

The VR Book

*Human-Centered Design
for Virtual Reality*



Jason Jerald, Ph.D.



The VR Book

ACM Books

Editor in Chief

M. Tamer Özsu, *University of Waterloo*

ACM Books is a new series of high-quality books for the computer science community, published by ACM in collaboration with Morgan & Claypool Publishers. ACM Books publications are widely distributed in both print and digital formats through booksellers and to libraries (and library consortia) and individual ACM members via the ACM Digital Library platform.

The VR Book: Human-Centered Design for Virtual Reality

Jason Jerald, *NextGen Interactions*

2016

Ada's Legacy

Robin Hammerman, *Stevens Institute of Technology*; Andrew L. Russell, *Stevens Institute of Technology*

2016

Edmund Berkeley and the Social Responsibility of Computer Professionals

Bernadette Longo, *New Jersey Institute of Technology*

2015

Candidate Multilinear Maps

Sanjam Garg, *University of California, Berkeley*

2015

Smarter than Their Machines: Oral Histories of Pioneers in Interactive Computing

John Cullinane, *Northeastern University*; Mossavar-Rahmani Center for Business and Government, John F. Kennedy School of Government, Harvard University

2015

A Framework for Scientific Discovery through Video Games

Seth Cooper, *University of Washington*

2014

Trust Extension as a Mechanism for Secure Code Execution on Commodity Computers

Bryan Jeffrey Parno, *Microsoft Research*

2014

Embracing Interference in Wireless Systems

Shyamnath Gollakota, *University of Washington*

2014

The VR Book

Human-Centered Design for Virtual Reality

Jason Jerald

NextGen Interactions

ACM Books #8



Copyright © 2016 by the Association for Computing Machinery
and Morgan & Claypool Publishers

All rights reserved. No part of this publication may be reproduced, stored in a retrieval system, or transmitted in any form or by any means—electronic, mechanical, photocopy, recording, or any other except for brief quotations in printed reviews—without the prior permission of the publisher.

This book is presented solely for educational and entertainment purposes. The author and publisher are not offering it as legal, accounting, or other professional services advice. While best efforts have been used in preparing this book, the author and publisher make no representations or warranties of any kind and assume no liabilities of any kind with respect to the accuracy or completeness of the contents and specifically disclaim any implied warranties of merchantability or fitness of use for a particular purpose. Neither the author nor the publisher shall be held liable or responsible to any person or entity with respect to any loss or incidental or consequential damages caused, or alleged to have been caused, directly or indirectly, by the information or programs contained herein. No warranty may be created or extended by sales representatives or written sales materials. Every company is different and the advice and strategies contained herein may not be suitable for your situation.

Designations used by companies to distinguish their products are often claimed as trademarks or registered trademarks. In all instances in which Morgan & Claypool is aware of a claim, the product names appear in initial capital or all capital letters. Readers, however, should contact the appropriate companies for more complete information regarding trademarks and registration.

The VR Book: Human-Centered Design for Virtual Reality

Jason Jerald

books.acm.org

www.morganclaypool.com

ISBN: 978-1-97000-112-9 paperback

ISBN: 978-1-97000-113-6 ebook

ISBN: 978-1-62705-114-3 ePub

ISBN: 978-1-97000-115-0 hardcover

Series ISSN: 2374-6769 print 2374-6777 electronic

A publication in the ACM Books series, #8

Editor in Chief: M. Tamer Özsu, *University of Waterloo*

Area Editor: John C. Hart, *University of Illinois*

First Edition

10 9 8 7 6 5 4 3 2 1

DOIs

[10.1145/2792790](#) Book

10.1145/2792790.2792791	Preface/Intro	10.1145/2792790.2792810	Chap. 16	10.1145/2792790.2792829	Chap. 32
10.1145/2792790.2792792	Part I	10.1145/2792790.2792811	Chap. 17	10.1145/2792790.2792830	Chap. 33
10.1145/2792790.2792793	Chap. 1	10.1145/2792790.2792812	Chap. 18	10.1145/2792790.2792831	Chap. 34
10.1145/2792790.2792794	Chap. 2	10.1145/2792790.2792813	Chap. 19	10.1145/2792790.2792832	Part VII
10.1145/2792790.2792795	Chap. 3	10.1145/2792790.2792814	Part IV	10.1145/2792790.2792833	Chap. 35
10.1145/2792790.2792796	Chap. 4	10.1145/2792790.2792815	Chap. 20	10.1145/2792790.2792834	Chap. 36
10.1145/2792790.2792797	Chap. 5	10.1145/2792790.2792816	Chap. 21	10.1145/2792790.2792835	Appendix A
10.1145/2792790.2792798	Part II	10.1145/2792790.2792817	Chap. 22	10.1145/2792790.2792836	Appendix B
10.1145/2792790.2792799	Chap. 6	10.1145/2792790.2792818	Chap. 23	10.1145/2792790.2792837	Glossary/Refs
10.1145/2792790.2792800	Chap. 7	10.1145/2792790.2792819	Chap. 24		
10.1145/2792790.2792801	Chap. 8	10.1145/2792790.2792821	Chap. 25		
10.1145/2792790.2792802	Chap. 9	10.1145/2792790.2792820	Part V		
10.1145/2792790.2792803	Chap. 10	10.1145/2792790.2792822	Chap. 26		
10.1145/2792790.2792804	Chap. 11	10.1145/2792790.2792823	Chap. 27		
10.1145/2792790.2792805	Part III	10.1145/2792790.2792824	Chap. 28		
10.1145/2792790.2792806	Chap. 12	10.1145/2792790.2792825	Chap. 29		
10.1145/2792790.2792807	Chap. 13	10.1145/2792790.2792826	Part VI		
10.1145/2792790.2792808	Chap. 14	10.1145/2792790.2792827	Chap. 30		
10.1145/2792790.2792809	Chap. 15	10.1145/2792790.2792828	Chap. 31		

This book is dedicated to the entire community of VR researchers, developers, designers, entrepreneurs, managers, marketers, and users. It is their passion for, and contributions to, VR that makes this all possible. Without this community, working in isolation would make VR an interesting niche research project that could neither be shared nor improved upon by others. If you choose to join this community, your pursuit of VR experiences may very well be the most intense years of your life, but you will find the rewards well worth the effort. Perhaps the greatest rewards will come from the users of your experiences—for if you do VR well then your users will tell you how you have changed their lives—and that is how we change the world.

There are many facets to VR creation, ranging from getting the technology right, sometimes during exhausting overnight sessions, to the fascinating and abundant collaboration with others in the VR community. At times, what we are embarking on can feel overwhelming. When that happens, I look to a quote by George Bernard Shaw posted on my wall and am reminded about the joy of being a part of the VR revolution.

This is the true joy in life, the being used for a purpose recognized by yourself as a mighty one; the being a force of nature . . . I am of the opinion that my life belongs to the whole community and as long as I live it is my privilege to do for it whatever I can. I want to be thoroughly used up when I die, for the harder I work, the more I live. I rejoice in life for its own sake. Life is no “brief candle” to me. It is sort of a splendid torch which I have a hold of for the moment, and I want to make it burn as brightly as possible before handing it over to future generations.

This book is thus dedicated to the VR community and the future generations that will create many virtual worlds as well as change the real world. My purpose in writing this book is to welcome others into this VR community, to help fuel a VR revolution that changes the world and the way we interact with it and each other, in ways that have never before been possible—until now.

Contents

Preface	xix
Figure Credits	xxvii
Overview	1

PART I INTRODUCTION AND BACKGROUND 7

Chapter 1	What Is Virtual Reality?	9
1.1	The Definition of Virtual Reality	9
1.2	VR Is Communication	10
1.3	What Is VR Good For?	12
Chapter 2	A History of VR	15
2.1	The 1800s	15
2.2	The 1900s	18
2.3	The 2000s	27
Chapter 3	An Overview of Various Realities	29
3.1	Forms of Reality	29
3.2	Reality Systems	30
Chapter 4	Immersion, Presence, and Reality Trade-Offs	45
4.1	Immersion	45
4.2	Presence	46
4.3	Illusions of Presence	47
4.4	Reality Trade-Offs	49

*Practitioner chapters are marked with a star next to the chapter number. See page 5 for an explanation.

- ★ **Chapter 5** The Basics: Design Guidelines 53
 - 5.1 Introduction and Background 53
 - 5.2 VR Is Communication 53
 - 5.3 An Overview of Various Realities 54
 - 5.4 Immersion, Presence, and Reality Trade-Offs 54

PART II PERCEPTION 55

- Chapter 6** Objective and Subjective Reality 59
 - 6.1 Reality Is Subjective 59
 - 6.2 Perceptual Illusions 61
- Chapter 7** Perceptual Models and Processes 71
 - 7.1 Distal and Proximal Stimuli 71
 - 7.2 Sensation vs. Perception 72
 - 7.3 Bottom-Up and Top-Down Processing 73
 - 7.4 AffERENCE and Efference 73
 - 7.5 Iterative Perceptual Processing 74
 - 7.6 The Subconscious and Conscious 76
 - 7.7 Visceral, Behavioral, Reflective, and Emotional Processes 77
 - 7.8 Mental Models 79
 - 7.9 Neuro-Linguistic Programming 80
- Chapter 8** Perceptual Modalities 85
 - 8.1 Sight 85
 - 8.2 Hearing 99
 - 8.3 Touch 103
 - 8.4 Proprioception 105
 - 8.5 Balance and Physical Motion 106
 - 8.6 Smell and Taste 107
 - 8.7 Multimodal Perceptions 108
- Chapter 9** Perception of Space and Time 111
 - 9.1 Space Perception 111
 - 9.2 Time Perception 124
 - 9.3 Motion Perception 129

Chapter 10 Perceptual Stability, Attention, and Action 139

- 10.1 Perceptual Constancies 139
- 10.2 Adaptation 143
- 10.3 Attention 146
- 10.4 Action 151

★ Chapter 11 Perception: Design Guidelines 155

- 11.1 Objective and Subjective Reality 155
- 11.2 Perceptual Models and Processes 155
- 11.3 Perceptual Modalities 156
- 11.4 Perception of Space and Time 156
- 11.5 Perceptual Stability, Attention, and Action 157

PART III ADVERSE HEALTH EFFECTS 159**Chapter 12** Motion Sickness 163

- 12.1 Scene Motion 163
- 12.2 Motion Sickness and Vection 164
- 12.3 Theories of Motion Sickness 165
- 12.4 A Unified Model of Motion Sickness 169

Chapter 13 Eye Strain, Seizures, and Aftereffects 173

- 13.1 Accommodation-Vergence Conflict 173
- 13.2 Binocular-Occlusion Conflict 173
- 13.3 Flicker 174
- 13.4 Aftereffects 174

Chapter 14 Hardware Challenges 177

- 14.1 Physical Fatigue 177
- 14.2 Headset Fit 178
- 14.3 Injury 178
- 14.4 Hygiene 179

Chapter 15 Latency 183

- 15.1 Negative Effects of Latency 183
- 15.2 Latency Thresholds 184
- 15.3 Delayed Perception as a Function of Dark Adaptation 185

15.4 Sources of Delay 187

15.5 Timing Analysis 193

Chapter 16 Measuring Sickness 195

16.1 The Kennedy Simulator Sickness Questionnaire 195

16.2 Postural Stability 196

16.3 Physiological Measures 196

Chapter 17 Summary of Factors That Contribute to Adverse Effects 197

17.1 System Factors 198

17.2 Individual User Factors 200

17.3 Application Design Factors 203

17.4 Presence vs. Motion Sickness 205

★ **Chapter 18** Examples of Reducing Adverse Effects 207

18.1 Optimize Adaptation 207

18.2 Real-World Stabilized Cues 207

18.3 Manipulate the World as an Object 209

18.4 Leading Indicators 210

18.5 Minimize Visual Accelerations and Rotations 210

18.6 Ratcheting 211

18.7 Delay Compensation 211

18.8 Motion Platforms 212

18.9 Reducing Gorilla Arm 213

18.10 Warning Grids and Fade-Outs 213

18.11 Medication 213

★ **Chapter 19** Adverse Health Effects: Design Guidelines 215

19.1 Hardware 215

19.2 System Calibration 216

19.3 Latency Reduction 216

19.4 General Design 217

19.5 Motion Design 218

19.6 Interaction Design 219

19.7 Usage 220

19.8 Measuring Sickness 221

PART IV CONTENT CREATION 223

Chapter 20 High-Level Concepts of Content Creation 225

- 20.1 Experiencing the Story 225
- 20.2 The Core Experience 228
- 20.3 Conceptual Integrity 229
- 20.4 Gestalt Perceptual Organization 230

Chapter 21 Environmental Design 237

- 21.1 The Scene 237
- 21.2 Color and Lighting 238
- 21.3 Audio 239
- 21.4 Sampling and Aliasing 240
- 21.5 Environmental Wayfinding Aids 242
- 21.6 Real-World Content 246

Chapter 22 Affecting Behavior 251

- 22.1 Personal Wayfinding Aids 251
- 22.2 Center of Action 254
- 22.3 Field of View 255
- 22.4 Casual vs. High-End VR 255
- 22.5 Characters, Avatars, and Social Networking 257

★ **Chapter 23** Transitioning to VR Content Creation 261

- 23.1 Paradigm Shifts from Traditional Development to VR Development 261
- 23.2 Reusing Existing Content 262

★ **Chapter 24** Content Creation: Design Guidelines 267

- 24.1 High-Level Concepts of Content Creation 267
- 24.2 Environmental Design 269
- 24.3 Affecting Behavior 271
- 24.4 Transitioning to VR Content Creation 272

PART V INTERACTION 275

Chapter 25 Human-Centered Interaction 277

- 25.1 Intuitiveness 277

25.2	Norman's Principles of Interaction Design	278
25.3	Direct vs. Indirect Interaction	284
25.4	The Cycle of Interaction	285
25.5	The Human Hands	287
★ Chapter 26	VR Interaction Concepts	289
26.1	Interaction Fidelity	289
26.2	Proprioceptive and Egocentric Interaction	291
26.3	Reference Frames	291
26.4	Speech and Gestures	297
26.5	Modes and Flow	301
26.6	Multimodal Interaction	302
26.7	Beware of Sickness and Fatigue	303
26.8	Visual-Physical Conflict and Sensory Substitution	304
★ Chapter 27	Input Devices	307
27.1	Input Device Characteristics	307
27.2	Classes of Hand Input Devices	311
27.3	Classes of Non-hand Input Devices	317
★ Chapter 28	Interaction Patterns and Techniques	323
28.1	Selection Patterns	325
28.2	Manipulation Patterns	332
28.3	Viewpoint Control Patterns	335
28.4	Indirect Control Patterns	344
28.5	Compound Patterns	350
★ Chapter 29	Interaction: Design Guidelines	355
29.1	Human-Centered Interaction	355
29.2	VR Interaction Concepts	358
29.3	Input Devices	361
29.4	Interaction Patterns and Techniques	363
 PART VI ITERATIVE DESIGN 369		
Chapter 30	Philosophy of Iterative Design	373
30.1	VR Is Both an Art and a Science	373

30.2	Human-Centered Design	373
30.3	Continuous Discovery through Iteration	374
30.4	There Is No One Way—Processes Are Project Dependent	375
30.5	Teams	376
★ Chapter 31	The Define Stage	379
31.1	The Vision	380
31.2	Questions	380
31.3	Assessment and Feasibility	382
31.4	High-Level Design Considerations	383
31.5	Objectives	383
31.6	Key Players	384
31.7	Time and Costs	385
31.8	Risks	387
31.9	Assumptions	388
31.10	Project Constraints	388
31.11	Personas	391
31.12	User Stories	392
31.13	Storyboards	393
31.14	Scope	393
31.15	Requirements	395
★ Chapter 32	The Make Stage	401
32.1	Task Analysis	402
32.2	Design Specification	405
32.3	System Considerations	410
32.4	Simulation	413
32.5	Networked Environments	415
32.6	Prototypes	421
32.7	Final Production	423
32.8	Delivery	424
★ Chapter 33	The Learn Stage	427
33.1	Communication and Attitude	428
33.2	Research Concepts	429
33.3	Constructivist Approaches	436
33.4	The Scientific Method	443
33.5	Data Analysis	447

★ Chapter 34	Iterative Design: Design Guidelines	453
34.1	Philosophy of Iterative Design	453
34.2	The Define Stage	454
34.3	The Make Stage	458
34.4	The Learn Stage	464
PART VII	THE FUTURE STARTS NOW	471
Chapter 35	The Present and Future State of VR	473
35.1	Selling VR to the Masses	473
35.2	Culture of the VR Community	474
35.3	Communication	475
35.4	Standards and Open Source	480
35.5	Hardware	483
35.6	The Convergence of AR and VR	484
★ Chapter 36	Getting Started	485
Appendix A	Example Questionnaire	489
Appendix B	Example Interview Guidelines	495
	Glossary	497
	References	541
	Index	567
	Author's Biography	601

Preface

I've known for some time that I wanted to write a book on VR. However, I wanted to bring a unique perspective as opposed to simply writing a book for the sake of doing so. Then insight hit me during Oculus Connect in the fall of 2014. After experiencing the Oculus Crescent Bay demo, I realized the hardware is becoming really good. Not by accident, but as a result of some of the world's leading engineers diligently working on the technical challenges with great success. What the community now desperately needs is for content developers to understand human perception as it applies to VR, to design experiences that are comfortable (i.e., do not make you sick), and to create intuitive interactions within their immersive creations. That insight led me to the realization that I need to stop focusing primarily on technical implementation and start focusing on higher-level challenges in VR and its design. Focusing on these challenges offers the most value to the VR community of the present day, and is why a new VR book is necessary.

I had originally planned to self-publish instead of spending valuable time on pitching the idea of a new VR book to publishers, which I know can be a very long, arduous, and disappointing path. Then one of the most serendipitous things occurred. A few days after having the idea for a new VR book and committing to make it happen, my former undergraduate adviser, Dr. John Hart, now professor of computer science at the University of Illinois at Urbana-Champaign and, more germanely, the editor of the Computer Graphics Series of ACM Books, contacted me out of the blue. He informed me that Michael Morgan of Morgan & Claypool Publishers wanted to publish a book on VR content creation and John thought I was the person to make it happen. It didn't take much time to enthusiastically accept their proposal, as their vision for the book was the same as mine.

I've condensed approximately 20 years and 30,000 hours of personal study, application, notes, and VR sickness into this book. Those two decades of my VR career have somehow been summarized with six months of intense writing and editing from

January to July of 2015. I knew, as others working hard on VR know, the time is now, and after finishing up a couple of contracts I was finally able to put other aspects of my life on hold (clients have been very understanding!), sometimes writing more than 75 hours a week. I hope this rush to get these timely concepts into one book does not sacrifice the overall quality, and I am always seeking feedback. Please contact me at book@nextgeninteractions.com to let me know of any errors, lack of clarity, or important points missed so that an improved second edition can emerge at some point.

It all started in 1980

I owe much of my pursuit of developing VR systems and applications to my parents. It started at the age of six with an Atari game system that my family couldn't afford but I wanted so badly. For Christmas 1980, I somehow received it. Then in 1986 my mother took away the games in hopes of curing my addiction, which naturally forced me to make my own games (with advanced moving 2D sprites!) on the family's Commodore 64. Here, I taught myself programming and essential software design concepts, such as simple computer graphics and collision detection, which turned out to be quite important for VR development. At that point, she thankfully gave up on trying to cure me. I also owe my father, who connected me with my first internship in the summer of 1992 after my junior year of high school at a design firm where he worked. My job was to deliver plots from the printers to the designers. This did not completely fill my time, and I somehow managed to gain access to AutoCad and 3D Studio R2 (long before today's 3ds Max!). I was soon extruding 2D architectural plans into 3D worlds and animating horrible-looking flat-shaded polygons in the evenings and weekends when nobody was the wiser.

SIGGRAPH 1995 and 1996

After regrettably missing SIGGRAPH 1994, I somehow acquired the conference course notes on creating virtual worlds and other related concepts. Soon afterwards, I discovered the SIGGRAPH Student Volunteer Program and realized that was my ticket to the conference. After realizing what I missed, I was not going to leave the opportunity to chance. I sought out a referral from Dr. John Hart, a professor very heavily involved with SIGGRAPH. He must have given me a solid referral based on my initiative and passion for computer graphics, as I had not yet taken a class from him. John would later become my undergraduate adviser and eventually editor of this book. Both his advice through the years and SIGGRAPH had an enormous influence on my career. In fact, I ended up leading the SIGGRAPH Student Volunteer Program more than a

decade later, out of appreciation and respect for its ability to inspire careers in computer graphics.

I was quite fortunate to be accepted to the 1995 SIGGRAPH Student Volunteer Program in Los Angeles, which is where I first experienced VR and have been fully hooked ever since. Having never attended a conference up to that point that I was this passionate about, I was completely blown away by its people and its magnitude. I was no longer alone in this world; I had finally found “my people.” Soon afterward, I found my first VR mentor, Richard May, now director of the National Visualization and Analytics Center. I enthusiastically jumped in on helping him build one of the world’s first immersive VR medical applications at Battelle Pacific Northwest National Laboratories. Richard and I returned to SIGGRAPH 1996 in New Orleans, where I more intentionally sought out VR. VR was now bigger than ever and I remember two things from the conference that defined my future. One was a course on VR interactions by the legendary Dr. Frederick P. Brooks, Jr. and one of his students, Mark Miné. The other event was a VR demo that I still remember vividly to this day—and still one of the most compelling demos I’ve seen yet. It was a virtual Legoland where the user built an entire world around himself by snapping Legos and Lego neighborhoods together in an intuitive, almost effortless manner.

Post-1996

SIGGRAPH 1996 was a turning point for me in knowing what I wanted to do with my life. Since that time, I have been fortunate to know and work with many individuals that have inspired me. This book is largely a result of their work and their mentorship.

Since 1996, Dr. Brooks unknowingly inspired me to move on from a full-time dream VR job in the late 1990s at HRL Laboratories to pursue VR at the next level at UNC-Chapel Hill. Once there, I managed to persuade Dr. Brooks to become my PhD adviser in studying VR latency. In addition to being a major influence on my career, two of his books—*The Mythical Man Month* and more recently *The Design of Design*—are heavily referenced throughout Part VI, Iterative Design. He also had significant input for Part II, Perception, especially Chapter 16, Latency, as some of that writing originally came from my dissertation that was largely a result of significant suggestions from him. His serving as adviser for Mark Miné’s seminal work on VR interaction also indirectly affected Part V, Interaction. It is still quite difficult for me to fathom that before Dr. Brooks came along, bytes—the molecules of all digital technology—were not always 8 bits in size like they are defined to be today. That 8-bit design decision he made in the 1960s, which most all of us computer scientists now just assume to be an inherent truth, is just one of his many contributions to computers in general,

along with his more specific contributions to VR research. It boggles my mind even more how a small-town kid like myself, growing up in a tiny town of 1,200 people on the opposite side of the country, somehow came to study under an ACM Turing Award recipient (the equivalent of the Nobel Prize in computer science). For that, I will forever be grateful to Dr. Brooks and the UNC-Chapel Hill Department of Computer Science for admitting me.

In 2009, I interviewed with Paul Mlyniec, president of Digital ArtForms. At some point during the interview, I came to the realization that this was the man that led the VR Lego work from SIGGRAPH that had inspired and driven me for over a decade. The interface in that Lego demo is one of the first implementations of what I refer to as the 3D Multi-Touch Pattern in Section 28.3.3, a pattern two decades ahead of its time. I've now worked closely with Paul and Digital ArtForms for six years on various projects, including ones that have improved upon 3D Multi-Touch (Sixense's MakeVR also uses this same viewpoint control implementation). We are currently working together (along with Sixense and Wake Forest School of Medicine) on immersive games for neuroscience education funded by the National Institutes of Health. In addition to these games, several other examples of Digital ArtForms' work (along with work from its sister company Sixense that Paul was essential in helping to form) are featured throughout the book.

VR Today

After a long VR drought after the 1990s, VR is bigger than ever at SIGGRAPH. An entire venue, the VR Village, is dedicated specifically to VR, and I have the privilege of leading the Immersive Realities Contest. After so many years of heavy involvement with my SIGGRAPH family, it feels quite fitting that this book is launched at the SIGGRAPH bookstore on the 20th anniversary of my discovery of SIGGRAPH and VR.

I joke that when I did my PhD Dissertation on latency perception for head-mounted displays, perhaps ten people in the world cared about VR latency—and five of those people were on my committee (in three different time zones!). Then in 2011, people started wanting to know more. I remember having lunch with Amir Rubin, CEO of Sixense. He was inquiring about consumer HMDs for games. I thought the idea was crazy; we could barely get VR to work well in a lab and he wanted to put it in people's living rooms! Other inquiries led to work with companies such as Valve and Oculus. All three of these companies are doing spectacular jobs of making high-quality VR accessible, and now suddenly everyone wants to experience VR. Largely due to these companies' efforts, VR has turned a corner, transitioning from a specialized labo-

ratory instrument available only to the technically elite, to a mainstream mode of content consumption available to any consumer. Now, most everyone that is involved in VR technology understands at least the basics of latency and its challenges/dangers, most notably motion sickness—the greatest risk for VR. Even better, ultra-low-latency hardware technologies (e.g., low-persistence OLED displays) that I once unsuccessfully searched the world for (I ended up having to build/simulate my own; the best I did was 7.4 ms of end-to-end latency with tracking and rendering performed at ~1,500 frames per second) are being developed in mass quantities by giants such as Samsung, Sony, Valve, and Oculus. Times have certainly changed!

The last 20 years of pursuing VR have truly been a dream. During that time, I imagined and even seriously considered starting companies devoted to VR, but it was never feasible until recently. Today it is more of a real fantasy than a simple dream, now that VR technology is delivering upon its promise of the 1990s. Describing the feeling is like trying to describe a VR experience. Words cannot do justice as to what it is like to be a part of this VR community and contributing to the VR revolution. Through my VR consulting and contracting firm, NextGen Interactions, I have the privilege of working with some of the best companies in the world that are able to do things that could only previously be imagined. Virtual reality is unlike any technology devised to date and has the potential not only to change the fictional synthetic worlds we make up but to change people's real lives. I very much look forward to seeing what this community discovers and creates over the next 20 years!

Acknowledgments

It would be untrue if I claimed that this book was completely based on my own work. The book was certainly not written in isolation and is a result of contributions from a vast number of mentors, colleagues, and friends in the VR community who have been working just as hard as I have over the years to make VR something real. I couldn't possibly say everything there is about VR, and I encourage readers to further investigate the more than 300 references discussed throughout the book, along with other writings. I've learned so much from those who came before me as well as those newer to VR. Unfortunately, it is not possible to list everyone who has influenced this book, but the following gives thanks to those who have given the most support.

First of all, this book would never have been anything beyond a simple self-published work without the team at Morgan & Claypool Publishers and ACM Books. On the Morgan & Claypool Publishers side, I'd like to thank Executive Editor Diane Cerra, Editorial Assistant Samantha Draper, Copyeditor Sara Kreisman, Production

Manager Paul Anagnostopoulos, Artist Laurel Muller, Proofreader Jennifer McClain, and President Michael Morgan. On the ACM Books side, I'd like to thank Computer Graphics Series Editor John Hart and Editor-in-Chief M. Tamer Özsu.

This book you are reading is very different and much better than initial drafts due to suggestions from great reviewers. Reviewers of the book include Paul Mlyniec (president of Digital ArtForms), Arun Yoganandan (formerly of Digital ArtForms and Disney, now a research engineer at Samsung), Russell Taylor (former professor of computer science at UNC–Chapel Hill, now cofounder of Rheomics and independent consultant at Relia Solve), Beau Cronin (an expert VR neuroscientist at Salesforce, who is writing his own VR book), Mike McArdle (cofounder of the Virtual Reality Learning Experience), Ryan McMahan (professor at University of Texas at Dallas), Eugene Nalivaiko (professor at New Castle University), Kyle Yamamoto (cofounder and game developer at MochiBits), Ann McNamara (professor at Texas A&M), Mark Fagiano (professor at Emory University), Zach Wendt (computer games guru and independent software developer), Francisco Ortega (postdoc fellow at Florida International University and who is writing his own VR book), Matt Cook (emerging technologies librarian at the University of Oklahoma), Sharif Razzaque (CTO of Inneroptic), Neeta Nahta (cofounder of NextGen Interactions), Kevin Rio (UX researcher at Microsoft), Chris Puszczak (project manager at Burnout Game Ventures and SymbioVR), Daniel Ackley (psychologist), my mother, Susan Jerald (behavioral psychologist), and my father, Rick Jerald (CEO of Envirocept).

Additional specific input came from Mark Bolas (professor at the University of Southern California), Neil Schneider (founder of Meant to be Seen and founder/executive director of the Immersive Technology Alliance), Eric Greenbaum (attorney and founder of Jema VR), John Baker (COO of Chosen Realities), Denny Unger (founder and president of Cloudhead Games), Andrew Robinson and Sigurdur Gunnarsson (developers at CCP Games), Max Rheiner (founder and CTO of Somniacs), Chadwick Wingrave (founder of Conquest Creations), Barry Downes (CEO of TSSG), Denise Quesnel (research associate at Emily Carr University of Art and Design, cofounder of the Canadian & Advanced Imaging Society, and cochair of the SIGGRAPH VR Village), Nick England (founder and CEO of 3rdTech), Jesse Joudrey (CEO of VRChat and CTO of Jespionage Entertainment), David Collodi (cofounder and CTO of CI Dynamics), David Beaver (cofounder of the Overview Institute), and Bill Howe (CEO of the Growth Engine Group).

I've been very fortunate to be employed by some amazing organizations and even more amazing bosses that are world-leading experts in VR. These individuals gave me the opportunity to make a career out of what I love doing, and working with them led to much of what is contained within these pages: Richard May of Battelle

Pacific Northwest National Laboratories (now director of the National Visualization and Analytics Center), Michael Papka of Argonne National Laboratories, Mike Daily of HRL Laboratories, Greg Schmidt of the Naval Research Laboratory, Steve Ellis and Dov Adelstein of NASA Ames, Paul Mlyniec of Digital ArtForms, and Michael Abrash at Valve (now at Oculus).

In addition to those listed above, the following individuals have been invaluable as VR advisers/mentors throughout the years: Howard Neely, research project manager at HRL Laboratories and now CEO at Three Birds Systems; Ron Azuma, senior research staff member at HRL Laboratories and now at Intel; Fred Brooks, Henry Fuchs, Mary Whitton, and Anselmo Lastra—all professors at UNC–Chapel Hill; Jeff Bellinghausen, formerly lead engineer at Digital ArtForms and CTO at Sixense, now at Valve; and Amir Rubin, CEO of Sixense.

I also very much appreciate working with the teams at Sixense, Valve, and Oculus. I've been incredibly impressed with the people and their commitment to creating the highest-quality software and hardware, and not just resurrecting VR but raising awareness to levels beyond anything that previously existed.

Others I've had the pleasure of working with who have influenced this book include Rupert Meghnot and his team at Burnout Game Ventures, especially Chris Puszczak of the SymbioVR team; Jan Goetgeluk of Virtuix; Simon Solotko, the Kickstarter guru who marketed some of the biggest VR Kickstarters such as the Virtuix Omni, the Cyberith Virtualizer, and the Sixense STEM; the UNC Effective Virtual Environments team; Karl Krantz of Silicon Valley Virtual Reality, who was the first to pioneer local meetups where his events are duplicated throughout the world; and Henry Velez, founder of the RTP Virtual Reality Enthusiasts, which I now have the privilege of leading.

I've also enjoyed working with literally thousands of volunteers through ACM SIGGRAPH, IEEE VR, and IEEE 3DUI, ranging from student volunteers to the legends of computer graphics and VR. In particular, I'd like to thank the SIGGRAPH conference committees of the last 20 years. Of course, I've also enjoyed working with those outside of conferences, ranging from those in academics to those doing start-ups. Then there are those who have given me invaluable feedback on VR projects, ranging from middle schoolers to VR experts.

I'd especially like to thank Neeta Nahta, my partner in business and life, who comes from a completely different world of sales, financial analysis, and corporate boardrooms. Her input enables me to see VR from a different perspective than most anyone else in the VR community. Seeing VR as a sustainable and profitable business that adds real value to the world will ensure that this time VR is here for good.

I would also like to thank the Link Foundation for supporting me through the Link Foundation Advanced Simulation and Training Fellowship, which helped to fund

graduate work in studying and reducing latency. It is certainly an honor to be funded by the foundation started by Edwin A. Link, who built the world's first mechanical flight simulator in 1928 (Figure 3.5), which some consider to be one of the first VR systems, as well as to be among a list of world-leading VR experts who also received the Fellowship. I highly encourage any graduate student readers who have already selected a dissertation topic to apply for the Link Foundation Fellowship (\$28,500 to be awarded for 2016). Details are available at <http://www.linksim.org>.

Figure Credits

Figure 1.1 Adapted from: Dale, E. (1969). *Audio-Visual Methods in Teaching* (3rd ed.). The Dryden Press. Based on Edward L. Counts Jr.

Figure 2.1 Based on: Ellis, S. R. (2014). Where are all the Head Mounted Displays? Retrieved April 14, 2015, from http://humansystems.arc.nasa.gov/groups/acd/projects/hmd_dev.php.

Figure 2.3 Courtesy of The National Media Museum / Science & Society Picture Library, United Kingdom.

Figure 2.4 From: Pratt, A. B. (1916). Weapon. US.

Figure 2.5 Courtesy of Edwin A. Link and Marion Clayton Link Collections, Binghamton University Libraries' Special Collections and University Archives, Binghamton University.

Figure 2.6 From: Weinbaum, S. G. (1935, June). Pygmalion's Spectacles. *Wonder Stories*.

Figure 2.7 From: Heilig, M. (1960). Stereoscopic-television apparatus for individual use.

Figure 2.8 Courtesy of © Morton Heilig Legacy.

Figure 2.9 From: Comeau, C.P. & Brian, J.S. "Headsight television system provides remote surveillance," *Electronics*, November 10, 1961, pp. 86-90.

Figure 2.10 From: Rochester, N., & Seibel, R. (1962). Communication Device. US.

Figure 2.11 (left) Courtesy of Tom Furness.

Figure 2.11 (right) From: Sutherland, I. E. (1968). A Head-Mounted Three Dimensional Display. In *Proceedings of the 1968 Fall Joint Computer Conference AFIPS* (Vol. 33, part 1, pp. 757-764). Copyright © ACM 1968. Used with permission. DOI: [10.1145/1476589.1476686](https://doi.org/10.1145/1476589.1476686)

Figure 2.12 From: Brooks, F. P., Ouh-Young, M., Batter, J. J., & Jerome Kilpatrick, P. (1990). Project GROPE Haptic displays for scientific visualization. *ACM SIGGRAPH Computer Graphics*, 24(4), 177-185. Copyright © ACM 1990. Used with permission. DOI: [10.1145/97880.97899](https://doi.org/10.1145/97880.97899)

Figure 2.13 Courtesy of NASA/S.S. Fisher, W. Sisler, 1988

Figure 3.1 Adapted from: Milgram, P., & Kishino, F. (1994). Taxonomy of mixed reality visual displays. *IEICE Transactions on Information and Systems*, E77-D (12), 1321-1329. DOI: [10.1.1.102.4646](https://doi.org/10.1.1.102.4646)

Figure 3.2 Adapted from: Jerald, J. (2009). *Scene-Motion-and Latency-Perception Thresholds for Head-Mounted Displays*. Department of Computer Science, University of North Carolina at Chapel Hill.

Figure 3.3 (top left) Courtesy of Oculus VR.

Figure 3.3 (top right) Courtesy of CastAR.

Figure 3.3 (lower left) Courtesy of Marines Magazine.

Figure 3.3 (lower right) From: Jerald, J., Fuller, A. M., Lastra, A., Whitton, M., Kohli, L., & Brooks, F. (2007). Latency compensation by horizontal scanline selection for head-mounted displays. *Proceedings of SPIE*, 6490, 64901Q–64901Q–11. Copyright © 2007 Society of Photo Optical Instrumentation Engineers. Used with permission.

Figure 3.4 (left) From: Cruz-Neira, C., Sandin, D. J., DeFanti, T. A., Kenyon, R. V., & Hart, J. C. (1992). The CAVE: audio visual experience automatic virtual environment. *Communications of the ACM*. Copyright © ACM 1992. Used with permission. DOI: [10.1145/129888.129892](https://doi.org/10.1145/129888.129892)

Figure 3.4 (right) From: Daily, M., Sarfaty, R., Jerald, J., McInnes, D., & Tinker, P. (1999). The CABANA: A Re-configurable Spatially Immersive Display. In *Projection Technology Workshop* (pp. 123–132). Used with permission.

Figure 3.5 From: Jerald, J., Daily, M. J., Neely, H. E., & Tinker, P. (2001). Interacting with 2D Applications in Immersive Environments. In *EUROIMAGE International Conference on Augmented Virtual Environments and 3d Imaging* (pp. 267–270). Used with permission.

Figure 3.6 From: Krum, D. M., Suma, E. A., & Bolas, M. (2012). Augmented reality using personal projection and retroreflection. *Personal and Ubiquitous Computing*, 16(1), 17–26. Copyright © 2012 Springer. Used with Kind permission of Springer Science + Business Media. DOI: [10.1007/s00779-011-0374-4](https://doi.org/10.1007/s00779-011-0374-4)

Figure 3.7 Courtesy of USC Institute for Creative Technologies.

Figure 3.8 (left) Courtesy of Geomedia.

Figure 3.8 (right) Courtesy of NextGen Interactions.

Figure 3.9 Courtesy of Tactical Haptics.

Figure 3.10 Courtesy of Dexta Robotics.

Figure 3.11 Courtesy of INITION.

Figure 3.12 Courtesy of Haptic Workstation with HMD at VRLab in EPFL, Lausanne, 2005.

Figure 3.13 Courtesy of Shifz, Syntharturalist Art Association.

Figure 3.14 Courtesy of Swissnex San Francisco and Myleen Hollero.

Figure 3.15 Courtesy of Virtuix.

Figure 4.1 Courtesy of Gail Drinnan/Shutterstock.com.

Figure 4.2 Courtesy of NextGen Interactions.

Figure 4.3 Based on: Ho, C. C., & MacDorman, K. F. (2010). Revisiting the uncanny valley theory: Developing and validating an alternative to the Godspeed indices. *Computers in Human Behavior*, 26(6), 1508–1518. DOI: [10.1016/j.chb.2010.05.015](https://doi.org/10.1016/j.chb.2010.05.015)

Figure II.1 Courtesy of Sleeping cat/Shutterstock.com.

Figure 6.2 From: Harmon, Leon D. (1973). The Recognition of Faces. *Scientific American*, 229 (5). © 2015 Scientific American, A Division of Nature America, Inc. Used with permission.

Figure 6.4 Courtesy of Fibonacci.

Figure 6.5b From: Pietro Guardini and Luciano Gamberini, (2007) The Illusory Contoured Tilting Pyramid. Best illusion of the year contest 2007. Sarasota, Florida. Available on line at <http://illusionoftheyear.com/cat/top-10-finalists/2007/>. Used with permission.

Figure 6.5c From: Lehar, S. (2007). The Constructive Aspect of Visual Perception: A Gestalt Field Theory Principle of Visual Reification Suggests a Phase Conjugate Mirror Principle of Perceptual Computation. Used with permission.

Figure 6.6a, b Based on: Lehar, S. (2007). The Constructive Aspect of Visual Perception: A Gestalt Field Theory Principle of Visual Reification Suggests a Phase Conjugate Mirror Principle of Perceptual Computation.

Figure 6.9 Courtesy of PETER ENDIG/AFP/Getty Images.

Figure 6.12 Image © 2015 The Association for Research in Vision and Ophthalmology.

Figure 6.13 Courtesy of MarcoCapra/Shutterstock.com.

Figure 7.1 Based on: Razzaque, S. (2005). *Redirected Walking*. Department of Computer Science, University of North Carolina at Chapel Hill. Used with permission.

Adapted from: Gregory, R. L. (1973). *Eye and Brain: The Psychology of Seeing* (2nd ed.). London: Weidenfeld and Nicolson. Copyright © 1973 The Orion Publishing Group. Used with permission.

Figure 7.2 Adapted from: Goldstein, E. B. (2007). *Sensation and Perception* (7th ed.). Belmont, CA: Wadsworth Publishing.

Figure 7.3 Adapted from: James, T., & Woodsmall, W. (1988). *Time Line Therapy and the Basis of Personality*. Meta Pubns.

Figure 8.1 Based on: Coren, S., Ward, L. M., and Enns, J. T. (1999). *Sensation and Perception* (5th ed.). Harcourt Brace College Publishers.

Figure 8.2 Based on: Badcock, D. R., Palmisano, S., & May, J. G. (2014). Vision and Virtual Environments. In K. S. Hale & K. M. Stanney (Eds.), *Handbook of Virtual Environments* (2nd ed.). CRC Press.

Figure 8.3 Based on: Coren, S., Ward, L. M., & Enns, J. T. (1999). *Sensation and Perception* (5th ed.). Harcourt Brace College Publishers.

Figure 8.4 Adapted from: Clulow, F. W. (1972). *Color: Its principle and their applications*. New York: Morgan & Morgan.

xxx Figure Credits

Figure 8.5 Based on: Coren, S., Ward, L. M., & Enns, J. T. (1999). *Sensation and Perception* (5th ed.). Harcourt Brace College Publishers.

Figure 8.6 Adapted from: Coren, S., Ward, L. M., & Enns, J. T. (1999). *Sensation and Perception* (5th ed.). Harcourt Brace College Publishers.

Figure 8.7 Adapted from: Goldstein, E. B. (2014). *Sensation and Perception* (9th ed.). Cengage Learning.

Figure 8.8 Based on: Burton, T.M.W. (2012) *Robotic Rehabilitation for the Restoration of Functional Grasping Following Stroke*. Dissertation, University of Bristol, England.

Figure 8.9 Based on: Jerald, J. (2009). *Scene-Motion- and Latency-Perception Thresholds for Head-Mounted Displays*. Department of Computer Science, University of North Carolina at Chapel Hill. Used with permission. Adapted from: Martini (1998). *Fundamentals of Anatomy and Physiology*. Upper Saddle River, Prentice Hall.

Figure 9.1 Courtesy of Paolo Gianti/Shutterstock.com

Figure 9.4 Based on: Kersten, D., Mamassian, P. and Knill, D. C. (1997). Moving cast shadows induce apparent motion in depth. *Perception* 26, 171–192.

Figure 9.5 Courtesy of Galkin Grigory/Shutterstock.com

Figure 9.7 Courtesy of NextGen Interactions.

Figure 9.8 Adapted from: Mirzaie, H. (2009, March 16). *Stereoscopic Vision*.

Figure 9.9 Adapted from: Coren, S., Ward, L. M., & Enns, J. T. (1999). *Sensation and Perception* (5th ed.). Harcourt Brace College Publishers.

Figure 10.1 From: Pfeiffer, T., & Memili, C. (2015). GPU-Accelerated Attention Map Generation for Dynamic 3D Scenes. In *IEEE Virtual Reality*. Used with permission. Courtesy of “BMW 3 Series Coupe” by mikepan (Creative Commons Attribution, Share Alike 3.0 at <http://www.blendswap.com>).

Figure 12.1 Adapted from: Razzaque, S. (2005). *Redirected Walking*. Department of Computer Science, University of North Carolina at Chapel Hill.

Figure 14.1 Courtesy of Jema VR.

Figure 15.1 Based on: Gregory, R. L. (1973). *Eye and Brain: The Psychology of Seeing* (2nd ed.). London: Weidenfeld and Nicolson.

Figure 15.2 Adapted from: Jerald, J. (2009). *Scene-Motion-and Latency-Perception Thresholds for Head-Mounted Displays*. Department of Computer Science, University of North Carolina at Chapel Hill.

Figure 15.3 Adapted from: Jerald, J. (2009). *Scene-Motion-and Latency-Perception Thresholds for Head-Mounted Displays*. Department of Computer Science, University of North Carolina at Chapel Hill.

Figure 15.4 Based on: Jerald, J. (2009). *Scene-Motion- and Latency-Perception Thresholds for Head-Mounted Displays*. Department of Computer Science, University of North Carolina at Chapel Hill. Used with permission.

Figure 18.1 Courtesy of CCP Games.

Figure 18.2 Courtesy of NextGen Interactions.

Figure 18.3 Courtesy of Sixense.

Figure 20.1 From: Heider, F., & Simmel, M. (1944). An experimental study of apparent behavior. *The American Journal of Psychology*, 57, 243–259. DOI: [10.2307/1416950](https://doi.org/10.2307/1416950)

Figure 20.2 From: Lehar, S. (2007). The Constructive Aspect of Visual Perception: A Gestalt Field Theory Principle of Visual Reification Suggests a Phase Conjugate Mirror Principle of Perceptual Computation. Used with permission.

Figure 20.3 Courtesy of Sixense.

Figure 20.4 Based on: Wolfe, J. (2006). *Sensation & perception*. Sunderland, Mass.: Sinauer Associates.

Figure 20.5 From: Yoganandan, A., Jerald, J., & Mlyniec, P. (2014). Bimanual Selection and Interaction with Volumetric Regions of Interest. In *IEEE Virtual Reality Workshop on Immersive Volumetric Interaction*. Used with permission.

Figure 20.7 Courtesy of Sixense.

Figure 20.9 Courtesy of Sixense.

Figure 21.1 Courtesy of Digital ArtForms.

Figure 21.3 Courtesy of NextGen Interactions.

Figure 21.4 Courtesy of Digital ArtForms.

Figure 21.5 Courtesy of Digital ArtForms.

Figure 21.6 Image © Strangers with Patrick Watson / Félix & Paul Studios

Figure 21.7 Courtesy of 3rdTech.

Figure 21.8 Courtesy of Digital ArtForms.

Figure 21.9 Courtesy of UNC CISM NIH Resource 5-P41-EB002025 from data collected in Alisa S. Wolberg's laboratory under NIH award HL094740.

Figure 22.1 Courtesy of Digital ArtForms.

Figure 22.2 Courtesy of NextGen Interactions.

Figure 22.3 Courtesy of Visionary VR.

Figure 22.4 From: Daily, M., Howard, M., Jerald, J., Lee, C., Martin, K., McInnes, D., & Tinker, P. (2000). Distributed design review in virtual environments. In *Proceedings of the third international conference on Collaborative virtual environments* (pp. 57–63). ACM. Copyright © ACM 2000. Used with permission. DOI: [10.1145/351006.351013](https://doi.org/10.1145/351006.351013)

Figure 22.5 Courtesy of NextGen Interactions.

Figure 22.6 Courtesy of VRChat.

Figure 25.1 Adapted from: Norman, D. A. (2013). *The Design of Everyday Things, Expanded and Revised Edition. Human Factors and Ergonomics in Manufacturing*. New York, NY: Basic Books. DOI: [10.1002/hfm.20127](https://doi.org/10.1002/hfm.20127)

Figure 26.1 Courtesy of Digital ArtForms.

Figure 26.2 Courtesy of NextGen Interactions.

Figure 26.3 Courtesy of NextGen Interactions.

Figure 26.4 Courtesy of NextGen Interactions.

Figure 26.5 Courtesy of Cloudhead Games.

Figure 27.1 From: Pausch, R., Snoddy, J., Taylor, R., Watson, S., & Haseltine, E. (1996). Disney's Aladdin: first steps toward storytelling in virtual reality. In *Proceedings of the 23rd annual conference on Computer graphics and interactive techniques* (pp. 193–203). ACM Press. Copyright © ACM 1996. Used with permission. DOI: [10.1145/237170.237257](https://doi.org/10.1145/237170.237257)

Figure 27.2 Courtesy of Stock photo © juniorbeep.

Figure 27.3 Courtesy of Sixense and Oculus.

Figure 27.4 Courtesy of CyberGlove Systems LLC.

Figure 27.5 Courtesy of Leap Motion.

Figure 27.6 Courtesy of Dassault Systèmes, iV Lab.

Figure 28.1 (left) Courtesy of Cloudhead Games.

Figure 28.1 (center) Courtesy of NextGen Interactions.

Figure 28.1 (right) Courtesy of Digital ArtForms.

Figure 28.2 From: Pierce, J., Forsberg, A., Conway, M., Hong, S., Zeleznik, R., & Miné, M. R. (1997). Image Plane Interaction Techniques in 3D Immersive Environments. In *ACM Symposium on Interactive 3D Graphics* (pp. 39–44). ACM Press. Copyright © ACM 1997. Used with permission. DOI: [10.1145/253284.253303](https://doi.org/10.1145/253284.253303)

Figure 28.3 Courtesy of Digital ArtForms.

Figure 28.4 From: Hinckley, K., Pausch, R., Goble, J. C., & Kassell, N. F. (1994). Passive Real-world Interface Props for Neurosurgical Visualization. In *Proceedings of the SIGCHI conference on Human factors in computing systems celebrating interdependence - CHI '94* (Vol. 30, pp. 452–458). Copyright © ACM 1994. Used with permission. DOI: [10.1145/191666.191821](https://doi.org/10.1145/191666.191821)

Figure 28.5 Courtesy of Digital ArtForms and Sixense.

Figure 28.6 From: Mlyniec, P., Jerald, J., Yoganandan, A., Seagull, J., Toledo, F., & Schultheis, U. (2011). iMedic: A two-handed immersive medical environment for distributed interactive consultation. In *Studies in Health Technology and Informatics* (Vol. 163, pp. 372–378). Copyright © 2011 IOS Press. Reprinted with permission.

Figure 28.7 From: Schultheis, U., Jerald, J., Toledo, F., Yoganandan, A., & Mlyniec, P. (2012). Comparison of a two-handed interface to a wand interface and a mouse interface for fundamental 3D tasks. In *IEEE Symposium on 3D User Interfaces 2012, 3DUI 2012 - Proceedings* (pp. 117–124). Copyright © 2012 IEEE. Used with permission. DOI: [10.1109/3DUI.2012.6184195](https://doi.org/10.1109/3DUI.2012.6184195)

Figure 28.8 Courtesy of Sixense and Digital ArtForms.

Figure 28.9 From: Bowman, D. A., & Wingrave, C. A. (2001). Design and evaluation of menu systems for immersive virtual environments. In *IEEE Virtual Reality* (pp. 149–156). Copyright © 2001 IEEE. Used with permission. DOI: [10.1109/VR.2001.913781](https://doi.org/10.1109/VR.2001.913781)

Figure 31.1 Based on: Rasmusson, J. (2010). The Agile Samurai—How Agile Masters Deliver Great Software. *Pragmatic Bookshelf*.

Figure 31.2 Based on: Rasmusson, J. (2010). The Agile Samurai—How Agile Masters Deliver Great Software. *Pragmatic Bookshelf*.

Figure 31.6 Courtesy of Geomedia.

Figure 32.1 Courtesy of Andrew Robinson of CCP Games.

Figure 32.2 Based on: Neely, H. E., Belvin, R. S., Fox, J. R., & Daily, M. J. (2004). Multimodal interaction techniques for situational awareness and command of robotic combat entities. In *IEEE Aerospace Conference Proceedings* (Vol. 5, pp. 3297–3305). DOI: [10.1109/AERO.2004.1368136](https://doi.org/10.1109/AERO.2004.1368136)

Figure 32.4 Based on: Daily, M., Howard, M., Jerald, J., Lee, C., Martin, K., McInnes, D., & Tinker, P. (2000). Distributed design review in virtual environments. In *Proceedings of the third international conference on Collaborative virtual environments* (pp. 57–63). ACM. DOI: [10.1145/351006.351013](https://doi.org/10.1145/351006.351013)

Figure 33.2 Adapted from: Gabbard, J. L. (2014). Usability Engineering of Virtual Environments. In K. S. Hale & K. M. Stanney (Eds.), *Handbook of Virtual Environments* (2nd ed., pp. 721–747). CRC Press.

Figure 33.3 Courtesy of NextGen Interactions.

Figure 35.1 Courtesy of USC Institute for Creative Technologies. Principal Investigators: Albert (Skip) Rizzo and Louis-Philippe Morency.

Figure 35.2 From: Grau, C., Ginhoux, R., Riera, A., Nguyen, T. L., Chauvat, H., Berg, M., . . . Ruffini, G. (2014). Conscious Brain-to-Brain Communication in Humans Using Non-Invasive Technologies. *PloS One*, 9(8), e105225. DOI: [10.1371/journal.pone.0105225](https://doi.org/10.1371/journal.pone.0105225)

Figure 35.3 Courtesy of Tyndall and TSSG.

Overview

Virtual reality (VR) can provide our minds with direct access to digital media in a way that seemingly has no limits. However, creating compelling VR experiences is an incredibly complex challenge. When VR is done well, the results are brilliant and pleasurable experiences that go beyond what we can do in the real world. When VR is done badly, not only do users get frustrated, but they can get sick. There are many causes of bad VR; some failures come from the limitations of technology, but many come from a lack of understanding perception, interaction, design principles, and real users. This book discusses these issues by emphasizing the human element of VR. The fact is, if we do not get the human element correct, then no amount of technology will make VR anything more than an interesting tool confined to research laboratories. Even when VR principles are fully understood, the first implementation is rarely novel and almost never ideal due to the complex nature of VR and the countless possibilities that can be created. The VR principles discussed in this book will enable readers to intelligently experiment with the rules and iteratively design toward innovative experiences.

Historically, most VR creators have been engineers (the author included) with expertise in technology and logic but limited in their understanding of humans. This is primarily due to the fact that VR has previously been so technically challenging that it was difficult to build VR experiences without an engineering background. Unfortunately, we engineers often believe “I am human, therefore I understand other humans and what works for them.” However, the way humans perceive and interact with the world is incredibly complex and not typically based on logic, mathematics, or a user manual. If we stick solely to the expertise of knowing it all through engineering and logic, then VR will certainly be doomed. We have to accept human perception and behavior the way it is, not the way logic tells us it should be. Engineering will always be essential as it is the core of VR systems that everything else builds upon, but VR itself presents a fascinating interplay of technology and psychology, and we must understand both to do VR well.

Technically minded people tend to dislike the word “experience” as it is less logical and more subjective in nature. But when asked about their favorite tool or game they often convey emotion as they discuss how, for example, the tool responds. They talk about how it makes them feel often without them consciously realizing they are speaking in the language of emotion. Observe how they act when attempting to navigate through a poorly designed voice-response system for technical support via phone. In their frustration, they very well may give up on the product they are calling about and never purchase from that company again. The experience is important for everything, even for engineers, for it determines our quality of life on a moment-by-moment basis. For VR, the experience is even more critical. To create quality VR, we need to continuously ask ourselves and others questions about how we perceive the VR worlds that we create. Is the experience understandable and enjoyable? Or is it sometimes confusing to use while at other times just all-out frustrating and sickness inducing? If the VR experience is intuitive to understand, easily controlled, and comfortable, then just like anything in life, there will be a sense of mastery and satisfaction with “logical” reasons why it works well. Emotion and cognition are tightly coupled, with cognition more often than not justifying decisions that were actually made emotionally. We must create VR experiences with both emotion and logic.

What This Book Is

This book focuses on **human-centered design**, a design philosophy that puts human needs, capabilities, and behavior first, then designs to accommodate those needs, capabilities, and ways of behaving (Norman, 2013). More specifically, this book focuses on the human elements of VR—how users perceive and intuitively interact with various forms of reality, causes of VR sickness, creating content that is pleasing and useful, and how to design and iterate upon effective VR applications.

Good VR design starts with an understanding of both technology and perception. It requires good communication between human and machine, indicating what interactions are possible, what is currently occurring, and what is about to occur. Human-centered design also comes about through observation, for humans are often unaware of their perceptual processes and methods of interacting (at least for VR that works well). Getting the specification of a VR experience is difficult, and rarely does a VR creator do it well for his first few projects. In fact, even VR experts do not perfectly define the project from the start if they are creating a novel experience. A human-centered design principle, like lean methods, is to avoid completely defining the problem at the start and to iterate upon repeated approximations and modifications through rapid tests of ideas with real users.

Developing intuitive VR should not be driven by software/hardware engineering considerations alone (e.g., we need to do much more than figure out how to efficiently display at the highest resolution available based on the most current hardware). A good portion of this book is devoted to how the human mind works in order to help readers create higher-quality VR applications. VR design, however, goes beyond just technology and psychology; VR is intensely multidisciplinary. VR is an incredibly complex challenge and the study, design, and implementation of high-quality VR requires an understanding of various disciplines, including the behavioral and social sciences, neuroscience, information and computer science, physics, communications, art, and even philosophy. When reflecting back upon VR design and implementation, [Wingrave and LaViola \[2010\]](#) point out: “Practitioners must be carpenters, electricians, engineers, artists, and masters of duct tape and Velcro.” This book takes a broad perspective, applying insights from various disciplines to VR design.

In summary, this book provides basic theory, an overview of various concepts useful for VR, examples that put the theory and concepts into more understandable form, useful guidelines, and a foundation for further exploration and interaction design for virtual worlds that do not yet exist.

What This Book Is Not

There are more questions about VR than there are answers and the intent of this book is not to attempt to provide all of the answers. VR covers a broad range of imaginary spaces broader than the real world; nobody could possibly have all the answers to the real world, so it is unreasonable to expect to find all the answers for virtual worlds. Instead, this book attempts to help readers build and iterate upon creative answers and compelling experiences. Although this book can’t possibly cover all aspects of VR in detail, it does provide an overview of various topics and delves more deeply into those that are most important. References are provided throughout for those wishing to further study concepts of interest.

In some cases, the concepts presented in this book follow well-understood principles or conclusive research (although even conclusive research rarely holds 100% of the time across all conditions). In other cases, concepts are not the “truth” but have been found to be useful in the way we think about design and interaction. Studying theory can be useful, but VR development should always follow pragmatism over theory.

Although there is a brief chapter on getting started (Chapter 36), and there are tips throughout on high- and mid-level implementation concepts, this book is not a step-by-step tutorial of how to implement an example VR system. In fact, this book intentionally contains no code or equations so that all concepts can be understood

by anyone from any discipline (references are provided for more rigorous detail). Although researchers have been experimenting with VR for decades (Section 2.1), VR in no way comes close to reaching its full potential. Not only are there many unknowns, but VR implementation very much depends on the project. For example, a surgical training system is very different from an immersive film.

Who This Book Is For

This book is for the entire team that works on a VR project, not just for those who define themselves as designers. It is intended to act as a foundation for anyone and everyone involved with creating VR experiences. The book also is meant to serve as a bridge between academic research and practical advice to a wide range of individuals who wish to build compelling experiences. This includes designers, managers, programmers, artists, psychologists, engineers, students, educators, and user experience professionals so the entire team can have a common understanding and language of VR. Everyone involved with a VR project should understand at least the basics of perception, VR sickness, interaction, content creation, and iterative design. VR requires specialized experts in various disciplines to each contribute in their own unique way, but we each must also know at least a little about human-centered design in order to effectively communicate with teammates and to integrate the various components together into a seamless, quality experience.

How to Read This Book

Readers may wish to read and use this book differently depending on their background, particular interests, and how they would like to apply it.

Newcomers

Those completely new to VR who want a high-level understanding will most appreciate Part I, Introduction and Background. After reading Part I, the reader may want to skip ahead to Part VII, The Future Starts Now. Once these basics are understood, most of Part IV, Content Creation, should be able to be understood. As the reader learns more about VR, the other parts will become easier to digest.

Teachers

This book will be especially relevant to interdisciplinary VR courses. Teachers will want to choose chapters that are most relevant to the course requirements and student interests. It is highly suggested that any VR course take a project-centered approach. In such a case, the teacher should consider the suggestions below for both students

and practitioners. An outline for starting a first project, albeit slightly different for a class project, is outlined in Chapter 36, Getting Started.

Students

Students will gain a core understanding of VR by first gaining a high-level overview of VR through Part I, Introduction and Background. For those wishing to understand theory, Part II, Perception, and Part III, Adverse Health Effects, will be invaluable. For students working on VR projects, they should also follow the advice below for practitioners.

Practitioners

Practitioners who want to immediately get the most important points that apply to their VR creations will want to start with the practitioner chapters that have a leading star (★) in the table of contents (mostly Parts IV–VI). In particular, they may want to start with the design guidelines chapters at the end of each part, where each part contains multiple chapters on one of the primary topics. Most of the guidelines provide back references to the relevant sections for more detailed information.

VR Experts

VR experts will likely use this book more as a reference so they do not spend time reading material they are already familiar with. The references will also serve VR experts for further investigation. Those experts who primarily work with head-mounted displays may find Part I, Introduction and Background, useful to understand how head-mounted displays fit within the larger scheme of other implementations of VR and augmented reality (AR). Those interested in implementing straightforward applications that have already been built may not find Part II, Perception, useful. However, for those who want to innovate with new forms of VR and interaction, they may find this part useful to understand how we perceive the world in order to help invent novel creations.

Overview of the Seven Parts

Part I, Introduction and Background, provides a background of VR including a brief history of VR, different forms of VR and related technologies, and a broad overview of some of the most important concepts that will be further discussed in later parts.

Part II, Perception, provides a background in perception to educate VR creators on concepts and theories of how we perceive and interact with the world around us.

This part serves as an intellectual framework that will enable the reader to not only implement the ideas discussed in later chapters but more thoroughly understand why some techniques do or do not work, to extend those techniques, to intelligently experiment with new concepts that have a better chance of working without causing human factors issues, and to know when it might be appropriate to break the rules.

Part III, Adverse Health Effects, describes one of the most difficult challenges of VR and helps to reduce the greatest risk to VR succeeding at a massive scale: VR sickness. Whereas it may be impossible to remove 100% of VR sickness for the entire population, there are several ways to dramatically reduce it if we understand the theories of why it occurs. Other adverse health effects such as risk of injury, seizures, and aftereffects are also discussed.

Part IV, Content Creation, discusses high-level concepts for designing/building assets and how subtle design choices can influence user behavior. Examples include story creation, the core experience, environmental design, wayfinding aids, social networking, and porting existing content to VR.

Part V, Interaction, focuses on how to design the way users interact within the scenes they find themselves in. For many applications, we want to engage the user by creating an active experience that consists of more than simply looking around; we want to empower users by enabling them to reach out, touch, and manipulate that world in a way that makes them feel they are a part of the world instead of just a passive observer.

Part VI, Iterative Design, provides an overview of several different methods for creating, experimenting, and improving upon VR designs. Whereas each project may not utilize all methods, it is still good to understand them all to be able to apply them when appropriate. For example, you may not wish to conduct a formal and rigorous scientific user study, but you do want to understand the concepts to minimize mistaken conclusions due to confounding factors.

Part VII, The Future Starts Now, summarizes the book, discusses the current and future state of VR, and provides a brief plan to get started.

PART

INTRODUCTION AND BACKGROUND

What is virtual reality (VR)? What does VR consist of and for what situations is it useful? What is different about VR that gets people so excited? How do developers engage users so that they feel present in a virtual environment? This part of the book answers such questions, and provides a basic background that later chapters build upon. This introduction and background serves as a simple high-level toolbox of options to intelligently choose from, such as different forms of virtual and augmented reality (AR), different hardware options, various methods of presenting information to the senses, and ways to induce presence into the minds of users.

Part I consists of five chapters that cover the basics of VR.

Chapter 1, What Is Virtual Reality?, begins by describing what VR is at a high level and what it is suitable/effective for. This includes descriptions of different forms of communication that are at the heart of what VR is—communication between the user and a system created by the VR designer.

Chapter 2, A History of VR, provides a history of VR starting with stereoscopes created in the 1800s. The concept and implementation of VR is not new.

Chapter 3, An Overview of Various Realities, discusses forms of reality ranging from the real world to augmented reality (AR) to VR. Whereas the focus of this book is on fully immersive VR, this chapter provides context of where VR fits into the overall picture of related technologies. The chapter also gives a high-level description of various forms of input and output hardware options that can be used as part of AR and VR systems.

Chapter 4, Immersion, Presence, and Reality Trade-Offs, discusses the often-used terms of immersion and presence. Readers may be surprised to learn that realism is not necessarily the goal of VR and there are trade-offs for attempting to perfectly simulate reality, even if reality could be perfectly simulated.

Chapter 5, The Basics: Design Guidelines, concludes this introductory part of the book and gives a small number of guidelines for those looking to create VR experiences.

1 What Is Virtual Reality?

1.1

The Definition of Virtual Reality

The term virtual reality (VR) is commonly used by the popular media to describe imaginary worlds that only exist in computers and our minds. However, let us more precisely define the term. [Sherman and Craig \[2003\]](#) point out in their book *Understanding Virtual Reality* that [Webster's New Universal Unabridged Dictionary \[1989\]](#) defines virtual as “being in essence or effect, but not in fact” and reality as “the state or quality of being real. Something that exists independently of ideas concerning it. Something that constitutes a real or actual thing as distinguished from something that is merely apparent.” Thus, virtual reality is a term that contradicts itself—an oxymoron! Fortunately, the website [merriam-webster.com \[Merriam-Webster 2015\]](#) has more recently defined the full term virtual reality to be “an artificial environment which is experienced through sensory stimuli (as sights and sounds) provided by a computer and in which one's actions partially determine what happens in the environment.” In this book, **virtual reality** is defined to be a computer-generated digital environment that can be experienced and interacted with as if that environment were real.

An ideal VR system enables users to physically walk around objects and touch those objects as if they were real. Ivan Sutherland, the creator of one of the world's first VR systems in the 1960s, stated [[Sutherland 1965](#)]: “The ultimate display would, of course, be a room within which the computer can control the existence of matter. A chair displayed in such a room would be good enough to sit in. Handcuffs displayed in such a room would be confining, and a bullet displayed in such a room would be fatal.” We haven't yet come anywhere near Ivan Sutherland's vision (nor do we necessarily want to!) and perhaps we never will. However, there are some quite engaging virtual realities today—many of which are featured throughout this book.

1.2 VR Is Communication

Normally, communication is thought of as interaction between two or more people. This book defines **communication** more abstractly: the transfer of energy between two entities, even if just the cause and effect of one object colliding with another object. Communication can also be between human and technology—an essential component and basis of VR. VR design is concerned with the communication of how the virtual world works, how that world and its objects are controlled, and the relationship between user and content: ideally where users are focused on the experience rather than the technology.

Well-designed VR experiences can be thought of as collaboration between human and machine where both software and hardware work harmoniously together to provide intuitive communication with the human. Developers write complex software to create, if designed well, seemingly simple transfer functions to provide effective interactions and engaging experiences. Communication can be broken down into direct communication and indirect communication as discussed below.

1.2.1 Direct Communication

Direct communication is the direct transfer of energy between two entities with no intermediary and no interpretation attached. In the real world, pure direct communication between entities doesn't represent anything as the purpose is not communication, but it is a side effect. However, in VR, developers insert an artificial intermediary (the VR system that is ideally unperceivable) between the user and carefully controlled sensory stimuli (e.g., shapes, motions, sounds). When the goal is direct communication, VR creators should focus on making the intermediary transparent so users feel like they have direct access to those entities. If that can be achieved, then users will perceive, interpret, and interact with stimuli as if they are directly communicating with the virtual world and its entities.

Direct communication consists of structural communication and visceral communication.

Structural Communication

Structural communication is the physics of the world, not the description or the mathematical representation but the thing-in-itself [Kant 1781]. An example of structural communication is the bouncing of a ball off of the hand. We are always in relationship to objects, which help to define our state; e.g., the shape of our hand around a controller. The world, as well as our own bodies, directly tells us what the structure is through our senses. Although thinking and feeling do not exist within structural

communication, such communication does provide the starting point for perception, interpretation, thinking, and feeling.

In order for our ideas to persist through time, we must put those ideas into structural form, what Norman [2013] calls knowledge in the world. Recorded information and data is the obvious example of structural form, but sometimes less obvious structural forms are the signifiers and constraints (Section 25.1) of interaction. In order to induce experiences into others through VR, we present structural stimuli (e.g., pixels on a display, sound through headphones, or the rumble/vibration of a controller) so the users can sense and interact with our creations.

Visceral Communication

Visceral communication is the language of automatic emotion and primal behavior, not the rational representation of the emotions and behavior (Section 7.7). Visceral communication is always present for humans and is the in-between of structural communication and indirect communication. Presence (Chapter 4) is the act of being fully engaged via direct communication (albeit primarily one way). Examples of visceral communication are the feeling of awe while sitting on a mountaintop, looking down at the earth from space, or being with someone via solid eye contact (whether in the real world or via avatars in VR). The full experience of such visceral communication cannot be put into words, although we often attempt to do so at which point the experience is transformed into indirect communication (e.g., explaining VR to someone is not the same as experiencing VR).

1.2.2 Indirect Communication

Indirect communication connects two or more entities through some intermediary. The intermediary need not be physical; in fact, the intermediary is often our mind's interpretation that sits between the world and behavior/action. Once we interpret and give something meaning, then we have transformed the direct communication into indirect communication. Indirect communication includes what we normally think of as language, such as spoken and written language, as well as sign languages and our internal thoughts (i.e., communicating with oneself). Indirect communication consists of talking, understanding, creating stories/histories, giving meaning, comparing, negating, fantasizing, lying, and romancing. These are not part of the objective real world but are what we humans describe and create with our minds. Indirect VR communication includes the user's internal mental model (Section 7.8) of how the VR world works (e.g., the interpretation of what is occurring in the VR world), and indirect interactions (Section 28.4) such as moving a slider that changes an object property,

speech recognition that changes the system's state, and indirect gestures that act as a sign language to the computer.

1.3 What Is VR Good For?

The recent surge in media coverage about VR has inspired the public to become quite excited about its potential. This coverage has focused on the entertainment industry, specifically video games and immersive film. VR is a great fit for the entertainment industry and will certainly be the driving force behind VR in the short term. However, what is VR good for beyond entertainment? It turns out VR can have enormous benefit over a wide range of verticals. VR has been successfully deployed in various industries for many years now. Successful applications include oil and gas exploration, scientific visualization, architecture, flight simulation, therapy, military training, theme-park entertainment, engineering analysis, and design review. Using VR in such situations has successfully revealed costly design mistakes before manufacturing anything, reduced time to market by speeding up iterative processes, provided safe learning environments that would otherwise be dangerous, reduced PTSD by gradually increasing exposure to feared stimuli, and helped to visualize large datasets that would be difficult to comprehend with traditional systems.

Unfortunately, to date, VR has largely been limited to well-funded academic and corporate research labs to which few have access. That is all changing with consumer-priced systems now becoming widely available. The VR market is expected to first explode in the entertainment industry but will soon expand significantly in other industries. Education, telepresence, and professional training will likely be the next industries to take advantage of VR in a big way.

Whatever the industry, VR is largely about providing understanding—whether that is understanding an entertaining story, learning an abstract concept, or practicing a real skill. Actively using more of the human sensory capability and motor skills has been known to increase understanding/learning for some time [Dale 1969]. This is in part due to the increased sensory bandwidth between human and information, but there is much more to understanding. Actively participating in an action, making concepts intuitive, encouraging motivation through engaging experiences, and the thoughts inside one's head all contribute to understanding. This book focuses on how to design such concepts into VR experiences.

Figure 1.1 shows Edgar Dale's Cone of Experience [Dale 1969]. As can be seen in the figure, direct purposeful experiences provide the best basis for understanding. As Confucius stated, "I see and I forget. I hear and I remember. I do and I understand." Note this diagram does not suggest direct purposeful experiences should be

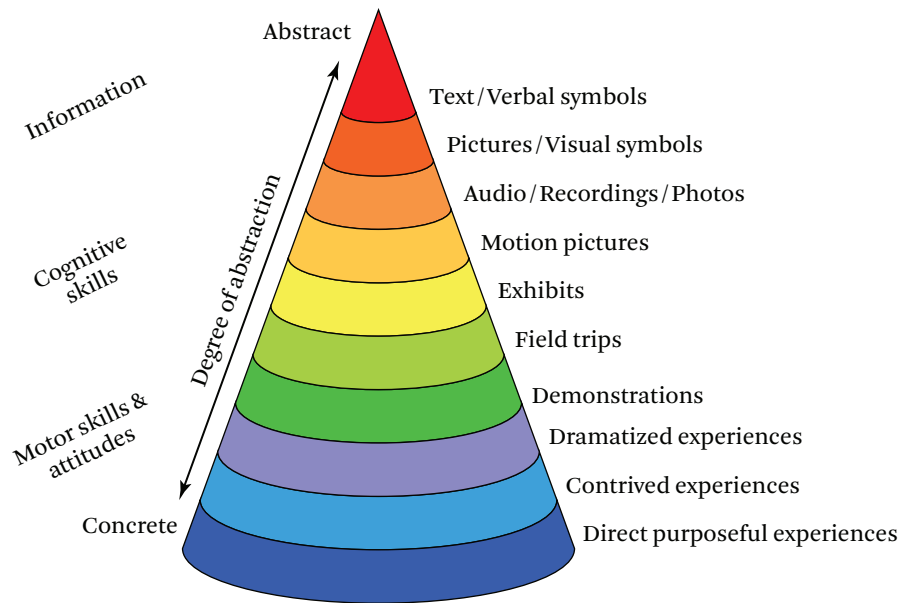


Figure 1.1 The Cone of Experience. VR uses many levels of abstraction. (Adapted from Dale [1969])

the only method of learning, but instead describes the progression of learning experience. Adding other indirect information within direct purposeful VR experiences can further enhance understanding. For example, embedding abstract information such as text, symbols, and multimedia directly into the scene and onto virtual objects can lead to more efficient understanding than what can be achieved in the real world.

A History of VR

When anything new comes along, everyone, like a child discovering the world, thinks that they've invented it, but you scratch a little and you find a caveman scratching on a wall is creating virtual reality in a sense. What is new here is that more sophisticated instruments give you the power to do it more easily.

—Morton Heilig [[Hamit 1993](#)]

Precursors to what we think of today as VR go back as far as humans have had imaginations and the ability to communicate through the spoken word and cave drawings (what could be called analog VR). The Egyptians, Chaldeans, Jews, Romans, and Greeks used magical illusions to entertain and control the masses. In the Middle Ages, magicians used smoke and concave mirrors to produce faint ghost and demon illusions to gull naive apprentices as well as larger audiences [[Hopkins 2013](#)]. Although the words and implementation have changed over the centuries, the core goals of creating the illusion of conveying that which is not actually present and capturing our imaginations remain the same.

2.1 The 1800s

The static version of today's stereoscopic 3D TVs is called a stereoscope and was invented before photography in 1832 by Sir Charles Wheatstone [[Gregory 1997](#)]. As shown in Figure 2.2, the device used mirrors angled at 45° to reflect images into the eye from the left and right side.

David Brewster, who earlier invented the kaleidoscope, used lenses to make a smaller consumer-friendly hand-held stereoscope (Figure 2.3). His stereoscope was demonstrated at the 1851 Exhibition at the Crystal Palace where Queen Victoria found it quite pleasing. Later the poet Oliver Wendell Holmes stated “. . . is a surprise such as no painting ever produced. The mind feels its way into the very depths of the picture.” [[Zone 2007](#)]. By 1856 Brewster estimated over a half million stereoscopes had been sold [[Brewster 1856](#)]. This first 3D craze included various forms of the stereoscope,

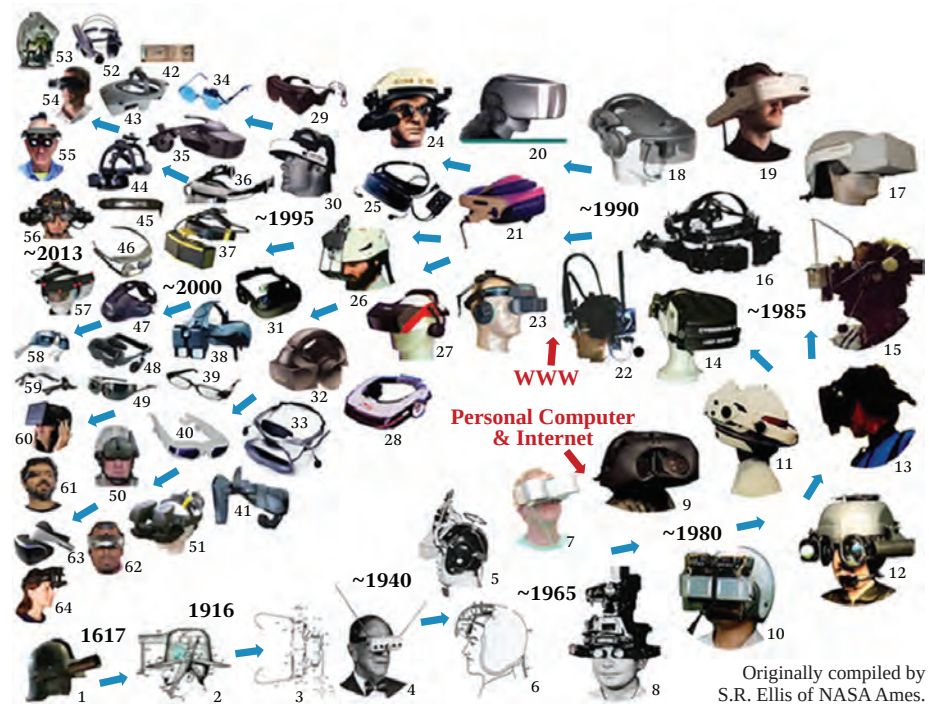


Figure 2.1 Some head-mounted displays and viewers over time. (Based on Ellis [2014])

including self-assembled cardboard versions with moving images controlled by the hand in 1860 [Zone 2007]. One company alone sold a million stereoscopic views in 1862. Brewster's design is conceptually the same as the 20th century View-Master and today's Google Cardboard. In the case of Google Cardboard and similar phone-based VR systems, a cellular phone is used to display the images in place of the actual physical images themselves.

Many years later, a 360° VR-type display known as the Haunted Swing was shown at the '95 Midwinter Fair in San Francisco, and is still one of the most compelling technical demonstrations of an illusion to this day. The demo consisted of a room and large swing that held approximately 40 people. After the audience seated themselves, the swing was put in motion and as the swing oscillated, users felt motion similar to being in an elevator while they involuntarily clutched their seats. In fact, the swing hardly moved at all, but the surrounding room moved substantially, resulting in the sense of self-motion (Section 9.3.10) and motion sickness (Chapter 12). The date was not 1995, but was 1895 [Wood 1895].



Figure 2.2 Charles Wheatstone's stereoscope.



Figure 2.3 A Brewster stereoscope from 1860. (Courtesy of The National Media Museum/Science & Society Picture Library, United Kingdom)

It was also in 1895 that film began to go mainstream; and when the audience saw a virtual train coming at them through the screen in the short film “L’Arrivée d’un train en gare de La Ciotat,” some people reportedly screamed and ran to the back of the room. Although the screaming and running away is more rumor than verified reports, there was certainly hype, excitement, and fear about the new artistic medium, perhaps similar to what is happening with VR today.

2.2 The 1900s

VR-related innovation continued in the 1900s that moved beyond simply presenting visual images. New interaction concepts started to emerge that might be considered novel for even today’s VR systems. For example, Figure 2.4 shows what is a head-worn gun pointing and firing device patented by Albert Pratt in 1916 [Pratt 1916]. No hand tracking is required to fire this device as the interface consists of a tube the user blows through.

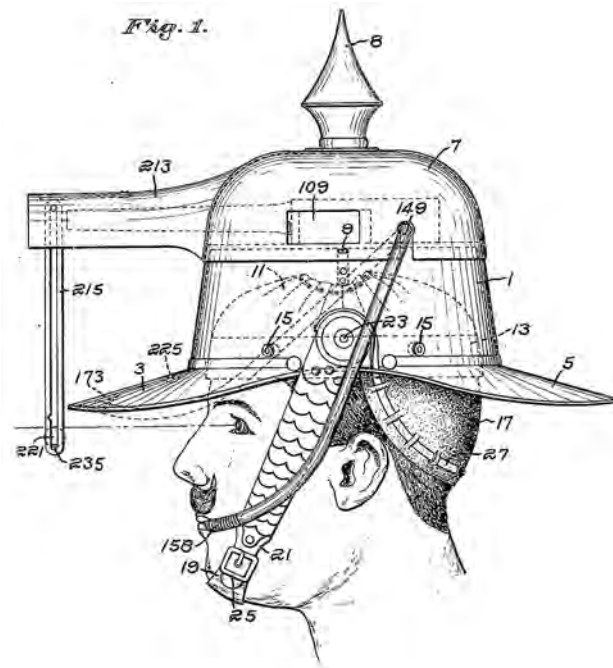


Figure 2.4 Albert Pratt’s head-mounted targeting and gun-firing interface. (From Pratt [1916])

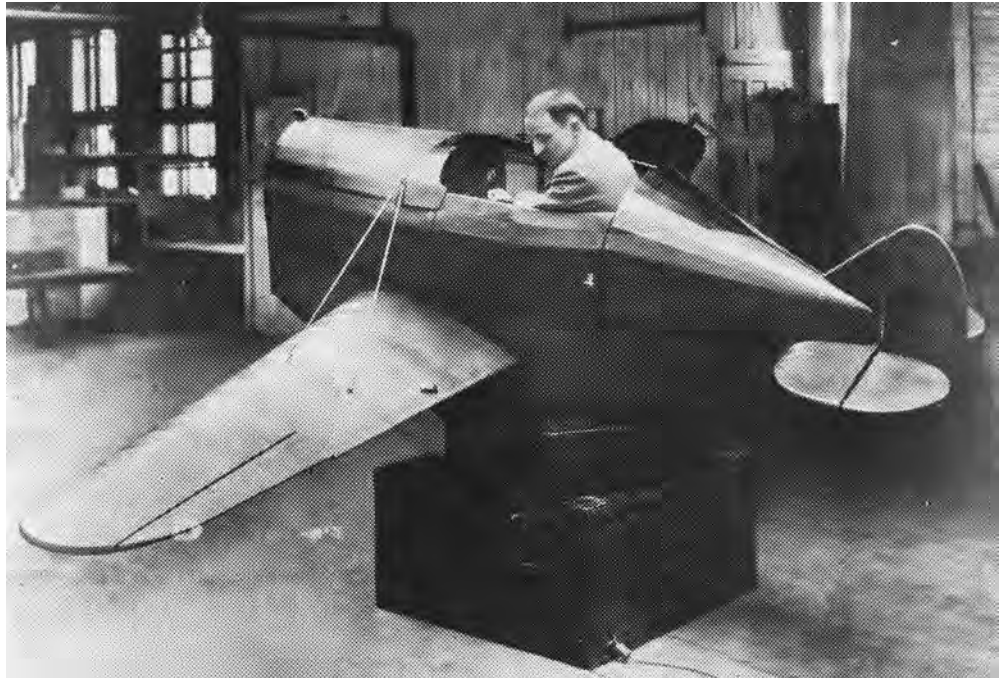


Figure 2.5 Edwin A. Link and the first flight simulator in 1928. (Courtesy of Edwin A. Link and Marion Clayton Link Collections, Binghamton University Libraries' Special Collections and University Archives, Binghamton University)

A little over a decade after Pratt received his weapon patent, Edwin Link developed the first simple mechanical flight simulator, a fuselage-like device with a cockpit and controls that produced the motions and sensations of flying (Figure 2.5). Surprisingly, his intended client—the military—was not initially interested, so he pivoted to selling to amusement parks. By 1935, the Army Air Corps ordered six systems and by the end of World War II, Link had sold 10,000 systems. Link trainers eventually evolved into astronaut training systems and advanced flight simulators complete with motion platform and real-time computer-generated imagery, and today is Link Simulation & Training, a division of L-3 Communications. Since 1991, the Link Foundation Advanced Simulation and Training Fellowship Program has funded many graduate students in their pursuits of improving upon VR systems, including work in computer graphics, latency, spatialized audio, avatars, and haptics [Link 2015].

As early 20th century technologies started to be built, science fiction and questions inquiring about what makes reality started to become popular. In 1935, for example,



Figure 2.6 *Pygmalion's Spectacles* is perhaps the first science fiction story written about an alternate world that is perceived through eyeglasses and other sensory equipment. (From Weinbaum [1935])

science fiction readers got excited about a surprisingly similar future that we now aspire to with head-mounted displays and other equipment through the book *Pygmalion's Spectacles* (Figure 2.6). The story opens with the words “But what is reality?” written by a professor friend of George Berkeley, the Father of Idealism (the philosophy that reality is mentally constructed) and for whom the University of California, Berkeley, is named. The professor then explains a set of eyeglasses along with other equipment that replaces real-world stimuli with artificial stimuli. The demo consists of a quite compelling interactive and immersive world where “The story is all about you, and you are in it” through vision sound, taste, smell, and even touch. One of the virtual characters calls the world Paracosma—Greek for “land beyond-the-world.” The demo is so good that the main character, although skeptical at first, becomes convinced that it is no longer illusion, but reality itself. Today there are countless books ranging from philosophy to science fiction that discuss the illusion of reality.

Perhaps inspired by *Pygmalion's Spectacles*, McCollum patented the first stereoscopic television glasses in 1945. Unfortunately, there is no record of the device ever actually having been built.

In the 1950s Morton Heilig designed both a head-mounted display and a world-fixed display. The head-mounted display (HMD) patent [Heilig 1960] shown in Figure 2.7 claims lenses that enable a 140° horizontal and vertical field of view, stereo

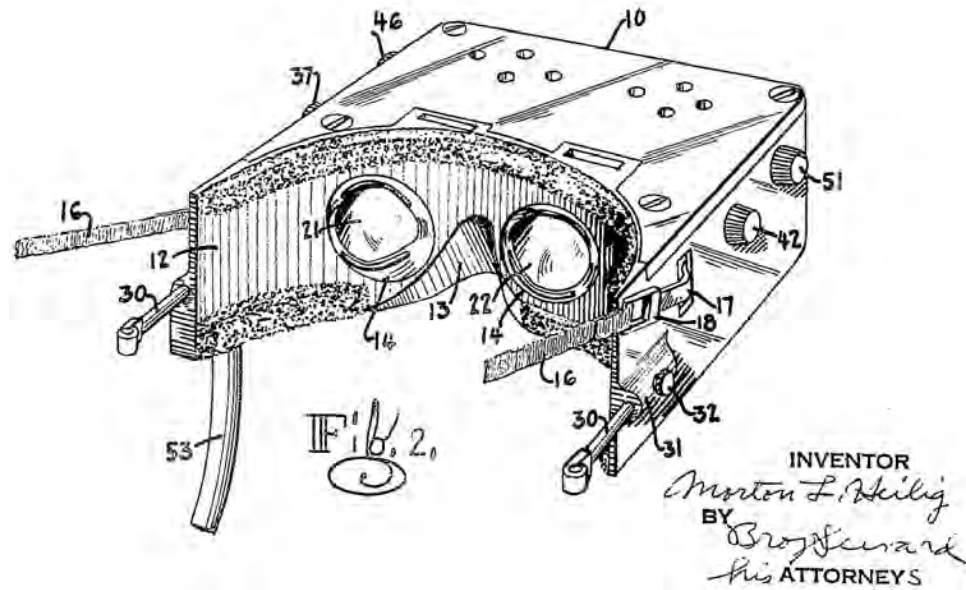


Figure 2.7 A drawing from Heilig's 1960 Stereoscopic Television Apparatus patent. (From Heilig [1960])

earphones, and air discharge nozzles that provide a sense of breezes at different temperatures as well as scent. He called his world-fixed display the Sensorama. As can be seen in Figure 2.8, the Sensorama was created for immersive film and it provided stereoscopic color views with a wide field of view, stereo sounds, seat tilting, vibrations, smell, and wind [Heilig 1992].

In 1961, Philco Corporation engineers built the first actual working tracked HMD that included head tracking (Figure 2.9). As the user moved his head, a camera in a different room moved so the user could see as if he were at the other location. This was the world's first working telepresence system.

One year later IBM was awarded a patent for the first glove input device (Figure 2.10). This glove was designed as a comfortable alternative to keyboard entry, and a sensor for each finger could recognize multiple finger positions. Four possible positions for each finger with a glove in each hand resulted in 1,048,575 possible input combinations. Glove input, albeit with very different implementations, later became a common VR input device in the 1990s.

Starting in 1965, Tom Furness and others at the Wright-Patterson Air Force Base worked on visually coupled systems for pilots that consisted of head-mounted displays



Figure 2.8 Morton Heilig’s Sensorama created the experience of being fully immersed in film.
(Courtesy of © Morton Heilig Legacy)

(Figure 2.11, left). While Furness was developing head-mounted displays at Wright-Patterson Air Force Base, Ivan Sutherland was doing similar work at Harvard and the University of Utah. Sutherland is known as the first to demonstrate a head-mounted display that used head tracking and computer-generated imagery [Oakes 2007]. The system was called the Sword of Damocles (Figure 2.11, right), named after the story of King Damocles who, with a sword hanging above his head by a single hair of a horse’s tail, was in constant peril. The story is a metaphor that can be applied to VR technology: (1) with great power comes great responsibility; (2) precarious situations give a sense of foreboding; and (3) as stated by Shakespeare [1598] in *Henry IV*, “uneasy lies the head that wears a crown.” All these seem very relevant for both VR developers and VR users even today.



Figure 2.9 The Philco Headsight from 1961. (From [Comeau and Brian \[1961\]](#))

Dr. Frederick P. Brooks, Jr., inspired by Ivan Sutherland's vision of the Ultimate Display [[Sutherland 1965](#)], established a new research program in interactive graphics at the University of North Carolina at Chapel Hill, with the initial focus being on molecular graphics. This not only resulted in a visual interaction with simulated molecules but also included force feedback where the docking of simulated molecules could be felt. Figure 2.12 shows the resulting Grope-III system that Dr. Brooks and his team built. UNC has since focused on building various VR systems and applications with the intent to help practitioners solve real problems ranging from architectural visualization to surgical simulation.

In 1982, Atari Research, led by legendary computer scientist Alan Kay, was formed to explore the future of entertainment. The Atari research team, which included Scott Fisher, Jaron Lanier, Thomas Zimmerman, Scott Foster, and Beth Wenzel, brainstormed novel ways of interacting with computers and designed technologies that would soon be essential for commercializing VR systems.

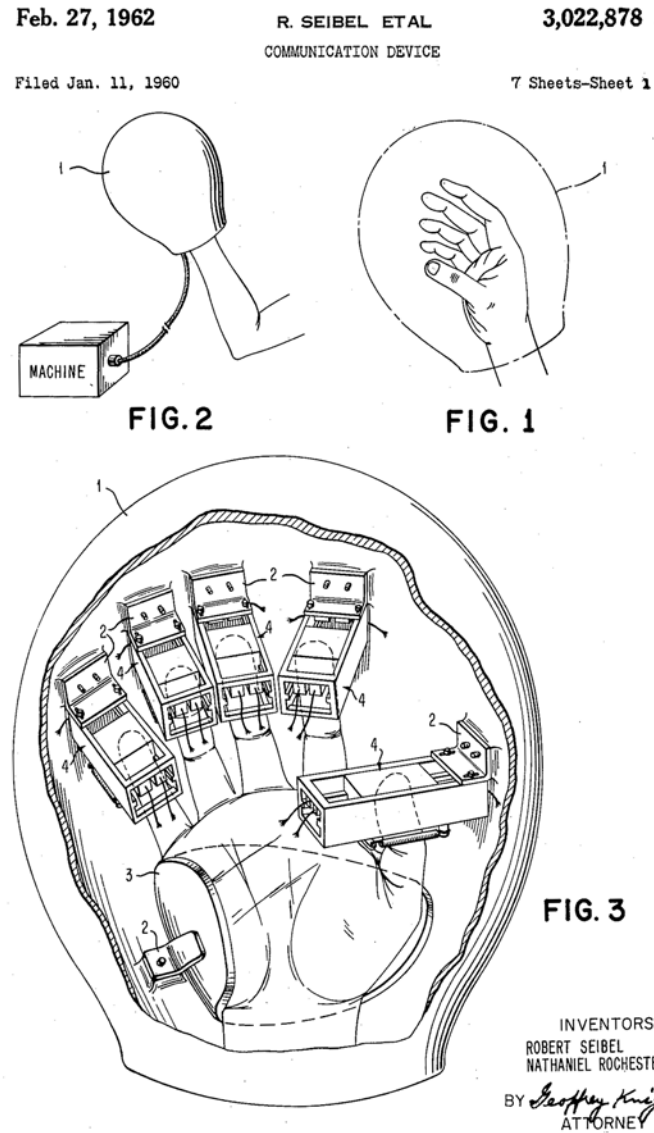


Figure 2.10 An image from IBM's 1962 glove patent. (From [Rochester and Seibel \[1962\]](#))



Figure 2.11 The Wright-Patterson Air Force Base head-mounted display from 1967 (courtesy of Tom Furness) and the Sword of Damocles [[Sutherland 1968](#)]

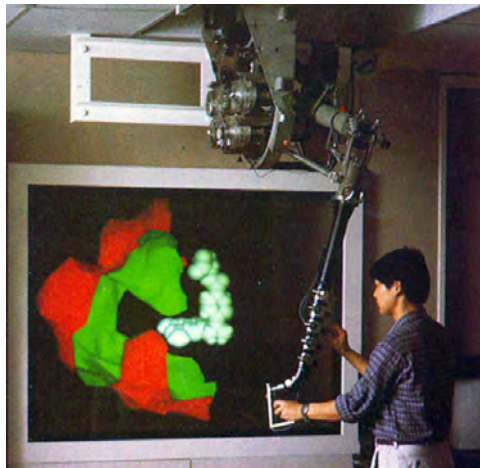


Figure 2.12 The Grope-III haptic display for molecular docking. (From [Brooks et al. \[1990\]](#))



Figure 2.13 The NASA VIEW System. (Courtesy of NASA/S.S. Fisher, W. Sisler, 1988)

In 1985, Scott Fisher, now at NASA Ames, along with other NASA researchers developed the first commercially viable, stereoscopic head-tracked HMD with a wide field of view, called the Virtual Visual Environment Display (VIVED). It was based on a scuba diver's face mask with the displays provided by two Citizen Pocket TVs (Scott Fisher, personal communication, Aug 25, 2015). Scott Foster and Beth Wenzel built a system called the Convolvotron that provided localized 3D sounds. The VR system was unprecedented as the HMD could be produced at a relatively affordable price, and as a result the VR industry was born. Figure 2.13 shows a later system called the VIEW (Virtual Interface Environment Workstation) system.

Jaron Lanier and Thomas Zimmerman left Atari in 1985 to start VPL Research (VPL stands for Visual Programming Language) where they built commercial VR gloves, head-mounted displays, and software. During this time Jaron coined the term “virtual reality.” In addition to building and selling head-mounted displays, VPL built the Dataglove specified by NASA—a VR glove with optical flex sensors to measure finger bending and tactile vibrator feedback [Zimmerman et al. 1987].

VR exploded in the 1990s with various companies focusing mostly on the professional research market and location-based entertainment. Examples of the more well-known newly formed VR companies were Virtuality, Division, and Fakespace. Existing companies such as Sega, Disney, and General Motors, as well as numerous universities and the military, also started to more extensively experiment with VR technologies. Movies were made, numerous books were written, journals emerged, and conferences formed—all focused exclusively on VR. In 1993, *Wired* magazine predicted that within five years more than one in ten people would wear HMDs while

traveling in buses, trains, and planes [Negroponte 1993]. In 1995, the *New York Times* reported that Virtuality Managing Director Jonathan Waldern predicted the VR market to reach \$4 billion by 1998 [Bailey 1995]. It seemed VR was about to change the world and there was nothing that could stop it. Unfortunately, technology could not support the promises of VR. In 1996, the VR industry peaked and then started to slowly contract with most VR companies, including Virtuality, going out of business by 1998.

2.3 The 2000s

The first decade of the 21st century is known as the “VR winter.” Although there was little mainstream media attention given to VR from 2000 to 2012, VR research continued in depth at corporate, government, academic, and military research laboratories around the world. The VR community started to turn toward human-centered design with an emphasis on user studies, and it became difficult to get a VR paper accepted at a conference without including some form of formal evaluation. Thousands of VR-related research papers from this era contain a wealth of knowledge that today is unfortunately largely unknown and ignored by those new to VR.

A wide field of view was a major missing component of consumer HMDs in the 1990s, and without it users were just not getting the “magic” feeling of presence (Mark Bolas, personal communication, June 13, 2015). In 2006, Mark Bolas of USC’s MxR Lab and Ian McDowall of Fakespace Labs created a 150° field of view HMD called the Wide5, which the lab later used to study the effects of field of view on the user experience and behavior. For example, users can more accurately judge distances when walking to a target when they have a larger field of view [Jones et al. 2012]. The team’s research led to the low-cost Field of View To Go (FOV2GO), which was shown at the IEEE VR 2012 conference in Orange County, California, where the device won the Best Demo Award and was part of the MxR Lab’s Open-source project that is the precursor to most of today’s consumer HMDs. Around that time, a member of that lab named Palmer Luckey started sharing his prototype on Meant to be Seen (mtbs3D.com) where he was a forum moderator and where he first met John Carmack (now CTO of Oculus VR) and formed Oculus VR. Shortly after that he left the lab and launched the Oculus Rift Kickstarter. The hacker community and media latched onto VR once again. Companies ranging from start-ups to the Fortune 500 began to see the value of VR and started providing resources for VR development, including Facebook, which acquired Oculus VR in 2014 for \$2 billion. The new era of VR was born.

PART

INTERACTION

For this book, interaction is defined to be a subset of general communication (Section 1.2). **Interaction** is the communication that occurs between a user and the VR application that is mediated through the use of input and output devices. An **interface** is the VR system side of the interaction that exists whether a user is interacting or not.

Interaction in VR at first seems obvious—simply interact with the virtual world in a similar way that we interact with the real world. Unfortunately, natural real-world interfaces (and Hollywood interfaces) often do not work well within VR as one would expect. This is not only because virtual worlds do not model the real world as closely as we would like, but even if they did, abstract interfaces are often superior (e.g., if one wanted to look up information in a virtual world, one would not want to have to travel to a virtual library to find a virtual book). Whereas some of the basic concepts from traditional desktop interfaces can be used within immersive environments, there are few similarities between the two. Creating quality interactions is currently one of the greatest challenges for VR.

Well-implemented interaction techniques enable high levels of performance and comfort while diminishing the impact of human and hardware limitations. It is the role of the interaction designer to make complex interactions intuitive and effective.

Part V consists of five chapters that provide an overview of the most important aspects of VR interaction design.

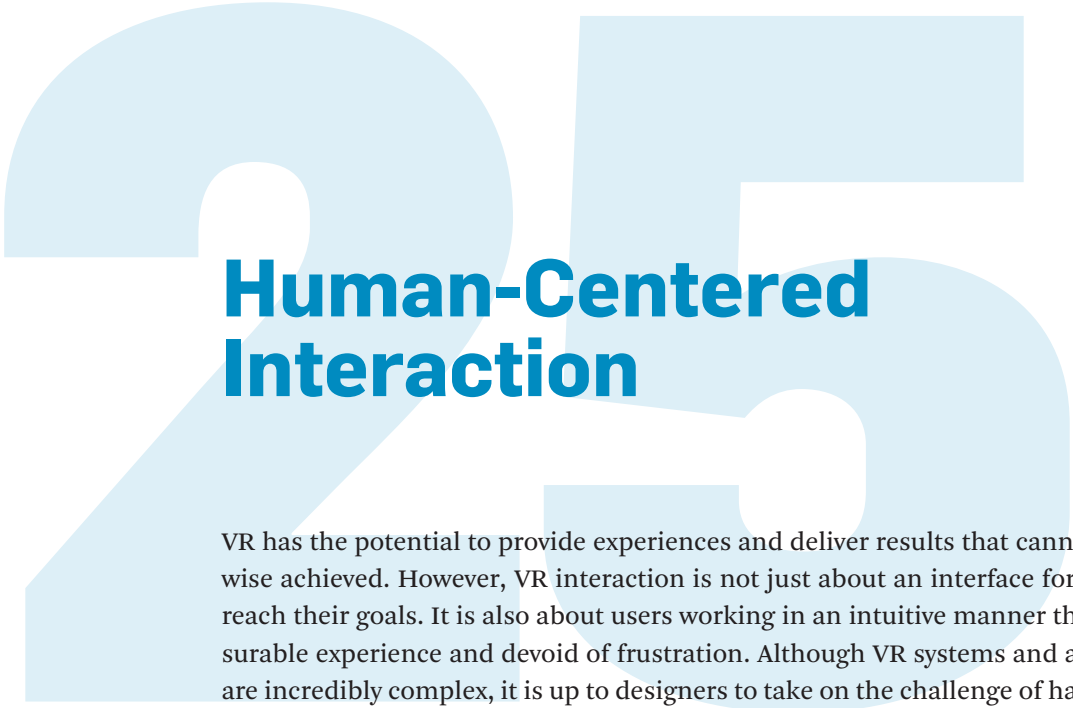
Chapter 25, Human-Centered Interaction, reviews core concepts for general human-centered interaction: intuitiveness, Norman's principles of interaction design, direct vs. indirect interaction, the cycle of interaction, and the human hands.

Chapter 26, VR Interaction, discusses some of the most important distinctions especially relevant to VR interaction, such as interaction fidelity, proprioceptive and egocentric interaction, reference frames, multimodal interaction, and unique challenges of VR interaction.

Chapter 27, Input Devices, provides an overview of input device characteristics and classes of various input devices. No single input device is appropriate for all VR applications, and choosing the most appropriate input device for the intended experience is essential for optimizing interactions.

Chapter 28, Interaction Patterns and Techniques, describes several interaction patterns and examples of various interaction techniques. An interaction pattern is a generalized high-level interaction concept that can be used over and over again across different applications to achieve common user goals. An interaction technique is a more specific implementation of an interaction pattern.

Chapter 29, Interaction: Design Guidelines, summarizes the previous four chapters and lists a number of actionable guidelines for developing interaction techniques that largely depend on application needs.



Human-Centered Interaction

VR has the potential to provide experiences and deliver results that cannot be otherwise achieved. However, VR interaction is not just about an interface for the user to reach their goals. It is also about users working in an intuitive manner that is a pleasurable experience and devoid of frustration. Although VR systems and applications are incredibly complex, it is up to designers to take on the challenge of having the VR application effectively communicate to users how the virtual world and its tools work so that those users can achieve their goals in an elegant manner.

Perhaps the most important part of VR interaction is the person doing the interacting. Human-centered interaction design focuses on the human side of communication between user and machine—the interface from the user's point of view. Quality interactions enhance user understanding of what has just occurred, what is happening, what can be done, and how to do it. In the best case, not only will goals and needs be efficiently achieved, but the experiences will be engaging and enjoyable.

This section describes several general human-centered design concepts that are essential for interaction designers to consider when designing VR interactions.

25.1 Intuitiveness

Mental models (Section 7.8) of how a virtual world works are almost always a simplified version of some complexity. When interacting, there is no need for users to understand the underlying algorithms—just the high-level relationship between objects, actions, and outcomes. Whether a VR interface attempts to be realistic or not, it should be intuitive. An intuitive interface is an interface that can be quickly understood, accurately predicted, and easily used. Intuitiveness is in the mind of the user, but the designer can help form this intuitiveness by conveying through the world and interface itself concepts that support the creation of a mental model.

An **interaction metaphor** is an interaction concept that exploits specific knowledge that users already have of other domains. Interaction metaphors help users quickly develop a mental model of how an interaction works. For example, VR users typically think of themselves as “walking” through an environment, even though in most implementations they are not physically moving their feet (e.g., they may be controlling the walking with a hand-held controller). For such an interface, users assume they are traveling at a set height above a surface, and if that height changes, then the interaction does not fit the mental model and the user will likely become confused.

Mental models can be made more consistent across users by providing manuals, communicating with other users, and from the virtual world itself. For fully immersive VR, users are cut off from the real world and there is no guarantee they will look at a manual or talk with others. Thus, the virtual world should be sufficient in projecting appropriate information needed to create a consistent conceptual model of how things work in the mind of each user, without requiring external explanation. Otherwise, problems will occur when an application does not fit with the user’s mental model. It should not be assumed that an expert will always be available to directly explain how an interface works, answer questions, or correct mistakes. Too often, a VR creator expects the user’s model to be identical to what she thinks she has designed, but unfortunately this is rarely the case since creators and users think differently (as they should). Tutorials that cannot be completed until the user has indicated a clear understanding and effective interaction are a great method of inducing mental models into the minds of users.

25.2 Norman’s Principles of Interaction Design

When users interact in VR, they need to figure out how to work the system. **Discoverability** is exploring what something does, how it works, and what operations are possible [Norman 2013]. Discoverability is especially important for fully immersive VR because the user is blind and deaf to the real world, cut off from those real-world humans who want to help. Essential tools can lead the mind into discovering how an interface works through consistent affordances, unambiguous signifiers, constraints to guide actions and ease interpretation, immediate and useful feedback, and obvious and understandable mappings. These principles as defined by Norman are summarized below along with how they relate to VR and are a good starting point when designing and refining interactions.

25.2.1 Affordances

Affordances define what actions are possible and how something can be interacted with by a user. We are used to thinking that properties are associated with objects, but

an affordance is not a property; an affordance is a *relationship* between the capabilities of a user and the properties of a thing. Interface elements afford interaction, such as a virtual hand affords selection. An affordance between an object and one user may be different between that object and another user. Light switches on a wall offer the ability to control the lighting in a room, but only for those who are able to reach the switches. Some objects in virtual environments afford selecting, moving, controlling, etc. Good interaction design focuses on creating appropriate affordances to make desired actions easily doable with the technology used (e.g., a tracking system that is able to track the hand near the point of the light switch) and by the intended user base (e.g., the designer may intentionally place a light switch to not be reachable by a certain class of users in order to encourage collaboration).

25.2.2 Signifiers

Some affordances are perceivable, others are not. To be effective, an affordance should be perceivable. A **signifier** is any perceivable indicator (a signal) that communicates appropriate purpose, structure, operation, and behavior of an object to a user. A good signifier informs a user what is possible before she interacts with its corresponding affordance.

Examples of signifiers are signs, labels, or images placed in the environment indicating what is to be acted upon, which direction to gesture, or where to navigate toward. Other signifiers directly represent an affordance, such as the handle on a door or the visual and/or physical feel of a button on a controller. A misleading signifier can be ambiguous or not represent an affordance—something may look like a drawer to be opened when in fact it cannot be opened. Such a false signifier is usually accidental or not yet implemented. But a misleading signifier can also be purposeful—such as to motivate users to find a key in order to turn the non-accessible drawer into something that can be opened. In such cases, the content creator should be aware of such anti-signifiers and be careful not to frustrate users.

Signifiers are most often intentional, but, as mentioned above, they may also be accidental. An example of an intentional signifier is a sign giving directions. In the real world, an example of an accidental and unintentional (but useful) signifier is garbage on a beach representing unhealthy conditions. At first thought, we might think signifiers are only intentionally created in VR, for the VR creator created everything from that which does not actually exist. However, this is not always the case. An unintended VR signifier might be an object that looks like it is designed to be picked up to be placed into a puzzle, but it can also be perceived as an object that can be picked up and thrown (a common occurrence much to the frustration of content creators). Or an unintended signifier in a social VR experience might be a gathering of users at an

area of interest, signifying to others to navigate to that location to investigate what is happening and what affordance might be available there.

Signifiers might not be attached to a specific object. Signifiers can be general information. Conveying what interaction mode a user is currently using can help prevent confusion.

Regardless of how and why signifiers are created, signifiers are important for communicating to users whether or not action is possible, and what those actions are. Good VR design ensures signifiers are effectively discoverable through signifiers that are well communicated and intelligible.

25.2.3 Constraints

Interaction constraints are limitations of actions and behaviors. Such constraints include logical, semantic, and cultural limitations to guide actions and ease interpretation. With that being said, this section focuses on the physical and mathematical constraints as such constraints most directly apply to VR interactions. For an overview of general project constraints, see Section 31.10.

Proper use of constraints can limit possible actions, which makes interaction design feasible and can simplify interaction while improving accuracy, precision, and user efficiency [Bowman et al. 2004]. A commonly used method for simplifying VR interactions is to constrain interfaces to only work in a limited number of dimensions. The **degrees of freedom** (DoF) for an entity are the number of independent dimensions available for the motion of that entity (also see Section 27.1.2). The DoF of an interface can be constrained by the physical limitations of an input device (e.g., a physical dial has one DoF, a joystick has two DoF), the possible motion of a virtual object (e.g., a slider on a panel is constrained to move along a single axis to control a single value), or travel that is constrained to the ground, enabling one to more easily navigate through a scene.

Physical limitations constrain possible actions and these limitations can also be simulated in software. In addition to being useful, constraints can also add more realism. For example, a virtual hand can be stopped even when there is nothing physically stopping a user's real hand from going through the object (although this results in visual-physical conflict; Section 26.8). However, physics should not always necessarily be used as physics can sometimes make interactions more difficult. For example, it can be useful to leave virtual tools hanging in the air when not using them.

Interaction constraints are more effective and useful if appropriate signifiers make them easy to perceive and interpret, so users can plan appropriately before taking action. Without effective signifiers, users might effectively be constrained because they are not able to determine what actions are possible. Consistency of constraints

can also be useful as learning can be transferred across tasks. People are highly resistant to change; if a new way of doing things is only slightly better than the old, then it is better to be consistent. For experts, providing the ability to remove constraints can be useful in some situations. For example, advanced flying techniques might be enabled after users have proved they can efficiently maneuver by being constrained to the ground.

25.2.4 Feedback

Feedback communicates to the user the results of an action or the status of a task, helps to aid understanding of the state of the thing being interacted with, and helps to drive future action. In VR, timely feedback is essential. Something as simple as moving the head requires immediate visual feedback, otherwise the illusion of a stable world is broken and a break-in-presence results (and, even worse, motion sickness). Input devices capture users' physical motion that is then transformed into visual, auditory, and haptic feedback. At the same time, feedback internal to the body is generated from within—proprioceptive feedback that enables one to feel the position and motion of the limbs and body. Unfortunately, it is quite difficult for VR to provide all possible types of feedback. Haptic feedback is especially difficult to implement in a way similar to how real forces occur in the real world. Section 26.8 discusses how sensory substitution can be used in place of strong haptic cues. For example, objects can make a sound, become highlighted, or vibrate a hand-held controller when a hand collides with them or they have been selected.

Feedback is essential for interaction, but not when it gets in the way of interaction. Too much feedback can overwhelm the senses, resulting in cluttered perception and understanding. Feedback should be prioritized so less important information is presented in an unobtrusive manner and essential information always captures attention. Instead of putting information on a heads-up display in the head reference frame, place it near the waist or toward the ground in the torso reference frame so the information is easily accessed when needed but not in the way otherwise. If information must always be visible with a heads-up display, then only provide the most essential minimal information as anything directly in front of a user reduces awareness of the virtual world. Too many audio beeps or, worse, overlapping audio announcements, can cause users to ignore all of them or even cause them to be indecipherable even if users wanted to listen to them. Such clutter not only can result in unusable applications, but can be annoying (think of backseat drivers!) and is inappropriate. In many cases, users should have the option to turn off or turn down feedback that is not important to them.

25.2.5 Mappings

A **mapping** is a relationship between two or more things. The relationship between a control and its results is easiest to learn where there is an obvious understandable mapping between controls, action, and the intended result. Mappings are useful even when the person is not directly holding the thing being manipulated, for example, when using a screwdriver to lift a lever up that cannot be directly reached.

Mappings from hardware to interaction techniques defined by software are especially important for VR. Often a device will have a natural mapping to one technique but poor mapping to another technique. For example, hand-tracked devices work well for a virtual hand technique and for pointing (Section 28.1) but not as well for a driving simulator where a physical steering wheel would be more appropriate.

Compliance

Compliance is the matching of sensory feedback with input devices across time and space. Maintaining compliance improves user performance and satisfaction. Compliance results in perceptual binding (Section 7.2.1) so that interaction feels as if one is interacting with a single coherent object. Visual-vestibular compliance is especially important for reducing motion sickness (Section 12.3.1). Compliance can be divided into spatial compliance and temporal compliance [Bowman et al. 2004] as described below.

Spatial compliance. Direct spatial mappings lead immediately to understanding. For example, we intuitively understand that once we have grabbed an object, to move the object up, we simply move the hand holding the object up. **Spatial compliance** consists of position compliance, directional compliance, and nulling compliance.

Position compliance is the co-location of sensory feedback with the input device position. An example of spatial compliance is when the proprioceptive sense of where the hand is matches the visual sense of where the hand is. Not all interaction techniques require position compliance, but position compliance results in direct intuitive interaction and should be used whenever appropriate. Labels placed at the location where physical controls are located on a hand-held device is an example of position compliance. Position compliance is important for tracked physical devices that the user can pick up and/or interact with. Consider users who have just placed an HMD on their head but have not yet picked up the hand controllers (or who have previously set the controller down); the user must be able to see the controller in the correct location in order to pick it up.

Directional compliance is the most important of the three spatial compliances. **Directional compliance** states virtual objects should move and rotate in the same

direction as the manipulated input device. This results in correspondence of what is seen and what is felt by the body, resulting in more direct interaction. This enables the user to effectively anticipate motion in response to physical input and therefore plan and execute appropriately.

An example of directional compliance is the mapping from a mouse to a cursor on the screen. Even though a mouse is spatially dislocated from the screen (i.e., it lacks position compliance), the hand/mouse movement results in an immediate and isomorphic movement of the cursor on the screen, which makes the user feel as if he is directly moving the cursor itself [van der Veer and del C. P. Melguizo 2002]. The user does not think about the offset between the mouse and the cursor when the mouse is used as intended. Even though the mouse typically sits on a flat horizontal surface and the screen sits vertically (i.e., the screen is rotated 90° from the mouse), intuitively we expect if we move the mouse forward/back then the cursor will move up/down in the same way because we think of both as up/down movements. However, for anyone who has attempted to use a mouse when it is rotated by 90° to the right, they know simple manipulations can be extremely difficult due to up becoming left and right becoming up. The same is true for moving objects in VR; the direction of movement for a grabbed object, even if at a distance, should be mapped directly to that selected object whenever possible. Likewise, when a user rotates a virtual object with an input device, the virtual object should rotate in the same direction; that is, both should rotate around the same axis of rotation [Poupyrev et al. 2000].

Nulling compliance states that when a device returns to its initial placement, the corresponding virtual object should also return to its initial placement [Buxton 1986]. Nulling compliance can be accomplished with absolute devices (relative to some reference), but not relative devices (relative to itself)—see Section 27.1.3. For example, if a device is attached to the user's belt, nulling compliance is important, as the user can use "muscle memory" to remember the initial, neutral placement of the device and corresponding virtual object (Section 26.2).

Temporal compliance. Temporal compliance states that different sensory feedback corresponding to the same action or event should be synced appropriately in time. Viewpoint feedback should be immediate to match vestibular cues, otherwise motion sickness may result (Chapter 15). But even for reasons unrelated to sickness, feedback should be immediate; otherwise users may become frustrated or give up before tasks are completed. Even if the entire action cannot be completed immediately, there should be some form of feedback implying the problem is being worked on. Without such information, users can become annoyed and computing resources can be wasted since the user may have forgotten about the task and moved on to something else. In

fact, slow or poor feedback may be worse than no feedback, as it can be distracting, irritating, and anxiety provoking. Anyone who has attempted to browse the Web on an extremely slow Internet connection can attest to this.

Non-spatial Mappings

Non-spatial mappings are functions that transform a spatial input into a non-spatial output or a non-spatial input into a spatial output. Some indirect spatial to non-spatial mappings are universal, such as moving the hand up signifies more and moving the hand down signifies less. Other mappings are personal, cultural, or task dependent. For example, some people think of time moving from left to right whereas others think of time moving from behind the body to the front of it.

25.3 Direct vs. Indirect Interaction

Different types of interaction can be thought of as being on a continuum from indirect interaction to direct interaction. Both direct and indirect interactions are useful for VR depending upon the task. Use each where appropriate and do not use one where not appropriate, e.g., don't try to force everything to be direct.

Direct interaction is an impression of direct involvement with an object rather than of communicating with an intermediary [Hutchins et al. 1986]. The most direct interaction occurs when the user directly interacts with a physical object in the hands. Well-designed hand-held tools (e.g., a knife) that directly affect an object are only slightly less direct, as once a user understands the tool, it seems to become one with the user as if the tool was an extension of the body instead of an intermediary object.

An example of direct interaction is when a user moves a virtual object on a touch screen with his finger. However, the virtual object does not follow the finger when the finger leaves the plane of the screen. VR enables more direct interactions than other digital technologies because virtual objects can be directly mapped to the hands in 3D space with full spatial (both directional and positional) and temporal compliance (Section 25.2.5). Directional compliance and temporal compliance are more important than position compliance; consider the feeling of directness with a mouse (albeit not quite as directly as moving the cursor with a finger on a touch screen), even though there is no position compliance. Similarly, manipulating an object at a distance can feel like one is directly manipulating the object.

Indirect interaction requires more cognition and conversion between input and output. Performing a search for an image through a typed or verbal query is an example of an indirect interaction. The user must put thought into what is being searched for, provide the query, wait for a response, and then interpret the result. Not only

must the query be thought about but the conversion from words to and from visual imagery must be considered. Even though such indirect interaction requires more cognition than direct interactions, that does not mean direct interaction is always better. Indirect interactions are more effective for their intended purposes.

Somewhere in the middle of the extremes of direct and indirect interactions are semi-direct interactions. An example is a slider on a panel that controls the intensity of a light. The user is directly controlling the intermediary slider but less directly controlling the light. However, once the slider is grabbed and moved back and forth a couple of times it feels more direct due to the mental mapping of up for lighter and down for darker.

25.4 The Cycle of Interaction

Interaction can be broken down into three parts: (1) forming the goal, (2) executing the action, and (3) evaluating the results [Norman 2013].

Execution bridges the gap between the goal and result. This feedforward is accomplished through the appropriate use of signifiers, constraints, mappings, and the user's mental model. There are three stages of execution that follow from the goal: plan, specify, and perform.

Evaluation enables judgment about achieving a goal, making adjustments, or the creation of new goals. This feedback is obtained by perceiving the impact of the action. Evaluation consists of three stages: perceive, interpret, and compare.

As can be seen in Figure 25.1, the seven stages of interaction consist of one stage for goals, three stages for execution (plan, specify, and perform), and three stages for evaluation (perceive, interpret, and compare). Quality interaction design considers requirements, intentions, and desires at each stage. An example interaction is as follows.

1. **Form the goal.** The question is “What do I want to accomplish?” Example: Move a boulder blocking an intended route of travel.
2. **Plan the action.** Determine which of many possible plans of action to follow. Question: “What are the alternative set of action sequences and what do I choose?” Example: Navigate to the boulder or select from a distance to move it.
3. **Specify an action sequence.** Even after the plan is determined, the specific sequence of actions must be determined. Question: “What are the specific actions of the sequence?” Example: Shoot a ray from the hand, intersect the boulder, push the grab button, move the hand to the new location, release the grab button.

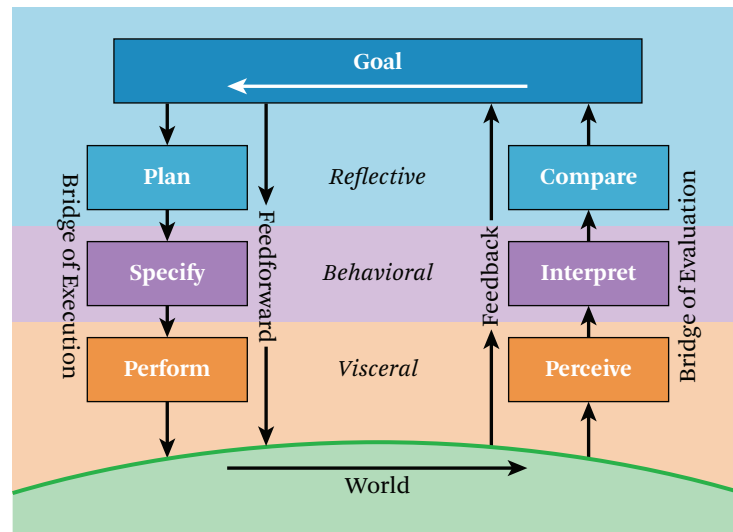


Figure 25.1 The cycle of interaction. (Adapted from Norman [2013])

4. **Perform** the action sequence. Question: “Can I take action now?” One must actually take action to achieve a result. Example: Move the boulder.
5. **Perceive** the state of the world. Question: “What happened?” Example: The boulder is now in a new location.
6. **Interpret** the perception. Question: “What does it mean?” Example: The boulder is no longer in the path of desired travel.
7. **Compare** the outcome with the goal. Questions: “Is this okay? Have I accomplished my goal?” Example: The route has been cleared so navigation along the route is possible.

The interaction cycle can be initiated by establishing a new goal (goal-driven behavior). The cycle can also initiate from some event in the world (data-driven or event-driven behavior). When initiated from the world, goals are opportunistic rather than planned. Opportunistic interactions are less precise and less certain than explicitly specified goals, but they result in less mental effort and more convenience.

Not all activities in these seven stages are conscious. Goals tend to be reflective (Section 7.7) but not always. Often, we are only vaguely aware of the execution and evaluation stages until we come across something new or run into an obstacle, at which point conscious attention is needed. At a high reflective level of goal setting and comparison, we assess results in terms of cause and effect. The middle levels of

specifying and interpreting are often only semi-conscious behavior. The visceral level of performing and perceiving is typically automatic and subconscious unless careful attention is drawn to the actual action.

Many times the goal is known, but it is not clear how to achieve it. This is known as the **gulf of execution**. Similarly, the **gulf of evaluation** occurs when there is a lack of understanding about the results of an action. Designers can help users bridge these gulfs by thinking about the seven stages of interaction, performing task analysis for creating interaction techniques (Section 32.1), and utilizing signifiers, constraints, and mappings to help users create effective mental models for interaction.

25.5 The Human Hands

The human hands are remarkable and complex input and output devices that naturally interact with other physical objects quickly, precisely, and with little conscious attention. Hand tools have been perfected over thousands of years to perform intended tasks effectively. Figure 8.8 shows that a large portion of the sensory cortex is devoted to the hands. It is not surprising, then, that the best fully interactive VR applications make use of the hands. Using interaction techniques that the hands can intuitively work with can add significant value to VR users.

25.5.1 Two-Handed Interaction

It is natural and intuitive to reach out with both hands to manipulate objects in the real world, so common sense says two-handed 3D interfaces should be appropriate and intuitive for VR [Schultheis et al. 2012]. Whereas such intuition tells us that two hands are better than one, two hands can in fact be worse than one hand if the interaction is designed inappropriately [Kabbash et al. 1994]. Perhaps this is a reason why, even though devices like the mouse have been around for decades, most computer interactions are one handed. Anyone who has developed 3D systems knows that 3D devices alone do not guarantee superior performance, and this is even more true for two hands, in part because the hands do not necessarily work in parallel [Hinckley et al. 1998]. Iterative design with feedback from real users (Part VI) is essential for creating high-quality bimanual interfaces.

Bimanual Classifications

Bimanual interaction (two-handed interaction) can be classified as **symmetric** (each hand performs identical actions) or **asymmetric** (each hand performs a different action), with asymmetric tasks being more common in the real world [Guiard 1987].

Bimanual symmetric interactions can further be classified as synchronous (e.g., pushing on a large object with both hands) or asynchronous (e.g., climbing a ladder with one arm reaching up at a time). Scaling by grabbing two sides of an object and spreading the hands apart is an example of a bimanual symmetric interaction.

Bimanual asymmetric interactions occur when both hands work differently but in a coordinated way to accomplish a task. The **dominant hand** is the user's preferred hand for performing fine motor skills. The **non-dominant hand** provides the reference frame, giving the ergonomic benefit of placing the object being worked upon in a way that is comfortable (often subconsciously) for the dominant hand to work with and that does not force the dominant hand to work in a single locked position. The non-dominant hand also typically initiates manipulation of the task and performs gross movements of the object being manipulated, in order to provide convenient, efficient, and precise manipulation by the dominant hand. The most commonly given example is writing; the non-dominant hand controls the orientation of the paper for writing by the dominant hand. Or consider peeling a potato—the potato is much easier to peel when held in the non-dominant hand than when the potato is sitting on a table (whether the potato is locked in place or not). In a similar manner, unimanual interactions in VR can be awkward when the non-dominant hand does not control the reference frame for the dominant hand.

By using two hands in a natural manner, the user can specify spatial relationships, not just absolute positions in space. Bimanual interactions ideally should be designed so the two hands work together in a fluid manner, switching between symmetric and asymmetric modes depending on the current task.

VR Interaction Concepts

VR interactions are not without their challenges. Trade-offs must be considered that may result in interactions being different than in the real world. However, there are enormous advantages VR has over the real world as well. This chapter focuses upon interaction concepts, challenges, and benefits specific to VR.

26.1 Interaction Fidelity

VR interactions are designed on a continuum ranging from an attempt to imitate reality as closely as possible to in no way resembling the real world. Which goal to strive toward depends on the application goals, and most interactions fall somewhere in the middle. **Interaction fidelity** is the degree to which physical actions used for a virtual task correspond to the physical actions used in the equivalent real-world task [Bowman et al. 2012].

On the high end of the interaction fidelity spectrum are **realistic interactions**—VR interactions that work as closely as possible to the way we interact in the real world. Realistic interactions strive to provide the highest level of interaction fidelity possible given the hardware being used. Holding one hand above the other as if holding a bat and swinging them together to hit a virtual baseball has high interaction fidelity. Realistic interactions are often important for training applications, so that which is learned in VR can be transferred to the real-world task. Realistic interactions can also be important for simulations, surgical applications, therapy, and human-factors evaluations. If interactions are not realistic in such applications, problems such as adaptation (Section 10.2) may occur, which can lead to negative training effects for the real-world task being trained for. An advantage of using realistic interactions is that there is little learning required of users since they already know how to perform the actions.

On the other end of the interaction fidelity spectrum are **non-realistic interactions** that in no way relate to reality. Pushing a button on a non-tracked controller to shoot a laser from the eyes is an example of an interaction technique that has low interaction

fidelity. Low interaction fidelity is not necessarily a disadvantage as it can increase performance, cause less fatigue, and increase enjoyment.

Somewhere in the middle of the interaction fidelity spectrum are **magical interactions** where users make natural physical movements but the technique makes users more powerful by giving them new and enhanced abilities or intelligent guidance [Bowman et al. 2012]. Such magical hyper-natural interactions attempt to create “better” ways of interacting by enhancing usability and performance through superhuman capabilities and unrealistic interactions [Smith 1987]. Although not realistic, magical often uses interaction metaphors (Section 25.1) to help users quickly develop a mental model of how an interaction works. Consider interaction metaphors as a source of inspiration for creating new magical interaction techniques. Grabbing an object at a distance, pointing to fly through a scene, or shooting fireballs from the hand are examples of magical interactions. Magic interactions strive to enhance the user experience by reducing interaction fidelity and circumventing the limitations of the real world. Magic works well for games and teaching of abstract concepts.

Interaction fidelity is a multi-dimensional continuum of components. The Framework for Interaction Fidelity Analysis [McMahan et al. 2015] categorizes interaction fidelity into three concepts: biomechanical symmetry, input veracity, and control symmetry.

Biomechanical symmetry is the degree to which physical body movements for a virtual interaction correspond to the body movements of the equivalent real-world task. Biomechanical symmetry makes heavy use of posture and gestures that replicate how one positions and moves their body in the real world. This provides a high sense of proprioception resulting in a high sense of presence since the user feels his body physically acting in the environment as if he were performing the task in the real world. Real walking for VR navigation purposely has a biomechanical symmetry to how we walk in the real world. Walking in place has a lower biomechanical symmetry due to its less realistic movements. Pressing a button or joystick to walk forward has no biomechanical symmetry.

Input veracity is the degree to which an input device captures and measures users’ actions. Three aspects that dictate the quality of input veracity are accuracy, precision, and latency. A system with low input veracity can significantly affect performance due to difficulty capturing quality input.

Control symmetry is the degree of control a user has for an interaction as compared to the equivalent real-world task. High-fidelity techniques provide the same control as the real world without the need for different modes of interaction. Low control symmetry can result in frustration due to the need to switch between techniques to obtain full control. For example, directly manipulating object position and rotations (6 DoF) with a tracked hand controller has greater control symmetry than indirectly manip-

ulating the same object with gamepad controls because the gamepad controls (less than 6 DoF) require using multiple translation and rotational modes. However, low control symmetry can also have superior performance if implemented well. For example, non-isomorphic rotations (Section 28.2.1) can be used to increase performance by amplifying hand rotations.

26.2 Proprioceptive and Egocentric Interaction

As described in Section 8.4, proprioception is the physical sense of the pose and motion of the body and limbs. Because most VR systems do not provide a sense of touch outside of hand-held devices, proprioception can be especially important for exploiting the one real object every user has—the human body [Miné et al. 1997]. The body provides an egocentric frame of reference (Section 26.3.3) in which to work, and interactions relative to the body’s reference frame are more effective than techniques relying solely on visual information. In fact, eyes-off interactions can be performed in peripheral vision or even outside the field of view of the display, which reduces visual clutter. The user also has a more direct sense of control within personal space—it is easier to place an object directly with the hand rather than through less direct means.

26.2.1 Mixing Egocentric and Exocentric Interactions

Exocentric interactions consist of viewing and manipulating a virtual model of the environment from outside of it. With egocentric interaction, the user has a first-person view of the world and typically interacts from within the environment. Don’t assume one or the other must be chosen. These egocentric and exocentric interactions can be mixed together so that the user can view himself on a smaller map (Sections 22.1.1 and 28.5.2), and/or manipulate the world in an exocentric manner but from an egocentric perspective (Figure 26.1).

26.3 Reference Frames

A **reference frame** is a coordinate system that serves as a basis to locate and orient objects. Understanding reference frames is essential to creating usable VR interactions. This section describes the most important reference frames as they relate specifically to VR interaction. The virtual-world reference frame, real-world reference frame, and torso reference frame are all consistent with one other when there is no capability to rotate, move, or scale the body or the world (e.g., no virtual body motion, no torso tracking, and no moving the world). The reference frames diverge when that is not the case.

Although thinking abstractly about reference frames and how they relate can be difficult, reference frames are naturally perceived and more intuitively understood



Figure 26.1 An exocentric map view from an egocentric perspective. (Courtesy of Digital ArtForms)

when one is immersed and interacting with the different reference frames. The reference frames discussed in this section will be more intuitively understood by actually experiencing them.

26.3.1 The Virtual-World Reference Frame

The **virtual-world reference frame** matches the layout of the virtual environment and includes geographic directions (e.g., north) and global distances (e.g., meters) independent of how the user is oriented, positioned, or scaled. When creating content over a wide area, forming a cognitive map, determining global position, or planning travel on a large scale (Section 10.4.3), it is typically best to think in terms of the exocentric virtual-world reference frames. Care should be taken when placing direct hand interfaces relative to the virtual-world reference frame as reaching specific locations can be difficult and awkward unless the user is able to easily and precisely navigate and turn through the environment.

26.3.2 The Real-World Reference Frame

The **real-world reference frame** is defined by real-world physical space and is independent of any user motion (virtual or physical). For example, as a user virtually flies forward the user's physical body is still located in the real-world reference frame. A physical desk, computer screen, or keyboard sitting in front of the user is in the real-world reference frame. A consistent physical location should be provided to set

any tracked or non-tracked hand-held controller when not being used. For tracked controllers or other tracked objects, make sure to match the virtual model with the physical controller in form and position/orientation in the real-world reference frame (i.e., full spatial compliance) so users can see it correctly and more easily pick it up.

In order for virtual objects, interfaces, or rest frames to be solidly locked into the real-world reference frame, the VR system must be well calibrated and have low latency. Such interfaces often, but not always, provide output cues only to help provide a rest frame (Section 12.3.4) in order for users to feel stabilized in physical space and to reduce motion sickness. Automobile interiors, cockpits (Figure 18.1), or non-realistic stabilizing cues (Figure 18.2) are examples of cues in the real-world reference frame. In some cases it makes sense to add the capability to input information through real-world reference-framed elements (e.g., buttons located on a virtual cockpit). A big advantage of real-world reference frames is that passive haptics (Section 3.2.3) can be added to provide a sense of touch that matches visually rendered elements.

26.3.3 The Torso Reference Frame

The **torso reference frame** is defined by the body's spinal axis and the forward direction perpendicular to the torso. The torso reference frames are especially useful for interaction because of proprioception (Sections 8.4 and 26.2)—the sense of where one's arms and hands are felt relative to the body. The torso reference frame can also be useful for steering in the direction the body is facing (Section 28.3.2).

The torso reference frame is similar to the real-world reference frame in the sense that both frames move with the user through the virtual world as the user virtually translates or scales. The difference is that virtual objects in the torso reference frame, virtual objects rotate with the body (both virtual and physical body turns) and move with physical translation whereas objects in the real-world reference frame do not. The chair a user is seated in can be tracked instead of the torso if it can be assumed the torso is stable relative to the chair. Systems with head tracking but not torso or chair tracking can assume the body is always facing forward (i.e., the torso reference frame and real-world reference frame are consistent). However, physical turning of the body can cause problems due to the system not knowing if only the head turned or the entire body turned.

If no hand tracking is available, the hand reference frame can be assumed to be consistent with the torso reference frame. For example, a visual representation of a non-tracked hand-held controller should move and rotate with the body (hand-held controllers are often assumed to be held in the lap). For VR, information displays often work better in torso reference frames rather than head reference frames as commonly done with heads-up displays in traditional first-person video games. Figure 26.2 shows



Figure 26.2 Information and a visual representation of a non-tracked hand-held controller in the torso reference frame. (Courtesy of NextGen Interactions)

an example of a visual representation of a non-tracked hand-held controller and other information at waist level in the torso reference frame.

Body-Relative Tools

Just like in the real world, tools in VR can be attached to the body so that they are always within reach no matter where the user goes. This is done in VR by simply placing the tool in the torso reference frame. This not only provides convenience of the tool always being available but also takes advantage of the user's body acting as a physical mnemonic, which helps in recall and acquisition of frequently used control [Miné et al. 1997].

Items should be placed outside the forward direction so that they do not get in the way of viewing the scene (e.g., the user simply looks down to see the options and then can select via a point or grab). Advanced users should be able to turn off items or make them invisible. Examples of physical mnemonics are pull-down menus located above the head (Section 28.4.1), tools surrounding the waste as a utility belt, audio options



Figure 26.3 A simple transparent texture in the hand can convey the physical interface. (Courtesy of NextGen Interactions)

at the ear, navigation options at the user's feet, and deletion by throwing an object behind the shoulder (and/or object retrieval by reaching behind the shoulder).

26.3.4 The Hand Reference Frames

The **hand reference frames** are defined by the position and orientation of the user's hands, and hand-centric judgments occur when holding an object in the hand. Hand-centric thinking is especially important when using a phone, tablet, or VR controller. Placing a visual representation of a tracked hand-held controller (Section 27.2.3) in the hand(s) can help add a sense of presence due to the sense of touch matching the visuals. Placing labels or icons/signifiers in the hand reference frame that point to buttons, analog sticks, or fingers is extremely helpful (Figure 26.3), especially for new users. The option to turn on/off such visuals should be provided so as to not occlude/clutter the scene when not using the interface or after the interface has been memorized. Although both the left and right hands can be thought of as separate reference frames, the non-dominant hand is useful for serving as a reference frame for the dominant hand to work in (Section 25.5.1), especially for hand-held panels (Section 28.4.1).

26.3.5 The Head Reference Frame

The **head reference frame** is based on the point between the two eyes and a reference direction perpendicular to the forehead. In psychological literature, this reference frame is known as the **cyclopean eye**, which is a hypothetical position in the head that serves as our reference point for the determination of a head-centric straight-ahead [Coren et al. 1999]. People generally tend to think of this straight-ahead as a direction in front of themselves, oriented around the midline of the head, regardless of where the eyes are actually looking. From an implementation point of view, the head reference frame is equivalent to the head-mounted-display reference frame, but from the user's point of view (assuming a wide field of view), the display is not visually perceived. A world-fixed secondary display that shows what the user is seeing matches the head reference frame.

Heads-up displays (HUDs) are often located in the head reference frame. Such heads-up display information should be minimized, if used at all, other than a selection pointer for gaze-directed selection (Section 28.1.2). If used, it is important to make cues small (but large enough to be easily perceived/readable), minimize the number of visual cues to not be annoying or distracting, not make the cues too far in the periphery, give the cues depth so they are occluded properly by other objects (Section 13.2), and place the cues at a far enough distance so that there is not an extreme accommodation-vergence conflict (Section 13.1). It can also be useful to make the cues transparent. Figure 26.4 shows an example HUD in the head reference frame that serves as a virtual helmet that helps the user to target objects.

26.3.6 The Eye Reference Frames

The **eye reference frames** are defined by the position and orientation of the eyeballs. Very few VR systems support the orientation portion of eye reference frames due to the requirement for eye tracking (Section 27.3.2). However, the left/right horizontal offset of the eyes is easily determined by the interpupillary distance of the specific user measured in advance of usage.

When looking at close objects (for example, when sighting down the barrel of a gun) and assuming a binocular display, eye reference frames are important to consider because they are differentiated from the head reference frame due to the offset from the forehead to the eye. The offset between the left and right eyes results in double images for close objects when looking at an object further in the distance (or double images of the further object when looking at the close object). Thus, when users are attempting to line up close objects with further objects (e.g., a targeting task), users should be advised to close the non-dominant eye and to sight with the dominant eye (Sections 9.1.1 and 23.2.2).



Figure 26.4 A heads-up display in the head reference frame. No matter where the user looks with the head, the cues are always visible. (Courtesy of NextGen Interactions)

26.4 Speech and Gestures

The usability of speech and gestures depends upon the number and complexity of the commands. More commands require more learning—the number of voice commands and gestures should be limited to keep interaction simple and learnable.

Voice interfaces and gesture recognition systems are normally invisible to the user. Use explicit signifiers, such as a list of possible commands or icons of gestures, in the users' view so they know and remember what is possible. Neither speech nor gesture recognition is perfect. In many cases it is appropriate to have users verify commands to confirm the system understands correctly before taking action. Feedback should also be provided to let the user know a command has been understood (e.g., highlight the signifier if the corresponding command has been activated).

Use a set of well-defined, natural, easy-to-understand, and easy-to-recognize gestures/words. Pushing a button to signal to the computer that a word or gesture is intended to start (i.e., push-to-talk or push-to-gesture) can keep the system from recognizing unintended commands. This is especially true when the user is also communicating with other humans, rather than just the system itself (for both voice and gestures as humans subconsciously make gestures as they talk).

26.4.1 Gestures

A **gesture** is a movement of the body or body part whereas a **posture** is a single static configuration. Each conveys some meaning whether intentional or not. Postures can be considered a subset of gestures (i.e., a gesture over a very short period of time or a gesture with imperceptible movement). Dynamic gestures consist of one or more tracked points (consider making a gesture with a controller) whereas a posture requires multiple tracked points (e.g., a hand posture).

Gestures can communicate four types of information [Hummels and Stappers 1998].

Spatial information is the spatial relationship that a gesture refers to. Such gestures can manipulate (e.g., push/pull), indicate (e.g., point or draw a path), describe form (e.g., convey size), describe functionality (e.g., twisting motion to describe twisting a screw), or use objects. Such direct interaction is a form of structural communication (Section 1.2.1) and can be quite effective for VR interaction due to its direct and immediate effect on objects.

Symbolic information is the sign that a gesture refers to. Such gestures can be concepts like forming a V shape with the fingers, waving to say hello or goodbye, and explicit rudeness with a finger. The formation of such gestures is structural communication (Section 1.2.1) whereas the interpretation of the gesture is indirect communication (Section 1.2.2). Symbolic information can be useful for both human-computer interaction and human-human interaction.

Pathic information is the process of thinking and doing that a gesture is used with (e.g., subconsciously talking with one's hands). Pathic information is most commonly visceral communication (Section 1.2.1) added on to indirect communication (Section 1.2.1) that is useful for human-human interaction.

Affective information is the emotion a gesture refers to. Such gestures are more typically body gestures that convey mood such as distressed, relaxed, or enthusiastic. Affective information is a form of visceral communication (Section 1.2.1) most often used for human-human interaction, although pathic information is less commonly recognized with computer vision as discussed in Section 35.3.3.

In the real world, hand gestures often augment communication with gestures such as okay, stop, size, silence, kill, goodbye, pointing, etc. Many early VR systems used gloves as input and gestures to indicate similar commands. Advantages of gestures include flexibility, the number of degrees of freedom of the human hand, the lack of having to hold a device in the hand, and not necessarily having to see (or at least look directly at) the hand. Gestures, like voice, can also be challenging due to having to remember them and most current systems have low recognition rates for more than a few gestures. Although gloves are not as comfortable, they are more consistent than camera-based systems due to not having line-of-sight issues. Push-to-gesture systems can drastically reduce false positives. This is especially true when the user is communicating with other humans, rather than just the system itself.

Direct vs. Indirect Gestures

Direct gestures are immediate and structural (Section 1.2.1) in nature and convey spatial information; they can be interpreted and responded to by the system as soon as the gesture starts. Direct manipulation, such as pushing an object, and selection via hand pointing are examples of direct gestures. **Indirect gestures** indicate more complex semantic meaning over a period of time so the start of the gesture is not sufficient—the application interprets over a range of movement so there is a delay from the start of the gesture. Indirect gestures convey symbolic, pathic, and affective information. A single posture command is somewhere between direct and indirect because the system response is immediate but not structural (the posture is interpreted as a command).

26.4.2 Speech Recognition

Speech recognition translates spoken words into textual and semantic form. If implemented well, voice commands have many advantages including keeping the head and hands free to interact while giving commands to the system. Voice recognition does have significant challenges including limited recognition capability, not always obvious command options, difficulty in selecting from a continuous scale, background noise, variability between speakers, and distraction to other individuals [McMahan et al. 2014]. Regardless, speech can work well for multimodal interactions (Section 26.6). Speech recognition categories, strategies, and errors are discussed below as described by Hannema [2001].

Speech Recognition Categories

Speech recognition is often categorized into the following groups.

Speaker-independent speech recognition has the flexibility to recognize a small number of words from a wide range of users. This type of speech recognition is used with telephone navigation systems and is best used with VR when there are

only a small number of options provided to the user (a VR system should visually show available commands so the user knows what the options are).

Speaker-dependent speech recognition recognizes a large number of words from a single user where the system has been extensively trained to recognize words from that specific user. This type of speech recognition can work well with VR when the user has a personal system that she uses often.

Adaptive recognition is a mix of speaker-independent and speaker-dependent speech recognition. The system does not need explicit training but learns the characteristics of the specific user as he speaks. This often requires that the user corrects the system when words are misinterpreted. Use adaptive recognition when users have their own system but don't want to bother with explicitly training the voice recognizer.

Speech Recognition Strategies

Each of the speech recognition categories listed above can use one or more of the following strategies to recognize words.

Discrete/isolated strategies recognize one word at a time from a predefined vocabulary. This strategy works well when only one word is used or there is a silence between consecutive words. Examples include commands such as “save,” “undo,” “restart,” or “freeze.”

Continuous/connected strategies recognize consecutive words from a predefined vocabulary. This is more challenging to implement than a discrete/isolated strategy.

Phonetic strategies recognize individual phonemes (small, perceptually distinct sounds; Section 8.2.3), diphones (combinations of two adjacent phonemes), or triphones (combinations of three adjacent phonemes). Triphones are computationally expensive, and the system may be slow to respond due to the number of combinations that must be recognized so are rarely used.

Spontaneous/conversational strategies attempt to determine the context of the words in sentences in a way similar to what humans do. This results in a natural spoken dialogue with the computer. This strategy can be difficult to implement well.

Speech Recognition Errors

There are several reasons why speech recognition is difficult. By being aware of the common types of errors listed below, the system can be better designed to minimize

such errors. Errors can also be reduced by using a microphone designed for speech recognition (Section 27.3.3).

Deletion/rejection occurs when the system fails to match or recognize a word from the predetermined vocabulary. The advantage of this type of error is that the system recognizes the failure and can request the user to repeat the word.

Substitution occurs when the system misrecognizes a word for a different word than that which was intended. If the error is not caught, the system might execute a wrong command. This error is difficult to detect, but statistical measures can be used to calculate confidence.

Insertion occurs when an unintended word is recognized. This most often occurs when the user is not intentionally speaking to the system, such as when thinking out loud or when speaking to another human. Similar to a substitution error, this can execute an unintended command. Requiring the user to push a button (e.g., a push-to-talk interface) can drastically reduce this type of error.

Context

The specific context that the user is engaged in at any particular time can help improve accuracy and better match the user's intention. This is important as words can be homonyms (the same word with multiple meanings, e.g., volume can be a volume of space or audio volume) or homophones (different words with the same sound, e.g., die and dye). Context-sensitive systems with a large vocabulary can be implemented by having the system only able to recognize a subset of that vocabulary at any particular time.

26.5 Modes and Flow

Although ideally the same metaphors should be applied across all interactions for a single application, this is often not possible. Complex applications with different types of tasks may require different interaction techniques. In such a case, different techniques might be combined together. The mechanism to choose a technique may be as simple as pressing a different button or a mode selection from a hand-held panel, or the technique may be order dependent (e.g., a specific manipulation technique only occurs after a specific selection technique). Whatever the mode, that mode should be made clear to the user.

All interactions should also integrate and flow together well. The overall usability of a system depends on the seamless integration of various tasks and techniques provided by the application. One way to think about flow is to consider the sequence of

basic actions. People may more often verbally state commands with the action coming before the object to be acted upon, but they tend to think about the object first. Objects are more concrete in nature so they are easier to first think about, whereas verbs are more abstract and are more easily thought about when being applied to something. For example, someone might think “pick up the book,” but before thinking about picking up the book the person must first perceive and think about the book. Users prefer object-action sequences over action-object sequences as it requires less mental effort [McMahan and Bowman 2007]. Thus, when designing interaction techniques, the selection of the object to be acted upon should be performed (at least in most cases) before taking action upon that object. The interaction technique should also enable easy and smooth transition between selecting an object and manipulating or using that object.

At a higher level, the flow of longer interactions should occur without distractions so the user can give full attention to the primary task. Ideally, users should not have to physically (whether the eyes, head, or hands) or cognitively move between tasks. Lightweight mode switching, physical props, and multimodal techniques can help to maintain the flow of interaction.

26.6 Multimodal Interaction

No single sensory input or output is appropriate for all situations. **Multimodal interactions** combine multiple input and output sensory modalities to provide the user with a richer set of interactions. The “put-that-there” interface is known as the first human-computer interface to effectively and naturally mix voice and gesture [Bolt 1980]. Note although “put-that-there” is an action-object sequence, as discussed above in Section 26.5, better flow often occurs by first selecting the object to be moved. The better implementation might be called a “that-moves-there” interface.

When choosing or designing multimodal interactions, it can be helpful to consider different ways of integrating the modalities together. Input can be categorized into six types of combinations: specialized, equivalence, redundancy, concurrency, complementarity, and transfer [Laviola 1999, Martin 1998]. All of the input modality types are multimodal other than specialized.

Specialized input limits input options to a single modality for a specific application. Specialization is ideal when there is clearly a single best modality for the task. For example, for some environments, selecting an object might only be performed by pointing.

Equivalent input modalities provide the user a choice of which input to use, even though the result would be the same across modalities. Equivalence can be thought of as the system being indifferent to user preferences. For example, a user might be able to create the same objects either by voice or through a panel.

Redundant input modalities take advantage of two or more simultaneous types of input that convey the same information to perform a single command. Redundancy can reduce noise and ambiguous signals, resulting in increased recognition rates. For example, a user might select a red cube with the hand while saying “select the red cube” or physically move an object with the hand while saying “move.”

Concurrent input modalities enable users to issue different commands simultaneously, and thus enable users to be more efficient. For example, a user might be pointing to fly while verbally requesting information about an object in the distance.

Complementarity input modalities merge different types of input together into a single command. Complementarity often results in faster interactions as the different modalities are typically close in time or even concurrent. For example, to delete an object, the application might require the user to move the object behind the shoulder while saying “delete.” Another example is a “put-that-there” interface [Bolt 1980] that merges voice and gesture to place an object.

Transfer occurs when information from one input modality is transferred to another input modality. Transfer can improve recognition and enable faster interactions. A user may achieve part of a task by one modality but then determine a different modality would be more appropriate to complete the task. In such a case, transfer would prevent the user from needing to start over. An example is verbally requesting a specific menu to appear, which can be interacted with by speaking or pointing. Another example is a “push-to-talk” interface. Transfer is most appropriate to use when hardware is unreliable or does not work well in some situations.

26.7 Beware of Sickness and Fatigue

Some interaction techniques, especially those controlling the viewpoint, can cause motion sickness. When choosing or creating navigation techniques, designers should carefully understand and consider scene motion and motion sickness as discussed in Part III. If motion sickness is a primary concern, then changing the viewpoint

should only occur through one-to-one mapping of real head motion or teleportation (Section 28.3.4).

Some users are not comfortable looking at interfaces close to the face for extended periods of time due to the accommodation-vergence conflict (Section 13.1) that occurs in most of today's HMDs. Visual interfaces close to the face should be minimized.

As mentioned in Section 14.1, gorilla arm can be a problem for interactions that require the user to hold their hands up high and out in front of themselves for more than a few seconds at a time. This occurs even with bare-hand systems (Section 27.2.5) where the user is not carrying any additional weight. Interactions should be designed to minimize holding the hands above the waist for more than a few seconds at a time. For example, shooting a ray from the hand held at the hip is quite comfortable.

26.8 Visual-Physical Conflict and Sensory Substitution

Most VR experiences offer little haptic feedback, and when they do the feedback is quite limited compared to the sense of touch in the real world. Not having full haptic feedback is more of a problem than just not feeling objects. The hand or other body part (or physical device) continues to move through the object since there is no (or limited) physical force stopping it from doing so. As a result, the physical location of the hand may no longer match the visual location.

Enforcing simulated physics so the hand does not visually pass through visual geometry is often preferred by users when the penetrations are only slight (shallow penetration). When deeper penetration occurs, then users prefer the visual hands to match the physical hands even though that breaks the intuition that hands do not pass through objects [Lindeman 1999]. Stopping the visual hand for deep penetration can be especially confusing when the visual hand pops out of a different part of the penetrated object than where the visual hand has previously been visually stopped. A compromise solution for non-realistic interactions is to draw two hands when the physical hand and physically simulated hand diverge (see ghosting below).

In some cases, the virtual hand can be considerably offset from the physical hand without the user noticing as visual representation tends to dominate proprioception [Burns et al. 2006]. However, this is not always the case. Vision is generally stronger than proprioception when moving the hand in a left/right and/or up/down direction, but proprioception can be stronger when moving the hand in depth (forward/back) [Van Beers et al. 2002].

Sensory substitution is the replacement of an ideal sensory cue that is not available with one or more other sensory cues. Examples of sensory substitution that work well with VR are described below.



Figure 26.5 In the game “The Gallery: Six Elements,” the bottle is highlighted to show the object can be grabbed. (Courtesy of Cloudhead Games)

Ghosting is a second simultaneous rendering of an object in a different pose than the actual object. In some cases it is appropriate to render the hand twice—both where the physical hand is located and where the physics simulation states the hand is located. Ghosting is also often used to provide a clue of where a virtual object will be snapped into place if released. Be careful of using ghosting for training applications as users can depend on ghosting as a crutch that will not be available for the real-world task.

Highlighting is visually outlining or changing the color of an object. Highlighting is most often used to show the hand has intersected with an object so that it can be selected or picked up. Highlighting is also used to convey that an object is able to be selected or grabbed when the hand is close even though a collision has not yet occurred. Figure 26.5 shows an example of highlighting.

Audio cues are very effective in conveying to a user that one of his hands has collided with some geometry. Audio might be as simple as a tone sound or real-world recorded audio track. In some cases, providing multiple audio files with variations (e.g., random grunt sounds when colliding with a virtual wall or when shot by an enemy) can help with a sense of realism and reduce annoyance. Continuous contact sounds can also be used to convey sliding along surfaces. Sound properties such as pitch or amplitude might also change depending on penetration depth.

Passive haptics (static physical objects that can be touched; Section 3.2.3) are effective when the virtual world is limited to the physical space where no virtual

navigation can occur (i.e., when the real-world reference frames and virtual-world reference frames are consistent; Section 26.3) or when tracked physical tools travel with the user (i.e., the physical and virtual objects are spatially compliant; Section 25.2.5). Because vision often dominates proprioception, perfect spatial compliance is not always required [Burns et al. 2006]. Redirected touching warps virtual space to map many differently shaped virtual objects onto a single real object (i.e., hand or finger tracking is not one-to-one) in a way that the discrepancy between virtual and physical is below the user's perceptual threshold [Kohli 2013]. For example, when one's real hand traces a physical object, the virtual hand can trace a slightly differently shaped virtual object.

Rumble causes an input device to vibrate. Although not the same haptic force that would occur in the real world, rumble feedback can be quite an effective cue for informing the user she has collided with an object.

Input Devices

Input devices are the physical tools/hardware used to convey information to the application and to interact with the virtual environment. Some interaction techniques work more or less across different input devices whereas other techniques work only with input devices that have specific characteristics. Thus, appropriately choosing input hardware that best fits the application's interaction techniques is an important design decision (or, conversely, designing and implementing interaction techniques depends upon available input hardware).

This chapter describes some general characteristics of input devices and then describes the primary classes of input devices.

27.1 Input Device Characteristics

Input devices can be very different, and the characteristics of each should be considered when choosing hardware and designing interactions.

27.1.1 Size and Shape

The most obvious characteristics to a new VR user are the basic shape and size of the input device. The shape and size has more to do with just how the controller looks and feels in the hand. Large hand-held devices are primarily controlled by large muscle groups of the shoulder, elbow, and wrist, whereas smaller hand-held devices utilize smaller and faster muscle groups in the fingers [Bowman et al. 2004]. Smaller devices can also decrease **clutching**—the releasing and regrasping of an object in order to complete a task due to not being able to complete it in a single motion (such as a wrench). Gloves also use these smaller muscle groups and have the advantage of being able to freely touch and feel other items.

27.1.2 Degrees of Freedom

Input devices are often classified by the number of degrees of freedom (DoF) they report. **Degrees of freedom (DoF)** are the number of dimensions that an input device

is capable of manipulating (also see Section 25.2.3). Devices range from a single DoF (e.g., an analog trigger), to 6 DoF that measure full 3D translation (up/down, left/right, forward/backward) and rotation (roll, pitch, and yaw), to full hand or body tracking with many DoFs. A traditional mouse, joystick, trackball, and touchpad (a rotatable ball that is essentially an upside-down mouse) are examples of 2 DoF devices. VR hand tracking should have a minimum of 6 DoF (multiple points tracked on the hand have more than 6 DoF). For a majority of active VR experiences, one or more 6 DoF hand-held controllers is often the most appropriate choice. For some simple tasks only requiring navigation and no direct interaction, a non-tracked hand-held controller is good enough.

27.1.3 Relative vs. Absolute

Relative input devices measure the differences between the current and last measurement. Mice, trackballs, and inertial trackers are examples of relative devices. Relative devices drift over time and thus are not nulling compliant (Section 25.2.5). Although limited, the Nintendo Wii proved relative devices can work well for natural interactions under some circumstances if the applications are carefully designed. VR relative devices typically use inertial measurement units (IMUs) that have the advantage of having a higher update rate (e.g., 1,000 Hz) and faster response (e.g., 1 ms) than absolute measurements. **Absolute input devices** sense pose relative to a constant point of reference independent of past measurements and are nulling compliant. **Hybrid tracking systems** fuse both relative and absolute trackers to provide the advantages of both. VR head and hand tracking should sense pose via absolute measurements (although relative devices can arguably estimate absolute pose of the hands from modeling the constraints imposed by the physicality of the arms and hand).

27.1.4 Separable vs. Integral

Integral input devices enable users to control all DoFs simultaneously from a single motion (a single composition) whereas **separable input devices** contain at least one DoF that cannot be controlled simultaneously from a single motion (two or more distinct compositions). A gamepad with two different analog sticks is an example of a separable device. A 2D device that enables control of more than two dimensions via mode switching is also an example of a separable device. VR hand tracking should be integral.

27.1.5 Isometric vs. Isotonic

Isometric input devices measure pressure or force that contains no or little actual movement. **Isotonic input devices** measure deflection from a center point and may

or may not have some resistance. Mice are isotonic input devices. Joysticks can be either isotonic or isometric. Isotonic input devices are best for controlling position, whereas isometric input devices are best for controlling rates such as navigation speed. An isometric joystick, for example, works well for controlling velocity (i.e., hold to continue moving).

27.1.6 Buttons

Buttons control one DoF via pushing with a finger and typically take on one of two states (e.g., pushed or not pushed) although some buttons can take on analog values (also known as analog triggers). Buttons are often used to change modes, to select and object, or to start an action. Although buttons can be useful for VR applications, too many buttons can lead to confusion and error—especially when the button-mapping functionality is unclear or inconsistent (although attaching labels to virtual controllers can help—see Figure 26.3 as an example). Consider the capabilities and intuitiveness of desktop applications that are controlled via no more than three buttons on a mouse.

There is a debate between bare-hand system (Section 27.2.5) and hand-held controller (Sections 27.2.2 and 27.2.3) advocates as to the utility of buttons. For example, Microsoft Kinect and Leap Motion developers believe buttons are a primitive and unnatural form of input whereas Playstation Move and Sixsense Stem developers believe buttons are an essential part of game play. Like any great debate, the answer is “it depends” [Jerald et al. 2012]. Buttons can be abstracted to indirectly trigger nearly any action, but this abstraction can cause a disconnect between the user and the application. Buttons are most effective when an action is binary, when the action needs to occur often, when reliability is required, and when physical feedback to the user is essential. Buttons are also ideal for time-sensitive actions, since very little time is required for the physical action (i.e., an entire dynamic gesture does not need to be completed in order to register the action). Gestures can be slower and more fatiguing than button presses, particularly in command-intensive tasks such as modeling or radiology. Natural buttonless hand manipulation is most effective for providing a sense of realism and presence, when abstraction is not appropriate, or when detailed tracking of the entire hand is required.

27.1.7 Encumbrance

Unencumbered input devices do not require physical hardware to be held or worn. Such systems are implemented with camera systems. Thus no “suit-up” time is required (although calibration might be necessary). Unencumbered systems can also reduce hygiene issues (Section 14.4) since a physical device is not passed between users. Non-encumbrance is not always a design goal; holding something physically in

the hand can add to the sense of presence (Section 3.2.3); consider holding a controller in the hand vs. not holding a controller in the hand for a shooting or golf experience.

27.1.8 Ability to Fully Interact with Physical Objects

Some devices enable one to touch the real world in natural ways without the device getting in the way. Bare-hand tracking systems (i.e., camera systems) and gloves are the most common examples of this. Held devices and world-grounded devices (Section 27.2.1) must be released before the hand can fully interact with other physical objects. In such cases, the physical devices should be tracked and rendered in the virtual world so the user can reach out to grab them again.

27.1.9 Device Reliability

Device **reliability** is the extent to which an input device can consistently work within the user's entire personal space (and a larger volume if the user is expected to physically move around). Devices should ideally have 100% reliability anywhere the user can reach with no loss in tracking acquisition. Reliability should carefully be considered when choosing an input device as unreliable devices can result in frustration, fatigue (e.g., due to having to hold the hand high and in front of the body; Section 14.1), increased cognitive load (e.g., because the user must think about holding the device in a certain way), breaks-in-presence (Section 4.2), and reduced performance.

Unreliable tracking can be due to multiple reasons and divided into two sets of factors: (1) implementation limitations and (2) inherent physical limitations. An inherent physical limitation is the best a device will ever be able to achieve given an optimal engineering solution/implementation. Some devices cannot possibly provide 100% reliability no matter what the engineering effort. A system that requires line of sight from a sensor to the tracked device can be occluded by another physical object (such as a hand or torso), in which case there is no way for the system to reliably determine the pose of the device (although the state of the device can be estimated for short periods of time). VR devices should ideally work in all orientations and hand postures (e.g., hands covering sensors or tracking the fingers when a fist is made).

Another challenge of reliability occurs for users attempting to work in a tracked volume smaller than the user's personal space. An example is a vision-based system with a limited field of view. Lighting can also be a challenge for some vision-based systems, especially in uncontrolled environments outside of the laboratory or some other highly controlled space. Other systems only recognize gestures when the hand is oriented or held in a certain way. Many camera-based hand tracking systems only reliably recognize poses and gestures when the hands are perpendicular to the camera with the fingers visible.

27.1.10 Haptics Capable

Active haptics can easily be added to physical devices that are worn or held. However, the degree of haptics might be limited depending on the size of the device and if attached to the world in some way (Section 3.2.3).

27.2 Classes of Hand Input Devices

The most important VR input devices are the human hands, and this section explores how different types of devices can integrate the hands into VR. An **input device class** is a set of input devices that share the same essential characteristics that are crucial to interaction. Input device classes described in this section focus on the hands and are classified as world-grounded input devices, non-tracked hand-held controllers, tracked hand-held controllers, hand-worn devices, and bare-hand controllers.

Table 27.1 summarizes some of the most essential characteristics of the hand input device classes described in this section and non-hand input device classes described in Section 27.3. As can be seen, no single input device class is universally advantageous although some classes can be combined to create a hybrid system with more advantages. For example, a bare-hands camera-based system might be used with tracked hand-held controllers (although simultaneous usage can be difficult due to different characteristics, so bare hands might be used until the controller is picked up). Note the table is based upon inherent physical limitations rather than today's existing implementations. Technical specifications such as update rate, latency, etc. are not included here as such specs are independent of the hardware (e.g., a fast update rate could potentially be implemented on any device class).

27.2.1 World-Grounded Input Devices

World-grounded input devices, like world-grounded haptics (Section 3.2.3), are designed to be constrained or fixed in the real world and are most often used to interact with desktop systems.

Keyboards and mice are considered to be world-grounded devices and are the most popular form of input that works extremely well for its intended task—2D desktop manipulation. However, such input is not a good way of interacting for most all immersive applications (with the possible exception of video-see-through augmented reality where users can see the mouse and keyboard).

Trackballs and joysticks mounted to a permanent location are world-grounded devices. Other devices offer up to 6 DoF through pushing, pulling, twisting, and/or buttons for mode changes. However, these devices suffer from similar limitations for VR as the mouse due to not being designed to be held comfortably and freely

Table 27.1 Comparison of hand and non-hand input device classes.

	Proprioception	Consistent	Usable in Lap or the Side	Haptics Capable	Unencumbered	Physical Buttons	Hands Free to Interact with Real World	General Purpose
Hand Input Device Class								
World-Grounded Devices	✓	✓		✓	✓	✓	✓	
Non-Tracked Hand-Held Controllers		✓	✓	✓		✓		
Bare Hands	✓				✓		✓	✓
Tracked Hand-Held Controllers	✓	✓	✓	✓		✓		✓
Hand Worn	✓	✓	✓	✓		✓	✓	✓
Non-Hand Input Device Class								
Head Tracking	✓	✓					✓	✓
Eye Tracking							✓	
Microphone			✓		✓		✓	✓
Full-Body Tracking	✓	✓	✓	✓			✓	✓
Treadmills	✓	✓			✓		✓	

in the hands. There are exceptions (e.g., to mount a joystick on the arm of a chair or if simulating a desktop environment where the physical controls precisely match a virtual desktop), but from a human-centered design perspective there should be a solid reason to choose such devices, instead of “because it is available” or “it is what someone else is using.”

World-grounded devices that work extremely well for VR are specialty devices such as handlebars, steering wheels, gas and brake pedals, cockpits, and automotive interior controls. Such devices can be especially good for travel, as most users already

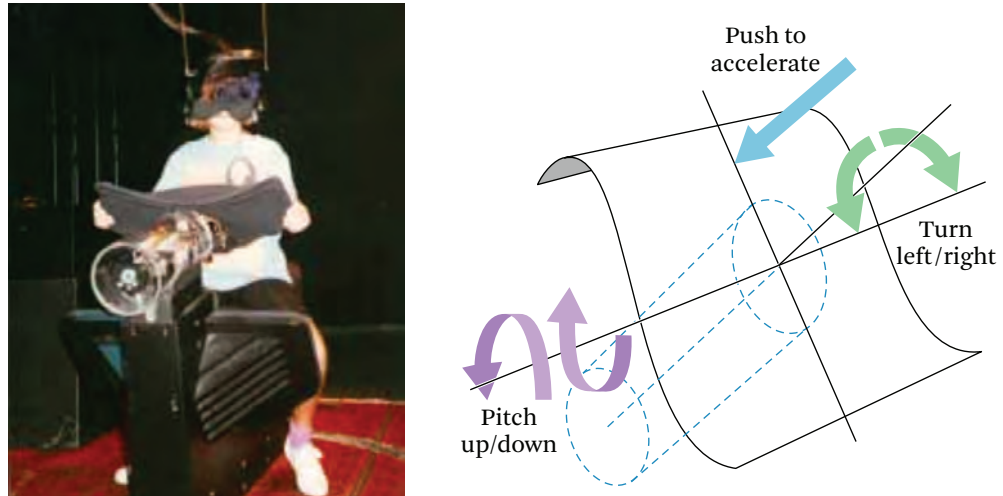


Figure 27.1 The Disney Aladdin world-grounded input device along with its mapping for viewpoint control. (From Pausch et al. [1996])

have real-world experience with devices. Even if world-grounded devices aren't actually used in the real world, they can still be quite effective and presence inducing if designed well. For example, Disney's Aladdin Magic Carpet Ride, 3 DoF controls (Figure 27.1) provides an intuitive physical interface for travel. One reason such controls are so effective is due to having physical signifiers, affordances, and feedback, e.g., the user feels what can be done and how she is doing it. Some challenges of these devices are that creators can't assume there is a wide user base that owns such hardware and it can be difficult to generalize the devices to work with a wide range of tasks. Thus, such devices are more commonly used at location-based entertainment venues where large groups of people use the same device(s) and the device can be designed or modified for the particular VR experience.

27.2.2 Non-Tracked Hand-Held Controllers

Non-tracked hand-held controllers are devices held in the hand that include buttons, joysticks/analog sticks, triggers, etc. but are not tracked in 3D space. Traditional video game input devices such as joysticks and gamepads are the most common form of non-tracked hand-held controllers (Figure 27.2). Many VR applications are starting to support such game controllers. These controllers work much better than the mouse and keyboard since the controller can be held comfortably in the lap where users can continually hold the controller. Many gamers have an intuitive feel of where the



Figure 27.2 The Xbox One controller is an example of a non-tracked hand-held controller.

buttons are through years of use. Controllers with analog sticks work surprisingly well for navigating within VR (Section 28.3.2).

Although not tracked, such controllers can increase presence for seated experiences by placing visual hands and controllers at the approximate location of the user's lap since most users hold the controller in such a position. The visual controller and hands have also been found to cause users to subconsciously move the hands and controller to the visual controller (Andrew Robinson and Sigurdur Gunnarsson, personal communication, May 11, 2015). Unfortunately, when a user moves his hands away from the assumed position, a break-in-presence typically occurs if the user sees the virtual hands stay in place.

27.2.3 Tracked Hand-Held Controllers

Tracked hand-held controllers are typically 6 DoF devices (known as “wands” in the VR research community where they have been used for decades) and can also contain functionality offered by non-tracked hand-held controllers. Tracked hand-held controllers are currently the best option for a majority of interactive VR applications. Tracked hand-held controllers are easy to use for many 3D tasks due to their natural, direct mapping to hand motion. Because the controllers are tracked, they can



Figure 27.3 The Sixense STEM (left) and Oculus Touch (right) tracked hand-held controllers. (Courtesy of Sixense (left) and Oculus (right))

be visually co-located with the real hands (i.e., spatially and temporally compliant; Section 25.2.5) as well as physically felt, providing proprioceptive and passive haptics/touch cues. Labels can also be attached to the virtual representation to provide immediate instruction of what the buttons do by simply looking at where the hands are physically located (Section 26.3.4 and Figure 26.3), adding a big advantage over traditional desktop and gamepad input. Viewpoint manipulation with these devices is typically achieved using buttons and a trackball, integrated analog sticks (as on the Sixense STEM and Oculus Touch controllers as shown in Figure 27.3), or by flying with the hands. Such techniques are described in Section 28.3. Other types of physical controls and feedback can also be added to these devices, such as trackpads and active haptics (e.g., vibration).

Tracked hand-held controllers have the advantage of acting as a physical prop, which enhances presence through physical touch. Not only do such controls facilitate communication with the virtual world, but they also help to make spatial relationships seem more concrete to the user [Hinckley et al. 1998]. However, such props come at the cost of not being able to directly/fully touch and feel other passive objects in the world and world-grounded input devices, such as seats, handlebars, and cockpit controls (Section 27.2), without first setting down the controller/prop.

Tracked hand-held devices typically use inertial, electromagnetic, ultrasonic, or optical (camera) technologies. Each of these technologies has advantages and disadvantages, and hand-held trackers ideally use some hybrid method of integrating multiple technologies together (sensor fusion) to provide both high precision and accuracy.

27.2.4 Hand-Worn Devices

Hand-worn input devices include gloves, muscle-tension sensors (electromyographic or EMG sensors), such as what Thalmic Labs recently made popular with the Myo (which is worn on the arm but measures hand motion), and rings.

Many believe gloves (Figure 27.4) to be the ultimate VR interface as they theoretically have many advantages, such as not having line-of-sight, sensor field-of-view, or lighting requirements so the hands can be held comfortably to the side or in the lap with no concern of losing tracking, resulting in less gorilla arm if the interaction techniques are designed well (Section 18.9). Like bare hands, gloves also have the advantage that the hands and fingers can still fully interact with other physical objects.

Unfortunately, like bare-hand systems, full-hand tracked gloves are lacking in their current form and will require dramatic improvement to be used by the masses. Consistent recognition of more than a few gestures is still challenging due to the lack of consistent finger tracking accuracy. Recognizing more than a few gestures requires the user to recalibrate often due to the glove moving on the hand. Gloves must also be put on and worn, which can become uncomfortable and result in sweaty hands. There is also a risk of social resistance to wearing gloves similar to the resistance of Google Glass—although those willing to wear an HMD on their face are unlikely to care what others think of wearing gloves. If such challenges can be solved, then gloves may eventually become the input device of choice for VR.

The Fakespace Pinch Gloves have functions more like buttons, with near 100% consistent recognition, more than typical gloves that do full hand and finger tracking. They work via a conductive cloth sewn into the tip of each finger. When two or more fingers touch then the circuit closes resulting in a signal. This simple design provides the capability for a large number of pinch gestures; combinations range from two to ten fingers touching each other plus poses involving separate but simultaneous pinches (e.g., left thumb to left index finger and right thumb to right index finger pinched at the same time). In practice, due to the usability of physical hand constraints and users' willingness to memorize gestures, applications will only use so many of these gestures, similar to how too many buttons on a hand controller would be too confusing. Pinch gloves can be used quite well with many of the example techniques described in Chapter 28.

Perhaps one of the most significant advantages of gloves is that both full hand tracking and button simulation via pinch gestures can be combined such as demonstrated by LaViola and Zeleznik [1999]. Haptics can also be used with gloves, such as is done with the CyberGlove CyberTouch (as shown in Figure 27.4 but with buzzers added to provide a sense of haptics).



Figure 27.4 The CyberGlove is an example of a hand-worn device. (Courtesy of CyberGlove Systems LLC)

If EMG sensors and rings can be made more accurate, then they too might become an ideal fit for many applications.

27.2.5 Bare Hands

Bare-hand input devices work via sensors aimed at the hands (mounted in the world or on the HMD). Figure 27.5 shows the hands and a skeletal model fit to the hands, as seen by the user. The obvious major advantage is that the user's hands are completely unencumbered. Many believe bare-hand systems will ultimately be the ideal VR interface. Although the bare hands work extremely well in the real world, it turns out consistent interaction with the bare hands in VR is an enormous challenge. Challenges include not having a sense of touch, fatigue from holding the hands in front of the sensor, line-of-sight requirements, and consistent recognition of gestures across a wide range of users. Such technical challenges lead to usability challenges, such as being able to comfortably work with the hands in the lap without concern for where the sensor(s) is located. The bare hands also lack physical buttons, which is important for some applications but not important for other applications (Section 27.1.6).

Regardless of the challenges of effectively working with the bare hands in VR, seeing the entirety of one's hands in 3D is extremely compelling and provides a nice sense of presence for the periods of time when the tracking does consistently work. It remains to be seen if such challenges will be overcome and/or accepted by a wide range of users.

27.3 Classes of Non-hand Input Devices

VR input can occur through more than just the hands. This section describes head tracking, eye tracking, microphones, and full-body tracking.

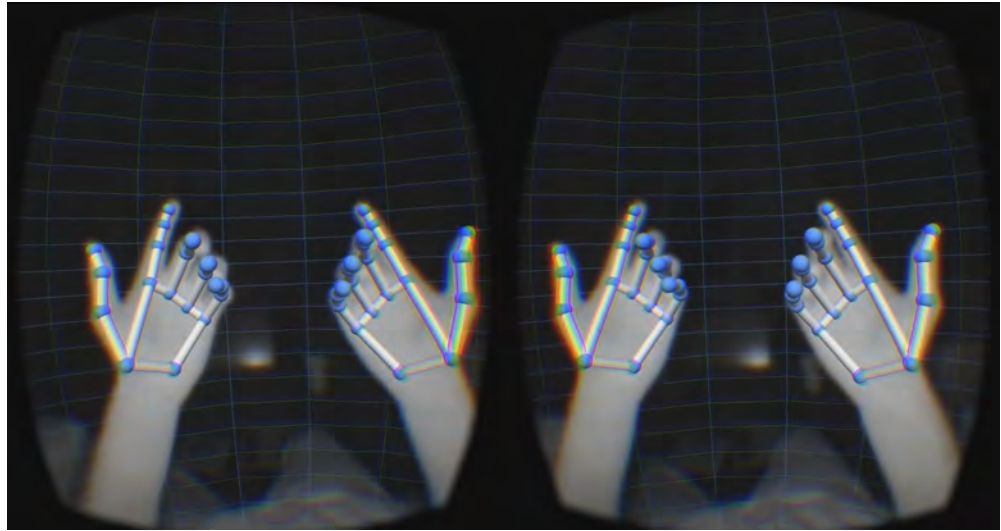


Figure 27.5 A depth-sensing camera on the HMD looking out enables one to see her own hands in detail. Here a skeleton model is also fit to the hands. (Courtesy of Leap Motion)

27.3.1 Head Tracking Input

Head tracking must be accurate, precise, fast, and well calibrated for the virtual world to appear stable. World stability is essential for VR and assumed to work well, but is not the focus of this chapter. Here, **head-tracking input** refers to interaction that modifies or provides feedback beyond just seeing the virtual environment. The most common form of head-tracking interaction is to aim by looking. One way of doing this is to provide a reticle or pointer in the middle of the screen (Section 26.3.5) that is triggered by a button press, firing in the direction of the reticle, or selecting an option the reticle projects onto. Other more subtle interactions can be used, such as having an action occur in the direction the user is looking, having characters respond when looked at, or simple head gestures such as nodding the head yes or no.

27.3.2 Eye Tracking Input

An **eye-tracking input device** tracks where the eyes are looking. Eye-tracking input for VR is a largely unexplored topic other than the obvious of selecting (Section 28.1.2) or firing where one is looking as mostly demonstrated with interactive eye-tracking systems that are integrated within some of today's HMDs.

The **Midas Touch problem** refers to the fact that people expect to look at things without that look “meaning” something [Jacob 1991]. Interacting via eye tracking

alone is usually not a good idea—eye tracking works better with multimodal input. For example, signaling via a clutch (e.g., push a button, blink, or say “select”) typically works better than signaling via a dwell time. Even with a clutch, designing interactions that work well with gaze is a challenge. Straightforward feedback in the form of a pointer/reticle that moves with the eyes can be annoying to users and occludes viewing in the part of vision with the highest visual acuity. Eye saccades can also make the pointer jitter or jump in unintended ways. These problems can be mitigated by only turning on the pointer when a button is held down and filtering out high-frequency motions.

Eye tracking can be more effective for specialized tasks and for subtle interactions, such as how a character responds when looked at. The following guidelines are useful to consider when designing eye-tracking interactions [Kumar 2007].

Maintain the natural function of the eyes. Our eyes are meant for looking, and interaction designers should maintain the natural function of the eye. Using the eyes for other purposes overloads the visual channel.

Augment rather than replace. Attempting to replace existing interfaces with eye tracking is typically not appropriate. Instead, think about how to add functionality onto already existing and newly created interfaces. Gaze can provide context and inform the system that the user is paying attention to a specific object or area in the scene.

Focus on interaction design. Focus on the overall experience instead of eye tracking alone. Consider the number of steps in the interaction, the amount of time it takes, the cost of an error/failure, cognitive load, and fatigue.

Improve the interpretation of eye movements. Gaze data is noisy. Consider how to best filter eye movements, classify gaze data, recognize gaze patterns, and take other input modalities into account.

Choose appropriate tasks. Don’t try to force gaze to solve every problem. Eye tracking is not appropriate for all tasks. Consider the task and scenario before choosing gaze.

Use passive gaze over active gaze. Consider ways in which gaze can be used more passively so the eyes can better maintain their natural function.

Leverage gaze information for other interactions. Leverage the system’s knowledge of where the user is paying attention in order to provide context for non-gaze interactions.

Although typically not interactive, eye tracking will have significant benefit for VR usability by providing clues to content creators of what most engages a user's attention (see attention maps in Section 10.3.2), similar to the use of eye tracking to inform and enhance website design.

27.3.3 Microphones

A **microphone** is an acoustic sensor that transforms physical sound into an electric signal. Use a headset microphone specifically designed for speech recognition that includes noise-canceling features that remove/reduce ambient noise. The microphone should be able to be easily adjusted/positioned while the HMD is on and while not being able to see it. Accuracy improves as the microphone comes closer to and in front of the mouth. Microphones should be comfortable and light.

To prevent speech recognition errors (Section 26.4.2), such as sounds picked up when thinking aloud or another person's voice, use a push-to-talk interface. If hand-held controllers are used, then the push-to-talk button should be located on the controller.

27.3.4 Full-Body Tracking

Full-body tracking consists of tracking more than just the head and hands. Full-body tracking can significantly add to the illusion of self-embodiment as well as the illusion of social presence (Section 4.3). Tracking a large number of features can also be used to enhance interaction (e.g., a game that allows kicking a ball).

VR full-body tracking is typically done with a motion capture suit, similar to what is used in the film industry. Different suits utilize different technologies such as electromagnetic sensors, retro-reflective markers, and inertial sensors. Depth cameras such as Microsoft Kinect can theoretically track the entire body, but it can be difficult to capture the entire body unless multiple cameras are used. Regardless, full-body camera capture systems, like Microsoft Kinect, can be used for creating extremely compelling experiences, even though resolution is not yet great and much of the body might go in and out of view. Figure 27.6 shows real-time capture of the real world and display of the resulting point-cloud data within an HMD.

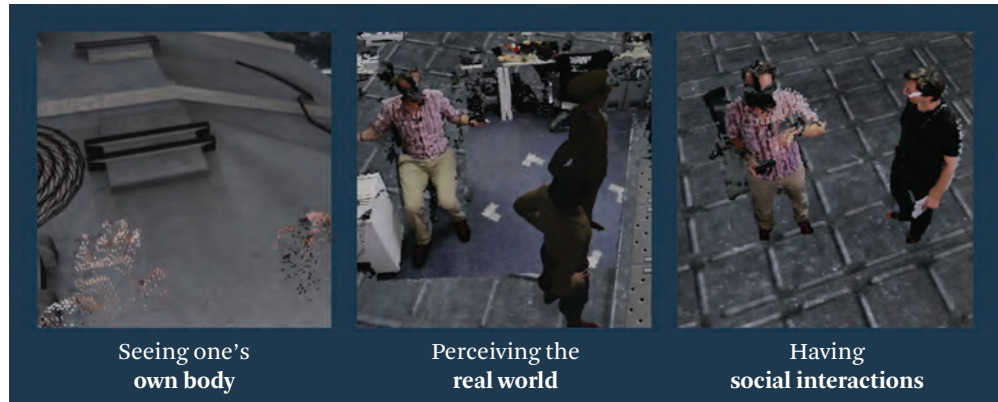


Figure 27.6 Depth cameras enable users to see their own body, the real world, and/or other people from the real world. (Courtesy of Dassault Systèmes, iV Lab)

Interaction Patterns and Techniques

An **interaction pattern** is a generalized high-level interaction concept that can be used over and over again across different applications to achieve common user goals. The interaction patterns here are intended to describe common approaches to general VR interaction concepts at a high level. Note interaction patterns are different than software design patterns (Section 32.2.5) that many system architects are familiar with. Interaction patterns are described from the user's point of view, are largely implementation independent, and state relationships/interactions between the user and the virtual world along with its perceived objects.

An **interaction technique** is more specific and more technology dependent than an interaction pattern. Different interaction techniques that are similar are grouped under the same interaction pattern. For example, the walking pattern (Section 28.3.1) covers several walking interaction techniques ranging from real walking to walking in place. The best interaction techniques consist of high-quality affordances, signifiers, feedback, and mappings (Section 25.2) that result in an immediate and useful mental model for users.

Distinguishing between interaction patterns and interaction techniques is important for multiple reasons [Sedig and Parsons 2013].

- There are too many existing interaction techniques with many names and characterizations to remember, and many more will be developed in the future.
- Organizing interaction techniques under the umbrella of a broader interaction pattern makes it easier to consider appropriate design possibilities by focusing on conceptual utility and higher-level design decisions before worrying about more specific details.
- Broader pattern names and concepts make it easier to communicate interaction concepts.
- Higher-level groupings enable easier systematic analysis and comparison.

- When a specific technique fails, then other techniques within the same pattern can be more easily thought about and explored, resulting in better understanding of why that specific interaction technique did not work as intended.

Both interaction patterns and interaction techniques provide conceptual models to experiment with, suggestions and warnings of use, and starting points for innovative new designs. Interaction designers should know and understand these patterns and techniques well so they have a library of options to choose from depending on their needs and a base to innovate upon. Do not fall into the trap that there is a single best interaction pattern or technique. Each pattern and technique has strengths and weaknesses depending on application goals and the type of user [Wingrave et al. 2005]. Understanding distinctions and managing trade-offs of different techniques is essential to creating high-quality interactive experiences.

This chapter divides VR interaction patterns into selection, manipulation, viewpoint control, indirect control, and compound patterns. The interaction patterns and their interaction techniques as organized in this chapter are shown in Table 28.1. The

Table 28.1 Interaction patterns as organized in this chapter. More specific example interaction techniques are described for each interaction pattern.

- Selection Patterns (Section 28.1)
 - Hand Selection Pattern
 - Pointing Pattern
 - Image-Plane Selection Pattern
 - Volume-Based Selection Pattern
- Manipulation Patterns (Section 28.2)
 - Direct Hand Manipulation Pattern
 - Proxy Pattern
 - 3D Tool Pattern
- Viewpoint Control Patterns (Section 28.3)
 - Walking Pattern
 - Steering Pattern
 - 3D Multi-Touch Pattern
 - Automated Pattern
- Indirect Control Patterns (Section 28.4)
 - Widgets and Panels Pattern
 - Non-Spatial Control Pattern
- Compound Patterns (Section 28.5)
 - Pointing Hand Pattern
 - World-in-Miniature Pattern
 - Multimodal Pattern

first four patterns are often used sequentially (e.g., a user may travel toward a table, select a tool on the table, and then use that tool to manipulate other objects on the table) or can be integrated together into compound patterns. Interaction techniques that researchers and practitioners have found to be useful are then described within the context of the broader pattern. These techniques are only examples, as many other techniques exist, and such a list could never be exhaustive. The intent for describing these patterns and techniques is for readers to directly use them, to extend them, and to serve as inspiration for creating entirely new ways of interacting within VR.

28.1 Selection Patterns

Selection is the specification of one or more objects from a set in order to specify an object to which a command will be applied, to denote the beginning of a manipulation task, or to specify a target to travel toward [McMahan et al. 2014]. Selection of objects is not necessarily obvious in VR, especially when most objects are located at a distance from the user. **Selection patterns** include the Hand Selection Pattern, Pointing Pattern, Image-Plane Selection Pattern, and Volume-Based Selection Pattern. Each has advantages over the others depending on the application and task.

28.1.1 Hand Selection Pattern

Related Patterns

Direct Hand Manipulation Pattern (Section 28.2.1) and 3D Tool Pattern (Section 28.2.3).

Description

The **Hand Selection Pattern** is a direct object-touching pattern that mimics real-world interaction—the user directly reaches out the hand to touch some object and then triggers a grab (e.g., pushing a button on a controller, making a fist, or uttering a voice command).

When to Use

Hand selection is ideal for realistic interactions.

Limitations

Physiology limits a fully realistic implementation of hand selection (the arm can only be stretched so far and the wrist rotated so far) to those objects within reach (personal space), requiring the user to first travel to place himself close to the object to be selected. Different user heights and arm lengths can make it uncomfortable for some people to select objects that are at the edge of personal space. Virtual hands and



Figure 28.1 A realistic hand with arm (left), semi-realistic hands with no arms (center), and abstract hands (right). (Courtesy of Cloudhead Games (left), NextGen Interactions (center), and Digital ArtForms (right))

arms often occlude objects of interest and can be too large to select small items. Non-realistic hand selection techniques are not as limiting.

Exemplar Interaction Techniques

Realistic hands. Realistic hands are extremely compelling for providing an illusion of self-embodiment (Section 4.3). Although ideally the entire arm would be tracked, inverse kinematics can estimate the pose of the arm quite well where users typically don't notice differences in arm pose if the head and/or torso is tracked along with the hand. Figure 28.1 (left) shows an example of a view from a user who has grabbed a bottle.

Modeling users (e.g., measuring arm length) and placing objects within a comfortable range depending on the measured arm length is ideal. However, Digital ArtForms found no complaints from ages 10 to adult in a semi-immersive world after setting a single arm-length scale value that was reasonable for the entire range of body sizes.

Non-realistic hands. Hands do not need to necessarily look real, and trying to make hands and arms realistic can limit interaction. **Non-realistic hands** does not try to mimic reality but instead focuses on ease of interaction. Often hands are used without arms (Figure 28.1, center) so that reach can be scaled to make the design of interactions easier. Although the lack of arms can be disturbing at first, users quickly learn to accept having no arms. The hands also need not look like hands. For abstract applications, users are quite accepting of abstract 3D cursors (Figure 28.1, right) and still feel like they are directly selecting objects. Such hand cursors reduce problems of visual occlusion. Hand occlusion can also be mitigated by making the hand transparent (although proper transparency rendering of multiple objects can be technically challenging to do correctly).

Go-go technique. The go-go technique [Poupyrev et al. 1996] expands upon the concept of a non-realistic hand by enabling one to reach far beyond personal space. The virtual hand is directly mapped to the physical hand when within 2/3 of the full arm's reach and when extended further, the hand “grows” in a nonlinear manner enabling the user to reach further into the environment. This technique enables closer objects to be selected (and manipulated) with greater accuracy while allowing further objects to be easily reached. Physical aspects of arm length and height have been found to be important for the go-go technique, so measuring arm length should be considered when using this technique [Bowman 1999]. Measuring arm length can be done by simply asking the user to hold out the hands in front of the body at the start of the application. Bowman and Hodges [1997] describes extensions to the go-go technique, such as providing rate control (i.e., velocity) options that enable infinite reach, and compares these to pointing techniques. Non-isomorphic hand rotations (Section 28.2.1) are similar but scale rotations instead of position for manipulation.

28.1.2 Pointing Pattern

Related Patterns

Widgets and Panels Pattern (Section 28.4.1) and Pointing Hand Pattern (Section 28.5.1).

Description

The Pointing Pattern is one of the most fundamental and often-used patterns for selection. The **Pointing Pattern** extends a ray into the distance and the first object intersected can then be selected via a user-controlled trigger. Pointing is most typically done with the head (e.g., a crosshair in the center of the field of view) or a hand/finger.

When to Use

The Pointing Pattern is typically better for selection than the Hand Selection Pattern unless realistic interaction is required. This is especially true for selection beyond personal space and when small hand motions are desired. Pointing is faster when speed of remote selection is important [Bowman 1999], but is also often used for precisely selecting close objects, such as pointing with the dominant hand to select components on a panel (Section 28.4.1) held in the non-dominant hand.

Limitations

Selection by pointing is usually not appropriate when realistic interaction is required (the exception might be if a laser pointer or remote control is being modeled).

Straightforward implementations result in difficulty selecting small objects in the distance. Pointing with the hand can be imprecise due to natural hand tremor [Bowman 1999]. Object snapping and precision mode pointing described below can mitigate this problem.

Exemplar Interaction Techniques

Hand pointing. When hand tracking is available, **hand pointing** with a ray extending from the hand or finger is the most common method of selection. The user then provides a signal to actually select the item of interest (e.g., a button press or gesture with the other hand).

Head pointing. When no hand tracking is available, selection via head pointing is the most common form of selection. **Head pointing** is typically implemented by drawing a small pointer or reticle at the center of the field of view so the user simply lines up the pointer with the object of interest and then provides a signal to select the object. The signal is most commonly a button press but when buttons are not available then the item is often selected by **dwell selection**, that is, by holding the pointer on the object for some defined period of time. Dwell selection is not ideal due to having to wait for objects to be selected and accidental selection when looking at an object of interest.

Eye gaze selection. Eye gaze selection is a form of pointing implemented with eye tracking (Section 27.3.2). The user simply looks at an item of interest and then provides a signal to select the looked-at object. In general, eye gaze selection is typically not a good selection technique, primarily due to the Midas Touch problem discussed in Section 27.3.2.

Object snapping. Object snapping [Haan et al. 2005] works by objects having scoring functions that cause the selection ray to snap/bend toward the object with the highest score. This technique works well when selectable objects are small and/or moving.

Precision mode pointing. Precision mode pointing [Kopper et al. 2010] is a non-isomorphic rotation technique that scales down the rotational mapping of the hand to the pointer, as defined by the **control/display (C/D) ratio**. The result is a “slow motion cursor” that enables fine pointer control. A zoom lens can also be used that scales the area around the cursor to enable seeing smaller objects (but the zoom should not be affected by head pose unless the zoom area on the display is small; Section 23.2.5). The user can control the amount of zoom with a scroll wheel on a hand-held device.

Two-handed pointing. Two-handed pointing originates the selection at the near hand and extends the ray through the far hand [Miné et al. 1997]. This provides more

precision when the hands are further apart and fast rotations about a full 360° range when the hands are closer together (whereas a full range of 360° for a single hand pointer is difficult due to physical hand constraints). The distance between the hands can also be used to control the length of the pointer.

28.1.3 Image-Plane Selection Pattern

Also Known As

Occlusion and Framing.

Related Pattern

World-in-Miniature Pattern (Section 28.5.2).

Description

The **Image-Plane Selection Pattern** uses a combination of eye position and hand position for selection [Pierce et al. 1997]. This pattern can be thought about as the scene and hand being projected onto a 2D image plane in front of the user (or on the eye). The user simply holds one or two hands between the eye and the desired object and then provides a signal to select the object when the object lines up with the hand and eye.

When to Use

Image-plane techniques simulate direct touch at a distance, thus are easy to use [Bowman et al. 2004]. These techniques work well at any distance as long as the object can be seen.

Limitations

Image-plane selection works for a single eye, so users should close one eye while using these techniques (or use a monoscopic display). Image-plane selection results in fatigue when used often due to having to hold the hand up high in front of the eye. As in the Hand Selection Pattern, the hand often occludes objects if not transparent.

Exemplar Interaction Techniques

Head crusher technique. In the head crusher technique (Figure 28.2), the user positions his thumb and forefinger around the desired object in the 2D image plane.

Sticky finger technique. The sticky finger technique provides an easier gesture—the object underneath the user's finger in the 2D image is selected.

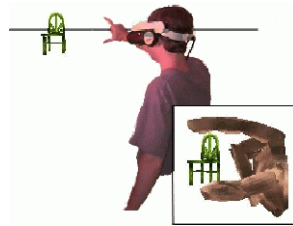


Figure 28.2 The head crusher selection technique. The inset shows the user's view of selecting the chair. (From [Pierce et al. \[1997\]](#))

Lifting palm technique. In the **lifting palm technique**, the user selects objects by flattening his outstretched hand and positions the palm so that it appears to lie below the desired object.

Framing hands technique. The **framing hands technique** is a two-handed technique where the hands are positioned to form the two corners of a frame in the 2D image surrounding an object.

28.1.4 Volume-Based Selection Pattern

Related Patterns

3D Multi-Touch Pattern (Section [28.3.3](#)) and World-in-Miniature Pattern (Section [28.5.2](#)).

Description

The **Volume-Based Selection Pattern** enables selection of a volume of space (e.g., a box, sphere, or cone) and is independent of the type of data being selected. Data to be selected can be volumetric (voxels), point clouds, geometric surfaces, or even space containing no data (e.g., to follow with filling that space with some new data). Figure [28.3](#) shows how a selection box can be used to carve out a volume of space from a medical dataset.

When to Use

The Volume-Based Selection Pattern is appropriate when the user needs to select a not-yet-defined set of data in 3D space or to carve out space within an existing dataset. This pattern enables selection of data when there are no geometric surfaces (e.g., medical CT datasets), whereas geometric surfaces are required for implementing many other selection patterns/techniques (e.g., pointing requires intersecting a ray with object surfaces).

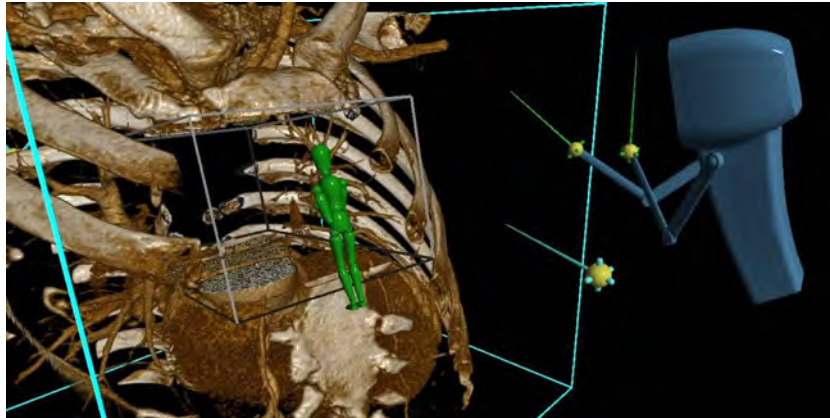


Figure 28.3 The blue user/avatar on the right has carved out a volume of space within a medical dataset via snapping, nudging, and reshaping a selection box (the gray box in front of the green avatar). The green avatar near the center is stepping inside the dataset to examine it from within. (Courtesy of Digital ArtForms)

Limitations

Selecting volumetric space can be more challenging than selecting a single object with the other more common selection patterns.

Exemplar Interaction Techniques

Cone-casting flashlight. The cone-casting flashlight technique uses pointing, but instead of using a ray, a cone is used. This results in easier selection of small objects than standard pointing via ray casting. If the intent is a single object, then the object closest to the cone's center line or the object closest to the user can be selected [Liang and Green 1994]. A modification of this technique is the aperture technique [Forsberg et al. 1996], which enables the user to control the spread of the selection volume by bringing the hand closer or further away.

Two-handed box selection. Two-handed box selection uses both hands to position, orient, and shape a box via snapping and nudging. Snap and nudge are asymmetric techniques where one hand controls the position and orientation of the selection box, and the second hand controls the shape of the box [Yoganandan et al. 2014]. Both snap and nudge mechanisms have two stages of interaction—grab and reshape. Grab influences the position and orientation of the box. Reshape changes the shape of the box.

Snap immediately brings the selection box to the hand and is designed to quickly access regions of interest that are within arm's reach of the user. Snap is an absolute interaction technique, i.e., every time snap is initiated the box position/orientation is reassigned. Therefore, snap focuses on setting the initial pose of the box and placing it comfortably within arm's reach.

Nudge enables incremental and precise adjustment and control of the selection box. Nudge works whether the selection box is near or far away from the user by maintaining the box's current position, orientation, and scale for the initial grab, but subsequent motion of the box is locked to the hand. Once attached to the hand, the box is positioned and oriented relative to its initial state—because of this, nudge can be thought of as a relative change in box pose. The box can then be simultaneously reshaped with the other hand while holding down a button.

28.2 Manipulation Patterns

Manipulation is the modification of attributes for one or more objects such as position, orientation, scale, shape, color, and texture. Manipulation typically follows selection, such as the need to first pick up an object before throwing it. **Manipulation Patterns** include the Direct Hand Manipulation Pattern, Proxy Pattern, and 3D Tool Pattern.

28.2.1 Direct Hand Manipulation Pattern

Related Patterns

Hand Selection Pattern (Section 28.1.1), Pointing Hand Pattern (Section 28.5.1), and 3D Tool Pattern (Section 28.2.3).

Description

The **Direct Hand Manipulation Pattern** corresponds to the way we manipulate objects with our hands in the real world. After selecting the object, the object is attached to the hand moving along with it until released.

When to Use

Direct positioning and orientation with the hand have been shown to be more efficient and result in greater user satisfaction than other manipulation patterns [Bowman and Hodges 1997].

Limitations

Like the Hand Selection Pattern (Section 28.1.1), a straightforward implementation is limited by the physical reach of the user.

Exemplar Interaction Techniques

Non-isomorphic rotations. Some form of clutching is required to rotate beyond certain angles, and clutching can hinder performance due to wasted motion [Zhai et al. 1996]. Clutching can be reduced by using non-isomorphic rotations [Poupyrev et al. 2000] that allow one to control larger ranges of 3D rotation with smaller wrist rotation. Non-isomorphic rotations can also be used to provide more precision by mapping large physical rotations to smaller virtual rotations.

Go-go technique. The go-go technique (Section 28.1.1) can be used for manipulation as well as selection with no mode change.

28.2.2 Proxy Pattern

Related Patterns

Direct Hand Manipulation Pattern (Section 28.2.1) and World-in-Miniature Pattern (Section 28.5.2).

Description

A **proxy** is a local object (physical or virtual) that represents and maps directly to a remote object. The **Proxy Pattern** uses a proxy to manipulate a remote object. As the user directly manipulates the local object(s), the remote object(s) is manipulated in the same way.

When to Use

This pattern works well when a remote object needs to be intuitively manipulated as if it were in the user's hands or when viewing and manipulating objects at multiple scales (e.g., the proxy object can stay the same size relative to the user even as the user scales himself relative to the world and remote object).

Limitations

The proxy can be difficult to manipulate as intended when there is a lack of directional compliance (Section 25.2.5); that is, when there is an orientation offset between the proxy and the remote object.

Exemplar Interaction Technique

Tracked physical props. Tracked physical props are objects directly manipulated by the user (a form of passive haptics; Section 3.2.3) that map to one or more virtual objects and are often used to specify spatial relationships between virtual objects. Hinckley et al. [1998] describe an asymmetric two-handed 3D neurosurgical visualization system where the non-dominant hand holds a doll's head and the dominant hand



Figure 28.4 A physical proxy prop used to control the orientation of a neurological dataset. (From [Hinckley et al. \[1994\]](#))

holds a planar object or a pointing device (Figure 28.4). The doll's head directly maps to a remotely viewed neurological dataset and the planar object controls a slicing plane to see inside the dataset. The pointing device controls a virtual probe. Such physical proxies provide direct action-task correspondence, facilitate natural two-handed interactions, provide tactile feedback to the user, and are extremely easy to use without requiring any training.

28.2.3 3D Tool Pattern

Related Patterns

Hand Selection Pattern (Section 28.1.1) and Direct Hand Manipulation Pattern (Section 28.2.1).

Description

The **3D Tool Pattern** enables users to directly manipulate an intermediary 3D tool with their hands that in turn directly manipulates some object in the world. An example of a 3D tool is a stick to extend one's reach or a handle on an object that enables the object to be reshaped.

When to Use

Use the 3D Tool Pattern to enhance the capability of the hands to manipulate objects. For example, a screwdriver provides precise control of an object by mapping large rotations to small translations along a single axis.

Limitations

3D tools can take more effort to use if the user must first travel and maneuver to an appropriate angle in order to apply the tool to an object.

Exemplar Interaction Techniques

Hand-held tools. Hand-held tools are virtual objects with geometry and behavior that are attached to/held with a hand. Such tools can be used to control objects from afar (like a TV remote control) or to work more directly on an object. A paintbrush used to draw on a surface of an object is an example of a hand-held tool. Hand-held tools are often easier to use and understand than widgets (Section 28.4.1) due to being more direct.

Object-attached tools. An object-attached tool is a manipulable tool that is attached/colocated with an object. Such a tool results in a more coupled signifier representing the affordance between the object, tool, and user. For example, a color icon might be located on an object, and the user simply selects the icon at which point a color cube appears so the user can choose the color of the object. Or if the shape of a box can be changed, then an adjustment tool can be made available on the corners of the box (e.g., dragging the vertex).

Jigs. Precisely aligning and modeling objects can be difficult with 6 DoF input devices due to having no physical constraints. One way to enable precision is to add virtual constraints with jigs. Jigs, similar to real-world physical guides used by carpenters and machinists, are grids, rulers, and other referenceable shapes that the user attaches to object vertices, edges, and faces. The user can adjust the jig parameters (e.g., grid spacing) and snap other objects into exact position and orientation relative to the object the jig is attached to. Figure 28.5 shows some examples of jigs. Jig kits support the snapping together of multiple jigs (e.g., snapping a ruler to a grid) for more complex alignments.

28.3 Viewpoint Control Patterns

Viewpoint control is the task of manipulating one's perspective and can include translation, orientation, and scale. Travel (Section 10.4.3) is a form of viewpoint control that does not allow scaling.

Controlling one's viewpoint is equivalent to moving, rotating, or scaling the world. For example, moving the viewpoint to the left is equivalent to moving the world to the right, or scaling oneself to be smaller is equivalent to scaling the world to be larger.

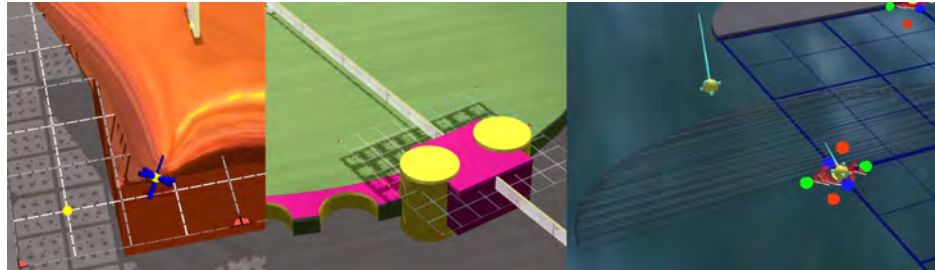


Figure 28.5 Jigs used for precision modeling. The blue 3D crosshairs in the left image represent the user's hand. The user drags the lower left corner of the orange object to a grid point (left). The user cuts shapes out of a cylinder at 15° angles (center). The user precisely snaps a wireframe-viewed object onto a grid (right). (Courtesy of Digital ArtForms and Sixense)

Thus, users perceive themselves either as moving through the world (self-motion) or as the world moving around them (world motion) as the viewpoint changes.

Viewpoint Control Patterns include the Walking Pattern, Steering Pattern, 3D Multi-Touch Pattern, and Automated Pattern. Warning: Some implementations of these patterns may induce motion sickness and are not appropriate for users new to VR or those who are sensitive to scene motion. In many cases, sickness can be reduced by integrating the techniques with the suggestions in Part III.

28.3.1 Walking Pattern

Description

The **Walking Pattern** leverages motion of the feet to control the viewpoint. Walking within VR [Steinicke et al. 2013] includes everything from real walking to mimicking walking by moving the feet when seated.

When to Use

This pattern matches or mimics real-world locomotion and therefore provides a high degree of interaction fidelity. Walking enhances presence and ease of navigation [Usoh et al. 1999] as well as spatial orientation and movement understanding [Chance et al. 1998]. Real walking is ideal for navigating small to medium-size spaces, and such travel results in no motion sickness if implemented well.

Limitations

Walking is not appropriate when rapid or distant navigation is important. True walking across large distances requires a large tracked space, and for wired headsets cable tangling can pull on the headset and be a tripping hazard (Section 14.3). A human

spotter should closely watch walking users to help stabilize them if necessary. Fatigue can result with prolonged use, and walking distance is limited by the physical motion that users are willing to endure. Cable tangling is also an issue when using a wired system; often an assistant holds the wires and follows the user to prevent tangling and tripping.

Exemplar Interaction Techniques

Real walking. Real walking matches physical walking with motion in the virtual environment; the mapping from real to virtual is one-to-one. Because of high interaction fidelity, real walking is an ideal interface for many VR experiences. Real walking typically does not directly measure foot motion, but instead tracks the head. Real walking results in less motion sickness due to better matching visual and vestibular cues (although sickness can still result due to issues such as latency and system miscalibration). The motion of one's feet in a self-embodied avatar can be estimated or the feet can be tracked to provide greater biomechanical symmetry. Unfortunately, real walking by itself limits travel in the virtual world to the physically tracked space.

Redirected walking. Redirected walking [Razzaque et al. 2001] is a technique that allows users to walk in a VR space larger than the physically tracked space. This is accomplished by rotation and translation gains that are different than the true motion of the user, directing the user away from the edges of tracked space (or physical obstacles). Ideally, the gains are below perceptible thresholds so that the user does not consciously realize he is being redirected.

Walking in place. Various forms of walking in place exist [Wendt 2010], but all consist of making physical walking motions (e.g., lifting the legs) while staying in the same physical spot but moving virtually. Walking in place works well when there is only a small tracked area and when safety is a primary concern. The safest form of walking in place is for the user to be seated. Theoretically users can walk in place for any distance. However, travel distances are limited by the physical effort the users are willing to make. Thus, walking in place works well for small and medium-size environments where only short durations of travel are required.

The human joystick. The human joystick [McMahan et al. 2012] utilizes the user's position relative to a central zone to create a 2D vector that defines the horizontal direction and velocity of virtual travel. The user simply steps forward to control speed. The human joystick has the advantage that only a small amount of tracked space is required (albeit more than walking in place).

Treadmill walking and running. Various types of treadmills exist for simulating the physical act of walking and running (Section 3.2.5). Although not as realistic as real walking, treadmills can be quite effective for controlling the viewpoint, providing a sense of self-motion, and walking an unlimited distance. Techniques should make sure foot direction movement is compliant with forward visual motion. Otherwise, treadmill techniques that lack directional and temporal compliance (Section 25.2.5) can be worse than no treadmill. Treadmills with safety harnesses are ideal, especially when physical running is required.

28.3.2 Steering Pattern

Description

The **Steering Pattern** is continuous control of viewpoint direction that does not involve movement of the feet. There is typically no need to control viewpoint pitch with controllers as is done with desktop systems since users can physically look up and down.

When to Use

Steering is appropriate for traveling great distances without the need for physical exertion. When exploring, viewpoint control techniques should allow continuous control, or at least the ability to interrupt a movement after it has begun. Such techniques should also require minimum cognitive load so the user can focus on spatial knowledge acquisition and information gathering. Steering works best when travel is constrained to some height above a surface, acceleration/deceleration can be minimized (Section 18.5), and real-world stabilized cues can be provided (Sections 12.3.4 and 18.2).

Limitations

Steering provides less biomechanical symmetry than the walking pattern. Many users report symptoms of motion sickness with steering. Virtual turning is more disorienting than physical turning.

Exemplar Interaction Techniques

Navigation by leaning. Navigation by leaning moves the user in the direction of the lean. The amount of lean typically maps to velocity. One advantage of this technique is no requirement for hand tracking. Motion sickness can be significant as velocity varies (i.e., acceleration).

Gaze-directed steering. Gaze-directed steering moves the user in the direction she is looking. Typically, the starting and stopping motion in the gaze direction is controlled

by the user via a hand-held button or joystick. This is easy to understand and can work well for novices or for those accustomed to first-person video games where the forward direction is identical to the look direction. However, it can also be disorienting as any small head motion changes the direction of travel, and frustrating since users cannot look in one direction while traveling in a different direction.

Torso-directed steering. **Torso-directed steering** (also called chair-directed steering when the torso is not tracked), utilized when traveling over a terrain, separates the direction of travel from the way one is looking. This has more interaction fidelity than gaze-directed steering since in the real world one does not always walk in the direction the head is pointed. In the case when the torso or chair is not tracked, a general forward direction can be assumed. This technique can have more of a nauseating effect if the user does not have a mental model of what the forward direction is or when the entire body is turned when the torso or chair is not tracked. Visual cues can be provided to help users maintain a sense of the forward direction (Figure 18.2).

One-handed flying. **One-handed flying** works by moving the user in the direction the finger or hand is pointing. Velocity can be determined by the horizontal distance of the hand from the head.

Two-handed flying. **Two-handed flying** works by moving the user in the direction determined by the vector between the two hands, and the speed is proportional to the distance between the hands [Miné et al. 1997]. A minimum hand separation is considered to be a “dead zone” where motion is stopped. This enables one to quickly stop motion by quickly bringing the hands together. Flying backward with two hands is more easily accomplished than one-handed flying (which requires an awkward hand or device rotation) by swapping the location of the hands.

Dual analog stick steering. **Dual analog stick steering** (also known as joysticks or analog pads) work surprisingly well for steering over a terrain (i.e., forward/back and left/right). In most cases, standard first-person game controls should be used where the left stick controls 2D translation (pushing up/down translates the body and viewpoint forward/backward, pushing left/right translates the body and viewpoint left/right) and the right stick controls left/right orientation (pushing left rotates the body and viewing direction to the left and pushing right rotates the body and viewing direction to the right). This mapping is surprisingly intuitive and is consistent with traditional first-person video games (i.e., gamers already understand how to use such controls so they have little learning curve).

Virtual rotations can be disorienting and sickness inducing for some people. Because of this, the designer might design the experience to have the content consistently in the forward direction so that no virtual rotation is required. Alternatively, if the system is wireless and the torso or chair is tracked, then there is no need for virtual rotations since the user can physically rotate 360°.

World-grounded steering devices. World-grounded input devices (Section 27.2.1) such as flight sticks or steering wheels are often used to steer through a world. Such devices can be quite effective for viewpoint control due to the sense of actively controlling a physical device.

Virtual steering devices. Instead of using physical steering devices, virtual steering devices can be used. **Virtual steering devices** are visual representations of real-world steering devices (although they do not actually physically exist in the experience) that are used to navigate through the environment. For example, a virtual steering wheel can be used to control a virtual vehicle the user sits in. Virtual devices are more flexible than physical devices as they can be easily changed in software. Unfortunately, virtual devices are difficult to control due to having no proprioceptive force feedback (although some haptic feedback can be provided when using a hand-held controller with haptic capability).

28.3.3 3D Multi-Touch Pattern

Related Patterns

World-in-Miniature Pattern (Section 28.5.2) and Volume-Based Selection Pattern (Section 28.1.4).

Description

The **3D Multi-Touch Pattern** enables simultaneous modification of the position, orientation, and scale of the world with the use of two hands. Similar to 2D multi-touch on a touch screen, translation via 3D multi-touch is obtained by grabbing and moving space with one hand (monomanual interaction) or with both hands (synchronous bimanual interaction). One difference from 2D multi-touch is that one of the most common ways of using 3D multi-touch is to “walk” with the hands by alternating the grabbing of space with each hand (like pulling on a rope but with hands typically wider apart). Scaling of the world is accomplished by grabbing space with both hands and moving the hands apart or bringing them closer together. Rotation of the world is accomplished by grabbing space with both hands and rotating about a point (typically either about one hand or the midpoint between the hands). Translation, rotation, and scale can all be performed simultaneously with a single two-handed gesture.

When to Use

3D multi-touch works well for non-realistic interactions when creating assets, manipulating abstract data, viewing scientific datasets, or rapidly exploring large and small areas of interest from arbitrary viewpoints.

Limitations

3D multi-touch is not appropriate when the user is confined to the ground. 3D multi-touch can be challenging to implement as small nuances can affect the usability of the system. If not implemented well, 3D multi-touch can be frustrating to use. Even if implemented well, there can be a learning curve on the order of minutes for some users. Constraints, such as those that keep the user upright, limit scale, and/or disable rotations, can be added for novice users. When scaling is enabled and the display is monoscopic (or there are few depth cues), it can be difficult to distinguish between a small nearby object and a larger object that is further away. Therefore, visual cues that help the user create and maintain a mental model of the world and one's place in it can be helpful.

Exemplar Interaction Techniques

Digital ArtForms' Two-Handed Interface. Digital ArtForms has built a mature 3D multi-touch interface called THI (Two-Handed Interface) [Schultheis et al. 2012] based off of the work of Mapes and Moshell [1995] and Multigen-Paradigm's SmartScene interface [Homan 1996] from the 1990s. Figure 28.6 shows a schematic for manipulating the viewpoint. Scale and rotation occur about the middle point between the two hands. Rotation of the world is accomplished by grabbing space with both hands and orbiting

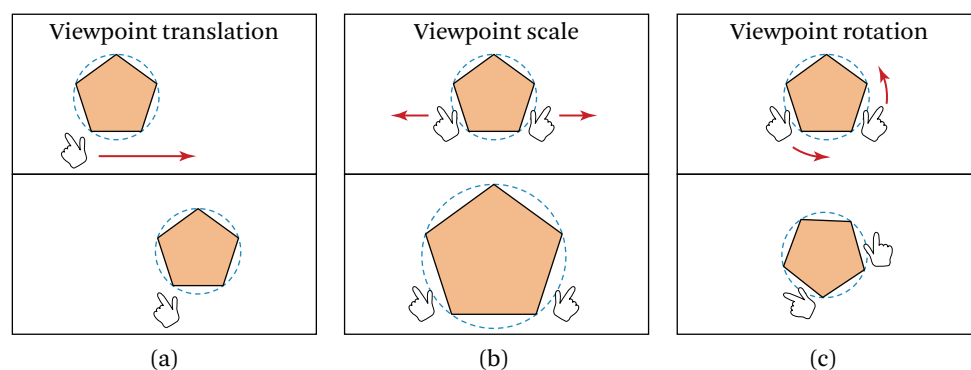


Figure 28.6 Translation, scale, and rotation using Digital ArtForms' Two-Handed Interface. (From Mlyniec et al. [2011])

the hands about the midpoint between the hands, similar to grabbing a globe on both sides and turning it. Note this is not the same as attaching the world to a single hand as rotating with a single hand can be quite nauseating. Translation, rotation, and scale can all be performed simultaneously with a single two-handed gesture. In multi-user environments, other users' avatars appear to grow or shrink as they scale themselves.

Once learned, this implementation of treating the world as an object works well when navigation and selection/manipulation tasks are frequent and interspersed, since viewpoint control and object control are similar other than pushing a different button (i.e., only a single interaction metaphor needs to be learned and cognitive load is reduced by not having to switch between techniques). This gives users the ability to place the world and objects of interest into personal space at the most comfortable working pose via position, rotation, and scaling operations. Digital ArtForms calls this “posture and approach” (similar to what [Bowman et al. \[2004\]](#) call maneuvering). Posture and approach reduces gorilla arm (Section 14.1) and users have worked for hours without reports of fatigue [[Jerald et al. 2013](#)]. In addition, physical hand motions are non-repetitive by nature and are reported not to be subject to repetitive stress due to the lack of a physical planar constraint as is the case with a mouse.

The spindle. Building a mental model of the point being scaled and rotated about along with visualizing intent can be difficult for new users, especially when depth cues are absent. A “spindle” (Figure 28.7) consisting of geometry connecting the two hands (what [Balakrishnan and Hinckley \[2000\]](#) call visual integration), along with a visual indication of the center of rotation/scale dramatically helps users plan their actions and speeds the training process. Users simply place the center point of the spindle at the point they want to scale and rotate about, push a button in each hand, and “pull/scale” themselves toward it (or equivalently “pull/scale” the point and the world toward the user) while also optionally rotating about that point. In addition to visualizing the center of rotation/scale, the connecting geometry provides depth-occlusion cues that provide information of where the hands are relative to the geometry.

28.3.4 Automated Pattern

Description

The **Automated Pattern** passively changes the user's viewpoint. Common methods of achieving this are by being seated on a moving vehicle controlled by the computer or by teleportation.

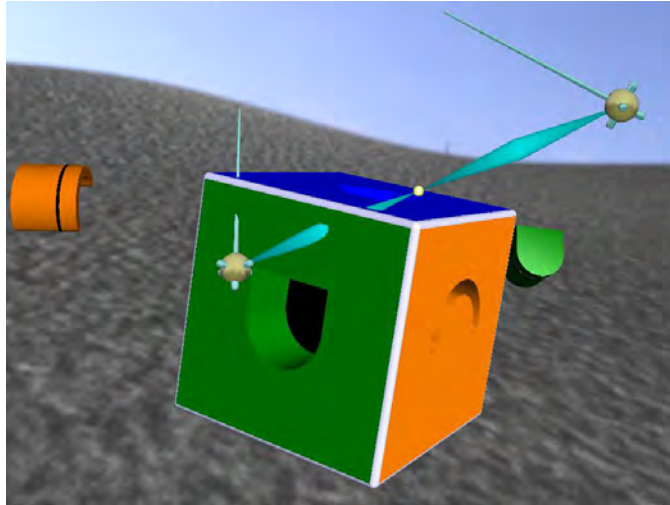


Figure 28.7 Two hand cursors and a spindle connecting those cursors. The yellow dot between the two cursors is the point that is rotated and scaled about. (From Schultheis et al. [2012])

When to Use

Use when the user is playing the role of a passive observer as a passenger controlled by some other entity or when free exploration of the environment is not important or not possible (e.g., due to limitations of today's cameras designed for immersive film).

Limitations

The passive nature of this technique can be disorienting and sometimes nauseating (depending on implementation). This pattern is not meant to be used to completely control the camera independent of user motion. VR applications should generally allow the look direction to be independent from the travel direction so the user can freely look around even if other viewpoint motion is occurring. Otherwise significant motion sickness will result.

Exemplar Interaction Techniques

Reducing motion sickness. Motion sickness can be significantly reduced with this technique by keeping travel speed and direction constant (i.e., keep velocity constant; Section 18.5), providing world-stabilized cues (Section 18.2), and creating a leading indicator (Section 18.4) so users know what motions to expect.

Passive vehicles. **Passive vehicles** are virtual objects users can enter or step onto that transport the user along some path not controlled by the user. Passenger trains, cars, airplanes, elevators, escalators, and moving sidewalks are examples of passive vehicles.

Target-based travel and route planning. **Target-based travel** [Bowman et al. 1998] gives a user the ability to select a goal or location he wishes to travel to before being passively moved to that location. **Route planning** [Bowman et al. 1999] is the active specification of a path between the current location and the goal before being passively moved. Route planning can consist of drawing a path on a map or placing markers that the system uses to create a smooth path.

Teleportation. Teleportation is relocation to a new location without any motion. Teleportation is most appropriate when traveling large distances, between worlds, and/or when reducing motion sickness is a primary concern. Fading out and then fading in a scene is less startling than an instantaneous change. Unfortunately, straightforward teleportation comes at the cost of decreasing spatial orientation—users find it difficult to get their bearings when transported to a new location [Bowman and Hodges 1997].

28.4 Indirect Control Patterns

Indirect Control Patterns provide control through an intermediary to modify an object, the environment, or the system. Indirect control is more abstract than selection, manipulation, and viewpoint control. Indirect control is ideal when an obvious spatial mapping does not exist or it is difficult to directly manipulate an aspect of the environment. Example uses include controlling the overall system, issuing commands, changing modes, and modifying non-spatial parameters.

Whereas the previously described techniques primarily describe both what should be done and how it is done, indirect control typically specifies only what should be done and the system determines how to do it. Because indirect control is not directly linked to that which it is controlling, signifiers such as the shape and size of controls, their visual representation and labeling, and apparent affordances of their underlying control structure are extremely important.

Indirect Control Patterns include the Widgets and Panels Pattern and Non-Spatial Control Pattern.

28.4.1 Widgets and Panels Pattern

Related Patterns

Hand Selection Pattern (Section 28.1.1), Pointing Pattern (Section 28.1.2), and Direct Hand Manipulation Pattern.

Description

The **Widgets and Panels Pattern** is the most common form of VR indirect control, and typically follows 2D desktop widget and panel/window metaphors. A **widget** is a geometric user interface element. A widget might only provide information to the user or might be directly interacted with by the user. The simplest widget is a single label that only provides information. Such a label can also act as a signifier for another widget (e.g., a label on a button). Many system controls are a 1D task and thus can be implemented with widgets such as pull-down menus, radio buttons, sliders, dials, and linear/rotary menus. **Panels** are container structures that multiple widgets and other panels can be placed upon. Placement of panels is important for being able to easily access them. For example, panels can be placed on an object, floating in the world, inside a vehicle, on the display, on the hand or input device, or somewhere near the body (e.g., a semicircular menu always surrounding the waist).

When to Use

Widgets and panels are useful for complex tasks where it is difficult to directly interact with an object. Widgets are normally activated via a Pointing Pattern (Section 28.1.2) but can also be combined with other selection options (e.g., select a property for objects within some defined volume). Widgets can provide more accuracy than directly manipulating objects [McMahan et al. 2014]. Use gestalt concepts of perceptual organization (Section 20.4) when designing a panel—use position, color, and shape to emphasize relationships between widgets. For example, put widgets with similar functions close together.

Limitations

Widgets and panels are not as obvious or intuitive as direct mappings and may take longer to learn. The placement of panels and widgets have a big impact on usability. If panels are not within personal space or attached to an appropriate reference frame (Section 26.3), then the widgets can be difficult to use. If the widgets are too high, gorilla arm (Section 14.1) can result for actions that take longer than a few seconds. If the widgets are in front of the body or head, then they can be annoying due to occluding the view (although this can be reduced by making the panel translucent).

If not oriented to face the user, information on the widget can be difficult to see. Large panels can also block the user's view.

Exemplar Interaction Techniques

2D desktop integration. Many panels are straightforward adaptations of windows from 2D desktop systems (Figure 28.8, left). An advantage of using desktop metaphors is their familiar interaction style and thus users have an instant understanding of how to use them. **2D desktop integration** brings existing 2D desktop applications into the environment via texture maps and mouse control with pointing. The system shown in Figure 3.5 also provides desktop metaphors such as double-clicking on a selected window header to minimize the window to a small cube (or double clicking the cube to make it large again). Bringing in such existing WIMP (Windows Icon Mouse Pointer) applications is typically not ideal from a 3D interaction perspective, but doing so does provide access to software that otherwise could only be accessed by exiting the virtual world. For example, an existing 2D calculator app could be available as a cube attached to the user's "belt." The user then simply selects and double clicks the cube to use the calculator tool without causing a break-in-presence.

Ring menus. A **ring menu** is a rotary 1D menu where a number of options are displayed concentrically about a center point [Liang and Green 1994, Shaw and Green 1994]. Options are selected by rotating the wrist until the intended option rotates into the center position or a pointer rotates to the intended item (Figure 28.8, center). Ring menus can be useful but can cause wrist discomfort when large rotations are required. Non-isomorphic rotations (Section 28.2.1) can be used to make small wrist rotations map to a larger menu rotation.

Pie menus. **Pie menus** (also known as marking menus) are circular menus with slice-shaped menu entries, where selection is based on direction, not distance. A disadvantage of pie menus is they take up more space than traditional menus (although using icons instead of text can help). Advantages of pie menus compared to traditional menus are that they are faster, more reliable with less error, and have equal distance for each option [Callahan 1988]. The most important advantage, however, may be that commonly used options are embedded into muscle memory as usage increases. That is, pie menus are self-revealing gestures by showing users what they can do and directing how to do it. This helps novice users become experts in making directional gestures. For example, if a desired pie menu option is in the lower-right quadrant, then the user learns to initiate the pie menu, and then move the hand to the lower right. After extended usage, users can perform the task without the need to



Figure 28.8 Three examples of hand-held panels with various widgets. The left panel contains standard icons and radio buttons. The center panel contains buttons, a dial, and a rotary menu that can be used as both a ring menu and a pie menu. The right image contains a color cube that the user is selecting a color from. (Courtesy of Sixense and Digital ArtForms)

look at the pie menu or even for it to be visible. Some pie menu systems only display the pie menu after a delay so the menu does not show up and occlude the scene for fast expert users—known as “mark ahead” since the user marks the pie menu element before it appears [Kurtenbach et al. 1993].

Hierarchical pie menus can be used to extend the number of options. For example, to change a color of an object to red, a user might (1) select the object to be modified, (2) press a button to bring up a property’s pie menu, (3) move the hand to the right to select “colors,” which brings up a “colors” pie menu, or (4) move the hand down to select the color red by releasing the button. If the user had to commonly change objects to the color red then she would quickly learn to simply move the hand to the right and then down after selecting the object and initiating the pie menu.

Gebhardt et al. [2013] compared different VR pie menu selection methods and found pointing to take less time and to be more preferred by users compared to hand projection (i.e., translation) or wrist roll (twist) rotation.

Color cube. A color cube is a 3D space that users can select colors from. Figure 28.8 (right) shows a 3D color cube widget—the color selection puck can be moved with the hand in 2D about the planar surface while the planar surface can be moved in and out.

Finger menus. Finger menus consist of menu options attached to the fingers. A pinch gesture with the thumb touching a finger can be used to select different options. Once learned the user is not required to look at the menus; the thumb simply touches the

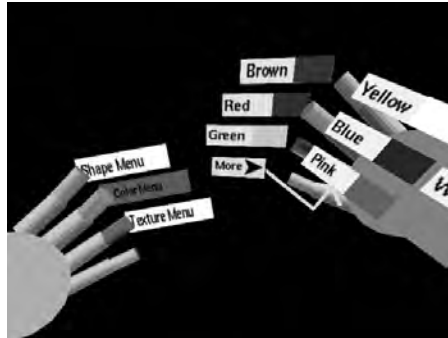


Figure 28.9 TULIP menus on seven fingers and the right palm. (From Bowman and Wingrave [2001])

appropriate finger. This prevents occlusion as well as decreases fatigue. The non-dominant hand can select a menu (up to four menus) and the dominant hand can then select one of four items within that menu.

For complex applications where more options are required, a TULIP menu (Three-Up, Labels In Palm) [Bowman and Wingrave 2001] can be used. The dominant hand contains three menu options at a time and the pinky contains a “more” option. When the “more” option is selected, then the other three finger options are replaced with new options. By placing the upcoming options on the palm of the hand, users know what options will become available if they select the “more” option (Figure 28.9).

Above-the-head widgets and panels. Above-the-head widgets and panels are placed out of the way above the user, and accessed via reaching up and pulling down the widget or panel with the non-dominant hand. Once the panel is released, then it moves up to its former location out of view. The panel might be visible above, especially for new users, but after learning where the panels are located relative to the body, the panel might be made invisible since the user can use his sense of proprioception to know where the panels are without looking. Miné et al. [1997] found users could easily select among three options above their field of view (up to the left, up in the middle, and up to the right).

Virtual hand-held panels. If a widget or panel is attached somewhere in the environment, then it can be difficult to find. If it is locked in screen space, then it can occlude the scene. One solution is to use **virtual hand-held panels** that have the advantage of always being available (as well as turned off) at the click of a button. Attaching the panel to the hand greatly diminishes many of the problems of world-spaced panels (e.g., panels that are difficult to read or get in the way of other objects can be reori-

ented and moved in an intuitive way without any cognitive effort). The panel should be attached to the non-dominant hand, and the panel is typically interacted with by pointing with the dominant hand (Figure 28.8). Such an interface provides a “double dexterity” where the panel can be brought to the pointer and the pointer can be brought to the panel.

Physical panels. Virtual panels offer no physical feedback, which can make it difficult to make precise movements. A **physical panel** is a real-world tracked surface that the user carries and interacts with via a tracked finger, object, or stylus. Using a physical panel can provide fast and accurate manipulation of widgets [Stoakley et al. 1995, Lindeman et al. 1999] due to the surface acting as a physical constraint when touched. The disadvantage of a physical panel is that users can become fatigued from carrying it and it can be misplaced if set down. Providing a physical table or other location to set the panel on can help reduce this problem where the panel still travels with the user when virtually moving. Another option is to strap the panel to the forearm [Wang and Lindeman 2014]. Alternatively, the surface of the arm and/or hand can be used in place of a carried panel.

28.4.2 Non-Spatial Control Pattern

Related Patterns

Multimodal Pattern (Section 28.5.3).

Description

The **Non-Spatial Control Pattern** provides global action performed through description instead of a spatial relationship. This pattern is most commonly implemented through speech or gestures (Section 26.4).

When to Use

Use when options can be visually presented (e.g., gesture icons or text to speak) and appropriate feedback can be provided. This pattern is best used when there are a small number of options to choose from and when a button is available to push-to-talk or push-to-gesture. Use voice when moving the hands or the head would interrupt a task.

Limitations

Gestures and accents are highly variable from user to user and even for a single user. There is often a trade-off of accuracy and generality—the more gestures or words to be recognized then the less accurate the recognition rate (Section 26.4.2). Defining each gesture or word to be based on its invariant properties and to be independent from

others makes the task easier for both the system and the user. System recognition of voice can be problematic when many users are present or there is a lot of noise. For important commands, verification may be required and can be annoying to users.

Even for hypothetically perfectly working systems, providing too many options for users can be overwhelming and confusing. Typically, it is best to only recognize a few options that are simple and easy to remember.

Depending too heavily on gestures can cause fatigue, especially if the gestures must be made often and above the waist. Some locations are inappropriate for speaking (e.g., a library) and some users are uncomfortable speaking to a computer.

Exemplar Interaction Techniques

Voice menu hierarchies. Voice menu hierarchies [Darken 1994] are similar to traditional desktop menus where submenus are brought up after higher-level menu options are selected. Menu options should be visually shown to users so users explicitly know what options are available. See Section 26.4.2 for more information about speech recognition.

Gestures. Gestures (Section 26.4.1) can work well for non-spatial commands. Gestures should be intuitive and easy to remember. For example, a thumbs-up to confirm, raising the index finger to select “option 1,” or raising the index and middle finger to select “option 2.” Visual signifiers showing the gesture options available should be presented to the user, especially for users who are learning the gestures. The system should always provide feedback to the user when a gesture has been recognized.

28.5 Compound Patterns

Compound Patterns combines two or more patterns into more complicated patterns. Compound Patterns include the Pointing Hand Pattern, World-in-Miniature Pattern, and Multimodal Pattern.

28.5.1 Pointing Hand Pattern

Related Patterns

Pointing Pattern (Section 28.1.2), Direct Hand Manipulation Pattern (Section 28.2.1), and Proxy Pattern (Section 28.2.2).

Description

Hand selection (Section 28.1.1) has limited reach. Pointing (Section 28.1.2) can be used to select distant objects and does not require as much hand movement. However, pointing is often (depending on the task) not good for spatially manipulating objects

because of the radial nature of pointing (i.e., positioning is done primarily by rotation about an arc around the user) [Bowman et al. 2004]. Thus, pointing is often better for selection and a virtual hand is better for manipulation. The **Pointing Hand Pattern** combines the Pointing and Direct Hand Manipulation Patterns together so that far objects are first selected via pointing and then manipulated as if held in the hand. The user's real hand can also be thought of (and possibly rendered) as a proxy to the remote object.

When to Use

Use the Pointing Hand Pattern when objects are beyond the user's personal space.

Limitations

This pattern is typically not appropriate for applications requiring high interaction fidelity.

Exemplar Interaction Techniques

HOMER. The **HOMER technique** (Hand-centered Object Manipulation Extending Ray-casting) [Bowman and Hodges 1997] causes the hand to jump to the object after selection by pointing, enabling the user to directly position and rotate the object as if it were held in the hand. The scaled HOMER technique [Wilkes and Bowman 2008] scales object movement based on how fast the hand is moving (i.e., object translation is based on hand velocity). Fast hand motions enable gross manipulation whereas slow hand motions enable more precise manipulation, providing flexibility of object placement.

Extender grab. The **extender grab** [Miné et al. 1997] maps the object orientation to the user's hand orientation. Translations are scaled depending on the distance of the object from the user at the start of the grab (the further the object, the larger the scale factor).

Scaled world grab. A **scaled world grab** scales the user to be larger or the environment to be smaller so that the virtual hand, which was originally far from the selected object, can directly manipulate the object in personal space [Miné et al. 1997]. Because the scaling is about the midpoint between the eyes, the user often does not realize scaling has taken place. If the interpupillary distance is scaled in the same way, then stereoscopic cues will remain the same. Likewise, if the virtual hand is scaled appropriately, then the hand will not appear to change size. What is noticeable is head-motion parallax due to the same physical head movement mapping to a larger movement relative to the scaled-down environment.

28.5.2 World-in-Miniature Pattern

Related Patterns

Image-Plane Selection Pattern (Section 28.1.4), Proxy Pattern (Section 28.2.2), 3D Multi-Touch Pattern (Section 28.3.3), and Automated Pattern (Section 28.3.4).

Description

A **world-in-miniature** (WIM) is an interactive live 3D map—an exocentric miniature graphical representation of the virtual environment one is simultaneously immersed in [Stoakley et al. 1995]. An avatar or “doll” representing the self matches the user’s movements giving an exocentric view of oneself in the world. A transparent viewing frustum extending out from the doll’s head showing the direction the user is looking along with the user’s field of view is also useful. When the user moves his smaller avatar, he also moves in the virtual environment. When the user moves a proxy object in the WIM, the object also moves in the surrounding virtual environment. Multiple WIMs can be created to view the world from different perspectives, the VR equivalent of 3D CAD Windowing System.

When to Use

WIMs work well as a way to provide situational awareness via an exocentric view of oneself and the surrounding environment. WIMs are also useful to quickly define user-defined proxies and to quickly move oneself.

Limitations

A straightforward implementation can cause confusion due to the focus being on the WIM, not on the full-scale virtual world.

A challenge with rotating the WIM is that it results in a lack of directional compliance (Section 25.2.5); translating and rotating a proxy within the WIM can be confusing when looking at the WIM and larger surrounding world from different angles. Because of this challenge, the orientation of the WIM might be directly linked to the larger world orientation to prevent confusion (a forward-up map; Section 22.1.1). However, this comes at the price of not being able to lock in a specific perspective (e.g., to keep an eye on a bridge as the user is performing a task at some other location) as the user reorients himself in the larger space. For advanced users, there should be an option to turn on/off the forward-up locked perspective.

Exemplar Interaction Techniques

Voodoo doll technique. The voodoo doll technique uses image-plane selection techniques (Section 28.1.3) by having users temporarily create miniature hand-held proxies of distant objects called **dolls** [Pierce et al. 1999]. Dolls are selected and used in

pairs (with each doll representing a different object or set of objects in the scene). The doll in the non-dominant hand (typically representing multiple objects as a partial WIM) acts as a frame of reference, and the doll in the dominant hand (typically representing a single object) defines the position and orientation of its corresponding world object relative to the frame-of-reference doll. This provides the capability for users to quickly and easily position and orient objects relative to each other in the larger world. For example, a user can position a lamp on a table by first selecting the table with the non-dominant hand and then selecting the lamp with the dominant hand. Now, the user simply places the lamp doll on top of the table doll. The table does not move in the larger virtual world and the lamp is placed relative to that table.

Moving into one's own avatar. By moving one's own avatar in the WIM, the user's egocentric viewpoint can be changed (i.e., a proxy controlling one's own viewpoint). However, directly mapping the doll's orientation to the viewpoint can cause significant motion sickness and disorientation. To reduce sickness and confusion, the doll pose can be independent of the egocentric viewpoint; then when the doll icon is released or the user gives a command, the system automatically animates/navigates or teleports the user into the WIM via an automated viewpoint control technique (Section 28.3.4) where the user "becomes" the doll [Stoakley et al. 1995]. This avoids the problem of shifting the user's cognitive focus back and forth from the map to the full-scale environment. That is, users think in terms of either looking at their doll in an exocentric manner or looking at the larger surrounding world in an egocentric manner, but not both simultaneously. In practice, users do not perceive a change in scale of either themselves or the WIM; they express a sense of going to the new location. Using multiple WIMs allows each WIM to act as a portal to a different, distant space.

Viewbox. The viewbox [Mlyniec et al. 2011] is a WIM that also uses the Volume-Based Selection Pattern (Section 28.1.4) and 3D Multi-Touch Pattern (Section 28.3.3). After capturing a portion of the virtual world via volume-based selection, the space within that box is referenced so that it acts as a real-time copy. The viewbox can then be selected and manipulated like any other object. The viewbox can be attached to the hand or body (e.g., near the torso acting as a belt tool). In addition, the user can reach inside the space and manipulate that space via 3D Multi-Touch (e.g., translate, rotate, or scale) in the same way that she can manipulate the larger surrounding space. Because the viewbox is a reference, any object manipulation that occurs in either space occurs in the other space (e.g., painting on a surface in one space results in painting in both spaces). Note due to the object being a reference, care must be taken to stop the recursive nature of the view, otherwise an infinite number of viewboxes within other viewboxes can result.

28.5.3 Multimodal Pattern

Description

Multimodal Patterns integrate different sensory/motor input modalities together (e.g., speech with pointing). Section 26.6 categorizes how different modalities can work together for interaction.

When to Use

Multimodal Patterns are appropriate when multiple facets of a task are required, reduction in input error is needed, or no single input modality can convey what is needed.

Limitations

Multimodal techniques can be difficult to implement due to the need to integrate multiple systems/technologies into a single coherent interface and the risk of multiple points of failure. Techniques can also be confusing to users if not implemented well. In order to keep interaction as simple as possible, a multimodal technique should not be used unless there is a good reason to do so.

Exemplar Interaction Techniques

Immersive “put-that-there.” The classic example of the Multimodal Pattern is a **put-that-there** interface [Bolt 1980] that uses a combination of the Pointing Pattern (to select the “that” and “there”) and the Non-Spatial Control Pattern via voice (to select the verb “put”). Neely et al. [2004] implemented an immersive form of a “put-that-there” style of region definition. With this technique, the user names and defines the vertices of a polygonal region on a terrain via pointing and speaking. An example is “Make Target Zone Apple from here (pointing gesture) to here (pointing gesture) . . . and to here (pointing gesture).” Figure 32.2 shows the architectural block diagram for this system.

For manipulating a single object, a “that-moves-there” interface (Section 26.6) where the object is selected before specifying the action is often more efficient (Section 26.5).

Automatic mode switching. One knows when the hand is within the center of vision and can effortlessly bring the hand into view when needed. Mode switching can take advantage of this. For example, an application might switch to an image-plane selection technique (Section 28.1.3) when the hand is moved to the center of vision and to a pointing selection technique (Section 28.1.2) when moved to the periphery (since the source of a ray pointer does not need to be seen).

Interaction: Design Guidelines

Just as there is no single tool in the real world that solves every problem, there is no single input device, concept, interaction pattern, or interaction technique that is best for all VR applications. Although it is generally better to use the same interaction metaphor across different types of tasks, that is not always possible or appropriate. Interaction designers should take into account the specific task when choosing, modifying, or creating new interactions.

29.1 Human-Centered Interaction (Chapter 25)

Intuitiveness (Section 25.1)

- Focus on making interfaces intuitive—that is, design interfaces that can quickly be understood, accurately predicted, and easily used.
- Use interaction metaphors (concepts that exploit specific knowledge that users already have of other domains) to help users quickly develop a mental model of how an interface works.
- Include within the virtual world everything that is needed to help users form a consistent conceptual model of how things work (e.g., within world tutorials). Users should not have to rely on any external explanation.

Norman's Principles of Interaction Design (Section 25.2)

- Practice human-centered design and follow well-known general principles to help users create simplified mental models of how the interactions work. These include consistent affordances, unambiguous signifiers, constraints to guide actions and ease interpretation, obvious and understandable mappings, and immediate and useful feedback.

Affordances (Section 25.2.1)

- Remember that an affordance is not a property of an object, but is a relationship between an object and a user. Affordances don't only depend on the object being afforded, but also depend on the user; affordances may be different for users with different capabilities.

Signifiers (Section 25.2.2)

- Make affordance perceivable through signifiers. A good signifier informs users what is possible before interacting.
- Make obvious to the user the state of the current interaction mode.

Constraints (Section 25.2.3)

- Where appropriate, add constraints to limit possible actions and to improve accuracy and efficiency.
- Use constraints to add realism (e.g., do not allow users to travel through walls).
- Do not assume real-world rules such as gravity are always appropriate. For example, hanging tools in the space around the user makes them easier to grab.
- Use signifiers wisely to prevent users from making the wrong assumptions about constraints. If a user doesn't know what to do, she is effectively constrained.
- Make constraints consistent so learning can be transferred across tasks.
- Consider allowing expert users to remove constraints for advanced interactions.

Feedback (Section 25.2.4)

- Use feedback substitution when haptics are not available. For example, use audio and highlighting to signify touching of objects.
- Do not overwhelm users with too much feedback.
- Consider putting information in front of the user in the torso reference frame near the waist rather than on a heads-up display in the head reference frame.
- If a heads-up display in the head reference frame must be used, only present minimal information on the display.
- Provide the capability to turn on/off (or make visible/invisible) widgets, tools, and interface cues.

Mappings (Section 25.2.5)

- To maximize performance and satisfaction, maintain directional compliance, nulling compliance, and temporal compliance.
- Focus first on creating mappings that have directional compliance (the direction of sensory feedback should match the direction of the interface device) so users can anticipate motion in response to their physical input.
- For fully direct interactions, maintain position compliance (i.e., make the virtual position of objects match the physical position of the device).
- If position compliance is not appropriate, use nulling compliance (the virtual object should return to its original location when the device returns to its original location).
- Use nulling compliance to take advantage of muscle memory.
- Choose absolute input devices over relative input devices in order to maintain compliance.
- If the results of an interaction cannot be immediately computed (i.e., a lack of temporal compliance), then provide some form of immediate feedback to inform the user the problem is being worked on.
- For non-spatial mappings, use commonly accepted metaphors (e.g., up is more, down is less).

Direct vs. Indirect Interaction (Section 25.3)

- Use tools to extend one's reach so users feel like they are directly interacting.
- Use direct, semi-direct, and indirect interactions where appropriate and not where not appropriate. Do not try to force everything to be direct.

The Cycle of Interaction (Section 25.4)

- To design or improve interactions, use Norman's seven stages of interaction to break up typically subconscious processes into explicit steps.
- Think about what stages are missing or what stages do not work well that make interaction difficult. Then add or modify signifiers, constraints, mappings, and feedback as appropriate for each stage.
- Use the seven stages of interaction with task analysis as a stepping stone toward implementation.

The Human Hands (Section 25.5)

- Support interactions with two hands where appropriate.
- Do not assume just because two hands are used for an interface that the interface will be better. Two-handed interfaces can be difficult to use if designed inappropriately.
- Feedback from users is even more essential for two-handed interaction than one-handed interaction.
- Have the non-dominant hand maintain the reference frame so the dominant hand can work precisely without having to work in a locked position.
- Design bimanual interactions to work together in a fluid manner, switching between symmetric and asymmetric modes as appropriate for the task.

29.2 VR Interaction Concepts (Chapter 26)

Interaction Fidelity (Section 26.1)

- Consider using realistic interactions for training applications, simulations, surgical applications, therapy, and human-factors evaluations.
- Consider using non-realistic interactions for increasing performance and minimizing fatigue.
- Use magical interactions to enhance the user experience, circumvent the limitations of the real world, and teach abstract concepts.
- Consider interaction metaphors as a source of inspiration. Unless realistic interaction is a primary goal, use intuitive and useful magical techniques.

Proprioceptive and Egocentric Interaction (Section 26.2)

- Exploit the one real object every user has—the human body.
- Use torso tracking in addition to head and hands tracking whenever possible to maximize proprioception. Alternatively, chair rotation can be tracked to estimate torso rotation.
- Place commonly used tools relative to the body to take advantage of muscle memory so users do not have to take visual attention away from that which is being worked on (tools do not even have to be within the user's field of view).

- Don't assume exocentric perspectives preclude egocentric perspectives. Take advantage of egocentric intuition even when designing an exocentric experience.

Reference Frames (Section 26.3)

- Provide the capability for users to think and interact in an exocentric virtual-world reference frame when the intent is for users to create content over a wide area, form a cognitive map of the environment, determine their own global position, or plan travel on a large scale.
- Be careful of placing direct interfaces in virtual-world reference frames as they can be awkward to reach without easy and precise navigation capability.
- Draw rest frames (e.g., an automobile interior, a cockpit, or non-realistic cues) in the real-world reference frame for users to feel stabilized in space and to reduce motion sickness.
- Provide a place for users to set physical devices when not needed and render those objects in the real-world reference frame so they can easily be seen and picked up.
- Place information, interfaces, and tools relative to the body by placing them in the torso reference frame.
- Allow advanced users to turn on/off or make invisible (but still usable) items in the torso, hand, and head reference frames.
- Place a visual representation of hand controllers in the torso reference frame for non-tracked controllers and in the hand reference frame for tracked hand-held controllers.
- Place signifiers in the hand reference frame pointing to buttons, analog sticks, and/or fingers so it is obvious what they do. Provide the capability for users to turn them on and off.
- Minimize cues in the head reference frame for anything other than a pointer if head tracking is used for input.

Speech and Gestures (Section 26.4)

- Use explicit visual signifiers (e.g., a list of words or hand icons) to portray what speech or gesture commands are available. Provide feedback by highlighting the selected option.
- Have users verify important commands to prevent major errors.

- Use a small number of words or gestures that are well defined, natural, easy for the user to remember, and easy for the system to recognize.
- When more than one person will be present in the same space, use push-to-talk and/or push-to-gesture to prevent accidental unintended commands.
- Use direct structural gestures for immediate system response.
- When users have their own unique system, use speaker-dependent recognition if they are willing to train the system and adaptive recognition if they are not.
- Reduce the potential for error by only allowing a subset of vocabulary to be recognized depending upon the context.

Modes and Flow (Section 26.5)

- When multiple interaction modes are used, make it clear to the user what the current mode is.
- Use object-action (or selection-manipulation) sequences over action-object sequences.
- Enable smooth and easy transition between selecting an object and manipulating or using that object.
- Minimize distractions to enhance the flow of interactions and allow for full attention to the primary task.
- Design interactions so users do not have to physically (whether the eyes, head, or hands) or cognitively move between tasks.
- Use lightweight mode switching, physical props, and multimodal techniques to help maintain the flow of interaction.

Multimodal Interaction (Section 26.6)

- Use a single specialized input modality when it is clear that single modality is best for the task and there is no reason to include other modalities. Do not add a modality just for the sake of adding a modality.
- Consider using equivalent input modalities when user preferences are strongly divided.
- Use redundant input modalities to reduce noise and ambiguous signals.
- Use concurrent input modalities to improve efficiency by enabling the user to perform two interactions simultaneously.
- Use complementarity input modalities for “put-that-there” or “that-moves-there” types of interfaces.

- Allow transfer of input modalities when one modality device is unreliable so users will not have to start over if there is a failure.

Beware of Sickness and Fatigue (Section 26.7)

- Be extra careful of viewpoint control techniques that can induce motion sickness. Study Part III and follow the guidelines in Chapter 19 to minimize adverse health effects.
- When motion sickness is a primary concern (e.g., for users new to VR or a general audience), only use one-to-one mapping of real head motion or teleportation.
- Avoid using and creating interactions that require the user to hold the hands up high or in front of the body for more than a few seconds at a time. Use devices that do not require line of sight so interactions can be performed comfortably in the lap or from the sides of the body.

Visual-Physical Conflict and Sensory Substitution (Section 26.8)

- Enforce physics constraints when hand penetration into objects is shallow so the hand does not pass through the object surface. Do not enforce physics constraints when penetration is deep; instead allow the hand to pass through virtual objects.
- Consider drawing two virtual hands for a single physical hand—one that penetrates objects and one that doesn't.
- Use highlighting to convey an object is selectable when the hand is close to the object.
- Use audio to convey collisions.
- Use passive haptics or vibrotactile haptics whenever possible.
- If training transfer is not important, use ghosting to signify a new potential position until the user confirms placement.

29.3 Input Devices (Chapter 27)

Input Device Characteristics (Section 27.1)

- Match the interaction technique with the device and match the device with the interaction technique. Understand the different device characteristics and classes in order to determine what is best for the project.

- Use 6 DoF devices whenever possible and then decrease DoFs in software where appropriate.
- Choose input devices that work in a user's entire personal space, do not require line of sight, are robust to various lighting conditions, and work for all hand orientations.
- Use buttons for tasks that are binary, when the action needs to occur often, when immediate response and reliability are required, when abstract interactions are appropriate, and when physical feedback of feeling the button pressed/released is important.
- Do not overwhelm users with too many buttons.
- Use bare-hand systems, gloves, and/or haptic devices that correspond to virtual objects when a high sense of realism and presence is important.

Classes of Hand Input Devices (Section 27.2)

- If not sure what to use or there is no strong preference, then start with tracked hand-held controllers. They are currently the best option for a majority of interactive VR experiences.
- For public location-based entertainment, consider building customized interfaces such as world-grounded devices that are optimized for the particular experience.
- Attach labels to virtual representations of controllers to signify what the controls do.
- Use tracked hand-held controllers when the user commonly holds virtual devices to enhance presence through physical touch.
- Don't assume gloves are only for full hand tracking. Consider using pinch gloves that don't require holding a physical controller but have the advantages of buttons via pressing the fingers together.

Classes of Non-hand Input Devices (Section 27.3)

- If eye tracking is available, do not overuse it. Instead use eye gaze for specialized tasks and in subtle ways (e.g., have virtual characters respond when looked at).
- When designing eye gaze interactions, maintain the natural function of the eyes, augment rather than replace, focus on interaction design, improve the

interpretation of eye movements, choose appropriate tasks, use passive gaze over active gaze, and leverage gaze information for other interactions.

- Use microphones that are specially designed for speech recognition.

29.4 Interaction Patterns and Techniques (Chapter 28)

Selection Patterns (Section 28.1)

- When interactions do not need to be realistic, use the Pointing Pattern or the Image-Plane Selection Pattern.

Hand Selection Pattern (Section 28.1.1)

- Use the Hand Selection Pattern when interactions need to be realistic.
- For high interaction fidelity, use a realistic virtual hand to select objects.
- For mid-interaction fidelity, use hands without arms to reach just beyond personal space (i.e., near action space).
- Consider the go-go technique for personal space and midrange action space.

Pointing Pattern (Section 28.1.2)

- For more controlled pointing, consider using a precision mode.
- For selecting objects that are small, consider using pointing with object snapping.
- Do not use dwell selection unless there is no other good way to provide a signal or there is another good reason to do so.
- Do not use eye tracking for selecting objects unless there is a good reason to do so.

Image-Plane Selection Pattern (Section 28.1.3)

- For easy-to-use touch at a distance, use image-plane selection. However, do not use when the user must frequently select objects as gorilla arm can result.

Volume-Based Selection Pattern (Section 28.1.4)

- Use the Volume-Based Selection Pattern when selecting data/space that has no geometric surfaces.
- Be careful of requiring volume-based selection for novice users.

Manipulation Patterns (Section 28.2)

Direct Hand Manipulation Pattern (Section 28.2.1)

- Use the Direct Hand Manipulation Pattern unless there is a reason not to as it is more efficient and satisfying than other manipulation patterns.
- For high interaction fidelity, use a virtual hand for both selection and manipulation.
- Consider using non-isomorphic rotations to reduce clutching and to increase performance and precision.

Proxy Pattern (Section 28.2.2)

- Use the Proxy Pattern to intuitively manipulate remote objects or when the user scales himself or the world.
- Use tracked physical props for direct action-task correspondence. Such props result in an easy-to-use interface that does not require any training, facilitate natural two-handed interactions, and provide tactile feedback to the user.

3D Tool Pattern (Section 28.2.3)

- Use the 3D Tool Pattern to enhance the capability of the hands to manipulate objects.
- Use signifiers for object-attached tools to make it obvious how to use those tools to manipulate the object.
- To enable precise modeling and reduce complexity, use jigs that enable user-controlled constraints, snapping, and discrete manipulations.

Viewpoint Control Patterns (Section 28.3)

- Be especially careful of sickness and injury when choosing, designing, and implementing viewpoint control techniques, especially for users new to VR. See Part III.

Walking Pattern (Section 28.3.1)

- Use the Walking Pattern when high biomechanical symmetry and high presence is desired, fatigue is not a concern, and safety measures to prevent tripping, physical collisions, and falling are covered.
- Use real walking when the physically tracked space is as large as or larger than the virtual walkable space, and both spatial understanding and minimizing sickness are important.

- Use redirected walking when the physically tracked space is smaller than the virtual walkable space.
- Use walking in place when the physically tracked space is small or safety is a concern.
- Use a treadmill when walking/running vast distances is required.

Steering Pattern (Section 28.3.2)

- Use the Steering Pattern when sickness is not a primary concern, interaction fidelity is not important, acceleration/deceleration can be minimized, and real-world stabilized cues can be provided.
- Make steering as simple as possible to minimize cognitive load so the user can focus on spatial knowledge acquisition and information gathering.
- Provide visual cues to help users know what direction is forward.
- Consider not using virtual rotations if torso/chair tracking is available and the system is wireless.
- If virtual rotations are required and the user can be constrained to the ground, use dual analog sticks.

3D Multi-Touch Pattern (Section 28.3.3)

- For applications not requiring high interaction fidelity, use the 3D Multi-Touch Pattern when creating assets, manipulating abstract data, viewing scientific datasets, or rapidly exploring large and small areas of interest from arbitrary viewpoints.
- Consider adding constraints (e.g., force uprightness, limit scale, and/or disable rotations) for novice users.
- Provide a visual indication of the center of rotation/scale.

Automated Pattern (Section 28.3.4)

- Use the Automated Pattern when free exploration of the environment is not desired or not possible.
- Use teleportation when traveling large distances, or between worlds, when efficiency is important, or when motion sickness must be minimized. Do not use if spatial orientation must be maintained.
- Use smooth transitions when maintaining spatial orientation is the primary concern.

- When passively moving the user, reduce motion sickness by keeping visual velocity as constant as possible, providing stable real-world reference cues (e.g., a cockpit), and/or providing a leading indicator.

Indirect Control Patterns (Section 28.4)

- Use an Indirect Control Pattern when spatial mappings are not appropriate or details of how something is done are not important to the user. Uses include controlling the overall system, issuing commands, changing modes, and changing non-spatial parameters.
- For indirect control, make signifiers obvious, such as the shape and size of controls, their visual representation and labeling, and apparent affordances of their underlying control structure.

Widgets and Panels Pattern (Section 28.4.1)

- Use the Widgets and Panels Pattern when it is difficult to directly interact with an object.
- Where appropriate, use well-known 2D interaction metaphors such as pull-down/pop-up menus, radio buttons, and checkboxes.
- Use gestalt concepts of perceptual organization when designing a panel—use position, color, and shape to emphasize relationships between widgets. For example, put widgets with similar functions close together.
- Place widgets and panels in a way that is easy for users to access (e.g., on the non-dominant hand or in the torso reference frame).
- For commonly used commands, use pie/marking menus to teach gestures and to embed those gestures into muscle memory.
- For pie/marking menus, use pointing over projection or roll.
- If pinch gestures are available, place menu options on the fingers.
- Consider placing panels or widgets above the head so users can pull them down when needed.
- Consider placing a panel in the non-dominant hand that can be turned on or off, and where widgets on the panel are controlled by the dominant hand.
- For 2D tasks that require precision, consider using a physical panel the user holds in the non-dominant hand or that is attached to the forearm.

Non-Spatial Control Pattern (Section 28.4.2)

- Use the Non-Spatial Control Pattern for global action performed through description instead of a spatial relationship.
- Use real-world words and gestures that are intuitive and easy to remember. Keep the number of options small and simple.
- Provide signifiers (e.g., icons for gestures or a list of words for voice) to remind non-expert users what options are available.
- Always provide some form of feedback.
- When accuracy is more important than speed, verify commands. Make the confirmation process fast and do not require precision. Examples include clicking a physical button or saying “confirm.”
- Use push-to-talk or push-to-gesture to prevent accidental unintended commands.
- Use voice control when moving the hands or the head would interrupt a task.
- Be careful of depending on voice recognition when multiple people are in the same physical environment or there is a lot of noise.

Compound Patterns (Section 28.5)**Pointing Hand Pattern (Section 28.5.1)**

- When high interaction fidelity is not required, use the Pointing Hand Pattern to select objects at a distance but to manipulate them as if held in the hand.

World-in-Miniature Pattern (Section 28.5.2)

- Use the World-In-Miniature Pattern to provide situational awareness, to quickly define user-defined proxies, or to quickly move oneself.
- Consider using the equivalent of a forward-up map for the world-in-miniature so that the orientation of the map matches the orientation of the larger world. When appropriate, provide the capability to turn this feature off.
- To reduce sickness, do not directly map the user’s doll in the world-in-miniature to the user’s movement. Instead, animate/navigate or teleport the viewpoint into the doll when commanded by the user.

Multimodal Pattern (Section 28.5.3)

- Use the Multimodal Pattern when multiple facets of a task are required, when reduction in input error is needed, or when no single modality can convey what is needed.

- Only use a multimodal technique when there is a good reason to do so. When used, keep interactions as simple as possible.
- Consider using automatic mode switching when each technique is only usable in specific situations and those situations are clear to the user.

PART

ITERATIVE DESIGN

Design thinking takes a solution-focused approach to problem solving, working collaboratively to iterate an endless, shifting path toward perfection. It works toward product goals via specific ideation, prototyping, implementation, and learning steps to bring the appropriate solution to light.

—Jeff Gothelf and Josh Seiden [2013]

Up to this point, this book has provided a background on the mechanisms of perception, VR sickness, content creation, and interaction. However, such mechanisms are not entirely understood and very much depend on the goals and design of the VR project. Furthermore, assumptions about the real world or how we perceive and interact with it do not always hold in VR. One cannot simply look up the numbers when designing VR experiences or get the experience perfect on the first attempt. VR design is also very different than 2D desktop or mobile application design, and there is not nearly as much known or standardized about VR. VR design must be iteratively created based off of continual redesign, prototyping, and feedback from real users. In fact, the design of many VR experiences is due to “accidental discoveries” that were not initially intended (Andrew Robinson and Sigurdur Gunnarsson, personal communication, May 11, 2015). This chapter focuses on iterative design concepts to help the team move toward ideal experiences as quickly and efficiently as possible.

With new development platforms such as Unity and other tools, VR experiences can be created in a fraction of the time they could be created compared to just a few years ago. Experienced designers can now literally create a simple VR experience in hours, without any programming experience required. Some modifications can be as easy as a click of a button, and some assets and behaviors built by others can be integrated with

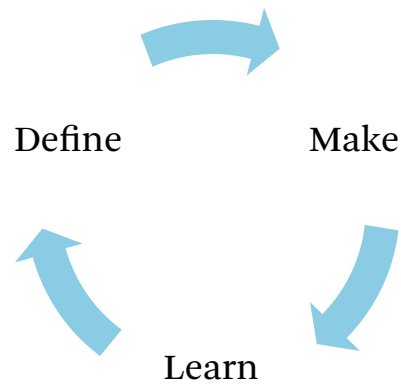


Figure VI.1 High-level iterative process common to all VR development.

basic programming skills. Because of how quickly simple VR experiences can be built and modified, prototyping basic individual components can often be done in a day or less. However, to go beyond the basics, designers should either be programmers themselves, learn basic programming skills for simple prototyping and testing, or work directly in pairs with programmers. Note that this is not to say professional programmers are not required for creating compelling VR experiences. Integrating advanced/innovative features and behaviors, understanding software architecture, and writing clean code takes time to do properly and are essential for creating efficient and maintainable code.

The Define-Make-Learn Cycle

Although a single overall detailed process is not appropriate for VR design, there is general agreement that **iterative design** consists of defining the project, building prototypes, learning from users, and continually improving upon previous ideas. Careful consideration of these stages is critical to optimizing user experience and comfortable VR.

Part VI discusses the philosophy of iterative design and then the three iterative stages (Figure VI.1) as summarized below.

1. **The Define Stage.** This stage attempts to answer the question “What do we make?” and includes everything from the high-level vision to listing requirements.
2. **The Make Stage.** This stage answers the question “How do we make it?” and then proceeds to make it.