last row of $\mathtt{P}$.

The first of these is preferred for routine use and will be used for the moment. We will return to the second normalization constraint in section 7.2.1. In either case, the residual $\mathtt{A}\mathbf{p}$ is known as the *algebraic error*. Using these equations, the complete DLT algorithm for computation of the camera matrix $\mathtt{P}$ proceeds in the same manner as that for $\mathtt{H}$ given in algorithm 4.1(p91).

Degenerate configurations. Analysis of the degenerate configurations for estimation of $\mathtt{P}$ is rather more involved than in the case of the 2D homography. There are two types of configurations in which ambiguous solutions exist for $\mathtt{P}$. These configurations will be investigated in detail in chapter 22. The most important critical configurations are as follows:

(i) The camera and points all lie on a twisted cubic.

Samsung/Google, EX1010, p. 197

(ii) The points all lie on the union of a plane and a single straight line containing the camera centre.

For such configurations, the camera cannot be obtained uniquely from the images of the points. Instead, it may move arbitrarily along the twisted cubic, or straight line respectively. If data is close to a degenerate configuration then a poor estimate for P is obtained. For example, if the camera is distant from a scene with low relief, such as a near-nadir aerial view, then this situation is close to the planar degeneracy.

Data normalization. It is important to carry out some sort of data normalization just as in the 2D homography estimation case. The points $\mathbf{x}_i$ in the image are appropriately normalized in the same way as before. Namely the points should be translated so that their centroid is at the origin, and scaled so that their RMS (root-mean-squared) distance from the origin is $\sqrt{2}$. What normalization should be applied to the 3D points $\mathbf{X}_i$ is a little more problematical. In the case where the variation in depth of the points from the camera is relatively slight it makes sense to carry out the same sort of normalization. Thus, the centroid of the points is translated to the origin, and their coordinates are scaled so that the RMS distance from the origin is $\sqrt{3}$ (so that the "average" point has coordinates of magnitude $(1, 1, 1, 1)^\top$). This approach is suitable for a compact distribution of points, such as those on the calibration object of figure 7.1.

In the case where there are some points that lie at a great distance from the camera, the previous normalization technique does not work well. For instance, if there are points close to the camera, as well as points that lie at infinity (which are imaged as vanishing points) or close to infinity, as may occur in oblique views of terrain, then it is not possible or reasonable to translate the points so that their centroid is at the origin. The normalization method described in exercise (iii) on page 128 would be more appropriately used in such a case, though this has not been thoroughly tested.

With appropriate normalization the estimate of P is carried out in the same manner as algorithm 4.2(p109) for H.

Line correspondences. It is a simple matter to extend the DLT algorithm to take account of line correspondences as well. A line in 3D may be represented by two points $\mathbf{X}_0$ and $\mathbf{X}_1$ through which the line passes. Now, according to result 8.2(p197) the plane formed by back-projecting from the image line l is equal to $\mathtt{P}^\top\mathbf{l}$. The condition that the point $\mathbf{X}_j$ lies on this plane is then

$$\mathbf{l}^\top\mathtt{P}\mathbf{X}_j = 0 \quad \text{for } j = 0, 1. \tag{7.3}$$

Each choice of j gives a single linear equation in the entries of the matrix P, so two equations are obtained for each 3D to 2D line correspondence. These equations, being linear in the entries of P, may be added to the equations (7.1) obtained from point correspondences and a solution to the composite equation set may be computed.

7.2 Geometric error

As in the case of 2D homographies (chapter 4), one may define geometric error. Suppose for the moment that world points $\mathbf{X}_i$ are known far more accurately than the

Samsung/Google, EX1010, p. 198

Objective

Given $n \geq 6$ world to image point correspondences $\{\mathbf{X}_i \leftrightarrow \mathbf{x}_i\}$, determine the Maximum Likelihood estimate of the camera projection matrix $\mathtt{P}$, i.e. the $\mathtt{P}$ which minimizes $\sum_i d(\mathbf{x}_i, \mathtt{P}\mathbf{X}_i)^2$.

Algorithm

(i) **Linear solution.** Compute an initial estimate of $\mathtt{P}$ using a linear method such as algorithm 4.2(p109):

  (a) **Normalization:** Use a similarity transformation $\mathtt{T}$ to normalize the image points, and a second similarity transformation $\mathtt{U}$ to normalize the space points. Suppose the normalized image points are $\tilde{\mathbf{x}}_i = \mathtt{T}\mathbf{x}_i$, and the normalized space points are $\tilde{\mathbf{X}}_i = \mathtt{U}\mathbf{X}_i$.

  (b) **DLT:** Form the $2n \times 12$ matrix $\mathtt{A}$ by stacking the equations (7.2) generated by each correspondence $\tilde{\mathbf{X}}_i \leftrightarrow \tilde{\mathbf{x}}_i$. Write $\mathbf{p}$ for the vector containing the entries of the matrix $\tilde{\mathtt{P}}$. A solution of $\mathtt{A}\mathbf{p} = \mathbf{0}$, subject to $\|\mathbf{p}\| = 1$, is obtained from the unit singular vector of $\mathtt{A}$ corresponding to the smallest singular value.

(ii) **Minimize geometric error.** Using the linear estimate as a starting point minimize the geometric error (7.4):

$$\sum_i d(\tilde{\mathbf{x}}_i, \tilde{\mathtt{P}}\tilde{\mathbf{X}}_i)^2$$

over $\tilde{\mathtt{P}}$, using an iterative algorithm such as Levenberg–Marquardt.

(iii) **Denormalization.** The camera matrix for the original (unnormalized) coordinates is obtained from $\tilde{\mathtt{P}}$ as

$$\mathtt{P} = \mathtt{T}^{-1}\tilde{\mathtt{P}}\mathtt{U}.$$

Algorithm 7.1. *The Gold Standard algorithm for estimating* $\mathtt{P}$ *from world to image point correspondences in the case that the world points are very accurately known.*

measured image points. For example the points $\mathbf{X}_i$ might arise from an accurately machined calibration object. Then the geometric error in the image is

$$\sum_i d(\mathbf{x}_i, \tilde{\mathbf{x}}_i)^2$$

where $\mathbf{x}_i$ is the measured point and $\tilde{\mathbf{x}}_i$ is the point $\mathtt{P}\mathbf{X}_i$, i.e. the point which is the exact image of $\mathbf{X}_i$ under $\mathtt{P}$. If the measurement errors are Gaussian then the solution of

$$\min_{\mathtt{P}} \sum_i d(\mathbf{x}_i, \mathtt{P}\mathbf{X}_i)^2 \tag{7.4}$$

is the Maximum Likelihood estimate of $\mathtt{P}$.

Just as in the 2D homography case, minimizing geometric error requires the use of iterative techniques, such as Levenberg–Marquardt. A parametrization of $\mathtt{P}$ is required, and the vector of matrix elements $\mathbf{p}$ provides this. The DLT solution, or a minimal solution, may be used as a starting point for the iterative minimization. The complete Gold Standard algorithm is summarized in algorithm 7.1.

Example 7.1. Camera estimation from a calibration object

We will compare the DLT algorithm with the Gold Standard algorithm 7.1 for data

Samsung/Google, EX1010, p. 199



Fig. 7.1. *An image of a typical calibration object. The black and white checkerboard pattern (a "Tsai grid") is designed to enable the positions of the corners of the imaged squares to be obtained to high accuracy. A total of 197 points were identified and used to calibrate the camera in the examples of this chapter.*

	f_y	f_x/f_y	skew	x_0	y_0	residual
linear	1673.3	1.0063	1.39	379.96	305.78	0.365
iterative	1675.5	1.0063	1.43	379.79	305.25	0.364

Table 7.1. *DLT and Gold Standard calibration.*

from the calibration object shown in figure 7.1. The image points $\mathbf{x}_i$ are obtained from the calibration object using the following steps:

(i) Canny edge detection [Canny-86].
(ii) Straight line fitting to the detected linked edges.
(iii) Intersecting the lines to obtain the imaged corners.

If sufficient care is taken the points $\mathbf{x}_i$ are obtained to a localization accuracy of far better than $1/10$ of a pixel. A rule of thumb is that for a good estimation the number of constraints (point measurements) should exceed the number of unknowns (the 11 camera parameters) by a factor of five. This means that at least 28 points should be used.

Table 7.1 shows the calibration results obtained by using the linear DLT method and the Gold Standard method. Note that the improvement achieved using the Gold Standard algorithm is very slight. The difference of residual of one thousandth of a pixel is insignificant. △

Errors in the world points

It may be the case that world points are not measured with "infinite" accuracy. In this case one may choose to estimate P by minimizing a 3D geometric error, or an image geometric error, or both.

If only errors in the world points are considered then the 3D geometric error is defined as

$$\sum_i d(\mathbf{X}_i, \hat{\mathbf{X}}_i)^2$$

Samsung/Google, EX1010, p. 200

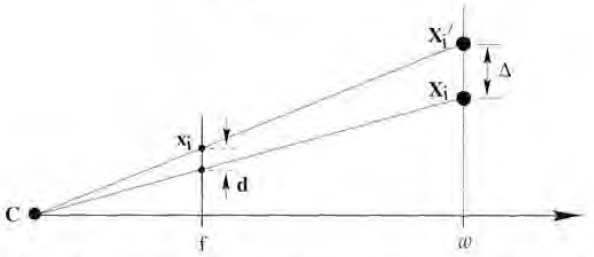


Fig. 7.2. *The DLT algorithm minimizes the sum of squares of geometric distance Δ between the point $\mathbf{X}_i$ and the point $\mathbf{X}_i'$ mapping exactly onto $\mathbf{x}_i$ and lying in the plane through $\mathbf{X}_i$ parallel to the principal plane of the camera. A short calculation shows that $wd = f\Delta$.*

where $\widehat{\mathbf{X}}_i$ is the closest point in space to $\mathbf{X}_i$ that maps exactly onto $\mathbf{x}_i$ via $\mathbf{x}_i = \mathtt{P}\widehat{\mathbf{X}}_i$.

More generally, if errors in both the world and image points are considered, then a weighted sum of world and image errors is minimized. As in the 2D homography case, this requires that one augment the set of parameters by including parameters $\widehat{\mathbf{X}}_i$, the estimated 3D points. One minimizes

$$\sum_{i=1}^{n} d_{\text{Mah}}(\mathbf{x}_i, \mathtt{P}\widehat{\mathbf{X}}_i)^2 + d_{\text{Mah}}(\mathbf{X}_i, \widehat{\mathbf{X}}_i)^2$$

where d_{Mah} represents Mahalanobis distance with respect to the known error covariance matrices for each of the measurements $\mathbf{x}_i$ and $\mathbf{X}_i$. In the simplest case, the Mahalanobis distance is simply a weighted geometric distance, where the weights are chosen to reflect the relative accuracy of measurements of the image and 3D points, and also the fact that image and world points are typically measured in different units.

7.2.1 Geometric interpretation of algebraic error

Suppose all the points $\mathbf{X}_i$ in the DLT algorithm are normalized such that $\mathbf{X}_i = (\mathrm{X}_i, \mathrm{Y}_i, \mathrm{Z}_i, 1)^\mathsf{T}$, and $\mathbf{x}_i = (x_i, y_i, 1)^\mathsf{T}$. In this case, it was seen in section 4.2.4-(*p*95) that the quantity being minimized by the DLT algorithm is $\sum_i (\bar{w}_i d(\mathbf{x}_i, \hat{\mathbf{x}}_i))^2$, where $\bar{w}_i(\hat{x}_i, \hat{y}_i, 1)^\mathsf{T} = \mathtt{P}\mathbf{X}_i$. However, according to (6.15–*p*162),

$$\bar{w}_i = \pm \|\tilde{\mathbf{p}}^3\| \operatorname{depth}(\mathbf{X}; \mathtt{P}) \ .$$

Thus, the value $\bar{w}_i$ may be interpreted as the depth of the point $\mathbf{X}_i$ from the camera in the direction along the principal ray, provided the camera is normalized so that $\|\tilde{\mathbf{p}}^3\|^2 = p_{31}^2 + p_{32}^2 + p_{33}^2 = 1$. Referring to figure 7.2 one sees that $\bar{w}_i d(\mathbf{x}_i, \hat{\mathbf{x}}_i)$ is proportional to $f d(\mathbf{X}', \mathbf{X})$, where f is the focal length and $\mathbf{X}_i'$ is a point mapping to $\mathbf{x}_i$ and lying in a plane through $\mathbf{X}_i$ parallel to the principal plane of the camera. Thus, the algebraic error being minimized is equal to $f \sum_i d(\mathbf{X}_i, \mathbf{X}_i')^2$.

The distance $d(\mathbf{X}_i, \mathbf{X}_i')$ is the correction that needs to be made to the measured 3D points in order to correspond precisely with the measured image points $\mathbf{x}_i$. The restriction is that the correction must be made in the direction perpendicular to the principal axis of the camera. Because of this restriction, the point $\mathbf{X}_i'$ is not the same as the closest point $\widehat{\mathbf{X}}_i$ to $\mathbf{X}_i$ that maps to $\mathbf{x}_i$. However, for points $\mathbf{X}_i$ not too far from the principal

Samsung/Google, EX1010, p. 201

ray of the camera, the distance $d(\mathbf{X}_i, \mathbf{X}_i')$ is a reasonable approximation to the distance $d(\mathbf{X}_i, \widehat{\mathbf{X}}_i)$. The DLT slightly weights the points farther away from the principal ray by minimizing the squared sum of $d(\mathbf{X}_i, \mathbf{X}_i')$, which is slightly larger than $d(\mathbf{X}_i, \widehat{\mathbf{X}}_i)$. In addition, the presence of the focal length f in the expression for algebraic error suggests that the DLT algorithm will be biased towards minimizing focal length at a cost of a slight increase in 3D geometric error.

Transformation invariance. We have just seen that by minimizing $\|\mathtt{A}\mathbf{p}\|$ subject to the constraint $\|\hat{\mathbf{p}}^3\| = 1$ one may interpret the solution in terms of minimizing 3D geometric distances. Such an interpretation is not affected by similarity transformations in either 3D space or the image space. Thus, one is led to expect that carrying out translation and scaling of the data, either in the image or in 3D point coordinates, will not have any effect on the solutions. This is indeed the case as may be shown using the arguments of section 4.4.2(p105).

7.2.2 Estimation of an affine camera

The methods developed above for the projective cameras can be applied directly to affine cameras. An affine camera is one for which the projection matrix has last row $(0, 0, 0, 1)$. In the DLT estimation of the camera in this case one minimizes $\|\mathtt{A}\mathbf{p}\|$ subject to this condition on the last row of $\mathtt{P}$. As in the case of computing 2D affine transformations, for affine cameras, algebraic error and geometric image error are equal. This means that geometric image distances may be minimized by a linear algorithm.

Suppose as above that all the points $\mathbf{X}_i$ are normalized such that $\mathbf{X}_i = (\mathtt{X}_i, \mathtt{Y}_i, \mathtt{Z}_i, 1)^\mathsf{T}$, and $\mathbf{x}_i = (x_i, y_i, 1)^\mathsf{T}$, and also that the last row of $\mathtt{P}$ has the affine form. Then (7.2) for a single correspondence reduces to

$$\begin{bmatrix} \mathbf{0}^\mathsf{T} & -\mathbf{X}_i^\mathsf{T} \\ \mathbf{X}_i^\mathsf{T} & \mathbf{0}^\mathsf{T} \end{bmatrix} \begin{pmatrix} \mathbf{P}^1 \\ \mathbf{P}^2 \end{pmatrix} + \begin{pmatrix} y_i \\ -x_i \end{pmatrix} = \mathbf{0} \tag{7.5}$$

which shows that the squared algebraic error in this case equals the squared geometric error

$$\|\mathtt{A}\mathbf{p}\|^2 = \sum_i \left(x_i - \mathbf{P}^{1\mathsf{T}}\mathbf{X}_i\right)^2 + \left(y_i - \mathbf{P}^{2\mathsf{T}}\mathbf{X}_i\right)^2 = \sum_i d(\mathbf{x}_i, \hat{\mathbf{x}}_i)^2.$$

This result may also be seen geometrically by comparison of figure 6.8(p170) and figure 7.2.

A linear estimation algorithm for an affine camera which minimizes geometric error is given in algorithm 7.2. Under the assumption of Gaussian measurement errors this is the Maximum Likelihood estimate of $\mathtt{P}_\mathtt{A}$.

7.3 Restricted camera estimation

The DLT algorithm, as it has so far been described, computes a general projective camera matrix $\mathtt{P}$ from a set of 3D to 2D point correspondences. The matrix $\mathtt{P}$ with centre at a finite point may be decomposed as $\mathtt{P} = \mathtt{K}[\mathtt{R} \mid -\mathtt{R}\widetilde{\mathtt{C}}]$ where $\mathtt{R}$ is a 3×3

Samsung/Google, EX1010, p. 202

Objective

Given $n \geq 4$ world to image point correspondences $\{\mathbf{X}_i \leftrightarrow \mathbf{x}_i\}$, determine the Maximum Likelihood Estimate of the affine camera projection matrix $\mathtt{P}_A$, i.e. the camera $\mathtt{P}$ which minimizes $\sum_i d(\mathbf{x}_i, \mathtt{P}\mathbf{X}_i)^2$ subject to the affine constraint $\mathbf{P}^{3\mathsf{T}} = (0, 0, 0, 1)$.

Algorithm

(i) **Normalization:** Use a similarity transformation $\mathtt{T}$ to normalize the image points, and a second similarity transformation $\mathtt{U}$ to normalize the space points. Suppose the normalized image points are $\tilde{\mathbf{x}}_i = \mathtt{T}\mathbf{x}_i$, and the normalized space points are $\tilde{\mathbf{X}}_i = \mathtt{U}\mathbf{X}_i$, with unit last component.

(ii) Each correspondence $\tilde{\mathbf{X}}_i \leftrightarrow \tilde{\mathbf{x}}_i$ contributes (from (7.5)) equations

$$\begin{bmatrix} \tilde{\mathbf{X}}_i^{\mathsf{T}} & \mathbf{0}^{\mathsf{T}} \\ \mathbf{0}^{\mathsf{T}} & \tilde{\mathbf{X}}_i^{\mathsf{T}} \end{bmatrix} \begin{pmatrix} \tilde{\mathbf{P}}^1 \\ \tilde{\mathbf{P}}^2 \end{pmatrix} = \begin{pmatrix} \tilde{x}_i \\ \tilde{y}_i \end{pmatrix}$$

which are stacked into a $2n \times 8$ matrix equation $\mathtt{A}_8\mathbf{p}_8 = \mathbf{b}$, where $\mathbf{p}_8$ is the 8-vector containing the first two rows of $\tilde{\mathtt{P}}_A$.

(iii) The solution is obtained by the pseudo-inverse of $\mathtt{A}_8$ (see section A5.2(*p*590))

$$\mathbf{p}_8 = \mathtt{A}_8^+ \mathbf{b}$$

and $\tilde{\mathbf{P}}^{3\mathsf{T}} = (0, 0, 0, 1)$.

(iv) **Denormalization:** The camera matrix for the original (unnormalized) coordinates is obtained from $\tilde{\mathtt{P}}_A$ as

$$\mathtt{P}_A = \mathtt{T}^{-1}\tilde{\mathtt{P}}_A\mathtt{U}$$

Algorithm 7.2. *The Gold Standard Algorithm for estimating an affine camera matrix* $\mathtt{P}_A$ *from world to image correspondences.*

rotation matrix and $\mathtt{K}$ has the form (6.10–*p*157):

$$\mathtt{K} = \begin{bmatrix} \alpha_x & s & x_0 \\ & \alpha_y & y_0 \\ & & 1 \end{bmatrix} . \tag{7.6}$$

The non-zero entries of $\mathtt{K}$ are geometrically meaningful quantities, the internal calibration parameters of $\mathtt{P}$. One may wish to find the best-fit camera matrix $\mathtt{P}$ subject to restrictive conditions on the camera parameters. Common assumptions are

(i) The skew s is zero.
(ii) The pixels are square: $\alpha_x = \alpha_y$.
(iii) The principal point (x_0, y_0) is known.
(iv) The complete camera calibration matrix $\mathtt{K}$ is known.

In some cases it is possible to estimate a restricted camera matrix with a linear algorithm (see the exercises at the end of the chapter).

As an example of restricted estimation, suppose that we wish to find the best pinhole camera model (that is projective camera with $s = 0$ and $\alpha_x = \alpha_y$) that fits a set of point measurements. This problem may be solved by minimizing either geometric or algebraic error, as will be discussed next.

Samsung/Google, EX1010, p. 203

Minimizing geometric error. To minimize geometric error, one selects a set of parameters that characterize the camera matrix to be computed. For instance, suppose we wish to enforce the constraints $s = 0$ and $\alpha_x = \alpha_y$. One can parametrize the camera matrix using the remaining 9 parameters. These are x_0, y_0, α, plus 6 parameters representing the orientation $\mathtt{R}$ and location $\tilde{\mathtt{C}}$ of the camera. Let this set of parameters be denoted collectively by $\mathbf{q}$. The camera matrix $\mathtt{P}$ may then be explicitly computed in terms of the parameters.

The geometric error may then be minimized with respect to the set of parameters using iterative minimization (such as Levenberg–Marquardt). Note that in the case of minimization of image error only, the size of the minimization problem is $9 \times 2n$ (supposing 9 unknown camera parameters). In other words the LM minimization is minimizing a function $f : \mathbb{R}^9 \to \mathbb{R}^{2n}$. In the case of minimization of 3D and 2D error, the function f is from $\mathbb{R}^{3n+9} \to \mathbb{R}^{5n}$, since the 3D points must be included among the measurements and minimization also includes estimation of the true positions of the 3D points.

Minimizing algebraic error. It is possible to minimize algebraic error instead, in which case the iterative minimization problem becomes much smaller, as will be explained next. Consider the parametrization map taking a set of parameters $\mathbf{q}$ to the corresponding camera matrix $\mathtt{P} = \mathtt{K}[\mathtt{R} \mid -\mathtt{R}\tilde{\mathtt{C}}]$. Let this map be denoted by g. Effectively, one has a map $\mathbf{p} = g(\mathbf{q})$, where $\mathbf{p}$ is the vector of entries of the matrix $\mathtt{P}$. Minimizing algebraic error over all point matches is equivalent to minimizing $\|\mathtt{A}g(\mathbf{q})\|$.

The reduced measurement matrix. In general, the $2n \times 12$ matrix $\mathtt{A}$ may have a very large number of rows. It is possible to replace $\mathtt{A}$ by a square 12×12 matrix $\hat{\mathtt{A}}$ such that $\|\mathtt{A}\mathbf{p}\| = \mathbf{p}^\mathsf{T}\mathtt{A}^\mathsf{T}\mathtt{A}\mathbf{p} = \|\hat{\mathtt{A}}\mathbf{p}\|$ for any vector $\mathbf{p}$. Such a matrix $\hat{\mathtt{A}}$ is called a *reduced measurement matrix*. One way to do this is using the Singular Value Decomposition (SVD). Let $\mathtt{A} = \mathtt{U}\mathtt{D}\mathtt{V}^\mathsf{T}$ be the SVD of $\mathtt{A}$, and define $\hat{\mathtt{A}} = \mathtt{D}\mathtt{V}^\mathsf{T}$. Then

$$\mathtt{A}^\mathsf{T}\mathtt{A} = (\mathtt{V}\mathtt{D}\mathtt{U}^\mathsf{T})(\mathtt{U}\mathtt{D}\mathtt{V}^\mathsf{T}) = (\mathtt{V}\mathtt{D})(\mathtt{D}\mathtt{V}^\mathsf{T}) = \hat{\mathtt{A}}^\mathsf{T}\hat{\mathtt{A}}$$

as required. Another way of obtaining $\hat{\mathtt{A}}$ is to use the QR decomposition $\mathtt{A} = \mathtt{Q}\hat{\mathtt{A}}$, where $\mathtt{Q}$ has orthogonal columns and $\hat{\mathtt{A}}$ is upper triangular and square.

Note that the mapping $\mathbf{q} \mapsto \hat{\mathtt{A}}g(\mathbf{q})$ is a mapping from $\mathbb{R}^9$ to $\mathbb{R}^{12}$. This is a simple parameter-minimization problem that may be solved using the Levenberg–Marquardt method. The important point to note is the following:

- *Given a set of n world to image correspondences, $\mathbf{X}_i \leftrightarrow \mathbf{x}_i$, the problem of finding a constrained camera matrix $\mathtt{P}$ that minimizes the sum of algebraic distances $\sum_i d_{alg}(\mathbf{x}_i, \mathtt{P}\mathbf{X}_i)^2$ reduces to the minimization of a function $\mathbb{R}^9 \to \mathbb{R}^{12}$, independent of the number n of correspondences.*

Minimization of $\|\hat{\mathtt{A}}g(\mathbf{q})\|$ takes place over all values of the parameters $\mathbf{q}$. Note that if $\mathtt{P} = \mathtt{K}[\mathtt{R} \mid -\mathtt{R}\tilde{\mathtt{C}}]$ with $\mathtt{K}$ as in (7.6) then $\mathtt{P}$ satisfies the condition $p_{31}^2 + p_{32}^2 + p_{33}^2 = 1$, since these entries are the same as the last row of the rotation matrix $\mathtt{R}$. Thus, minimizing $\mathtt{A}g(\mathbf{q})$ will lead to a matrix $\mathtt{P}$ satisfying the constraints $s = 0$ and $\alpha_x = \alpha_y$ and scaled

Samsung/Google, EX1010, p. 204

such that $p_{31}^2 + p_{32}^2 + p_{33}^2 = 1$, and which in addition minimizes the algebraic error for all point correspondences.

Initialization. One way of finding camera parameters to initialize the iteration is as follows.

(i) Use a linear algorithm such as DLT to find an initial camera matrix.
(ii) Clamp fixed parameters to their desired values (for instance set $s = 0$ and set $\alpha_x = \alpha_y$ to the average of their values obtained using DLT).
(iii) Set variable parameters to their values obtained by decomposition of the initial camera matrix (see section 6.2.4).

Ideally, the assumed values of the fixed parameters will be close to the values obtained by the DLT. However, in practice this is not always the case. Then altering these parameters to their desired values results in an incorrect initial camera matrix that may lead to large residuals, and difficulty in converging. A method which works better in practice is to use *soft* constraints by adding extra terms to the cost function. Thus, for the case where $s = 0$ and $\alpha_x = \alpha_y$, one adds extra terms $ws^2 + w(\alpha_x - \alpha_y)^2$ to the cost function. In the case of geometric image error, the cost function becomes

$$\sum_i d(\mathbf{x}_i, \mathbf{P}\mathbf{X}_i)^2 + ws^2 + w(\alpha_x - \alpha_y)^2 \ .$$

One begins with the values of the parameters estimated using the DLT. The weights begin with low values and are increased at each iteration of the estimation procedure. Thus, the values of s and the aspect ratio are drawn gently to their desired values. Finally they may be clamped to their desired values for a final estimation.

Exterior orientation. Suppose that all the internal parameters of the camera are known, then all that remains to be determined are the position and orientation (or *pose*) of the camera. This is the "exterior orientation" problem, which is important in the analysis of calibrated systems.

To compute the exterior orientation a configuration with accurately known position in a world coordinate frame is imaged. The pose of the camera is then sought. Such a situation arises in hand–eye calibration for robotic systems, where the position of the camera is required, and also in model-based recognition using alignment where the position of an object relative to the camera is required.

There are six parameters that must be determined, three for the orientation and three for the position. As each world to image point correspondence generates two constraints it would be expected that three points are sufficient. This is indeed the case, and the resulting non-linear equations have four solutions in general.

Experimental evaluation

Results of constrained estimation for the calibration grid of example 7.1 are given in table 7.2.

Both the algebraic and geometric minimization involve an iterative minimization

Samsung/Google, EX1010, p. 205

	f_y	f_x/f_y	skew	x_0	y_0	residual
algebraic	1633.4	1.0	0.0	371.21	293.63	0.601
geometric	1637.2	1.0	0.0	371.32	293.69	0.601

Table 7.2. *Calibration for a restricted camera matrix.*

over 9 parameters. However, the algebraic method is far quicker, since it minimizes only 12 errors, instead of $2n = 396$ in the geometric minimization. Note that fixing skew and aspect ratio has altered the values of the other parameters (compare table 7.1) and increased the residual.

Covariance estimation. The techniques of covariance estimation and propagation of the errors into an image may be handled in just the same way as in the 2D homography case (chapter 5). Similarly, the minimum expected residual error may be computed as in result 5.2(p136). Assuming that all errors are in the image measurements, the expected ML residual error is equal to

$$\epsilon_{\text{res}} = \sigma(1 - d/2n)^{1/2} \ .$$

where d is the number of camera parameters being fitted (11 for a full pinhole camera model). This formula may also be used to estimate the accuracy of the point measurements, given a residual error. In the case of example 7.1 where $n = 197$ and $\epsilon_{\text{res}} = 0.365$ this results in a value of $\sigma = 0.37$. This value is greater than expected. The reason, as we will see later, lies in the camera model – we are ignoring radial distortion.

Example 7.2. Covariance ellipsoid for an estimated camera
Suppose that the camera is estimated using the Maximum Likelihood (Gold Standard) method, optimizing over a set of camera parameters. The estimated covariance of the point measurements can then be used to compute the covariance of the camera model by back-propagation, according to result 5.10(p142). This gives $\Sigma_{\text{camera}} = (\mathrm{J}^{\mathsf{T}}\Sigma_{\text{points}}^{-1}\mathrm{J})^{-1}$ where J is the Jacobian matrix of the measured points in terms of the camera parameters. Uncertainty in 3D world points may also be taken into account in this way. If the camera is parametrized in terms of meaningful parameters (such as camera position), then the variance of each parameter can be measured directly from the diagonal entries of the covariance matrix.

Knowing the covariance of the camera parameters, error bounds or ellipsoids can be computed. For instance, from the covariance matrix for all the parameters we may extract the subblock representing the 3×3 covariance matrix of the camera position, $\Sigma_{\mathbf{C}}$. A confidence ellipsoid for the camera centre is then defined by

$$(\mathbf{C} - \bar{\mathbf{C}})^{\mathsf{T}}\Sigma_{\mathbf{C}}^{-1}(\mathbf{C} - \bar{\mathbf{C}}) = k^2$$

where k^2 is computed from the inverse cumulative χ_n^2 distribution in terms of the desired confidence level α: namely $k^2 = F_n^{-1}(\alpha)$ (see figure A2.1(p567)). Here n is the

Samsung/Google, EX1010, p. 206

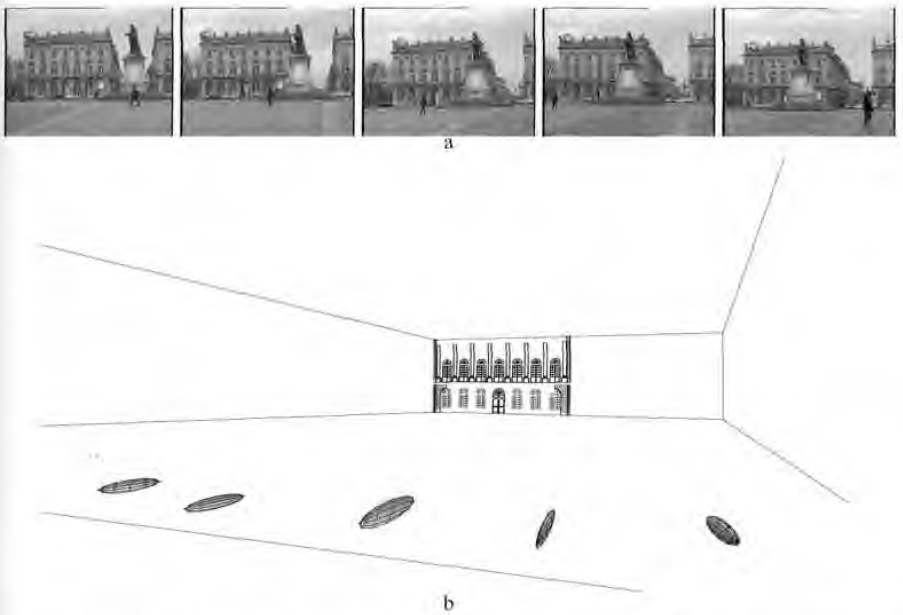


Fig. 7.3. **Camera centre covariance ellipsoids**. *(a) Five images of Stanislas square (Nancy, France), for which 3D calibration points are known. (b) Camera centre covariance ellipsoids corresponding to each image, computed for cameras estimated from the imaged calibration points. Note, the typical cigar shape of the ellipsoid aligned towards the scene data. Figure courtesy of Vincent Lepetit, Marie-Odile Berger and Gilles Simon.*

number of variables – that is 3 in the case of the camera centre. With the chosen level of certainty α, the camera centre lies inside the ellipsoid.

Figure 7.3 shows an example of ellipsoidal uncertainty regions for computed camera centres. Given the estimated covariance matrix for the computed camera, the techniques of section 5.2.6(p148) may be used to compute the uncertainty in the image positions of further 3D world points. △

7.4 Radial distortion

The assumption throughout these chapters has been that a linear model is an accurate model of the imaging process. Thus the world point, image point and optical centre are collinear, and world lines are imaged as lines and so on. For real (non-pinhole) lenses this assumption will not hold. The most important deviation is generally a radial distortion. In practice this error becomes more significant as the focal length (and price) of the lens decreases. See figure 7.4.

The cure for this distortion is to correct the image measurements to those that would have been obtained under a perfect linear camera action. The camera is then effectively again a linear device. This process is illustrated in figure 7.5. This correction must

Samsung/Google, EX1010, p. 207



a b

Fig. 7.4. *(a) Short vs (b) long focal lengths. Note the curved imaged lines at the periphery in (a) which are images of straight scene lines.*

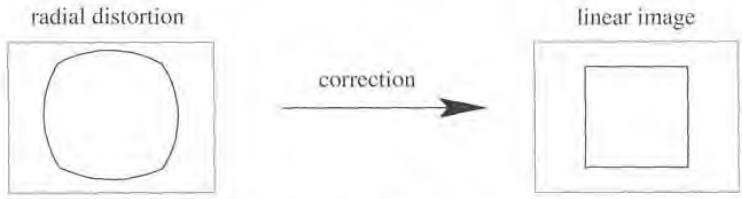


Fig. 7.5. *The image of a square with significant radial distortion is corrected to one that would have been obtained under a perfect linear lens.*

be carried out in the right place in the projection process. Lens distortion takes place during the initial projection of the world onto the image plane, according to (6.2–*p*154). Subsequently, the calibration matrix (7.6) reflects a choice of affine coordinates in the image, translating physical locations in the image plane to pixel coordinates.

We will denote the image coordinates of a point under ideal (non-distorted) pinhole projection by $(\tilde{x}, \tilde{y})$, measured in units of focal-length. Thus, for a point $\mathbf{X}$ we have (see (6.5–*p*155))

$$(\tilde{x}, \tilde{y}, 1)^\top = [\mathtt{I} \mid 0]\mathbf{X}_{\text{cam}}$$

where $\mathbf{X}_{\text{cam}}$ is the 3D point in camera coordinates, related to world coordinates by (6.6–*p*156). The actual projected point is related to the ideal point by a radial displacement. Thus, radial (lens) distortion is modelled as

$$\begin{pmatrix} x_d \\ y_d \end{pmatrix} = L(\tilde{r}) \begin{pmatrix} \tilde{x} \\ \tilde{y} \end{pmatrix} \tag{7.7}$$

where

- $(\tilde{x}, \tilde{y})$ is the ideal image position (which obeys linear projection).
- (x_d, y_d) is the actual image position, after radial distortion.
- $\tilde{r}$ is the radial distance $\sqrt{\tilde{x}^2 + \tilde{y}^2}$ from the centre for radial distortion.
- $L(\tilde{r})$ is a distortion factor, which is a function of the radius $\tilde{r}$ only.

Samsung/Google, EX1010, p. 208

Correction of distortion. In pixel coordinates the correction is written

$$\hat{x} = x_c + L(r)(x - x_c) \qquad \hat{y} = y_c + L(r)(y - y_c).$$

where (x, y) are the measured coordinates, $(\hat{x}, \hat{y})$ are the corrected coordinates, and (x_c, y_c) is the centre of radial distortion, with $r^2 = (x - x_c)^2 + (y - y_c)^2$. Note, if the aspect ratio is not unity then it is necessary to correct for this when computing r. With this correction the coordinates $(\hat{x}, \hat{y})$ are related to the coordinates of the 3D world point by a linear projective camera.

Choice of the distortion function and centre. The function $L(r)$ is only defined for positive values of r and $L(0) = 1$. An approximation to an arbitrary function $L(r)$ may be given by a Taylor expansion $L(r) = 1 + \kappa_1 r + \kappa_2 r^2 + \kappa_3 r^3 + \ldots$. The coefficients for radial correction $\{\kappa_1, \kappa_2, \kappa_3, \ldots, x_c, y_c\}$ are considered part of the interior calibration of the camera. The principal point is often used as the centre for radial distortion, though these need not coincide exactly. This correction, together with the camera calibration matrix, specifies the mapping from an image point to a ray in the camera coordinate system.

Computing the distortion function. The function $L(r)$ may be computed by minimizing a cost based on the deviation from a linear mapping. For example, algorithm 7.1(p181) estimates P by minimizing geometric image error for calibration objects such as the Tsai grids of figure 7.1. The distortion function may be included as part of the imaging process, and the parameters κ_i computed together with P during the iterative minimization of the geometric error. Similarly, the distortion function may be computed when estimating the homography between a single Tsai grid and its image.

A simple and more general approach is to determine $L(r)$ by the requirement that images of straight scene lines should be straight. A cost function is defined on the imaged lines (such as the distance between the line joining the imaged line's ends and its mid-point) after the corrective mapping by $L(r)$. This cost is iteratively minimized over the parameters κ_i of the distortion function and the centre of radial distortion. This is a very practical method for images of urban scenes since there are usually plenty of lines available. It has the advantage that no special calibration pattern is required as the scene provides the calibration entities.

Example 7.3. Radial correction. The function $L(r)$ is computed for the image of figure 7.6a by minimizing a cost based on the straightness of imaged scene lines. The image is 640×480 pixels and the correction and centre are computed as $\kappa_1 = 0.103689$, $\kappa_2 = 0.00487908$, $\kappa_3 = 0.00116894$, $\kappa_4 = 0.000841614$, $x_c = 321.87$, $y_c = 241.18$ pixels, where pixels are normalized by the average half-size of the image. This is a correction by 30 pixels at the image periphery. The result of warping the image is shown in figure 7.6b. △

Example 7.4. We continue with the example of the calibration grid shown in figure 7.1 and discussed in example 7.1(p181). Radial distortion was removed by the straight line

Samsung/Google, EX1010, p. 209



a b

Fig. 7.6. **Radial distortion correction.** *(a) The original image with lines which are straight in the world, but curved in the image. Several of these lines are annotated by dashed curves. (b) The image warped to remove the radial distortion. Note that the lines in the periphery of the image are now straight, but that the boundary of the image is curved.*

method, and then the camera calibrated using the methods described in this chapter. The results are given in table 7.3.

Note that the residuals after radial correction are substantially smaller. Estimation of the error of point measurements from the residual leads to a value of $\sigma = 0.18$ pixels. Since radial distortion involves selective stretching of the image, it is quite plausible that the effective focal length of the image is changed, as seen here. △

	f_y	f_x/f_y	skew	x_0	y_0	residual
linear	1580.5	1.0044	0.75	377.53	299.12	0.179
iterative	1580.7	1.0044	0.70	377.42	299.02	0.179
algebraic	1556.0	1.0000	0.00	372.42	291.86	0.381
iterative	1556.6	1.0000	0.00	372.41	291.86	0.380
linear	1673.3	1.0063	1.39	379.96	305.78	0.365
iterative	1675.5	1.0063	1.43	379.79	305.25	0.364
algebraic	1633.4	1.0000	0.00	371.21	293.63	0.601
iterative	1637.2	1.0000	0.00	371.32	293.69	0.601

Table 7.3. **Calibration with and without radial distortion correction.** *The results above the line are after radial correction – the results below for comparison are without radial distortion (from the previous tables). The upper two methods in each case solve for the general camera model, the lower two are for a constrained model with square pixels.*

In correcting for radial distortion, it is often not actually necessary to warp the image. Measurements can be made in the original image, for example the position of a corner feature, and the measurement simply mapped according to (7.7). The question of where features *should* be measured does not have an unambiguous answer. Warping the image will distort noise models (because of averaging) and may well introduce aliasing effects. For this reason feature detection on the unwarped image will often be preferable. However, feature grouping, such as linking edgels into straight line primitives,

Samsung/Google, EX1010, p. 210

is best performed after warping since thresholds on linearity may well be erroneously exceeded in the original image.

7.5 Closure

7.5.1 The literature

The original application of the DLT in [Sutherland-63] was for camera computation. Estimation by iterative minimization of geometric errors is a standard procedure of photogrammetrists, e.g. see [Slama-80].

A minimal solution for a calibrated camera (pose from the image of 3 points) was the original problem studied by Fischler and Bolles [Fischler-81] in their RANSAC paper. Solutions to this problem reoccur *often* in the literature; a good treatment is given in [Wolfe-91] and also [Haralick-91]. Quasi-linear solutions for one more than the minimum number of point correspondences $\mathbf{X}_i \leftrightarrow \mathbf{x}_i$ are in [Quan-98] and [Triggs-99a].

Another class of methods, which are not covered here, is the iterative estimation of a projective camera starting from an affine one. The algorithm of "Model based pose in 25 lines of code" by Dementhon and Davis [Dementhon-95] is based on this idea. A similar method is used in [Christy-96].

Devernay and Faugeras [Devernay-95] introduced a straight line method for computing radial distortion into the computer vision literature. In the photogrammetry literature the method is known as "plumb line correction", see [Brown-71].

7.5.2 Notes and exercises

(i) Given 5 world-to-image point correspondences, $\mathbf{X}_i \leftrightarrow \mathbf{x}_i$, show that there are in general four solutions for a camera matrix $\mathtt{P}$ *with zero skew* that exactly maps the world to image points.

(ii) Given 3 world-to-image point correspondences, $\mathbf{X}_i \leftrightarrow \mathbf{x}_i$, show that there are in general four solutions for a camera matrix $\mathtt{P}$ *with known calibration* $\mathtt{K}$ that exactly maps the world to image points.

(iii) Find a linear algorithm for computing the camera matrix $\mathtt{P}$ under each of the following conditions:

- (a) The camera location (but not orientation) is known.
- (b) The direction of the principal ray of the camera is known.
- (c) The camera location and the principal ray of the camera are known.
- (d) The camera location and complete orientation of the camera are known.
- (e) The camera location and orientation are known, as well as some subset of the internal camera parameters (α_x, α_y, s, x_0 and y_0).

(iv) **Conflation of focal length and position on principal axis.** Compare the imaged position of a point of depth d before and after an increase in camera focal length Δf, or a displacement Δt_3 of the camera backwards along the principal axis. Let $(x, y)^\mathsf{T}$ and $(x', y')^\mathsf{T}$ be the image coordinates of the point before and

Samsung/Google, EX1010, p. 211

after the change. Following a similar derivation to that of (6.19–*p*169), show that

$$\begin{pmatrix} x' \\ y' \end{pmatrix} = \begin{pmatrix} x \\ y \end{pmatrix} + k \begin{pmatrix} x - x_0 \\ y - y_0 \end{pmatrix}$$

where $k^f = \Delta f / f$ for a focal length change, or $k^{t_3} = -\Delta t_3 / d$ for a displacement (here skew $s = 0$, and $\alpha_x = \alpha_y = f$).

For a set of calibration points $\mathbf{X}_i$ with depth relief (Δ_i) small compared to the average depth (d_0),

$$k_i^{t_3} = -\Delta t_3 / d_i = -\Delta t_3 / (d_0 + \Delta_i) \approx -\Delta t_3 / d_0$$

i.e. $k_i^{t_3}$ is approximately constant across the set. It follows that in calibrating from such a set, similar image residuals are obtained by changing the focal length k^f or displacing the camera k^{t_3}. Consequently, the estimated parameters of focal length and position on the principal axis are correlated.

(v) **Pushbroom camera computation.** The pushbroom camera, described in section 6.4.1, may also be computed using a DLT method. The x (orthographic) part of the projection matrix has 4 degrees of freedom which may be determined by four or more point correspondences $\mathbf{X}_i \leftrightarrow \mathbf{x}_i$; the y (perspective) part of the projection matrix has 7 degrees of freedom and may be determined from 7 correspondences. Hence, a minimal solution requires 7 points. Details are given in [Gupta-97].

Samsung/Google, EX1010, p. 212

8

More Single View Geometry

Chapter 6 introduced the projection matrix as the model for the action of a camera on points. This chapter describes the link between other 3D entities and their images under perspective projection. These entities include planes, lines, conics and quadrics; and we develop their forward and back-projection properties.

The camera is dissected further, and reduced to its centre point and image plane. Two properties are established: images acquired by cameras with the same centre are related by a plane projective transformation; and images of entities on the plane at infinity, π_∞, do not depend on camera position, only on camera rotation and internal parameters, $\mathtt{K}$.

The images of entities (points, lines, conics) on π_∞ are of particular importance. It will be seen that the image of a point on π_∞ is a vanishing point, and the image of a line on π_∞ a vanishing line; their images depend on both $\mathtt{K}$ and camera rotation. However, the image of the absolute conic, ω, depends only on $\mathtt{K}$; it is unaffected by the camera's rotation. The conic ω is intimately connected with camera calibration, $\mathtt{K}$, and the relation $\omega = (\mathtt{K}\mathtt{K}^\mathsf{T})^{-1}$ is established. It follows that ω defines the angle between rays back-projected from image points.

These properties enable camera relative rotation to be computed from vanishing points independently of camera position. Further, since $\mathtt{K}$ enables the angle between rays to be computed from image points, in turn $\mathtt{K}$ may be computed from the known angle between rays. In particular $\mathtt{K}$ may be determined from vanishing points corresponding to orthogonal scene directions. This means that a camera can be calibrated from scene features, without requiring known world coordinates.

A final geometric entity introduced in this chapter is the calibrating conic, which enables a geometric visualization of $\mathtt{K}$.

8.1 Action of a projective camera on planes, lines, and conics

In this section (and indeed in most of this book) it is only the 3×4 *form* and rank of the camera projection matrix $\mathtt{P}$ that is important in determining its action. The particular properties and relations of its elements are often not relevant.

Samsung/Google, EX1010, p. 213

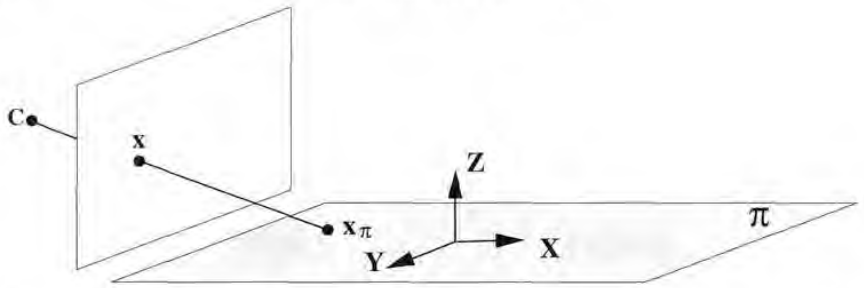


Fig. 8.1. **Perspective image of points on a plane.** *The XY-plane of the world coordinate frame is aligned with the plane π. Points on the image and scene planes are related by a plane projective transformation.*

8.1.1 On planes

The point imaging equation $\mathbf{x} = \mathtt{P}\mathbf{X}$ is a map from a point in a world coordinate frame, to a point in image coordinates. We have the freedom to choose the world coordinate frame. Suppose it is chosen such that the XY-plane corresponds to a plane π in the scene, so that points on the scene plane have zero Z-coordinate as shown in figure 8.1 (it is assumed that the camera centre does not lie on the scene plane). Then, if the columns of P are denoted as $\mathbf{p}_i$, the image of a point on π is given by

$$\mathbf{x} = \mathtt{P}\mathbf{X} = \begin{bmatrix} \mathbf{p}_1 & \mathbf{p}_2 & \mathbf{p}_3 & \mathbf{p}_4 \end{bmatrix} \begin{pmatrix} \mathtt{X} \\ \mathtt{Y} \\ 0 \\ 1 \end{pmatrix} = \begin{bmatrix} \mathbf{p}_1 & \mathbf{p}_2 & \mathbf{p}_4 \end{bmatrix} \begin{pmatrix} \mathtt{X} \\ \mathtt{Y} \\ 1 \end{pmatrix}.$$

So that the map between points $\mathbf{x}_\pi = (\mathtt{X}, \mathtt{Y}, 1)^\mathsf{T}$ on π and their image $\mathbf{x}$ is a general planar homography (a plane to plane projective transformation): $\mathbf{x} = \mathtt{H}\mathbf{x}_\pi$, with H a 3×3 matrix of rank 3. This shows that:

- *The most general transformation that can occur between a scene plane and an image plane under perspective imaging is a plane projective transformation.*

If the camera is affine, then a similar derivation shows that the scene and image planes are related by an affine transformation.

Example 8.1. For a calibrated camera (6.8–*p*156) $\mathtt{P} = \mathtt{K}[\mathtt{R} \mid \mathbf{t}]$, the homography between a world plane at $\mathtt{Z} = 0$ and the image is

$$\mathtt{H} = \mathtt{K}\left[\mathbf{r}_1, \mathbf{r}_2, \mathbf{t}\right] \tag{8.1}$$

where $\mathbf{r}_i$ are the columns of R. △

8.1.2 On lines

Forward projection. A line in 3-space projects to a line in the image. This is easily seen geometrically – the line and camera centre define a plane, and the image is the intersection of this plane with the image plane (figure 8.2) – and is proved algebraically

Samsung/Google, EX1010, p. 214

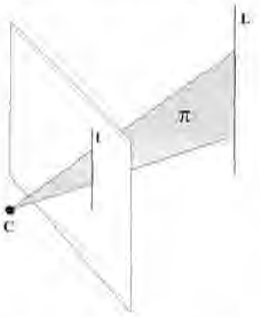


Fig. 8.2. **Line projection.** *A line* $\mathbf{L}$ *in 3-space is imaged as a line* $\mathbf{l}$ *by a perspective camera. The image line* $\mathbf{l}$ *is the intersection of the plane* $\boldsymbol{\pi}$*, defined by* $\mathbf{L}$ *and the camera centre* $\mathbf{C}$*, with the image plane. Conversely an image line* $\mathbf{l}$ *back-projects to a plane* $\boldsymbol{\pi}$ *in 3-space. The plane is the "pull-back" of the line.*

by noting that if $\mathbf{A}, \mathbf{B}$ are points in 3-space, and $\mathbf{a}, \mathbf{b}$ their images under $\mathtt{P}$, then a point $\mathbf{X}(\mu) = \mathbf{A} + \mu\mathbf{B}$ on a line which is the join of $\mathbf{A}, \mathbf{B}$ in 3-space projects to a point

$$\begin{aligned} \mathbf{x}(\mu) &= \mathtt{P}(\mathbf{A} + \mu\mathbf{B}) = \mathtt{P}\mathbf{A} + \mu\mathtt{P}\mathbf{B} \\ &= \mathbf{a} + \mu\mathbf{b} \end{aligned}$$

which is on the line joining $\mathbf{a}$ and $\mathbf{b}$.

Back-projection of lines. The set of points in space which map to a line in the image is a plane in space defined by the camera centre and image line, as shown in figure 8.2. Algebraically,

Result 8.2. *The set of points in space mapping to a line* $\mathbf{l}$ *via the camera matrix* $\mathtt{P}$ *is the plane* $\mathtt{P}^\mathsf{T}\mathbf{l}$.

Proof. A point $\mathbf{x}$ lies on $\mathbf{l}$ if and only if $\mathbf{x}^\mathsf{T}\mathbf{l} = 0$. A space point $\mathbf{X}$ maps to a point $\mathtt{P}\mathbf{X}$, which lies on the line if and only if $\mathbf{X}^\mathsf{T}\mathtt{P}^\mathsf{T}\mathbf{l} = 0$. Thus, if $\mathtt{P}^\mathsf{T}\mathbf{l}$ is taken to represent a plane, then $\mathbf{X}$ lies on this plane if and only if $\mathbf{X}$ maps to a point on the line $\mathbf{l}$. In other words, $\mathtt{P}^\mathsf{T}\mathbf{l}$ is the back-projection of the line $\mathbf{l}$. □

Geometrically there is a *star* (two-parameter family) of planes through the camera centre, and the three rows of the projection matrix $\mathtt{P}^{i\mathsf{T}}$ (6.12–*p*159) are a basis for this star. The plane $\mathtt{P}^\mathsf{T}\mathbf{l}$ is a linear combination of this basis corresponding to the element of the star containing the camera centre and the line $\mathbf{l}$. For example, if $\mathbf{l} = (0, 1, 0)^\mathsf{T}$ then the plane is $\mathbf{P}^2$, and is the back projection of the image x-axis.

Plücker line representation. *Understanding this material on Plücker line mapping is not required for following the rest of the book.*

We now turn to forward projection of lines. If a line in 3-space is represented by Plücker coordinates then its image can be expressed as a linear map on these coordinates. We will develop this map for both the 4×4 matrix and 6-vector line representations.

Samsung/Google, EX1010, p. 215

Result 8.3. *Under the camera mapping* P, *a line in 3-space represented as a Plücker matrix* L, *as defined in (3.8–p70), is imaged as the line* l *where*

$$[\mathbf{l}]_\times = \mathtt{PLP}^\mathsf{T}. \tag{8.2}$$

where the notation $[\mathbf{l}]_\times$ *is defined in (A4.5–p581).*

Proof. Suppose as above that $\mathbf{a} = \mathtt{PA}$, $\mathbf{b} = \mathtt{PB}$. The Plücker matrix L for the line through A, B in 3-space is $\mathtt{L} = \mathbf{AB}^\mathsf{T} - \mathbf{BA}^\mathsf{T}$. Then the matrix $\mathtt{M} = \mathtt{PLP}^\mathsf{T} = \mathbf{ab}^\mathsf{T} - \mathbf{ba}^\mathsf{T}$ is 3×3 and antisymmetric, with null-space $\mathbf{a} \times \mathbf{b}$. Consequently, $\mathtt{M} = [\mathbf{a} \times \mathbf{b}]_\times$, and since the line through the image points is given by $\mathbf{l} = \mathbf{a} \times \mathbf{b}$, this completes the proof. □

It is clear from the form of (8.2) that there is a linear relation between the image line coordinates l_i and the world line coordinates L_{jk}, but that this relation is quadratic in the elements of the point projection matrix P. Thus, (8.2) may be rearranged such that the map between the Plücker line coordinates, $\mathcal{L}$ (a 6-vector), and the image line coordinates l (a 3-vector) is represented by a single 3×6 matrix. It can be shown that

Definition 8.4. The *line projection matrix* $\mathcal{P}$ is the 3×6 matrix of rank 3 given by

$$\mathcal{P} = \begin{bmatrix} \mathbf{P}^2 \wedge \mathbf{P}^3 \\ \mathbf{P}^3 \wedge \mathbf{P}^1 \\ \mathbf{P}^1 \wedge \mathbf{P}^2 \end{bmatrix} \tag{8.3}$$

where $\mathbf{P}^{i\mathsf{T}}$ are the rows of the point camera matrix P, and $\mathbf{P}^i \wedge \mathbf{P}^j$ are the Plücker line coordinates of the intersection of the planes $\mathbf{P}^i$ and $\mathbf{P}^j$.

Then the forward line projection is given by

Result 8.5. *Under the line projection matrix* $\mathcal{P}$, *a line in* $\mathbb{P}^3$ *represented by Plücker line coordinates* $\mathcal{L}$, *as defined in (3.11–p72), is mapped to the image line*

$$\mathbf{l} = \mathcal{P}\mathcal{L} = \begin{bmatrix} (\mathbf{P}^2 \wedge \mathbf{P}^3 | \mathcal{L}) \\ (\mathbf{P}^3 \wedge \mathbf{P}^1 | \mathcal{L}) \\ (\mathbf{P}^1 \wedge \mathbf{P}^2 | \mathcal{L}) \end{bmatrix} \tag{8.4}$$

where the product $(\mathcal{L}|\hat{\mathcal{L}})$ *is defined in (3.13–p72).*

Proof. Suppose the line in 3-space is the join of the points A and B, and these project to $\mathbf{a} = \mathtt{PA}$, $\mathbf{b} = \mathtt{PB}$ respectively. Then the image line $\mathbf{l} = \mathbf{a} \times \mathbf{b} = (\mathtt{PA}) \times (\mathtt{PB})$. Consider the first element

$$\begin{aligned} l_1 &= (\mathbf{P}^{2\mathsf{T}}\mathbf{A})(\mathbf{P}^{3\mathsf{T}}\mathbf{B}) - (\mathbf{P}^{2\mathsf{T}}\mathbf{B})(\mathbf{P}^{3\mathsf{T}}\mathbf{A}) \\ &= (\mathbf{P}^2 \wedge \mathbf{P}^3 | \mathcal{L}) \end{aligned}$$

where the second equality follows from (3.14–*p*73). The other components follow in a similar manner. □

Samsung/Google, EX1010, p. 216

The line projection matrix $\mathcal{P}$ plays the same role for lines as $\mathtt{P}$ does for points. The rows of $\mathcal{P}$ may be interpreted geometrically as *lines*, in a similar manner to the interpretation of the rows of the point camera matrix $\mathtt{P}$ as *planes* given in section 6.2.1(*p*158). The rows $\mathbf{P}^{i\mathsf{T}}$ of $\mathtt{P}$ are the principal plane and axis planes of the camera. The rows of $\mathcal{P}$ are the lines of intersection of pairs of these camera planes. For example, the first row of $\mathcal{P}$ is $\mathbf{P}^2 \wedge \mathbf{P}^3$, and this is the 6-vector Plücker line representation of the line of intersection of the $y = 0$ axis plane, $\mathbf{P}^2$, and the principal plane, $\mathbf{P}^3$. The three lines corresponding to the three rows of $\mathcal{P}$ intersect at the camera centre. Consider lines $\mathcal{L}$ in 3-space for which $\mathcal{P}\mathcal{L} = \mathbf{0}$. These lines are in the null-space of $\mathcal{P}$. Since each row of $\mathcal{P}$ is a line, and from result 3.5(*p*72) the product $(\mathcal{L}_1|\mathcal{L}_2) = 0$ if two lines intersect, if follows that $\mathcal{L}$ intersects each of the lines represented by the rows of $\mathcal{P}$. These lines are the intersection of the camera planes, and the only point on all 3 camera planes is the camera centre. Thus we have

- *The lines $\mathcal{L}$ in $\mathbb{P}^3$ for which $\mathcal{P}\mathcal{L} = \mathbf{0}$ pass through the camera centre.*

The 3×6 matrix $\mathcal{P}$ has a 3-dimensional null-space. Allowing for the homogeneous scale factor, this null-space is a two-parameter family of lines containing the camera centre. This is to be expected since there is a star (two parameter family) of lines in $\mathbb{P}^3$ concurrent with a point.

8.1.3 On conics

Back-projection of conics. A conic $\mathtt{C}$ back-projects to a cone. A cone is a degenerate quadric, i.e. the 4×4 matrix representing the quadric does not have full rank. The cone vertex, in this case the camera centre, is the null-vector of the quadric matrix.

Result 8.6. *Under the camera $\mathtt{P}$ the conic $\mathtt{C}$ back-projects to the cone*

$$\mathtt{Q}_{\mathrm{co}} = \mathtt{P}^{\mathsf{T}}\mathtt{C}\mathtt{P}.$$

Proof. A point $\mathbf{x}$ lies on $\mathtt{C}$ if and only if $\mathbf{x}^{\mathsf{T}}\mathtt{C}\mathbf{x} = 0$. A space point $\mathbf{X}$ maps to a point $\mathtt{P}\mathbf{X}$, which lies on the conic if and only if $\mathbf{X}^{\mathsf{T}}\mathtt{P}^{\mathsf{T}}\mathtt{C}\mathtt{P}\mathbf{X} = 0$. Thus, if $\mathtt{Q}_{\mathrm{co}} = \mathtt{P}^{\mathsf{T}}\mathtt{C}\mathtt{P}$ is taken to represent a quadric, then $\mathbf{X}$ lies on this quadric if and only if $\mathbf{X}$ maps to a point on the conic $\mathtt{C}$. In other words, $\mathtt{Q}_{\mathrm{co}}$ is the back-projection of the conic $\mathtt{C}$. □

Note the camera centre $\mathbf{C}$ is the vertex of the degenerate quadric since $\mathtt{Q}_{\mathrm{co}}\mathbf{C} = \mathtt{P}^{\mathsf{T}}\mathtt{C}(\mathtt{P}\mathbf{C}) = \mathbf{0}$.

Example 8.7. Suppose that $\mathtt{P} = \mathtt{K}[\mathtt{I} \mid \mathbf{0}]$; then the conic $\mathtt{C}$ back-projects to the cone

$$\mathtt{Q}_{\mathrm{co}} = \begin{bmatrix} \mathtt{K}^{\mathsf{T}} \\ \mathbf{0}^{\mathsf{T}} \end{bmatrix} \mathtt{C} \begin{bmatrix} \mathtt{K} \mid \mathbf{0} \end{bmatrix} = \begin{bmatrix} \mathtt{K}^{\mathsf{T}}\mathtt{C}\mathtt{K} & \mathbf{0} \\ \mathbf{0}^{\mathsf{T}} & 0 \end{bmatrix}.$$

The matrix $\mathtt{Q}_{\mathrm{co}}$ has rank 3. Its null-vector is the camera centre $\mathbf{C} = (0, 0, 0, 1)^{\mathsf{T}}$. △

Samsung/Google, EX1010, p. 217

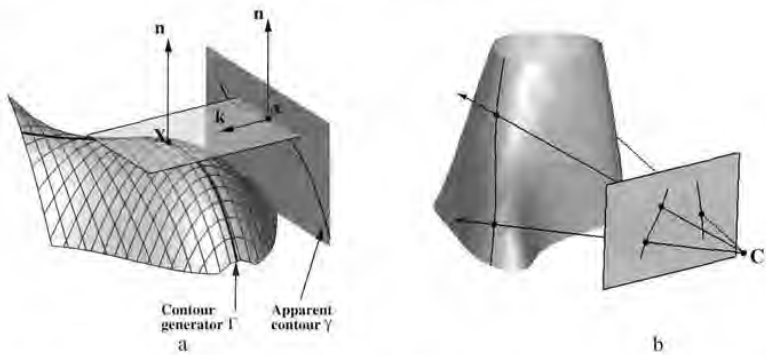


Fig. 8.3. **Contour generator and apparent contour.** *(a) for parallel projection; (b) for central projection. The ray from the camera centre through* $\mathbf{x}$ *is tangent to the surface at* $\mathbf{X}$*. The set of such tangent points* $\mathbf{X}$ *defines the contour generator, and their image defines the apparent contour. In general the contour generator is a space curve. Figure courtesy of Roberto Cipolla and Peter Giblin.*

8.2 Images of smooth surfaces

The image outline of a smooth surface S results from surface points at which the imaging rays are *tangent* to the surface, as shown in figure 8.3. Similarly, *lines* tangent to the outline back-project to planes which are *tangent planes* to the surface.

Definition 8.8. The *contour generator* Γ is the set of points $\mathbf{X}$ on S at which rays are tangent to the surface. The corresponding image *apparent contour* γ is the set of points $\mathbf{x}$ which are the image of $\mathbf{X}$, i.e. γ is the image of Γ.

The apparent contour is also called the "outline" and "profile". If the surface is viewed in the direction of $\mathbf{X}$ from the camera centre, then the surface appears to fold, or to have a boundary or occluding contour.

It is evident that the contour generator Γ depends only on the relative position of the camera centre and surface, not on the image plane. However, the apparent contour γ is defined by the intersection of the image plane with the rays to the contour generator, and so does depend on the position of the image plane.

In the case of parallel projection with direction $\mathbf{k}$, consider all the rays parallel to $\mathbf{k}$ which are tangent to S, see figure 8.3a. These rays form a "cylinder" of tangent rays, and the curve along which this cylinder is tangent to S is the contour generator Γ. The curve in which the cylinder meets the image plane is the apparent contour γ. Note that both Γ and γ depend in an essential way on $\mathbf{k}$. The set Γ slips over the surface as the direction of $\mathbf{k}$ changes. For example, with S a sphere, Γ is the great circle orthogonal to $\mathbf{k}$. In this case, the contour generator Γ is a plane curve, but in general Γ is a space curve.

We next describe the projection properties of quadrics. For this class of surface algebraic expressions can be developed for the contour generator and apparent contour.

Samsung/Google, EX1010, p. 218



Fig. 8.4. **The cone of rays for a quadric.** *The vertex of the cone is the camera centre. (a) The contour generator* Γ *of a quadric is a plane curve (a conic) which is the intersection of the quadric with the polar plane of the camera centre,* $\mathbf{C}$.

8.3 Action of a projective camera on quadrics

A quadric is a smooth surface and so its outline curve is given by points where the back-projected rays are tangent to the quadric surface as shown in figure 8.4.

Suppose the quadric is a sphere, then the cone of rays between the camera centre and quadric is right-circular, i.e. the contour generator is a circle, with the plane of the circle orthogonal to the line joining the camera and sphere centres. This can be seen from the rotational symmetry of the geometry about this line. The image of the sphere is obtained by intersecting the cone with the image plane. It is clear that this is a classical conic section, so that the apparent contour of a sphere is a conic. In particular if the sphere centre lies on the principal (Z) camera axis, then the conic is a circle.

Now consider a 3-space projective transformation of this geometry. Under this map the sphere is transformed to a quadric and the apparent contour to a conic. However, since intersection and tangency are preserved, the contour generator is a (plane) conic. Consequently, the apparent contour of a general quadric is a conic, and the contour generator is also a conic. We will now give algebraic representations for these geometric results.

Forward projection of quadrics. Since the outline arises from tangency, it is not surprising that the dual of the quadric, $\mathtt{Q}^*$, is important here since it defines the tangent planes to the quadric $\mathtt{Q}$.

Result 8.9. *Under the camera matrix* $\mathtt{P}$ *the outline of the quadric* $\mathtt{Q}$ *is the conic* $\mathtt{C}$ *given by*

$$\mathtt{C}^* = \mathtt{P}\mathtt{Q}^*\mathtt{P}^\mathsf{T}. \tag{8.5}$$

Proof. This expression is simply derived from the observation that lines $\mathbf{l}$ tangent to the conic outline satisfy $\mathbf{l}^\mathsf{T}\mathtt{C}^*\mathbf{l} = 0$. These lines back-project to planes $\boldsymbol{\pi} = \mathtt{P}^\mathsf{T}\mathbf{l}$ that are tangent to the quadric and thus satisfy $\boldsymbol{\pi}^\mathsf{T}\mathtt{Q}^*\boldsymbol{\pi} = 0$. Then it follows that for each line

$$\begin{aligned}\boldsymbol{\pi}^\mathsf{T}\mathtt{Q}^*\boldsymbol{\pi} &= \mathbf{l}^\mathsf{T}\mathtt{P}\mathtt{Q}^*\mathtt{P}^\mathsf{T}\mathbf{l} \\ &= \mathbf{l}^\mathsf{T}\mathtt{C}^*\mathbf{l} = 0\end{aligned}$$

Samsung/Google, EX1010, p. 219

and since this is true for all lines tangent to C the result follows. □

Note the similarity of (8.5) with the projection of a line represented by a Plücker matrix (8.2). An expression for the projection of the point quadric Q can be derived from (8.5) but it is quite complicated. However, the plane of the contour generator is easily expressed in terms of Q:

- *The plane of* Γ *for a quadric* Q *and camera with centre* C *is given by* $\pi_\Gamma = \mathtt{Q}\mathbf{C}$.

This result follows directly from the pole–polar relation for a point and quadric of section 3.2.3(p73). Its proof is left as an exercise. Note, the intersection of a quadric and plane is a conic. So Γ is a conic and its image γ, which is the apparent contour, is also a conic as has been seen above.

We may also derive an expression for the cone of rays formed by the camera centre and quadric. This cone is a degenerate quadric of rank 3.

Result 8.10. *The cone with vertex* V *and tangent to the quadric* Q *is the degenerate quadric*

$$\mathtt{Q}_{\mathrm{co}} = (\mathbf{V}^\top \mathtt{Q}\mathbf{V})\mathtt{Q} - (\mathtt{Q}\mathbf{V})(\mathtt{Q}\mathbf{V})^\top.$$

Note that $\mathtt{Q}_{\mathrm{co}}\mathbf{V} = \mathbf{0}$, so that V is the vertex of the cone as required. The proof is omitted.

Example 8.11. We write the quadric in block form:

$$\mathtt{Q} = \begin{bmatrix} \mathtt{Q}_{3\times 3} & \mathbf{q} \\ \mathbf{q}^\top & q_{44} \end{bmatrix}.$$

Then if $\mathbf{V} = (0, 0, 0, 1)^\top$, which corresponds to the cone vertex being at the centre of the world coordinate frame,

$$\mathtt{Q}_{\mathrm{co}} = \begin{bmatrix} q_{44}\mathtt{Q}_{3\times 3} - \mathbf{q}\mathbf{q}^\top & \mathbf{0} \\ \mathbf{0}^\top & 0 \end{bmatrix}$$

which is clearly a degenerate quadric. △

8.4 The importance of the camera centre

An object in 3-space and camera centre define a set of rays, and an image is obtained by intersecting these rays with a plane. Often this set is referred to as a *cone* of rays, even though it is not a classical cone. Suppose the cone of rays is intersected by two planes, as shown in figure 8.5, then the two images, I and I', are clearly related by a perspective map. This means that images obtained with the same camera centre may be mapped to one another by a plane projective transformation, in other words they are projectively equivalent and so have the same projective properties. A camera can thus be thought of as a projective imaging device – measuring projective properties of the cone of rays with vertex the camera centre.

The result that the two images I and I' are related by a homography will now be derived algebraically to obtain a formula for this homography. Consider two cameras

$$\mathtt{P} = \mathtt{K}\mathtt{R}[\mathtt{I} \mid -\widetilde{\mathbf{C}}], \quad \mathtt{P}' = \mathtt{K}'\mathtt{R}'[\mathtt{I} \mid -\widetilde{\mathbf{C}}]$$

Samsung/Google, EX1010, p. 220

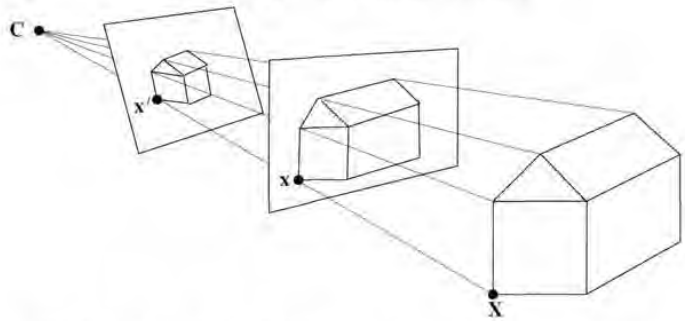


Fig. 8.5. **The cone of rays with vertex the camera centre.** *An image is obtained by intersecting this cone with a plane. A ray between a 3-space point* $\mathbf{X}$ *and the camera centre* $\mathbf{C}$ *pierces the planes in the image points* $\mathbf{x}$ *and* $\mathbf{x}'$*. All such image points are related by a planar homography,* $\mathbf{x}' = \mathtt{H}\mathbf{x}$.

with the same centre. Note that since the cameras have a common centre there is a simple relation between them, namely $\mathtt{P}' = (\mathtt{K}'\mathtt{R}')(\mathtt{K}\mathtt{R})^{-1}\mathtt{P}$. It then follows that the images of a 3-space point $\mathbf{X}$ by the two cameras are related as

$$\mathbf{x}' = \mathtt{P}'\mathbf{X} = (\mathtt{K}'\mathtt{R}')(\mathtt{K}\mathtt{R})^{-1}\mathtt{P}\mathbf{X} = (\mathtt{K}'\mathtt{R}')(\mathtt{K}\mathtt{R})^{-1}\mathbf{x}.$$

That is, the corresponding image points are related by a planar homography (a 3×3 matrix) as $\mathbf{x}' = \mathtt{H}\mathbf{x}$, where $\mathtt{H} = (\mathtt{K}'\mathtt{R}')(\mathtt{K}\mathtt{R})^{-1}$.

We will now investigate several cases of moving the image plane whilst fixing the camera centre. For simplicity the world coordinate frame will be chosen to coincide with the camera's, so that $\mathtt{P} = \mathtt{K}[\mathtt{I} \mid \mathbf{0}]$ (and it will be assumed that the image plane never contains the centre, as the image would then be degenerate).

8.4.1 Moving the image plane

Consider first an increase in focal length. To a first approximation this corresponds to a displacement of the image plane along the principal axis. The image effect is a simple magnification. This is only a first approximation because with a compound lens zooming will perturb both the principal point and the effective camera centre. Algebraically, if $\mathbf{x}$, $\mathbf{x}'$ are the images of a point $\mathbf{X}$ before and after zooming, then

$$\begin{aligned} \mathbf{x} &= \mathtt{K}[\mathtt{I} \mid \mathbf{0}]\mathbf{X} \\ \mathbf{x}' &= \mathtt{K}'[\mathtt{I} \mid \mathbf{0}]\mathbf{X} = \mathtt{K}'\mathtt{K}^{-1}(\mathtt{K}[\mathtt{I} \mid \mathbf{0}]\mathbf{X}) = \mathtt{K}'\mathtt{K}^{-1}\mathbf{x} \end{aligned}$$

so that $\mathbf{x}' = \mathtt{H}\mathbf{x}$ with $\mathtt{H} = \mathtt{K}'\mathtt{K}^{-1}$. If only the focal lengths differ between $\mathtt{K}$ and $\mathtt{K}'$ then a short calculation shows that

$$\mathtt{K}'\mathtt{K}^{-1} = \begin{bmatrix} k\mathtt{I} & (1-k)\tilde{\mathbf{x}}_0 \\ \mathbf{0}^\top & 1 \end{bmatrix}.$$

where $\tilde{\mathbf{x}}_0$ is the inhomogeneous principal point, and $k = f'/f$ is the magnification factor. This result follows directly from similar triangles: the effect of zooming by a

Samsung/Google, EX1010, p. 221



Fig. 8.6. *Between images (a) and (b) the camera rotates about the camera centre. Corresponding points (that is images of the same 3D point) are related by a plane projective transformation. Note that 3D points at different depths which are coincident in image (a), such as the mug lip and cat body, are also coincident in (b), so there is no motion parallax in this case. However, between images (a) and (c) the camera rotates about the camera centre* and *translates. Under this general motion coincident points of differing depth in (a) are imaged at different points in (c), so there is motion parallax in this case due to the camera translation.*

factor k is to move the image point $\tilde{\mathbf{x}}$ on a line radiating from the principal point $\tilde{\mathbf{x}}_0$ to the point $\tilde{\mathbf{x}}' = k\tilde{\mathbf{x}} + (1-k)\tilde{\mathbf{x}}_0$. Algebraically, using the most general form (6.10–p157) of the calibration matrix $\mathtt{K}$, we may write

$$\begin{aligned}\mathtt{K}' &= \begin{bmatrix} k\mathtt{I} & (1-k)\tilde{\mathbf{x}}_0 \\ \mathbf{0}^\top & 1 \end{bmatrix} \mathtt{K} = \begin{bmatrix} k\mathtt{I} & (1-k)\tilde{\mathbf{x}}_0 \\ \mathbf{0}^\top & 1 \end{bmatrix} \begin{bmatrix} \mathtt{A} & \tilde{\mathbf{x}}_0 \\ \mathbf{0}^\top & 1 \end{bmatrix} \\ &= \begin{bmatrix} k\mathtt{A} & \tilde{\mathbf{x}}_0 \\ \mathbf{0}^\top & 1 \end{bmatrix} = \mathtt{K} \begin{bmatrix} k\mathtt{I} & \\ & 1 \end{bmatrix}.\end{aligned}$$

This shows that

- *The effect of zooming by a factor k is to multiply the calibration matrix $\mathtt{K}$ on the right by* $\mathrm{diag}(k, k, 1)$.

8.4.2 Camera rotation

A second common example is where the camera is rotated about its centre with no change in the internal parameters. Examples of this "pure" rotation are given in figure 8.6 and figure 8.9. Algebraically, if $\mathbf{x}$, $\mathbf{x}'$ are the images of a point $\mathbf{X}$ before and after the pure rotation

$$\begin{aligned}\mathbf{x} &= \mathtt{K}[\mathtt{I} \mid \mathbf{0}]\mathbf{X} \\ \mathbf{x}' &= \mathtt{K}[\mathtt{R} \mid \mathbf{0}]\mathbf{X} = \mathtt{K}\mathtt{R}\mathtt{K}^{-1}\mathtt{K}[\mathtt{I} \mid \mathbf{0}]\mathbf{X} = \mathtt{K}\mathtt{R}\mathtt{K}^{-1}\mathbf{x}\end{aligned}$$

so that $\mathbf{x}' = \mathtt{H}\mathbf{x}$ with $\mathtt{H} = \mathtt{K}\mathtt{R}\mathtt{K}^{-1}$. This homography is a *conjugate rotation* and is discussed further in section A7.1(p628). For now, we mention a few of its properties by way of an example.

Example 8.12. Properties of a conjugate rotation
The homography $\mathtt{H} = \mathtt{K}\mathtt{R}\mathtt{K}^{-1}$ has the same eigenvalues (up to scale) as the rotation matrix, namely $\{\mu, \mu e^{i\theta}, \mu e^{-i\theta}\}$, where μ is an unknown scale factor (if $\mathtt{H}$ is scaled such that $\det \mathtt{H} = 1$, then $\mu = 1$). Consequently the angle of rotation between views may be computed directly from the phase of the complex eigenvalues of $\mathtt{H}$. Similarly, it can be

Samsung/Google, EX1010, p. 222

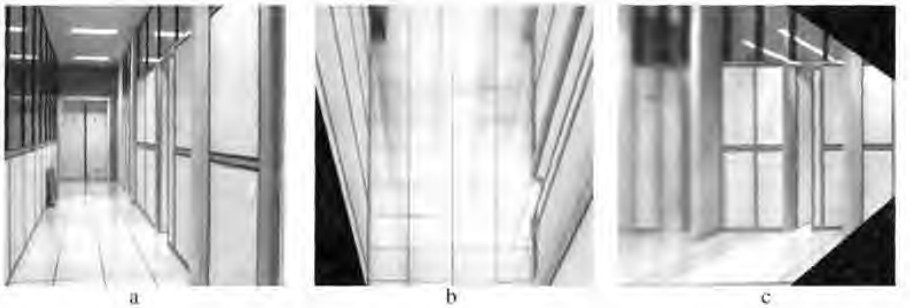


Fig. 8.7. **Synthetic views.** *(a) Source image. (b) Fronto-parallel view of the corridor floor generated from (a) using the four corners of a floor tile to compute the homography. (c) Fronto-parallel view of the corridor wall generated from (a) using the four corners of the door frame to compute the homography.*

shown (see exercises) that the eigenvector of $\mathtt{H}$ corresponding to the real eigenvalue is the vanishing point of the rotation axis.

For example, between images (a) and (b) of figure 8.6 there is a pure rotation of the camera. The homography $\mathtt{H}$ is computed by algorithm 4.6(p123), and from this the angle of rotation is estimated as 4.66°, and the axis vanishing point as $(-0.0088, 1, 0.0001)^\mathsf{T}$, i.e. virtually at infinity in the y direction, so the rotation axis is almost parallel to the y-axis. △

The transformation $\mathtt{H} = \mathtt{K}\mathtt{R}\mathtt{K}^{-1}$ is an example of the *infinite homography* mapping $\mathtt{H}_\infty$, that will appear many times through this book. It is defined in section 13.4(p338). The conjugation property is used for auto-calibration in chapter 19.

8.4.3 Applications and examples

The homographic relation between images with the same camera centre can be exploited in several ways. One is the creation of synthetic images by projective warping. Another is mosaicing, where panoramic images can be created by using planar homographies to "sew" together views obtained by a rotating camera.

Example 8.13. Synthetic views

New images corresponding to different camera orientations (with the same camera centre) can be generated from an existing image by warping with planar homographies.

In a fronto-parallel view a rectangle is imaged as a rectangle, and the world and image rectangle have the same aspect ratio. Conversely, a fronto-parallel view can be synthesized by warping an image with the homography that maps a rectangle imaged as a quadrilateral to a rectangle with the correct aspect ratio. The algorithm is:

(i) Compute the homography $\mathtt{H}$ which maps the image quadrilateral to a rectangle with the correct aspect ratio.
(ii) Projectively warp the source image with this homography.

Examples are shown in figure 8.7. △

Samsung/Google, EX1010, p. 223

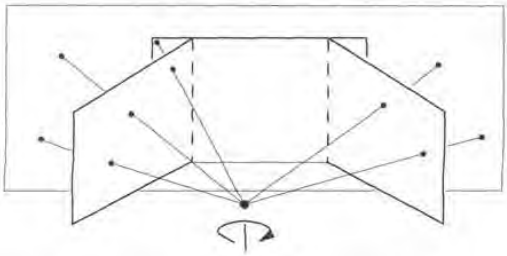


Fig. 8.8. *Three images acquired by a rotating camera may be registered to the frame of the middle one, as shown, by projectively warping the outer images to align with the middle one.*

Example 8.14. Planar panoramic mosaicing

Images acquired by a camera rotating about its centre are related to each other by a planar homography. A set of such images may be registered with the plane of one of the images by projectively warping the other images, as illustrated in figure 8.8.



Fig. 8.9. **Planar panoramic mosaicing.** *Eight images (out of thirty) acquired by rotating a camcorder about its centre. The thirty images are registered (automatically) using planar homographies and composed into the single panoramic mosaic shown. Note the characteristic "bow tie" shape resulting from registering to an image at the middle of the sequence.*

In outline the algorithm is:

(i) Choose one image of the set as a reference.
(ii) Compute the homography H which maps one of the other images of the set to this reference image.

Samsung/Google, EX1010, p. 224

(iii) Projectively warp the image with this homography, and augment the reference image with the non-overlapping part of the warped image.
(iv) Repeat the last two steps for the remaining images of the set.

The homographies may be computed by identifying (at least) four corresponding points, or by using the automatic method of algorithm 4.6(p123). An example mosaic is shown in figure 8.9. △

8.4.4 Projective (reduced) notation

It will be seen in chapter 20 that if canonical projective coordinates are chosen for world and image points, i.e.

$$\mathbf{X}_1 = (1,0,0,0)^\mathsf{T},\ \mathbf{X}_2 = (0,1,0,0)^\mathsf{T},\ \mathbf{X}_3 = (0,0,1,0)^\mathsf{T},\ \mathbf{X}_4 = (0,0,0,1)^\mathsf{T},$$

and

$$\mathbf{x}_1 = (1,0,0)^\mathsf{T},\ \mathbf{x}_2 = (0,1,0)^\mathsf{T},\ \mathbf{x}_3 = (0,0,1)^\mathsf{T},\ \mathbf{x}_4 = (1,1,1)^\mathsf{T},$$

then the camera matrix

$$\mathtt{P} = \begin{bmatrix} a & 0 & 0 & -d \\ 0 & b & 0 & -d \\ 0 & 0 & c & -d \end{bmatrix} \tag{8.6}$$

satisfies $\mathbf{x}_i = \mathtt{P}\mathbf{X}_i$, $i = 1,\ldots,4$, and also that $\mathtt{P}(a^{-1}, b^{-1}, c^{-1}, d^{-1})^\mathsf{T} = \mathbf{0}$, which means that the camera centre is $\mathbf{C} = (a^{-1}, b^{-1}, c^{-1}, d^{-1})^\mathsf{T}$. This is known as the *reduced* camera matrix, and it is clearly completely specified by the 3 degrees of freedom of the camera centre $\mathbf{C}$. This is a further illustration of the fact that all images acquired by cameras with the same camera centre are projectively equivalent – the camera has been reduced to its essence: a projective device whose action is to map $\mathbb{P}^3$ to $\mathbb{P}^2$ with only the position of the camera centre affecting the result. This camera representation is used in establishing duality relations in chapter 20.

8.4.5 Moving the camera centre

The cases of zooming and camera rotation illustrate that moving the image plane, whilst fixing the camera centre, induces a transformation between images that depends only on the image plane motion, but *not* on the 3-space structure. Conversely, no information on 3-space structure can be obtained by this action. However, if the camera centre is moved then the map between corresponding image points *does* depend on the 3-space structure, and indeed may often be used to (partially) determine the structure. This is the subject of much of the remainder of this book.

How can one determine from the images alone whether the camera centre has moved? Consider two 3-space points which have coincident images in the first view, i.e. the points are on the same ray. If the camera centre is moved (not along that ray) the image coincidence is lost. This relative displacement of previously coincident image points is termed *parallax*, and is illustrated in figure 8.6 and shown schematically in figure 8.10. If the scene is static and motion parallax is evident between two views then the camera centre has moved. Indeed, a convenient method for obtaining a camera

Samsung/Google, EX1010, p. 225

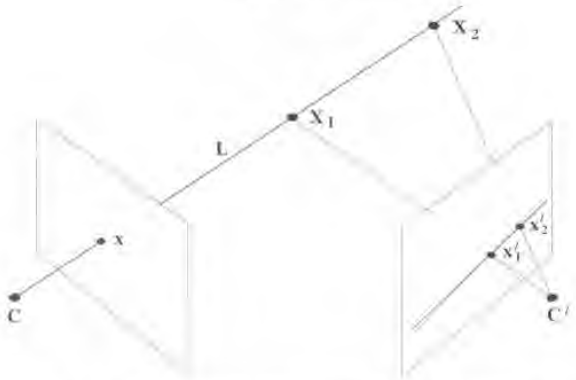


Fig. 8.10. **Motion parallax.** *The images of the space points* $\mathbf{X}_1$ *and* $\mathbf{X}_2$ *are coincident when viewed by the camera with centre* $\mathbf{C}$. *However, when viewed by a camera with centre* $\mathbf{C}'$, *which does not lie on the line* $\mathbf{L}$ *through* $\mathbf{X}_1$ *and* $\mathbf{X}_2$, *the images of the space points are not coincident. In fact the line through the image points* $\mathbf{x}'_1$ *and* $\mathbf{x}'_2$ *is the image of the ray* $\mathbf{L}$, *and will be seen in chapter 9 to be an* **epipolar line**. *The vector between the points* $\mathbf{x}'_1$ *and* $\mathbf{x}'_2$ *is the parallax.*

motion that is only a rotation about its centre (for example for a camera mounted on a robot head) is to adjust the motion until there is no parallax.

An important special case of 3-space structure is when all scene points are coplanar. In this case the images of corresponding points are related by a planar homography even if the camera centre is moved. The map between images in this case is discussed in detail in chapter 13 on planes. In particular vanishing points, which are images of points on the plane π_∞, are related by a planar homography for any camera motion. We return to this in section 8.6.

8.5 Camera calibration and the image of the absolute conic

Up to this point we have discussed projective properties of the forward and back-projection of various entities (point, lines, conics ...). These properties depend only on the 3×4 form of the projective camera matrix $\mathtt{P}$. Now we describe what is gained if the camera internal calibration, $\mathtt{K}$, is known. It will be seen that Euclidean properties, such as the angle between two rays, can then be measured.

What does calibration give? An image point $\mathbf{x}$ back-projects to a ray defined by $\mathbf{x}$ and the camera centre. Calibration relates the image point to the ray's *direction*. Suppose points on the ray are written as $\widetilde{\mathbf{X}} = \lambda \mathbf{d}$ in the camera Euclidean coordinate frame, then these points map to the point $\mathbf{x} = \mathtt{K}[\mathtt{I} \mid \mathbf{0}](\lambda \mathbf{d}^\top, 1)^\top = \mathtt{K}\mathbf{d}$ up to scale. Conversely the direction $\mathbf{d}$ is obtained from the image point $\mathbf{x}$ as $\mathbf{d} = \mathtt{K}^{-1}\mathbf{x}$. Thus we have established:

Result 8.15. *The camera calibration matrix* $\mathtt{K}$ *is the (affine) transformation between* $\mathbf{x}$ *and the ray's direction* $\mathbf{d} = \mathtt{K}^{-1}\mathbf{x}$ *measured in the camera's Euclidean coordinate frame.*

Note, $\mathbf{d} = \mathtt{K}^{-1}\mathbf{x}$ is in general *not* a unit vector.

Samsung/Google, EX1010, p. 226

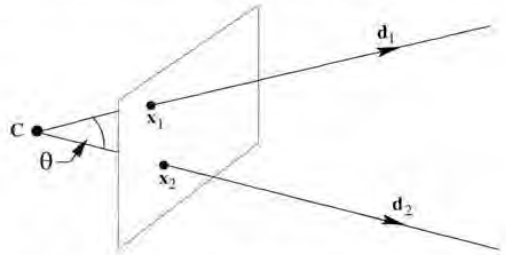


Fig. 8.11. *The angle θ between two rays.*

The angle between two rays, with directions $\mathbf{d}_1$, $\mathbf{d}_2$ corresponding to image points $\mathbf{x}_1, \mathbf{x}_2$ respectively, may be obtained from the familiar cosine formula for the angle between two vectors:

$$\begin{aligned}\cos\theta &= \frac{\mathbf{d}_1^\top \mathbf{d}_2}{\sqrt{\mathbf{d}_1^\top \mathbf{d}_1}\sqrt{\mathbf{d}_2^\top \mathbf{d}_2}} = \frac{(\mathtt{K}^{-1}\mathbf{x}_1)^\top(\mathtt{K}^{-1}\mathbf{x}_2)}{\sqrt{(\mathtt{K}^{-1}\mathbf{x}_1)^\top(\mathtt{K}^{-1}\mathbf{x}_1)}\sqrt{(\mathtt{K}^{-1}\mathbf{x}_2)^\top(\mathtt{K}^{-1}\mathbf{x}_2)}} \\ &= \frac{\mathbf{x}_1^\top(\mathtt{K}^{-\top}\mathtt{K}^{-1})\mathbf{x}_2}{\sqrt{\mathbf{x}_1^\top(\mathtt{K}^{-\top}\mathtt{K}^{-1})\mathbf{x}_1}\sqrt{\mathbf{x}_2^\top(\mathtt{K}^{-\top}\mathtt{K}^{-1})\mathbf{x}_2}} \,. \end{aligned} \tag{8.7}$$

The formula (8.7) shows that if $\mathtt{K}$, and consequently the matrix $\mathtt{K}^{-\top}\mathtt{K}^{-1}$, is known, then the angle between rays can be measured from their corresponding image points. A camera for which $\mathtt{K}$ is known is termed *calibrated*. A calibrated camera is a *direction sensor*, able to measure the direction of rays – like a 2D protractor.

The calibration matrix $\mathtt{K}$ also provides a relation between an image line and a scene plane:

Result 8.16. *An image line $\mathbf{l}$ defines a plane through the camera centre with normal direction $\mathbf{n} = \mathtt{K}^\top\mathbf{l}$ measured in the camera's Euclidean coordinate frame.*

Note, the normal $\mathbf{n}$ will not in general be a unit vector.

Proof. Points $\mathbf{x}$ on the line $\mathbf{l}$ back-project to directions $\mathbf{d} = \mathtt{K}^{-1}\mathbf{x}$ which are orthogonal to the plane normal $\mathbf{n}$, and thus satisfy $\mathbf{d}^\top\mathbf{n} = \mathbf{x}^\top\mathtt{K}^{-\top}\mathbf{n} = 0$. Since points on $\mathbf{l}$ satisfy $\mathbf{x}^\top\mathbf{l} = 0$, it follows that $\mathbf{l} = \mathtt{K}^{-\top}\mathbf{n}$, and hence $\mathbf{n} = \mathtt{K}^\top\mathbf{l}$. □

8.5.1 The image of the absolute conic

We now derive a very important result which relates the calibration matrix $\mathtt{K}$ to the image of the absolute conic, ω. First we must determine the map between the plane at infinity, π_∞, and the camera image plane. Points on π_∞ may be written as $\mathbf{X}_\infty = (\mathbf{d}^\top, 0)^\top$, and are imaged by a general camera $\mathtt{P} = \mathtt{K}\mathtt{R}[\mathtt{I} \mid -\widetilde{\mathbf{C}}]$ as

$$\mathbf{x} = \mathtt{P}\mathbf{X}_\infty = \mathtt{K}\mathtt{R}[\mathtt{I} \mid -\widetilde{\mathbf{C}}]\begin{pmatrix}\mathbf{d}\\ 0\end{pmatrix} = \mathtt{K}\mathtt{R}\mathbf{d}.$$

This shows that

Samsung/Google, EX1010, p. 227

- *the mapping between π_∞ and an image is given by the planar homography* $\mathbf{x} = \mathtt{H}\mathbf{d}$ *with*

$$\mathtt{H} = \mathtt{K}\mathtt{R}. \tag{8.8}$$

Note that this map is independent of the position of the camera, C, and depends only on the camera internal calibration and orientation with respect to the world coordinate frame.

Now, since the absolute conic Ω_∞ (section 3.6(p81)) is on π_∞ we can compute its image under H, and find

Result 8.17. *The image of the absolute conic (the IAC) is the conic* $\omega = (\mathtt{K}\mathtt{K}^\mathsf{T})^{-1} = \mathtt{K}^{-\mathsf{T}}\mathtt{K}^{-1}$.

Proof. From result 2.13(p37) under a point homography $\mathbf{x} \mapsto \mathtt{H}\mathbf{x}$ a conic C maps as $\mathtt{C} \mapsto \mathtt{H}^{-\mathsf{T}}\mathtt{C}\mathtt{H}^{-1}$. It follows that Ω_∞, which is the conic $\mathtt{C} = \Omega_\infty = \mathtt{I}$ on π_∞, maps to $\omega = (\mathtt{K}\mathtt{R})^{-\mathsf{T}}\mathtt{I}(\mathtt{K}\mathtt{R})^{-1} = \mathtt{K}^{-\mathsf{T}}\mathtt{R}\mathtt{R}^{-1}\mathtt{K}^{-1} = (\mathtt{K}\mathtt{K}^\mathsf{T})^{-1}$. So the IAC $\omega = (\mathtt{K}\mathtt{K}^\mathsf{T})^{-1}$. □

Like Ω_∞ the conic ω is an imaginary point conic with no real points. For the moment it may be thought of as a convenient algebraic device, but it will be used in computations later in this chapter, and also in chapter 19 on camera auto-calibration.
A few remarks here:

(i) The image of the absolute conic, ω, depends only on the internal parameters K of the matrix P; it does not depend on the camera orientation or position.

(ii) It follows from (8.7) that the angle between two rays is given by the simple expression

$$\cos\theta = \frac{\mathbf{x}_1^\mathsf{T}\omega\mathbf{x}_2}{\sqrt{\mathbf{x}_1^\mathsf{T}\omega\mathbf{x}_1}\sqrt{\mathbf{x}_2^\mathsf{T}\omega\mathbf{x}_2}}. \tag{8.9}$$

This expression is independent of the projective coordinate frame in the image, that is, it is unchanged under projective transformation of the image. To see this consider any 2D projective transformation, H. The points $\mathbf{x}_i$ are transformed to $\mathtt{H}\mathbf{x}_i$, and ω transforms (as any image conic) to $\mathtt{H}^{-\mathsf{T}}\omega\mathtt{H}^{-1}$. Thus, (8.9) is unchanged, and hence holds in any projective coordinate frame in the image.

(iii) A particularly important specialization of (8.9) is that if two image points $\mathbf{x}_1$ and $\mathbf{x}_2$ correspond to orthogonal directions then

$$\mathbf{x}_1^\mathsf{T}\omega\mathbf{x}_2 = 0. \tag{8.10}$$

This equation will be used at several points later in the book as it provides a linear constraint on ω.

(iv) We may also define the dual image of the absolute conic (the DIAC) as

$$\omega^* = \omega^{-1} = \mathtt{K}\mathtt{K}^\mathsf{T}. \tag{8.11}$$

This is a dual (line) conic, whereas ω is a point conic (though it contains no real points). The conic ω^* is the image of $\mathtt{Q}_\infty^*$ and is given by (8.5) $\omega^* = \mathtt{P}\mathtt{Q}_\infty^*\mathtt{P}^\mathsf{T}$.

Samsung/Google, EX1010, p. 228

(v) Result 8.17 shows that once ω (or equivalently ω^*) is identified in an image then $\mathtt{K}$ is also determined. This follows because a symmetric matrix ω may be uniquely decomposed into a product $\omega^* = \mathtt{K}\mathtt{K}^\mathsf{T}$ of an upper-triangular matrix with positive diagonal entries and its transpose by the Cholesky factorization (see result A4.5(p582)).

(vi) It was seen in chapter 3 that a plane π intersects π_∞ in a line, and this line intersects Ω_∞ in two points which are the circular points of π. The imaged circular points lie on ω at the points at which the vanishing line of the plane π intersects ω.

These final two properties of ω are the basis for a calibration algorithm, as shown in the following example.

Example 8.18. A simple calibration device

The image of three squares (on planes which are not parallel, but which need not be orthogonal) provides sufficiently many constraints to compute $\mathtt{K}$. Consider one of the squares. The correspondences between its four corner points and their images define the homography $\mathtt{H}$ between the plane π of the square and the image. Applying this homography to circular points on π determines their images as $\mathtt{H}(1, \pm i, 0)^\mathsf{T}$. Thus we have two points on the (as yet unknown) ω. A similar procedure applied to the other squares generates a total of six points on ω, from which it may be computed (since five points are required to determine a conic). In outline the algorithm has the following steps:

(i) For each square compute the homography $\mathtt{H}$ that maps its corner points, $(0,0)^\mathsf{T}, (1,0)^\mathsf{T}, (0,1)^\mathsf{T}, (1,1)^\mathsf{T}$, to their imaged points. (The alignment of the plane coordinate system with the square is a similarity transformation and does not affect the position of the circular points on the plane).

(ii) Compute the imaged circular points for the plane of that square as $\mathtt{H}(1, \pm i, 0)^\mathsf{T}$. Writing $\mathtt{H} = [\mathbf{h}_1, \mathbf{h}_2, \mathbf{h}_3]$, the imaged circular points are $\mathbf{h}_1 \pm i\mathbf{h}_2$.

(iii) Fit a conic ω to the six imaged circular points. The constraint that the imaged circular points lie on ω may be rewritten as two real constraints. If $\mathbf{h}_1 \pm i\mathbf{h}_2$ lies on ω then $(\mathbf{h}_1 \pm i\mathbf{h}_2)^\mathsf{T}\,\omega\,(\mathbf{h}_1 \pm i\mathbf{h}_2) = 0$, and the imaginary and real parts give respectively:

$$\mathbf{h}_1^\mathsf{T}\omega\mathbf{h}_2 = 0 \quad \text{and} \quad \mathbf{h}_1^\mathsf{T}\omega\mathbf{h}_1 = \mathbf{h}_2^\mathsf{T}\omega\mathbf{h}_2 \tag{8.12}$$

which are equations linear in ω. The conic ω is determined up to scale from five or more such equations.

(iv) Compute the calibration $\mathtt{K}$ from $\omega = (\mathtt{K}\mathtt{K}^\mathsf{T})^{-1}$ using the Cholesky factorization.

Figure 8.12 shows a calibration object consisting of three planes imprinted with squares, and the computed matrix $\mathtt{K}$. For the purpose of internal calibration, the squares have the advantage over a standard calibration object (e.g. figure 7.1(p182)) that no measured 3D co-ordinates are required. △

Samsung/Google, EX1010, p. 229

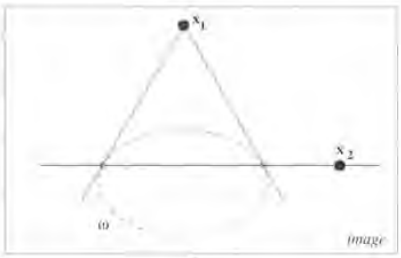


a

$$\mathtt{K} = \begin{bmatrix} 1108.3 & -9.8 & 525.8 \\ 0 & 1097.8 & 395.9 \\ 0 & 0 & 1 \end{bmatrix}$$

b

Fig. 8.12. **Calibration from metric planes.** *(a) Three squares provide a simple calibration object. The planes need not be orthogonal. (b) The computed calibration matrix using the algorithm of example 8.18. The image size is* 1024×768 *pixels.*



a b

Fig. 8.13. **Orthogonality represented by conjugacy and pole–polar relationships.** *(a) Image points* $\mathbf{x}_1, \mathbf{x}_2$ *back-project to orthogonal rays if the points are conjugate with respect to* ω*, i.e.* $\mathbf{x}_1^\top \omega \mathbf{x}_2 = 0$*. (b) The point* $\mathbf{x}$ *and line* $\mathbf{l}$ *back-project to a ray and plane that are orthogonal if* $\mathbf{x}$ *and* $\mathbf{l}$ *are pole–polar with respect to* ω*, i.e.* $\mathbf{l} = \omega\mathbf{x}$*. For example (see section 8.6.3), the vanishing point of the normal direction to a plane and the vanishing line of the plane are pole–polar with respect to* ω*.*

We will return to camera calibration in section 8.8, where vanishing points and lines provide constraints on K. The geometric constraints that are used in example 8.18 are discussed further in section 8.8.1.

8.5.2 Orthogonality and ω

The conic ω is a device for representing orthogonality in an image. It has already been seen (8.10) that if two image points $\mathbf{x}_1$ and $\mathbf{x}_2$ back-project to orthogonal rays, then the points satisfy $\mathbf{x}_1^\top \omega \mathbf{x}_2 = 0$. Similarly, it may be shown that

Result 8.19. *A point* $\mathbf{x}$ *and line* $\mathbf{l}$ *back-projecting to a ray and plane respectively that are orthogonal are related by* $\mathbf{l} = \omega\mathbf{x}$*.*

Geometrically these relations express that image points back-projecting to orthogonal rays are conjugate with respect to ω ($\mathbf{x}_1^\top \omega \mathbf{x}_2 = 0$), and that a point and line back-projecting to an orthogonal ray and plane are in a pole–polar relationship ($\mathbf{l} = \omega\mathbf{x}$). See section 2.8.1($p$58). A schematic representation of these two relations is given in figure 8.13.

Samsung/Google, EX1010, p. 230

These geometric representations of orthogonality, and indeed the projective representation (8.9) of the angle between two rays measured from image points, are simply specializations and a recapitulation of relations derived earlier in the book. For example, we have already developed a projective representation (3.23–*p*82) of the angle between two lines in 3-space, namely

$$\cos\theta = \frac{\mathbf{d}_1^\top \Omega_\infty \mathbf{d}_2}{\sqrt{\mathbf{d}_1^\top \Omega_\infty \mathbf{d}_1}\sqrt{\mathbf{d}_2^\top \Omega_\infty \mathbf{d}_2}}$$

where $\mathbf{d}_1$ and $\mathbf{d}_2$ are the directions of the lines (which are the points at which the lines intersect π_∞). Rays are lines in 3-space which are coincident at the camera centre, and so (3.23–*p*82) may be applied directly to rays. This is precisely what (8.9) does – it is simply (3.23–*p*82) computed in the image.

Under the map (8.8) $\mathtt{H} = \mathtt{KR}$, which is the homography between the plane π_∞ in the world coordinate frame and the image plane, $\Omega_\infty \mapsto \mathtt{H}^\top \boldsymbol{\omega} \mathtt{H} = (\mathtt{KR})^\top \boldsymbol{\omega} (\mathtt{KR})$ and $\mathbf{d}_i = \mathtt{H}^{-1}\mathbf{x}_i = (\mathtt{KR})^{-1}\mathbf{x}_i$. Substituting these relations into (3.23–*p*82) gives (8.9). Similarly the conjugacy and pole–polar relations for orthogonality in the image are a direct image of those on π_∞, as can be seen by comparing figure 3.8(*p*83) with figure 8.13.

In practice these orthogonality results find greatest application in the case of vanishing points and vanishing lines.

8.6 Vanishing points and vanishing lines

One of the distinguishing features of perspective projection is that the image of an object that stretches off to infinity can have finite extent. For example, an infinite scene line is imaged as a line terminating in a *vanishing point*. Similarly, parallel world lines, such as railway lines, are imaged as converging lines, and their image intersection is the vanishing point for the direction of the railway.

8.6.1 Vanishing points

The perspective geometry that gives rise to vanishing points is illustrated in figure 8.14. It is evident that geometrically the vanishing point of a line is obtained by intersecting the image plane with a ray parallel to the world line and passing through the camera centre. Thus a vanishing point depends only on the *direction* of a line, not on its position. Consequently a set of parallel world lines have a common vanishing point, as illustrated in figure 8.16.

Algebraically the vanishing point may be obtained as a limiting point as follows: Points on a line in 3-space through the point $\mathbf{A}$ and with direction $\mathbf{D} = (\mathbf{d}^\top, 0)^\top$ are written as $\mathbf{X}(\lambda) = \mathbf{A} + \lambda\mathbf{D}$, see figure 8.14b. As the parameter λ varies from 0 to ∞ the point $\mathbf{X}(\lambda)$ varies from the finite point $\mathbf{A}$ to the point $\mathbf{D}$ at infinity. Under a projective camera $\mathtt{P} = \mathtt{K}[\mathtt{I} \mid \mathtt{0}]$, a point $\mathbf{X}(\lambda)$ is imaged at

$$\mathbf{x}(\lambda) = \mathtt{P}\mathbf{X}(\lambda) = \mathtt{P}\mathbf{A} + \lambda\mathtt{P}\mathbf{D} = \mathbf{a} + \lambda\mathtt{K}\mathbf{d}$$

where $\mathbf{a}$ is the image of $\mathbf{A}$. Then the vanishing point $\mathbf{v}$ of the line is obtained as the

Samsung/Google, EX1010, p. 231

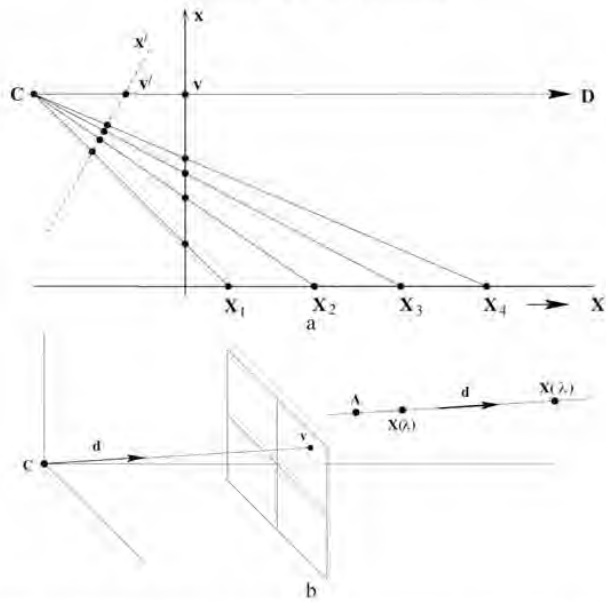


Fig. 8.14. **Vanishing point formation.** *(a) Plane to line camera. The points* $\mathbf{X}_i, i = 1, \ldots, 4$ *are equally spaced on the world line, but their spacing on the image line monotonically decreases. In the limit* $\mathbf{X} \to \infty$ *the world point is imaged at* $\mathbf{x} = \mathbf{v}$ *on the vertical image line, and at* $\mathbf{x}' = \mathbf{v}'$ *on the inclined image line. Thus the vanishing point of the world line is obtained by intersecting the image plane with a ray parallel to the world line through the camera centre* $\mathbf{C}$*. (b) 3-space to plane camera. The vanishing point,* $\mathbf{v}$*, of a line with direction* $\mathbf{d}$ *is the intersection of the image plane with a ray parallel to* $\mathbf{d}$ *through* $\mathbf{C}$*. The world line may be parametrized as* $\mathbf{X}(\lambda) = \mathbf{A} + \lambda\mathbf{D}$*, where* $\mathbf{A}$ *is a point on the line, and* $\mathbf{D} = (\mathbf{d}^\top, 0)^\top$*.*

limit

$$\mathbf{v} = \lim_{\lambda \to \infty} \mathbf{x}(\lambda) = \lim_{\lambda \to \infty} (\mathbf{a} + \lambda \mathtt{K}\mathbf{d}) = \mathtt{K}\mathbf{d}.$$

From result 8.15, $\mathbf{v} = \mathtt{K}\mathbf{d}$ means that the vanishing point $\mathbf{v}$ back-projects to a ray with direction $\mathbf{d}$. Note that $\mathbf{v}$ depends only on the direction $\mathbf{d}$ of the line, not on its position specified by $\mathbf{A}$.

In the language of projective geometry this result is obtained directly: In projective 3-space the plane at infinity $\boldsymbol{\pi}_\infty$ is the plane of directions, and all lines with the same direction intersect $\boldsymbol{\pi}_\infty$ in the same point (see chapter 3). The vanishing point is simply the image of this intersection. Thus if a line has direction $\mathbf{d}$, then it intersects $\boldsymbol{\pi}_\infty$ in the point $\mathbf{X}_\infty = (\mathbf{d}^\top, 0)^\top$. Then $\mathbf{v}$ is the image of $\mathbf{X}_\infty$

$$\mathbf{v} = \mathtt{P}\mathbf{X}_\infty = \mathtt{K}[\mathtt{I} \mid \mathbf{0}] \begin{pmatrix} \mathbf{d} \\ 0 \end{pmatrix} = \mathtt{K}\mathbf{d}.$$

To summarize:

Result 8.20. *The vanishing point of lines with direction* $\mathbf{d}$ *in 3-space is the intersection*

Samsung/Google, EX1010, p. 232

$\mathbf{v}$ *of the image plane with a ray through the camera centre with direction* $\mathbf{d}$, *namely* $\mathbf{v} = \mathtt{K}\mathbf{d}$.

Note, lines parallel to the image plane are imaged as parallel lines, since $\mathbf{v}$ is at infinity in the image. However, the converse – that parallel image lines are the image of parallel scene lines – does not hold since lines which intersect on the principal plane are imaged as parallel lines.

Example 8.21. Camera rotation from vanishing points
Vanishing points are images of points at infinity, and provide orientation (attitude) information in a similar manner to that provided by the fixed stars. Consider two images of a scene obtained by calibrated cameras, where the two cameras differ in orientation *and position*. The points at infinity are part of the scene and so are independent of the camera. Their images, the vanishing points, are not affected by the change in camera position, but are affected by the camera rotation. Suppose both cameras have the same calibration matrix $\mathtt{K}$, and the camera rotates by $\mathtt{R}$ between views.

Let a scene line have vanishing point $\mathbf{v}_i$ in the first view, and $\mathbf{v}'_i$ in the second. The vanishing point $\mathbf{v}_i$ has direction $\mathbf{d}_i$ measured in the first camera's Euclidean coordinate frame, and the corresponding vanishing point $\mathbf{v}'_i$ has direction $\mathbf{d}'_i$ measured in the second camera's Euclidean coordinate frame. These directions can be computed from the vanishing points, for example $\mathbf{d}_i = \mathtt{K}^{-1}\mathbf{v}_i / \|\mathtt{K}^{-1}\mathbf{v}_i\|$, where the normalizing factor $\|\mathtt{K}^{-1}\mathbf{v}_i\|$ is included to ensure that $\mathbf{d}_i$ is a unit vector. The directions $\mathbf{d}_i$ and $\mathbf{d}'_i$ are related by the camera rotation as $\mathbf{d}'_i = \mathtt{R}\mathbf{d}_i$, which represents two independent constraints on $\mathtt{R}$. Thus the rotation matrix $\mathtt{R}$ can be computed from two such corresponding directions. △

The angle between two scene lines. We have seen that the vanishing point of a scene line back-projects to a ray parallel to the scene line. Consequently (8.9), which determines the angle between rays back-projected from image points, enables the angle between the directions of two scene lines to be measured from their vanishing points:

Result 8.22. *Let* $\mathbf{v}_1$ *and* $\mathbf{v}_2$ *be the vanishing points of two lines in an image, and let* $\boldsymbol{\omega}$ *be the image of the absolute conic in the image. If* θ *is the angle between the two line directions, then*

$$\cos\theta = \frac{\mathbf{v}_1^\top \boldsymbol{\omega} \mathbf{v}_2}{\sqrt{\mathbf{v}_1^\top \boldsymbol{\omega} \mathbf{v}_1}\sqrt{\mathbf{v}_2^\top \boldsymbol{\omega} \mathbf{v}_2}} \,. \tag{8.13}$$

A note on computing vanishing points
Often vanishing points are computed from the image of a set of parallel line segments, though they may be determined in other ways for example by using equal length intervals on a line as described in example 2.18(p50) and example 2.20(p51). In the case of imaged parallel line segments the objective is to estimate their common image intersection – which is the image of the direction of the parallel scene lines. Due to measurement noise the imaged line segments will generally *not* intersect in a unique

Samsung/Google, EX1010, p. 233



Fig. 8.15. **ML estimate of a vanishing point from imaged parallel scene lines.** *(a) Estimating the vanishing point* $\mathbf{v}$ *involves fitting a line (shown thin here) through* $\mathbf{v}$ *to each measured line (shown thick here). The ML estimate of* $\mathbf{v}$ *is the point which minimizes the sum of squared orthogonal distances between the fitted lines and the measured lines' end points. (b) Measured line segments are shown in white, and fitted lines in black. (c) A close-up of the dashed square in (b). Note the very slight angle between the measured and fitted lines.*

point. Commonly the vanishing point is then computed by intersecting the lines pairwise and using the centroid of these intersections, or finding the closest point to all the measured lines. However, these are *not* optimal procedures.

Under the assumption of Gaussian measurement noise, the maximum likelihood estimate (MLE) of the vanishing point *and* line segments is computed by determining a set of lines that do intersect in a single point, and which minimize the sum of squared orthogonal distances from the endpoints of the measured line segments as shown in figure 8.15(a). This minimization may be computed numerically using the Levenberg–Marquardt algorithm (section A6.2(p600)). Note that if the lines are defined by fitting to many points, rather than just their end points, one can use the method described in section 16.7.2(p404) to reduce each line to an equivalent pair of weighted end points which can then be used in this algorithm. Figure 8.15(b)(c) shows an example of a vanishing point computed in this manner. It is evident that the residuals between the measured and fitted lines are very small.

8.6.2 Vanishing lines

Parallel planes in 3-space intersect π_∞ in a common line, and the image of this line is the vanishing line of the plane. Geometrically the vanishing line is constructed, as shown in figure 8.16, by intersecting the image with a plane parallel to the scene plane through the camera centre. It is clear that a vanishing line depends only on the *orientation* of the scene plane; it does not depend on its position. Since lines parallel to a plane intersect the plane at π_∞, it is easily seen that the vanishing point of a line parallel to a plane lies on the vanishing line of the plane. An example is shown in figure 8.17.

If the camera calibration $\mathtt{K}$ is known then a scene plane's vanishing line may be used to determine information about the plane, and we mention three examples here:

(i) The plane's orientation relative to the camera may be determined from its vanishing line. From result 8.16 a plane through the camera centre with normal

Samsung/Google, EX1010, p. 234

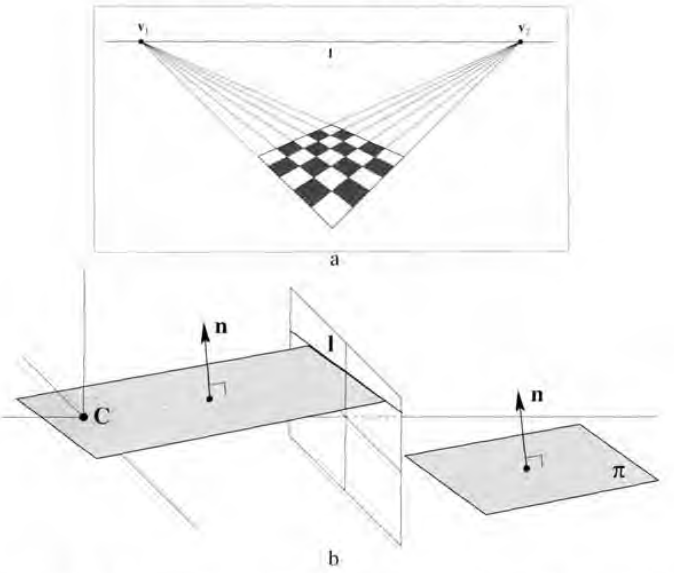


Fig. 8.16. **Vanishing line formation.** *(a) The two sets of parallel lines on the scene plane converge to the vanishing points* $\mathbf{v}_1$ *and* $\mathbf{v}_2$ *in the image. The line* $\mathbf{l}$ *through* $\mathbf{v}_1$ *and* $\mathbf{v}_2$ *is the vanishing line of the plane. (b) The vanishing line* $\mathbf{l}$ *of a plane* π *is obtained by intersecting the image plane with a plane through the camera centre* $\mathbf{C}$ *and parallel to* π.

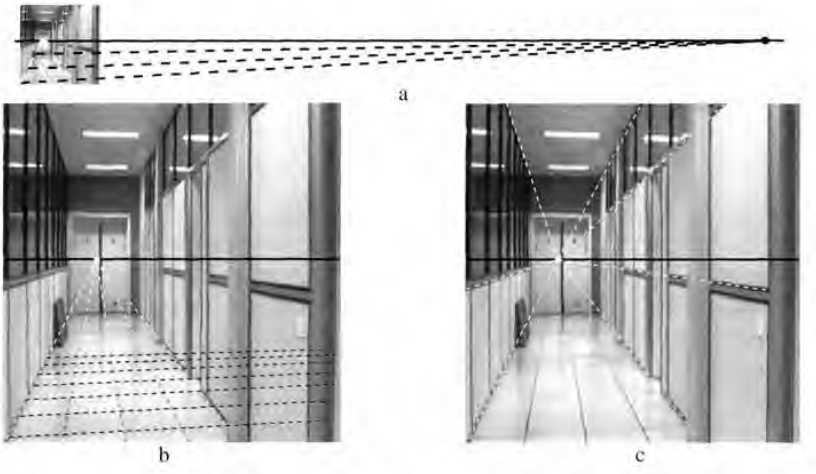


Fig. 8.17. **Vanishing points and lines.** *The vanishing line of the ground plane (the horizon) of the corridor may be obtained from two sets of parallel lines on the plane. (a) The vanishing points of lines which are nearly parallel to the image plane are distant from the finite (actual) image. (b) Note the monotonic decrease in the spacing of the imaged equally spaced parallel lines corresponding to the sides of the floor tiles. (c) The vanishing point of lines parallel to a plane (here the ground plane) lies on the vanishing line of the plane.*

Samsung/Google, EX1010, p. 235

direction $\mathbf{n}$ intersects the image plane in the line $\mathbf{l} = \mathtt{K}^{-\mathsf{T}}\mathbf{n}$. Consequently, $\mathbf{l}$ is the vanishing line of planes perpendicular to $\mathbf{n}$. Thus a plane with vanishing line $\mathbf{l}$ has orientation $\mathbf{n} = \mathtt{K}^{\mathsf{T}}\mathbf{l}$ in the camera's Euclidean coordinate frame.

(ii) The plane may be metrically rectified given only its vanishing line. This can be seen by considering a synthetic rotation of the camera in the manner of example 8.13(p205). Since the plane normal is known from the vanishing line, the camera can be synthetically rotated by a homography so that the plane is fronto-parallel (i.e. parallel to the image plane). The computation of this homography is discussed in exercise (ix).

(iii) The angle between two scene planes can be determined from their vanishing lines. Suppose the vanishing lines are $\mathbf{l}_1$ and $\mathbf{l}_2$, then the angle θ between the planes is given by

$$\cos\theta = \frac{\mathbf{l}_1^{\mathsf{T}}\boldsymbol{\omega}^*\mathbf{l}_2}{\sqrt{\mathbf{l}_1^{\mathsf{T}}\boldsymbol{\omega}^*\mathbf{l}_1}\sqrt{\mathbf{l}_2^{\mathsf{T}}\boldsymbol{\omega}^*\mathbf{l}_2}}. \tag{8.14}$$

The proof is left as an exercise.

Computing vanishing lines

A common way to determine a vanishing line of a scene plane is first to determine vanishing points for two sets of lines parallel to the plane, and then to construct the line through the two vanishing points. This construction is illustrated in figure 8.17. Alternative methods of determining vanishing points are shown in example 2.19(p51) and example 2.20(p51).

However, the vanishing line may be determined directly, without using vanishing points as an intermediate step. For example, the vanishing line may be computed given an imaged set of equally spaced coplanar parallel lines. This is a useful method in practice because such sets commonly occur in man-made structures, such as: stairs, windows on the wall of a building, fences, radiators and zebra crossings. The following example illustrates the projective geometry involved.

Example 8.23. The vanishing line given the image of three coplanar equally spaced parallel lines

A set of equally spaced lines on the scene plane may be represented as $ax' + by' + \lambda = 0$, where λ takes integer values. This set (a pencil) of lines may be written as $\mathbf{l}'_n = (a, b, n)^{\mathsf{T}} = (a, b, 0)^{\mathsf{T}} + n(0, 0, 1)^{\mathsf{T}}$, where $(0, 0, 1)^{\mathsf{T}}$ is the line at infinity on the scene plane. Under perspective imaging the point transformation is $\mathbf{x} = \mathtt{H}\mathbf{x}'$, and the corresponding line map is $\mathbf{l}_n = \mathtt{H}^{-\mathsf{T}}\mathbf{l}'_n = \mathbf{l}_0 + n\mathbf{l}$, where $\mathbf{l}$, the image of $(0, 0, 1)^{\mathsf{T}}$, is the vanishing line of the plane. The imaged geometry is shown in figure 8.18(c). Note all lines $\mathbf{l}_n$ intersect in a common vanishing point (which is given by $\mathbf{l}_i \times \mathbf{l}_j$, for $i \neq j$) and the spacing decreases monotonically with n. The vanishing line $\mathbf{l}$ may be determined from three lines of the set provided their index (n) is identified. For example, from the image of three equally spaced lines, $\mathbf{l}_0, \mathbf{l}_1$ and $\mathbf{l}_2$, the closed form solution for the vanishing line is:

$$\mathbf{l} = \Big((\mathbf{l}_0 \times \mathbf{l}_2)^{\mathsf{T}}(\mathbf{l}_1 \times \mathbf{l}_2)\Big)\,\mathbf{l}_1 + 2\,\Big((\mathbf{l}_0 \times \mathbf{l}_1)^{\mathsf{T}}(\mathbf{l}_2 \times \mathbf{l}_1)\Big)\,\mathbf{l}_2. \tag{8.15}$$

Samsung/Google, EX1010, p. 236

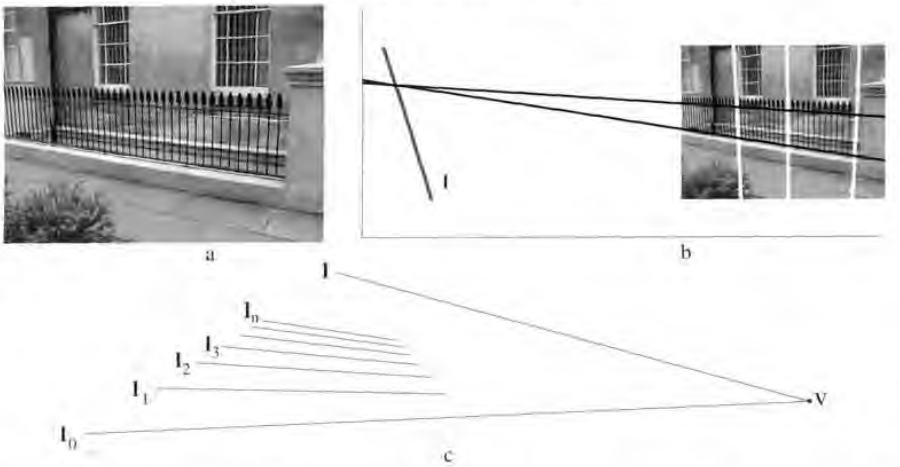


Fig. 8.18. **Determining a planes vanishing line from imaged equally spaced parallel lines.** *(a) Image of a vertical fence with equally spaced bars. (b) The computed vanishing line* $\mathbf{l}$ *from three equally spaced bars (12 apart). Note the vanishing point of the horizontal lines lies on this vanishing line. (c) The spacing between the imaged lines* $\mathbf{l}_n$ *monotonically decreases with* n.

The proof is left as an exercise. Figure 8.18(b) shows a vanishing line computed in this way. △

8.6.3 Orthogonality relationships amongst vanishing points and lines

It is often the case in practice that the lines and planes giving rise to vanishing points are orthogonal. In this case there are particularly simple relationships amongst their vanishing points and lines involving $\boldsymbol{\omega}$, and furthermore these relations can be used to (partially) determine $\boldsymbol{\omega}$, and consequently the camera calibration $\mathtt{K}$ as will be seen in section 8.8.

It follows from (8.13) that the vanishing points, $\mathbf{v}_1, \mathbf{v}_2$, of two perpendicular world lines satisfy $\mathbf{v}_1^\top \boldsymbol{\omega} \mathbf{v}_2 = 0$. This means that the vanishing points are conjugate with respect to $\boldsymbol{\omega}$, as illustrated in figure 8.13. Similarly it follows from result 8.19 that the vanishing point $\mathbf{v}$ of a direction perpendicular to a plane with vanishing line $\mathbf{l}$ satisfies $\mathbf{l} = \boldsymbol{\omega}\mathbf{v}$. This means that the vanishing point and line are in a pole–polar relation with respect to $\boldsymbol{\omega}$, as is also illustrated in figure 8.13. Summarizing these image relations:

(i) The vanishing points of lines with perpendicular directions satisfy

$$\mathbf{v}_1^\top \boldsymbol{\omega} \mathbf{v}_2 = 0. \tag{8.16}$$

(ii) If a line is perpendicular to a plane then their respective vanishing point $\mathbf{v}$ and vanishing line $\mathbf{l}$ are related by

$$\mathbf{l} = \boldsymbol{\omega}\mathbf{v} \tag{8.17}$$

and inversely $\mathbf{v} = \boldsymbol{\omega}^* \mathbf{l}$.

(iii) The vanishing lines of two perpendicular planes satisfy $\mathbf{l}_1^\top \boldsymbol{\omega}^* \mathbf{l}_2 = 0$.

Samsung/Google, EX1010, p. 237

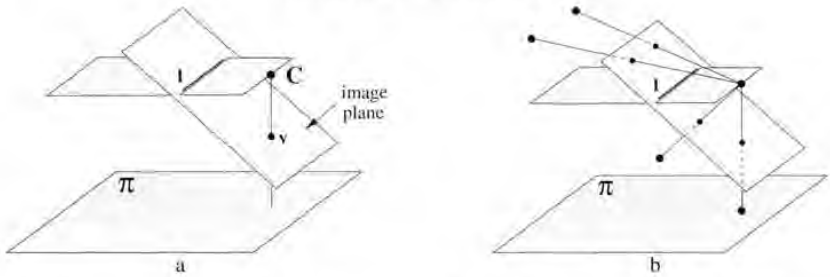


Fig. 8.19. **Geometry of a vertical vanishing point and ground plane vanishing line.** *(a) The vertical vanishing point* $\mathbf{v}$ *is the image of the vertical "footprint" of the camera centre on the ground plane* π. *(b) The vanishing line* $\mathbf{l}$ *partitions all points in scene space. Any scene point projecting onto the vanishing line is at the same distance from the plane* π *as the camera centre: if it lies "above" the line it is farther from the plane, and if "below" then it is closer to the plane than the camera centre.*

For example, suppose the vanishing line $\mathbf{l}$ of the ground plane (the horizon) is identified in an image, and the internal calibration matrix $\mathtt{K}$ is known, then the vertical vanishing point $\mathbf{v}$ (which is the vanishing point of the normal direction to the plane) may be obtained from $\mathbf{v} = \omega^*\mathbf{l}$.

8.7 Affine 3D measurements and reconstruction

It has been seen in section 2.7.2(*p*49) that identifying a scene plane's vanishing line allows affine properties of the scene plane to be measured. If in addition a vanishing point for a direction not parallel to the plane is identified, then affine properties can be computed for the 3-space of the perspectively imaged scene. We will illustrate this idea for the case where the vanishing point corresponds to a direction orthogonal to the plane, although orthogonality is not necessary for the construction. The method described in this section does not require that the internal calibration of the camera $\mathtt{K}$ be known.

It will be convenient to think of the scene plane as the horizontal ground plane, in which case the vanishing line is the *horizon*. Similarly, it will be convenient to think of the direction orthogonal to the scene plane as vertical, so that $\mathbf{v}$ is the vertical vanishing point. This situation is illustrated in figure 8.19.

Suppose we wish to measure the relative lengths of two line segments in the vertical direction as shown in figure 8.20(a). We will show the following result:

Result 8.24. *Given the vanishing line of the ground plane* $\mathbf{l}$ *and the vertical vanishing point* $\mathbf{v}$, *then the relative length of vertical line segments can be measured provided their end point lies on the ground plane.*

Clearly the relative lengths cannot be measured directly from their imaged lengths because as a vertical line recedes deeper into the scene (i.e. further from the camera) then its imaged length decreases. The construction to determine the relative lengths proceeds in two steps:

Samsung/Google, EX1010, p. 238

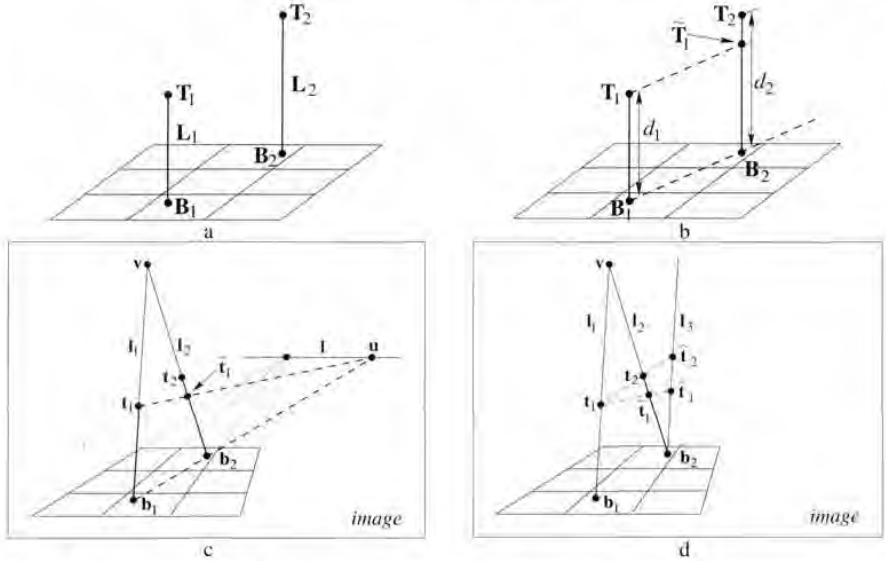


Fig. 8.20. **Computing length ratios of parallel scene lines.** *(a)* **3D geometry:** *The vertical line segments* $\mathbf{L}_1 = \langle \mathbf{B}_1, \mathbf{T}_1 \rangle$ *and* $\mathbf{L}_2 = \langle \mathbf{B}_2, \mathbf{T}_2 \rangle$ *have length* d_1 *and* d_2 *respectively. The base points* $\mathbf{B}_1, \mathbf{B}_2$ *are on the ground plane. We wish to compute the scene length ratio* $d_1 : d_2$ *from the imaged configuration. (b) In the scene the length of the line segment* $\mathbf{L}_1$ *may be transferred to* $\mathbf{L}_2$ *by constructing a line parallel to the ground plane to generate the point* $\widetilde{\mathbf{T}}_1$*. (c)* **Image geometry:** $\mathbf{l}$ *is the ground plane vanishing line, and* $\mathbf{v}$ *the vertical vanishing point. A corresponding parallel line construction in the image requires first determining the vanishing point* $\mathbf{u}$ *from the images* $\mathbf{b}_i$ *of* $\mathbf{B}_i$*, and then determining* $\tilde{\mathbf{t}}_1$ *(the image of* $\widetilde{\mathbf{T}}_1$*) by the intersection of* $\mathbf{l}_2$ *and the line* $\langle \mathbf{t}_1, \mathbf{u} \rangle$*. (d) The line* $\mathbf{l}_3$ *is parallel to* $\mathbf{l}_1$ *in the image. The points* $\mathbf{t}_1$ *and* $\mathbf{t}_2$ *are constructed by intersecting* $\mathbf{l}_3$ *with the lines* $\langle \mathbf{t}_1, \tilde{\mathbf{t}}_1 \rangle$ *and* $\langle \mathbf{t}_1, \mathbf{t}_2 \rangle$ *respectively. The distance ratio* $d(\mathbf{b}_2, \tilde{\mathbf{t}}_1) : d(\mathbf{b}_2, \mathbf{t}_2)$ *is the computed estimate of* $d_1 : d_2$*.*

Step 1: Map the length of one line segment onto the other. In 3D the length of $\mathbf{L}_1$ may be compared to $\mathbf{L}_2$ by constructing a line parallel to the ground plane in the direction $\langle \mathbf{B}_1, \mathbf{B}_2 \rangle$ that transfers $\mathbf{T}_1$ onto $\mathbf{L}_2$. This transferred point will be denoted $\widetilde{\mathbf{T}}_1$ (see figure 8.20(b)). In the image a corresponding construction is carried out by first determining the vanishing point $\mathbf{u}$ which is the intersection of $\langle \mathbf{b}_1, \mathbf{b}_2 \rangle$ with $\mathbf{l}$. Now any scene line parallel to $\langle \mathbf{B}_1, \mathbf{B}_2 \rangle$ is imaged as a line through $\mathbf{u}$, so in particular the image of the line through $\mathbf{T}_1$ parallel to $\langle \mathbf{B}_1, \mathbf{B}_2 \rangle$ is the line through $\mathbf{t}_1$ and $\mathbf{u}$. The intersection of the line $\langle \mathbf{t}_1, \mathbf{u} \rangle$ with $\mathbf{l}_2$ defines the image $\tilde{\mathbf{t}}_1$ of the transferred point $\widetilde{\mathbf{T}}_1$ (see figure 8.20(c)).

Step 2: Determine the ratio of lengths on the scene line. We now have four collinear points on an imaged scene line and wish to determine the actual length ratio in the scene. The four collinear image points are $\mathbf{b}_2, \tilde{\mathbf{t}}_1, \mathbf{t}_2$ and $\mathbf{v}$. These may be treated as images of scene points at distances $0, d_1, d_2$ and ∞, respectively, along the scene line. The affine ratio $d_1 : d_2$ may be obtained by applying a projective transfor-

Samsung/Google, EX1010, p. 239

Objective

Given the vanishing line of the ground plane $\mathbf{l}$ and the vertical vanishing point $\mathbf{v}$ and the top $(\mathbf{t}_1, \mathbf{t}_2)$ and base $(\mathbf{b}_1, \mathbf{b}_2)$ points of two line segments as in figure 8.20, compute the ratio of lengths of the line segments in the scene.

Algorithm

(i) Compute the vanishing point $\mathbf{u} = (\mathbf{b}_1 \times \mathbf{b}_2) \times \mathbf{l}$.
(ii) Compute the transferred point $\tilde{\mathbf{t}}_1 = (\mathbf{t}_1 \times \mathbf{u}) \times \mathbf{l}_2$ (where $\mathbf{l}_2 = \mathbf{v} \times \mathbf{b}_2$).
(iii) Represent the four points $\mathbf{b}_2, \tilde{\mathbf{t}}_1, \mathbf{t}_2$ and $\mathbf{v}$ on the image line $\mathbf{l}_1$ by their distance from $\mathbf{b}_2$, as $0, \tilde{t}_1, t_2$ and v respectively.
(iv) Compute a 1D projective transformation $\mathtt{H}_{2\times2}$ mapping homogeneous coordinates $(0,1) \mapsto (0,1)$ and $(v,1) \mapsto (1,0)$ (which maps the vanishing point $\mathbf{v}$ to infinity). A suitable matrix is given by

$$\mathtt{H}_{2\times2} = \begin{bmatrix} 1 & 0 \\ 1 & -v \end{bmatrix}.$$

(v) The (scaled) distance of the scene points $\tilde{\mathbf{T}}_1$ and $\mathbf{T}_2$ from $\mathbf{B}_2$ on $\mathbf{L}_2$ may then be obtained from the position of the points $\mathtt{H}_{2\times2}(\tilde{t}_1, 1)^\top$ and $\mathtt{H}_{2\times2}(t_2, 1)^\top$. Their distance ratio is then given by

$$\frac{d_1}{d_2} = \frac{\tilde{t}_1(v - t_2)}{t_2(v - \tilde{t}_1)}$$

Algorithm 8.1. *Computing scene length ratios from a single image.*

mation to the image line which maps $\mathbf{v}$ to infinity. A geometric construction of this projectivity is shown in figure 8.20(d) (see example 2.20(p51)).

Details of the algorithm to carry out these two steps are given in algorithm 8.1.

Note, no knowledge of the camera calibration $\mathtt{K}$ or pose is necessary to apply the algorithm. In fact, the position of the camera centre relative to the ground plane can also be computed. The algorithm is well conditioned even when the vanishing point and/or line are at infinity in the image. For example, under affine image conditionings, or if the image plane is parallel to the vertical scene direction (so that $\mathbf{v}$ is at infinity). In these cases the distance ratio simplifies to $\frac{d_1}{d_2} = \frac{\tilde{t}_1}{t_2}$.

Example 8.25. Measuring a person's height in a single image

Suppose we have an image which contains sufficient information to compute the ground plane vanishing line and the vertical vanishing point, and also one object of known height for which the top and base are imaged. Then the height of a person standing on the ground plane can be measured anywhere in the scene provided that their head and feet are both visible. Figure 8.21(a) shows an example. The scene contains plenty of horizontal lines from which to compute a horizontal vanishing point. Two such vanishing points determine the vanishing line of the floor (which is the horizon for this image). The scene also contains plenty of vertical lines from which to compute a vertical vanishing point (figure 8.21(c)). Assuming that the two people are standing vertically, then their relative height may be computed directly from their length ratio using algorithm 8.1. Their absolute height may be determined by comput-

Samsung/Google, EX1010, p. 240



Fig. 8.21. **Height measurements using affine properties.** *(a) The original image. We wish to measure the height of the two people. (b) The image after radial distortion correction (see section 7.4(p189)). (c) The vanishing line (shown) is computed from two vanishing points corresponding to horizontal directions. The lines used to compute the vertical vanishing points are also shown. The vertical vanishing point is not shown since it lies well below the image. (d) Using the known height of the filing cabinet on the left of the image, the absolute height of the two people are measured as described in algorithm 8.1. The measured heights are within 2cm of ground truth. The computation of the uncertainty is described in [Criminisi-00].*

ing their height relative to an object on the ground plane with known height. Here the known height is provided by the filing cabinet. The result is shown in figure 8.21(d).

△

8.8 Determining camera calibration K from a single view

We have seen that once ω is known the angle between rays can be measured. Conversely if the angle between rays is known then a constraint is placed on ω. Each known angle between two rays gives a constraint of the form (8.13) on ω. Unfortunately, for arbitrary angles, and known $\mathbf{v}_1$ and $\mathbf{v}_2$, this gives a quadratic constraint on the entries of ω. If the lines are perpendicular, however, (8.13) reduces to (8.16) $\mathbf{v}_1^\top \omega \mathbf{v}_2 = 0$, and the constraint on ω is linear.

A linear constraint on ω also results from a vanishing point and vanishing line arising from a line and its orthogonal plane. A common example is a vertical direction and horizontal plane as in figure 8.19. From (8.17) $\mathbf{l} = \omega \mathbf{v}$. Writing this as $\mathbf{l} \times (\omega \mathbf{v}) = 0$

Samsung/Google, EX1010, p. 241

Condition	constraint	type	# constraints
vanishing points $\mathbf{v}_1, \mathbf{v}_2$ corresponding to orthogonal lines	$\mathbf{v}_1^\top \boldsymbol{\omega} \mathbf{v}_2 = 0$	linear	1
vanishing point $\mathbf{v}$ and vanishing line $\mathbf{l}$ corresponding to orthogonal line and plane	$[\mathbf{l}]_\times \boldsymbol{\omega} \mathbf{v} = \mathbf{0}$	linear	2
metric plane imaged with known homography $\mathtt{H} = [\mathbf{h}_1, \mathbf{h}_2, \mathbf{h}_3]$	$\mathbf{h}_1^\top \boldsymbol{\omega} \mathbf{h}_2 = 0$ $\mathbf{h}_1^\top \boldsymbol{\omega} \mathbf{h}_1 = \mathbf{h}_2^\top \boldsymbol{\omega} \mathbf{h}_2$	linear	2
zero skew	$\omega_{12} = \omega_{21} = 0$	linear	1
square pixels	$\omega_{12} = \omega_{21} = 0$ $\omega_{11} = \omega_{22}$	linear	2

Table 8.1. **Scene and internal constraints on $\boldsymbol{\omega}$.**

removes the homogeneous scaling factor and results in three homogeneous equations linear in the entries of $\boldsymbol{\omega}$. These are equivalent to two independent constraints on $\boldsymbol{\omega}$.

All these conditions provide linear constraints on $\boldsymbol{\omega}$. Given a sufficient number of such constraints $\boldsymbol{\omega}$ may be computed and hence the camera calibration $\mathtt{K}$ also follows since $\boldsymbol{\omega} = (\mathtt{K}\mathtt{K}^\top)^{-1}$.

The number of entries of $\boldsymbol{\omega}$ that need be determined from scene constraints of this sort can be reduced if the calibration matrix $\mathtt{K}$ has a more specialized form than (6.10–*p*157). In the case where $\mathtt{K}$ is known to have zero skew ($s = 0$), or square pixels ($\alpha_x = \alpha_y$ and $s = 0$), we can take advantage of this condition to help find $\boldsymbol{\omega}$. In particular, it is quickly verified by direct computation that:

Result 8.26. *If $s = \mathtt{K}_{12} = 0$ then $\omega_{12} = \omega_{21} = 0$. If in addition $\alpha_x = \mathtt{K}_{11} = \mathtt{K}_{22} = \alpha_y$, then $\omega_{11} = \omega_{22}$.*

Thus, in solving for the image of the absolute conic, one may easily take into account the zero-skew or square-aspect ratio constraint on the camera, if such a constraint is known to exist. One may also verify that no such simple connection as result 8.26 exists between the entries of $\mathtt{K}$ and those of $\boldsymbol{\omega}^* = \mathtt{K}\mathtt{K}^\top$.

We have now seen three sources of constraints on $\boldsymbol{\omega}$:

(i) metric information on a plane imaged with a known homography, see (8.12–*p*211)
(ii) vanishing points and lines corresponding to perpendicular directions and planes, (8.16)
(iii) "internal constraints" such as zero skew or square pixels, as in result 8.26

These constraints are summarized in table 8.1. We now describe how these constraints may be combined to estimate $\boldsymbol{\omega}$ and thence $\mathtt{K}$.

Since all the above constraints (including the internal constraints) are described algebraically as linear equations on $\boldsymbol{\omega}$, it is a simple matter to combine them as rows of

Samsung/Google, EX1010, p. 242

Objective

Compute K via ω by combining scene and internal constraints.

Algorithm

(i) Represent ω as a homogeneous 6-vector $\mathbf{w} = (w_1, w_2, w_3, w_4, w_5, w_6)^\top$ where:

$$\omega = \begin{bmatrix} w_1 & w_2 & w_4 \\ w_2 & w_3 & w_5 \\ w_4 & w_5 & w_6 \end{bmatrix}$$

(ii) Each available constraint from table 8.1 may be written as $\mathbf{a}^\top\mathbf{w} = 0$. For example, for the orthogonality constraint $\mathbf{u}^\top\omega\mathbf{v} = 0$, where $\mathbf{u} = (u_1, u_2, u_3)^\top$ and $\mathbf{v} = (v_1, v_2, v_3)^\top$, the 6-vector $\mathbf{a}$ is given by

$$\mathbf{a} = (v_1u_1, v_1u_2 + v_2u_1, v_2u_2, v_1u_3 + v_3u_1, v_2u_3 + v_3u_2, v_3u_3)^\top.$$

Similar constraints vectors are obtained from the other sources of scene and internal constraints. For example a metric plane generates two such constraints.

(iii) Stack the equations $\mathbf{a}^\top\mathbf{w} = 0$ from each constraint in the form $\mathtt{A}\mathbf{w} = \mathbf{0}$, where A is a $n \times 6$ matrix for n constraints.

(iv) Solve for $\mathbf{w}$ using the SVD as in algorithm 4.2(p109). This determines ω.

(v) Decompose ω into K using matrix inversion and Cholesky factorization (see section A4.2.1(p582)).

Algorithm 8.2. *Computing K from scene and internal constraints.*

a constraint matrix. All constraints may be collected together so that for n constraints the system of equations may be written as $\mathtt{A}\mathbf{w} = \mathbf{0}$, where A is a $n \times 6$ matrix and $\mathbf{w}$ is a 6-vector containing the six distinct homogeneous entries of ω. With a minimum of 5 constraint equations an exact solution is found. With more than five equations, a least-squares solution is found by algorithm A5.4(p593). The method is summarized in algorithm 8.2.

With more than the minimum required five constraints, we have the option to apply some of the constraints as hard constraints – that is, constraints that will be satisfied exactly. This can be done by parametrizing ω so that the constraints are satisfied explicitly (for instance setting $\omega_{21} = \omega_{12} = 0$ for the zero skew constraint, and also $\omega_{11} = \omega_{22}$ for the square-pixel constraint). The minimization method of algorithm A5.5(p594) may also be used to enforce hard constraints. Otherwise, treating all constraints as soft constraints and using algorithm A5.4(p593) will produce a solution in which the constraints are not satisfied exactly in the presence of noise – for instance, pixels may not be quite square.

Finally, an important issue in practice is that of degeneracy. This occurs when the combined constraints are not independent and results in the matrix A dropping rank. If the rank is less than the number of unknowns, then a parametrized family of solutions for ω (and hence K) is obtained. Also, if conditions are near degenerate then the solution is ill-conditioned and the particular member of the family is determined by "noise". These degeneracies can often be understood geometrically – for example in example 8.18 if the three metric planes are parallel then the three pairs of imaged

Samsung/Google, EX1010, p. 243



Fig. 8.22. *For the case that image skew is zero and the aspect ratio unity* **the principal point is the orthocentre of an orthogonal triad of vanishing points**. *(a) Original image. (b) Three sets of parallel lines in the scene, with each set having direction orthogonal to the others. (c) The principal point is the orthocentre of the triangle with the vanishing points as vertices.*

circular points are coincident and only provide a total of two constraints instead of six. A pragmatic solution to the problem of degeneracy, popularized by Zhang [Zhang-00], is to image a metric plane many times in varying positions. This reduces the chances of degeneracy occurring, and also provides a very over-determined solution.

Example 8.27. Calibration from three orthogonal vanishing points
Suppose that it is known that the camera has zero skew, and that the pixels are square (or equivalently their aspect ratio is known). A triad of orthogonal vanishing point directions supplies three more constraints. This gives a total of 5 constraints – sufficient to compute $\boldsymbol{\omega}$, and hence $\mathtt{K}$.

In outline the algorithm has the following steps:

(i) In the case of square pixels $\boldsymbol{\omega}$ has the form

$$\boldsymbol{\omega} = \begin{bmatrix} w_1 & 0 & w_2 \\ 0 & w_1 & w_3 \\ w_2 & w_3 & w_4 \end{bmatrix}.$$

(ii) Each pair of vanishing points $\mathbf{v}_i, \mathbf{v}_j$ generates an equation $\mathbf{v}_i^\top \boldsymbol{\omega} \mathbf{v}_j = 0$, which is linear in the elements of $\boldsymbol{\omega}$. The constraints from the three pairs of vanishing points are stacked together to form an equation $\mathtt{A}\mathbf{w} = \mathbf{0}$, where $\mathtt{A}$ is a 3×4 matrix.

(iii) The vector $\mathbf{w}$ is obtained as the null vector of $\mathtt{A}$, and this determines $\boldsymbol{\omega}$. The matrix $\mathtt{K}$ is obtained from $\boldsymbol{\omega} = (\mathtt{K}\mathtt{K}^\top)^{-1}$ by Cholesky factorization of $\boldsymbol{\omega}$, followed by inversion.

An example is shown in figure 8.22(a). Vanishing points are computed corresponding to the three perpendicular directions shown in figure 8.22(b). The image is 1024×768 pixels, and the calibration matrix is computed to be

$$\mathtt{K} = \begin{bmatrix} 1163 & 0 & 548 \\ 0 & 1163 & 404 \\ 0 & 0 & 1 \end{bmatrix}.$$

△

Samsung/Google, EX1010, p. 244

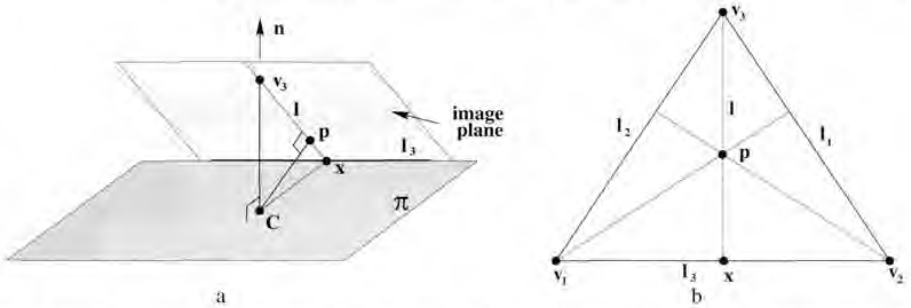


Fig. 8.23. **Geometric construction of the principal point.** *The vanishing line $\mathbf{l}_3$ back-projects to a plane π with normal $\mathbf{n}$. The vanishing point $\mathbf{v}_3$ back-projects to a line orthogonal to the plane π. (a) The normal $\mathbf{n}$ of the plane π through the camera centre $\mathbf{C}$ and the principal axis define a plane, which intersects the image in the line $\mathbf{l} = \langle \mathbf{v}_3, \mathbf{x} \rangle$. The line $\mathbf{l}_3$ is the intersection of π with the image plane, and is also its vanishing line. The point $\mathbf{v}_3$ is the intersection of the normal with the image plane, and is also its vanishing point. Clearly the principal point lies on $\mathbf{l}$, and $\mathbf{l}$ and $\mathbf{l}_3$ are perpendicular on the image plane. (b) The principal point may be determined from three such constraints as the orthocentre of the triangle.*

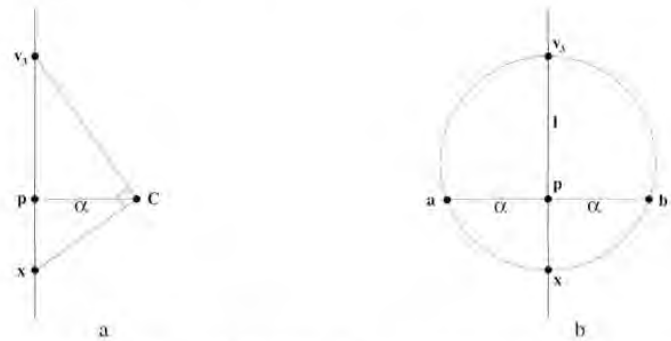


Fig. 8.24. **Geometric construction of the focal length.** *(a) Consider the plane defined by the camera centre $\mathbf{C}$, principal point and one of the vanishing points, e.g. $\mathbf{v}_3$ as shown in figure 8.23(a). The rays from $\mathbf{C}$ to $\mathbf{v}_3$ and $\mathbf{x}$ are perpendicular to each other. The focal length, α, is the distance from the camera centre to the image plane. By similar triangles, $\alpha^2 = d(\mathbf{p}, \mathbf{v}_3)d(\mathbf{p}, \mathbf{x})$, where $d(\mathbf{u}, \mathbf{v})$ is the distance between the points $\mathbf{u}$ and $\mathbf{v}$. (b) In the image a circle is drawn with diameter the line between $\mathbf{v}_3$ and $\mathbf{x}$. A line through $\mathbf{p}$ perpendicular to $\langle \mathbf{v}_3, \mathbf{x} \rangle$ meets the circle in two points $\mathbf{a}$ and $\mathbf{b}$. The focal length equals the distance $d(\mathbf{p}, \mathbf{a})$.*

The principal point and focal length may also be computed geometrically in this case. The principal point is the orthocentre of the triangle with vertices the vanishing points. Figure 8.23 shows that the principal point lies on the perpendicular line from one triangle side to the opposite vertex. A similar construction for the other two sides shows that the principal point is the orthocentre. An algebraic derivation of this result is left to the exercises. The focal length can also be computed geometrically as shown in figure 8.24.

As a cautionary note, this estimation method is degenerate if one of the vanishing

Samsung/Google, EX1010, p. 245



a b

Fig. 8.25. **Plane rectification via partial internal parameters** *(a) Original image. (b) Rectification assuming the camera has square pixels and principal point at the centre of the image. The focal length is computed from the single orthogonal vanishing point pair. The aspect ratio of a window in the rectified image differs from the ground truth value by 3.7%. Note that the two parallel planes, the upper building facade and the lower shopfront, are both mapped to fronto-parallel planes.*

points, say the vertical, is at infinity. In this case $\mathtt{A}$ drops rank to two, and there is a one-parameter family of solutions for ω and correspondingly for $\mathtt{K}$. This degeneracy can be seen geometrically from the orthocentre construction of figure 8.23. If $\mathbf{v}_3$ is at infinity then the principal point $\mathbf{p}$ lies on the line $\mathbf{l}_3 = \langle \mathbf{v}_1, \mathbf{v}_2 \rangle$, but its x position is not defined.

Example 8.28. Determining the focal length when the other internal parameters are known

We consider a further example of calibration from a single view. Suppose that it is known that the camera has zero skew, that the pixels are square (or equivalently their aspect ratio is known), and also that the principal point is at the image centre. Then only the focal length is unknown. In this case, the form of ω is very simple: it is a diagonal matrix $\mathrm{diag}(1/f^2, 1/f^2, 1)$ with only one degree of freedom. Using algorithm 8.2, the focal length f may be determined from one further constraint, such as the one arising from two vanishing points corresponding to orthogonal directions.

An example is shown in figure 8.25(a). Here the vanishing points used in the constraint are computed from the horizontal edges of the windows and pavement, and the vertical edges of the windows. These vanishing points also determine the vanishing line $\mathbf{l}$ of the building facade. Given $\mathtt{K}$ and the vanishing line $\mathbf{l}$, the camera can be synthetically rotated such that the facade is fronto-parallel by mapping the image with a homography as in example 8.13(p205). The result is shown in figure 8.25(b). Note, in example 8.13 it was necessary to know the aspect ratio of a rectangle on the scene plane in order to rectify the plane. Here it is only necessary to know the vanishing line of the plane because the camera calibration $\mathtt{K}$ provides the additional information required for the homography. △

8.8.1 The geometry of the constraints

Although the algebraic constraints given in table 8.1 appear to arise from distinct sources, they are in fact all equivalent to one of two simple geometric relations: two points lying on the conic ω, or conjugacy of two points with respect to ω.

For example, the zero skew constraint is an orthogonality constraint: it specifies that

Samsung/Google, EX1010, p. 246

the image x and y axes are orthogonal. These axes correspond to rays with directions in the camera's Euclidean coordinate frame, $(1,0,0)^\top$ and $(0,1,0)^\top$, respectively, that are imaged at $\mathbf{v}_x = (1,0,0)^\top$ and $\mathbf{v}_y = (0,1,0)^\top$ (since the rays are parallel to the image plane). The zero skew constraint $\omega_{12} = \omega_{21} = 0$ is just another way of writing the orthogonality constraint (8.16) $\mathbf{v}_y^\top \boldsymbol{\omega} \mathbf{v}_x = 0$. Geometrically skew zero is equivalent to conjugacy of the points $(1,0,0)^\top$ and $(0,1,0)^\top$ with respect to $\boldsymbol{\omega}$.

The square pixel constraint may be interpreted in two ways. A square has the property of defining two sets of orthogonal lines: adjacent edges are orthogonal, and so are the two diagonals. Thus, the square pixel constraint may be interpreted as a pair of orthogonal line constraints. The diagonal vanishing points of a square pixel are $(1,1,0)^\top$ and $(-1,1,0)^\top$. The resulting orthogonality constraints lead to the square pixel constraints given in table 8.1.

Alternatively, the square pixel constraint can be interpreted in terms of two known points lying on the IAC. If the image plane has square pixels, then it has a Euclidean coordinate system and the circular points have known coordinates $(1,\pm i,0)^\top$. It may be verified that the two square pixel equations are equivalent to $(1,\pm i,0)\boldsymbol{\omega}(1,\pm i,0)^\top = 0$.

This is the most important geometric equivalence. In essence an image plane with square pixels acts as a metric plane in the scene. A square pixel image plane is equivalent to a metric plane imaged with a homography given by the identity. Indeed if the homography $\mathtt{H}$ in the "metric plane imaged with known homography" constraint of table 8.1 is replaced by the identity then the square pixel constraints are immediately obtained.

Thus, we see that all the constraints given in table 8.1 are derived either from known points lying on $\boldsymbol{\omega}$, or from pairs of points that are conjugate with respect to $\boldsymbol{\omega}$. Determining $\boldsymbol{\omega}$ may therefore be viewed as a conic fitting problem, given points on the conic and conjugate point pairs.

It is well to bear in mind that conic fitting is a delicate problem, often unstable ([Bookstein-79]) if the points are not well distributed on the conic. The same observation is true of the present problem, which we have seen is equivalent to conic fitting. The method given in algorithm 8.2 for finding the calibration from vanishing points amounts to minimization of algebraic error, and therefore does not give an optimal solution. For greater accuracy, the methods of chapter 4, for instance the Sampson error method of section 4.2.6(p98) should be used.

8.9 Single view reconstruction

As an application of the methods developed in this chapter we demonstrate now the 3D reconstruction of a texture mapped piecewise planar graphical model from a single image. The camera calibration methods of section 8.8 and the rectification method of example 8.28 may be combined to back project image regions to texture the planes of the model.

The method will be illustrated for the image of figure 8.26(a), where the scene contains three dominant and mutually orthogonal planes: the building facades on the left and right and the ground plane. The parallel line sets in three orthogonal directions de-

Samsung/Google, EX1010, p. 247

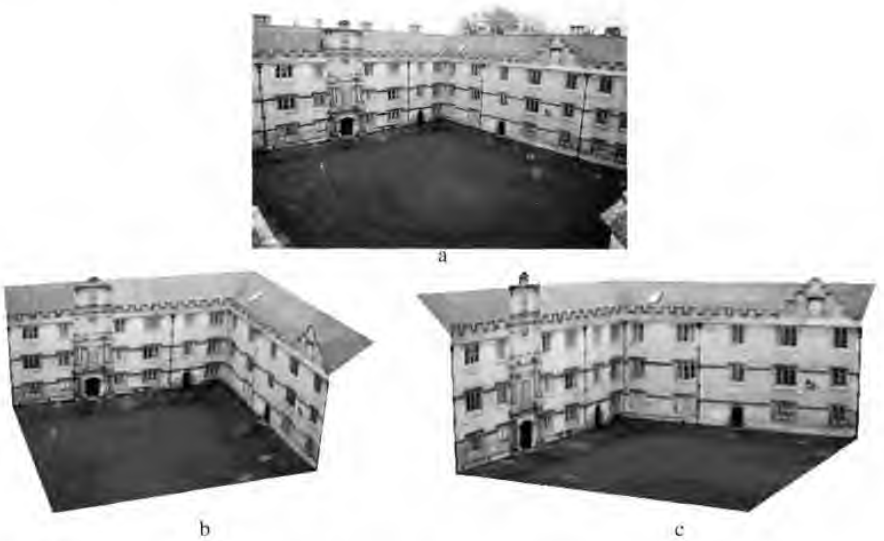


Fig. 8.26. **Single view reconstruction.** *(a) Original image of the Fellows quad, Merton College, Oxford. (b) (c) Views of the 3D model created from the single image. The vanishing line of the roof planes is computed from the repetition of the texture pattern.*

fine three vanishing points and together with the constraint of square pixels the camera calibration may be computed using the method described in section 8.8. From the vanishing lines of the three planes, likewise determined by the vanishing points, together with the computed ω, homographies may be computed to texture map the appropriate image regions onto the orthogonal planes of the model.

In more detail taking the left facade as a reference plane in figure 8.26(a), its correctly proportioned width and height are determined by the rectification. The right facade and ground planes define 3D planes orthogonal to the reference (we have assumed the orthogonality of the planes in computing the camera, so relative orientations are defined). Scaling of the right and ground planes is computed from the points common to the planes and this completes a three orthogonal plane model.

Having computed the calibration, the relative orientation of planes in the scene that are not orthogonal (such as the roof) can be computed if their vanishing lines can be found using (8.14–*p*218). Their relative positions and dimensions can be determined if the intersection of a pair of planes is visible in the image, so that there are points common to both planes. Relative size can be computed from the rectification of a distance between common points using the homographies of both planes. Views of the model, with texture mapped correctly to the planes, appear in figure 8.26(b) and (c).

Samsung/Google, EX1010, p. 248

8.10 The calibrating conic

The image of the absolute conic (IAC) is an imaginary conic in an image, and hence is not visible. Sometimes it is useful for visualization purposes to consider a different conic that is closely related to the calibration of the camera. Such a conic is the *calibrating conic*, which is the image of a cone with apex angle 45° and axis coinciding with the principal axis of the camera.

We wish to compute a formula for this cone in terms of the calibration matrix of the camera. Since the 45° cone moves with the camera, its image is clearly independent of the orientation and position of the camera. Thus, we may assume that the camera is located at the origin and oriented directly along the Z-axis. Thus, let the camera matrix be $\mathtt{P} = \mathtt{K}[\mathtt{I} \mid \mathtt{0}]$. Now, any point on the 45° cone satisfies $\mathtt{X}^2 + \mathtt{Y}^2 = \mathtt{Z}^2$. Points on this cone map to points on the conic

$$\mathtt{C} = \mathtt{K}^{-\mathsf{T}} \begin{bmatrix} 1 & & \\ & 1 & \\ & & -1 \end{bmatrix} \mathtt{K}^{-1} \tag{8.18}$$

as one easily verifies from result 8.6(p199). This conic will be referred to as the calibrating conic of the camera. For a calibrated camera with identity calibration matrix $\mathtt{K} = \mathtt{I}$, the calibrating conic is a unit circle centred at the origin (which is the principal point of the image). The conic of (8.18) is simply this unit circle transformed by an affine transformation according to the conic transformation rule of result 2.13(p37): $(\mathtt{C} \mapsto \mathtt{H}^{-\mathsf{T}}\mathtt{C}\mathtt{H}^{-1})$. Thus the calibrating conic of a camera with calibration matrix K is the affine transformation of a unit circle centred on the origin by the matrix K.

The calibration parameters are easily read from the calibrating conic. The principal point is the centre of the conic, and the scale factors and skew are easily identified, as in figure 8.27. In the case of zero skew, the calibrating conic has its principal axes aligned with the image coordinate axes. An example on a real image is shown in figure 8.29.

Example 8.29. Suppose $\mathtt{K} = \mathrm{diag}(f, f, 1)$, which is the calibration matrix for a camera of focal length f pixels, with no skew, square pixels, and image origin coincident with the principal point. Then from (8.18) the calibrating conic is $\mathtt{C} = \mathrm{diag}(1, 1, -f^2)$, which is a circle of radius f centred on the principal point. △

Orthogonality and the calibrating conic

A formula was given in (8.9–p210) for the angle between the rays corresponding to two image points. In particular the rays corresponding to two points x and x′ are perpendicular when $\mathbf{x}'^{\mathsf{T}}\boldsymbol{\omega}\mathbf{x} = 0$. As shown in figure 8.13(p212) this may be interpreted as the point x′ lying on the line $\boldsymbol{\omega}\mathbf{x}$, which is the polar of x with respect to the IAC.

We wish to carry out a similar analysis in terms of the calibrating conic. Writing $\mathtt{C} = \mathtt{K}^{-\mathsf{T}}\mathtt{D}\mathtt{K}^{-1}$, where $\mathtt{D} = \mathrm{diag}(1, 1, -1)$, we find

$$\mathtt{C} = (\mathtt{K}^{-\mathsf{T}}\mathtt{K}^{-1})(\mathtt{K}\mathtt{D}\mathtt{K}^{-1}) = \boldsymbol{\omega}\mathtt{S}$$

where $\mathtt{S} = \mathtt{K}\mathtt{D}\mathtt{K}^{-1}$. However, for any point x, the product Sx represents the reflection of

Samsung/Google, EX1010, p. 249

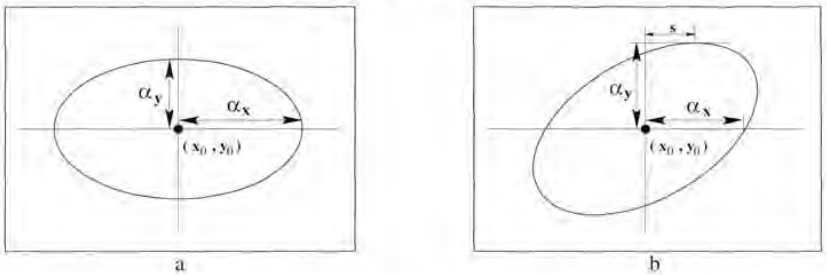


Fig. 8.27. **Reading the internal camera parameters** K **from the calibrating conic.** *(a) Skew s is zero. (b) Skew s is non-zero. The skew parameter of* K *(see (6.10–p157), is given by the x-coordinate of the highest point of the conic.*

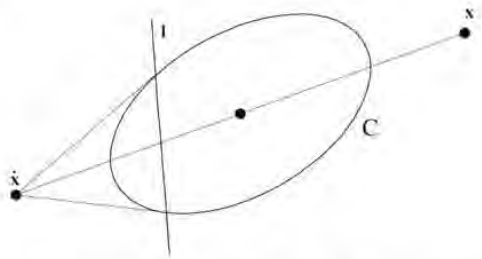


Fig. 8.28. *To construct the line perpendicular to the ray through image point* $\mathbf{x}$ *proceed as follows: (i) Reflect* $\mathbf{x}$ *through the centre of* C *to get point* $\dot{\mathbf{x}}$ *(i.e. at the same distance from the centre as* $\mathbf{x}$*). (ii) The desired line is the polar of* $\dot{\mathbf{x}}$.

the point $\mathbf{x}$ through the centre of the conic C, that is, the principal point of the camera. Representing this reflected point by $\dot{\mathbf{x}}$, one finds that

$$\mathbf{x}'^{\mathsf{T}}\omega\mathbf{x} = \mathbf{x}'^{\mathsf{T}}\mathtt{C}\dot{\mathbf{x}} \quad . \tag{8.19}$$

This leads to the following geometric result:

Result 8.30. *The line in an image corresponding to the plane perpendicular to a ray through image point* $\mathbf{x}$ *is the polar* $\mathtt{C}\dot{\mathbf{x}}$ *of the reflected point* $\dot{\mathbf{x}}$ *with respect to the calibrating conic.*

This construction is illustrated in figure 8.28.

Example 8.31. The calibrating conic given three orthogonal vanishing points
The calibrating conic can be drawn directly for the example of figure 8.22. Again assume there is no skew and square pixels, then the calibrating conic is a circle. Now given three mutually perpendicular vanishing points, one can find the calibrating conic by direct geometric construction as shown in figure 8.29.

(i) First, construct the triangle with vertices the three vanishing points $\mathbf{v}_1$, $\mathbf{v}_2$ and $\mathbf{v}_3$.
(ii) The centre of C is the orthocentre of the triangle.

Samsung/Google, EX1010, p. 250

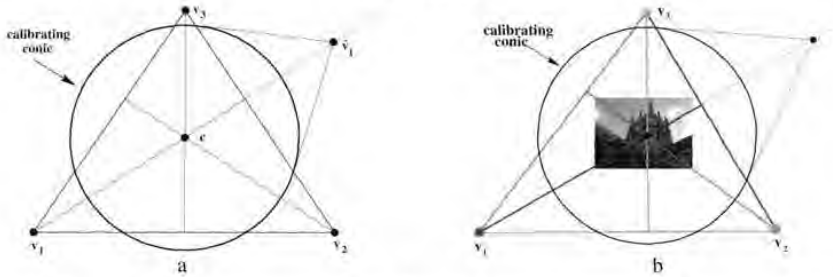


Fig. 8.29. *The calibrating conic computed from three orthogonal vanishing points. (a) The geometric construction. (b) The calibrating conic for the image of figure 8.22.*

(iii) Reflect one of the vanishing points (say $\mathbf{v}_1$) in the centre to get $\dot{\mathbf{v}}_1$.
(iv) The radius of $\mathtt{C}$ is determined by the condition that the polar of $\dot{\mathbf{v}}_1$ is the line passing though $\mathbf{v}_2$ and $\mathbf{v}_3$.

△

8.11 Closure

8.11.1 The literature

Faugeras and Mourrain [Faugeras-95a], and Faugeras and Papadopoulo [Faugeras-97] develop the projection of lines using Plücker coordinates. Koenderink [Koenderink-84, Koenderink-90], and Giblin and Weiss [Giblin-87] give many properties of the contour generator and apparent contour, and their relation to the differential geometry of surfaces.

[Kanatani-92] gives an alternative, calibrated, treatment of vanishing points and lines, and of the result that images acquired by cameras with the same centre are related by a planar homography. Mundy and Zisserman [Mundy-92] showed this result geometrically, and [Hartley-94a] gave a simple algebraic derivation based on camera projection matrices. [Faugeras-92b] introduced the projective (reduced) camera matrix. The link between the image of the absolute conic and camera calibration was first given in [Faugeras-92a].

The computation of panoramic mosaics is described in [Capel-98, Sawhney-98, Szeliski-97]. The ML method of computing vanishing points is given in Liebowitz & Zisserman [Liebowitz-98]. Applications of automatic vanishing line estimation from coplanar equally spaced lines are given in [Schaffalitzky-00b] and also [Se-00]. Affine 3D measurements from a single view is described in [Criminisi-00, Proesmans-98].

The result that $\mathtt{K}$ may be computed from multiple scene planes on which metric structure (such as a square) is known was given in [Liebowitz-98]. Algorithms for this computation are given in [Liebowitz-99a, Sturm-99c, Zhang-00]. The advantage in using ω, rather than ω^*, when imposing the skew zero constraint was first noted in [Armstrong-96b]. The method of internal calibration using three vanishing points for orthogonal directions was given by Caprile and Torre [Caprile-90], though there

Samsung/Google, EX1010, p. 251

is an earlier reference to this result in the photogrammetry literature [Gracie-68]. A simple formula for the focal length in this case is given in [Cipolla-99, Hartley-02b]. A discussion of the degeneracies that arise when combining multiple constraints is given in [Liebowitz-99b, Liebowitz-01]. Single view reconstruction is investigated in [Criminisi-99a, Criminisi-01, Horry-97, Liebowitz-99a, Sturm-99a].

8.11.2 Notes and exercises

(i) **Homography from a world plane**. Suppose $\mathtt{H}$ is computed (e.g. from the correspondence between four or more known world points and their images) and $\mathtt{K}$ known, then the pose of the camera $\{\mathtt{R}, \mathbf{t}\}$ may be computed from the camera matrix $[\mathbf{r}_1, \mathbf{r}_2, \mathbf{r}_1 \times \mathbf{r}_2, \mathbf{t}]$, where

$$[\mathbf{r}_1, \mathbf{r}_2, \mathbf{t}] = \pm \mathtt{K}^{-1}\mathtt{H} / \|\mathtt{K}^{-1}\mathtt{H}\|.$$

Note that there is a two-fold ambiguity. This result follows from (8.1–$p196$) which gives the homography between a world plane and calibrated camera $\mathtt{P} = \mathtt{K}[\mathtt{R} \mid \mathbf{t}]$.

Show that the homography $\mathbf{x} = \mathtt{H}\tilde{\mathbf{X}}$ between points on a world plane $(\mathbf{n}^\top, d)^\top$ and the image may be expressed as $\mathtt{H} = \mathtt{K}(\mathtt{R} - \mathbf{t}\mathbf{n}^\top/d)$. The points on the plane have coordinates $\tilde{\mathbf{X}} = (\mathtt{X}, \mathtt{Y}, \mathtt{Z})^\top$.

(ii) **Line projection.**

(a) Show that any line containing the camera centre lies in the null-space of the map (8.2–$p198$), i.e. it is projected to the line $\mathbf{l} = \mathbf{0}$.

(b) Show that the line $\mathcal{L} = \mathcal{P}^\top \mathbf{x}$ in $\mathbb{P}^3$ is the ray through the image point $\mathbf{x}$ and the camera centre. Hint: start from result 3.5($p72$), and show that the camera centre $\mathbf{C}$ lies on $\mathcal{L}$.

(c) What is the geometric interpretation of the columns of $\mathcal{P}$?

(iii) **Contour generator of a quadric.** The contour generator Γ of a quadric consists of the set of points $\mathbf{X}$ on $\mathtt{Q}$ for which the tangent planes contain the camera centre, $\mathbf{C}$. The tangent plane at a point $\mathbf{X}$ on $\mathtt{Q}$ is given by $\boldsymbol{\pi} = \mathtt{Q}\mathbf{X}$, and the condition that $\mathbf{C}$ is on $\boldsymbol{\pi}$ is $\mathbf{C}^\top\boldsymbol{\pi} = \mathbf{C}^\top\mathtt{Q}\mathbf{X} = 0$. Thus points $\mathbf{X}$ on Γ satisfy $\mathbf{C}^\top\mathtt{Q}\mathbf{X} = 0$, and thus lie on the *plane* $\boldsymbol{\pi}_\Gamma = \mathtt{Q}\mathbf{C}$ since $\boldsymbol{\pi}_\Gamma^\top\mathbf{X} = \mathbf{C}^\top\mathtt{Q}\mathbf{X} = 0$. This shows that the contour generator of a quadric is a plane curve and furthermore, since $\boldsymbol{\pi}_\Gamma = \mathtt{Q}\mathbf{C}$, that the plane of Γ is the polar plane of the camera centre with respect to the quadric.

(iv) **Apparent contour of an algebraic surface.** Show that the apparent contour of a homogeneous algebraic surface of degree n is a curve of degree $n(n-1)$. For example, if $n = 2$ then the surface is a quadric and the apparent contour a conic. Hint: write the surface as $F(\mathtt{X}, \mathtt{Y}, \mathtt{Z}, \mathtt{W}) = 0$, then the tangent plane contains the camera centre $\mathbf{C}$ if

$$C_\mathtt{X}\frac{\partial F}{\partial \mathtt{X}} + C_\mathtt{Y}\frac{\partial F}{\partial \mathtt{Y}} + C_\mathtt{Z}\frac{\partial F}{\partial \mathtt{Z}} + C_\mathtt{W}\frac{\partial F}{\partial \mathtt{W}} = 0$$

which is a surface of a degree $n - 1$.

Samsung/Google, EX1010, p. 252

(v) **Rotation axis vanishing point for $\mathtt{H} = \mathtt{KRK}^{-1}$.** The homography of a conjugate rotation $\mathtt{H} = \mathtt{KRK}^{-1}$ has an eigenvector $\mathtt{Ka}$, where $\mathbf{a}$ is the direction of the rotation axis, since $\mathtt{HKa} = \mathtt{KRa} = \mathtt{Ka}$. The last equality follows because $\mathtt{Ra} = 1\mathbf{a}$, i.e. $\mathbf{a}$ is the unit eigenvector of $\mathtt{R}$. It follows that (a) $\mathtt{Ka}$ is a fixed point under the homography $\mathtt{H}$; and (b) from result 8.20(p215) $\mathbf{v} = \mathtt{Ka}$ is the vanishing point of the rotation axis.

(vi) **Synthetic rotations.** Suppose, as in example 8.12(p205), that a homography is estimated between two images related by a pure rotation about the camera centre. Then the estimated homography will be a conjugate rotation, so that $\mathtt{H} = \mathtt{KR}(\theta)\mathtt{K}^{-1}$ (though $\mathtt{K}$ and $\mathtt{R}$ are unknown). However, $\mathtt{H}^2$ applied to the first image generates the image that would have been obtained by rotating the camera about the same axis through twice the angle, since $\mathtt{H}^2 = \mathtt{KR}^2\mathtt{K}^{-1} = \mathtt{KR}(2\theta)\mathtt{K}^{-1}$.

More generally we may write $\mathtt{H}^\lambda$ to represent a rotation through any fractional angle $\lambda\theta$. To make sense of $\mathtt{H}^\lambda$, observe that the eigenvalue decomposition of $\mathtt{H}$ is $\mathtt{H}(\theta) = \mathtt{U}\,\mathrm{diag}(1, e^{i\theta}, e^{-i\theta})\,\mathtt{U}^{-1}$, and both θ and $\mathtt{U}$ may be computed from the estimated $\mathtt{H}$. Then

$$\mathtt{H}^\lambda = \mathtt{U}\,\mathrm{diag}(1, e^{i\lambda\theta}, e^{-i\lambda\theta})\,\mathtt{U}^{-1} = \mathtt{KR}(\lambda\theta)\mathtt{K}^{-1}.$$

which is the conjugate of a rotation through angle $\lambda\theta$. Writing ϕ instead of $\lambda\theta$, we may use this homography to generate synthetic images rotated through any angle ϕ. The images are interpolated between the original images (if $0 < \phi < \theta$), or extrapolated (if $\phi > \theta$).

(vii) Show that the imaged circular points of a perspectively imaged plane may be computed if any of the following are on the plane: (i) a square grid; (ii) two rectangles arranged such that the sides of one rectangle are not parallel to the sides of the other; (iii) two circles of equal radius; (iv) two circles of unequal radius.

(viii) Show that in the case of zero skew, $\boldsymbol{\omega}$ is the conic

$$\left(\frac{x - x_0}{\alpha_x}\right)^2 + \left(\frac{y - y_0}{\alpha_y}\right)^2 + 1 = 0$$

which may be interpreted as an ellipse aligned with the axes, centred on the principal point, and with axes of length $i\alpha_x$ and $i\alpha_y$ in the x and y directions respectively.

(ix) If the camera calibration $\mathtt{K}$ and the vanishing line $\mathbf{l}$ of a scene plane are known then the scene plane can be metric rectified by a homography corresponding to a synthetic rotation $\mathtt{H} = \mathtt{KRK}^{-1}$ that maps $\mathbf{l}$ to $\mathbf{l}_\infty$, i.e. it is required that $\mathtt{H}^{-\mathsf{T}}\mathbf{l} = (0, 0, 1)^\mathsf{T}$. This condition arises because if the plane is rotated such that its vanishing line is $\mathbf{l}_\infty$ then it is fronto-parallel. Show that $\mathtt{H}^{-\mathsf{T}}\mathbf{l} = (0, 0, 1)^\mathsf{T}$ is equivalent to $\mathtt{Rn} = (0, 0, 1)^\mathsf{T}$, where $\mathbf{n} = \mathtt{K}^\mathsf{T}\mathbf{l}$ is the normal to the scene plane. This is the condition that the scene normal is rotated to lie along the camera $\mathtt{Z}$ axis. Note the rotation is not uniquely defined since a rotation about the plane's

Samsung/Google, EX1010, p. 253

normal does not affect its metric rectification. However, the last row of $\mathtt{R}$ equals $\mathbf{n}$, so that $\mathtt{R} = [\mathbf{r}_1, \mathbf{r}_2, \mathbf{n}]^\top$ where $\mathbf{n}, \mathbf{r}_1$ and $\mathbf{r}_2$ are a triad of orthonormal vectors.

(x) Show that the angle between two planes with vanishing lines $\mathbf{l}_1$ and $\mathbf{l}_2$ is

$$\cos\theta = \frac{\mathbf{l}_1^\top \boldsymbol{\omega}^* \mathbf{l}_2}{\sqrt{\mathbf{l}_1^\top \boldsymbol{\omega}^* \mathbf{l}_1}\sqrt{\mathbf{l}_2^\top \boldsymbol{\omega}^* \mathbf{l}_2}}.$$

(xi) Derive (8.15–*p*218). Hint, the line $\mathbf{l}$ lies in the pencil defined by $\mathbf{l}_1$ and $\mathbf{l}_2$, so it can be expressed as $\mathbf{l} = \alpha\mathbf{l}_1 + \beta\mathbf{l}_2$. Then use the relations $\mathbf{l}_n = \mathbf{l}_0 + n\mathbf{l}$ for $n = 1, 2$ to solve for α and β.

(xii) For the case of vanishing points arising from three orthogonal directions, and for an image with square pixels, show algebraically that the principal point is the orthocentre of the triangle with vertices the vanishing points. Hint: suppose the vanishing point at one vertex of the triangle is $\mathbf{v}$ and the line of the opposite side (through the other two vanishing points) is $\mathbf{l}$. Then from (8.17–*p*219) $\mathbf{v} = \boldsymbol{\omega}^*\mathbf{l}$ since $\mathbf{v}$ and $\mathbf{l}$ arise from an orthogonal line and plane respectively. Show that the principal point lies on the line from $\mathbf{v}$ to $\mathbf{l}$ which is perpendicular *in the image* to $\mathbf{l}$. Since this result is true for any vertex the principal point is the orthocentre of the triangle.

(xiii) Show that the vanishing points of an orthogonal triad of directions are the vertices of a self-polar triangle [Springer-64] with respect to $\boldsymbol{\omega}$.

(xiv) If a camera has square pixels, then the apparent contour of a sphere centred on the principal axis is a circle. If the sphere is translated parallel to the image plane, then the apparent contour deforms from a circle to an ellipse with the principal point on its major axis.

(a) How can this observation be used as a method of internal parameter calibration?

(b) Show by a geometric argument that the aspect ratio of the ellipse does not depend on the distance of the sphere from the camera.

If the sphere is now translated parallel to the principal axis the apparent contour can deform to a hyperbola, but only one branch of the hyperbola is imaged. Why is this?

(xv) Show that for a general camera the apparent contour of a sphere is related to the IAC as:

$$\boldsymbol{\omega} = \mathtt{C} + \mathbf{v}\mathbf{v}^\top$$

where $\mathtt{C}$ is the conic outline of the imaged sphere, and $\mathbf{v}$ is a 3-vector that depends on the position of the sphere. A proof is given in [Agrawal-03]. Note this relation places two constraints on $\boldsymbol{\omega}$, so that in principle $\boldsymbol{\omega}$, and hence the calibration $\mathtt{K}$, may be computed from three images of a sphere. However, in practice this is not a well conditioned method for computing $\mathtt{K}$ because the deviation of the sphere's outline from a circle is small.

Samsung/Google, EX1010, p. 254

Part II

Two-View Geometry



The Birth of Venus (detail), c. 1485 (tempera on canvas) by Sandro Botticelli (1444/5-1510)
Galleria degli Uffizi, Florence, Italy/Bridgeman Art Library

Samsung/Google, EX1010, p. 255

Outline

This part of the book covers the geometry of two perspective views. These views may be acquired simultaneously as in a stereo rig, or acquired sequentially, for example by a camera moving relative to the scene. These two situations are geometrically equivalent and will not be differentiated here. Each view has an associated camera matrix, $\mathtt{P}, \mathtt{P}'$, where $'$ indicates entities associated with the second view, and a 3-space point $\mathbf{X}$ is imaged as $\mathbf{x} = \mathtt{P}\mathbf{X}$ in the first view, and $\mathbf{x}' = \mathtt{P}'\mathbf{X}$ in the second. Image points $\mathbf{x}$ and $\mathbf{x}'$ *correspond* because they are the image of the same 3-space point. There are three questions that will be addressed:

(i) **Correspondence geometry.** Given an image point $\mathbf{x}$ in the first view, how does this constrain the position of the corresponding point $\mathbf{x}'$ in the second view?

(ii) **Camera geometry (motion).** Given a set of corresponding image points $\{\mathbf{x}_i \leftrightarrow \mathbf{x}_i'\}, i = 1, \ldots, n$, what are the cameras $\mathtt{P}$ and $\mathtt{P}'$ for the two views?

(iii) **Scene geometry (structure).** Given corresponding image points $\mathbf{x} \leftrightarrow \mathbf{x}'$ and cameras $\mathtt{P}, \mathtt{P}'$, what is the position of (their pre-image) $\mathbf{X}$ in 3-space?

Chapter 9 describes the *epipolar geometry* of two views, and directly answers the first question: a point in one view defines an epipolar line in the other view on which the corresponding point lies. The epipolar geometry depends only on the cameras – their relative position and their internal parameters. It does *not* depend at all on the scene structure. The epipolar geometry is represented by a 3×3 matrix called the *fundamental matrix* $\mathtt{F}$. The anatomy of the fundamental matrix is described, and its computation from camera matrices $\mathtt{P}$ and $\mathtt{P}'$ given. It is then shown that $\mathtt{P}$ and $\mathtt{P}'$ may be computed from $\mathtt{F}$ up to a projective ambiguity of 3-space.

Chapter 10 describes one of the most important results in uncalibrated multiple view geometry – a *reconstruction* of both cameras and scene structure can be computed from image point correspondences alone; no other information is required. This answers both the second and third questions simultaneously. The reconstruction obtained from point correspondences alone is up to a projective ambiguity of 3-space, and this ambiguity can be resolved by supplying well defined additional information on the cameras or scene. In this manner an affine or metric reconstruction may be computed from uncalibrated images. The following two chapters then fill in the details and numerical algorithms for computing this reconstruction.

Chapter 11 describes methods for computing $\mathtt{F}$ from a set of corresponding image points $\{\mathbf{x}_i \leftrightarrow \mathbf{x}_i'\}$, even though the structure (3D pre-image $\mathbf{X}_i$) of these points is unknown and the camera matrices are unknown. The cameras $\mathtt{P}$ and $\mathtt{P}'$ may then be determined, up to a projective ambiguity, from the computed $\mathtt{F}$.

Chapter 12 then describes the computation of scene structure by *triangulation* given the cameras and corresponding image points – the point $\mathbf{X}$ in 3-space is computed as the intersection of rays back-projected from the corresponding points $\mathbf{x}$ and $\mathbf{x}'$ via their associated cameras $\mathtt{P}, \mathtt{P}'$. Similarly, the 3D position of other geometric entities, such as lines or conics, may also be computed given their image correspondences.

Chapter 13 covers the two-view geometry of planes. It provides an alternative answer to the first question: if scene points lie on a plane, then once the geometry of this plane is computed, the image $\mathbf{x}$ of a point in one image determines the position of $\mathbf{x}'$ in the other image. The points are related by a plane projective transformation. This chapter also describes a particularly important projective transformation between views – the *infinite homography*, which is the transformation arising from the plane at infinity.

Chapter 14 describes two-view geometry in the specialized case that the two cameras $\mathtt{P}$ and $\mathtt{P}'$ are affine. This case has a number of simplifications over the general projective case, and provides a very good approximation in many practical situations.

Samsung/Google, EX1010, p. 256

9

Epipolar Geometry and the Fundamental Matrix

The epipolar geometry is the intrinsic projective geometry between two views. It is independent of scene structure, and only depends on the cameras' internal parameters and relative pose.

The fundamental matrix F encapsulates this intrinsic geometry. It is a 3×3 matrix of rank 2. If a point in 3-space $\mathbf{X}$ is imaged as $\mathbf{x}$ in the first view, and $\mathbf{x}'$ in the second, then the image points satisfy the relation $\mathbf{x}'^{\mathsf{T}}\mathbf{F}\mathbf{x} = 0$.

We will first describe epipolar geometry, and derive the fundamental matrix. The properties of the fundamental matrix are then elucidated, both for general motion of the camera between the views, and for several commonly occurring special motions. It is next shown that the cameras can be retrieved from F up to a projective transformation of 3-space. This result is the basis for the projective reconstruction theorem given in chapter 10. Finally, if the camera internal calibration is known, it is shown that the Euclidean motion of the cameras between views may be computed from the fundamental matrix up to a finite number of ambiguities.

The fundamental matrix is independent of scene structure. However, it can be computed from correspondences of imaged scene points alone, without requiring knowledge of the cameras' internal parameters or relative pose. This computation is described in chapter 11.

9.1 Epipolar geometry

The epipolar geometry between two views is essentially the geometry of the intersection of the image planes with the pencil of planes having the baseline as axis (the baseline is the line joining the camera centres). This geometry is usually motivated by considering the search for corresponding points in stereo matching, and we will start from that objective here.

Suppose a point $\mathbf{X}$ in 3-space is imaged in two views, at $\mathbf{x}$ in the first, and $\mathbf{x}'$ in the second. What is the relation between the corresponding image points $\mathbf{x}$ and $\mathbf{x}'$? As shown in figure 9.1a the image points $\mathbf{x}$ and $\mathbf{x}'$, space point $\mathbf{X}$, and camera centres are coplanar. Denote this plane as π. Clearly, the rays back-projected from $\mathbf{x}$ and $\mathbf{x}'$ intersect at $\mathbf{X}$, and the rays are coplanar, lying in π. It is this latter property that is of most significance in searching for a correspondence.

Samsung/Google, EX1010, p. 257

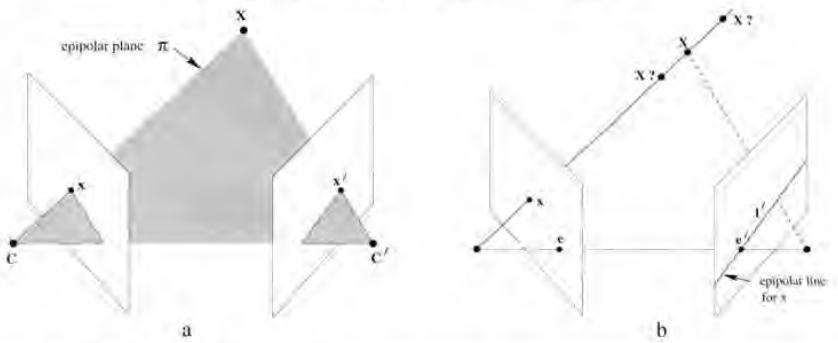


Fig. 9.1. **Point correspondence geometry.** *(a) The two cameras are indicated by their centres* $\mathbf{C}$ *and* $\mathbf{C}'$ *and image planes. The camera centres, 3-space point* $\mathbf{X}$*, and its images* $\mathbf{x}$ *and* $\mathbf{x}'$ *lie in a common plane* π*. (b) An image point* $\mathbf{x}$ *back-projects to a ray in 3-space defined by the first camera centre,* $\mathbf{C}$*, and* $\mathbf{x}$*. This ray is imaged as a line* $\mathbf{l}'$ *in the second view. The 3-space point* $\mathbf{X}$ *which projects to* $\mathbf{x}$ *must lie on this ray, so the image of* $\mathbf{X}$ *in the second view must lie on* $\mathbf{l}'$*.*

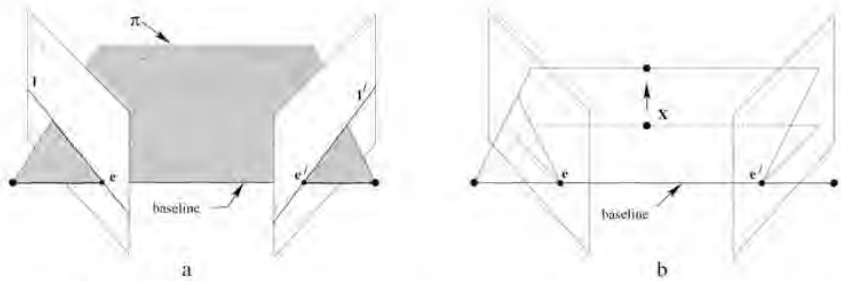


Fig. 9.2. **Epipolar geometry.** *(a) The camera baseline intersects each image plane at the epipoles* $\mathbf{e}$ *and* $\mathbf{e}'$*. Any plane* π *containing the baseline is an epipolar plane, and intersects the image planes in corresponding epipolar lines* $\mathbf{l}$ *and* $\mathbf{l}'$*. (b) As the position of the 3D point* $\mathbf{X}$ *varies, the epipolar planes "rotate" about the baseline. This family of planes is known as an epipolar pencil. All epipolar lines intersect at the epipole.*

Supposing now that we know only $\mathbf{x}$, we may ask how the corresponding point $\mathbf{x}'$ is constrained. The plane π is determined by the baseline and the ray defined by $\mathbf{x}$. From above we know that the ray corresponding to the (unknown) point $\mathbf{x}'$ lies in π, hence the point $\mathbf{x}'$ lies on the line of intersection $\mathbf{l}'$ of π with the second image plane. This line $\mathbf{l}'$ is the image in the second view of the ray back-projected from $\mathbf{x}$. It is the *epipolar line* corresponding to $\mathbf{x}$. In terms of a stereo correspondence algorithm the benefit is that the search for the point corresponding to $\mathbf{x}$ need not cover the entire image plane but can be restricted to the line $\mathbf{l}'$.

The geometric entities involved in epipolar geometry are illustrated in figure 9.2. The terminology is

- The **epipole** is the *point* of intersection of the line joining the camera centres (the baseline) with the image plane. Equivalently, the epipole is the image in one view

Samsung/Google, EX1010, p. 258

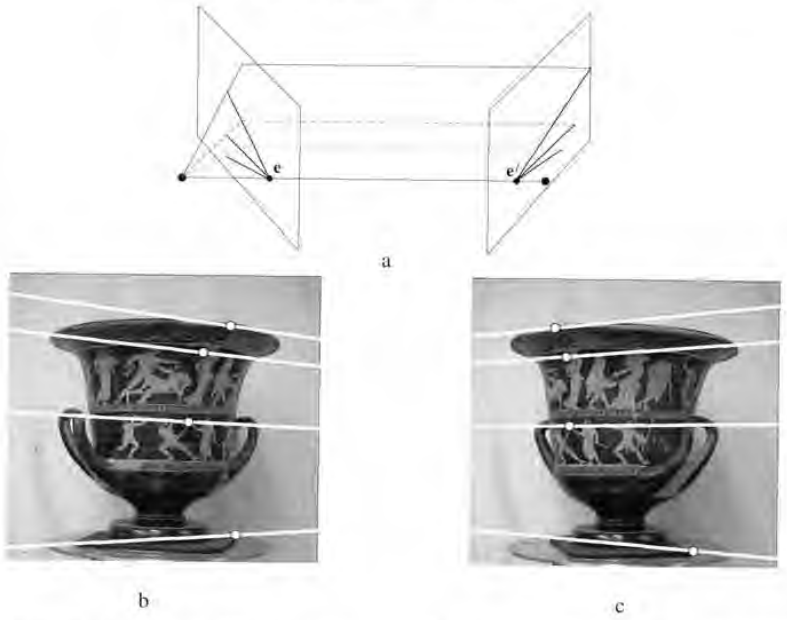


Fig. 9.3. **Converging cameras.** *(a) Epipolar geometry for converging cameras. (b) and (c) A pair of images with superimposed corresponding points and their epipolar lines (in white). The motion between the views is a translation and rotation. In each image, the direction of the other camera may be inferred from the intersection of the pencil of epipolar lines. In this case, both epipoles lie outside of the visible image.*

of the camera centre of the other view. It is also the vanishing point of the baseline (translation) direction.

- An **epipolar plane** is a plane containing the baseline. There is a one-parameter family (a pencil) of epipolar planes.
- An **epipolar line** is the intersection of an epipolar plane with the image plane. All epipolar lines intersect at the epipole. An epipolar plane intersects the left and right image planes in epipolar lines, and defines the correspondence between the lines.

Examples of epipolar geometry are given in figure 9.3 and figure 9.4. The epipolar geometry of these image pairs, and indeed all the examples of this chapter, is computed directly from the images as described in section 11.6(p290).

9.2 The fundamental matrix F

The fundamental matrix is the algebraic representation of epipolar geometry. In the following we derive the fundamental matrix from the mapping between a point and its epipolar line, and then specify the properties of the matrix.

Given a pair of images, it was seen in figure 9.1 that to each point $\mathbf{x}$ in one image, there exists a corresponding epipolar line $\mathbf{l}'$ in the other image. Any point $\mathbf{x}'$ in the second image matching the point $\mathbf{x}$ must lie on the epipolar line $\mathbf{l}'$. The epipolar line

Samsung/Google, EX1010, p. 259

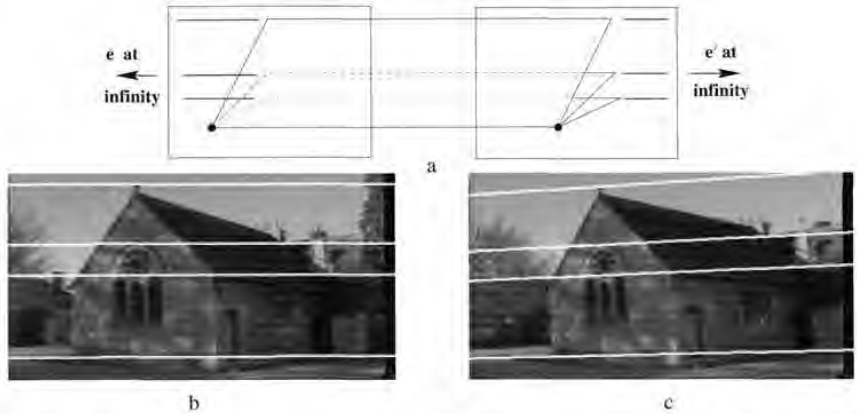


Fig. 9.4. **Motion parallel to the image plane.** *In the case of a special motion where the translation is parallel to the image plane, and the rotation axis is perpendicular to the image plane, the intersection of the baseline with the image plane is at infinity. Consequently the epipoles are at infinity, and epipolar lines are parallel. (a) Epipolar geometry for motion parallel to the image plane. (b) and (c) a pair of images for which the motion between views is (approximately) a translation parallel to the x-axis, with no rotation. Four corresponding epipolar lines are superimposed in white. Note that corresponding points lie on corresponding epipolar lines.*

is the projection in the second image of the ray from the point $\mathbf{x}$ through the camera centre $\mathbf{C}$ of the first camera. Thus, there is a map

$$\mathbf{x} \mapsto \mathbf{l}'$$

from a point in one image to its corresponding epipolar line in the other image. It is the nature of this map that will now be explored. It will turn out that this mapping is a (singular) *correlation*, that is a projective mapping from points to lines, which is represented by a matrix $\mathbf{F}$, the fundamental matrix.

9.2.1 Geometric derivation

We begin with a geometric derivation of the fundamental matrix. The mapping from a point in one image to a corresponding epipolar line in the other image may be decomposed into two steps. In the first step, the point $\mathbf{x}$ is mapped to some point $\mathbf{x}'$ in the other image lying on the epipolar line $\mathbf{l}'$. This point $\mathbf{x}'$ is a potential match for the point $\mathbf{x}$. In the second step, the epipolar line $\mathbf{l}'$ is obtained as the line joining $\mathbf{x}'$ to the epipole $\mathbf{e}'$.

Step 1: Point transfer via a plane. Refer to figure 9.5. Consider a plane π in space not passing through either of the two camera centres. The ray through the first camera centre corresponding to the point $\mathbf{x}$ meets the plane π in a point $\mathbf{X}$. This point $\mathbf{X}$ is then projected to a point $\mathbf{x}'$ in the second image. This procedure is known as transfer via the plane π. Since $\mathbf{X}$ lies on the ray corresponding to $\mathbf{x}$, the projected point $\mathbf{x}'$ must lie on the epipolar line $\mathbf{l}'$ corresponding to the image of this ray, as illustrated in

Samsung/Google, EX1010, p. 260

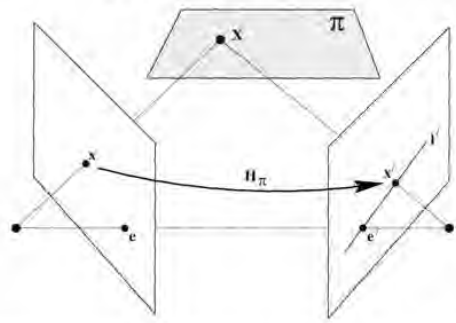


Fig. 9.5. *A point* $\mathbf{x}$ *in one image is transferred via the plane* π *to a matching point* $\mathbf{x}'$ *in the second image. The epipolar line through* $\mathbf{x}'$ *is obtained by joining* $\mathbf{x}'$ *to the epipole* $\mathbf{e}'$. *In symbols one may write* $\mathbf{x}' = \mathtt{H}_\pi\mathbf{x}$ *and* $\mathbf{l}' = [\mathbf{e}']_\times\mathbf{x}' = [\mathbf{e}']_\times\mathtt{H}_\pi\mathbf{x} = \mathtt{F}\mathbf{x}$ *where* $\mathtt{F} = [\mathbf{e}']_\times\mathtt{H}_\pi$ *is the fundamental matrix.*

figure 9.1b. The points $\mathbf{x}$ and $\mathbf{x}'$ are both images of the 3D point $\mathbf{X}$ lying on a plane. The set of all such points $\mathbf{x}_i$ in the first image and the corresponding points $\mathbf{x}'_i$ in the second image are projectively equivalent, since they are each projectively equivalent to the planar point set $\mathbf{X}_i$. Thus there is a 2D homography $\mathtt{H}_\pi$ mapping each $\mathbf{x}_i$ to $\mathbf{x}'_i$.

Step 2: Constructing the epipolar line. Given the point $\mathbf{x}'$ the epipolar line $\mathbf{l}'$ passing through $\mathbf{x}'$ and the epipole $\mathbf{e}'$ can be written as $\mathbf{l}' = \mathbf{e}' \times \mathbf{x}' = [\mathbf{e}']_\times\mathbf{x}'$ (the notation $[\mathbf{e}']_\times$ is defined in (A4.5–*p*581)). Since $\mathbf{x}'$ may be written as $\mathbf{x}' = \mathtt{H}_\pi\mathbf{x}$, we have

$$\mathbf{l}' = [\mathbf{e}']_\times\mathtt{H}_\pi\mathbf{x} = \mathtt{F}\mathbf{x}$$

where we define $\mathtt{F} = [\mathbf{e}']_\times\mathtt{H}_\pi$, the fundamental matrix. This shows

Result 9.1. *The fundamental matrix* F *may be written as* $\mathtt{F} = [\mathbf{e}']_\times\mathtt{H}_\pi$, *where* $\mathtt{H}_\pi$ *is the transfer mapping from one image to another via any plane* π. *Furthermore, since* $[\mathbf{e}']_\times$ *has rank 2 and* $\mathtt{H}_\pi$ *rank 3,* F *is a matrix of rank 2.*

Geometrically, F represents a mapping from the 2-dimensional projective plane $\mathbb{P}^2$ of the first image to the pencil of epipolar lines through the epipole $\mathbf{e}'$. Thus, it represents a mapping from a 2-dimensional onto a 1-dimensional projective space, and hence must have rank 2.

Note, the geometric derivation above involves a scene plane π, but a plane is *not* required in order for F to exist. The plane is simply used here as a means of defining a point map from one image to another. The connection between the fundamental matrix and transfer of points from one image to another via a plane is dealt with in some depth in chapter 13.

9.2.2 Algebraic derivation

The form of the fundamental matrix in terms of the two camera projection matrices, $\mathtt{P}, \mathtt{P}'$, may be derived algebraically. The following formulation is due to Xu and Zhang [Xu-96].

Samsung/Google, EX1010, p. 261

The ray back-projected from $\mathbf{x}$ by $\mathtt{P}$ is obtained by solving $\mathtt{P}\mathbf{X} = \mathbf{x}$. The one-parameter family of solutions is of the form given by (6.13–p162) as

$$\mathbf{X}(\lambda) = \mathtt{P}^{+}\mathbf{x} + \lambda\mathbf{C}$$

where $\mathtt{P}^{+}$ is the pseudo-inverse of $\mathtt{P}$, i.e. $\mathtt{P}\mathtt{P}^{+} = \mathtt{I}$, and $\mathbf{C}$ its null-vector, namely the camera centre, defined by $\mathtt{P}\mathbf{C} = \mathbf{0}$. The ray is parametrized by the scalar λ. In particular two points on the ray are $\mathtt{P}^{+}\mathbf{x}$ (at $\lambda = 0$), and the first camera centre $\mathbf{C}$ (at $\lambda = \infty$). These two points are imaged by the second camera $\mathtt{P}'$ at $\mathtt{P}'\mathtt{P}^{+}\mathbf{x}$ and $\mathtt{P}'\mathbf{C}$ respectively in the second view. The epipolar line is the line joining these two projected points, namely $\mathbf{l}' = (\mathtt{P}'\mathbf{C}) \times (\mathtt{P}'\mathtt{P}^{+}\mathbf{x})$. The point $\mathtt{P}'\mathbf{C}$ is the epipole in the second image, namely the projection of the first camera centre, and may be denoted by $\mathbf{e}'$. Thus, $\mathbf{l}' = [\mathbf{e}']_{\times}(\mathtt{P}'\mathtt{P}^{+})\mathbf{x} = \mathtt{F}\mathbf{x}$, where $\mathtt{F}$ is the matrix

$$\mathtt{F} = [\mathbf{e}']_{\times}\mathtt{P}'\mathtt{P}^{+}. \tag{9.1}$$

This is essentially the same formula for the fundamental matrix as the one derived in the previous section, the homography $\mathtt{H}_{\pi}$ having the explicit form $\mathtt{H}_{\pi} = \mathtt{P}'\mathtt{P}^{+}$ in terms of the two camera matrices. Note that this derivation breaks down in the case where the two camera centres are the same for, in this case, $\mathbf{C}$ is the common camera centre of both $\mathtt{P}$ and $\mathtt{P}'$, and so $\mathtt{P}'\mathbf{C} = \mathbf{0}$. It follows that $\mathtt{F}$ defined in (9.1) is the zero matrix.

Example 9.2. Suppose the camera matrices are those of a calibrated stereo rig with the world origin at the first camera

$$\mathtt{P} = \mathtt{K}[\mathtt{I} \mid \mathbf{0}] \qquad \mathtt{P}' = \mathtt{K}'[\mathtt{R} \mid \mathbf{t}].$$

Then

$$\mathtt{P}^{+} = \begin{bmatrix} \mathtt{K}^{-1} \\ \mathbf{0}^{\mathsf{T}} \end{bmatrix} \qquad \mathbf{C} = \begin{pmatrix} \mathbf{0} \\ 1 \end{pmatrix}$$

and

$$\begin{aligned} \mathtt{F} &= [\mathtt{P}'\mathbf{C}]_{\times}\mathtt{P}'\mathtt{P}^{+} \\ &= [\mathtt{K}'\mathbf{t}]_{\times}\mathtt{K}'\mathtt{R}\mathtt{K}^{-1} = \mathtt{K}'^{-\mathsf{T}}[\mathbf{t}]_{\times}\mathtt{R}\mathtt{K}^{-1} = \mathtt{K}'^{-\mathsf{T}}\mathtt{R}[\mathtt{R}^{\mathsf{T}}\mathbf{t}]_{\times}\mathtt{K}^{-1} = \mathtt{K}'^{-\mathsf{T}}\mathtt{R}\mathtt{K}^{\mathsf{T}}[\mathtt{K}\mathtt{R}^{\mathsf{T}}\mathbf{t}]_{\times} \end{aligned} \tag{9.2}$$

where the various forms follow from result A4.3(p582). Note that the epipoles (defined as the image of the other camera centre) are

$$\mathbf{e} = \mathtt{P}\begin{pmatrix} -\mathtt{R}^{\mathsf{T}}\mathbf{t} \\ 1 \end{pmatrix} = \mathtt{K}\mathtt{R}^{\mathsf{T}}\mathbf{t} \qquad \mathbf{e}' = \mathtt{P}'\begin{pmatrix} \mathbf{0} \\ 1 \end{pmatrix} = \mathtt{K}'\mathbf{t}. \tag{9.3}$$

Thus we may write (9.2) as

$$\mathtt{F} = [\mathbf{e}']_{\times}\mathtt{K}'\mathtt{R}\mathtt{K}^{-1} = \mathtt{K}'^{-\mathsf{T}}[\mathbf{t}]_{\times}\mathtt{R}\mathtt{K}^{-1} = \mathtt{K}'^{-\mathsf{T}}\mathtt{R}[\mathtt{R}^{\mathsf{T}}\mathbf{t}]_{\times}\mathtt{K}^{-1} = \mathtt{K}'^{-\mathsf{T}}\mathtt{R}\mathtt{K}^{\mathsf{T}}[\mathbf{e}]_{\times}. \tag{9.4}$$

△

The expression for the fundamental matrix can be derived in many ways, and indeed will be derived again several times in this book. In particular, (17.3–p412) expresses $\mathtt{F}$ in terms of 4×4 determinants composed from rows of the camera matrices for each view.

Samsung/Google, EX1010, p. 262

9.2.3 Correspondence condition

Up to this point we have considered the map $\mathbf{x} \rightarrow \mathbf{l}'$ defined by $\mathtt{F}$. We may now state the most basic properties of the fundamental matrix.

Result 9.3. *The fundamental matrix satisfies the condition that for any pair of corresponding points* $\mathbf{x} \leftrightarrow \mathbf{x}'$ *in the two images*

$$\mathbf{x}'^{\mathsf{T}}\mathtt{F}\mathbf{x} = 0.$$

This is true, because if points $\mathbf{x}$ and $\mathbf{x}'$ correspond, then $\mathbf{x}'$ lies on the epipolar line $\mathbf{l}' = \mathtt{F}\mathbf{x}$ corresponding to the point $\mathbf{x}$. In other words $0 = \mathbf{x}'^{\mathsf{T}}\mathbf{l}' = \mathbf{x}'^{\mathsf{T}}\mathtt{F}\mathbf{x}$. Conversely, if image points satisfy the relation $\mathbf{x}'^{\mathsf{T}}\mathtt{F}\mathbf{x} = 0$ then the rays defined by these points are coplanar. This is a necessary condition for points to correspond.

The importance of the relation of result 9.3 is that it gives a way of characterizing the fundamental matrix without reference to the camera matrices, i.e. only in terms of corresponding image points. This enables $\mathtt{F}$ to be computed from image correspondences alone. We have seen from (9.1) that $\mathtt{F}$ may be computed from the two camera matrices, $\mathtt{P}, \mathtt{P}'$, and in particular that $\mathtt{F}$ is determined uniquely from the cameras, up to an overall scaling. However, we may now enquire how many correspondences are required to compute $\mathtt{F}$ from $\mathbf{x}'^{\mathsf{T}}\mathtt{F}\mathbf{x} = 0$, and the circumstances under which the matrix is uniquely defined by these correspondences. The details of this are postponed until chapter 11, where it will be seen that in general at least 7 correspondences are required to compute $\mathtt{F}$.

9.2.4 Properties of the fundamental matrix

Definition 9.4. Suppose we have two images acquired by cameras with non-coincident centres, then the **fundamental matrix** $\mathtt{F}$ is the unique 3×3 rank 2 homogeneous matrix which satisfies

$$\mathbf{x}'^{\mathsf{T}}\mathtt{F}\mathbf{x} = 0 \tag{9.5}$$

for all corresponding points $\mathbf{x} \leftrightarrow \mathbf{x}'$.

We now briefly list a number of properties of the fundamental matrix. The most important properties are also summarized in table 9.1.

(i) **Transpose**: If $\mathtt{F}$ is the fundamental matrix of the pair of cameras $(\mathtt{P}, \mathtt{P}')$, then $\mathtt{F}^{\mathsf{T}}$ is the fundamental matrix of the pair in the opposite order: $(\mathtt{P}', \mathtt{P})$.

(ii) **Epipolar lines**: For any point $\mathbf{x}$ in the first image, the corresponding epipolar line is $\mathbf{l}' = \mathtt{F}\mathbf{x}$. Similarly, $\mathbf{l} = \mathtt{F}^{\mathsf{T}}\mathbf{x}'$ represents the epipolar line corresponding to $\mathbf{x}'$ in the second image.

(iii) The **epipole**: for any point $\mathbf{x}$ (other than $\mathbf{e}$) the epipolar line $\mathbf{l}' = \mathtt{F}\mathbf{x}$ contains the epipole $\mathbf{e}'$. Thus $\mathbf{e}'$ satisfies $\mathbf{e}'^{\mathsf{T}}(\mathtt{F}\mathbf{x}) = (\mathbf{e}'^{\mathsf{T}}\mathtt{F})\mathbf{x} = 0$ for all $\mathbf{x}$. It follows that $\mathbf{e}'^{\mathsf{T}}\mathtt{F} = \mathbf{0}$, i.e. $\mathbf{e}'$ is the left null-vector of $\mathtt{F}$. Similarly $\mathtt{F}\mathbf{e} = \mathbf{0}$, i.e. $\mathbf{e}$ is the right null-vector of $\mathtt{F}$.

Samsung/Google, EX1010, p. 263

- $\mathtt{F}$ is a rank 2 homogeneous matrix with 7 degrees of freedom.
- **Point correspondence**: If $\mathbf{x}$ and $\mathbf{x}'$ are corresponding image points, then $\mathbf{x}'^{\mathsf{T}}\mathtt{F}\mathbf{x} = 0$.
- **Epipolar lines**:
  - $\mathbf{l}' = \mathtt{F}\mathbf{x}$ is the epipolar line corresponding to $\mathbf{x}$.
  - $\mathbf{l} = \mathtt{F}^{\mathsf{T}}\mathbf{x}'$ is the epipolar line corresponding to $\mathbf{x}'$.
- **Epipoles**:
  - $\mathtt{F}\mathbf{e} = \mathbf{0}$.
  - $\mathtt{F}^{\mathsf{T}}\mathbf{e}' = \mathbf{0}$.
- **Computation from camera matrices** $\mathtt{P}, \mathtt{P}'$:
  - General cameras, $\mathtt{F} = [\mathbf{e}']_\times \mathtt{P}'\mathtt{P}^+$, where $\mathtt{P}^+$ is the pseudo-inverse of $\mathtt{P}$, and $\mathbf{e}' = \mathtt{P}'\mathbf{C}$, with $\mathtt{P}\mathbf{C} = \mathbf{0}$.
  - Canonical cameras, $\mathtt{P} = [\mathtt{I} \mid \mathbf{0}]$, $\mathtt{P}' = [\mathtt{M} \mid \mathbf{m}]$, $\mathtt{F} = [\mathbf{e}']_\times \mathtt{M} = \mathtt{M}^{-\mathsf{T}}[\mathbf{e}]_\times$, where $\mathbf{e}' = \mathbf{m}$ and $\mathbf{e} = \mathtt{M}^{-1}\mathbf{m}$.
  - Cameras not at infinity $\mathtt{P} = \mathtt{K}[\mathtt{I} \mid \mathbf{0}]$, $\mathtt{P}' = \mathtt{K}'[\mathtt{R} \mid \mathbf{t}]$, $\mathtt{F} = \mathtt{K}'^{-\mathsf{T}}[\mathbf{t}]_\times \mathtt{R}\mathtt{K}^{-1} = [\mathtt{K}'\mathbf{t}]_\times \mathtt{K}'\mathtt{R}\mathtt{K}^{-1} = \mathtt{K}'^{-\mathsf{T}}\mathtt{R}\mathtt{K}^{\mathsf{T}}[\mathtt{K}\mathtt{R}^{\mathsf{T}}\mathbf{t}]_\times$.

Table 9.1. *Summary of fundamental matrix properties.*

(iv) $\mathtt{F}$ has seven degrees of freedom: a 3×3 homogeneous matrix has eight independent ratios (there are nine elements, and the common scaling is not significant); however, $\mathtt{F}$ also satisfies the constraint $\det \mathtt{F} = 0$ which removes one degree of freedom.

(v) $\mathtt{F}$ is a *correlation*, a projective map taking a point to a line (see definition 2.29-(*p*59)). In this case a point in the first image $\mathbf{x}$ defines a line in the second $\mathbf{l}' = \mathtt{F}\mathbf{x}$, which is the epipolar line of $\mathbf{x}$. If $\mathbf{l}$ and $\mathbf{l}'$ are corresponding epipolar lines (see figure 9.6a) then any point $\mathbf{x}$ on $\mathbf{l}$ is mapped to the same line $\mathbf{l}'$. This means there is no inverse mapping, and $\mathtt{F}$ is not of full rank. For this reason, $\mathtt{F}$ is not a proper correlation (which would be invertible).

9.2.5 The epipolar line homography

The set of epipolar lines in each of the images forms a pencil of lines passing through the epipole. Such a pencil of lines may be considered as a 1-dimensional projective space. It is clear from figure 9.6b that corresponding epipolar lines are perspectively related, so that there is a homography between the pencil of epipolar lines centred at $\mathbf{e}$ in the first view, and the pencil centred at $\mathbf{e}'$ in the second. A homography between two such 1-dimensional projective spaces has 3 degrees of freedom.

The degrees of freedom of the fundamental matrix can thus be counted as follows: 2 for $\mathbf{e}$, 2 for $\mathbf{e}'$, and 3 for the epipolar line homography which maps a line through $\mathbf{e}$ to a line through $\mathbf{e}'$. A geometric representation of this homography is given in section 9.4. Here we give an explicit formula for this mapping.

Samsung/Google, EX1010, p. 264

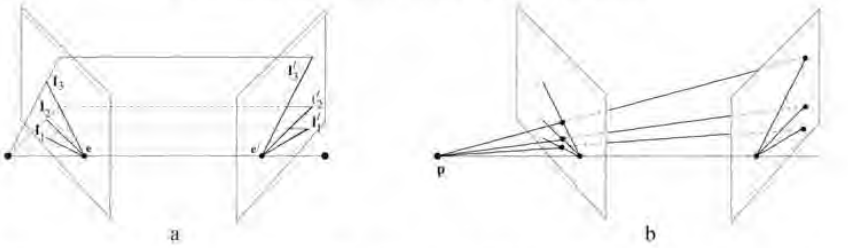


Fig. 9.6. **Epipolar line homography.** *(a) There is a pencil of epipolar lines in each image centred on the epipole. The correspondence between epipolar lines, $\mathbf{l}_i \leftrightarrow \mathbf{l}'_i$, is defined by the pencil of planes with axis the baseline. (b) The corresponding lines are related by a perspectivity with centre any point* $\mathbf{p}$ *on the baseline. It follows that the correspondence between epipolar lines in the pencils is a 1D homography.*

Result 9.5. *Suppose* $\mathbf{l}$ *and* $\mathbf{l}'$ *are corresponding epipolar lines, and* $\mathbf{k}$ *is any line not passing through the epipole* $\mathbf{e}$*, then* $\mathbf{l}$ *and* $\mathbf{l}'$ *are related by* $\mathbf{l}' = \mathtt{F}[\mathbf{k}]_\times \mathbf{l}$*. Symmetrically,* $\mathbf{l} = \mathtt{F}^\top[\mathbf{k}']_\times \mathbf{l}'$.

Proof. The expression $[\mathbf{k}]_\times \mathbf{l} = \mathbf{k} \times \mathbf{l}$ is the point of intersection of the two lines $\mathbf{k}$ and $\mathbf{l}$, and hence a point on the epipolar line $\mathbf{l}$ – call it $\mathbf{x}$. Hence, $\mathtt{F}[\mathbf{k}]_\times \mathbf{l} = \mathtt{F}\mathbf{x}$ is the epipolar line corresponding to the point $\mathbf{x}$, namely the line $\mathbf{l}'$. □

Furthermore a convenient choice for $\mathbf{k}$ is the line $\mathbf{e}$, since $\mathbf{k}^\top \mathbf{e} = \mathbf{e}^\top \mathbf{e} \neq 0$, so that the line $\mathbf{e}$ does not pass through the point $\mathbf{e}$ as is required. A similar argument holds for the choice of $\mathbf{k}' = \mathbf{e}'$. Thus the epipolar line homography may be written as

$$\mathbf{l}' = \mathtt{F}[\mathbf{e}]_\times \mathbf{l} \qquad \mathbf{l} = \mathtt{F}^\top[\mathbf{e}']_\times \mathbf{l}' \ .$$

9.3 Fundamental matrices arising from special motions

A special motion arises from a particular relationship between the translation direction, $\mathbf{t}$, and the direction of the rotation axis, $\mathbf{a}$. We will discuss two cases: *pure translation*, where there is no rotation; and *pure planar motion*, where $\mathbf{t}$ is orthogonal to $\mathbf{a}$ (the significance of the planar motion case is described in section 3.4.1(*p*77)). The 'pure' indicates that there is no change in the internal parameters. Such cases are important, firstly because they occur in practice, for example a camera viewing an object rotating on a turntable is equivalent to planar motion for pairs of views; and secondly because the fundamental matrix has a special form and thus additional properties.

9.3.1 Pure translation

In considering pure translations of the camera, one may consider the equivalent situation in which the camera is stationary, and the world undergoes a translation $-\mathbf{t}$. In this situation points in 3-space move on straight lines parallel to $\mathbf{t}$, and the imaged intersection of these parallel lines is the vanishing point $\mathbf{v}$ in the direction of $\mathbf{t}$. This is illustrated in figure 9.7 and figure 9.8. It is evident that $\mathbf{v}$ is the epipole for both views, and the imaged parallel lines are the epipolar lines. The algebraic details are given in the following example.

Samsung/Google, EX1010, p. 265

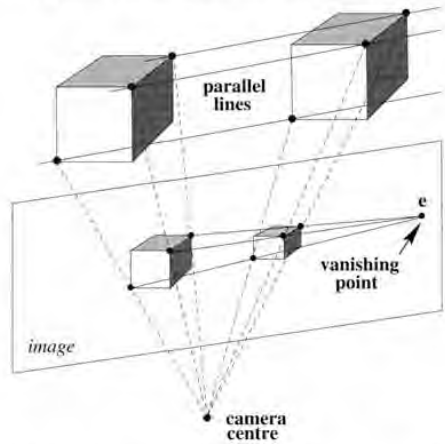


Fig. 9.7. *Under a pure translational camera motion, 3D points appear to slide along parallel rails. The images of these parallel lines intersect in a vanishing point corresponding to the translation direction. The epipole* e *is the vanishing point.*

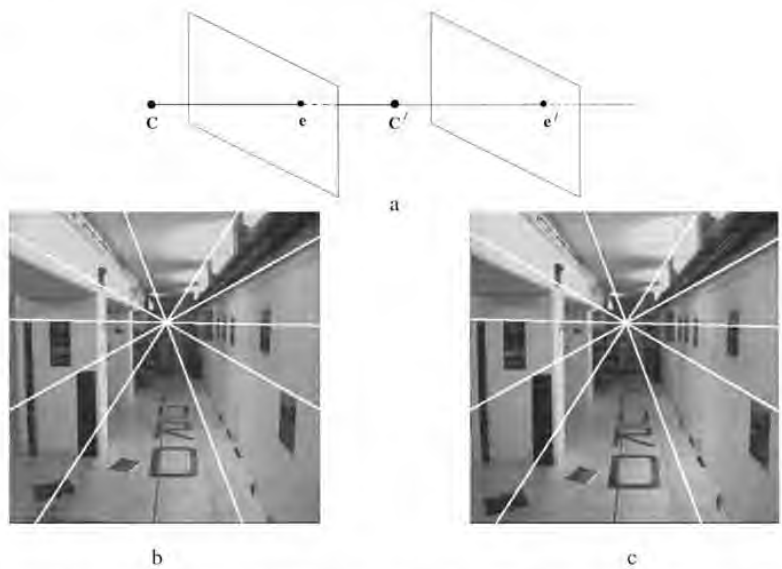


Fig. 9.8. **Pure translational motion.** *(a) under the motion the epipole is a fixed point, i.e. has the same coordinates in both images, and points appear to move along lines radiating from the epipole. The epipole in this case is termed the* Focus of Expansion *(FOE). (b) and (c) the same epipolar lines are overlaid in both cases. Note the motion of the posters on the wall which slide along the epipolar line.*

Example 9.6. Suppose the motion of the cameras is a pure translation with no rotation and no change in the internal parameters. One may assume that the two cameras are

Samsung/Google, EX1010, p. 266

$\mathtt{P} = \mathtt{K}[\mathtt{I} \mid \mathtt{0}]$ and $\mathtt{P}' = \mathtt{K}[\mathtt{I} \mid \mathtt{t}]$. Then from (9.4) (using $\mathtt{R} = \mathtt{I}$ and $\mathtt{K} = \mathtt{K}'$)

$$\mathtt{F} = [\mathbf{e}']_\times \mathtt{K}\mathtt{K}^{-1} = [\mathbf{e}']_\times.$$

If the camera translation is parallel to the x-axis, then $\mathbf{e}' = (1, 0, 0)^\mathsf{T}$, so

$$\mathtt{F} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & -1 \\ 0 & 1 & 0 \end{bmatrix}.$$

The relation between corresponding points, $\mathbf{x}'^\mathsf{T}\mathtt{F}\mathbf{x} = 0$, reduces to $y = y'$, i.e. the epipolar lines are corresponding rasters. This is the situation that is sought by image rectification described in section 11.12(p302). △

Indeed if the image point $\mathbf{x}$ is normalized as $\mathbf{x} = (x, y, 1)^\mathsf{T}$, then from $\mathbf{x} = \mathtt{P}\mathbf{X} = \mathtt{K}[\mathtt{I} \mid \mathtt{0}]\mathbf{X}$, the space point's (inhomogeneous) coordinates are $(\mathtt{X}, \mathtt{Y}, \mathtt{Z})^\mathsf{T} = \mathtt{Z}\mathtt{K}^{-1}\mathbf{x}$, where $\mathtt{Z}$ is the depth of the point $\mathbf{X}$ (the distance of $\mathbf{X}$ from the camera centre measured along the principal axis of the first camera). It then follows from $\mathbf{x}' = \mathtt{P}'\mathbf{X} = \mathtt{K}[\mathtt{I} \mid \mathtt{t}]\mathbf{X}$ that the mapping from an image point $\mathbf{x}$ to an image point $\mathbf{x}'$ is

$$\mathbf{x}' = \mathbf{x} + \mathtt{K}\mathbf{t}/\mathtt{Z}. \tag{9.6}$$

The motion $\mathbf{x}' = \mathbf{x} + \mathtt{K}\mathbf{t}/\mathtt{Z}$ of (9.6) shows that the image point "starts" at $\mathbf{x}$ and then moves along the line defined by $\mathbf{x}$ and the epipole $\mathbf{e} = \mathbf{e}' = \mathbf{v}$. The extent of the motion depends on the magnitude of the translation $\mathbf{t}$ (which is not a homogeneous vector here) and the inverse depth $\mathtt{Z}$, so that points closer to the camera appear to move faster than those further away – a common experience when looking out of a train window.

Note that in this case of pure translation $\mathtt{F} = [\mathbf{e}']_\times$ is skew-symmetric and has only 2 degrees of freedom, which correspond to the position of the epipole. The epipolar line of $\mathbf{x}$ is $\mathbf{l}' = \mathtt{F}\mathbf{x} = [\mathbf{e}]_\times\mathbf{x}$, and $\mathbf{x}$ lies on this line since $\mathbf{x}^\mathsf{T}[\mathbf{e}]_\times\mathbf{x} = 0$, i.e. $\mathbf{x}$, $\mathbf{x}'$ and $\mathbf{e} = \mathbf{e}'$ are collinear (assuming both images are overlaid on top of each other). This collinearity property is termed *auto-epipolar*, and does not hold for general motion.

General motion. The pure translation case gives additional insight into the general motion case. Given two arbitrary cameras, we may rotate the camera used for the first image so that it is aligned with the second camera. This rotation may be simulated by applying a projective transformation to the first image. A further correction may be applied to the first image to account for any difference in the calibration matrices of the two images. The result of these two corrections is a projective transformation $\mathtt{H}$ of the first image. If one assumes these corrections to have been made, then the effective relationship of the two cameras to each other is that of a pure translation. Consequently, the fundamental matrix corresponding to the corrected first image and the second image is of the form $\hat{\mathtt{F}} = [\mathbf{e}']_\times$, satisfying $\mathbf{x}'^\mathsf{T}\hat{\mathtt{F}}\hat{\mathbf{x}} = 0$, where $\hat{\mathbf{x}} = \mathtt{H}\mathbf{x}$ is the corrected point in the first image. From this one deduces that $\mathbf{x}'^\mathsf{T}[\mathbf{e}']_\times\mathtt{H}\mathbf{x} = 0$, and so the fundamental matrix corresponding to the initial point correspondences $\mathbf{x} \leftrightarrow \mathbf{x}'$ is $\mathtt{F} = [\mathbf{e}']_\times\mathtt{H}$. This is illustrated in figure 9.9.

Samsung/Google, EX1010, p. 267

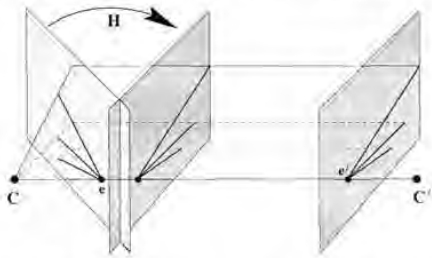


Fig. 9.9. **General camera motion.** *The first camera (on the left) may be rotated and corrected to simulate a pure translational motion. The fundamental matrix for the original pair is the product* $\mathtt{F} = [\mathbf{e}']_\times \mathtt{H}$, *where* $[\mathbf{e}']_\times$ *is the fundamental matrix of the translation, and* $\mathtt{H}$ *is the projective transformation corresponding to the correction of the first camera.*

Example 9.7. Continuing from example 9.2, assume again that the two cameras are $\mathtt{P} = \mathtt{K}[\mathtt{I} \mid \mathbf{0}]$ and $\mathtt{P}' = \mathtt{K}'[\mathtt{R} \mid \mathbf{t}]$. Then as described in section 8.4.2(p204) the requisite projective transformation is $\mathtt{H} = \mathtt{K}'\mathtt{R}\mathtt{K}^{-1} = \mathtt{H}_\infty$, where $\mathtt{H}_\infty$ is the infinite homography (see section 13.4(p338)), and $\mathtt{F} = [\mathbf{e}']_\times \mathtt{H}_\infty$.

If the image point $\mathbf{x}$ is normalized as $\mathbf{x} = (x, y, 1)^\mathsf{T}$, as in example 9.6, then $(\mathtt{X}, \mathtt{Y}, \mathtt{Z})^\mathsf{T} = \mathtt{Z}\mathtt{K}^{-1}\mathbf{x}$, and from $\mathbf{x} = \mathtt{P}'\mathbf{X} = \mathtt{K}'[\mathtt{R} \mid \mathbf{t}]\mathbf{X}$ the mapping from an image point $\mathbf{x}$ to an image point $\mathbf{x}'$ is

$$\mathbf{x}' = \mathtt{K}'\mathtt{R}\mathtt{K}^{-1}\mathbf{x} + \mathtt{K}'\mathbf{t}/\mathtt{Z}. \tag{9.7}$$

The mapping is in two parts: the first term depends on the image position alone, i.e. $\mathbf{x}$, but not the point's depth $\mathtt{Z}$, and takes account of the camera rotation and change of internal parameters; the second term depends on the depth, but not on the image position $\mathbf{x}$, and takes account of camera translation. In the case of pure translation ($\mathtt{R} = \mathtt{I}, \mathtt{K} = \mathtt{K}'$) (9.7) reduces to (9.6). △

9.3.2 Pure planar motion

In this case the rotation axis is orthogonal to the translation direction. Orthogonality imposes one constraint on the motion, and it is shown in the exercises at the end of this chapter that if $\mathtt{K}' = \mathtt{K}$ then $\mathtt{F}_\mathtt{S}$, the symmetric part of $\mathtt{F}$, has rank 2 in this planar motion case (note, for a general motion the symmetric part of $\mathtt{F}$ has full rank). Thus, the condition that $\det \mathtt{F}_\mathtt{S} = 0$ is an additional constraint on $\mathtt{F}$ and reduces the number of degrees of freedom from 7, for a general motion, to 6 degrees of freedom for a pure planar motion.

9.4 Geometric representation of the fundamental matrix

This section is not essential for a first reading and the reader may optionally skip to section 9.5.

In this section the fundamental matrix is decomposed into its symmetric and skew-symmetric parts, and each part is given a geometric representation. The symmetric and

Samsung/Google, EX1010, p. 268

skew-symmetric parts of the fundamental matrix are

$$\mathtt{F_S} = \left(\mathtt{F} + \mathtt{F}^\top\right)/2 \qquad \mathtt{F_a} = \left(\mathtt{F} - \mathtt{F}^\top\right)/2$$

so that $\mathtt{F} = \mathtt{F_S} + \mathtt{F_a}$.

To motivate the decomposition, consider the points $\mathbf{X}$ in 3-space that map to the same point in two images. These image points are fixed under the camera motion so that $\mathbf{x} = \mathbf{x}'$. Clearly such points are corresponding and thus satisfy $\mathbf{x}^\top\mathtt{F}\mathbf{x} = 0$, which is a necessary condition on corresponding points. Now, for any skew-symmetric matrix $\mathtt{A}$ the form $\mathbf{x}^\top\mathtt{A}\mathbf{x}$ is identically zero. Consequently only the symmetric part of $\mathtt{F}$ contributes to $\mathbf{x}^\top\mathtt{F}\mathbf{x} = 0$, which then reduces to $\mathbf{x}^\top\mathtt{F_S}\mathbf{x} = 0$. As will be seen below the matrix $\mathtt{F_S}$ may be thought of as a conic in the image plane.

Geometrically the conic arises as follows. The locus of all points in 3-space for which $\mathbf{x} = \mathbf{x}'$ is known as the *horopter* curve. Generally this is a twisted cubic curve in 3-space (see section 3.3(p75)) passing through the two camera centres [Maybank-93]. The image of the horopter is the conic defined by $\mathtt{F_S}$. We return to the horopter in chapter 22.

Symmetric part. The matrix $\mathtt{F_S}$ is symmetric and is of rank 3 in general. It has 5 degrees of freedom and is identified with a point conic, called the *Steiner conic* (the name is explained below). The epipoles $\mathbf{e}$ and $\mathbf{e}'$ lie on the conic $\mathtt{F_S}$. To see that the epipoles lie on the conic, i.e. that $\mathbf{e}^\top\mathtt{F_S}\mathbf{e} = 0$, start from $\mathtt{F}\mathbf{e} = \mathbf{0}$. Then $\mathbf{e}^\top\mathtt{F}\mathbf{e} = \mathbf{0}$ and so $\mathbf{e}^\top\mathtt{F_S}\mathbf{e} + \mathbf{e}^\top\mathtt{F_a}\mathbf{e} = 0$. However, $\mathbf{e}^\top\mathtt{F_a}\mathbf{e} = 0$, since for any skew-symmetric matrix $\mathtt{S}$, $\mathbf{x}^\top\mathtt{S}\mathbf{x} = 0$. Thus $\mathbf{e}^\top\mathtt{F_S}\mathbf{e} = 0$. The derivation for $\mathbf{e}'$ follows in a similar manner.

Skew-symmetric part. The matrix $\mathtt{F_a}$ is skew-symmetric and may be written as $\mathtt{F_a} = [\mathbf{x_a}]_\times$, where $\mathbf{x_a}$ is the null-vector of $\mathtt{F_a}$. The skew-symmetric part has 2 degrees of freedom and is identified with the point $\mathbf{x_a}$.

The relation between the point $\mathbf{x_a}$ and conic $\mathtt{F_S}$ is shown in figure 9.10a. The polar of $\mathbf{x_a}$ intersects the Steiner conic $\mathtt{F_S}$ at the epipoles $\mathbf{e}$ and $\mathbf{e}'$ (the pole–polar relation is described in section 2.2.3(p30)). The proof of this result is left as an exercise.

Epipolar line correspondence. It is a classical theorem of projective geometry due to Steiner [Semple-79] that for two line pencils related by a homography, the locus of intersections of corresponding lines is a conic. This is precisely the situation here. The pencils are the epipolar pencils, one through $\mathbf{e}$ and the other through $\mathbf{e}'$. The epipolar lines are related by a 1D homography as described in section 9.2.5. The locus of intersection is the conic $\mathtt{F_S}$.

The conic and epipoles enable epipolar lines to be determined by a geometric construction as illustrated in figure 9.10b. This construction is based on the fixed point property of the Steiner conic $\mathtt{F_S}$. The epipolar line $\mathbf{l} = \mathbf{x} \times \mathbf{e}$ in the first view defines an epipolar plane in 3-space which intersects the horopter in a point, which we will call $\mathbf{X}_c$. The point $\mathbf{X}_c$ is imaged in the first view at $\mathbf{x}_c$, which is the point at which $\mathbf{l}$ intersects the conic $\mathtt{F_S}$ (since $\mathtt{F_S}$ is the image of the horopter). Now the image of $\mathbf{X}_c$ is also $\mathbf{x}_c$ in the second view due to the fixed-point property of the horopter. So $\mathbf{x}_c$ is the

Samsung/Google, EX1010, p. 269

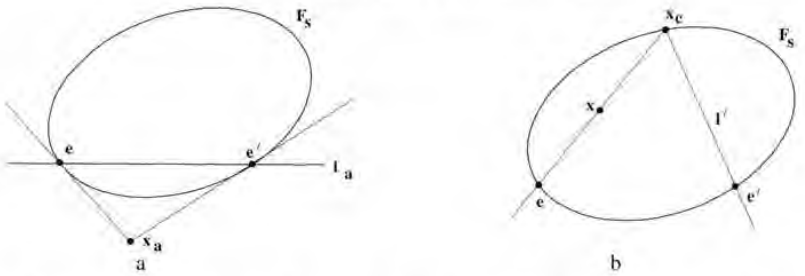


Fig. 9.10. **Geometric representation of F.** *(a) The conic F_S represents the symmetric part of F, and the point x_a the skew-symmetric part. The conic F_S is the locus of intersection of corresponding epipolar lines, assuming both images are overlaid on top of each other. It is the image of the horopter curve. The line l_a is the polar of x_a with respect to the conic F_S. It intersects the conic at the epipoles e and e'. (b) The epipolar line l' corresponding to a point x is constructed as follows: intersect the line defined by the points e and x with the conic. This intersection point is x_c. Then l' is the line defined by the points x_c and e'.*

image in the second view of a point on the epipolar plane of $\mathbf{x}$. It follows that $\mathbf{x}_c$ lies on the epipolar line $\mathbf{l}'$ of $\mathbf{x}$, and consequently $\mathbf{l}'$ may be computed as $\mathbf{l}' = \mathbf{x}_c \times \mathbf{e}'$.

The conic together with two points on the conic account for the 7 degrees of freedom of F: 5 degrees of freedom for the conic and one each to specify the two epipoles on the conic. Given F, then the conic F_S, epipoles e, e' and skew-symmetric point x_a are defined uniquely. However, F_S and x_a do not uniquely determine F since the identity of the epipoles is not recovered, i.e. the polar of x_a determines the epipoles but does not determine which one is e and which one e'.

9.4.1 Pure planar motion

We return to the case of planar motion discussed above in section 9.3.2, where F_S has rank 2. It is evident that in this case the Steiner conic is degenerate and from section 2.2.3(*p*30) is equivalent to two non-coincident lines:

$$F_S = l_h l_s^T + l_s l_h^T$$

as depicted in figure 9.11a. The geometric construction of the epipolar line $\mathbf{l}'$ corresponding to a point $\mathbf{x}$ of section 9.4 has a simple algebraic representation in this case. As in the general motion case, there are three steps, illustrated in figure 9.11b: first the line $\mathbf{l} = \mathbf{e} \times \mathbf{x}$ joining e and $\mathbf{x}$ is computed; second, its intersection point with the "conic" $\mathbf{x}_c = \mathbf{l}_s \times \mathbf{l}$ is determined; third the epipolar line $\mathbf{l}' = \mathbf{e}' \times \mathbf{x}_c$ is the join of $\mathbf{x}_c$ and e'. Putting these steps together we find

$$\mathbf{l}' = \mathbf{e}' \times [\mathbf{l}_s \times (\mathbf{e} \times \mathbf{x})] = [\mathbf{e}']_\times [\mathbf{l}_s]_\times [\mathbf{e}]_\times \mathbf{x}.$$

It follows that F may be written as

$$F = [\mathbf{e}']_\times [\mathbf{l}_s]_\times [\mathbf{e}]_\times. \tag{9.8}$$

The 6 degrees of freedom of F are accounted for as 2 degrees of freedom for each of the two epipoles and 2 degrees of freedom for the line.

Samsung/Google, EX1010, p. 270

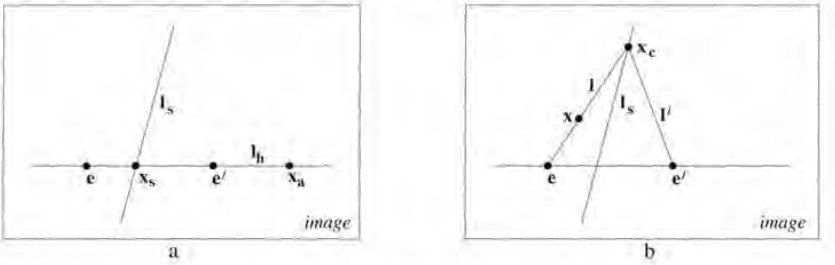


Fig. 9.11. **Geometric representation of F for planar motion.** *(a) The lines l_s and l_h constitute the Steiner conic for this motion, which is degenerate. Compare this figure with the conic for general motion shown in figure 9.10. (b) The epipolar line l' corresponding to a point x is constructed as follows: intersect the line defined by the points e and x with the (conic) line l_s. This intersection point is x_c. Then l' is the line defined by the points x_c and e'.*

The geometry of this situation can be easily visualized: the horopter for this motion is a degenerate twisted cubic consisting of a circle in the plane of the motion (the plane orthogonal to the rotation axis and containing the camera centres), and a line parallel to the rotation axis and intersecting the circle. The line is the screw axis (see section 3.4.1($p77$)). The motion is equivalent to a rotation about the screw axis with zero translation. Under this motion points on the screw axis are fixed, and consequently their images are fixed. The line l_s is the image of the screw axis. The line l_h is the intersection of the image with the plane of the motion. This geometry is used for auto-calibration in chapter 19.

9.5 Retrieving the camera matrices

To this point we have examined the properties of F and of image relations for a point correspondence $x \leftrightarrow x'$. We now turn to one of the most significant properties of F, that the matrix may be used to determine the camera matrices of the two views.

9.5.1 Projective invariance and canonical cameras

It is evident from the derivations of section 9.2 that the map $l' = Fx$ and the correspondence condition $x'^{\mathsf{T}}Fx = 0$ are *projective* relationships: the derivations have involved only projective geometric relationships, such as the intersection of lines and planes, and in the algebraic development only the linear mapping of the projective camera between world and image points. Consequently, the relationships depend only on projective coordinates in the image, and not, for example on Euclidean measurements such as the angle between rays. In other words the image relationships are projectively invariant: under a projective transformation of the image coordinates $\hat{x} = Hx$, $\hat{x}' = H'x'$, there is a corresponding map $\hat{l}' = \hat{F}\hat{x}$ with $\hat{F} = H'^{-\mathsf{T}}FH^{-1}$ the corresponding rank 2 fundamental matrix.

Similarly, F only depends on projective properties of the cameras P, P'. The camera matrix relates 3-space measurements to image measurements and so depends on both the image coordinate frame and the choice of world coordinate frame. F does not

Samsung/Google, EX1010, p. 271

depend on the choice of world frame, for example a rotation of world coordinates changes $\mathtt{P}, \mathtt{P}'$, but not $\mathtt{F}$. In fact, the fundamental matrix is unchanged by a projective transformation of 3-space. More precisely,

Result 9.8. *If* $\mathtt{H}$ *is a* 4×4 *matrix representing a projective transformation of 3-space, then the fundamental matrices corresponding to the pairs of camera matrices* $(\mathtt{P}, \mathtt{P}')$ *and* $(\mathtt{PH}, \mathtt{P}'\mathtt{H})$ *are the same.*

Proof. Observe that $\mathtt{P}\mathbf{X} = (\mathtt{PH})(\mathtt{H}^{-1}\mathbf{X})$, and similarly for $\mathtt{P}'$. Thus if $\mathbf{x} \leftrightarrow \mathbf{x}'$ are matched points with respect to the pair of cameras $(\mathtt{P}, \mathtt{P}')$, corresponding to a 3D point $\mathbf{X}$, then they are also matched points with respect to the pair of cameras $(\mathtt{PH}, \mathtt{P}'\mathtt{H})$, corresponding to the point $\mathtt{H}^{-1}\mathbf{X}$. □

Thus, although from (9.1–*p*244) a pair of camera matrices $(\mathtt{P}, \mathtt{P}')$ uniquely determine a fundamental matrix $\mathtt{F}$, the converse is not true. The fundamental matrix determines the pair of camera matrices at best up to right-multiplication by a 3D projective transformation. It will be seen below that this is the full extent of the ambiguity, and indeed the camera matrices are determined up to a projective transformation by the fundamental matrix.

Canonical form of camera matrices. Given this ambiguity, it is common to define a specific *canonical form* for the pair of camera matrices corresponding to a given fundamental matrix in which the first matrix is of the simple form $[\mathtt{I} \mid \mathbf{0}]$, where $\mathtt{I}$ is the 3×3 identity matrix and $\mathbf{0}$ a null 3-vector. To see that this is always possible, let $\mathtt{P}$ be augmented by one row to make a 4×4 non-singular matrix, denoted $\mathtt{P}^*$. Now letting $\mathtt{H} = \mathtt{P}^{*-1}$, one verifies that $\mathtt{PH} = [\mathtt{I} \mid \mathbf{0}]$ as desired.

The following result is very frequently used

Result 9.9. *The fundamental matrix corresponding to a pair of camera matrices* $\mathtt{P} = [\mathtt{I} \mid \mathbf{0}]$ *and* $\mathtt{P}' = [\mathtt{M} \mid \mathbf{m}]$ *is equal to* $[\mathbf{m}]_\times \mathtt{M}$.

This is easily derived as a special case of (9.1–*p*244).

9.5.2 Projective ambiguity of cameras given F

It has been seen that a pair of camera matrices determines a unique fundamental matrix. This mapping is not injective (one-to-one) however, since pairs of camera matrices that differ by a projective transformation give rise to the same fundamental matrix. It will now be shown that this is the only ambiguity. We will show that a given fundamental matrix determines the pair of camera matrices up to right multiplication by a projective transformation. Thus, the fundamental matrix captures the projective relationship of the two cameras.

Theorem 9.10. *Let* $\mathtt{F}$ *be a fundamental matrix and let* $(\mathtt{P}, \mathtt{P}')$ *and* $(\tilde{\mathtt{P}}, \tilde{\mathtt{P}}')$ *be two pairs of camera matrices such that* $\mathtt{F}$ *is the fundamental matrix corresponding to each of these pairs. Then there exists a non-singular* 4×4 *matrix* $\mathtt{H}$ *such that* $\tilde{\mathtt{P}} = \mathtt{PH}$ *and* $\tilde{\mathtt{P}}' = \mathtt{P}'\mathtt{H}$.

Samsung/Google, EX1010, p. 272

Proof. Suppose that a given fundamental matrix $\mathtt{F}$ corresponds to two different pairs of camera matrices $(\mathtt{P}, \mathtt{P}')$ and $(\tilde{\mathtt{P}}, \tilde{\mathtt{P}}')$. As a first step, we may simplify the problem by assuming that each of the two pair of camera matrices is in canonical form with $\mathtt{P} = \tilde{\mathtt{P}} = [\mathtt{I} \mid \mathtt{0}]$, since this may be done by applying projective transformations to each pair as necessary. Thus, suppose that $\mathtt{P} = \tilde{\mathtt{P}} = [\mathtt{I} \mid \mathtt{0}]$ and that $\mathtt{P}' = [\mathtt{A} \mid \mathbf{a}]$ and $\tilde{\mathtt{P}}' = [\tilde{\mathtt{A}} \mid \tilde{\mathbf{a}}]$. According to result 9.9 the fundamental matrix may then be written $\mathtt{F} = [\mathbf{a}]_\times \mathtt{A} = [\tilde{\mathbf{a}}]_\times \tilde{\mathtt{A}}$.
We will need the following lemma:

Lemma 9.11. *Suppose the rank 2 matrix* $\mathtt{F}$ *can be decomposed in two different ways as* $\mathtt{F} = [\mathbf{a}]_\times \mathtt{A}$ *and* $\mathtt{F} = [\tilde{\mathbf{a}}]_\times \tilde{\mathtt{A}}$*; then* $\tilde{\mathbf{a}} = k\mathbf{a}$ *and* $\tilde{\mathtt{A}} = k^{-1}(\mathtt{A} + \mathbf{a}\mathbf{v}^\mathsf{T})$ *for some non-zero constant* k *and 3-vector* $\mathbf{v}$*.*

Proof. First, note that $\mathbf{a}^\mathsf{T}\mathtt{F} = \mathbf{a}^\mathsf{T}[\mathbf{a}]_\times \mathtt{A} = \mathbf{0}$, and similarly, $\tilde{\mathbf{a}}^\mathsf{T}\mathtt{F} = \mathbf{0}$. Since $\mathtt{F}$ has rank 2, it follows that $\tilde{\mathbf{a}} = k\mathbf{a}$ as required. Next, from $[\mathbf{a}]_\times \mathtt{A} = [\tilde{\mathbf{a}}]_\times \tilde{\mathtt{A}}$ it follows that $[\mathbf{a}]_\times \left(k\tilde{\mathtt{A}} - \mathtt{A}\right) = \mathtt{0}$, and so $k\tilde{\mathtt{A}} - \mathtt{A} = \mathbf{a}\mathbf{v}^\mathsf{T}$ for some $\mathbf{v}$. Hence, $\tilde{\mathtt{A}} = k^{-1}(\mathtt{A} + \mathbf{a}\mathbf{v}^\mathsf{T})$ as required. □

Applying this result to the two camera matrices $\mathtt{P}'$ and $\tilde{\mathtt{P}}'$ shows that $\mathtt{P}' = [\mathtt{A} \mid \mathbf{a}]$ and $\tilde{\mathtt{P}}' = [k^{-1}(\mathtt{A} + \mathbf{a}\mathbf{v}^\mathsf{T}) \mid k\mathbf{a}]$ if they are to generate the same $\mathtt{F}$. It only remains now to show that these camera pairs are projectively related. Let $\mathtt{H}$ be the matrix $\mathtt{H} = \begin{bmatrix} k^{-1}\mathtt{I} & \mathbf{0} \\ k^{-1}\mathbf{v}^\mathsf{T} & k \end{bmatrix}$. Then one verifies that $\mathtt{P}\mathtt{H} = k^{-1}[\mathtt{I} \mid \mathbf{0}] = k^{-1}\tilde{\mathtt{P}}$, and furthermore,

$$\mathtt{P}'\mathtt{H} = [\mathtt{A} \mid \mathbf{a}]\mathtt{H} = [k^{-1}(\mathtt{A} + \mathbf{a}\mathbf{v}^\mathsf{T}) \mid k\mathbf{a}] = [\tilde{\mathtt{A}} \mid \tilde{\mathbf{a}}] = \tilde{\mathtt{P}}'$$

so that the pairs $\mathtt{P}, \mathtt{P}'$ and $\tilde{\mathtt{P}}, \tilde{\mathtt{P}}'$ are indeed projectively related. □

This can be tied precisely to a counting argument: the two cameras $\mathtt{P}$ and $\mathtt{P}'$ each have 11 degrees of freedom, making a total of 22 degrees of freedom. To specify a projective world frame requires 15 degrees of freedom (section 3.1(p65)), so once the degrees of freedom of the world frame are removed from the two cameras $22 - 15 = 7$ degrees of freedom remain – which corresponds to the 7 degrees of freedom of the fundamental matrix.

9.5.3 Canonical cameras given $\mathtt{F}$

We have shown that $\mathtt{F}$ determines the camera pair up to a projective transformation of 3-space. We will now derive a specific formula for a pair of cameras with canonical form given $\mathtt{F}$. We will make use of the following characterization of the fundamental matrix $\mathtt{F}$ corresponding to a pair of camera matrices:

Result 9.12. *A non-zero matrix* $\mathtt{F}$ *is the fundamental matrix corresponding to a pair of camera matrices* $\mathtt{P}$ *and* $\mathtt{P}'$ *if and only if* $\mathtt{P}'^\mathsf{T}\mathtt{F}\mathtt{P}$ *is skew-symmetric.*

Proof. The condition that $\mathtt{P}'^\mathsf{T}\mathtt{F}\mathtt{P}$ is skew-symmetric is equivalent to $\mathbf{X}^\mathsf{T}\mathtt{P}'^\mathsf{T}\mathtt{F}\mathtt{P}\mathbf{X} = 0$ for all $\mathbf{X}$. Setting $\mathbf{x}' = \mathtt{P}'\mathbf{X}$ and $\mathbf{x} = \mathtt{P}\mathbf{X}$, this is equivalent to $\mathbf{x}'^\mathsf{T}\mathtt{F}\mathbf{x} = 0$, which is the defining equation for the fundamental matrix. □

Samsung/Google, EX1010, p. 273

One may write down a particular solution for the pairs of camera matrices in canonical form that correspond to a fundamental matrix as follows:

Result 9.13. *Let* $\mathtt{F}$ *be a fundamental matrix and* $\mathtt{S}$ *any skew-symmetric matrix. Define the pair of camera matrices*

$$\mathtt{P} = [\mathtt{I} \mid \mathbf{0}] \quad \textit{and} \quad \mathtt{P}' = [\mathtt{SF} \mid \mathbf{e}'],$$

where $\mathbf{e}'$ *is the epipole such that* $\mathbf{e}'^{\mathsf{T}}\mathtt{F} = \mathbf{0}$*, and assume that* $\mathtt{P}'$ *so defined is a valid camera matrix (has rank 3). Then* $\mathtt{F}$ *is the fundamental matrix corresponding to the pair* $(\mathtt{P}, \mathtt{P}')$.

To demonstrate this, we invoke result 9.12 and simply verify that

$$[\mathtt{SF} \mid \mathbf{e}']^{\mathsf{T}}\mathtt{F}[\mathtt{I} \mid \mathbf{0}] = \begin{bmatrix} \mathtt{F}^{\mathsf{T}}\mathtt{S}^{\mathsf{T}}\mathtt{F} & \mathbf{0} \\ \mathbf{e}'^{\mathsf{T}}\mathtt{F} & 0 \end{bmatrix} = \begin{bmatrix} \mathtt{F}^{\mathsf{T}}\mathtt{S}^{\mathsf{T}}\mathtt{F} & \mathbf{0} \\ \mathbf{0}^{\mathsf{T}} & 0 \end{bmatrix} \tag{9.9}$$

which is skew-symmetric.

The skew-symmetric matrix $\mathtt{S}$ may be written in terms of its null-vector as $\mathtt{S} = [\mathbf{s}]_\times$. Then $[[\mathbf{s}]_\times\mathtt{F} \mid \mathbf{e}']$ has rank 3 provided $\mathbf{s}^{\mathsf{T}}\mathbf{e}' \neq 0$, according to the following argument. Since $\mathbf{e}'\mathtt{F} = \mathbf{0}$, the column space (span of the columns) of $\mathtt{F}$ is perpendicular to $\mathbf{e}'$. But if $\mathbf{s}^{\mathsf{T}}\mathbf{e}' \neq 0$, then $\mathbf{s}$ is not perpendicular to $\mathbf{e}'$, and hence not in the column space of $\mathtt{F}$. Now, the column space of $[\mathbf{s}]_\times\mathtt{F}$ is spanned by the cross-products of $\mathbf{s}$ with the columns of $\mathtt{F}$, and therefore equals the plane perpendicular to $\mathbf{s}$. So $[\mathbf{s}]_\times\mathtt{F}$ has rank 2. Since $\mathbf{e}'$ is not perpendicular to $\mathbf{s}$, it does not lie in this plane, and so $[[\mathbf{s}]_\times\mathtt{F} \mid \mathbf{e}']$ has rank 3, as required.

As suggested by Luong and Viéville [Luong-96] a good choice for $\mathtt{S}$ is $\mathtt{S} = [\mathbf{e}']_\times$, for in this case $\mathbf{e}'^{\mathsf{T}}\mathbf{e}' \neq 0$, which leads to the following useful result.

Result 9.14. *The camera matrices corresponding to a fundamental matrix* $\mathtt{F}$ *may be chosen as* $\mathtt{P} = [\mathtt{I} \mid \mathbf{0}]$ *and* $\mathtt{P}' = [[\mathbf{e}']_\times\mathtt{F} \mid \mathbf{e}']$.

Note that the camera matrix $\mathtt{P}'$ has left 3×3 submatrix $[\mathbf{e}']_\times\mathtt{F}$ which has rank 2. This corresponds to a camera with centre on $\boldsymbol{\pi}_\infty$. However, there is no particular reason to avoid this situation.

The proof of theorem 9.10 shows that the four parameter family of camera pairs in canonical form $\tilde{\mathtt{P}} = [\mathtt{I} \mid \mathbf{0}]$, $\tilde{\mathtt{P}}' = [\mathtt{A} + \mathbf{a}\mathbf{v}^{\mathsf{T}} \mid k\mathbf{a}]$ have the same fundamental matrix as the canonical pair, $\mathtt{P} = [\mathtt{I} \mid \mathbf{0}]$, $\mathtt{P}' = [\mathtt{A} \mid \mathbf{a}]$; and that this is the most general solution. To summarize:

Result 9.15. *The general formula for a pair of canonic camera matrices corresponding to a fundamental matrix* $\mathtt{F}$ *is given by*

$$\mathtt{P} = [\mathtt{I} \mid \mathbf{0}] \quad \mathtt{P}' = [[\mathbf{e}']_\times\mathtt{F} + \mathbf{e}'\mathbf{v}^{\mathsf{T}} \mid \lambda\mathbf{e}'] \tag{9.10}$$

where $\mathbf{v}$ *is any 3-vector, and* λ *a non-zero scalar.*

Samsung/Google, EX1010, p. 274

9.6 The essential matrix

The essential matrix is the specialization of the fundamental matrix to the case of normalized image coordinates (see below). Historically, the essential matrix was introduced (by Longuet-Higgins) before the fundamental matrix, and the fundamental matrix may be thought of as the generalization of the essential matrix in which the (inessential) assumption of calibrated cameras is removed. The essential matrix has fewer degrees of freedom, and additional properties, compared to the fundamental matrix. These properties are described below.

Normalized coordinates. Consider a camera matrix decomposed as $\mathtt{P} = \mathtt{K}[\mathtt{R} \mid \mathbf{t}]$, and let $\mathbf{x} = \mathtt{P}\mathbf{X}$ be a point in the image. If the calibration matrix $\mathtt{K}$ is known, then we may apply its inverse to the point $\mathbf{x}$ to obtain the point $\hat{\mathbf{x}} = \mathtt{K}^{-1}\mathbf{x}$. Then $\hat{\mathbf{x}} = [\mathtt{R} \mid \mathbf{t}]\mathbf{X}$, where $\hat{\mathbf{x}}$ is the image point expressed in *normalized coordinates*. It may be thought of as the image of the point $\mathbf{X}$ with respect to a camera $[\mathtt{R} \mid \mathbf{t}]$ having the identity matrix $\mathtt{I}$ as calibration matrix. The camera matrix $\mathtt{K}^{-1}\mathtt{P} = [\mathtt{R} \mid \mathbf{t}]$ is called a *normalized camera matrix*, the effect of the known calibration matrix having been removed.

Now, consider a pair of normalized camera matrices $\mathtt{P} = [\mathtt{I} \mid \mathbf{0}]$ and $\mathtt{P}' = [\mathtt{R} \mid \mathbf{t}]$. The fundamental matrix corresponding to the pair of normalized cameras is customarily called the *essential matrix*, and according to (9.2–*p*244) it has the form

$$\mathtt{E} = [\mathbf{t}]_\times \mathtt{R} = \mathtt{R}[\mathtt{R}^\top \mathbf{t}]_\times.$$

Definition 9.16. The defining equation for the essential matrix is

$$\hat{\mathbf{x}}'^\top \mathtt{E} \hat{\mathbf{x}} = 0 \tag{9.11}$$

in terms of the normalized image coordinates for corresponding points $\mathbf{x} \leftrightarrow \mathbf{x}'$.

Substituting for $\hat{\mathbf{x}}$ and $\hat{\mathbf{x}}'$ gives $\mathbf{x}'^\top \mathtt{K}'^{-\top} \mathtt{E} \mathtt{K}^{-1} \mathbf{x} = 0$. Comparing this with the relation $\mathbf{x}'^\top \mathtt{F} \mathbf{x} = 0$ for the fundamental matrix, it follows that the relationship between the fundamental and essential matrices is

$$\mathtt{E} = \mathtt{K}'^\top \mathtt{F} \mathtt{K}. \tag{9.12}$$

9.6.1 Properties of the essential matrix

The essential matrix, $\mathtt{E} = [\mathbf{t}]_\times \mathtt{R}$, has only five degrees of freedom: both the rotation matrix $\mathtt{R}$ and the translation $\mathbf{t}$ have three degrees of freedom, but there is an overall scale ambiguity – like the fundamental matrix, the essential matrix is a homogeneous quantity.

The reduced number of degrees of freedom translates into extra constraints that are satisfied by an essential matrix, compared with a fundamental matrix. We investigate what these constraints are.

Result 9.17. *A 3×3 matrix is an essential matrix if and only if two of its singular values are equal, and the third is zero.*

Samsung/Google, EX1010, p. 275

Proof. This is easily deduced from the decomposition of $\mathtt{E}$ as $[\mathtt{t}]_\times \mathtt{R} = \mathtt{SR}$, where $\mathtt{S}$ is skew-symmetric. We will use the matrices

$$\mathtt{W} = \begin{bmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad \text{and} \quad \mathtt{Z} = \begin{bmatrix} 0 & 1 & 0 \\ -1 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}. \tag{9.13}$$

It may be verified that $\mathtt{W}$ is orthogonal and $\mathtt{Z}$ is skew-symmetric. From Result A4.1-($p581$), which gives a block decomposition of a general skew-symmetric matrix, the 3×3 skew-symmetric matrix $\mathtt{S}$ may be written as $\mathtt{S} = k\mathtt{U}\mathtt{Z}\mathtt{U}^\mathsf{T}$ where $\mathtt{U}$ is orthogonal. Noting that, up to sign, $\mathtt{Z} = \mathrm{diag}(1,1,0)\mathtt{W}$, then up to scale, $\mathtt{S} = \mathtt{U}\,\mathrm{diag}(1,1,0)\mathtt{W}\mathtt{U}^\mathsf{T}$, and $\mathtt{E} = \mathtt{SR} = \mathtt{U}\,\mathrm{diag}(1,1,0)(\mathtt{W}\mathtt{U}^\mathsf{T}\mathtt{R})$. This is a singular value decomposition of $\mathtt{E}$ with two equal singular values, as required. Conversely, a matrix with two equal singular values may be factored as $\mathtt{SR}$ in this way. □

Since $\mathtt{E} = \mathtt{U}\,\mathrm{diag}(1,1,0)\mathtt{V}^\mathsf{T}$, it may seem that $\mathtt{E}$ has six degrees of freedom and not five, since both $\mathtt{U}$ and $\mathtt{V}$ have three degrees of freedom. However, because the two singular values are equal, the SVD is not unique – in fact there is a one-parameter family of SVDs for $\mathtt{E}$. Indeed, an alternative SVD is given by $\mathtt{E} = (\mathtt{U}\,\mathrm{diag}(\mathtt{R}_{2\times 2}, 1))\,\mathrm{diag}(1,1,0)(\mathrm{diag}(\mathtt{R}_{2\times 2}^\mathsf{T}, 1))\mathtt{V}^\mathsf{T}$ for any 2×2 rotation matrix $\mathtt{R}$.

9.6.2 Extraction of cameras from the essential matrix

The essential matrix may be computed directly from (9.11) using normalized image coordinates, or else computed from the fundamental matrix using (9.12). (Methods of computing the fundamental matrix are deferred to chapter 11). Once the essential matrix is known, the camera matrices may be retrieved from $\mathtt{E}$ as will be described next. In contrast with the fundamental matrix case, where there is a projective ambiguity, the camera matrices may be retrieved from the essential matrix up to scale and a four-fold ambiguity. That is there are four possible solutions, except for overall scale, which cannot be determined.

We may assume that the first camera matrix is $\mathtt{P} = [\mathtt{I} \mid \mathtt{0}]$. In order to compute the second camera matrix, $\mathtt{P}'$, it is necessary to factor $\mathtt{E}$ into the product $\mathtt{SR}$ of a skew-symmetric matrix and a rotation matrix.

Result 9.18. *Suppose that the SVD of* $\mathtt{E}$ *is* $\mathtt{U}\,\mathrm{diag}(1,1,0)\mathtt{V}^\mathsf{T}$. *Using the notation of (9.13), there are (ignoring signs) two possible factorizations* $\mathtt{E} = \mathtt{SR}$ *as follows:*

$$\mathtt{S} = \mathtt{U}\mathtt{Z}\mathtt{U}^\mathsf{T} \quad \mathtt{R} = \mathtt{U}\mathtt{W}\mathtt{V}^\mathsf{T} \quad \textit{or} \quad \mathtt{U}\mathtt{W}^\mathsf{T}\mathtt{V}^\mathsf{T}\ . \tag{9.14}$$

Proof. That the given factorization is valid is true by inspection. That there are no other factorizations is shown as follows. Suppose $\mathtt{E} = \mathtt{SR}$. The form of $\mathtt{S}$ is determined by the fact that its left null-space is the same as that of $\mathtt{E}$. Hence $\mathtt{S} = \mathtt{U}\mathtt{Z}\mathtt{U}^\mathsf{T}$. The rotation $\mathtt{R}$ may be written as $\mathtt{U}\mathtt{X}\mathtt{V}^\mathsf{T}$, where $\mathtt{X}$ is some rotation matrix. Then

$$\mathtt{U}\,\mathrm{diag}(1,1,0)\mathtt{V}^\mathsf{T} = \mathtt{E} = \mathtt{SR} = (\mathtt{U}\mathtt{Z}\mathtt{U}^\mathsf{T})(\mathtt{U}\mathtt{X}\mathtt{V}^\mathsf{T}) = \mathtt{U}(\mathtt{Z}\mathtt{X})\mathtt{V}^\mathsf{T}$$

from which one deduces that $\mathtt{Z}\mathtt{X} = \mathrm{diag}(1,1,0)$. Since $\mathtt{X}$ is a rotation matrix, it follows that $\mathtt{X} = \mathtt{W}$ or $\mathtt{X} = \mathtt{W}^\mathsf{T}$, as required. □

Samsung/Google, EX1010, p. 276

The factorization (9.14) determines the $\mathbf{t}$ part of the camera matrix $\mathtt{P}'$, up to scale, from $\mathtt{S} = [\mathbf{t}]_\times$. However, the Frobenius norm of $\mathtt{S} = \mathtt{U}\mathtt{Z}\mathtt{U}^\mathsf{T}$ is $\sqrt{2}$, which means that if $\mathtt{S} = [\mathbf{t}]_\times$ *including scale* then $\|\mathbf{t}\| = 1$, which is a convenient normalization for the baseline of the two camera matrices. Since $\mathtt{S}\mathbf{t} = \mathbf{0}$, it follows that $\mathbf{t} = \mathtt{U}(0,0,1)^\mathsf{T} = \mathbf{u}_3$, the last column of $\mathtt{U}$. However, the sign of $\mathtt{E}$, and consequently $\mathbf{t}$, cannot be determined. Thus, corresponding to a given essential matrix, there are four possible choices of the camera matrix $\mathtt{P}'$, based on the two possible choices of $\mathtt{R}$ and two possible signs of $\mathbf{t}$. To summarize:

Result 9.19. *For a given essential matrix* $\mathtt{E} = \mathtt{U}\,\mathrm{diag}(1,1,0)\mathtt{V}^\mathsf{T}$, *and first camera matrix* $\mathtt{P} = [\mathtt{I} \mid \mathbf{0}]$, *there are four possible choices for the second camera matrix* $\mathtt{P}'$, *namely*

$$\mathtt{P}' = [\mathtt{U}\mathtt{W}\mathtt{V}^\mathsf{T} \mid +\mathbf{u}_3] \text{ or } [\mathtt{U}\mathtt{W}\mathtt{V}^\mathsf{T} \mid -\mathbf{u}_3] \text{ or } [\mathtt{U}\mathtt{W}^\mathsf{T}\mathtt{V}^\mathsf{T} \mid +\mathbf{u}_3] \text{ or } [\mathtt{U}\mathtt{W}^\mathsf{T}\mathtt{V}^\mathsf{T} \mid -\mathbf{u}_3].$$

9.6.3 Geometrical interpretation of the four solutions

It is clear that the difference between the first two solutions is simply that the direction of the translation vector from the first to the second camera is reversed.

The relationship of the first and third solutions in result 9.19 is a little more complicated. However, it may be verified that

$$[\mathtt{U}\mathtt{W}^\mathsf{T}\mathtt{V}^\mathsf{T} \mid \mathbf{u}_3] = [\mathtt{U}\mathtt{W}\mathtt{V}^\mathsf{T} \mid \mathbf{u}_3] \begin{bmatrix} \mathtt{V}\mathtt{W}^\mathsf{T}\mathtt{W}^\mathsf{T}\mathtt{V}^\mathsf{T} & \\ & 1 \end{bmatrix}$$

and $\mathtt{V}\mathtt{W}^\mathsf{T}\mathtt{W}^\mathsf{T}\mathtt{V}^\mathsf{T} = \mathtt{V}\,\mathrm{diag}(-1,-1,1)\mathtt{V}^\mathsf{T}$ is a rotation through 180° about the line joining the two camera centres. Two solutions related in this way are known as a "twisted pair".

The four solutions are illustrated in figure 9.12, where it is shown that a reconstructed point $\mathbf{X}$ will be in front of both cameras in one of these four solutions only. Thus, testing with a single point to determine if it is in front of both cameras is sufficient to decide between the four different solutions for the camera matrix $\mathtt{P}'$.

Note. The point of view has been taken here that the essential matrix is a homogeneous quantity. An alternative point of view is that the essential matrix is defined exactly by the equation $\mathtt{E} = [\mathbf{t}]_\times\mathtt{R}$, (i.e. including scale), and is determined only up to indeterminate scale by the equation $\mathbf{x}'^\mathsf{T}\mathtt{E}\mathbf{x} = 0$. The choice of point of view depends on which of these two equations one regards as the defining property of the essential matrix.

9.7 Closure

9.7.1 The literature

The essential matrix was introduced to the computer vision community by Longuet-Higgins [LonguetHiggins-81], with a matrix analogous to $\mathtt{E}$ appearing in the photogrammetry literature, e.g. [VonSanden-08]. Many properties of the essential matrix have been elucidated particularly by Huang and Faugeras [Huang-89], [Maybank-93], and [Horn-90].

The realization that the essential matrix could also be applied in uncalibrated situations, as it represented a projective relation, developed in the early part of the 1990s,

Samsung/Google, EX1010, p. 277

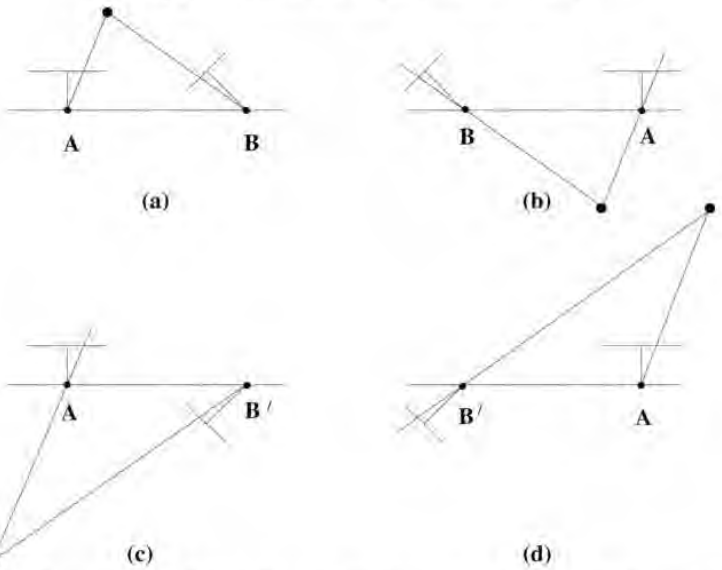


Fig. 9.12. **The four possible solutions for calibrated reconstruction from E.** *Between the left and right sides there is a baseline reversal. Between the top and bottom rows camera B rotates* 180° *about the baseline. Note, only in (a) is the reconstructed point in front of both cameras.*

and was published simultaneously by Faugeras [Faugeras-92b, Faugeras-92a], and Hartley *et al.* [Hartley-92a, Hartley-92c].

The special case of pure planar motion was examined by [Maybank-93] for the essential matrix. The corresponding case for the fundamental matrix is investigated by Beardsley and Zisserman [Beardsley-95a] and Viéville and Lingrand [Vieville-95], where further properties are given.

9.7.2 Notes and exercises

(i) **Fixating cameras.** Suppose two cameras fixate on a point in space such that their principal axes intersect at that point. Show that if the image coordinates are normalized so that the coordinate origin coincides with the principal point then the F_{33} element of the fundamental matrix is zero.

(ii) **Mirror images.** Suppose that a camera views an object and its reflection in a plane mirror. Show that this situation is equivalent to two views of the object, and that the fundamental matrix is skew-symmetric. Compare the fundamental matrix for this configuration with that of: (a) a pure translation, and (b) a pure planar motion. Show that the fundamental matrix is auto-epipolar (as is (a)).

(iii) Show that if the vanishing line of a plane contains the epipole then the plane is parallel to the baseline.

(iv) **Steiner conic.** Show that the polar of $\mathbf{x}_a$ intersects the Steiner conic $\mathtt{F}_S$ at the epipoles (figure 9.10a). Hint, start from $\mathtt{F}\mathbf{e} = \mathtt{F}_S\mathbf{e} + \mathtt{F}_a\mathbf{e} = \mathbf{0}$. Since $\mathbf{e}$ lies on

Samsung/Google, EX1010, p. 278

the conic $\mathtt{F}_S$, then $\mathbf{l}_1 = \mathtt{F}_S\mathbf{e}$ is the tangent line at $\mathbf{e}$, and $\mathbf{l}_2 = \mathtt{F}_a\mathbf{e} = [\mathbf{x}_a]_\times\mathbf{e} = \mathbf{x}_a \times \mathbf{e}$ is a line through $\mathbf{x}_a$ and $\mathbf{e}$.

(v) The affine type of the Steiner conic (hyperbola, ellipse or parabola as given in section 2.8.2(*p*59)) depends on the relative configuration of the two cameras. For example, if the two cameras are facing each other then the Steiner conic is a hyperbola. This is shown in [Chum-03] where further results on oriented epipolar geometry are given.

(vi) **Planar motion.** It is shown by [Maybank-93] that if the rotation axis direction is orthogonal or parallel to the translation direction then the symmetric part of the essential matrix has rank 2. We assume here that $\mathtt{K} = \mathtt{K}'$. Then from (9.12), $\mathtt{F} = \mathtt{K}^{-\mathsf{T}}\mathtt{E}\mathtt{K}^{-1}$, and so

$$\mathtt{F}_S = (\mathtt{F} + \mathtt{F}^\mathsf{T})/2 = \mathtt{K}^{-\mathsf{T}}(\mathtt{E} + \mathtt{E}^\mathsf{T})\mathtt{K}^{-1}/2 = \mathtt{K}^{-\mathsf{T}}\mathtt{E}_S\mathtt{K}^{-1}.$$

It follows from $\det(\mathtt{F}_S) = \det(\mathtt{K}^{-1})^2 \det(\mathtt{E}_S)$ that the symmetric part of $\mathtt{F}$ is also singular. Does this result hold if $\mathtt{K} \neq \mathtt{K}'$?

(vii) Any matrix $\mathtt{F}$ of rank 2 is the fundamental matrix corresponding to some pair of camera matrices $(\mathtt{P}, \mathtt{P}')$ This follows directly from result 9.14 since the solution for the canonical cameras depends only on the rank 2 property of $\mathtt{F}$.

(viii) Show that the 3D points determined from one of the ambiguous reconstructions obtained from $\mathtt{E}$ are related to the corresponding 3D points determined from another reconstruction by either (i) an inversion through the second camera centre; or (ii) a harmonic homology of 3-space (see section A7.2(*p*629)), where the homology plane is perpendicular to the baseline and through the second camera centre, and the vertex is the first camera centre.

(ix) Following a similar development to section 9.2.2, derive the form of the fundamental matrix for two linear pushbroom cameras. Details of this matrix are given in [Gupta-97] where it is shown that affine reconstruction is possible from a pair of images.

Samsung/Google, EX1010, p. 279

10

3D Reconstruction of Cameras and Structure

This chapter describes how and to what extent the spatial layout of a scene and the cameras can be recovered from two views. Suppose that a set of image correspondences $\mathbf{x}_i \leftrightarrow \mathbf{x}'_i$ are given. It is assumed that these correspondences come from a set of 3D points $\mathbf{X}_i$, which are unknown. Similarly, the position, orientation and calibration of the cameras are not known. The reconstruction task is to find the camera matrices $\mathtt{P}$ and $\mathtt{P}'$, as well as the 3D points $\mathbf{X}_i$ such that

$$\mathbf{x}_i = \mathtt{P}\mathbf{X}_i \quad \mathbf{x}'_i = \mathtt{P}'\mathbf{X}_i \quad \text{for all } i.$$

Given too few points, this task is not possible. However, if there are sufficiently many point correspondences to allow the fundamental matrix to be computed uniquely, then the scene may be reconstructed up to a projective ambiguity. This is a very significant result, and one of the major achievements of the uncalibrated approach.

The ambiguity in the reconstruction may be reduced if additional information is supplied on the cameras or scene. We describe a two-stage approach where the ambiguity is first reduced to affine, and second to metric; each stage requiring information of the appropriate class.

10.1 Outline of reconstruction method

We describe a method for reconstruction from two views as follows.

(i) Compute the fundamental matrix from point correspondences.
(ii) Compute the camera matrices from the fundamental matrix.
(iii) For each point correspondence $\mathbf{x}_i \leftrightarrow \mathbf{x}'_i$, compute the point in space that projects to these two image points.

Many variants on this method are possible. For instance, if the cameras are calibrated, then one will compute the essential matrix instead of the fundamental matrix. Furthermore, one may use information about the motion of the camera, scene constraints or partial camera calibration to obtain refinements of the reconstruction.

Each of the steps of this reconstruction method will be discussed briefly in the following paragraphs. The method described is no more than a conceptual approach to reconstruction. The reader is warned not to implement a reconstruction method based solely on the description given in this section. For real images where measurements

Samsung/Google, EX1010, p. 280

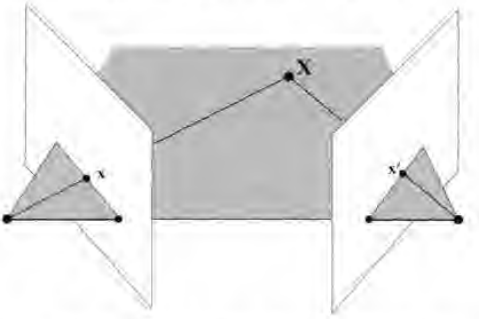


Fig. 10.1. **Triangulation.** *The image points* $\mathbf{x}$ *and* $\mathbf{x}'$ *back project to rays. If the epipolar constraint* $\mathbf{x}'^{\mathsf{T}}\mathtt{F}\mathbf{x} = 0$ *is satisfied, then these two rays lie in a plane, and so intersect in a point* $\mathbf{X}$ *in 3-space.*

are "noisy" preferred methods for reconstruction, based on this general outline, are described in chapter 11 and chapter 12.

Computation of the fundamental matrix. Given a set of correspondences $\mathbf{x}_i \leftrightarrow \mathbf{x}'_i$ in two images the fundamental matrix $\mathtt{F}$ satisfies the condition $\mathbf{x}'^{\mathsf{T}}_i\mathtt{F}\mathbf{x}_i = 0$ for all i. With the $\mathbf{x}_i$ and $\mathbf{x}'_i$ known, this equation is linear in the (unknown) entries of the matrix $\mathtt{F}$. In fact, each point correspondence generates one linear equation in the entries of $\mathtt{F}$. Given at least 8 point correspondences it is possible to solve linearly for the entries of $\mathtt{F}$ up to scale (a non-linear solution is available for 7 point correspondences). With more than 8 equations a least-squares solution is found. This is the general principle of a method for computing the fundamental matrix.

Recommended methods of computing the fundamental matrix from a set of point correspondences will be described later in chapter 11.

Computation of the camera matrices. A pair of camera matrices $\mathtt{P}$ and $\mathtt{P}'$ corresponding to the fundamental matrix $\mathtt{F}$ are easily computed using the direct formula in result 9.14.

Triangulation. Given the camera matrices $\mathtt{P}$ and $\mathtt{P}'$, let $\mathbf{x}$ and $\mathbf{x}'$ be two points in the two images that satisfy the epipolar constraint, $\mathbf{x}'^{\mathsf{T}}\mathtt{F}\mathbf{x} = 0$. As shown in chapter 9 this constraint may be interpreted geometrically in terms of the rays in space corresponding to the two image points. In particular it means that $\mathbf{x}'$ lies on the epipolar line $\mathtt{F}\mathbf{x}$. In turn this means that the two rays back-projected from image points $\mathbf{x}$ and $\mathbf{x}'$ lie in a common epipolar plane, that is, a plane passing through the two camera centres. Since the two rays lie in a plane, they will intersect in some point. This point $\mathbf{X}$ projects via the two cameras to the points $\mathbf{x}$ and $\mathbf{x}'$ in the two images. This is illustrated in figure 10.1.

The only points in 3-space that cannot be determined from their images are points on the baseline between the two cameras. In this case the back-projected rays are collinear (both being equal to the baseline) and intersect along their whole length. Thus, the point

Samsung/Google, EX1010, p. 281

$\mathbf{X}$ cannot be uniquely determined. Points on the baseline project to the epipoles in both images.

Numerically stable methods of actually determining the point $\mathbf{X}$ at the intersection of the two rays back-projected from $\mathbf{x}$ and $\mathbf{x}'$ will be described later in chapter 12.

10.2 Reconstruction ambiguity

In this section we discuss the inherent ambiguities involved in reconstruction of a scene from point correspondences. This topic will be discussed in a general context, without reference to a specific method of carrying out the reconstruction.

Without some knowledge of a scene's placement with respect to a 3D coordinate frame, it is generally not possible to reconstruct the absolute position or orientation of a scene from a pair of views (or in fact from any number of views). This is true independently of any knowledge which may be available about the internal parameters of the cameras, or their relative placement. For instance the exact latitude and longitude of the scene in figure 9.8(p248) (or any scene) cannot be computed, nor is it possible to determine whether the corridor runs north-south or east-west. This may be expressed by saying that the scene is determined at best up to a Euclidean transformation (rotation and translation) with respect to the world frame.

Only slightly less obvious is the fact that the overall scale of the scene cannot be determined. Considering figure 9.8(p248) once more, it is impossible based on the images alone to determine the width of the corridor. It may be two metres, one metre. It is even possible that this is an image of a doll's house and the corridor is 10 cm wide. Our common experience leads us to expect that ceilings are approximately 3m from the floor, which allows us to perceive the real scale of the scene. This extra information is an example of subsidiary knowledge of the scene not derived from image measurements. Without such knowledge therefore the scene is determined by the image only up to a similarity transformation (rotation, translation and scaling).

To give a mathematical basis to this observation, let $\mathbf{X}_i$ be a set of points and $\mathtt{P}, \mathtt{P}'$ be a pair of cameras projecting $\mathbf{X}_i$ to image points $\mathbf{x}_i$ and $\mathbf{x}'_i$. The points $\mathbf{X}_i$ and the camera pair constitute a reconstruction of the scene from the image correspondences. Now let

$$\mathtt{H}_\mathtt{S} = \begin{bmatrix} \mathtt{R} & \mathbf{t} \\ \mathbf{0}^\mathsf{T} & \lambda \end{bmatrix}$$

be any similarity transformation: $\mathtt{R}$ is a rotation, $\mathbf{t}$ a translation and λ^{-1} represents overall scaling. Replacing each point $\mathbf{X}_i$ by $\mathtt{H}_\mathtt{S}\mathbf{X}_i$ and cameras $\mathtt{P}$ and $\mathtt{P}'$ by $\mathtt{P}\mathtt{H}_\mathtt{S}^{-1}$ and $\mathtt{P}'\mathtt{H}_\mathtt{S}^{-1}$ respectively does not change the observed image points, since $\mathtt{P}\mathbf{X}_i = (\mathtt{P}\mathtt{H}_\mathtt{S}^{-1})(\mathtt{H}_\mathtt{S}\mathbf{X}_i)$. Furthermore, if $\mathtt{P}$ is decomposed as $\mathtt{P} = \mathtt{K}[\mathtt{R}_\mathtt{P} \mid \mathbf{t}_\mathtt{P}]$, then one computes

$$\mathtt{P}\mathtt{H}_\mathtt{S}^{-1} = \mathtt{K}[\mathtt{R}_\mathtt{P}\mathtt{R}^{-1} \mid \mathbf{t}']$$

for some $\mathbf{t}'$ that we do not need to compute more exactly. This result shows that multiplying by $\mathtt{H}_\mathtt{S}^{-1}$ does not change the calibration matrix of $\mathtt{P}$. Consequently this ambiguity of reconstruction exists even for calibrated cameras. It was shown by Longuet-Higgins

Samsung/Google, EX1010, p. 282

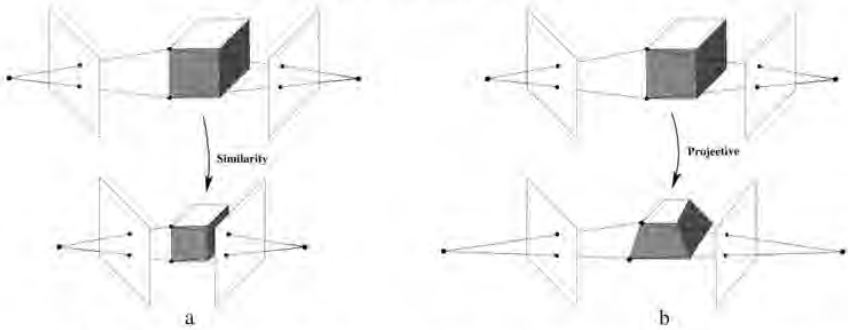


Fig. 10.2. **Reconstruction ambiguity.** *(a) If the cameras are calibrated then any reconstruction must respect the angle between rays measured in the image. A similarity transformation of the structure and camera positions does not change the measured angle. The angle between rays and the baseline (epipoles) is also unchanged. (b) If the cameras are uncalibrated then reconstructions must only respect the image points (the intersection of the rays with the image plane). A projective transformation of the structure and camera positions does not change the measured points, although the angle between rays is altered. The epipoles are also unchanged (intersection with baseline).*

([LonguetHiggins-81]) that for calibrated cameras, this is the only ambiguity of reconstruction. Thus for calibrated cameras, reconstruction is possible *up to a similarity transformation.* This is illustrated in figure 10.2a.

Projective ambiguity. If nothing is known of the calibration of either camera, nor the placement of one camera with respect to the other, then the ambiguity of reconstruction is expressed by an arbitrary projective transformation. In particular, if $\mathtt{H}$ is any 4×4 invertible matrix, representing a projective transformation of $\mathbb{P}^3$, then replacing points $\mathbf{X}_i$ by $\mathtt{H}\mathbf{X}_i$ and matrices $\mathtt{P}$ and $\mathtt{P}'$ by $\mathtt{P}\mathtt{H}^{-1}$ and $\mathtt{P}'\mathtt{H}^{-1}$ (as in the previous paragraph) does not change the image points. This shows that the points $\mathbf{X}_i$ and the cameras can be determined at best only up to a projective transformation. It is an important result, proved in this chapter (section 10.3), that this is the only ambiguity in the reconstruction of points from two images. Thus reconstruction from uncalibrated cameras is possible *up to a projective transformation.* This is illustrated in figure 10.2b.

Other types of reconstruction ambiguity result from certain assumptions on the types of motion, or partial knowledge of the cameras. For instance,

(i) If the two cameras are related via a translational motion, without change of calibration, then reconstruction is possible up to an affine transformation.
(ii) If the two cameras are calibrated apart from their focal lengths, then reconstruction is still possible up to a similarity transformation.

These two cases will be considered later in section 10.4.1 and example 19.8(p472), respectively.

Terminology. In any reconstruction problem derived from real data, consisting of point correspondences $\mathbf{x}_i \leftrightarrow \mathbf{x}'_i$, there exists a **true** reconstruction consisting of the actual points $\bar{\mathbf{X}}_i$ and actual cameras $\bar{\mathtt{P}}, \bar{\mathtt{P}}'$ that generated the measured observations. The

Samsung/Google, EX1010, p. 283

reconstructed point set $\mathbf{X}_i$ and cameras differ from the true reconstruction by a transformation belonging to a given class or group (for instance a similarity, projective or affine transformation). One speaks of *projective reconstruction*, *affine reconstruction*, *similarity reconstruction*, and so on, to indicate the type of transformation involved. However, the term *metric reconstruction* is normally used in preference to *similarity reconstruction*, being identical in meaning. The term indicates that metric properties, such as angles between lines and ratios of lengths, can be measured on the reconstruction and have their veridical values (since these are similarity invariants). In addition, the term *Euclidean reconstruction* is frequently used in the published literature to mean the same thing as a similarity or metric reconstruction, since true Euclidean reconstruction (including determination of overall scale) is not possible without extraneous information.

10.3 The projective reconstruction theorem

In this section the basic theorem of projective reconstruction from uncalibrated cameras is proved. Informally, the theorem may be stated as follows.

- *If a set of point correspondences in two views determine the fundamental matrix uniquely, then the scene and cameras may be reconstructed from these correspondences alone, and any two such reconstructions from these correspondences are projectively equivalent.*

Points lying on the line joining the two camera centres must be excluded, since such points cannot be reconstructed uniquely even if the camera matrices are determined. The formal statement is:

Theorem 10.1 (Projective reconstruction theorem). *Suppose that* $\mathbf{x}_i \leftrightarrow \mathbf{x}'_i$ *is a set of correspondences between points in two images and that the fundamental matrix* $\mathtt{F}$ *is uniquely determined by the condition* $\mathbf{x}'^{\mathsf{T}}_i \mathtt{F} \mathbf{x}_i = 0$ *for all* i. *Let* $(\mathtt{P}_1, \mathtt{P}'_1, \{\mathbf{X}_{1i}\})$ *and* $(\mathtt{P}_2, \mathtt{P}'_2, \{\mathbf{X}_{2i}\})$ *be two reconstructions of the correspondences* $\mathbf{x}_i \leftrightarrow \mathbf{x}'_i$. *Then there exists a non-singular matrix* $\mathtt{H}$ *such that* $\mathtt{P}_2 = \mathtt{P}_1\mathtt{H}^{-1}$, $\mathtt{P}'_2 = \mathtt{P}'_1\mathtt{H}^{-1}$ *and* $\mathbf{X}_{2i} = \mathtt{H}\mathbf{X}_{1i}$ *for all* i, **except** *for those* i *such that* $\mathtt{F}\mathbf{x}_i = \mathbf{x}'^{\mathsf{T}}_i \mathtt{F} = \mathbf{0}$.

Proof. Since the fundamental matrix is uniquely determined by the point correspondences, one deduces that $\mathtt{F}$ is the fundamental matrix corresponding to the camera pair $(\mathtt{P}_1, \mathtt{P}'_1)$ and also to $(\mathtt{P}_2, \mathtt{P}'_2)$. According to theorem 9.10(*p*254) there is a projective transformation $\mathtt{H}$ such that $\mathtt{P}_2 = \mathtt{P}_1\mathtt{H}^{-1}$ and $\mathtt{P}'_2 = \mathtt{P}'_1\mathtt{H}^{-1}$ as required.

As for the points, one observes that $\mathtt{P}_2(\mathtt{H}\mathbf{X}_{1i}) = \mathtt{P}_1\mathtt{H}^{-1}\mathtt{H}\mathbf{X}_{1i} = \mathtt{P}_1\mathbf{X}_{1i} = \mathbf{x}_i$. On the other hand $\mathtt{P}_2\mathbf{X}_{2i} = \mathbf{x}_i$, so $\mathtt{P}_2(\mathtt{H}\mathbf{X}_{1i}) = \mathtt{P}_2\mathbf{X}_{2i}$. Thus both $\mathtt{H}\mathbf{X}_{1i}$ and $\mathbf{X}_{2i}$ map to the same point $\mathbf{x}_i$ under the action of the camera $\mathtt{P}_2$. It follows that both $\mathtt{H}\mathbf{X}_{1i}$ and $\mathbf{X}_{2i}$ lie on the same ray through the camera centre of $\mathtt{P}_2$. Similarly, it may be deduced that these two points lie on the same ray through the camera centre of $\mathtt{P}'_2$. There are two possibilities: either $\mathbf{X}_{2i} = \mathtt{H}\mathbf{X}_{1i}$ as required, or they are distinct points lying on the line joining the two camera centres. In this latter case, the image points $\mathbf{x}_i$ and $\mathbf{x}'_i$ coincide with the epipoles in the two images, and so $\mathtt{F}\mathbf{x}_i = \mathbf{x}'^{\mathsf{T}}_i \mathtt{F} = \mathbf{0}$. □

Samsung/Google, EX1010, p. 284

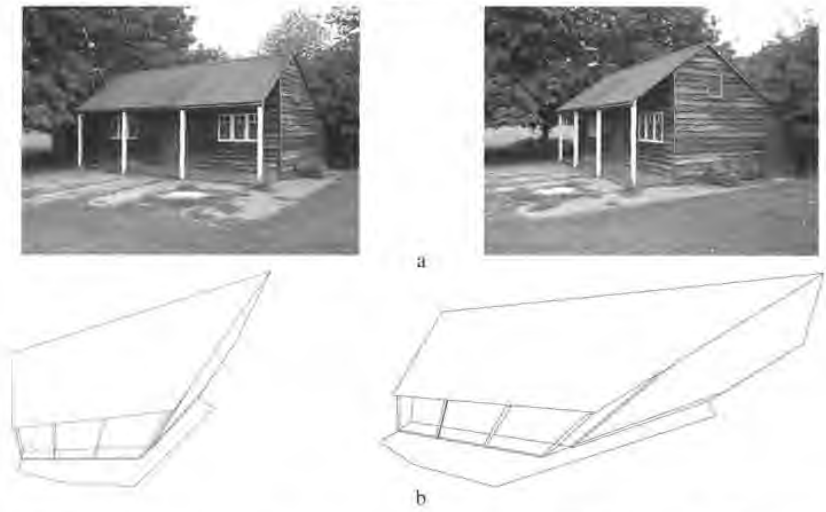


Fig. 10.3. **Projective reconstruction.** *(a) Original image pair. (b) 2 views of a 3D projective reconstruction of the scene. The reconstruction requires no information about the camera matrices, or information about the scene geometry. The fundamental matrix* F *is computed from point correspondences between the images, camera matrices are retrieved from* F*, and then 3D points are computed by triangulation from the correspondences. The lines of the wireframe link the computed 3D points.*

This is an enormously significant result, since it implies that one may compute a projective reconstruction of a scene from two views based on image correspondences alone, without knowing anything about the calibration or pose of the two cameras involved. In particular the true reconstruction is within a projective transformation of the projective reconstruction. Figure 10.3 shows an example of 3D structure computed as part of a projective reconstruction from two images.

In more detail suppose the true Euclidean reconstruction is $(\mathtt{P}_{\mathrm{E}}, \mathtt{P}'_{\mathrm{E}}, \{\mathbf{X}_{\mathrm{E}i}\})$ and the projective reconstruction is $(\mathtt{P}, \mathtt{P}', \{\mathbf{X}_i\})$, then the reconstructions are related by a non-singular matrix H such that

$$\mathtt{P}_{\mathrm{E}} = \mathtt{P}\mathtt{H}^{-1}, \quad \mathtt{P}'_{\mathrm{E}} = \mathtt{P}'\mathtt{H}^{-1}, \text{ and } \mathbf{X}_{\mathrm{E}i} = \mathtt{H}\mathbf{X}_i \tag{10.1}$$

where H is a 4×4 homography matrix which is unknown but the same for all points.

For some applications projective reconstruction is all that is required. For example, questions such as "at what point does a line intersect a plane?", "what is the mapping between two views induced by particular surfaces, such as a plane or quadric?" can be dealt with directly from the projective reconstruction. Furthermore it will be seen in the sequel that obtaining a projective reconstruction of a scene is the first step towards affine or metric reconstruction.

10.4 Stratified reconstruction

The "stratified" approach to reconstruction is to begin with a projective reconstruction and then to refine it progressively to an affine and finally a metric reconstruction, if

Samsung/Google, EX1010, p. 285

possible. Of course, as has just been seen, affine and metric reconstruction are not possible without further information either about the scene, the motion or the camera calibration.

10.4.1 The step to affine reconstruction

The essence of affine reconstruction is to locate the plane at infinity by some means, since this knowledge is equivalent to an affine reconstruction. This equivalence is explained in the 2D case in section 2.7(*p*47). To see this equivalence for reconstruction, suppose we have determined a projective reconstruction of a scene, consisting of a triple $(\mathtt{P}, \mathtt{P}', \{\mathbf{X}_i\})$. Suppose further that by some means a certain plane $\boldsymbol{\pi}$ has been identified as the true plane at infinity. The plane $\boldsymbol{\pi}$ is expressed as a 4-vector in the coordinate frame of the projective reconstruction. In the true reconstruction, $\boldsymbol{\pi}$ has coordinates $(0, 0, 0, 1)^\mathsf{T}$, and we may find a projective transformation that maps $\boldsymbol{\pi}$ to $(0, 0, 0, 1)^\mathsf{T}$. Considering the way a projective transformation acts on planes, we want to find $\mathtt{H}$ such that $\mathtt{H}^{-\mathsf{T}}\boldsymbol{\pi} = (0, 0, 0, 1)^\mathsf{T}$. Such a transformation is given by

$$\mathtt{H} = \begin{bmatrix} \mathtt{I} & \mathbf{0} \\ \boldsymbol{\pi}^\mathsf{T} & \end{bmatrix}. \tag{10.2}$$

Indeed, it is immediately verified that $\mathtt{H}^\mathsf{T}(0, 0, 0, 1)^\mathsf{T} = \boldsymbol{\pi}$, and thus $\mathtt{H}^{-\mathsf{T}}\boldsymbol{\pi} = (0, 0, 0, 1)^\mathsf{T}$, as desired. The transformation $\mathtt{H}$ is now applied to all points and the two cameras. Notice, however that this formula will not work if the final coordinate of $\boldsymbol{\pi}^\mathsf{T}$ is zero. In this case, one may compute a suitable $\mathtt{H}$ by computing $\mathtt{H}^{-\mathsf{T}}$ as a Householder matrix (A4.2–*p*580) such that $\mathtt{H}^{-\mathsf{T}}\boldsymbol{\pi} = (0, 0, 0, 1)^\mathsf{T}$.

At this point, the reconstruction that one has is not necessarily the true reconstruction – all one knows is that the plane at infinity is correctly placed. The present reconstruction differs from the true reconstruction by a projective transformation that fixes the plane at infinity. However, according to result 3.7(*p*80), a projective transformation fixing the plane at infinity is an affine transformation. Hence the reconstruction differs by an affine transformation from the true reconstruction – it is an *affine reconstruction*.

An affine reconstruction may well be sufficient for some applications. For example, the mid-point of two points and the centroid of a set of points may now be computed, and lines constructed parallel to other lines and to planes. Such computations are not possible from a projective reconstruction.

As has been stated, the plane at infinity cannot be identified unless some extra information is given. We will now give several examples of the type of information that suffices for this identification.

Translational motion

Consider the case where the camera is known to undergo a purely translational motion. In this case, it is possible to carry out affine reconstruction from two views. A simple way of seeing this is to observe that a point $\mathbf{X}$ on the plane at infinity will map to the same point in two images related by a translation. This is easily verified formally. It is also part of our common experience that as one moves in a straight line (for instance in

Samsung/Google, EX1010, p. 286

a car on a straight road), objects at a great distance (such as the moon) do not appear to move – only the nearby objects move past the field of view. This being so, one may invent any number of matched points $\mathbf{x}_i \leftrightarrow \mathbf{x}_i$ where a point in one image corresponds with the same point in the other image. Note that one does not actually have to observe such a correspondence in the two images – any point and the same point in the other image will do. Given a projective reconstruction, one may then reconstruct the point $\mathbf{X}_i$ corresponding to the match $\mathbf{x}_i \leftrightarrow \mathbf{x}_i$. Point $\mathbf{X}_i$ will lie on the plane at infinity. From three such points one can get three points on the plane at infinity – sufficient to determine it uniquely.

Although this argument gives a constructive proof that affine reconstruction is possible from a translating camera, this does not mean that this is the best way to proceed numerically. In fact in this case, the assumption of translational motion implies a very restricted form for the fundamental matrix – it is skew-symmetric as shown in section 9.3.1. This special form should be taken into account when solving for the fundamental matrix.

Result 10.2. *Suppose the motion of the cameras is a pure translation with no rotation and no change in the internal parameters. As shown in example 9.6(*p249*)* $\mathtt{F} = [\mathbf{e}]_\times = [\mathbf{e}']_\times$*, and for an affine reconstruction one may choose the two cameras as* $\mathtt{P} = [\mathtt{I} \mid \mathbf{0}]$ *and* $\mathtt{P}' = [\mathtt{I} \mid \mathbf{e}']$.

Scene constraints

Scene constraints or conditions may also be used to obtain an affine reconstruction. As long as three points can be identified that are known to lie on the plane at infinity, then that plane may be identified, and the reconstruction transformed to an affine reconstruction.

Parallel lines. The most obvious such condition is the knowledge that 3D lines are in reality parallel. The intersection of the two parallel lines in space gives a point on the plane at infinity. The image of this point is the vanishing point of the line, and is the point of intersection of the two imaged lines. Suppose that three sets of parallel lines can be identified in the scene. Each set intersects in a point on the plane at infinity. Provided each set has a different direction, the three points will be distinct. Since three points determine a plane, this information is sufficient to identify the plane π.

The best way of actually computing the intersection of lines in space is a somewhat delicate problem, since in the presence of noise, lines that are intended to intersect rarely do. It is discussed in some detail in chapter 12. Correct numerical procedures for computing the plane are given in chapter 13. An example of an affine reconstruction computed from three sets of parallel scene lines is given in figure 10.4.

Note that it is not necessary to find the vanishing point in both images. Suppose the vanishing point $\mathbf{v}$ is computed from imaged parallel lines in the first image, and $\mathbf{l}'$ is a corresponding line in the second image. Vanishing points satisfy the epipolar constraint, so the corresponding vanishing point $\mathbf{v}'$ in the second image may be computed as the intersection of $\mathbf{l}'$ and the epipolar line $\mathtt{F}\mathbf{v}$ of $\mathbf{v}$. The construction of the

Samsung/Google, EX1010, p. 287

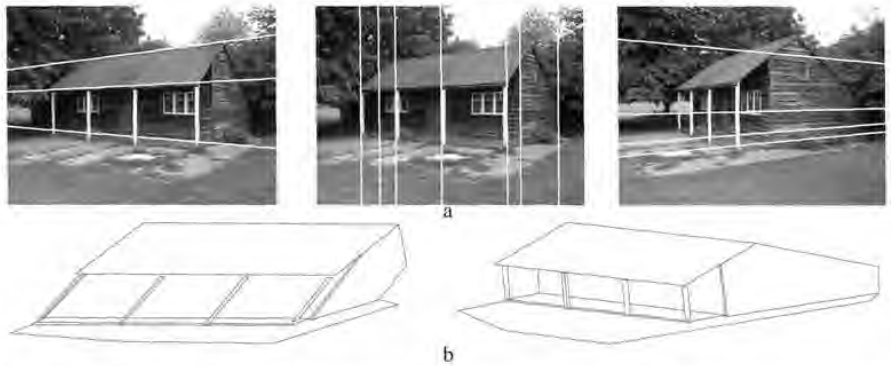


Fig. 10.4. **Affine reconstruction.** *The projective reconstruction of figure 10.3 may be upgraded to affine using parallel scene lines. (a) There are 3 sets of parallel lines in the scene, each set with a different direction. These 3 sets enable the position of the plane at infinity, π_∞, to be computed in the projective reconstruction. The wireframe projective reconstruction of figure 10.3 is then affinely rectified using the homography (10.2). (b) Shows two orthographic views of the wireframe affine reconstruction. Note that parallel scene lines are parallel in the reconstruction, but lines that are perpendicular in the scene are not perpendicular in the reconstruction.*

3-space point $\mathbf{X}$ can be neatly expressed algebraically as the solution of the equations $([\mathbf{v}]_\times \mathtt{P})\mathbf{X} = \mathbf{0}$ and $(\mathbf{l}'^\mathsf{T}\mathtt{P}')\mathbf{X} = 0$. These equations expresses the fact that $\mathbf{X}$ maps to $\mathbf{v}$ in the first image, and to a point on $\mathbf{l}'$ in the second image.

Distance ratios on a line. An alternative to computing vanishing points as the intersection of imaged parallel scene lines is to use knowledge of affine length ratios in the scene. For example, given two intervals on a line with a known length ratio, the point at infinity on the line may be determined. This means that from an image of a line on which a world distance ratio is known, for example that three points are equally spaced, the vanishing point may be determined. This computation, and other means of computing vanishing points and vanishing lines, are described in section 2.7(*p*47).

The infinite homography

Once the plane at infinity has been located, so that we have an affine reconstruction, then we also have an image-to-image map called the "infinite homography". This map, which is a 2D homography , is described in greater detail in chapter 13. Briefly, it is the map that transfers points from the $\mathtt{P}$ image to the $\mathtt{P}'$ image via the plane at infinity as follows: the ray corresponding to a point $\mathbf{x}$ is extended to meet the plane at infinity in a point $\mathbf{X}$; this point is projected to a point $\mathbf{x}'$ in the other image. The homography from $\mathbf{x}$ to $\mathbf{x}'$ is written as $\mathbf{x}' = \mathtt{H}_\infty\mathbf{x}$.

Having an affine reconstruction is equivalent to knowing the infinite homography as will now be shown. Given two cameras $\mathtt{P} = [\mathtt{M} \mid \mathbf{m}]$ and $\mathtt{P}' = [\mathtt{M}' \mid \mathbf{m}']$ of an affine reconstruction, the infinite homography is given by $\mathtt{H}_\infty = \mathtt{M}'\mathtt{M}^{-1}$. This is because a point $\mathbf{X} = (\tilde{\mathbf{X}}^\mathsf{T}, 0)^\mathsf{T}$ on the plane at infinity maps to $\mathbf{x} = \mathtt{M}\tilde{\mathbf{X}}$ in one image and $\mathbf{x}' = \mathtt{M}'\tilde{\mathbf{X}}$ in the other, so $\mathbf{x}' = \mathtt{M}'\mathtt{M}^{-1}\mathbf{x}$ for points on π_∞. Furthermore, it may be verified that

Samsung/Google, EX1010, p. 288

this is unchanged by a 3-space affine transformation of the cameras. Hence, the infinite homography may be computed explicitly from an affine reconstruction, and vice versa:

Result 10.3. *If an affine reconstruction has been obtained in which the camera matrices are* $\mathtt{P} = [\mathtt{I} \mid \mathbf{0}]$ *and* $\mathtt{P}' = [\mathtt{M}' \mid \mathbf{e}']$*, then the infinite homography is given by* $\mathtt{H}_\infty = \mathtt{M}'$*. Conversely, if the infinite homography* $\mathtt{H}_\infty$ *has been obtained, then the cameras of an affine reconstruction may be chosen as* $\mathtt{P} = [\mathtt{I} \mid \mathbf{0}]$ *and* $\mathtt{P}' = [\mathtt{H}_\infty \mid \mathbf{e}']$*.*

The infinite homography may be computed directly from corresponding image entities, rather than indirectly from an affine reconstruction. For example, $\mathtt{H}_\infty$ can be computed from the correspondence of three vanishing points together with $\mathtt{F}$, or the correspondence of a vanishing line and vanishing point, together with $\mathtt{F}$. The correct numerical procedure for these computations is given in chapter 13. However, such direct computations are completely equivalent to determining π_∞ in a projective reconstruction.

One of the cameras is affine

Another important case in which affine reconstruction is possible is when one of the cameras is known to be an affine camera as defined in section 6.3.1(p166). To see that this implies that affine reconstruction is possible, refer to section 6.3.5(p172) where it was shown that the principal plane of an affine camera is the plane at infinity. Hence to convert a projective reconstruction to an affine reconstruction, it is sufficient to find the principal plane of the camera supposed to be affine and map it to the plane $(0, 0, 0, 1)^\mathsf{T}$. Recall (section 6.2(p158)) that the principal plane of a camera is simply the third row of the camera matrix. For example, consider a projective reconstruction with camera matrices $\mathtt{P} = [\mathtt{I} \mid \mathbf{0}]$ and $\mathtt{P}'$, for which the first camera is supposed to be affine. To map the third row of $\mathtt{P}$ to $(0, 0, 0, 1)$ it is sufficient to swap the last two columns of the two camera matrices, while at the same time swapping the 3rd and 4th coordinates of each $\mathbf{X}_i$. This is a projective transformation corresponding to a permutation matrix $\mathtt{H}$. This shows:

Result 10.4. *Let* $(\mathtt{P}, \mathtt{P}', \{\mathbf{X}_i\})$ *be a projective reconstruction from a set of point correspondences for which* $\mathtt{P} = [\mathtt{I} \mid \mathbf{0}]$*. Suppose in truth,* $\mathtt{P}$ *is known to be an affine camera, then an affine reconstruction is obtained by swapping the last two columns of* $\mathtt{P}$ *and* $\mathtt{P}'$ *and the last two coordinates of each* $\mathbf{X}_i$*.*

Note that the condition that one of the cameras is affine places no restriction on the fundamental matrix, since any canonical camera pair $\mathtt{P} = [\mathtt{I} \mid \mathbf{0}]$ and $\mathtt{P}'$ can be transformed to a pair in which $\mathtt{P}$ is affine. If both the cameras are known to be affine, then it will be seen that the fundamental matrix has the restricted form given in (14.1–p345). In this case, for numerical stability, one must solve for the fundamental matrix enforcing this special form of the fundamental matrix.

Of course there is no such thing as a real affine camera – the affine camera model is an approximation which is only valid when the set of points seen in the image has small depth variation compared with the distance from the camera. Nevertheless, an assumption of an affine camera may be useful to effect the significant restriction from projective to affine reconstruction.

Samsung/Google, EX1010, p. 289

10.4.2 The step to metric reconstruction

Just as the key to affine reconstruction is the identification of the plane at infinity, the key to metric reconstruction is the identification of the absolute conic (section 3.6-($p81$)). Since the absolute conic, Ω_∞, is a planar conic, lying in the plane at infinity, identifying the absolute conic implies identifying the plane at infinity.

In a stratified approach, one proceeds from projective to affine to metric reconstruction, so one knows the plane at infinity before finding the absolute conic. Suppose one has identified the absolute conic on the plane at infinity. In principle the next step is to apply an affine transformation to the affine reconstruction such that the identified absolute conic is mapped to the absolute conic in the standard Euclidean frame (it will then have the equation $X_1^2 + X_2^2 + X_3^2 = 0$, on π_∞). The resulting reconstruction is then related to the true reconstruction by a projective transformation which fixes the absolute conic. It follows from result 3.9($p82$) that the projective transformation is a similarity transformation, so we have achieved a metric reconstruction.

In practice the easiest way to accomplish this is to consider the image of the absolute conic in one of the images. The image of the absolute conic (as any conic) is a conic in the image. The back-projection of this conic is a cone, which will meet the plane at infinity in a single conic, which therefore defines the absolute conic. Remember that the image of the absolute conic is a property of the image itself, and like any image point, line or other feature, is not dependent on any particular reconstruction, hence it is unchanged by 3D transformations of the reconstruction.

Suppose that in the affine reconstruction the image of the absolute conic as seen by the camera with matrix $\mathtt{P} = [\mathtt{M} \mid \mathbf{m}]$ is a conic $\boldsymbol{\omega}$. We will show how $\boldsymbol{\omega}$ may be used to define the homography $\mathtt{H}$ which transforms the affine reconstruction to a metric reconstruction:

Result 10.5. *Suppose that the image of the absolute conic is known in some image to be $\boldsymbol{\omega}$, and one has an affine reconstruction in which the corresponding camera matrix is given by $\mathtt{P} = [\mathtt{M} \mid \mathtt{m}]$. Then, the affine reconstruction may be transformed to a metric reconstruction by applying a 3D transformation of the form*

$$\mathtt{H} = \begin{bmatrix} \mathtt{A}^{-1} & \\ & 1 \end{bmatrix}$$

where $\mathtt{A}$ is obtained by Cholesky factorization from the equation $\mathtt{A}\mathtt{A}^\top = (\mathtt{M}^\top \boldsymbol{\omega} \mathtt{M})^{-1}$.

Proof. Under the transformation $\mathtt{H}$, the camera matrix $\mathtt{P}$ is transformed to a matrix $\mathtt{P}_M = \mathtt{P}\mathtt{H}^{-1} = [\mathtt{M}_M \mid \mathbf{m}_M]$. If $\mathtt{H}^{-1}$ is of the form

$$\mathtt{H}^{-1} = \begin{bmatrix} \mathtt{A} & \mathbf{0} \\ \mathbf{0}^\top & 1 \end{bmatrix}$$

then $\mathtt{M}_M = \mathtt{M}\mathtt{A}$. However, the image of the absolute conic is related to the camera matrix $\mathtt{P}_M$ of a Euclidean frame by the relationship

$$\boldsymbol{\omega}^* = \mathtt{M}_M \mathtt{M}_M^\top \ .$$

Samsung/Google, EX1010, p. 290

This is because the camera matrix may be decomposed as $\mathtt{M}_M = \mathtt{KR}$, and from (8.11–$p210$) $\omega^* = \omega^{-1} = \mathtt{KK}^\mathsf{T}$. Combining this with $\mathtt{M}_M = \mathtt{MA}$ gives $\omega^{-1} = \mathtt{MAA}^\mathsf{T}\mathtt{M}^\mathsf{T}$, which may be rearranged as $\mathtt{AA}^\mathsf{T} = (\mathtt{M}^\mathsf{T}\omega\mathtt{M})^{-1}$. A particular value of A that satisfies this relationship is found by taking the Cholesky factorization of $(\mathtt{M}^\mathsf{T}\omega\mathtt{M})^{-1}$. This latter matrix is guaranteed to be positive-definite (see result A4.5($p582$)), otherwise no such matrix A will exist, and metric reconstruction will not be possible. □

This approach to metric reconstruction relies on identifying the image of the absolute conic. There are various ways of doing this and these are discussed next. Three sources of constraint on the image of the absolute conic are given, and in practice a combination of these constraints is used.

1. Constraints arising from scene orthogonality. Pairs of vanishing points, $\mathbf{v}_1$ and $\mathbf{v}_2$, arising from orthogonal scene lines place a single linear constraint on ω:

$$\mathbf{v}_1^\mathsf{T}\omega\mathbf{v}_2 = 0.$$

Similarly, a vanishing point $\mathbf{v}$ and a vanishing line $\mathbf{l}$ arising from a direction and plane which are orthogonal place two constraints on ω:

$$\mathbf{l} = \omega\mathbf{v}.$$

A common example is the vanishing point for the vertical direction and a vanishing line from the horizontal ground plane. Finally an imaged scene plane containing metric information, such as a square grid, places two constraints on ω.

2. Constraints arising from known internal parameters. If the calibration matrix of a camera is equal to K, then the image of the absolute conic is $\omega = \mathtt{K}^{-\mathsf{T}}\mathtt{K}^{-1}$. Thus, knowledge of the internal parameters (6.10–$p157$) contained in K may be used to constrain or determine the elements of ω. In the case where K is known to have zero skew ($s = 0$),

$$\omega_{12} = \omega_{21} = 0$$

and if the pixels are square (zero skew and $\alpha_x = \alpha_y$) then

$$\omega_{11} = \omega_{22}.$$

These first two sources of constraint are discussed in detail in section 8.8($p223$) on single view calibration, where examples are given of calibrating a camera solely from such information. Here there is an additional source of constraints available arising from the multiple views.

3. Constraints arising from the same cameras in all images. One of the properties of the absolute conic is that its projection into an image depends only on the calibration matrix of the camera, and not on the position or orientation of the camera. In the case where both cameras P and P′ have the same calibration matrix (usually meaning that both the images were taken with the same camera with different pose) one has that $\omega = \omega'$, that is the image of the absolute conic is the same in both images. Given

Samsung/Google, EX1010, p. 291

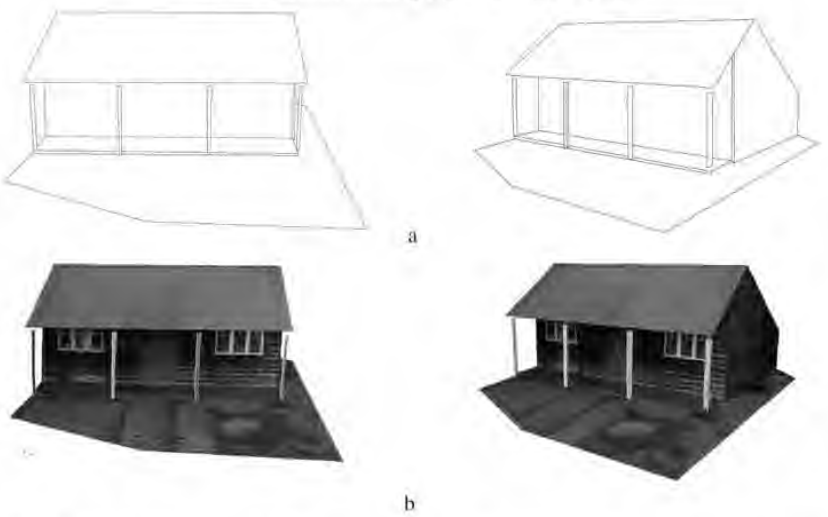


Fig. 10.5. **Metric reconstruction.** *The affine reconstruction of figure 10.4 is upgraded to metric by computing the image of the absolute conic. The information used is the orthogonality of the directions of the parallel line sets shown in figure 10.4, together with the constraint that both images have square pixels. The square pixel constraint is transferred from one image to the other using* $\mathtt{H}_\infty$. *(a) Two views of the metric reconstruction. Lines which are perpendicular in the scene are perpendicular in the reconstruction and also the aspect ratio of the sides of the house is veridical. (b) Two views of a texture mapped piecewise planar model built from the wireframes.*

sufficiently many images, one may use this property to obtain a metric reconstruction from an affine reconstruction. This method of metric reconstruction, and its use for self-calibration of a camera, will be treated in greater detail in chapter 19. For now, we give just the general principle.

Since the absolute conic lies on the plane at infinity, its image may be transferred from one view to the other via the infinite homography. This implies an equation (see result 2.13(p37))

$$\omega' = \mathtt{H}_\infty^{-\mathsf{T}} \omega \mathtt{H}_\infty^{-1} \tag{10.3}$$

where ω and ω' are images of Ω_∞ in the two views. In forming these equations it is necessary to have an affine reconstruction already, since the infinite homography must be known. If $\omega = \omega'$, then (10.3) gives a set of linear equations in the entries of ω. In general this set of linear equations places four constraints on ω, and since ω has 5 degrees of freedom it is not completely determined. However, by combining these linear equations with those above provided by scene orthogonality or known internal parameters, ω may be determined uniquely. Indeed (10.3) may be used to transfer constraints on ω to constraints on ω'. Figure 10.5 shows an example of a metric reconstruction computed by combining constraints in this manner.

Samsung/Google, EX1010, p. 292

10.4.3 Direct metric reconstruction using ω

The previous discussion showed how knowledge of the image of the absolute conic (IAC) may be used to transform an affine to a metric reconstruction. However, knowing ω it is possible to proceed directly to metric reconstruction, given at least two views. This can be accomplished in at least two different ways. The most evident approach is to use the IAC to compute calibration of each of the cameras, and then carry out a calibrated reconstruction.

This method relies on the connection of ω to the calibration matrix $\mathtt{K}$, namely $\omega = (\mathtt{K}\mathtt{K}^\mathsf{T})^{-1}$. Thus one can compute $\mathtt{K}$ from ω by inverting it and then applying Cholesky factorization to obtain $\mathtt{K}$. If the IAC is known in each image, then both cameras may be calibrated in this way. Next with calibrated cameras, a metric reconstruction of the scene may be computed using the essential matrix, as in section 9.6. Note that four possible solutions may result. Two of these are just mirror images, but the other two are different, forming a twisted pair. (Though all solutions but one may be ruled out by consideration of points lying in front of the cameras.)

A more conceptual approach to metric reconstruction is to use knowledge of the IAC to directly determine the plane at infinity and the absolute conic. Knowing the camera matrices $\mathtt{P}$ and $\mathtt{P}'$ in a projective frame, and a conic (specifically the image of the absolute conic) in each image, then Ω_∞ may be explicitly computed in 3-space. This is achieved by back-projecting the conics to cones, which must intersect in the absolute conic. Thus, Ω_∞ and its support plane π_∞ are determined (see exercise (x) on page 342 for an algebraic solution). However, two cones will in general intersect in two different plane conics, each lying in a different support plane. Thus there are two possible solutions for the absolute conic, which one can identify as belonging to the two different reconstructions constituting the twisted pair ambiguity.

10.5 Direct reconstruction – using ground truth

It is possible to jump directly from a projective reconstruction to a metric reconstruction if "ground control points" (that is points with known 3D locations in a Euclidean world frame) are given. Suppose we have a set of n such ground control points $\{\mathbf{X}_{Ei}\}$ which are imaged at $\mathbf{x}_i \leftrightarrow \mathbf{x}'_i$. We wish to use these points to transform the projective reconstruction to metric.

The 3D location $\{\mathbf{X}_i\}$ of the control points in the projective reconstruction may be computed from their image correspondences $\mathbf{x}_i \leftrightarrow \mathbf{x}'_i$. Since the projective reconstruction is related by a homography to the true reconstruction we then have from (10.1) the equations:

$$\mathbf{X}_{Ei} = \mathtt{H}\mathbf{X}_i, \quad i = 1, \ldots, n.$$

Each point correspondence provides 3 linearly independent equations on the elements of $\mathtt{H}$, and since $\mathtt{H}$ has 15 degrees of freedom a linear solution is obtained provided $n \geq 5$ (and no four of the control points are coplanar). This computation, and the proper numerical procedures, are described in chapter 4.

Alternatively, one may bypass the computation of the $\mathbf{X}_i$ and compute $\mathtt{H}$ by relating

Samsung/Google, EX1010, p. 293

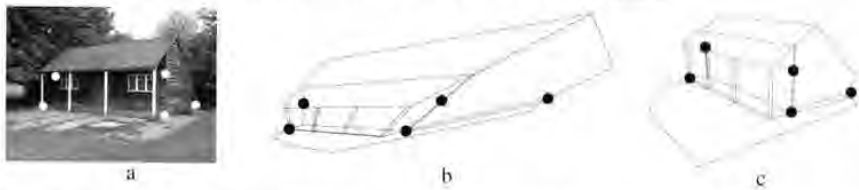


Fig. 10.6. **Direct reconstruction.** *The projective reconstruction of figure 10.3 may be upgraded to metric by specifying the position of five (or more) world points: (a) the five points used; (b) the corresponding points on the projective reconstruction of figure 10.3; (c) the reconstruction after the five points are mapped to their world positions.*

the known ground control points directly to image measurements. Thus as in the DLT algorithm for camera resection (section 7.1(p178)) the equation

$$\mathbf{x}_i = \mathtt{P}\mathtt{H}^{-1}\mathbf{X}_{Ei}$$

provides two linearly independent equations in the entries of the unknown $\mathtt{H}^{-1}$, all other quantities being known. Similarly equations may be derived from the other image if $\mathbf{x}'_i$ is known. It is not necessary for the ground control points to be visible in both images. Note however that if both $\mathbf{x}_i$ and $\mathbf{x}'_i$ *are* visible for a given control point $\mathbf{X}_{Ei}$ then because of the coplanarity constraint on $\mathbf{x}$ and $\mathbf{x}'$, the four equations generated in this way contain only three independent ones.

Once $\mathtt{H}$ has been computed it may be used to transform the cameras $\mathtt{P}, \mathtt{P}'$ of the projective reconstruction to their true Euclidean counterparts. An example of metric structure computed by this direct method is shown in figure 10.6.

10.6 Closure

In this chapter we have overviewed the steps necessary to produce a metric reconstruction from a pair of images. This overview is summarized in algorithm 10.1, and the computational procedures for these steps are described in the following chapters. As usual the general discussion has been restricted mainly to points, but the ideas (triangulation, ambiguity, stratification) apply equally to other image features such as lines, conics etc.

It has been seen that for a metric reconstruction it is necessary to identify two entities in the projective frame; these are the plane at infinity $\boldsymbol{\pi}_\infty$ (for affine), together with the absolute conic Ω_∞ (for metric). Conversely, given $\mathtt{F}$ and a pair of calibrated cameras then $\boldsymbol{\pi}_\infty$ and Ω_∞ may be explicitly computed in 3-space. These entities each have an image-based counterpart: specification of the infinite homography, $\mathtt{H}_\infty$, is equivalent to specifying $\boldsymbol{\pi}_\infty$ in 3-space; and specifying the image of the absolute conic, $\boldsymbol{\omega}$, in each view is equivalent to specifying $\boldsymbol{\pi}_\infty$ and Ω_∞ in 3-space. This equivalence is summarized in table 10.1.

Finally, it is worth noting that if metric precision is not the goal then an acceptable metric reconstruction is generally obtained directly from the projective if approximately correct internal parameters are guessed. Such a "quasi-Euclidean reconstruction" is often suitable for visualization purposes.

Samsung/Google, EX1010, p. 294

Objective

Given two uncalibrated images compute a metric reconstruction $(\mathtt{P}_M, \mathtt{P}'_M, \{\mathbf{X}_{Mi}\})$ of the cameras and scene structure, i.e. a reconstruction that is within a similarity transformation of the true cameras and scene structure.

Algorithm

(i) **Compute a projective reconstruction** $(\mathtt{P}, \mathtt{P}', \{\mathbf{X}_i\})$**:**

(a) **Compute the fundamental matrix** from point correspondences $\mathbf{x}_i \leftrightarrow \mathbf{x}'_i$ between the images.

(b) **Camera retrieval:** compute the camera matrices $\mathtt{P}, \mathtt{P}'$ from the fundamental matrix.

(c) **Triangulation:** for each point correspondence $\mathbf{x}_i \leftrightarrow \mathbf{x}'_i$, compute the point $\mathbf{X}_i$ in space that projects to these two image points.

(ii) **Rectify the projective reconstruction to metric:**

- either **Direct method:** Compute the homography $\mathtt{H}$ such that $\mathbf{X}_{Ei} = \mathtt{H}\mathbf{X}_i$ from five or more ground control points $\mathbf{X}_{Ei}$ with known Euclidean positions. Then the metric reconstruction is

$$\mathtt{P}_M = \mathtt{P}\mathtt{H}^{-1}, \quad \mathtt{P}'_M = \mathtt{P}'\mathtt{H}^{-1}, \quad \mathbf{X}_{Mi} = \mathtt{H}\mathbf{X}_i.$$

- or **Stratified method:**

(a) **Affine reconstruction:** Compute the plane at infinity, π_∞, as described in section 10.4.1, and then upgrade the projective reconstruction to an affine reconstruction with the homography

$$\mathtt{H} = \begin{bmatrix} \mathtt{I} & \mathbf{0} \\ \pi_\infty^\mathsf{T} & \end{bmatrix}.$$

(b) **Metric reconstruction:** Compute the image of the absolute conic, ω, as described in section 10.4.2, and then upgrade the affine reconstruction to a metric reconstruction with the homography

$$\mathtt{H} = \begin{bmatrix} \mathtt{A}^{-1} & \\ & 1 \end{bmatrix}$$

where $\mathtt{A}$ is obtained by Cholesky factorization from the equation $\mathtt{A}\mathtt{A}^\mathsf{T} = (\mathtt{M}^\mathsf{T}\omega\mathtt{M})^{-1}$, and $\mathtt{M}$ is the first 3×3 submatrix of the camera in the affine reconstruction for which ω is computed.

Algorithm 10.1. *Computation of a metric reconstruction from two uncalibrated images.*

10.6.1 The literature

Koenderink and van Doorn [Koenderink-91] give a very elegant discussion of stratification for affine cameras. This was extended to perspective in [Faugeras-95b], with developments given by Luong and Viéville [Luong-94, Luong-96]. The possibility of projective reconstruction given $\mathtt{F}$ appeared in [Faugeras-92b] and Hartley *et al.* [Hartley-92c].

The method of computing affine reconstruction from pure translation first appeared in Moons *et al.* [Moons-94]. Combining scene and internal parameter constraints over multiple views is described in [Faugeras-95c, Liebowitz-99b, Sturm-99c].

Samsung/Google, EX1010, p. 295

Image information provided	View relations and projective objects	3-space objects	Reconstruction ambiguity
Point correspondences	F		Projective
Point correspondences including vanishing points	F, H_∞	π_∞	Affine
Point correspondences and internal camera calibration	F, H_∞ ω, ω'	π_∞ Ω_∞	Metric

Table 10.1. *The two-view relations, image entities, and their 3-space counterparts for various classes of reconstruction ambiguity.*

10.6.2 Notes and exercises

(i) Using only (implicit) image relations (i.e. without an explicit 3D reconstruction) and given the images of a line L and point X (not on L) in two views, together with H_∞ between the views, compute the image of the line in 3-space parallel to L and through X. Other examples of this implicit approach to computation are given in [Zeller-96].

Samsung/Google, EX1010, p. 296

11

Computation of the Fundamental Matrix F

This chapter describes numerical methods for estimating the fundamental matrix given a set of point correspondences between two images. We begin by describing the equations on F generated by point correspondences in two images, and their minimal solution. The following sections then give linear methods for estimating F using algebraic distance, and then various geometric cost functions and solution methods including the MLE ("Gold Standard") algorithm, and Sampson distance.

An algorithm is then described for automatically obtaining point correspondences, so that F may be estimated directly from an image pair. We discuss the estimation of F for special camera motions.

The chapter also covers a method of image rectification based on the computed F.

11.1 Basic equations

The fundamental matrix is defined by the equation

$$\mathbf{x}'^{\mathsf{T}}\mathbf{F}\mathbf{x} = 0 \tag{11.1}$$

for any pair of matching points $\mathbf{x} \leftrightarrow \mathbf{x}'$ in two images. Given sufficiently many point matches $\mathbf{x}_i \leftrightarrow \mathbf{x}'_i$ (at least 7), equation (11.1) can be used to compute the unknown matrix F. In particular, writing $\mathbf{x} = (x, y, 1)^{\mathsf{T}}$ and $\mathbf{x}' = (x', y', 1)^{\mathsf{T}}$ each point match gives rise to one linear equation in the unknown entries of F. The coefficients of this equation are easily written in terms of the known coordinates x and x′. Specifically, the equation corresponding to a pair of points $(x, y, 1)$ and $(x', y', 1)$ is

$$x'xf_{11} + x'yf_{12} + x'f_{13} + y'xf_{21} + y'yf_{22} + y'f_{23} + xf_{31} + yf_{32} + f_{33} = 0. \tag{11.2}$$

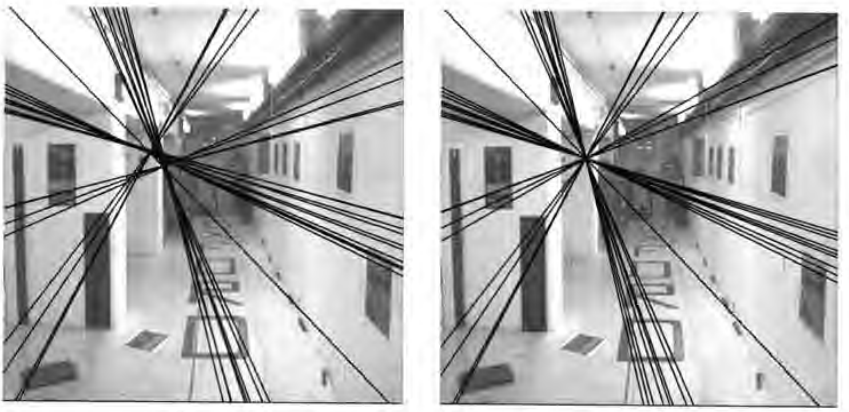
Denote by $\mathbf{f}$ the 9-vector made up of the entries of F in row-major order. Then (11.2) can be expressed as a vector inner product

$$(x'x, x'y, x', y'x, y'y, y', x, y, 1)\,\mathbf{f} = 0.$$

From a set of n point matches, we obtain a set of linear equations of the form

$$\mathbf{A}\mathbf{f} = \begin{bmatrix} x'_1x_1 & x'_1y_1 & x'_1 & y'_1x_1 & y'_1y_1 & y'_1 & x_1 & y_1 & 1 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \\ x'_nx_n & x'_ny_n & x'_n & y'_nx_n & y'_ny_n & y'_n & x_n & y_n & 1 \end{bmatrix} \mathbf{f} = \mathbf{0}. \tag{11.3}$$

Samsung/Google, EX1010, p. 297



a b

Fig. 11.1. **Epipolar lines.** *(a) the effect of a non-singular fundamental matrix. Epipolar lines computed as* $\mathbf{l}' = \mathtt{F}\mathbf{x}$ *for varying* $\mathbf{x}$ *do not meet in a common epipole. (b) the effect of enforcing singularity using the SVD method described here.*

This is a homogeneous set of equations, and $\mathbf{f}$ can only be determined up to scale. For a solution to exist, matrix $\mathtt{A}$ must have rank at most 8, and if the rank is exactly 8, then the solution is unique (up to scale), and can be found by linear methods – the solution is the generator of the right null-space of $\mathtt{A}$.

If the data is not exact, because of noise in the point coordinates, then the rank of $\mathtt{A}$ may be greater than 8 (in fact equal to 9, since $\mathtt{A}$ has 9 columns). In this case, one finds a least-squares solution. Apart from the specific form of the equations (compare (11.3) with (4.3–*p*89)) the problem is essentially the same as the estimation problem considered in section 4.1.1(*p*90). Refer to the algorithm 4.1(*p*91). The least-squares solution for $\mathbf{f}$ is the singular vector corresponding to the smallest singular value of $\mathtt{A}$, that is, the last column of $\mathtt{V}$ in the SVD $\mathtt{A} = \mathtt{U}\mathtt{D}\mathtt{V}^\top$. The solution vector $\mathbf{f}$ found in this way minimizes $\|\mathtt{A}\mathbf{f}\|$ subject to the condition $\|\mathbf{f}\| = 1$. The algorithm just described is the essence of a method called the 8-point algorithm for computation of the fundamental matrix.

11.1.1 The singularity constraint

An important property of the fundamental matrix is that it is singular, in fact of rank 2. Furthermore, the left and right null-spaces of $\mathtt{F}$ are generated by the vectors representing (in homogeneous coordinates) the two epipoles in the two images. Most applications of the fundamental matrix rely on the fact that it has rank 2. For instance, if the fundamental matrix is not singular then computed epipolar lines are not coincident, as is demonstrated by figure 11.1. The matrix $\mathtt{F}$ found by solving the set of linear equations (11.3) will not in general have rank 2, and we should take steps to enforce this constraint. The most convenient way to do this is to correct the matrix $\mathtt{F}$ found by the SVD solution from $\mathtt{A}$. Matrix $\mathtt{F}$ is replaced by the matrix $\mathtt{F}'$ that minimizes the Frobenius norm $\|\mathtt{F} - \mathtt{F}'\|$ subject to the condition $\det \mathtt{F}' = 0$. A convenient method of

Samsung/Google, EX1010, p. 298

doing this is to again use the SVD. In particular, let $\mathtt{F} = \mathtt{U}\mathtt{D}\mathtt{V}^\mathsf{T}$ be the SVD of $\mathtt{F}$, where $\mathtt{D}$ is a diagonal matrix $\mathtt{D} = \mathrm{diag}(r, s, t)$ satisfying $r \geq s \geq t$. Then $\mathtt{F}' = \mathtt{U}\mathrm{diag}(r, s, 0)\mathtt{V}^\mathsf{T}$ minimizes the Frobenius norm of $\mathtt{F} - \mathtt{F}'$.

Thus, the 8-point algorithm for computation of the fundamental matrix may be formulated as consisting of two steps, as follows.

(i) **Linear solution.** A solution $\mathtt{F}$ is obtained from the vector $\mathbf{f}$ corresponding to the smallest singular value of $\mathtt{A}$, where $\mathtt{A}$ is defined in (11.3).

(ii) **Constraint enforcement.** Replace $\mathtt{F}$ by $\mathtt{F}'$, the closest singular matrix to $\mathtt{F}$ under a Frobenius norm. This correction is done using the SVD.

The algorithm thus stated is extremely simple, and readily implemented, assuming that appropriate linear algebra routines are available. As usual normalization is required, and we return to this in section 11.2.

11.1.2 The minimum case – seven point correspondences

The equation $\mathbf{x}_i'^\mathsf{T}\mathtt{F}\mathbf{x}_i = 0$ gives rise to a set of equations of the form $\mathtt{A}\mathbf{f} = \mathbf{0}$. If $\mathtt{A}$ has rank 8, then it is possible to solve for $\mathbf{f}$ up to scale. In the case where the matrix $\mathtt{A}$ has rank seven, it is still possible to solve for the fundamental matrix by making use of the singularity constraint. The most important case is when only 7 point correspondences are known (other cases are discussed in section 11.9). This leads to a 7×9 matrix $\mathtt{A}$, which generally will have rank 7.

The solution to the equations $\mathtt{A}\mathbf{f} = \mathbf{0}$ in this case is a 2-dimensional space of the form $\alpha\mathtt{F}_1 + (1-\alpha)\mathtt{F}_2$, where α is a scalar variable. The matrices $\mathtt{F}_1$ and $\mathtt{F}_2$ are obtained as the matrices corresponding to the generators $\mathbf{f}_1$ and $\mathbf{f}_2$ of the right null-space of $\mathtt{A}$. Now, we use the constraint that $\det \mathtt{F} = 0$. This may be written as $\det(\alpha\mathtt{F}_1 + (1-\alpha)\mathtt{F}_2) = 0$. Since $\mathtt{F}_1$ and $\mathtt{F}_2$ are known, this leads to a cubic polynomial equation in α. This polynomial equation may be solved to find the value of α. There will be either one or three real solutions (the complex solutions are discarded [Hartley-94c]). Substituting back in the equation $\mathtt{F} = \alpha\mathtt{F}_1 + (1-\alpha)\mathtt{F}_2$ gives one or three possible solutions for the fundamental matrix.

This method of computing one or three fundamental matrices for the minimum number of points (seven) is used in the robust algorithm of section 11.6. We return to the issue of the number of solutions in section 11.9.

11.2 The normalized 8-point algorithm

The 8-point algorithm is the simplest method of computing the fundamental matrix, involving no more than the construction and (least-squares) solution of a set of linear equations. If care is taken, then it can perform extremely well. The original algorithm is due to Longuet-Higgins [LonguetHiggins-81]. The key to success with the 8-point algorithm is proper careful normalization of the input data before constructing the equations to solve. The subject of normalization of input data has applications to many of the algorithms of this book, and is treated in general terms in section 4.4-(p104). In the case of the 8-point algorithm, a simple transformation (translation and

Samsung/Google, EX1010, p. 299

Objective

Given $n \geq 8$ image point correspondences $\{\mathbf{x}_i \leftrightarrow \mathbf{x}'_i\}$, determine the fundamental matrix $\mathtt{F}$ such that $\mathbf{x}'^{\top}_i \mathtt{F} \mathbf{x}_i = 0$.

Algorithm

(i) **Normalization:** Transform the image coordinates according to $\hat{\mathbf{x}}_i = \mathtt{T}\mathbf{x}_i$ and $\hat{\mathbf{x}}'_i = \mathtt{T}'\mathbf{x}'_i$, where $\mathtt{T}$ and $\mathtt{T}'$ are normalizing transformations consisting of a translation and scaling.

(ii) Find the fundamental matrix $\hat{\mathtt{F}}'$ corresponding to the matches $\hat{\mathbf{x}}_i \leftrightarrow \hat{\mathbf{x}}'_i$ by

  (a) **Linear solution:** Determine $\hat{\mathtt{F}}$ from the singular vector corresponding to the smallest singular value of $\hat{\mathtt{A}}$, where $\hat{\mathtt{A}}$ is composed from the matches $\hat{\mathbf{x}}_i \leftrightarrow \hat{\mathbf{x}}'_i$ as defined in (11.3).

  (b) **Constraint enforcement:** Replace $\hat{\mathtt{F}}$ by $\hat{\mathtt{F}}'$ such that $\det \hat{\mathtt{F}}' = 0$ using the SVD (see section 11.1.1).

(iii) **Denormalization:** Set $\mathtt{F} = \mathtt{T}'^{\top}\hat{\mathtt{F}}'\mathtt{T}$. Matrix $\mathtt{F}$ is the fundamental matrix corresponding to the original data $\mathbf{x}_i \leftrightarrow \mathbf{x}'_i$.

Algorithm 11.1. *The normalized 8-point algorithm for* F.

scaling) of the points in the image before formulating the linear equations leads to an enormous improvement in the conditioning of the problem and hence in the stability of the result. The added complexity of the algorithm necessary to do this transformation is insignificant.

The suggested normalization is a translation and scaling of each image so that the centroid of the reference points is at the origin of the coordinates and the RMS distance of the points from the origin is equal to $\sqrt{2}$. This is carried out for essentially the same reasons as in chapter 4. The basic method is analogous to algorithm 4.2(p109) and is summarized in algorithm 11.1.

Note that it is recommended that the singularity condition should be enforced before denormalization. For a justification of this, refer to [Hartley-97c].

11.3 The algebraic minimization algorithm

The normalized 8-point algorithm includes a method for enforcing the singularity constraint on the fundamental matrix. The initial estimate $\mathtt{F}$ is replaced by the singular matrix $\mathtt{F}'$ that minimizes the difference $\|\mathtt{F}' - \mathtt{F}\|$. This is done using the SVD, and has the advantage of being simple and rapid.

Numerically, however, this method is not optimal, since all the entries of $\mathtt{F}$ do not have equal importance, and indeed some entries are more tightly constrained by the point-correspondence data than others. A more correct procedure would be to compute a covariance matrix from the entries of $\mathtt{F}$ in terms of the input data, and then to find the singular matrix $\mathtt{F}'$ closest to $\mathtt{F}$ in terms of Mahalanobis distance with respect to this covariance. Unfortunately, minimization of the Mahalanobis distance $\|\mathtt{F} - \mathtt{F}'\|_{\Sigma}$ cannot be done linearly for a general covariance matrix Σ, so this approach is unattractive.

An alternative procedure is to find the desired singular matrix $\mathtt{F}'$ directly. Thus, just as $\mathtt{F}$ is computed by minimizing the norm $\|\mathtt{A}\mathbf{f}\|$ subject to $\|\mathbf{f}\| = 1$, so one should aim

Samsung/Google, EX1010, p. 300

to find the *singular* matrix $\mathtt{F}'$ that minimizes $\|\mathtt{A}\mathbf{f}'\|$ subject to $\|\mathbf{f}'\| = 1$. It turns out not to be possible to do this by linear non-iterative means, chiefly because $\det \mathtt{F}' = 0$ is a cubic, rather than a linear constraint. Nevertheless, it will be seen that a simple iterative method is effective.

An arbitrary singular 3×3 matrix, such as the fundamental matrix $\mathtt{F}$, may be written as a product $\mathtt{F} = \mathtt{M}[\mathbf{e}]_\times$ where $\mathtt{M}$ is a non-singular matrix and $[\mathbf{e}]_\times$ is any skew-symmetric matrix, with $\mathbf{e}$ corresponding to the epipole in the first image.

Suppose we wish to compute the fundamental matrix $\mathtt{F}$ of the form $\mathtt{F} = \mathtt{M}[\mathbf{e}]_\times$ that minimizes the algebraic error $\|\mathtt{A}\mathbf{f}\|$ subject to the condition $\|\mathbf{f}\| = 1$. Let us assume for now that the epipole $\mathbf{e}$ is known. Later we will let $\mathbf{e}$ vary, but for now it is fixed. The equation $\mathtt{F} = \mathtt{M}[\mathbf{e}]_\times$ can be written in terms of the vectors $\mathbf{f}$ and $\mathbf{m}$ comprising the entries of $\mathtt{F}$ and $\mathtt{M}$ as an equation $\mathbf{f} = \mathtt{E}\mathbf{m}$ where $\mathtt{E}$ is a 9×9 matrix. Supposing that $\mathbf{f}$ and $\mathbf{m}$ contain the entries of the corresponding matrices in row-major order, then it can be verified that $\mathtt{E}$ has the form

$$\mathtt{E} = \begin{bmatrix} [\mathbf{e}]_\times & & \\ & [\mathbf{e}]_\times & \\ & & [\mathbf{e}]_\times \end{bmatrix}. \tag{11.4}$$

Since $\mathbf{f} = \mathtt{E}\mathbf{m}$, the minimization problem becomes: [1]

$$\text{Minimize } \|\mathtt{A}\mathtt{E}\mathbf{m}\| \text{ subject to the condition } \|\mathtt{E}\mathbf{m}\| = 1. \tag{11.5}$$

This minimization problem is solved using algorithm A5.6(*p*595). For the purposes of this algorithm one observes that $\text{rank}(\mathtt{E}) = 6$, since each of its diagonal blocks has rank 2.

11.3.1 Iterative estimation

The minimization (11.5) gives a way of computing an algebraic error vector $\mathtt{A}\mathbf{f}$ given a value for the epipole $\mathbf{e}$. This mapping $\mathbf{e} \mapsto \mathtt{A}\mathbf{f}$ is a map from $\mathbb{R}^3$ to $\mathbb{R}^9$. Note that the value of $\mathtt{A}\mathbf{f}$ is unaffected by scaling $\mathbf{e}$. Starting from an estimated value of $\mathbf{e}$ derived as the generator of the right null-space of an initial estimate of $\mathtt{F}$, one may iterate to find the final $\mathtt{F}$ that minimizes algebraic error. The initial estimate of $\mathtt{F}$ may be obtained from the 8-point algorithm, or any other simple algorithm. The complete algorithm for computation of $\mathtt{F}$ is given in algorithm 11.2.

Note the advantage of this method of computing $\mathtt{F}$ is that the iterative part of the algorithm consists of a very small parameter minimization problem, involving the estimation of only three parameters (the homogeneous coordinates of $\mathbf{e}$). Despite this, the algorithm finds the fundamental matrix that minimizes the algebraic error for all matched points. The matched points themselves do not come into the final iterative estimation.

[1] It does not do to minimize $\|\mathtt{A}\mathtt{E}\mathbf{m}\|$ subject to the condition $\|\mathbf{m}\| = 1$, since a solution to this occurs when $\mathbf{m}$ is a unit vector in the right null-space of $\mathtt{E}$. In this case, $\mathtt{E}\mathbf{m} = \mathbf{0}$, and hence $\|\mathtt{A}\mathtt{E}\mathbf{m}\| = 0$.

Samsung/Google, EX1010, p. 301

Objective

Find the fundamental matrix $\mathtt{F}$ that minimizes the algebraic error $\|\mathtt{A}\mathbf{f}\|$ subject to $\|\mathbf{f}\| = 1$ and $\det \mathtt{F} = 0$.

Algorithm

(i) Find a first approximation $\mathtt{F}_0$ for the fundamental matrix using the normalized 8-point algorithm 11.1. Then find the right null-vector $\mathbf{e}_0$ of $\mathtt{F}_0$.

(ii) Starting with the estimate $\mathbf{e}_i = \mathbf{e}_0$ for the epipole, compute the matrix $\mathtt{E}_i$ according to (11.4), then find the vector $\mathbf{f}_i = \mathtt{E}_i\mathbf{m}_i$ that minimizes $\|\mathtt{A}\mathbf{f}_i\|$ subject to $\|\mathbf{f}_i\| = 1$. This is done using algorithm A5.6($p595$).

(iii) Compute the algebraic error $\boldsymbol{\epsilon}_i = \mathtt{A}\mathbf{f}_i$. Since $\mathbf{f}_i$ and hence $\boldsymbol{\epsilon}_i$ is defined only up to sign, correct the sign of $\boldsymbol{\epsilon}_i$ (multiplying by minus 1 if necessary) so that $\mathbf{e}_i^\mathsf{T}\mathbf{e}_{i-1} > 0$ for $i > 0$. This is done to ensure that $\boldsymbol{\epsilon}_i$ varies smoothly as a function of $\mathbf{e}_i$.

(iv) The previous two steps define a mapping $\mathbb{R}^3 \to \mathbb{R}^9$ mapping $\mathbf{e}_i \mapsto \boldsymbol{\epsilon}_i$. Now use the Levenberg–Marquardt algorithm (section A6.2($p600$)) to vary $\mathbf{e}_i$ iteratively so as to minimize $\|\boldsymbol{\epsilon}_i\|$.

(v) Upon convergence, $\mathbf{f}_i$ represents the desired fundamental matrix.

Algorithm 11.2. *Computation of* $\mathtt{F}$ *with* $\det \mathtt{F} = 0$ *by iteratively minimizing algebraic error.*

11.4 Geometric distance

This section describes three algorithms which minimize a geometric image distance. The one we recommend, which is the Gold Standard method, unfortunately requires the most effort in implementation. The other algorithms produce extremely good results and are easier to implement, but are not optimal under the assumption that the image errors are Gaussian. Two important issues for each of the algorithms are the intitialization for the non-linear minimization, and the parametrization of the cost function. The algorithms are generally initialized by one of the linear algorithms of the previous section. An alternative, which is used in the automatic algorithm, is to select 7 correspondences and thus generate one or three solutions for $\mathtt{F}$. Various parametrizations are discussed in section 11.4.2. In all cases we recommend that the image points be normalized by a translation and scaling. This normalization does not skew the noise characteristics, so does not interfere with the optimality of the Gold Standard algorithm, which is described next.

11.4.1 The Gold Standard method

The Maximum Likelihood estimate of the fundamental matrix depends on the assumption of an error model. We make the assumption that noise in image point measurements obeys a Gaussian distribution. In that case the ML estimate is the one that minimizes the geometric distance (which is reprojection error)

$$\sum_i d(\mathbf{x}_i, \hat{\mathbf{x}}_i)^2 + d(\mathbf{x}'_i, \hat{\mathbf{x}}'_i)^2 \tag{11.6}$$

where $\mathbf{x}_i \leftrightarrow \mathbf{x}'_i$ are the measured correspondences, and $\hat{\mathbf{x}}_i$ and $\hat{\mathbf{x}}'_i$ are estimated "true" correspondences that satisfy $\hat{\mathbf{x}}_i'^\mathsf{T}\mathtt{F}\hat{\mathbf{x}}_i = 0$ exactly for some rank-2 matrix $\mathtt{F}$, the estimated fundamental matrix.

Samsung/Google, EX1010, p. 302

Objective

Given $n \geq 8$ image point correspondences $\{\mathbf{x}_i \leftrightarrow \mathbf{x}'_i\}$, determine the Maximum Likelihood estimate $\hat{\mathtt{F}}$ of the fundamental matrix.

The MLE involves also solving for a set of subsidiary point correspondences $\{\hat{\mathbf{x}}_i \leftrightarrow \hat{\mathbf{x}}'_i\}$, such that $\hat{\mathbf{x}}_i'^{\mathsf{T}}\hat{\mathtt{F}}\hat{\mathbf{x}}_i = 0$, and which minimizes

$$\sum_i d(\mathbf{x}_i, \hat{\mathbf{x}}_i)^2 + d(\mathbf{x}'_i, \hat{\mathbf{x}}'_i)^2.$$

Algorithm

(i) Compute an initial rank 2 estimate of $\hat{\mathtt{F}}$ using a linear algorithm such as algorithm 11.1.
(ii) Compute an initial estimate of the subsidiary variables $\{\hat{\mathbf{x}}_i, \hat{\mathbf{x}}'_i\}$ as follows:
  (a) Choose camera matrices $\mathtt{P} = [\mathtt{I} \mid \mathbf{0}]$ and $\mathtt{P}' = [[\mathbf{e}']_\times \hat{\mathtt{F}} \mid \mathbf{e}']$, where $\mathbf{e}'$ is obtained from $\hat{\mathtt{F}}$.
  (b) From the correspondence $\mathbf{x}_i \leftrightarrow \mathbf{x}'_i$ and $\hat{\mathtt{F}}$ determine an estimate of $\widehat{\mathbf{X}}_i$ using the triangulation method of chapter 12.
  (c) The correspondence consistent with $\hat{\mathtt{F}}$ is obtained as $\hat{\mathbf{x}}_i = \mathtt{P}\widehat{\mathbf{X}}_i$, $\hat{\mathbf{x}}'_i = \mathtt{P}'\widehat{\mathbf{X}}_i$.
(iii) Minimize the cost

$$\sum_i d(\mathbf{x}_i, \hat{\mathbf{x}}_i)^2 + d(\mathbf{x}'_i, \hat{\mathbf{x}}'_i)^2$$

over $\hat{\mathtt{F}}$ and $\widehat{\mathbf{X}}_i$, $i = 1, \ldots, n$. The cost is minimized using the Levenberg–Marquardt algorithm over $3n + 12$ variables: $3n$ for the n 3D points $\widehat{\mathbf{X}}_i$, and 12 for the camera matrix $\mathtt{P}' = [\mathtt{M} \mid \mathbf{t}]$, with $\hat{\mathtt{F}} = [\mathbf{t}]_\times \mathtt{M}$, and $\hat{\mathbf{x}}_i = \mathtt{P}\widehat{\mathbf{X}}_i$, $\hat{\mathbf{x}}'_i = \mathtt{P}'\widehat{\mathbf{X}}_i$.

Algorithm 11.3. *The Gold Standard algorithm for estimating* $\mathtt{F}$ *from image correspondences.*

This error function may be minimized in the following manner. A pair of camera matrices $\mathtt{P} = [\mathtt{I} \mid \mathbf{0}]$ and $\mathtt{P}' = [\mathtt{M} \mid \mathbf{t}]$ are defined. In addition one defines 3D points $\mathbf{X}_i$. Now letting $\hat{\mathbf{x}}_i = \mathtt{P}\mathbf{X}_i$ and $\hat{\mathbf{x}}'_i = \mathtt{P}'\mathbf{X}_i$, one varies $\mathtt{P}'$ and the points $\mathbf{X}_i$ so as to minimize the error expression. Subsequently $\mathtt{F}$ is computed as $\mathtt{F} = [\mathbf{t}]_\times \mathtt{M}$. The vectors $\hat{\mathbf{x}}_i$ and $\hat{\mathbf{x}}'_i$ will satisfy $\hat{\mathbf{x}}_i'^{\mathsf{T}}\mathtt{F}\hat{\mathbf{x}}_i = 0$. Minimization of the error is carried out using the Levenberg–Marquardt algorithm described in section A6.2(p600). An initial estimate of the parameters is computed using the normalized 8-point algorithm, followed by projective reconstruction, as described in chapter 12. Thus, estimation of the fundamental matrix using this method is effectively equivalent to projective reconstruction. The steps of the algorithm are summarized in algorithm 11.3.

It may seem that this method for computing $\mathtt{F}$ will be expensive in computing cost. However, the use of the sparse LM techniques means that it is not much more expensive than other iterative techniques, and details of this are given in section A6.5(p609).

11.4.2 Parametrization of rank-2 matrices

The non-linear minimization of the geometric distance cost functions requires a parametrization of the fundamental matrix which enforces the rank 2 property of the matrix. We describe three such parametrizations.

Samsung/Google, EX1010, p. 303

Over-parametrization. One way that we have already seen for parametrizing F is to write $\mathtt{F} = [\mathbf{t}]_\times \mathtt{M}$, where M is an arbitrary 3×3 matrix. This ensures that F is singular, since $[\mathbf{t}]_\times$ is. This way, F is parametrized by the nine entries of M and the three entries of $\mathbf{t}$ – a total of 12 parameters, more than the minimum number of parameters, which is 7. In general this should not cause a significant problem.

Epipolar parametrization. An alternative way of parametrizing F is by specifying the first two columns of F, along with two multipliers α and β such that the third column may be written as a linear combination $\mathbf{f}_3 = \alpha \mathbf{f}_1 + \beta \mathbf{f}_2$. Thus, the fundamental matrix is parametrized as

$$\mathtt{F} = \begin{bmatrix} a & b & \alpha a + \beta b \\ c & d & \alpha c + \beta d \\ e & f & \alpha e + \beta f \end{bmatrix}. \tag{11.7}$$

This has a total of 8 parameters. To achieve a minimum set of parameters, one of the elements, for instance f, may be set to 1. In practice whichever of $a, \ldots, f$ has greatest absolute value is set to 1. This method ensures a singular matrix F, while using the minimum number of parameters. The main disadvantage is that it has a singularity – it does not work when the first two columns of F are linearly dependent, for then it is not possible to write column 3 in terms of the first two columns. This problem can be significant, since it will occur in the case where the right epipole lies at infinity. For then $\mathtt{F}\mathbf{e} = \mathtt{F}(e_1, e_2, 0)^\mathsf{T} = \mathbf{0}$ and the first two columns are linearly dependent. Nevertheless, this parametrization is widely used and works well if steps are taken to avoid this singularity. Instead of using the first two columns as a basis, another pair of columns can be used, in which case the singularity occurs when the epipole is on one of the coordinate axes. In practice such singularities can be detected during the minimization and the parametrization switched to one of the alternative parametrizations.

Note that $(\alpha, \beta, -1)^\mathsf{T}$ is the right epipole for this fundamental matrix – the coordinates of the epipole occur explicitly in the parametrization. For best results, the parametrization should be chosen so that the largest entry (in absolute value) of the epipole is the one set to 1.

Note how the complete manifold of possible fundamental matrices is not covered by a single parametrization, but rather by a set of minimally parametrized patches. As a path is traced out through the manifold during a parameter minimization procedure, it is necessary to switch from one patch to another as the boundary between patches is crossed. In this case there are actually 18 different parameter patches, depending on which of $a, \ldots, f$ is greatest, and which pair of columns are taken as the basis.

Both epipoles as parameters. The previous parametrization uses one of the epipoles as part of the parametrization. For symmetry one may use both the epipoles as parameters. The resulting form of F is

$$\mathtt{F} = \begin{bmatrix} a & b & \alpha a + \beta b \\ c & d & \alpha c + \beta d \\ \alpha' a + \beta' c & \alpha' b + \beta' d & \alpha' \alpha a + \alpha' \beta b + \beta' \alpha c + \beta' \beta d \end{bmatrix}. \tag{11.8}$$

Samsung/Google, EX1010, p. 304

The two epipoles are $(\alpha, \beta, -1)^\mathsf{T}$ and $(\alpha', \beta', -1)^\mathsf{T}$. As above, one can set one of a, b, c, d to 1. To avoid singularities, one must switch between different choices of the two rows and two columns to use as the basis. Along with four choices of which of a, b, c, d to set to 1, there are a total of 36 parameter patches used to cover the complete manifold of fundamental matrices.

11.4.3 First-order geometric error (Sampson distance)

The concept of Sampson distance was discussed at length in section 4.2.6(p98). Here the Sampson approximation is used in the case of the variety defined by $\mathbf{x}'^\mathsf{T}\mathbf{F}\mathbf{x} = 0$ to provide a first-order approximation to the geometric error.

The general formula for the Sampson cost function is given in (4.13–p100). In the case of fundamental matrix estimation, the formula is even simpler, since there is only one equation per point correspondence (see also example 4.2(p100)). The partial-derivative matrix J has only one row, and hence $\mathtt{J}\mathtt{J}^\mathsf{T}$ is a scalar and (4.12–p99) becomes

$$\frac{\boldsymbol{\epsilon}^\mathsf{T}\boldsymbol{\epsilon}}{\mathtt{J}\mathtt{J}^\mathsf{T}} = \frac{(\mathbf{x}_i'^\mathsf{T}\mathbf{F}\mathbf{x}_i)^2}{\mathtt{J}\mathtt{J}^\mathsf{T}}.$$

From the definition of J and the explicit form of $\mathtt{A}_i = \mathbf{x}_i'^\mathsf{T}\mathbf{F}\mathbf{x}_i$ given in the left hand side of (11.2), we obtain

$$\mathtt{J}\mathtt{J}^\mathsf{T} = (\mathbf{F}\mathbf{x}_i)_1^2 + (\mathbf{F}\mathbf{x}_i)_2^2 + (\mathbf{F}^\mathsf{T}\mathbf{x}_i')_1^2 + (\mathbf{F}^\mathsf{T}\mathbf{x}_i')_2^2$$

where for instance $(\mathbf{F}\mathbf{x}_i)_j^2$ represents the square of the j-th entry of the vector $\mathbf{F}\mathbf{x}_i$. Thus, the cost function is

$$\sum_i \frac{(\mathbf{x}_i'^\mathsf{T}\mathbf{F}\mathbf{x}_i)^2}{(\mathbf{F}\mathbf{x}_i)_1^2 + (\mathbf{F}\mathbf{x}_i)_2^2 + (\mathbf{F}^\mathsf{T}\mathbf{x}_i')_1^2 + (\mathbf{F}^\mathsf{T}\mathbf{x}_i')_2^2}. \tag{11.9}$$

This gives a first-order approximation to geometric error, which may be expected to give good results if higher order terms are small in comparison to the first. The approximation has been used successfully in estimation algorithms by [Torr-97, Torr-98, Zhang-98]. Note that this approximation is undefined at the point in $\mathbb{R}^4$ determined by the two epipoles, as here $\mathtt{J}\mathtt{J}^\mathsf{T}$ is zero. This point should be avoided in any numerical implementation.

The key advantage of approximating the geometric error in this way is that the resulting cost function only involves the parameters of F. This means that to first-order the Gold Standard cost function (11.6) is minimized *without* introducing a set of subsidiary variables, namely the coordinates of the n space points $\mathbf{X}_i$. Consequently a minimization problem with $7 + 3n$ degrees of freedom is reduced to one with only 7 degrees of freedom.

Symmetric epipolar distance. Equation (11.9) is similar in form to another cost function

$$\sum_i \quad d(\mathbf{x}_i', \mathbf{F}\mathbf{x}_i)^2 + d(\mathbf{x}_i, \mathbf{F}^\mathsf{T}\mathbf{x}_i')^2$$

Samsung/Google, EX1010, p. 305

$$= \sum_i (\mathbf{x}_i'^{\mathsf{T}} \mathbf{F} \mathbf{x}_i)^2 \left(\frac{1}{(\mathbf{F}\mathbf{x}_i)_1^2 + (\mathbf{F}\mathbf{x}_i)_2^2} + \frac{1}{(\mathbf{F}^{\mathsf{T}}\mathbf{x}_i')_1^2 + (\mathbf{F}^{\mathsf{T}}\mathbf{x}_i')_2^2} \right) \quad (11.10)$$

which minimizes the distance of a point from its projected epipolar line, computed in each of the images. However, this cost function seems to give slightly inferior results to (11.9) (see [Zhang-98]), and hence is not discussed further.

11.5 Experimental evaluation of the algorithms

Three of the algorithms of the previous sections are now compared by estimating F from point correspondences for a number of image pairs. The algorithms are:

(i) The normalized 8-point algorithm (algorithm 11.1).
(ii) Minimization of algebraic error whilst imposing the singularity constraint (algorithm 11.2).
(iii) The Gold Standard geometric algorithm (algorithm 11.3).

The experimental procedure was as follows. For each pair of images, a number n of matched points were chosen randomly from the matches and the fundamental matrix estimated and residual error (see below) computed. This experiment was repeated 100 times for each value of n and each pair of images, and the average residual error plotted against n. This gives an idea of how the different algorithms behave as the number of points is increased. The number of points used, n, ranged from 8 up to three-quarters of the total number of matched points.

Residual error

The error is defined as

$$\frac{1}{N} \sum_i^N d(\mathbf{x}_i', \mathbf{F}\mathbf{x}_i)^2 + d(\mathbf{x}_i, \mathbf{F}^{\mathsf{T}}\mathbf{x}_i')^2$$

where $d(\mathbf{x}, \mathbf{l})$ here is the distance (in pixels) between a point $\mathbf{x}$ and a line $\mathbf{l}$. The error is the squared distance between a point's epipolar line and the matching point in the other image (computed for both points of the match), averaged over all N matches. Note the error is evaluated over *all* N matched points, and not just the n matches used to compute F. The residual error corresponds to the epipolar distance defined in (11.10). Note that this particular error is *not* minimized directly by any of the algorithms evaluated here.

The various algorithms were tried with 5 different pairs of images. The images are presented in figure 11.2 and show the diversity of image types, and placement of the epipoles. A few of the epipolar lines are shown in the images. The intersection of the pencil of lines is the epipole. There was a wide variation in the accuracy of the matched points for the different images, though mismatches were removed in a pre-processing step.

Results. The results of these experiments are shown and explained in figure 11.3. They show that minimizing algebraic error gives essentially indistinguishable results from minimizing geometric error.

Samsung/Google, EX1010, p. 306

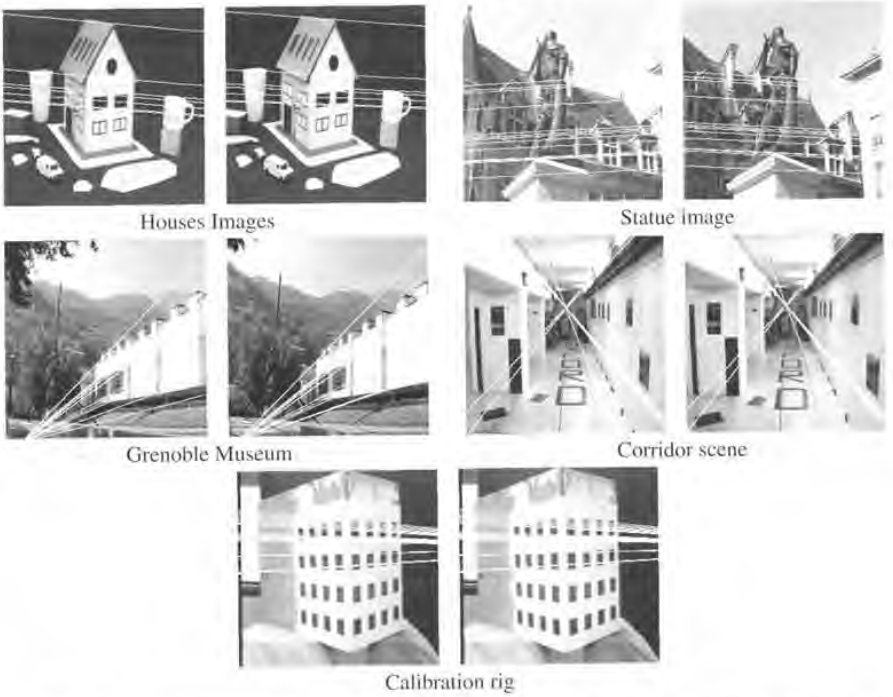


Fig. 11.2. *Image pairs used for the algorithm comparison. In the top two the epipoles are far from the image centres. In the middle two the epipoles are close (Grenoble) and in the image (Corridor). For the calibration images the matched points are known extremely accurately.*

11.5.1 Recommendations

Several methods of computing the fundamental matrix have been discussed in this chapter, and some pointers on which method to use are perhaps desirable. Briefly, these are our recommendations:

- Do not use the unnormalized 8-point algorithm.
- For a quick method, easy to implement, use the normalized 8-point algorithm 11.1. This often gives adequate results, and is ideal as a first step in other algorithms.
- If more accuracy is desired, use the algebraic minimization method, either with or without iteration on the position of the epipole.
- As an alternative that gives excellent results, use an iterative-minimization method that minimizes the Sampson cost function (11.9). This and the iterative algebraic method give similar results.
- To be certain of getting the best results, if Gaussian noise is a viable assumption, implement the Gold Standard algorithm.

Samsung/Google, EX1010, p. 307

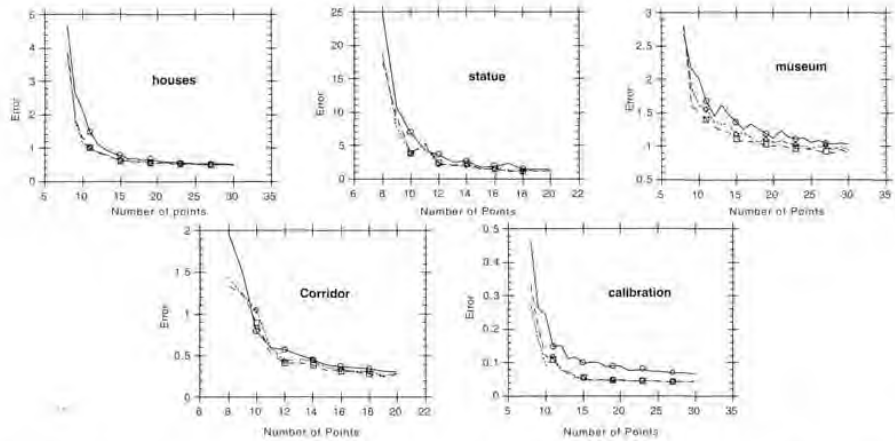


Fig. 11.3. **Results of the experimental evaluation of the algorithms.** *In each case, three methods of computing* F *are compared. Residual error is plotted against the number of points used to compute* F. *In each graph, the top (solid line) shows the results of the normalized 8-point algorithm. Also shown are the results of minimizing geometric error (long dashed line) and iteratively minimizing algebraic error subject to the determinant constraint (short dashed line). In most cases, the result of iteratively minimizing algebraic error is almost indistinguishable from minimizing geometric error. Both are noticeably better than the non-iterative normalized 8-point algorithm, though that algorithm also gives good results.*

11.6 Automatic computation of F

This section describes an algorithm to compute the epipolar geometry between two images automatically. The input to the algorithm is simply the pair of images, with no other *a priori* information required; and the output is the estimated fundamental matrix together with a set of interest points in correspondence.

The algorithm uses RANSAC as a search engine in a similar manner to its use in the automatic computation of a homography described in section 4.8(p123). The ideas and details of the algorithm are given there, and are not repeated here. The method is summarized in algorithm 11.4, with an example of its use shown in figure 11.4.

A few remarks on the method:

(i) **The RANSAC sample.** Only 7 point correspondences are used to estimate F. This has the advantage that a rank 2 matrix is produced, and it is not necessary to coerce the matrix to rank 2 as in the linear algorithms. A second reason for using 7 correspondences, rather than 8 say with a linear algorithm, is that the number of samples that must be tried in order to ensure a high probability of no outliers is exponential in the size of the sample set. For example, from table 4.3-(p119) for a 99% confidence of no outliers (when drawing from a set containing 50% outliers) twice as many samples are required for 8 correspondences as for 7. The slight disadvantage in using 7 correspondences is that it may result in 3 real solutions for F, and all 3 must be tested for support.

Samsung/Google, EX1010, p. 308

Objective Compute the fundamental matrix between two images.

Algorithm

(i) **Interest points:** Compute interest points in each image.
(ii) **Putative correspondences:** Compute a set of interest point matches based on proximity and similarity of their intensity neighbourhood.
(iii) **RANSAC robust estimation:** Repeat for N samples, where N is determined adaptively as in algorithm 4.5(p121):
  (a) Select a random sample of 7 correspondences and compute the fundamental matrix F as described in section 11.1.2. There will be one or three real solutions.
  (b) Calculate the distance $d_\perp$ for each putative correspondence.
  (c) Compute the number of inliers consistent with F by the number of correspondences for which $d_\perp < t$ pixels.
  (d) If there are three real solutions for F the number of inliers is computed for each solution, and the solution with most inliers retained.

  Choose the F with the largest number of inliers. In the case of ties choose the solution that has the lowest standard deviation of inliers.
(iv) **Non-linear estimation:** re-estimate F from all correspondences classified as inliers by minimizing a cost function, e.g. (11.6), using the Levenberg–Marquardt algorithm of section A6.2(p600).
(v) **Guided matching:** Further interest point correspondences are now determined using the estimated F to define a search strip about the epipolar line.

The last two steps can be iterated until the number of correspondences is stable.

Algorithm 11.4. *Algorithm to automatically estimate the fundamental matrix between two images using RANSAC.*

(ii) **The distance measure.** Given a current estimate of F (from the RANSAC sample) the distance $d_\perp$ measures how closely a matched pair of points satisfies the epipolar geometry. There are two clear choices for $d_\perp$: reprojection error, i.e. the distance minimized in the cost function (11.6) (the value may be obtained using the triangulation algorithm of section 12.5); or the Sampson approximation to reprojection error ($d_\perp^2$ is given by (11.9)). If the Sampson approximation is used, then the Sampson cost function should be used to iteratively estimate F. Otherwise distances used in RANSAC and elsewhere in the algorithm will be inconsistent.

(iii) **Guided matching.** The current estimate of F defines a search band in the second image around the epipolar line $\mathbf{Fx}$ of $\mathbf{x}$. For each corner $\mathbf{x}$ a match is sought within this band. Since the search area is restricted a weaker similarity threshold can be employed, and it is not necessary to enforce a "winner takes all" scheme.

(iv) **Implementation and run details.** For the example of figure 11.4, the search window was ± 300 pixels. The inlier threshold was $t = 1.25$ pixels. A total of 407 samples were required. The RMS pixel error after RANSAC was 0.34 (for 99 correspondences), and after MLE and guided matching it was 0.33 (for 157 correspondences). The guided matching MLE required 10 iterations of the Levenberg–Marquardt algorithm.

Samsung/Google, EX1010, p. 309



Fig. 11.4. **Automatic computation of the fundamental matrix between two images using RANSAC.** *(a) (b) left and right images of Keble College, Oxford. The motion between views is a translation and rotation. The images are* 640×480 *pixels. (c) (d) detected corners superimposed on the images. There are approximately 500 corners on each image. The following results are superimposed on the left image: (e) 188 putative matches shown by the line linking corners, note the clear mismatches; (f) outliers – 89 of the putative matches. (g) inliers – 99 correspondences consistent with the estimated* F*; (h) final set of 157 correspondences after guided matching and MLE. There are still a few mismatches evident, e.g. the long line on the left.*

Samsung/Google, EX1010, p. 310

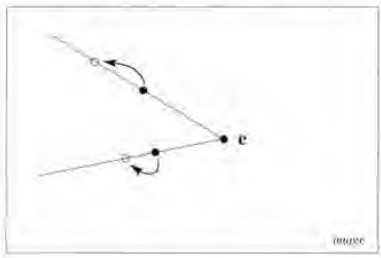


Fig. 11.5. *For a pure translation the epipole can be estimated from the image motion of two points.*

11.7 Special cases of F-computation

Certain special cases of motion, or partially known camera calibration, allow computation of the fundamental matrix to be simplified. In each case the number of degrees of freedom of the fundamental matrix is less than the 7 of general motion. We give three examples.

11.7.1 Pure translational motion

This is the simplest possible case. The matrix can be estimated linearly whilst simultaneously imposing the constraints that the matrix must satisfy, namely that it is skew-symmetric (see section 9.3.1($p247$)), and thus has the required rank of 2. In this case $\mathtt{F} = [\mathbf{e}']_\times$, and has two degrees of freedom. It may be parametrized by the three entries of $\mathbf{e}'$.

Each point correspondence provides one linear constraint on the homogeneous parameters, as is clear from figure 11.5. The matrix can be computed uniquely from two point correspondences.

Note, in the general motion case if all 3D points are coplanar, which is a structure degeneracy (see section 11.9), the fundamental matrix cannot be determined uniquely from image correspondences. However, for pure translational motion this is not a problem (two 3D points are always coplanar). The only degeneracy is if the two 3D points are coplanar with both camera centres.

This special form also simplifies the Gold Standard estimation, and correspondingly triangulation for structure recovery. The Gold Standard estimation of the epipole from point correspondences under pure translation is identical to the estimation of a vanishing point given the end points of a set of imaged parallel lines, see section 8.6.1($p213$).

11.7.2 Planar motion

In the case of planar motion, described in section 9.3.2($p250$), we require that the symmetric part of F has rank 2, in addition to the standard rank 2 condition for the full matrix. It can be verified that the parametrization of (9.8–$p252$), namely $\mathtt{F} = [\mathbf{e}']_\times [\mathbf{l}_s]_\times [\mathbf{e}]_\times$, satisfies both these conditions. If unconstrained 3-vectors are used to represent $\mathbf{e}', \mathbf{l}_s$ and $\mathbf{e}$ then 9 parameters are used, whereas the fundamental matrix for planar motion has only 6 degrees of freedom. As usual this over-parametrization is not a problem.

Samsung/Google, EX1010, p. 311

An alternative parametrization with similar properties is

$$\mathtt{F} = \alpha[\mathbf{x_a}]_\times + \beta\left(\mathbf{l}_s\mathbf{l}_h^\top + \mathbf{l}_h\mathbf{l}_s^\top\right) \quad \text{with} \quad \mathbf{x_a}^\top\mathbf{l}_h = 0$$

where α and β are scalars, and the meaning of the 3-vectors $\mathbf{x_a}, \mathbf{l}_s$ and $\mathbf{l}_h$ is evident from figure 9.11(*p*253)(a).

11.7.3 The calibrated case

In the case of calibrated cameras normalized image coordinates may be used, and the essential matrix E computed instead of the fundamental matrix. As with the fundamental matrix, the essential matrix may be computed using linear techniques from 8 points or more, since corresponding points satisfy the defining equation $\mathbf{x}_i'^\top \mathtt{E}\mathbf{x}_i = 0$.

Where the method differs from the computation of the fundamental matrix is in the enforcement of the constraints. For, whereas the fundamental matrix satisfies $\det \mathtt{F} = 0$, the essential matrix satisfies the additional condition that its two singular values are equal. This constraint may be handled by the following result, which is offered here without proof.

Result 11.1. *Let* $\mathtt{E}$ *be a* 3×3 *matrix with SVD given by* $\mathtt{E} = \mathtt{U}\mathtt{D}\mathtt{V}^\top$, *where* $\mathtt{D} = \mathrm{diag}(a, b, c)$ *with* $a \geq b \geq c$. *Then the closest essential matrix to* $\mathtt{E}$ *in Frobenius norm is given by* $\hat{\mathtt{E}} = \mathtt{U}\hat{\mathtt{D}}\mathtt{V}^\top$, *where* $\hat{\mathtt{D}} = \mathrm{diag}((a+b)/2, (a+b)/2, 0)$.

If the goal is to compute the two normalized camera matrices P and P′ as part of a reconstruction process, then it is not actually necessary to compute $\hat{\mathtt{E}}$ by multiplying out $\hat{\mathtt{E}} = \mathtt{U}\hat{\mathtt{D}}\mathtt{V}^\top$. Matrix P′ can be computed directly from the SVD according to result 9.19-(*p*259). The choice between the four solutions for P′ is determined by the consideration that the visible points must lie in front of the two cameras, as explained in section 9.6.3-(*p*259).

11.8 Correspondence of other entities

So far in this chapter only point correspondences have been employed, and the question naturally arises: can F be computed from the correspondence of image entities other than points? The answer is yes, but not from all types of entities. We will now discuss some common examples.

Lines. The correspondence of image lines between views places *no* constraint at all on F. Here a line is an infinite line, not a line segment. Consider the case of corresponding image points: the points in each image back-project to rays, one through each camera centre, and these rays intersect at the 3-space point. Now in general two lines in 3-space are skew (i.e. they do not intersect); so the condition that the rays intersect places a constraint on the epipolar geometry. In contrast in the case of corresponding image lines, the back-projection is a plane from each view. However, two planes in 3-space always intersect so there is no constraint on the epipolar geometry (there is a constraint in the case of 3-views).

In the case of parallel lines, the correspondence of vanishing points does provide a

Samsung/Google, EX1010, p. 312

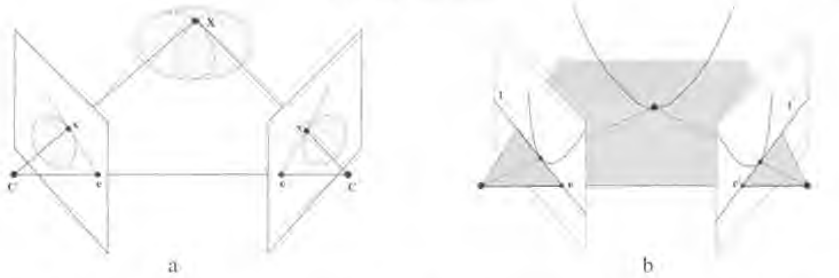


Fig. 11.6. **Epipolar tangency.** *(a) for a surface; (b) for a space curve – figure after Porrill and Pollard [Porrill-91]. In (a) the epipolar plane* $\mathbf{CC}'\mathbf{X}$ *is tangent to the surface at* $\mathbf{X}$*. The imaged outline is tangent to the epipolar lines at* $\mathbf{x}$ *and* $\mathbf{x}'$ *in the two views. The dashed curves on the surface are the contour generators. In (b) the epipolar plane is tangent to the space curve. The corresponding epipolar lines* $\mathrm{l} \leftrightarrow \mathrm{l}'$ *are tangent to the imaged curve.*

constraint on F. However, a vanishing point has the same status as any finite point, i.e. it provides one constraint.

Space curves and surfaces. As illustrated in figure 11.6, at points at which the epipolar plane is tangent to a space curve the imaged curve is tangent to the corresponding epipolar lines. This provides a constraint on the 2 view geometry, i.e. if an epipolar line is tangent to an imaged curve in one view, then the corresponding epipolar line must be tangent to the imaged curve in the other view. Similarly, in the case of surfaces, at points at which the epipolar plane is tangent to the surface the imaged outline is tangent to the corresponding epipolar lines. Epipolar tangent points act effectively as point correspondences and may be included in estimation algorithms as described by [Porrill-91].

Particularly important cases are those of conics and quadrics which are algebraic objects and so algebraic solutions can be developed. Examples are given in the notes and exercises at the end of this chapter.

11.9 Degeneracies

A set of correspondences $\{\mathbf{x}_i \leftrightarrow \mathbf{x}'_i, i = 1, \ldots, n\}$ is geometrically degenerate with respect to F if it fails to uniquely define the epipolar geometry, or equivalently if there exist linearly independent rank-2 matrices, F_j, $j = 1, 2$, such that

$$\mathbf{x}_i'^{\mathsf{T}}\mathrm{F}_1\mathbf{x}_i^{\mathsf{T}} = 0 \quad \text{and} \quad \mathbf{x}_i'^{\mathsf{T}}\mathrm{F}_2\mathbf{x}_i = 0 \qquad (1 \leq i \leq n) \ .$$

The subject of degeneracy is investigated in detail in chapter 22. However, a brief preview is given now for the two important cases of scene points on a ruled quadric, or on a plane.

Provided the two camera centres are not coincident the epipolar geometry *is* uniquely defined. It can always be computed from the camera matrices P, P' as in (9.1–p244) for example. What is at issue here are configurations where the epipolar geometry cannot be estimated from point correspondences. An awareness of the degeneracies of

Samsung/Google, EX1010, p. 313

$\dim(N) = 1$: Unique solution – no degeneracy.

Arises from $n \geq 8$ point correspondences in general position. If $n > 8$ then the point correspondences must be perfect (i.e. noise-free).

$\dim(N) = 2$: 1 or 3 solutions.

Arises in the case of seven point correspondences, and also in the case of $n > 7$ perfect point correspondences where the 3D points and camera centres lie on a ruled quadric referred to as a critical surface. The quadric may be non-degenerate (a hyperboloid of one sheet) or degenerate.

$\dim(N) = 3$: Two-parameter family of solutions.

Arises if $n \geq 6$ perfect point correspondences are related by a homography, $\mathbf{x}'_i = \mathtt{H}\mathbf{x}_i$.

- Rotation about the camera centre (a degenerate motion).
- All world points on a plane (a degenerate structure).

Table 11.1. *Degeneracies in estimating* F *from point correspondences, classified by the dimension of the null-space* N *of* A *in (11.3–*p279*).*

estimation algorithms is important because configurations "close to" degenerate ones are likely to lead to a numerically ill-conditioned estimation. The degeneracies are summarized in table 11.1.

11.9.1 Points on a ruled quadric

It will be shown in chapter 22 that degeneracy occurs if both camera centres and all the 3D points lie on a (ruled) quadric surface referred to as the *critical surface* [Maybank-93]. A ruled quadric may be non-degenerate (a hyperboloid of one sheet – a cooling tower) or degenerate (for instance two planes, cones, and cylinders) – see section 3.2.4(*p*74); but a critical surface cannot be an ellipsoid or hyperboloid of two sheets. For a critical surface configuration there are three possible fundamental matrices.

Note that in the case of just 7 point correspondences, together with the two camera centres there are 9 points in total. A general quadric has 9 degrees of freedom, and one may always construct a quadric through 9 points. In the case where this quadric is a ruled quadric it will be a critical surface, and there will be three possible solutions for F. The case where the quadric is not ruled corresponds to the case where there is only one real solution for F.

11.9.2 Points on a plane

An important degeneracy is when all the points lie in a plane. In this case, all the points plus the two camera centres lie on a ruled quadric surface, namely the degenerate quadric consisting of two planes – the plane through the points, plus a plane passing through the two camera centres.

Two views of a planar set of points are related via a 2D projective transformation H. Thus, suppose that a set of correspondences $\mathbf{x}_i \leftrightarrow \mathbf{x}'_i$ is given for which $\mathbf{x}'_i = \mathtt{H}\mathbf{x}_i$. Any number of points $\mathbf{x}_i$ and the corresponding points $\mathbf{x}'_i = \mathtt{H}\mathbf{x}_i$ may be given.

Samsung/Google, EX1010, p. 314

The fundamental matrix corresponding to the pair of cameras satisfies the equation $\mathbf{x}_i'^{\mathsf{T}}\mathtt{F}\mathbf{x}_i = \mathbf{x}_i'^{\mathsf{T}}(\mathtt{F}\mathtt{H}^{-1})\mathbf{x}_i' = 0$. This set of equations is satisfied whenever $\mathtt{F}\mathtt{H}^{-1}$ is skew-symmetric. Thus, the solution for F is any matrix of the form $\mathtt{F} = \mathtt{S}\mathtt{H}$, where S is skew-symmetric. Now, a 3×3 skew-symmetric matrix S may be written in the form $\mathtt{S} = [\mathbf{t}]_\times$, for any 3-vector $\mathbf{t}$. Thus, S has three degrees of freedom, and consequently so does F. More precisely, the correspondences $\mathbf{x}_i \leftrightarrow \mathbf{x}_i'$ lead to a three-parameter family of possible fundamental matrices F (note, one of the parameters accounts for scaling the matrix so there is only a *two*-parameter family of homogeneous matrices). The equation matrix A derived from the set of correspondences must therefore have rank no greater than 6.

From the decomposition of $\mathtt{F} = \mathtt{S}\mathtt{H}$, it follows from result 9.9(p254) that the pair of camera matrices $[\mathtt{I} \mid \mathbf{0}]$ and $[\mathtt{H} \mid \mathbf{t}]$ correspond to the fundamental matrix F. Here, the vector $\mathbf{t}$ may take on any value. If point $\mathbf{x}_i = (x_i, y_i, 1)^{\mathsf{T}}$ and $\mathbf{x}_i' = \mathtt{H}\mathbf{x}_i$, then one verifies that the point $\mathbf{X}_i = (x, y, 1, 0)^{\mathsf{T}}$ maps to $\mathbf{x}_i$ and $\mathbf{x}_i'$ through the two cameras. Thus, the points $\mathbf{X}_i$ constitute a reconstruction of the scene.

11.9.3 No translation

If the two camera centres are coincident then the epipolar geometry is not defined. In addition, formulae such as result 9.9(p254) give a value of 0 for the fundamental matrix. In this case the two images are related by a 2D homography (see section 8.4.2(p204)).

If one attempts to find the fundamental matrix then, as shown above, there will be at least a 2-parameter family of solutions for F. Even if the camera motion involves no translation, then a method such as the 8-point algorithm used to compute the fundamental matrix will still produce a matrix F satisfying $\mathbf{x}_i'^{\mathsf{T}}\mathtt{F}\mathbf{x}_i = 0$, where F has the form $\mathtt{F} = \mathtt{S}\mathtt{H}$, H is the homography relating the points, and S is an essentially arbitrary skew-symmetric matrix. Points $\mathbf{x}_i$ and $\mathbf{x}_i'$ related by H will satisfy this relationship.

11.10 A geometric interpretation of F-computation

The estimation of F from a set of image correspondences $\{\mathbf{x}_i \leftrightarrow \mathbf{x}_i'\}$ has many similarities with the problem of estimating a conic from a set of 2D points $\{x_i, y_i\}$ (or a quadric from a set of 3D points).

The equation $\mathbf{x}'^{\mathsf{T}}\mathtt{F}\mathbf{x} = 0$ is a single constraint in x, y, x', y' and so defines a surface (variety) $\mathcal{V}$ of codimension 1 (dimension 3) in $\mathbb{R}^4$. The surface is a quadric because the equation is quadratic in the coordinates x, y, x', y' of $\mathbb{R}^4$. There is a natural mapping from projective 3-space to the variety $\mathcal{V}$ that takes any 3D point to the quadruple $(x, y, x', y')^{\mathsf{T}}$ of the corresponding image points in the two views. The quadric form is evident if $\mathbf{x}'^{\mathsf{T}}\mathtt{F}\mathbf{x} = 0$ is rewritten as

$$\begin{pmatrix} x & y & x' & y' & 1 \end{pmatrix} \begin{bmatrix} 0 & 0 & f_{11} & f_{21} & f_{31} \\ 0 & 0 & f_{12} & f_{22} & f_{32} \\ f_{11} & f_{12} & 0 & 0 & f_{13} \\ f_{21} & f_{22} & 0 & 0 & f_{23} \\ f_{31} & f_{32} & f_{13} & f_{23} & 2f_{33} \end{bmatrix} \begin{pmatrix} x \\ y \\ x' \\ y' \\ 1 \end{pmatrix} = 0 \ .$$

The case of conic fitting is a good (lower-dimensional) model of F estimation. To

Samsung/Google, EX1010, p. 315

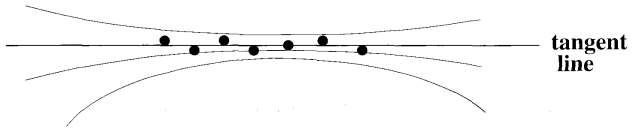


Fig. 11.7. *Estimating a conic from point data (shown as •) may be poorly conditioned. All of the conics shown have residuals within the point error distribution. However, even though there is ambiguity in the estimated conic, the tangent line is well defined, and can be computed from the points.*

bring out the analogy between the two estimation problems: a point (x_i, y_i) places one constraint on the 5 degrees of freedom of a conic as described in section 2.2.3(*p*30):

$$ax_i^2 + bx_iy_i + cy_i^2 + dx_i + ey_i + f = 0.$$

Similarly, a point correspondence (x_i, y_i, x_i', y_i') places one constraint on the (8) degrees of freedom of F as (11.2–*p*279):

$$x_i'x_if_{11} + x_i'y_if_{12} + x_i'f_{13} + y_i'x_if_{21} + y_i'y_if_{22} + y_i'f_{23} + x_if_{31} + y_if_{32} + f_{33} = 0.$$

It is not quite an exact analogy, since the defining relationship expressed by the fundamental matrix is bilinear in the two sets of indices, as is also evident from the zeros in the quadric matrix above, whereas in the case of a conic section the equation is an arbitrary quadratic. Also the surface defined by F must satisfy an additional constraint arising from $\det(\mathtt{F}) = 0$, and there is no such constraint in the conic fitting analogue.

The problems of extrapolation when data has only been fitted to a small section of a conic are well known, and similar issues arise in fitting the fundamental matrix to data. Indeed, there are cases where the data is sufficient to determine an accurate tangent line to the conic, but insufficient to determine the conic itself, see figure 11.7. In the case of the fundamental matrix the tangent plane to the quadric in $\mathbb{R}^4$ is the affine fundamental matrix (chapter 14), and this approximation may be fitted when perspective effects are small.

11.11 The envelope of epipolar lines

One of the uses of the fundamental matrix is to determine epipolar lines in a second image corresponding to points in a first image. For instance, if one is seeking matched points between two images, the match of a given point x in the first image may be found by searching along the epipolar line Fx in the second image. In the presence of noise, of course, the matching point will not lie precisely on the line Fx because the fundamental matrix will be known only within certain bounds, expressed by its covariance matrix. In general, instead of searching along the epipolar line only, it will be necessary to search in a region on either side of the line Fx. We will now consider how the covariance matrix of the fundamental matrix may be used to determine the region in which to search.

Let x be a point and F be a fundamental matrix for which one has computed a covariance matrix $\Sigma_{\mathtt{F}}$. The point x corresponds to an epipolar line $\mathtt{l} = \mathtt{Fx}$, and one

Samsung/Google, EX1010, p. 316

may transfer the covariance matrix $\Sigma_{\mathtt{F}}$ to a covariance matrix $\Sigma_{\mathtt{l}}$ according to result 5.6-(p139). Also by result 5.6(p139), the mean value of the epipolar line is given by $\bar{\mathtt{l}} = \bar{\mathtt{F}}\mathbf{x}$. To avoid singular cases, the vector $\mathtt{l}$ representing an epipolar line is normalized so that $\|\mathtt{l}\| = 1$. Then the mapping $\mathbf{x} \mapsto \mathtt{l}$ is given by $\mathtt{l} = (\mathtt{F}\mathbf{x})/\|\mathtt{F}\mathbf{x}\|$. If $\mathtt{J}$ is the Jacobian matrix of this mapping with respect to the entries of $\mathtt{F}$, then $\mathtt{J}$ is a 3×9 matrix, and $\Sigma_{\mathtt{l}} = \mathtt{J}\Sigma_{\mathtt{F}}\mathtt{J}^{\mathsf{T}}$.

Though the constraint $\|\mathtt{l}\| = 1$ is the most convenient constraint, the following analysis applies for any constraint used to confine the vector representing the epipolar line to vary on a 2-dimensional surface in $\mathbb{R}^3$. In this case, the covariance matrix $\Sigma_{\mathtt{l}}$ is singular, having rank 2, since no variation is allowed in the direction normal to the constraint surface. For a particular instance of $\mathtt{l}$, the deviation from the mean, $\bar{\mathtt{l}} - \mathtt{l}$, must be along the constraint surface, and hence (in the linear approximation) perpendicular to the null-space of $\Sigma_{\mathtt{l}}$.

For the remainder of this derivation, $\bar{\mathtt{l}}$, the vector representing the mean epipolar line, will be denoted by $\mathbf{m}$, so as to avoid confusing notation. Now, assuming a Gaussian distribution for the vectors $\mathtt{l}$ representing the epipolar line, the set of all lines having a given likelihood is given by the equation

$$(\mathtt{l} - \mathbf{m})^{\mathsf{T}}\Sigma_{\mathtt{l}}^{+}(\mathtt{l} - \mathbf{m}) = k^2 \tag{11.11}$$

where k is some constant. To analyze this further, we apply an orthogonal change of coordinates such that $\Sigma_{\mathtt{l}}$ becomes diagonal. Thus, one may write

$$\mathtt{U}\Sigma_{\mathtt{l}}\mathtt{U}^{\mathsf{T}} = \Sigma'_{\mathtt{l}} = \begin{bmatrix} \tilde{\Sigma}'_{\mathtt{l}} & \mathbf{0} \\ \mathbf{0}^{\mathsf{T}} & 0 \end{bmatrix}$$

where $\tilde{\Sigma}'_{\mathtt{l}}$ is a 2×2 non-singular diagonal matrix. Applying the same transformation to the lines, one defines 2-vectors $\mathbf{m}' = \mathtt{U}\mathbf{m}$ and $\mathtt{l}' = \mathtt{U}\mathtt{l}$. Since $\mathtt{l}' - \mathbf{m}'$ is orthogonal to the null-space $(0, 0, 1)^{\mathsf{T}}$ of $\Sigma'_{\mathtt{l}}$, both $\mathbf{m}'$ and $\mathtt{l}'$ have the same third coordinate. By multiplying $\mathtt{U}$ by a constant as necessary, one may assume that this coordinate is 1. Thus we may write $\mathtt{l}' = (\tilde{\mathtt{l}}'^{\mathsf{T}}, 1)^{\mathsf{T}}$ and $\mathbf{m}' = (\tilde{\mathbf{m}}'^{\mathsf{T}}, 1)^{\mathsf{T}}$ for certain 2-vectors $\tilde{\mathtt{l}}'$ and $\tilde{\mathbf{m}}'$. Then, one verifies that

$$\begin{aligned} k^2 &= (\mathtt{l} - \mathbf{m})^{\mathsf{T}}\Sigma_{\mathtt{l}}^{+}(\mathtt{l} - \mathbf{m}) \\ &= (\mathtt{l}' - \mathbf{m}')^{\mathsf{T}}\Sigma_{\mathtt{l}}'^{+}(\mathtt{l}' - \mathbf{m}') \\ &= (\tilde{\mathtt{l}}' - \tilde{\mathbf{m}}')^{\mathsf{T}}\tilde{\Sigma}_{\mathtt{l}}'^{-1}(\tilde{\mathtt{l}}' - \tilde{\mathbf{m}}'). \end{aligned}$$

This equation expands out to

$$\tilde{\mathtt{l}}'^{\mathsf{T}}\tilde{\Sigma}_{\mathtt{l}}'^{-1}\tilde{\mathtt{l}}' - \tilde{\mathbf{m}}'^{\mathsf{T}}\tilde{\Sigma}_{\mathtt{l}}'^{-1}\tilde{\mathtt{l}}' - \tilde{\mathtt{l}}'^{\mathsf{T}}\tilde{\Sigma}_{\mathtt{l}}'^{-1}\tilde{\mathbf{m}}' + \tilde{\mathbf{m}}'^{\mathsf{T}}\tilde{\Sigma}_{\mathtt{l}}'^{-1}\tilde{\mathbf{m}}' - k^2 = 0$$

which may be written as

$$(\tilde{\mathtt{l}}'^{\mathsf{T}} \;\; 1)\begin{bmatrix} \tilde{\Sigma}_{\mathtt{l}}'^{-1} & -\tilde{\Sigma}_{\mathtt{l}}'^{-1}\tilde{\mathbf{m}}' \\ -\tilde{\mathbf{m}}'^{\mathsf{T}}\tilde{\Sigma}_{\mathtt{l}}'^{-1} & \tilde{\mathbf{m}}'^{\mathsf{T}}\tilde{\Sigma}_{\mathtt{l}}'^{-1}\tilde{\mathbf{m}}' - k^2 \end{bmatrix}\begin{pmatrix} \tilde{\mathtt{l}}' \\ 1 \end{pmatrix} = 0$$

Samsung/Google, EX1010, p. 317

or equivalently (as one may verify)

$$(\tilde{\mathbf{l}}'^{\mathsf{T}} \;\; 1) \begin{bmatrix} \tilde{\mathbf{m}}'\tilde{\mathbf{m}}'^{\mathsf{T}} - k^2\tilde{\Sigma}_{\mathbf{l}}' & \tilde{\mathbf{m}}' \\ \tilde{\mathbf{m}}'^{\mathsf{T}} & 1 \end{bmatrix}^{-1} \begin{pmatrix} \tilde{\mathbf{l}}' \\ 1 \end{pmatrix} = 0. \tag{11.12}$$

Finally, this is equivalent to

$$\mathbf{l}'^{\mathsf{T}}[\mathbf{m}'\mathbf{m}'^{\mathsf{T}} - k^2\Sigma_{\mathbf{l}}']^{-1}\mathbf{l}' = 0. \tag{11.13}$$

This shows that the lines satisfying (11.11) form a line conic defined by the matrix $(\mathbf{m}'\mathbf{m}'^{\mathsf{T}} - k^2\Sigma_{\mathbf{l}}')^{-1}$. The corresponding point conic, which forms the envelope of the lines, is defined by the matrix $\mathbf{m}'\mathbf{m}'^{\mathsf{T}} - k^2\Sigma_{\mathbf{l}}'$. One may now transform back to the original coordinate system to determine the envelope of the lines in the original coordinate system. The transformed conic is

$$\mathtt{C} = \mathtt{U}^{\mathsf{T}}(\mathbf{m}'\mathbf{m}'^{\mathsf{T}} - k^2\Sigma_{\mathbf{l}}')\mathtt{U} = \mathbf{m}\mathbf{m}^{\mathsf{T}} - k^2\Sigma_{\mathbf{l}}. \tag{11.14}$$

Note that when $k = 0$, the conic $\mathtt{C}$ degenerates to $\mathbf{m}\mathbf{m}^{\mathsf{T}}$, which represents the set of points lying on the line $\mathbf{m}$. As k increases, the conic becomes a hyperbola the two branches of which lie on opposite sides of the line $\mathbf{m}$.

Suppose we want to choose k so that some fraction α of the epipolar lines lie inside the region bounded by this hyperbola. The value $k^2 = (\mathbf{l} - \mathbf{m})^{\mathsf{T}}\Sigma_{\mathbf{l}}^{+}(\mathbf{l} - \mathbf{m})$ of (11.11) follows a χ_n^2 distribution, and the cumulative chi-squared distribution $F_n(k^2) = \int_0^{k^2} \chi_n^2(\xi)d\xi$ represents the probability that the value of a χ_n^2 random variable is less than k^2 (the χ_n^2 and F_n distributions are defined in section A2.2(*p*566)). Applying this to a random line $\mathbf{l}$, one sees that in order to ensure that a fraction α of lines lie within region bounded by the hyperbola defined by (11.14), one must choose k^2 such that $F_2(k^2) = \alpha$ ($n = 2$ since the covariance matrix $\Sigma_{\mathbf{l}}$ has rank 2). Thus, $k^2 = F_2^{-1}(\alpha)$, and for a value of $\alpha = 0.95$, for instance, one finds that $k^2 = 5.9915$. The corresponding hyperbola given by (11.14) is $\mathtt{C} = \mathbf{m}\mathbf{m}^{\mathsf{T}} - 5.9915\,\Sigma_{\mathbf{l}}$. To sum up this discussion:

Result 11.2. *If* $\mathbf{l}$ *is a random line obeying a Gaussian distribution with mean* $\bar{\mathbf{l}}$ *and covariance matrix* $\Sigma_{\mathbf{l}}$ *of rank 2, then the plane conic*

$$\mathtt{C} = \bar{\mathbf{l}}\bar{\mathbf{l}}^{\mathsf{T}} - k^2\Sigma_{\mathbf{l}} \tag{11.15}$$

represents an equal-likelihood contour bounding some fraction of all instances of $\mathbf{l}$. *If* $F_2(k^2)$ *represents the cumulative* χ_2^2 *distribution, and* k^2 *is chosen such that* $F_2^{-1}(k^2) = \alpha$, *then a fraction* α *of all lines lie within the region bounded by* $\mathtt{C}$. *In other words with probability* α *the lines lie within this region.*

In applying this formula, one must be aware that it represents only an approximation, since epipolar lines are not normally distributed. We have consistently made the assumption that the distributions may be correctly transformed using the Jacobian, that is an assumption of linearity. This assumption will be most reasonable for distributions with small variance, and close to the mean. Here, we are applying it to find the region in which as many as 95% of samples fall, namely almost the whole of the error

Samsung/Google, EX1010, p. 318

distribution. In this case, the assumption of a Gaussian distribution of errors is less tenable.

11.11.1 Verification of epipolar line covariance

We now present some examples of epipolar line envelopes, confirming and illustrating the theory developed above. Before doing this, however, a direct verification of the theory will be given, concerning the covariance matrix of epipolar lines. Since the 3×3 covariance matrix of a line is not easily understood quantitatively, we consider the variance of the direction of epipolar lines. Given a line $\mathbf{l} = (l_1, l_2, l_3)^\mathsf{T}$, the angle representing its direction is given by $\theta = \arctan(-l_1/l_2)$. Letting $\mathtt{J}$ equal the 1×3 Jacobian matrix of the mapping $\mathbf{l} \rightarrow \theta$, one finds the variance of the angle θ to be $\sigma_\theta^2 = \mathtt{J}\Sigma_\mathbf{l}\mathtt{J}^\mathsf{T}$. This result may be verified by simulation, as follows.

One considers a pair of images for which point correspondences have been identified. The fundamental matrix is computed from the point correspondences and the points are then corrected so as to correspond precisely under the epipolar mapping (as described in section 12.3). A set of n of these corrected correspondences are used to compute the covariance matrix of the fundamental matrix $\mathtt{F}$. Then, for a further set of "test" corrected points $\mathbf{x}_i$ in the first image, the mean and covariance of the corresponding epipolar line $\mathbf{l}'_i = \mathtt{F}\mathbf{x}_i$ are computed, and subsequently the mean and variance of the orientation direction of this line are computed. This gives the theoretical values of these quantities.

Next, Monte Carlo simulation is done, in which Gaussian noise is added to the coordinates of the points used to compute $\mathtt{F}$. Using the computed $\mathtt{F}$, the epipolar lines corresponding to each of the test points are computed, and subsequently their angle, and the deviation of the angle from the mean. This is done many times, and the standard deviation of angle is computed, and finally compared with the theoretical value. The results of this are shown in figure 11.8 for the statue image pair of figure 11.2-(p289).

Epipolar envelopes for statue image. The statue image pair of figure 11.2(p289) is interesting because of the large depth variation across the image. There are close points (on the statue) and distant points (on the building behind) in close proximity in the images. The fundamental matrix was computed from several points. A point in the first image (see figure 11.9) was selected and Monte Carlo simulation was used to compute several possible epipolar lines corresponding to a noise level of 0.5 pixels in each matched point coordinate. To test the theory, the mean and covariance of the epipolar line were next computed theoretically. The 95% envelope of the epipolar lines was computed and drawn in the second image. The results are shown in figure 11.10 for different numbers of points used to compute $\mathtt{F}$. The 95% envelope for $n = 15$ corresponds closely to the simulated envelope of the lines.

The results shown in figure 11.10 show the practical importance of computing the epipolar envelopes in point matching. Thus, suppose one is attempting to find a match for the foreground point in figure 11.9. If the epipolar line is computed from just 10 point matches, then epipolar search is unlikely to succeed, given the width of the

Samsung/Google, EX1010, p. 319

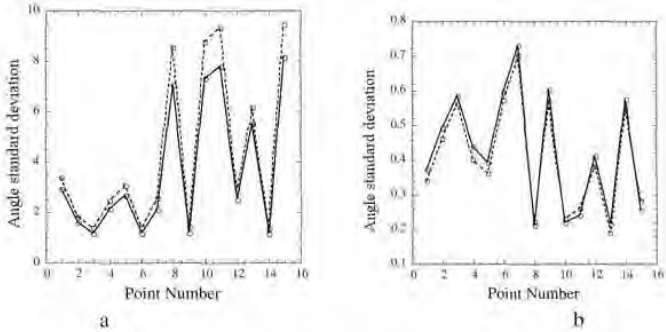


Fig. 11.8. *Comparison of theoretical and Monte Carlo simulated values of orientation angle of epipolar lines for 15 test points form the statue image pair of figure 11.2*(p289). *The horizontal axis represents the point number (1 to 15) and the vertical axis the standard deviation of angle. (a) the results when the epipolar structure (fundamental matrix) is computed from 15 points. (b) the results when 50 point matches are used.* **Note :** *the horizontal axis of these graphs represent discrete points numbered 1 to 15. The graphs are shown as a continuous curve only for visual clarity.*

envelope. Even for $n = 15$, the width of the envelope at the level of the correct match is several tens of pixels. For $n = 25$, the situation is more favourable. Note that this instability is inherent in the problem, and not the result of any specific algorithm for computing F.

An interesting point concerns the location of the narrowest point of the envelope. In this case, it appears to be close to the correct match position for the background point in figure 11.9. The match for the foreground point (leg of statue) lies far from the narrowest point of the envelope. Though the precise location of the narrow point of the envelope is not fully understood, it appears that in this case, most points used to the computation of F are on the background building. This biases towards the supposition that other matched points lie close to the plane of the building. The match for a point at significantly different depth is less precisely known.

Matching points close to the epipole – the corridor scene. In the case where the points to be matched are close to the epipole, then the determination of the epipolar line is more unstable, since any uncertainty in the position of the epipole results in uncertainty in the slope of the epipolar line. In addition, as one approaches this unstable position, the linear approximations implicit in the derivation of (11.14) become less tenable. In particular, the distribution of the epipolar lines deviates from a normal distribution.

11.12 Image rectification

This section gives a method for image rectification, the process of resampling pairs of stereo images taken from widely differing viewpoints in order to produce a pair of "matched epipolar projections". These are projections in which the epipolar lines run parallel with the x-axis and match up between views, and consequently disparities between the images are in the x-direction only, i.e. there is no y disparity.

Samsung/Google, EX1010, p. 320



a b

Fig. 11.9. *(a) The point in the first image used to compute the epipolar envelopes in the second images. Note the ambiguity of which point is to be found in the second image. The marked point may represent the point on the statue's leg (foreground) or the point on the building behind the statue (background). In the second image, these two points are quite separate, and the epipolar line must pass through them both. (b) Computed corresponding epipolar lines computed from $n = 15$ point matches. The different lines correspond to different instances of injected noise in the matched points. Gaussian noise of 0.5 pixels in each coordinate was added to the ideal matched point positions before computing the epipolar line corresponding to the selected point. The ML estimator (Gold Standard algorithm) was used to compute* F. *This experiment shows the basic instability of the computation of the epipolar lines from small numbers of points. To find the point matching the selected point in the image at left, one needs to search over the regions covered by all these epipolar lines.*

The method is based on the fundamental matrix. A pair of 2D projective transformations are applied to the two images in order to match the epipolar lines. It is shown that the two transformations may be chosen in such a way that matching points have approximately the same x-coordinate as well. In this way, the two images, if overlaid on top of each other, will correspond as far as possible, and any disparities will be parallel to the x-axis. Since the application of arbitrary 2D projective transformations may distort the image substantially, the method for finding the pair of transformations subjects the images to a minimal distortion.

In effect, transforming the two images by the appropriate projective transformations reduces the problem to the epipolar geometry produced by a pair of identical cameras placed side by side with their principal axes parallel. Many stereo matching algorithms described in previous literature have assumed this geometry. After this rectification the search for matching points is vastly simplified by the simple epipolar structure and by the near-correspondence of the two images. It may be used as a preliminary step to comprehensive image matching.

11.12.1 Mapping the epipole to infinity

In this section we will discuss the question of finding a projective transformation H of an image that maps the epipole to a point at infinity. In fact, if epipolar lines are to be transformed to lines parallel with the x-axis, then the epipole should be mapped to

Samsung/Google, EX1010, p. 321



Fig. 11.10. *The 95% envelopes of epipolar lines are shown for a noise level of 0.5 pixels, with* F *being computed from* $n = 10, 15, 25$ *and* 50 *points. In each case, Monte Carlo simulated results agreed closely with these results (though not shown here). For the case* $n = 15$*, compare with figure 11.9. Note that for* $n = 10$*, the epipolar envelope is very wide (> 90 degrees), showing that one can have very little confidence in an epipolar line computed from 10 points in this case. For* $n = 15$*, the envelope is still quite wide. For* $n = 25$ *and* $n = 50$*, the epipolar line is known with quite good precision. Of course, the precise shape of the envelope depends strongly on just what matched points are used to compute the epipolar structure.*

the particular infinite point $(1, 0, 0)^\mathsf{T}$. This leaves many degrees of freedom (in fact four) open for H, and if an inappropriate H is chosen, severe projective distortion of the image can take place. In order that the resampled image should look somewhat like the original image, we may put closer restrictions on the choice of H.

One condition that leads to good results is to insist that the transformation H should act as far as possible as a rigid transformation in the neighbourhood of a given selected point $\mathbf{x}_0$ of the image. By this is meant that to first-order the neighbourhood of $\mathbf{x}_0$ may undergo rotation and translation only, and hence will look the same in the original and resampled images. An appropriate choice of point $\mathbf{x}_0$ may be the centre of the image. For instance, this would be a good choice in the context of aerial photography if the view is known not to be excessively oblique.

For the present, suppose $\mathbf{x}_0$ is the origin and the epipole $\mathbf{e} = (f, 0, 1)^\mathsf{T}$ lies on the

Samsung/Google, EX1010, p. 322

x-axis. Now consider the following transformation

$$\mathtt{G} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ -1/f & 0 & 1 \end{bmatrix}. \tag{11.16}$$

This transformation takes the epipole $(f, 0, 1)^\mathsf{T}$ to the point at infinity $(f, 0, 0)^\mathsf{T}$ as required. A point $(x, y, 1)^\mathsf{T}$ is mapped by G to the point $(\hat{x}, \hat{y}, 1)^\mathsf{T} = (x, y, 1 - x/f)^\mathsf{T}$. If $|x/f| < 1$ then we may write

$$(\hat{x}, \hat{y}, 1)^\mathsf{T} = (x, y, 1 - x/f)^\mathsf{T} = (x(1 + x/f + \ldots), y(1 + x/f + \ldots), 1)^\mathsf{T}.$$

The Jacobian is

$$\frac{\partial(\hat{x}, \hat{y})}{\partial(x, y)} = \begin{bmatrix} 1 + 2x/f & 0 \\ y/f & 1 + x/f \end{bmatrix}$$

plus higher order terms in x and y. Now if $x = y = 0$ then this is the identity map. In other words, G is approximated (to first-order) at the origin by the identity mapping.

For an arbitrarily placed point of interest $\mathbf{x}_0$ and epipole $\mathbf{e}$, the required mapping H is a product $\mathtt{H} = \mathtt{GRT}$ where T is a translation taking the point $\mathbf{x}_0$ to the origin, R is a rotation about the origin taking the epipole $\mathbf{e}'$ to a point $(f, 0, 1)^\mathsf{T}$ on the x-axis, and G is the mapping just considered taking $(f, 0, 1)^\mathsf{T}$ to infinity. The composite mapping is to first-order a rigid transformation in the neighbourhood of $\mathbf{x}_0$.

11.12.2 Matching transformations

In the previous section it was shown how the epipole in one image may be mapped to infinity. Next, it will be seen how a map may be applied to the other image to match up the epipolar lines. We consider two images J and J'. The intention is to resample these two images according to transformations H to be applied to J and H′ to be applied to J'. The resampling is to be done in such a way that an epipolar line in J is matched with its corresponding epipolar line in J'. More specifically, if $\mathbf{l}$ and $\mathbf{l}'$ are any pair of corresponding epipolar lines in the two images, then $\mathtt{H}^{-\mathsf{T}}\mathbf{l} = \mathtt{H}'^{-\mathsf{T}}\mathbf{l}'$. (Recall that $\mathtt{H}^{-\mathsf{T}}$ is the line map corresponding to the point map H.) Any pair of transformations satisfying this condition will be called a *matched pair* of transformations.

Our strategy in choosing a matched pair of transformations is to choose H′ first to be some transformation that sends the epipole $\mathbf{e}'$ to infinity as described in the previous section. We then seek a matching transformation H chosen so as to minimize the sum-of-squared distances

$$\sum_i d(\mathtt{H}\mathbf{x}_i, \mathtt{H}'\mathbf{x}_i')^2. \tag{11.17}$$

The first question to be determined is how to find a transformation matching H′. That question is answered in the following result.

Result 11.3. *Let J and J' be images with fundamental matrix $\mathtt{F} = [\mathbf{e}']_\times \mathtt{M}$, and let H′ be a projective transformation of J'. A projective transformation H of J matches H′ if and*

Samsung/Google, EX1010, p. 323

only if $\mathtt{H}$ *is of the form*

$$\mathtt{H} = (\mathtt{I} + \mathtt{H}'\mathbf{e}'\mathbf{a}^\mathsf{T})\mathtt{H}'\mathtt{M} \tag{11.18}$$

for some vector $\mathbf{a}$.

Proof. If $\mathbf{x}$ is a point in J, then $\mathbf{e} \times \mathbf{x}$ is the epipolar line in the first image, and $\mathtt{F}\mathbf{x}$ is the epipolar line in the second image. Transformations $\mathtt{H}$ and $\mathtt{H}'$ are a matching pair if and only if $\mathtt{H}^{-\mathsf{T}}(\mathbf{e} \times \mathbf{x}) = \mathtt{H}'^{-\mathsf{T}}\mathtt{F}\mathbf{x}$. Since this must hold for all $\mathbf{x}$ we may write equivalently $\mathtt{H}^{-\mathsf{T}}[\mathbf{e}]_\times = \mathtt{H}'^{-\mathsf{T}}\mathtt{F} = \mathtt{H}'^{-\mathsf{T}}[\mathbf{e}']_\times\mathtt{M}$ or, applying result A4.3(p582),

$$[\mathtt{H}\mathbf{e}]_\times\mathtt{H} = [\mathtt{H}'\mathbf{e}']_\times\mathtt{H}'\mathtt{M}. \tag{11.19}$$

In view of lemma 9.11(p255), this implies $\mathtt{H} = (\mathtt{I} + \mathtt{H}'\mathbf{e}'\mathbf{a}^\mathsf{T})\mathtt{H}'\mathtt{M}$ as required.
To prove the converse, if (11.18) holds, then

$$\begin{aligned} \mathtt{H}\mathbf{e} &= (\mathtt{I} + \mathtt{H}'\mathbf{e}'\mathbf{a}^\mathsf{T})\mathtt{H}'\mathtt{M}\mathbf{e} = (\mathtt{I} + \mathtt{H}'\mathbf{e}'\mathbf{a}^\mathsf{T})\mathtt{H}'\mathbf{e}' \\ &= (1 + \mathbf{a}^\mathsf{T}\mathtt{H}'\mathbf{e}')\mathtt{H}'\mathbf{e}' = \mathtt{H}'\mathbf{e}'. \end{aligned}$$

This, along with (11.18), is sufficient for (11.19) to hold, and so $\mathtt{H}$ and $\mathtt{H}'$ are matching transformations. □

We are particularly interested in the case when $\mathtt{H}'$ is a transformation taking the epipole $\mathbf{e}'$ to a point at infinity $(1, 0, 0)^\mathsf{T}$. In this case, $\mathtt{I} + \mathtt{H}'\mathbf{e}'\mathbf{a}^\mathsf{T} = \mathtt{I} + (1, 0, 0)^\mathsf{T}\mathbf{a}^\mathsf{T}$ is of the form

$$\mathtt{H}_A = \begin{bmatrix} a & b & c \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{11.20}$$

which represents an affine transformation. Thus, a special case of result 11.3 is

Corollary 11.4. *Let* J *and* J' *be images with fundamental matrix* $\mathtt{F} = [\mathbf{e}']_\times\mathtt{M}$, *and let* $\mathtt{H}'$ *be a projective transformation of* J' *mapping the epipole* $\mathbf{e}'$ *to the infinite point* $(1, 0, 0)^\mathsf{T}$. *A transformation* $\mathtt{H}$ *of* J *matches* $\mathtt{H}'$ *if and only if* $\mathtt{H}$ *is of the form* $\mathtt{H} = \mathtt{H}_A\mathtt{H}_0$, *where* $\mathtt{H}_0 = \mathtt{H}'\mathtt{M}$ *and* $\mathtt{H}_A$ *is an affine transformation of the form (11.20).*

Given $\mathtt{H}'$ mapping the epipole to infinity, we may use this corollary to make the choice of a matching transformation $\mathtt{H}$ to minimize the disparity. Writing $\hat{\mathbf{x}}'_i = \mathtt{H}'\mathbf{x}'_i$ and $\hat{\mathbf{x}}_i = \mathtt{H}_0\mathbf{x}_i$, the minimization problem (11.17) is to find $\mathtt{H}_A$ of the form (11.20) such that

$$\sum_i d(\mathtt{H}_A\hat{\mathbf{x}}_i, \hat{\mathbf{x}}'_i)^2 \tag{11.21}$$

is minimized.

In particular, let $\hat{\mathbf{x}}_i = (\hat{x}_i, \hat{y}_i, 1)^\mathsf{T}$, and let $\hat{\mathbf{x}}'_i = (\hat{x}'_i, \hat{y}'_i, 1)^\mathsf{T}$. Since $\mathtt{H}'$ and $\mathtt{M}$ are known, these vectors may be computed from the matched points $\mathbf{x}_i \leftrightarrow \mathbf{x}'_i$. Then the quantity to be minimized (11.21) may be written as

$$\sum_i (a\hat{x}_i + b\hat{y}_i + c - \hat{x}'_i)^2 + (\hat{y}_i - \hat{y}'_i)^2.$$

Samsung/Google, EX1010, p. 324

Since $(\tilde{y}_i - \tilde{y}'_i)^2$ is a constant, this is equivalent to minimizing

$$\sum_i (a\tilde{x}_i + b\tilde{y}_i + c - \tilde{x}'_i)^2.$$

This is a simple linear least-squares parameter minimization problem, and is easily solved using linear techniques (see section A5.1($p588$)) to find a, b and c. Then H_A is computed from (11.20) and $\mathtt{H}$ from (11.18). Note that a linear solution is possible because $\mathtt{H}_A$ is an affine transformation. If it were simply a projective transformation, this would not be a linear problem.

11.12.3 Algorithm outline

The resampling algorithm will now be summarized. The input is a pair of images containing a common overlap region. The output is a pair of images resampled so that the epipolar lines in the two images are horizontal (parallel with the x-axis), and such that corresponding points in the two images are as close to each other as possible. Any remaining disparity between matching points will be along the the horizontal epipolar lines. A top-level outline of the algorithm is as follows.

(i) Identify a seed set of image-to-image matches $\mathbf{x}_i \leftrightarrow \mathbf{x}'_i$ between the two images. Seven points at least are needed, though more are preferable. It is possible to find such matches by automatic means.

(ii) Compute the fundamental matrix $\mathtt{F}$ and find the epipoles $\mathbf{e}$ and $\mathbf{e}'$ in the two images.

(iii) Select a projective transformation $\mathtt{H}'$ that maps the epipole $\mathbf{e}'$ to the point at infinity, $(1, 0, 0)^\mathsf{T}$. The method of section 11.12.1 gives good results.

(iv) Find the matching projective transformation $\mathtt{H}$ that minimizes the least-squares distance

$$\sum_i d(\mathtt{H}\mathbf{x}_i, \mathtt{H}'\mathbf{x}'_i). \tag{11.22}$$

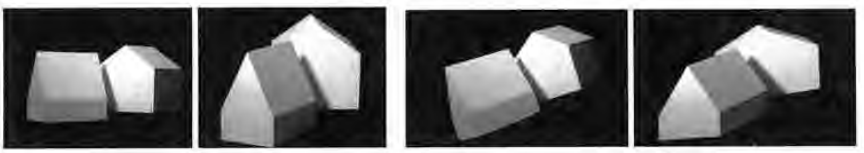
The method used is a linear method described in section 11.12.2.

(v) Resample the first image according to the projective transformation $\mathtt{H}$ and the second image according to the projective transformation $\mathtt{H}'$.

Example 11.5. Model house images

Figure 11.11(a) shows a pair of images of some wooden block houses. Edges and vertices in these two images were extracted automatically and a small number of common vertices were matched by hand. The two images were then resampled according to the methods described here. The results are shown in figure 11.11(b). In this case, because of the wide difference in viewpoint, and the three-dimensional shape of the objects, the two images even after resampling look quite different. However, it is the case that any point in the first image will now match a point in the second image with the same y-coordinate. Therefore, in order to find further point matches between the images only a 1-dimensional search is required. △

Samsung/Google, EX1010, p. 325



a b

Fig. 11.11. **Image rectification.** *(a) A pair of images of a house. (b) Resampled images computed from (a) using a projective transformation computed from* F*. Note, corresponding points in (b) match horizontally.*



a b

Fig. 11.12. **Image rectification using affinities.** *(a) A pair of original images and (b) a detail of the images rectified using affine transformations. The average y-disparity after rectification is of the order of 3 pixels in a* 512×512 *image. (For correctly rectified images the y-disparity should be zero.)*

11.12.4 Affine rectification

The theory discussed in this section can equally be applied to affine resampling. If the two cameras can be well approximated by affine cameras, then one can rectify the images using just affine transformations. To do this, one uses the affine fundamental matrix (see section 14.2(p345)) instead of the general fundamental matrix. The above method with only minor variations can then be applied to compute a pair of matching affine transformations. Figure 11.12 shows a pair of images rectified using affine transformations.

11.13 Closure

11.13.1 The literature

The basic idea behind the computation of the fundamental matrix is given in [LonguetHiggins-81], which is well worth reading. It addresses the case of calibrated matrices only, but the principles apply to the uncalibrated case as well. A good reference for the uncalibrated case is [Zhang-98] which considers most of the best methods. In addition, that paper considers the uncertainty envelopes of epipolar lines, following earlier work by Csurka *et al.* [Csurka-97]. A more detailed study of the 8-point algorithm in the uncalibrated case is given in [Hartley-97c]. Weng *et al.* [Weng-89] used Sampson approximation for the fundamental matrix cost function. The SVD method of coercing the estimated F to have rank 2 was suggested by Tsai & Huang [Tsai-84].

There is a wealth of literature on conic fitting – minimizing algebraic distance

Samsung/Google, EX1010, p. 326

[Bookstein-79]; approximating geometric distance [Sampson-82, Pratt-87, Taubin-91]; optimal fitting [Kanatani-94]; and fitting special forms [Fitzgibbon-99].

11.13.2 Notes and exercises

(i) Six point correspondences constrain $\mathbf{e}$ and $\mathbf{e}'$ to a plane cubic in each image ([Faugeras-93], page 298). The cubic also passes through the six points in each image. A sketch derivation of these results follows. Given six correspondences, the null-space of $\mathtt{A}$ in (11.3–$p279$) will be 3-dimensional. Then the solution is $\mathtt{F} = \alpha_1\mathtt{F}_1 + \alpha_2\mathtt{F}_2 + \alpha_3\mathtt{F}_3$, where $\mathtt{F}_i$ denotes the matrices corresponding to the vectors spanning the null-space. The epipole satisfies $\mathtt{F}\mathbf{e} = \mathbf{0}$, so that $[(\mathtt{F}_1\mathbf{e}), (\mathtt{F}_2\mathbf{e}), (\mathtt{F}_3\mathbf{e})](\alpha_1, \alpha_2, \alpha_3)^\mathsf{T} = \mathbf{0}$. Since this equation has a solution it follows that $\det[(\mathtt{F}_1\mathbf{e}), (\mathtt{F}_2\mathbf{e}), (\mathtt{F}_3\mathbf{e})] = 0$ which is a cubic in $\mathbf{e}$.

(ii) Show that the image correspondence of four coplanar points and a quadric outline determines the fundamental matrix up to a two-fold ambiguity (Hint, see algorithm 13.2(p336)).

(iii) Show that the corresponding images of a (plane) conic are equivalent to two constraints on $\mathtt{F}$. See [Kahl-98b] for details.

(iv) Suppose that a stereo pair of images is acquired by a camera translating forward along its principal axis. Can the geometry of image rectification described in section 11.12 be applied in this case? See [Pollefeys-99a] for an alternative rectification geometry.

Samsung/Google, EX1010, p. 327

12

Structure Computation

This chapter describes how to compute the position of a point in 3-space given its image in two views and the camera matrices of those views. It is assumed that there are errors only in the measured image coordinates, not in the projection matrices $\mathtt{P}, \mathtt{P}'$.

Under these circumstances naïve triangulation by back-projecting rays from the measured image points will fail, because the rays will not intersect in general. It is thus necessary to *estimate* a best solution for the point in 3-space.

A best solution requires the definition and minimization of a suitable cost function. This problem is especially critical in affine and projective reconstruction in which there is no meaningful metric information about the object space. It is desirable to find a triangulation method that is invariant to projective transformations of space.

In the following sections we describe the estimation of $\mathbf{X}$ and of its covariance. An optimal (MLE) estimator for the point is developed, and it is shown that a solution can be obtained without requiring numerical minimization.

Note, this is the scenario where $\mathtt{F}$ is given *a priori* and then $\mathbf{X}$ is determined. An alternative scenario is where $\mathtt{F}$ and $\{\mathbf{X}_i\}$ are estimated simultaneously from the image point correspondences $\{\mathbf{x}_i \leftrightarrow \mathbf{x}'_i\}$, but this is not considered in this chapter. It may be solved using the Gold Standard algorithm of section 11.4.1, using the method considered in this chapter as an initial estimate.

12.1 Problem statement

It is supposed that the camera matrices, and hence the fundamental matrix, are provided; or that the fundamental matrix is provided, and hence a pair of consistent camera matrices can be constructed (as in section 9.5(p253)). In either case it is assumed that these matrices are known exactly, or at least with great accuracy compared with a pair of matching points in the two images.

Since there are errors in the *measured* points $\mathbf{x}$ and $\mathbf{x}'$, the rays back-projected from the points are skew. This means that there will *not* be a point $\mathbf{X}$ which exactly satisfies $\mathbf{x} = \mathtt{P}\mathbf{X}$, $\mathbf{x}' = \mathtt{P}'\mathbf{X}$; and that the image points do *not* satisfy the epipolar constraint $\mathbf{x}'^{\mathsf{T}}\mathtt{F}\mathbf{x} = 0$. These statements are equivalent since the two rays corresponding to a matching pair of points $\mathbf{x} \leftrightarrow \mathbf{x}'$ will meet in space if and only if the points satisfy the epipolar constraint. See figure 12.1.

Samsung/Google, EX1010, p. 328

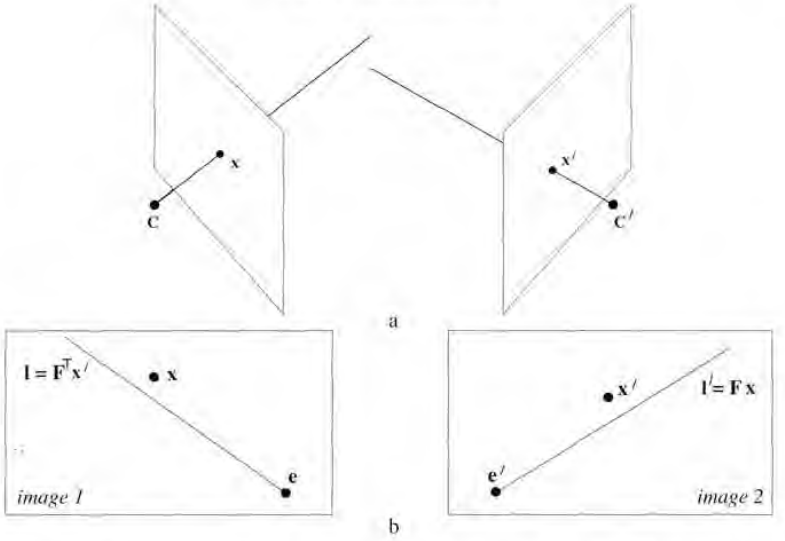


Fig. 12.1. *(a) Rays back-projected from imperfectly measured points* $\mathbf{x}, \mathbf{x}'$ *are skew in 3-space in general. (b) The epipolar geometry for* $\mathbf{x}, \mathbf{x}'$. *The measured points do not satisfy the epipolar constraint. The epipolar line* $\mathbf{l}' = \mathtt{F}\mathbf{x}$ *is the image of the ray through* $\mathbf{x}$, *and* $\mathbf{l} = \mathtt{F}^\top\mathbf{x}'$ *is the image of the ray through* $\mathbf{x}'$. *Since the rays do not intersect,* $\mathbf{x}'$ *does not lie on* $\mathbf{l}'$, *and* $\mathbf{x}$ *does not lie on* $\mathbf{l}$.

A desirable feature of the method of triangulation used is that it should be invariant under transformations of the appropriate class for the reconstruction – if the camera matrices are known only up to an affine (or projective) transformation, then it is clearly desirable to use an affine (resp. projective) invariant triangulation method to compute the 3D space points. Thus, denote by τ a triangulation method used to compute a 3D space point $\mathbf{X}$ from a point correspondence $\mathbf{x} \leftrightarrow \mathbf{x}'$ and a pair of camera matrices $\mathtt{P}$ and $\mathtt{P}'$. We write

$$\mathbf{X} = \tau(\mathbf{x}, \mathbf{x}', \mathtt{P}, \mathtt{P}').$$

The triangulation is said to be invariant under a transformation $\mathtt{H}$ if

$$\tau(\mathbf{x}, \mathbf{x}', \mathtt{P}, \mathtt{P}') = \mathtt{H}^{-1}\tau(\mathbf{x}, \mathbf{x}', \mathtt{P}\mathtt{H}^{-1}, \mathtt{P}'\mathtt{H}^{-1}).$$

This means that triangulation using the transformed cameras results in the transformed point.

It is clear, particularly for projective reconstruction, that it is inappropriate to minimize errors in the 3D projective space, $\mathbb{P}^3$. For instance, the method that finds the midpoint of the common perpendicular to the two rays in space is not suitable for projective reconstruction, since concepts such as distance and perpendicularity are not valid in the context of projective geometry. In fact, in projective reconstruction, this method will give different results depending on which particular projective reconstruction is considered – the method is not projective-invariant.

Here we will give a triangulation method that is projective-invariant. The key idea

Samsung/Google, EX1010, p. 329

is to estimate a 3D point $\widehat{\mathbf{X}}$ which exactly satisfies the supplied camera geometry, so it projects as

$$\hat{\mathbf{x}} = \mathtt{P}\widehat{\mathbf{X}} \quad \hat{\mathbf{x}}' = \mathtt{P}'\widehat{\mathbf{X}}$$

and the aim is to estimate $\widehat{\mathbf{X}}$ from the image measurements $\mathbf{x}$ and $\mathbf{x}'$. As described in section 12.3 the maximum likelihood estimate, under Gaussian noise, is given by the point $\widehat{\mathbf{X}}$ which minimizes the reprojection error – the (summed squared) distances between the projections of $\widehat{\mathbf{X}}$ and the measured image points.

Such a triangulation method is projective-invariant because only image distances are minimized, and the points $\hat{\mathbf{x}}$ and $\hat{\mathbf{x}}'$ which are the projections of $\widehat{\mathbf{X}}$ do not depend on the projective frame in which $\widehat{\mathbf{X}}$ is defined, i.e. a different projective reconstruction will project to the same points.

In the following sections simple linear triangulation methods are given. Then the MLE is defined, and it is shown that an optimal solution can be obtained via the root of a sixth-degree polynomial, thus avoiding a non-linear minimization of a cost function.

12.2 Linear triangulation methods

In this section, we describe simple linear triangulation methods. As usual the estimated point does not exactly satisfy the geometric relations, and is not an optimal estimate.

The linear triangulation method is the direct analogue of the DLT method described in section 4.1(p88). In each image we have a measurement $\mathbf{x} = \mathtt{P}\mathbf{X}$, $\mathbf{x}' = \mathtt{P}'\mathbf{X}$, and these equations can be combined into a form $\mathtt{A}\mathbf{X} = \mathbf{0}$, which is an equation linear in $\mathbf{X}$.

First the homogeneous scale factor is eliminated by a cross product to give three equations for each image point, of which two are linearly independent. For example for the first image, $\mathbf{x} \times (\mathtt{P}\mathbf{X}) = \mathbf{0}$ and writing this out gives

$$\begin{aligned} x(\mathbf{p}^{3\mathsf{T}}\mathbf{X}) - (\mathbf{p}^{1\mathsf{T}}\mathbf{X}) &= 0 \\ y(\mathbf{p}^{3\mathsf{T}}\mathbf{X}) - (\mathbf{p}^{2\mathsf{T}}\mathbf{X}) &= 0 \\ x(\mathbf{p}^{2\mathsf{T}}\mathbf{X}) - y(\mathbf{p}^{1\mathsf{T}}\mathbf{X}) &= 0 \end{aligned}$$

where $\mathbf{p}^{i\mathsf{T}}$ are the rows of $\mathtt{P}$. These equations are *linear* in the components of $\mathbf{X}$.

An equation of the form $\mathtt{A}\mathbf{X} = \mathbf{0}$ can then be composed, with

$$\mathtt{A} = \begin{bmatrix} x\mathbf{p}^{3\mathsf{T}} - \mathbf{p}^{1\mathsf{T}} \\ y\mathbf{p}^{3\mathsf{T}} - \mathbf{p}^{2\mathsf{T}} \\ x'\mathbf{p}'^{3\mathsf{T}} - \mathbf{p}'^{1\mathsf{T}} \\ y'\mathbf{p}'^{3\mathsf{T}} - \mathbf{p}'^{2\mathsf{T}} \end{bmatrix}$$

where two equations have been included from each image, giving a total of four equations in four homogeneous unknowns. This is a redundant set of equations, since the solution is determined only up to scale. Two ways of solving the set of equations of the form $\mathtt{A}\mathbf{X} = \mathbf{0}$ were discussed in section 4.1(p88) and will be considered again here.

Homogeneous method (DLT). The method of section 4.1.1(p90) finds the solution as the unit singular vector corresponding to the smallest singular value of $\mathtt{A}$, as shown

Samsung/Google, EX1010, p. 330

in section A5.3($p592$). The discussion in section 4.1.1 on the merits of normalization, and of including two or three equations from each image, applies equally well here.

Inhomogeneous method. In section 4.1.2($p90$) the solution of this system as a set of inhomogeneous equations is discussed. By setting $\mathbf{X} = (\mathtt{X}, \mathtt{Y}, \mathtt{Z}, 1)^\mathsf{T}$ the set of homogeneous equations, $\mathtt{A}\mathbf{X} = \mathbf{0}$, is reduced to a set of four inhomogeneous equations in three unknowns. The least-squares solution to these inhomogeneous equations is described in section A5.1($p588$). As explained in section 4.1.2, however, difficulties arise if the true solution $\mathbf{X}$ has last coordinate equal or close to 0. In this case, it is not legitimate to set it to 1 and instabilities can occur.

Discussion. These two methods are quite similar, but in fact have quite different properties in the presence of noise. The inhomogeneous method assumes that the solution point $\mathbf{X}$ is not at infinity, for otherwise we could not assume that $\mathbf{X} = (x, y, z, 1)^\mathsf{T}$. This is a disadvantage of this method when we are seeking to carry out a projective reconstruction, where reconstructed points may lie on the plane at infinity. Furthermore, neither of these two linear methods is quite suitable for projective reconstruction, since they are not projective-invariant. To see this, suppose that camera matrices $\mathtt{P}$ and $\mathtt{P}'$ are replaced by $\mathtt{P}\mathtt{H}^{-1}$ and $\mathtt{P}'\mathtt{H}^{-1}$. One sees that in this case the matrix of equations, $\mathtt{A}$, becomes $\mathtt{A}\mathtt{H}^{-1}$. A point $\mathbf{X}$ such that $\mathtt{A}\mathbf{X} = \boldsymbol{\epsilon}$ for the original problem corresponds to a point $\mathtt{H}\mathbf{X}$ satisfying $(\mathtt{A}\mathtt{H}^{-1})(\mathtt{H}\mathbf{X}) = \boldsymbol{\epsilon}$ for the transformed problem. Thus, there is a one-to-one correspondence between points $\mathbf{X}$ and $\mathtt{H}\mathbf{X}$ giving the same error. However, neither the condition $\|\mathbf{X}\| = 1$ for the homogeneous method, nor the condition $\mathbf{X} = (\mathtt{X}, \mathtt{Y}, \mathtt{Z}, 1)^\mathsf{T}$ for the inhomogeneous method, is invariant under application of the projective transformation $\mathtt{H}$. Thus, in general the point $\mathbf{X}$ solving the original problem will not correspond to a solution $\mathtt{H}\mathbf{X}$ for the transformed problem.

For affine transformations, on the other hand, the situation is different. In fact, although the condition $\|\mathbf{X}\| = 1$ is not preserved under affine transformations, the condition $\mathbf{X} = (\mathtt{X}, \mathtt{Y}, \mathtt{Z}, 1)^\mathsf{T}$ is preserved, since for an affine transformation, $\mathtt{H}(\mathtt{X}, \mathtt{Y}, \mathtt{Z}, 1)^\mathsf{T} = (\mathtt{X}', \mathtt{Y}', \mathtt{Z}', 1)^\mathsf{T}$. This means that there is a one-to-one correspondence between a vector $\mathbf{X} = (\mathtt{X}, \mathtt{Y}, \mathtt{Z}, 1)^\mathsf{T}$ such that $\mathtt{A}(x, y, z, 1)^\mathsf{T} = \boldsymbol{\epsilon}$ and the vector $\mathtt{H}\mathbf{X} = (\mathtt{X}', \mathtt{Y}', \mathtt{Z}', 1)^\mathsf{T}$ such that $(\mathtt{A}\mathtt{H}^{-1})(\mathtt{X}', \mathtt{Y}', \mathtt{Z}', 1)^\mathsf{T} = \boldsymbol{\epsilon}$. The error is the same for corresponding points. Thus, the points that minimize the error $\|\boldsymbol{\epsilon}\|$ correspond as well. Hence, the inhomogeneous method is affine-invariant, whereas the homogeneous method is not.

In the remainder of this chapter we will describe a method for triangulation that is invariant to the projective frame of the cameras, and minimizes a geometric image error. This will be the recommended triangulation method. Nevertheless, the homogeneous linear method described above often provides acceptable results. Furthermore, it has the virtue that it generalizes easily to triangulation when more than two views of the point are available.

12.3 Geometric error cost function

A typical observation consists of a noisy point correspondence $\mathbf{x} \leftrightarrow \mathbf{x}'$ which does not in general satisfy the epipolar constraint. In reality, the correct values of the cor-

Samsung/Google, EX1010, p. 331

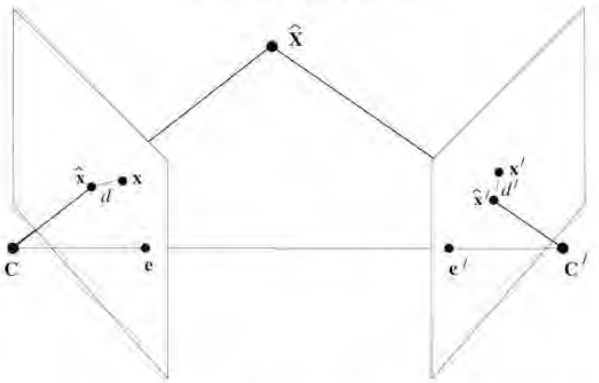


Fig. 12.2. **Minimization of geometric error.** *The estimated 3-space point $\widehat{\mathbf{X}}$ projects to the two images at $\hat{\mathbf{x}}, \hat{\mathbf{x}}'$. The corresponding image points $\hat{\mathbf{x}}, \hat{\mathbf{x}}'$ satisfy the epipolar constraint, unlike the measured points $\mathbf{x}$ and $\mathbf{x}'$. The point $\widehat{\mathbf{X}}$ is chosen so that the reprojection error $d^2 + d'^2$ is minimized.*

responding image points should be points $\hat{\mathbf{x}} \leftrightarrow \hat{\mathbf{x}}'$ lying close to the measured points $\mathbf{x} \leftrightarrow \mathbf{x}'$ and satisfying the epipolar constraint $\hat{\mathbf{x}}'^{\mathsf{T}}\mathbf{F}\hat{\mathbf{x}} = 0$ exactly.

We seek the points $\hat{\mathbf{x}}$ and $\hat{\mathbf{x}}'$ that minimize the function

$$\mathcal{C}(\mathbf{x}, \mathbf{x}') = d(\mathbf{x}, \hat{\mathbf{x}})^2 + d(\mathbf{x}', \hat{\mathbf{x}}')^2 \quad \text{subject to} \quad \hat{\mathbf{x}}'^{\mathsf{T}}\mathbf{F}\hat{\mathbf{x}} = 0 \tag{12.1}$$

where $d(*, *)$ is the Euclidean distance between the points. This is equivalent to minimizing the reprojection error for a point $\widehat{\mathbf{X}}$ which is mapped to $\hat{\mathbf{x}}$ and $\hat{\mathbf{x}}'$ by projection matrices consistent with $\mathbf{F}$, as illustrated in figure 12.2.

As explained in section 4.3(p102), assuming a Gaussian error distribution, the points $\hat{\mathbf{x}}'$ and $\hat{\mathbf{x}}$ are Maximum Likelihood Estimates (MLE) for the true image point correspondences. Once $\hat{\mathbf{x}}'$ and $\hat{\mathbf{x}}$ are found, the point $\widehat{\mathbf{X}}$ may be found by any triangulation method, since the corresponding rays will meet precisely in space.

This cost function could, of course, be minimized using a numerical minimization method such as Levenberg–Marquardt (section A6.2(p600)). A close approximation to the minimum may also be found using a first-order approximation to the geometric cost function, namely the Sampson error, as described in the next section. However, in section 12.5 it is shown that the minimum can be obtained non-iteratively by the solution of a sixth-degree polynomial.

12.4 Sampson approximation (first-order geometric correction)

Before deriving the exact polynomial solution we develop the Sampson approximation, which is valid when the measurement errors are small compared with the measurements. The Sampson approximation to the geometric *cost* function in the case of the fundamental matrix has already been discussed in section 11.4.3. Here we are concerned with computing the *correction* to the measured points.

The Sampson correction $\boldsymbol{\delta}_{\mathbf{X}}$ to the measured point $\mathbf{X} = (x, y, x', y')^{\mathsf{T}}$ (note, in this section $\mathbf{X}$ does not denote a homogeneous 3-space point) is shown in section 4.2.6(p98)

Samsung/Google, EX1010, p. 332



Fig. 12.3. *The projections $\hat{\mathbf{x}}$ and $\hat{\mathbf{x}}'$ of an estimated 3D point $\widehat{\mathbf{X}}$ lie on a pair of corresponding epipolar lines. The optimal $\hat{\mathbf{x}}$ and $\hat{\mathbf{x}}'$ will lie at the foot of the perpendiculars from the measured points $\mathbf{x}$ and $\mathbf{x}'$. Parametrizing the corresponding epipolar lines as a one-parameter family, the optimal estimation of $\widehat{\mathbf{X}}$ is reduced to a one-parameter search for corresponding epipolar lines so as to minimize the squared sum of perpendicular distances $d^2 + d'^2$.*

to be (4.11–p99)

$$\boldsymbol{\delta}_{\mathbf{X}} = -\mathtt{J}^{\mathsf{T}}(\mathtt{J}\mathtt{J}^{\mathsf{T}})^{-1}\boldsymbol{\epsilon}$$

and the corrected point is

$$\widehat{\mathbf{X}} = \mathbf{X} + \boldsymbol{\delta}_{\mathbf{X}} = \mathbf{X} - \mathtt{J}^{\mathsf{T}}(\mathtt{J}\mathtt{J}^{\mathsf{T}})^{-1}\boldsymbol{\epsilon}.$$

As shown in section 11.4.3 in the case of the variety defined by $\mathbf{x}'^{\mathsf{T}}\mathtt{F}\mathbf{x} = 0$, the error $\boldsymbol{\epsilon} = \mathbf{x}'^{\mathsf{T}}\mathtt{F}\mathbf{x}$, and the Jacobian is

$$J = \partial\boldsymbol{\epsilon}/\partial x = [(\mathtt{F}^{\mathsf{T}}\mathbf{x}')_1, (\mathtt{F}^{\mathsf{T}}\mathbf{x}')_2, (\mathtt{F}\mathbf{x})_1, (\mathtt{F}\mathbf{x})_2]$$

where for instance $(\mathtt{F}^{\mathsf{T}}\mathbf{x}')_1 = f_{11}x' + f_{21}y' + f_{31}$, etc. Then the first-order approximation to the corrected point is simply

$$\begin{pmatrix} \tilde{x} \\ \tilde{y} \\ \tilde{x}' \\ \tilde{y}' \end{pmatrix} = \begin{pmatrix} x \\ y \\ x' \\ y' \end{pmatrix} - \frac{\mathbf{x}'^{\mathsf{T}}\mathtt{F}\mathbf{x}}{(\mathtt{F}\mathbf{x})_1^2 + (\mathtt{F}\mathbf{x})_2^2 + (\mathtt{F}^{\mathsf{T}}\mathbf{x}')_1^2 + (\mathtt{F}^{\mathsf{T}}\mathbf{x}')_2^2} \begin{pmatrix} (\mathtt{F}^{\mathsf{T}}\mathbf{x}')_1 \\ (\mathtt{F}^{\mathsf{T}}\mathbf{x}')_2 \\ (\mathtt{F}\mathbf{x})_1 \\ (\mathtt{F}\mathbf{x})_2 \end{pmatrix}.$$

The approximation is accurate if the correction in each image is small (less than a pixel), and is cheap to compute. Note, however, that the corrected points will *not* satisfy the epipolar relation $\hat{\mathbf{x}}'^{\mathsf{T}}\mathtt{F}\hat{\mathbf{x}} = 0$ exactly. The method of the following section computes the points $\hat{\mathbf{x}}, \hat{\mathbf{x}}'$ which do exactly satisfy the epipolar constraint, but is more costly.

12.5 An optimal solution

In this section, we describe a method of triangulation that finds the global minimum of the cost function (12.1) using a non-iterative algorithm. If the Gaussian noise model can be assumed to be correct, this triangulation method is then provably optimal.

12.5.1 Reformulation of the minimization problem

Given a measured correspondence $\mathbf{x} \leftrightarrow \mathbf{x}'$, we seek a pair of points $\hat{\mathbf{x}}'$ and $\hat{\mathbf{x}}$ that minimize the sum of squared distances (12.1) subject to the epipolar constraint $\hat{\mathbf{x}}'^{\mathsf{T}}\mathtt{F}\hat{\mathbf{x}} = 0$.

The following discussion relates to figure 12.3. Any pair of points satisfying the

Samsung/Google, EX1010, p. 333

epipolar constraint must lie on a pair of corresponding epipolar lines in the two images. Thus, in particular, the optimum point $\hat{\mathbf{x}}$ lies on an epipolar line $\mathbf{l}$ and $\hat{\mathbf{x}}'$ lies on the corresponding epipolar line $\mathbf{l}'$. On the other hand, any other pair of points lying on the lines $\mathbf{l}$ and $\mathbf{l}'$ will also satisfy the epipolar constraint. This is true in particular for the point $\mathbf{x}_\perp$ on $\mathbf{l}$ lying closest to the measured point $\mathbf{x}$, and the correspondingly defined point $\mathbf{x}'_\perp$ on $\mathbf{l}'$. Of all pairs of points on the lines $\mathbf{l}$ and $\mathbf{l}'$, the points $\mathbf{x}_\perp$ and $\mathbf{x}'_\perp$ minimize the squared distance sum of (12.1). It follows that $\hat{\mathbf{x}}' = \mathbf{x}'_\perp$ and $\hat{\mathbf{x}} = \mathbf{x}_\perp$, where $\mathbf{x}_\perp$ and $\mathbf{x}'_\perp$ are defined with respect to a pair of matching epipolar lines $\mathbf{l}$ and $\mathbf{l}'$. Consequently, we may write $d(\mathbf{x}, \hat{\mathbf{x}}) = d(\mathbf{x}, \mathbf{l})$, where $d(\mathbf{x}, \mathbf{l})$ represents the perpendicular distance from the point $\mathbf{x}$ to the line $\mathbf{l}$. A similar expression holds for $d(\mathbf{x}', \hat{\mathbf{x}}')$.

In view of the previous paragraph, we may formulate the minimization problem differently as follows. We seek to minimize

$$d(\mathbf{x}, \mathbf{l})^2 + d(\mathbf{x}', \mathbf{l}')^2 \tag{12.2}$$

where $\mathbf{l}$ and $\mathbf{l}'$ range over all choices of corresponding epipolar lines. The point $\hat{\mathbf{x}}$ is then the closest point on the line $\mathbf{l}$ to the point $\mathbf{x}$ and the point $\hat{\mathbf{x}}'$ is similarly defined.

Our strategy for minimizing (12.2) is as follows:

(i) Parametrize the pencil of epipolar lines in the first image by a parameter t. Thus an epipolar line in the first image may be written as $\mathbf{l}(t)$.
(ii) Using the fundamental matrix $\mathbf{F}$, compute the corresponding epipolar line $\mathbf{l}'(t)$ in the second image.
(iii) Express the distance function $d(\mathbf{x}, \mathbf{l}(t))^2 + d(\mathbf{x}', \mathbf{l}'(t))^2$ explicitly as a function of t.
(iv) Find the value of t that minimizes this function.

In this way, the problem is reduced to that of finding the minimum of a function of a single variable t, i.e.

$$\min_{\hat{\mathbf{x}}} \mathcal{C} = d(\mathbf{x}, \hat{\mathbf{x}})^2 + d(\mathbf{x}', \hat{\mathbf{x}}')^2 = \min_t \mathcal{C} = d(\mathbf{x}, \mathbf{l}(t))^2 + d(\mathbf{x}', \mathbf{l}'(t))^2.$$

It will be seen that for a suitable parametrization of the pencil of epipolar lines the distance function is a rational polynomial function of t. Using techniques of elementary calculus, the minimization problem reduces to finding the real roots of a polynomial of degree 6.

12.5.2 Details of the minimization

If both of the image points correspond with the epipoles, then the point in space lies on the line joining the camera centres. In this case it is impossible to determine the position of the point in space. If only one of the corresponding points lies at an epipole, then we conclude that the point in space must coincide with the other camera centre. Consequently, we assume that neither of the two image points $\mathbf{x}$ and $\mathbf{x}'$ corresponds with an epipole.

In this case, we may simplify the analysis by applying a rigid transformation to each image in order to place both points $\mathbf{x}$ and $\mathbf{x}'$ at the origin, $(0, 0, 1)^\mathsf{T}$ in homogeneous

Samsung/Google, EX1010, p. 334

coordinates. Furthermore, the epipoles may be placed on the x-axis at points $(1, 0, f)^\mathsf{T}$ and $(1, 0, f')^\mathsf{T}$ respectively. A value f equal to 0 means that the epipole is at infinity. Applying these two rigid transforms has no effect on the sum-of-squares distance function in (12.1), and hence does not change the minimization problem.

Thus, in future we assume that in homogeneous coordinates, $\mathbf{x} = \mathbf{x}' = (0, 0, 1)^\mathsf{T}$ and that the two epipoles are at points $(1, 0, f)^\mathsf{T}$ and $(1, 0, f')^\mathsf{T}$. In this case, since $\mathtt{F}(1, 0, f)^\mathsf{T} = (1, 0, f')\mathtt{F} = \mathbf{0}$, the fundamental matrix has a special form

$$\mathtt{F} = \begin{pmatrix} ff'd & -f'c & -f'd \\ -fb & a & b \\ -fd & c & d \end{pmatrix}. \tag{12.3}$$

Consider an epipolar line in the first image passing through the point $(0, t, 1)^\mathsf{T}$ (still in homogeneous coordinates) and the epipole $(1, 0, f)^\mathsf{T}$. We denote this epipolar line by $\mathbf{l}(t)$. The vector representing this line is given by the cross product $(0, t, 1) \times (1, 0, f) = (tf, 1, -t)$, so the squared distance from the line to the origin is

$$d(\mathbf{x}, \mathbf{l}(t))^2 = \frac{t^2}{1 + (tf)^2}.$$

Using the fundamental matrix to find the corresponding epipolar line in the other image, we see that

$$\mathbf{l}'(t) = \mathtt{F}(0, t, 1)^\mathsf{T} = (-f'(ct + d), at + b, ct + d)^\mathsf{T}. \tag{12.4}$$

This is the representation of the line $\mathbf{l}'(t)$ as a homogeneous vector. The squared distance of this line from the origin is equal to

$$d(\mathbf{x}', \mathbf{l}'(t))^2 = \frac{(ct + d)^2}{(at + b)^2 + f'^2(ct + d)^2}.$$

The total squared distance is therefore given by

$$s(t) = \frac{t^2}{1 + f^2t^2} + \frac{(ct + d)^2}{(at + b)^2 + f'^2(ct + d)^2}. \tag{12.5}$$

Our task is to find the minimum of this function.

We may find the minimum using techniques of elementary calculus, as follows. We compute the derivative

$$s'(t) = \frac{2t}{(1 + f^2t^2)^2} - \frac{2(ad - bc)(at + b)(ct + d)}{((at + b)^2 + f'^2(ct + d)^2)^2}. \tag{12.6}$$

Maxima and minima of $s(t)$ will occur when $s'(t) = 0$. Collecting the two terms in $s'(t)$ over a common denominator and equating the numerator to 0 gives a condition

$$\begin{aligned} g(t) &= t((at + b)^2 + f'^2(ct + d)^2)^2 \\ &\quad -(ad - bc)(1 + f^2t^2)^2(at + b)(ct + d) \\ &= 0. \end{aligned} \tag{12.7}$$

The minima and maxima of $s(t)$ will occur at the roots of this polynomial. This is a

Samsung/Google, EX1010, p. 335

Objective

Given a measured point correspondence $\mathbf{x} \leftrightarrow \mathbf{x}'$, and a fundamental matrix $\mathtt{F}$, compute the corrected correspondences $\hat{\mathbf{x}} \leftrightarrow \hat{\mathbf{x}}'$ that minimize the geometric error (12.1) subject to the epipolar constraint $\hat{\mathbf{x}}'^{\mathsf{T}}\mathtt{F}\hat{\mathbf{x}} = 0$.

Algorithm

(i) Define transformation matrices

$$\mathtt{T} = \begin{bmatrix} 1 & & -x \\ & 1 & -y \\ & & 1 \end{bmatrix} \text{ and } \mathtt{T}' = \begin{bmatrix} 1 & & -x' \\ & 1 & -y' \\ & & 1 \end{bmatrix}.$$

These are the translations that take $\mathbf{x} = (x, y, 1)^{\mathsf{T}}$ and $\mathbf{x}' = (x', y', 1)^{\mathsf{T}}$ to the origin.

(ii) Replace $\mathtt{F}$ by $\mathtt{T}'^{-\mathsf{T}}\mathtt{F}\mathtt{T}^{-1}$. The new $\mathtt{F}$ corresponds to translated coordinates.

(iii) Compute the right and left epipoles $\mathbf{e} = (e_1, e_2, e_3)^{\mathsf{T}}$ and $\mathbf{e}' = (e'_1, e'_2, e'_3)^{\mathsf{T}}$ such that $\mathbf{e}'^{\mathsf{T}}\mathtt{F} = \mathbf{0}$ and $\mathtt{F}\mathbf{e} = \mathbf{0}$. Normalize (multiply by a scale) $\mathbf{e}$ such that $e_1^2 + e_2^2 = 1$ and do the same to $\mathbf{e}'$.

(iv) Form matrices

$$\mathtt{R} = \begin{bmatrix} e_1 & e_2 & \\ -e_2 & e_1 & \\ & & 1 \end{bmatrix} \text{ and } \mathtt{R}' = \begin{bmatrix} e'_1 & e'_2 & \\ -e'_2 & e'_1 & \\ & & 1 \end{bmatrix}$$

and observe that $\mathtt{R}$ and $\mathtt{R}'$ are rotation matrices, and $\mathtt{R}\mathbf{e} = (1, 0, e_3)^{\mathsf{T}}$ and $\mathtt{R}'\mathbf{e}' = (1, 0, e'_3)^{\mathsf{T}}$.

(v) Replace $\mathtt{F}$ by $\mathtt{R}'\mathtt{F}\mathtt{R}^{\mathsf{T}}$. The resulting $\mathtt{F}$ must have the form (12.3).

(vi) Set $f = e_3$, $f' = e'_3$, $a = F_{22}$, $b = F_{23}$, $c = F_{32}$ and $d = F_{33}$.

(vii) Form the polynomial $g(t)$ as a polynomial in t according to (12.7). Solve for t to get 6 roots.

(viii) Evaluate the cost function (12.5) at the real part of each of the roots of $g(t)$ (alternatively evaluate at only the real roots of $g(t)$). Also, find the asymptotic value of (12.1) for $t = \infty$, namely $1/f^2 + c^2/(a^2 + f'^2c^2)$. Select the value t_{min} of t that gives the smallest value of the cost function.

(ix) Evaluate the two lines $\mathbf{l} = (tf, 1, -t)$ and $\mathbf{l}'$ given by (12.4) at t_{min} and find $\hat{\mathbf{x}}$ and $\hat{\mathbf{x}}'$ as the closest points on these lines to the origin. For a general line (λ, μ, ν), the formula for the closest point on the line to the origin is $(-\lambda\nu, -\mu\nu, \lambda^2 + \mu^2)$.

(x) Transfer back to the original coordinates by replacing $\hat{\mathbf{x}}$ by $\mathtt{T}^{-1}\mathtt{R}^{\mathsf{T}}\hat{\mathbf{x}}$ and $\hat{\mathbf{x}}'$ by $\mathtt{T}'^{-1}\mathtt{R}'^{\mathsf{T}}\hat{\mathbf{x}}'$.

(xi) The 3-space point $\widehat{\mathbf{X}}$ may then be obtained by the homogeneous method of section 12.2.

Algorithm 12.1. *The optimal triangulation method.*

polynomial of degree 6, which may have up to 6 real roots, corresponding to 3 minima and 3 maxima of the function $s(t)$. The absolute minimum of the function $s(t)$ may be found by finding the roots of $g(t)$ and evaluating the function $s(t)$ given by (12.5) at each of the real roots. More simply, one checks the value of $s(t)$ at the real part of each root (complex or real) of $g(t)$, which saves the trouble of determining if a root is real or complex. One should also check the asymptotic value of $s(t)$ as $t \to \infty$ to see if the minimum distance occurs when $t = \infty$, corresponding to an epipolar line $fx = 1$ in the first image.

The overall method is summarized in algorithm 12.1.

Samsung/Google, EX1010, p. 336



a b

Fig. 12.4. *(a) Example of a cost function with three minima. (b) This is the cost function for a perfect point match, which nevertheless has two minima.*

12.5.3 Local minima

The fact that $g(t)$ in (12.7) has degree 6 means that $s(t)$ may have as many as three minima. In fact, this is indeed possible, as the following case shows. Setting $f = f' = 1$ and

$$\mathtt{F} = \begin{pmatrix} 4 & -3 & -4 \\ -3 & 2 & 3 \\ -4 & 3 & 4 \end{pmatrix}$$

gives a function

$$s(t) = \frac{t^2}{1+t^2} + \frac{(3t+4)^2}{(2t+3)^2 + (3t+4)^2}$$

with graph as shown in figure 12.4a[1]. The three minima are clearly shown.

As a second example, we consider the case where $f = f' = 1$, and

$$\mathtt{F} = \begin{pmatrix} 0 & -1 & 0 \\ 1 & 2 & -1 \\ 0 & 1 & 0 \end{pmatrix}.$$

In this case, the function $s(t)$ is given by

$$s(t) = \frac{t^2}{t^2+1} + \frac{t^2}{t^2 + (2t-1)^2}$$

and both terms of the cost function vanish for a value of $t = 0$, which means that the corresponding points $\mathbf{x}$ and $\mathbf{x}'$ exactly satisfy the epipolar constraint. This can be verified by observing that $\mathbf{x}'^{\mathsf{T}}\mathtt{F}\mathbf{x} = 0$. Thus the two points are exactly matched. A graph of the cost function $s(t)$ is shown in figure 12.4b. Apart from the absolute minimum at $t = 0$ there is also a local minimum at $t = 1$. Thus, even in the case of perfect matches local minima may occur. This example shows that an algorithm that attempts to minimize the cost function in (12.1), or equivalently (12.2), by an iterative

[1] In this graph and also figure 12.4b we make the substitution $t = \tan(\theta)$ and plot for θ in the range $-\pi/2 \leq \theta \leq \pi/2$, so as to show the whole infinite range for t.

Samsung/Google, EX1010, p. 337

search beginning from an arbitrary initial point is in danger of finding a local minimum, even in the case of perfect point matches.

12.5.4 Evaluation on real images

An experiment was carried out using the calibration cube images shown in figure 11.2-(*p*289) with the goal of determining how the triangulation method effects the accuracy of reconstruction. A Euclidean model of the cubes, to be used as ground truth, was estimated and refined using accurate image measurements. The measured pixel locations were corrected to correspond exactly to the Euclidean model, requiring coordinate corrections averaging 0.02 pixels.

At this stage we had a model and a set of matched points corresponding exactly to the model. Next, a projective reconstruction of the points was computed and a projective transformation H computed that brought the projective reconstruction into agreement with the Euclidean model. Controlled zero-mean Gaussian noise was introduced into the point coordinates, and triangulation using two methods was carried out in the projective frame, the transformation H applied, and the error of each method was measured in the Euclidean frame. Figure 12.5 shows the results of this experiment for the two triangulation methods. The graph shows the average reconstruction error over all points in 10 separate runs at each chosen noise level. It clearly shows that the optimal method gives superior reconstruction results.

In this pair of images the two epipoles are distant from the image. For cases where the epipoles are close to the images, results on synthetic images show that the advantage of the polynomial method will be more pronounced.

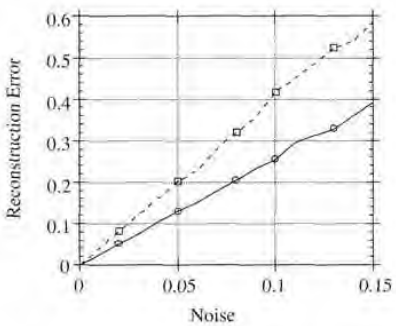


Fig. 12.5. **Reconstruction error comparison of triangulation methods.** *The graph shows the reconstruction error obtained using two triangulation methods: (i) selection of the midpoint of the common perpendicular to the rays in the projective frame (top curve), and (ii) the optimal polynomial method (lower curve). On the horizontal axis is the noise, on the vertical axis the reconstruction error. The units for reconstruction error are relative to a unit distance equal to the side of one of the dark squares in the calibration cube image figure 11.2*(p289)*. Even for the best method the error is large for higher noise levels, because there is little movement between the images.*

Samsung/Google, EX1010, p. 338

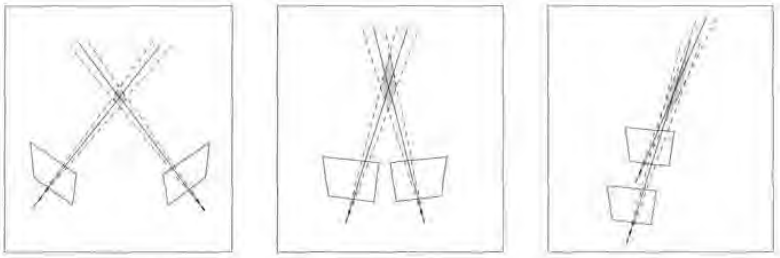


Fig. 12.6. **Uncertainty of reconstruction.** *The shaded region in each case illustrates the shape of the uncertainty region, which depends on the angle between the rays. Points are less precisely localized along the ray as the rays become more parallel. Forward motion in particular can give poor reconstructions since rays are almost parallel for much of the field of view.*

12.6 Probability distribution of the estimated 3D point

An illustration of the distribution of the reconstructed point is given in figure 12.6. A good rule of thumb is that the angle between the rays determines the accuracy of reconstruction. This is a better guide than simply considering the baseline, which is the more commonly used measure.

More formally the probability of a particular 3D point $\mathbf{X}$ depends on the probability of obtaining its image in each view. We will consider a simplified example where the objective is to estimate the probability that a point on a plane has position $\mathbf{X} = (X, Y)^\mathsf{T}$ given its images $x = f(\mathbf{X})$ and $x' = f'(\mathbf{X})$ in two line cameras. (The projections f and f' are expressible in terms of 2×3 projection matrices $\mathtt{P}_{2\times3}$ and $\mathtt{P}'_{2\times3}$ respectively – see section 6.4.2(*p*175)). The imaging geometry is shown in figure 12.7(a).

Suppose that the measured image point is at x in the first image, and the measurement process is corrupted by Gaussian noise with mean zero and variance σ^2, then the probability of obtaining x, given that the true image point is $f(\mathbf{X})$, is given by

$$p(x|\mathbf{X}) = (2\pi\sigma^2)^{-1/2} \exp\left(-|f(\mathbf{X}) - x|^2/(2\sigma^2)\right).$$

with a similar expression for $p(x'|\mathbf{X})$. We wish to compute the *a posteriori* distribution:

$$p(\mathbf{X}|x, x') = p(x, x'|\mathbf{X})p(\mathbf{X})/p(x, x').$$

Assuming a uniform prior probability $p(\mathbf{X})$, and independent image measurements in the two images, it follows that

$$p(\mathbf{X}|x, x') \sim p(x, x'|\mathbf{X}) = p(x|\mathbf{X})p(x'|\mathbf{X}).$$

Figure 12.7 shows an example of this Probability Density Function (PDF). The bias and variance of this example is discussed in appendix 3(*p*568).

12.7 Line reconstruction

Suppose a line in 3-space is projected to lines in two views as $\mathbf{l}$ and $\mathbf{l}'$. The line in 3-space can be reconstructed by back-projecting each line to give a plane in 3-space, and intersecting the planes.

Samsung/Google, EX1010, p. 339

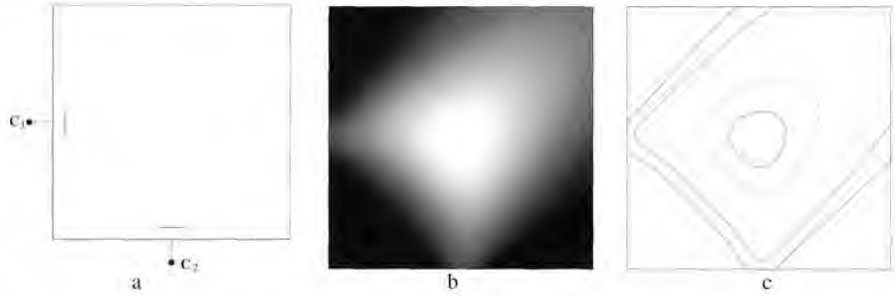


Fig. 12.7. PDF **for a triangulated point.** *(a) The camera configuration. There are two line cameras with centres at* $\mathbf{C}_1$ *and* $\mathbf{C}_2$*. The image lines are the left and lower edge of the square. The bar indicates the* 2σ *range of the noise. The plots show the* PDF *for a triangulated point computed from the two perspective projections. A large noise variance* σ^2 *is chosen to emphasize the effect. (b) The* PDF *shown as an image with white representing a higher value. (b) A contour plot of the* PDF*. Note that the* PDF *is not a Gaussian.*

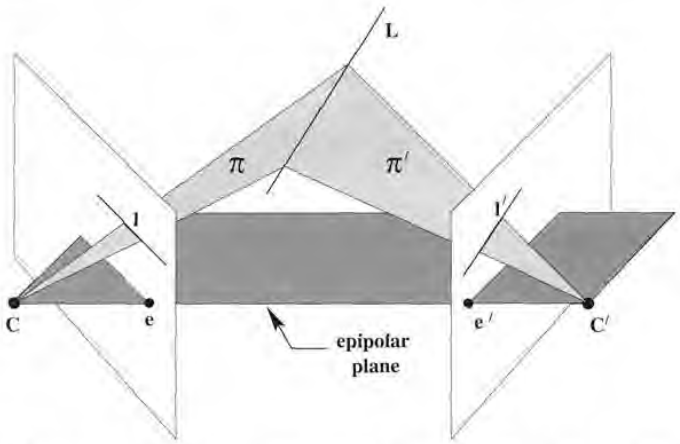


Fig. 12.8. **Line reconstruction.** *The image lines* l, l' *back-project to planes* π, π' *respectively. The plane intersection determines the line* $\mathbf{L}$ *in 3-space. If the line in 3-space lies on an epipolar plane then its position in 3-space cannot be determined from its images. In this case the epipoles lie on the image lines.*

The planes defined by the lines are $\pi = \mathrm{P}^\mathsf{T}\mathrm{l}$ and $\pi' = \mathrm{P}'^\mathsf{T}\mathrm{l}'$. It is often quite convenient in practice to parametrize the line in 3-space by the two planes defined by the image lines, i.e. to represent the line as the 2×4 matrix

$$\mathrm{L} = \begin{bmatrix} \mathrm{l}^\mathsf{T}\mathrm{P} \\ \mathrm{l}'^\mathsf{T}\mathrm{P}' \end{bmatrix}$$

as described in the span representation of section 3.2.2(p68). Then for example a point $\mathbf{x}$ lies on the line if $\mathrm{L}\mathbf{x} = \mathbf{0}$.

In the case of corresponding points the pre-image (i.e. the point in 3-space that projects to the image points) is over-determined since there are four measurements

Samsung/Google, EX1010, p. 340

on the three degrees of freedom of the 3-space point. In contrast in the case of lines the pre-image is exactly determined because a line in 3-space has four degrees of freedom, and the image line provides two measurements in each view. Note, here we are considering the lines as infinite, and not using their endpoints.

Degeneracy. As illustrated in figure 12.8 lines in 3-space lying on epipolar planes cannot be determined from their images in two views. Such lines intersect the camera baseline. In practice, when there is measurement error, lines which are close to intersecting the baseline can be poorly localized in a reconstruction.

The degeneracy for lines is far more severe than for points: in the case of points there is a one-parameter family of points on the baseline which cannot be recovered. For lines there is a three-parameter family: one parameter for position on the baseline, and the other two for the star of lines through each point on the baseline.

Intersection of more than two planes

In later chapters (particularly chapter 15) we will be considering reconstruction from three or more views. To reconstruct the line that results from the intersection of several planes it is appropriate to proceed as follows. Represent each plane π_i by a 4-vector and form an $n \times 4$ matrix $\mathtt{A}$ for n planes with rows $\pi_i^\top$. Let $\mathtt{A} = \mathtt{UDV}^\top$ be the singular value decomposition. The two columns of $\mathtt{V}$ corresponding to the two largest singular values span the best rank 2 approximation to $\mathtt{A}$ and may be used to define the line of intersection of the planes. If the planes are defined by back-projecting image lines, then the Maximum Likelihood estimate of the line $\mathtt{L}$ in 3-space is found by minimizing a geometric image distance between $\mathtt{L}$ projected into each image and the measured line in that image. This is discussed in section 16.4.1(*p*396).

12.8 Closure

It is not evident how to extend the polynomial method of triangulation to 3 or more views. However, the linear method extends in an obvious manner. More interestingly, the Sampson method also may be extended to 3 or more views, as is described in [Torr-97]. The disadvantage is that the computational cost (and also coding effort) increases noticeably with more views.

12.8.1 The literature

The optimal triangulation method was given by Hartley & Sturm [Hartley-97b].

12.8.2 Notes and exercises

(i) Derive a method for triangulation in the case of pure translational motion of the cameras. Hint, see figure 12.9. A closed form solution for the parameter θ is possible. This method was used in [Armstrong-94].

(ii) Adapt the polynomial triangulation method to a pair of affine cameras (or more generally, to cameras with the same principal plane). In this case, the fundamental matrix has a simple form, (14.1–*p*345), and the method reduces to a linear algorithm.

Samsung/Google, EX1010, p. 341

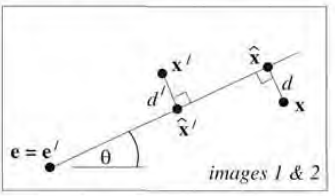


Fig. 12.9. **The epipolar geometry for pure translation.** *In this case, corresponding epipolar lines are identical (see section 11.7.1(p293)). The epipolar line (parametrized by θ) that minimizes $d^2 + d'^2$ can be computed directly.*

(iii) Show that the Sampson method (section 12.4) is invariant under Euclidean coordinate changes in the images (and the corresponding change in F).

(iv) Derive the analogue of the polynomial solution for triangulation in the case of a planar homography, i.e. given a measured correspondence $\mathbf{x} \leftrightarrow \mathbf{x}'$, compute points $\hat{\mathbf{x}}$ and $\hat{\mathbf{x}}'$ that minimize the function

$$\mathcal{C}(\mathbf{x}, \mathbf{x}') = d(\mathbf{x}, \hat{\mathbf{x}})^2 + d(\mathbf{x}', \hat{\mathbf{x}}')^2 \quad \text{subject to } \hat{\mathbf{x}}' = \mathtt{H}\hat{\mathbf{x}}.$$

See [Sturm-97b], where it is shown that the solution is a degree 8 polynomial in one variable.

Samsung/Google, EX1010, p. 342

13

Scene planes and homographies

This chapter describes the projective geometry of two cameras and a world plane.

Images of points on a plane are related to corresponding image points in a second view by a (planar) homography as shown in figure 13.1. This is a projective relation since it depends only on the intersections of planes with lines. It is said that the plane *induces* a homography between the views. The homography map *transfers* points from one view to the other as if they were images of points on the plane.

There are then two relations between the two views: first, through the epipolar geometry a point in one view determines a line in the other which is the image of the ray through that point; and second, through the homography a point in one view determines a point in the other which is the image of the intersection of the ray with a plane. This chapter ties together these two relations of 2-view geometry.

Two other important notions are described here: the parallax with respect to a plane, and the infinite homography.

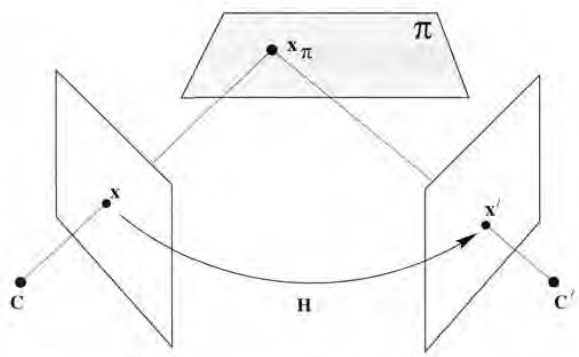


Fig. 13.1. **The homography induced by a plane.** *The ray corresponding to a point* $\mathbf{x}$ *is extended to meet the plane* π *in a point* $\mathbf{x}_\pi$*; this point is projected to a point* $\mathbf{x}'$ *in the other image. The map from* $\mathbf{x}$ *to* $\mathbf{x}'$ *is the homography induced by the plane* π*. There is a perspectivity,* $\mathbf{x} = \mathtt{H}_{1\pi}\mathbf{x}_\pi$*, between the world plane* π *and the first image plane; and a perspectivity,* $\mathbf{x}' = \mathtt{H}_{2\pi}\mathbf{x}_\pi$*, between the world plane and second image plane. The composition of the two perspectivities is a homography,* $\mathbf{x}' = \mathtt{H}_{2\pi}\mathtt{H}_{1\pi}^{-1}\mathbf{x} = \mathtt{H}\mathbf{x}$*, between the image planes.*

Samsung/Google, EX1010, p. 343

13.1 Homographies given the plane and vice versa

We start by showing that for planes in general position the homography is determined uniquely by the plane and vice versa. General position in this case means that the plane does not contain either of the camera centres. If the plane does contain one of the camera centres then the induced homography is degenerate.

Suppose a plane π in 3-space is specified by its coordinates in the world frame. We first derive an explicit expression for the induced homography.

Result 13.1. *Given the projection matrices for the two views*

$$\mathtt{P} = [\mathtt{I} \mid \mathbf{0}] \qquad \mathtt{P}' = [\mathtt{A} \mid \mathbf{a}]$$

and a plane defined by $\boldsymbol{\pi}^\mathsf{T}\mathbf{X} = 0$ *with* $\boldsymbol{\pi} = (\mathbf{v}^\mathsf{T}, 1)^\mathsf{T}$, *then the homography induced by the plane is* $\mathbf{x}' = \mathtt{H}\mathbf{x}$ *with*

$$\mathtt{H} = \mathtt{A} - \mathbf{a}\mathbf{v}^\mathsf{T}. \tag{13.1}$$

We may assume that $\pi_4 = 1$ since the plane does not pass through the centre of the first camera at $(0, 0, 0, 1)^\mathsf{T}$.

Note, there is a three-parameter family of planes in 3-space, and correspondingly a three-parameter family of homographies between two views induced by planes in 3-space. These three parameters are specified by the elements of the vector $\mathbf{v}$, which is *not* a homogeneous 3-vector.

Proof. To compute $\mathtt{H}$ we back-project a point $\mathbf{x}$ in the first view and determine the intersection point $\mathbf{X}$ of this ray with the plane $\boldsymbol{\pi}$. The 3D point $\mathbf{X}$ is then projected into the second view.

For the first view $\mathbf{x} = \mathtt{P}\mathbf{X} = [\mathtt{I} \mid \mathbf{0}]\mathbf{X}$ and so any point on the ray $\mathbf{X} = (\mathbf{x}^\mathsf{T}, \rho)^\mathsf{T}$ projects to $\mathbf{x}$, where ρ parametrizes the point on the ray. Since the 3D point $\mathbf{X}$ is on $\boldsymbol{\pi}$ it satisfies $\boldsymbol{\pi}^\mathsf{T}\mathbf{X} = 0$. This determines ρ, and $\mathbf{X} = (\mathbf{x}^\mathsf{T}, -\mathbf{v}^\mathsf{T}\mathbf{x})^\mathsf{T}$. The 3D point $\mathbf{X}$ projects into the second view as

$$\begin{aligned} \mathbf{x}' &= \mathtt{P}'\mathbf{X} = [\mathtt{A} \mid \mathbf{a}]\mathbf{X} \\ &= \mathtt{A}\mathbf{x} - \mathbf{a}\mathbf{v}^\mathsf{T}\mathbf{x} = \left(\mathtt{A} - \mathbf{a}\mathbf{v}^\mathsf{T}\right)\mathbf{x} \end{aligned}$$

as required. □

Example 13.2. A calibrated stereo rig.

Suppose the camera matrices are those of a calibrated stereo rig with the world origin at the first camera

$$\mathtt{P}_\mathrm{E} = \mathtt{K}[\mathtt{I} \mid \mathbf{0}] \qquad \mathtt{P}'_\mathrm{E} = \mathtt{K}'[\mathtt{R} \mid \mathbf{t}].$$

and the world plane $\boldsymbol{\pi}_\mathrm{E}$ has coordinates $\boldsymbol{\pi}_\mathrm{E} = (\mathbf{n}^\mathsf{T}, d)^\mathsf{T}$ so that for points on the plane $\mathbf{n}^\mathsf{T}\widetilde{\mathbf{X}} + d = 0$. We wish to compute an expression for the homography induced by the plane.

Samsung/Google, EX1010, p. 344

From result 13.1, with $\mathbf{v} = \mathbf{n}/d$, the homography for the cameras $\mathtt{P} = [\mathtt{I} \mid \mathbf{0}]$, $\mathtt{P}' = [\mathtt{R} \mid \mathbf{t}]$ is

$$\mathtt{H} = \mathtt{R} - \mathbf{t}\mathbf{n}^\mathsf{T}/d.$$

Applying the transformations $\mathtt{K}$ and $\mathtt{K}'$ to the images we obtain the cameras $\mathtt{P}_\mathrm{E} = \mathtt{K}[\mathtt{I} \mid \mathbf{0}]$, $\mathtt{P}'_\mathrm{E} = \mathtt{K}'[\mathtt{R} \mid \mathbf{t}]$ and the resulting induced homography is

$$\mathtt{H} = \mathtt{K}'\left(\mathtt{R} - \mathbf{t}\mathbf{n}^\mathsf{T}/d\right)\mathtt{K}^{-1}. \tag{13.2}$$

This is a three-parameter family of homographies, parametrized by $\mathbf{n}/d$. It is defined by the plane, and the camera internal and relative external parameters. △

13.1.1 Homographies compatible with epipolar geometry

Suppose four points $\mathbf{X}_i$ are chosen on a scene plane. Then the correspondence $\mathbf{x}_i \leftrightarrow \mathbf{x}'_i$ of their images between two views defines a homography $\mathtt{H}$, which is the homography induced by the plane. These image correspondences also obey the epipolar constraint, i.e. $\mathbf{x}_i'^\mathsf{T}\mathtt{F}\mathbf{x}_i = 0$, since they arise from images of scene points. Indeed, the correspondence $\mathbf{x} \leftrightarrow \mathbf{x}' = \mathtt{H}\mathbf{x}$ obeys the epipolar constraint for *any* $\mathbf{x}$, since again $\mathbf{x}$ and $\mathbf{x}'$ are images of a scene point, in this case the point given by intersecting the scene plane with the ray back-projected from $\mathbf{x}$. The homography $\mathtt{H}$ is said to be consistent or *compatible* with $\mathtt{F}$.

Now suppose four *arbitrary* image points are chosen in the first view and four arbitrary image points chosen in the second. Then a homography $\tilde{\mathtt{H}}$ may be computed which maps one set of points into the other (provided no three are collinear in either view). However, correspondences $\mathbf{x} \leftrightarrow \mathbf{x}' = \tilde{\mathtt{H}}\mathbf{x}$ may *not* obey the epipolar constraint. If the correspondence $\mathbf{x} \leftrightarrow \mathbf{x}' = \tilde{\mathtt{H}}\mathbf{x}$ does not obey the epipolar constraint then there does not exist a scene plane which induces $\tilde{\mathtt{H}}$.

The epipolar geometry determines the projective geometry between two views, and can be used to define conditions on homographies which are induced by actual scene planes. Figure 13.2 illustrates several relations between epipolar geometry and scene planes which can be used to define such conditions. For example, since correspondences $\mathbf{x} \leftrightarrow \mathtt{H}\mathbf{x}$ obey the epipolar constraint if $\mathtt{H}$ is induced by a plane, then from $\mathbf{x}'^\mathsf{T}\mathtt{F}\mathbf{x} = 0$

$$(\mathtt{H}\mathbf{x})^\mathsf{T}\mathtt{F}\mathbf{x} = \mathbf{x}^\mathsf{T}\mathtt{H}^\mathsf{T}\mathtt{F}\mathbf{x} = 0,$$

This is true for all $\mathbf{x}$, so:

- *A homography* $\mathtt{H}$ *is compatible with a fundamental matrix* $\mathtt{F}$ *if and only if the matrix* $\mathtt{H}^\mathsf{T}\mathtt{F}$ *is skew-symmetric:*

$$\mathtt{H}^\mathsf{T}\mathtt{F} + \mathtt{F}^\mathsf{T}\mathtt{H} = \mathtt{0} \tag{13.3}$$

The argument above showed that the condition was necessary. The fact that this is a sufficient condition was shown by Luong and Viéville [Luong-96]. Counting degrees of freedom, (13.3) places six homogeneous (five inhomogeneous) constraints on the 8 degrees of freedom of $\mathtt{H}$. There are therefore $8 - 5 = 3$ degrees of freedom remaining

Samsung/Google, EX1010, p. 345

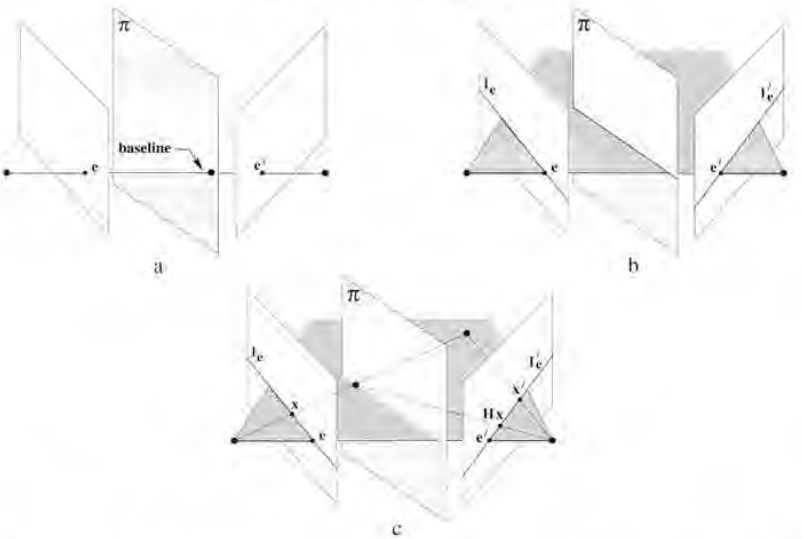


Fig. 13.2. **Compatibility constraints.** *The homography induced by a plane is coupled to the epipolar geometry and satisfies constraints. (a) The epipole is mapped by the homography, as* $\mathbf{e}' = \mathtt{H}\mathbf{e}$*, since the epipoles are images of the point on the plane where the baseline intersects* π*. (b) Epipolar lines are mapped by the homography as* $\mathtt{H}^{\mathsf{T}}\mathbf{l}'_e = \mathbf{l}_e$*. (c) Any point* $\mathbf{x}$ *mapped by the homography lies on its corresponding epipolar line* $\mathbf{l}'_e$*, so* $\mathbf{l}'_e = \mathtt{F}\mathbf{x} = \mathbf{x}' \times (\mathtt{H}\mathbf{x})$*.*

for $\mathtt{H}$; these 3 degrees of freedom correspond to the three-parameter family of planes in 3-space.

The compatibility constraint (13.3) is an implicit equation in $\mathtt{H}$ and $\mathtt{F}$. We now develop an explicit expression for a homography $\mathtt{H}$ induced by a plane given $\mathtt{F}$ which is more suitable for a computational algorithm.

Result 13.3. *Given the fundamental matrix* $\mathtt{F}$ *between two views, the three-parameter family of homographies induced by a world plane is*

$$\mathtt{H} = \mathtt{A} - \mathbf{e}'\mathbf{v}^{\mathsf{T}} \tag{13.4}$$

where $[\mathbf{e}']_\times \mathtt{A} = \mathtt{F}$ *is any decomposition of the fundamental matrix.*

Proof. Result 13.1 has shown that given the camera matrices for the view pair $\mathtt{P} = [\mathtt{I} \mid \mathbf{0}]$, $\mathtt{P}' = [\mathtt{A} \mid \mathbf{a}]$ a plane π induces a homography $\mathtt{H} = \mathtt{A} - \mathbf{a}\mathbf{v}^{\mathsf{T}}$ where $\pi = (\mathbf{v}^{\mathsf{T}}, 1)^{\mathsf{T}}$. However, according to result 9.9(p254), for the fundamental matrix $\mathtt{F} = [\mathbf{e}']_\times \mathtt{A}$ one can choose the two cameras to be $[\mathtt{I} \mid \mathbf{0}]$ and $[\mathtt{A} \mid \mathbf{e}']$. □

Remark. The above derivation, which is based on the projection of points on a plane, ensures that the homographies are compatible with the epipolar geometry. Algebraically, the homography (13.4) is compatible with the fundamental matrix since it obeys the necessary and sufficient condition (13.3) that $\mathtt{F}^{\mathsf{T}}\mathtt{H}$ is skew-symmetric. This

Samsung/Google, EX1010, p. 346