3D User Interfaces

Supercell
Exhibit 1009
Page 1

3D User Interfaces

Theory and Practice

Doug A. Bowman
Ernst Kruijff
Joseph J. LaViola, Jr.
Ivan Poupyrev

Addison-Wesley

Boston • San Francisco • New York • Toronto • Montreal
London • Munich • Paris • Madrid • Capetown
Sydney • Tokyo • Singapore • Mexico City

Many of the designations used by manufacturers and sellers to distinguish their products are claimed as trademarks. Where those designations appear in this book, and Addison-Wesley was aware of a trademark claim, the designations have been printed with initial capital letters or in all capitals.

The authors and publisher have taken care in the preparation of this book, but make no expressed or implied warranty of any kind and assume no responsibility for errors or omissions. No liability is assumed for incidental or consequential damages in connection with or arising out of the use of the information or programs contained herein.

The publisher offers discounts on this book when ordered in quantity for bulk purchases and special sales. For more information, please contact:

U.S. Corporate and Government Sales
(800) 382-3419
corpsales@pearsontechgroup.com

For sales outside of the U.S., please contact:

International Sales
(317) 581-3793
international@pearsontechgroup.com

Visit Addison-Wesley on the Web: www.awprofessional.com

Library of Congress Cataloging-in-Publication Data

3D user interfaces : theory and practice / Doug A. Bowman . . . [et al.].
p. cm.
Includes bibliographical references and index.
ISBN 0-201-75867-9 (hardbound : alk. paper)
1. User interfaces (Computer systems) 2. Three-dimensional display systems. I. Bowman, Doug A., 1971–

QA76.9.U83A14 2004
005.4'37—dc22

2004008403

Copyright © 2005 by Pearson Education, Inc.

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, photocopying, recording, or otherwise, without the prior consent of the publisher. Printed in the United States of America. Published simultaneously in Canada.

For information on obtaining permission for use of material from this work, please submit a written request to:

Pearson Education, Inc.
Rights and Contracts Department
75 Arlington Street, Suite 300
Boston, MA 02116
Fax: (617) 848-7047

ISBN 0-201-75867-9
Text printed on recycled paper
1 2 3 4 5 6 7 8 9 10—CRS—08070605040
First printing, July 2004

For Dawn, Drew, and Caroline, my true joys on this earth.
Doug

To my family, for their faith, affection, and support.
Ernst

To my family with love—they are my life.
Joe

To my parents for bringing me up.
Ivan

Contents

Foreword *xv*
Preface *xix*

PART I **FOUNDATIONS OF 3D USER INTERFACES** **1**
Chapter 1 **Introduction to 3D User Interfaces** **3**
1.1. What Are 3D User Interfaces? 3
1.2. Why 3D User Interfaces? 4
1.3. Terminology 6
1.4. Application Areas 8
1.5. Conclusion 9

Chapter 2 **3D User Interfaces: History and Roadmap** **11**
2.1. History of 3D UIs 11
2.2. Roadmap to 3D UIs 14
  2.2.1. Areas Informing the Design of 3D UIs 15
  2.2.2. 3D UI Subareas 18
  2.2.3. Areas Impacted by 3D UIs 22
2.3. Scope of This Book 25
2.4. Conclusion 26

PART II **HARDWARE TECHNOLOGIES FOR 3D USER INTERFACES** **27**
Chapter 3 **3D User Interface Output Hardware** **29**
3.1. Introduction 29
  3.1.1. Chapter Roadmap 30
3.2. Visual Displays 31
  3.2.1. Visual Display Characteristics 31
  3.2.2. Depth Cues 34
  3.2.3. Visual Display Device Types 40

3.3. Auditory Displays 59
  3.3.1. 3D Sound Localization Cues 59
  3.3.2. 3D Sound Generation 62
  3.3.3. Sound System Configurations 64
  3.3.4. Audio in 3D Interfaces 66
3.4. Haptic Displays 68
  3.4.1. Haptic Cues 68
  3.4.2. Haptic Display Characteristics 70
  3.4.3. Haptic Display Types 71
  3.4.4. Haptic Displays in 3D Interfaces 77
3.5. Design Guidelines: Choosing Output Devices for 3D Interfaces 77
3.6. Conclusion 83

Chapter 4 3D User Interface Input Hardware 87
4.1. Introduction 87
  4.1.1. Input Device Characteristics 88
  4.1.2. Chapter Roadmap 89
4.2. Desktop Input Devices 90
  4.2.1. Keyboards 91
  4.2.2. 2D Mice and Trackballs 91
  4.2.3. Pen-Based Tablets 92
  4.2.4. Joysticks 93
  4.2.5. Six-DOF Input Devices for the Desktop 95
4.3. Tracking Devices 96
  4.3.1. Motion Tracking 96
  4.3.2. Eye Tracking 105
  4.3.3. Data Gloves 106
4.4. 3D Mice 110
  4.4.1. Handheld 3D Mice 111
  4.4.2. User-Worn 3D Mice 113
4.5. Special Purpose Input Devices 114
4.6. Direct Human Input 118
  4.6.1. Speech Input 119
  4.6.2. Bioelectric Input 120
  4.6.3. Brain Input 120
4.7. Home-Brewed Input Devices 122
  4.7.1. Strategies for Building Input Devices 122
  4.7.2. Connecting the Home-Brewed Input Device to the Computer 124

4.8. Choosing Input Devices for 3D Interfaces 126
  4.8.1. Important Considerations 126
  4.8.2. Input Device Taxonomies 128
  4.8.3. Empirical Evaluations 132

PART III 3D INTERACTION TECHNIQUES 135

Chapter 5 Selection and Manipulation 139

5.1. Introduction 139
  5.1.1. Chapter Roadmap 140
5.2. 3D Manipulation Tasks 140
  5.2.1. Canonical Manipulation Tasks 141
  5.2.2. Application-Specific Manipulation Tasks 143
5.3. Manipulation Techniques and Input Devices 143
  5.3.1. Control Dimensions and Integrated Control in 3D Manipulation 144
  5.3.2. Force versus Position Control 145
  5.3.3. Device Placement and Form-Factor in 3D Manipulation 145
5.4. Interaction Techniques for 3D Manipulation 147
  5.4.1. Classifications of Manipulation Techniques 147
  5.4.2. Interacting by Pointing 150
  5.4.3. Direct Manipulation: Virtual Hand Techniques 158
  5.4.4. World-in-Miniature 162
  5.4.5. Combining Techniques 163
  5.4.6. Nonisomorphic 3D Rotation 168
  5.4.7. Desktop 3D Manipulation 171
5.5. Design Guidelines 179

Chapter 6 Travel 183

6.1. Introduction 183
  6.1.1. Chapter Roadmap 184
6.2. 3D Travel Tasks 184
  6.2.1. Exploration 185
  6.2.2. Search 185
  6.2.3. Maneuvering 186
  6.2.4. Additional Travel Task Characteristics 187
6.3. Travel Techniques 188
  6.3.1. Technique Classifications 188
  6.3.2. Physical Locomotion Techniques 192
  6.3.3. Steering Techniques 199

6.3.4. Route-Planning Techniques 206
6.3.5. Target-Based Techniques 210
6.3.6. Manual Manipulation Techniques 214
6.3.7. Travel-by-Scaling Techniques 216
6.3.8. Viewpoint Orientation Techniques 217
6.3.9. Velocity Specification Techniques 219
6.3.10. Integrated Camera Controls for Desktop 3D Environments 220
6.4. Design Guidelines 222

Chapter 7 Wayfinding 227
7.1. Introduction 227
7.1.1. Chapter Roadmap 229
7.2. Theoretical Foundations 229
7.2.1. Wayfinding Tasks 231
7.2.2. Types of Spatial Knowledge 231
7.2.3. Egocentric and Exocentric Reference Frames 232
7.3. User-Centered Wayfinding Support 234
7.3.1. Field of View 235
7.3.2. Motion Cues 235
7.3.3. Multisensory Output 236
7.3.4. Presence 237
7.3.5. Search Strategies 237
7.4. Environment-Centered Wayfinding Support 238
7.4.1. Environment Design 239
7.4.2. Artificial Cues 242
7.5. Evaluating Wayfinding Aids 250
7.6. Design Guidelines 251
7.7. Conclusions 253

Chapter 8 System Control 255
8.1. Introduction 255
8.1.1. Human Factors of System Control 257
8.1.2. Input Devices 257
8.1.3. System- and Application-Level Factors 258
8.1.4. Chapter Roadmap 258
8.2. Classification 259
8.3. Graphical Menus 260
8.3.1. Techniques 260
8.3.2. Design and Implementation Issues 265
8.3.3. Practical Application 267

8.4. Voice Commands 268
- 8.4.1. Techniques 268
- 8.4.2. Design and Implementation Issues 268
- 8.4.3. Practical Application 269

8.5. Gestural Commands 270
- 8.5.1. Techniques 271
- 8.5.2. Design and Implementation Issues 272
- 8.5.3. Practical Application 273

8.6. Tools 273
- 8.6.1. Techniques 274
- 8.6.2. Design and Implementation Issues 276
- 8.6.3. Practical Application 277

8.7. Multimodal System Control Techniques 278

8.8. Design Guidelines 280

8.9. Case Study: Mixing System Control Methods 282
- 8.9.1. The ProViT Application 282
- 8.9.2. System Control Design Approach for ProViT 283
- 8.9.3. Mapping of Tasks to Devices 283
- 8.9.4. Placement of System Control 284
- 8.9.5. System Control Feedback 284

8.10. Conclusions 285

Chapter 9 Symbolic Input 287

9.1. Introduction 287
- 9.1.1. Why Is Symbolic Input Important? 288
- 9.1.2. Scenarios of Use 288
- 9.1.3. Brief History of Symbolic Input 290
- 9.1.4. Distinctive Features of Symbolic Input in 3D UIs 291
- 9.1.5. Chapter Roadmap 292

9.2. Symbolic Input Tasks 293
- 9.2.1. Alphanumeric Input 293
- 9.2.2. Editing Alphanumeric Symbols 293
- 9.2.3. Markup Input 294

9.3. Symbolic Input Techniques 294
- 9.3.1. Keyboard-Based Techniques 294
- 9.3.2. Pen-Based Techniques 300
- 9.3.3. Gesture-Based Techniques 303
- 9.3.4. Speech-Based Techniques 304

9.4. Design Guidelines 306

9.5. Beyond Number and Text Entry 310

PART IV **DESIGNING AND DEVELOPING 3D USER INTERFACES** **311**

Chapter 10 **Strategies for Designing and Developing 3D User Interfaces** **313**

10.1. Introduction 313
  10.1.1. Designing for Humans 314
  10.1.2. Inventing 3D User Interfaces 314
  10.1.3. Chapter Roadmap 315
10.2. Designing for Humans 315
  10.2.1. Feedback in 3D User Interfaces 315
  10.2.2. Constraints 322
  10.2.3. Two-Handed Control 323
  10.2.4. Designing for Different User Groups 327
  10.2.5. Designing for User Comfort 328
10.3. Inventing 3D User Interfaces 330
  10.3.1. Borrowing from the Real World 331
  10.3.2. Adapting from 2D User Interfaces 335
  10.3.3. Magic and Aesthetics 340
10.4. Design Guidelines 345

Chapter 11 **Evaluation of 3D User Interfaces** **349**

11.1. Introduction 349
  11.1.1. Purposes of Evaluation 350
  11.1.2. Terminology 351
  11.1.3. Chapter Roadmap 351
11.2. Background 351
  11.2.1. Tools for Evaluation Design and Implementation 352
  11.2.2. Evaluation Methods Used for 3D Interfaces 354
11.3. Evaluation Metrics for 3D Interfaces 357
  11.3.1. System Performance Metrics 357
  11.3.2. Task Performance Metrics 357
  11.3.3. User Preference Metrics 358
11.4. Distinctive Characteristics of 3D Interface Evaluation 360
  11.4.1. Physical Environment Issues 360
  11.4.2. Evaluator Issues 362
  11.4.3. User Issues 363
  11.4.4. Evaluation Type Issues 365
  11.4.5. Miscellaneous Issues 366
11.5. Classification of 3D Evaluation Methods 367
11.6. Two Multimethod Approaches 369
  11.6.1. Testbed Evaluation Approach 369

11.6.2. Sequential Evaluation Approach 374
11.6.3. Comparison of Approaches 378
11.7. Guidelines for 3D Interface Evaluation 382
11.7.1. General Guidelines 382
11.7.2. Guidelines for Formal Experimentation 383

PART V THE FUTURE OF 3D USER INTERFACES 385

Chapter 12 Beyond Virtual: 3D User Interfaces for the Real World 387
12.1. Introduction 387
12.1.1. What Is Augmented Reality? 389
12.1.2. Bringing Virtual Interfaces into the Real World 390
12.1.3. Chapter Roadmap 391
12.2. AR Interfaces as 3D Data Browsers 391
12.3. 3D Augmented Reality Interfaces 394
12.4. Augmented Surfaces and Tangible Interfaces 395
12.5. Tangible AR Interfaces 397
12.5.1. Design of Tangible AR 398
12.5.2. Time-Multiplexed Interaction in Tangible AR 400
12.5.3. Advantages and Disadvantages of Tangible AR 402
12.6. Agents in AR 403
12.7. Transitional AR: VR Interfaces 404
12.8. Conclusions 405

Chapter 13 The Future of 3D User Interfaces 407
13.1. Questions About 3D UI Technology 407
13.2. Questions About 3D Interaction Techniques 410
13.3. Questions About 3D UI Design and Development 412
13.4. Questions About 3D UI Evaluation 415
13.5. Million-Dollar Questions 416

Appendix A Quick Reference Guide to 3D User Interface Mathematics 419
A.1. Scalars 420
A.2. Vectors 420
A.3. Points 421
A.4. Matrices 422
A.5. Quaternions 424

Bibliography 429

Index 457

Supercell
Exhibit 1009
Page 13

Foreword

Three-dimensional user interfaces are finally receiving their due! Research in 3D interaction and 3D display began in the 1960s, pioneered by researchers like Ivan Sutherland, Bob Sproull, Fred Brooks, Andrew Ortony, and Richard Feldman. While many commercially successful 3D applications exist—computer-aided design and simulation, radiation therapy, drug discovery, surgical simulation, scientific and information visualization, entertainment—no author or group of authors has written a comprehensive and authoritative text on the subject, despite a continuing and rich set of research findings, prototype systems, and products.

Why is that? Why is it that this book by Doug Bowman, Ernst Kruijff, Joe LaViola, and Ivan Poupyrev is the first thorough treatment of 3D UIs?

Perhaps it was our digression during the last 20 years to the WIMP GUI. After all, the Windows, Icons, Menus, and Pointers GUI is used very widely by millions of users. Mac OS and Microsoft Windows users know it well, as do many UNIX users. Indeed, every user of the Web works with a GUI, and this year there are many hundreds of millions of them. Two-dimensional GUIs will be with us for a long time. After all, a lot of the workaday world with which we deal is flat—not just our Web pages but our documents, presentations, and spreadsheets too. Yes, some of these can be extended to 3D, but most of the time, 2D is just fine, thank you very much. Furthermore, pointing and selecting and typing *are* relatively fast and relatively error-free—they work, and they work well.

Perhaps it is that not as many people use 3D GUIs as use the 2D WIMP GUI, and so they are not thought to be as important. But the above list of 3D applications involves multibillion-dollar manufacturing industries, such as aerospace and automotive, and equally large and even more important activities in the life-saving and life-giving pharmaceutical and health care industries.

Perhaps it was that we needed the particular set of backgrounds that Doug, Joe, Ivan, and Ernst bring to the table. Doug comes out of the GVU Center at Georgia Tech, where he worked on 3D UIs with Larry Hodges and others and learned the value of careful user studies and experimentation, and he is now a member of an influential HCI group at Virginia Tech; Joe works at Brown with Andy van Dam, a long-time proponent of rich 3D interaction; Ivan comes from the HIT Lab at the University of Washington, where he worked with Tom Furness and Suzanne Weghorst, and now works with Jun Rekimoto at Sony CSL; and Ernst works with Martin Goebel in the VE Group at Fraunhofer IMK in Germany.

Whatever the case, I am excited and pleased that this team has given us the benefit of their research and experience. As I reviewed the draft manuscript for this book, I jotted down some of the thoughts that came to my mind: comprehensive, encyclopedic, authoritative, taxonomic; grounded in the psychological, HCI, human factors, and computer graphics literature; grounded in the personal research experiences of the authors, their teachers, and their students.

I myself have long preached the importance of integrating the study of the computer with the study of the human. Indeed, this is the key premise on which I built the GVU Center at Georgia Tech. This book certainly follows that admonition. There are numerous discussions of human issues as they relate to 3D navigation and interaction, drawing on references in psychology and human factors.

This is indeed a book for both practitioners and researchers. The extensive literature reviews, examples, and guidelines help us understand what to do now. Combined with the research agenda in Chapter 13, “The Future of 3D User Interfaces,” the material also helps us have a sense of what it is that we do not yet know.

I particularly commend to the readers the Chapter 11 discussion of evaluating 3D UIs. We in the computer graphics community have tended to design devices and techniques and then “throw them over the wall” to the user community. This is not the route to success. Careful study of user needs coupled with evaluation as part of the ongoing design cycle is much more likely to lead to effective techniques. The authors, all of whom have grappled with the difficult task of designing 3D interfaces, know from first-hand experience how crucial this is. Their section 11.4, on the distinctive characteristics of the 3D interface evaluation process, is a wonderful codification of that first-hand knowledge.

Thanks to Doug and Ernst and Joe and Ivan!

Jim Foley
GVU Center
College of Computing
Georgia Tech
March 2004

Preface

An architect sits in her home office, putting the final touches on the design of the new entrance to the city park. A three-dimensional virtual model of the park appears in front of her on the desk's surface. She nudges a pathway slightly to the right to avoid a low-lying area, and then makes the model life-size so she can walk along the path to view the effect. "Those dark colors on the sign at the entrance are too foreboding," she thinks, so she quickly changes the color palette to brighter primary colors. She looks up and notices that the clients are arriving for the final design review meeting. They are located in other offices around the city, but they can all view the 3D model and make suggested changes, as well as communicate with one another. "What's the construction plan?" asks one of the clients. The architect starts an animation showing the progress of the project from start to finish. "That first step may not work," says the client. "The excavation is much too close to the existing playground. Let me show you." He looks out his window, which has a view of the park, and overlays the virtual construction plan on it. "You're right," says the architect, "let's plan to move the playground slightly—that will be much cheaper than changing the construction site." After viewing the effects of the change, all agree that this plan will work, and the meeting adjourns.

This scenario and others like it illustrate the enormous potential of 3D environments and applications. The technology to realize such a vision is available now, although it will certainly be improved. But the scenario also leaves out a great deal of information—information that is crucial to making this dream a reality. How did the architect load the park model, and how does she manipulate her view of it? What technique is used to change the pathway? How can multiple clients all manipulate the model at the same time? How do the participants appear to each other in the virtual space? How is the speed and playback of the animation controlled? How did the client instruct the system to merge the real and virtual scenes?

These questions all relate to the design of the *user interface* (UI) and *interaction techniques* for this 3D application, an area that is usually given only a cursory treatment in futuristic films and books. The scenarios usually either assume that all interaction between the user and the system will be "natural"—based on techniques like intuitive gestures and speech—or "automatic"—the system will be so intelligent that it will deduce the user's intentions. But is this type of interaction realistic, or even desirable?

This book addresses the critical area of *3D UI design*—a field that seeks to answer detailed questions, like those above, that make the difference between a 3D system that is usable and efficient and one that causes user frustration, errors, and even physical discomfort. We present practical information for developers, the latest research results, easy-to-follow guidelines for the UI designer, and relevant application examples. While there are quite a few books devoted to UIs in general and to 2D UI design in particular, 3D UIs have received significantly less attention. The results of work in the field are scattered throughout numerous conference proceedings, journal articles, single book chapters, and Web sites. This field deserves a reference and educational text that integrates the best practices and state-of-the-art research, and that's why this book was created.

How This Book Came to Be

The story of this book begins in April 1998, when Ivan Poupyrev and Doug Bowman were doctoral students at Hiroshima University and Georgia Tech respectively, working on 3D interaction techniques for object manipulation in virtual environments (VEs). We started a lively email discussion about the design and usability of these techniques and about 3D UIs in general. Ivan, who was at the time a visiting research student at the University of Washington, suggested that the discussion would be even more profitable if other researchers in this new area could join in as well, and so the 3DUI mailing list was born. Since that time, over 100 researchers from around the globe have joined the list and participated in the discussion (to see an archive of all the list traffic or to join the list, check out *http://www.3dui.org*). Joe LaViola and Ernst Kruijff were two of the first people to join the list.

In August of that same year, Doug forwarded to the list a call for tutorials for the upcoming IEEE Virtual Reality Conference. After some discussion, Joe, Ivan, and Ernst agreed to join Doug to organize a tutorial on "The Art and Science of 3D Interaction." The tutorial was a big hit at the conference in Houston, and the four of us continued to present courses on the topic at ACM Virtual Reality Software and Technology 1999, IEEE VR 2000, and ACM SIGGRAPH 2000 and 2001.

After developing a huge amount of content for the notes supplements of these courses, we decided it would be silly not to compile and expand all of this information in book form. Furthermore, there was no way to include all the information available on 3D UIs in a one-day course. And that's why you're holding this book in your hands today—a book containing information on 3D UIs that can't be found in any other single source.

What's in the Book

The title of this book emphasizes that we have written it for both academics/researchers and practitioners/developers; both those interested in basic research and those interested in applications. Most chapters of the book integrate both theory and practical information. We intend the book to be used both as a textbook (see suggestions below) and as a reference work.

Theory-related content includes the following:

- sections on the psychology and human factors of various 3D interaction tasks
- information on different approaches for the evaluation of 3D UIs (Chapter 11)
- results from empirical studies of 3D interaction techniques
- a research agenda for 3D interaction (Chapter 13)
- lists of recommended further reading at the end of most chapters
- a comprehensive bibliography of important research articles

Practice-related content includes the following:

- principles for choosing appropriate input and output devices for 3D systems (Chapters 3 and 4)
- details and helpful tips for the implementation of common 3D interaction techniques
- guidelines for the selection of interaction techniques for common 3D tasks
- case studies of 3D UIs in real-world applications

The book is organized into five parts. Part I introduces the topic of 3D UIs. Part II discusses the input and output device technology used in the development of 3D UIs, with an emphasis on the impact of these devices on usability and performance. Part III presents a wide range of 3D interaction techniques for the common tasks of navigation, selection and manipulation, system control, and symbolic input. In Part IV, we discuss the design, development, and evaluation of complete 3D UI metaphors and applications. Finally, Part V considers the future, with chapters on 3D interaction in augmented reality applications and a research agenda for 3D UIs. The appendices include information on required mathematical background and a bibliography of 3D UI references.

Throughout the book, we offer several special features. First, most chapters contain numerous *guidelines*—practical and proven advice for the designer and developer. Guidelines are indicated in the text like this:

> Follow the guidelines in this book to help you design usable 3D UIs.

We also include implementation details for many of the most common and useful interaction techniques. We describe these algorithms using a combination of textual description and mathematical notation (to avoid a bias toward any particular development tool or programming style).

How to Use the Book and Related Material

If you are a 3D UI developer: Professional developers can use the book for inspiration and guidance in the design, implementation, and evaluation of applications with 3D UIs. In the design process, developers can consider overall UI metaphors from Part IV, choose specific interaction techniques from Part III, and match these with appropriate input and display devices from Part II. The design guidelines from all of these sections should help developers make rational, informed decisions. The implementation of the 3D UI can benefit from the textual and mathematical descriptions of interaction techniques we provide in Part III. Finally, developers can choose evaluation methods and assess the usability of their applications based on the information in Chapter 11.

If you are a teacher: The book can also be used as a textbook in several different types of university-level courses. A graduate course on 3D UI design could use it as a primary textbook. A more generic virtual environments course could use Parts I, II, and III of this book as an introduction to the basic technology and techniques used in VE interaction. An undergraduate HCI course could pull information from parts I and IV in a module on 3D interfaces and their differences from traditional UIs. Implementation of common techniques from Part III could enhance a course on interactive 3D graphics.

If you are a researcher: This book can serve as a comprehensive reference guide for researchers engaged in 3D UI design or evaluation, the investigation of 3D applications, or the use of VEs or augmented reality. The research agenda in Chapter 13 also provides researchers and research students with a list of important questions to be addressed in the field. It could even be used as the starting point for a Ph.D. student looking for a topic related to 3D UIs.

3D UI design is a fast-moving and evolving field. Therefore, we are committed to updating the material in this book. One way we will do this is through the book's official Web site at *http://www3dui.org*. This site will contain information and links related to the latest 3D UI research and applications, organized in the same manner as the book so you can easily find new information about the topics in a particular part or chapter. The site will also allow you to join the 3DUI mailing list. We also ask for your help in keeping the book up to date. Send us your comments, clarification questions, or links to additional information by visiting the web site above and using the online feedback form. Or email us directly at 3dui@3dui.org. Your comments will help us update the Web site as well as future editions of this book.

30706 00 ppi-xxvi r2jm.ps 5/13/04 5:50 PM Page xxiv

Acknowledgments

This book would not have been possible without the hard work, support, and intelligence of a large group of people.

First, we offer our gratitude to the reviewers who gave their time and energy in improving the quality of the book. Their comments and suggestions have made the book more complete, more readable, and more useful. Thanks to Ben Shneiderman, Harry Hersh, D. Jay Newman, Jeff Pierce, Dieter Schmalstieg, and Bob Zeleznik for providing this invaluable service.

Next, we would like to thank our editor at Addison-Wesley, Peter Gordon, for his invaluable advice and encouragement. The rest of the staff, including Bernie Gaffney, Amy Fleischer, Julie Nahil, Heather Mullane, and Curt Johnson has also been extremely helpful. Thanks also to Simone Payment and Carol Lallier for their competent and professional work during the production phase.

All of us would like to personally thank our colleagues in the 3D UI community for their fruitful discussions and collaborations. They include Mark Mine, Robert Lindeman, Matthew Conway, Ken Hinckley, Shumin Zhai, Kiyoshi Kiyokawa, Chris Shaw, Mark Billinghurst, Rudy Darken, Pablo Figueroa, and Bernd Fröhlich.

Portions of this material are based upon work supported by the National Science Foundation under Grants No. DUE-0127326 and IIS-0237412. Any opinions, findings and conclusions, or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation (NSF).

Doug Bowman: I would like to thank my wife, Dawn, for her unfailing love and support, as well as my extended family and friends, especially those at Grace Covenant Presbyterian Church. Much gratitude is due to Joe, Ivan, and Ernst for seeing this project through and for all their years of friendship and collaboration. Thanks also go to my colleagues and students at Virginia Tech, including Chris North, Ron Kriz, Mehdi Setareh, Walid Thabet, Thomas Ollendick, David Cox. Debby Hix, Joe Gabbard, Chad Wingrave, Jian Chen, Nicholas Polys, Wendy Schafer, and Marcio Pinho. Past colleagues at Georgia Tech also deserve thanks. They include Larry Hodges, John Kelso, Drew Kessler, David Koller, Donald Johnson, Donald Allison, Brian Wills, Jean Wineman, Jay Bolter, Elizabeth Davis, Albert Badre, and Ben Watson.

Ernst Kruijff: First of all, my thanks go to Doug, Ivan, and Joe for their great cooperation, help, and extensive discussions. Thanks also go to my parents, my brother, and my sister-in-law for their support, each in their own way. Furthermore, my thanks go to all of my colleagues at the VE group of IMK, especially Martin Goebel for being a great chief, Gerold Wesche, Andreas Simon, Gernot Goebbels, Stefan Conrad, Aeldrik Pander, and Steffi Beckhaus for their past and current cooperation and help during the making of this book and its instigators, the courses we (the authors) gave together. My thanks also go to my past colleagues at Bauhaus University, especially the members of igroup, in particular Holger Regenbrecht. Furthermore, thanks to all the students who helped in many projects, especially Stefan Hansen, Arnold Mueller, Jakob Beetz and Hartmut Seichter. Finally, my thanks go to Dieter Schmalstieg for being patient with my Ph.D. efforts.

Joe LaViola: I would like to thank my Ph.D. thesis advisor, Andries van Dam, for his forgiveness and patience and for putting up with someone as hardheaded as me. Thanks also to my mom and dad, Jamie, Nick, and Heidi for their love and support throughout this writing of this book. My colleagues Robert Zeleznik, Daniel Keefe, Daniel Acevedo Feliz, Andrew Forsberg, David Karelitz, Tim Rowley, Christine Waggoner, and all the members of the Brown University Computer Graphics Group past and present also deserve thanks. Finally, thanks to my coauthors for their hard work and dedication.

Ivan Poupyrev: I would never have been able to work on this book without the help and support I received from many people I was fortunate to meet. I am deeply indebted to Professor Tadao Ichikawa, who supervised my Ph.D. thesis at Hiroshima University. I am also thankful to Professor Masahito Hirakawa for his help and support, as well to my fellow students and researchers Bryn Holmes, Olivier Liechti, and Numada Tomokazu. I am grateful to the Japanese Government for providing me with the Monbusho Scholarship for conducting graduate studies in Japan.

While working on my Ph.D. thesis, I had the exceptional opportunity to spend almost three years at the Human Interface Technology Laboratory (HITL) at the University of Washington. The HIT Lab experience made an enormous impact on me and formed me as a researcher. For this, I will always be grateful to Suzanne Weghorst, the Director of Research at the HIT Lab, who invited me in Summer 1995 to join HITL first as a summer intern and then as a Visiting Researcher; and Professor Tom Furness III, Director of HITL, who believed in me and provided invaluable moral and financial support. I was very fortunate to meet and work with Mark Billinghurst, our collaboration and friendship extended well beyond my stay at the HITL. I am also deeply thankful to Edward Miller for his help in developing a number of HITL 3D UI projects, in particular the V3D interface toolkit, as well as to Jerry Prothero and Hunter Hoffman for late-night discussions of chess, psychology, and the meaning of life; Tony Emerson for finding everything that I ever needed; Ann Elias for all her generous help while I at was the HITL; Mark Phillips, Jennifer Feyma, and Chris Airola for being friends and giving me life outside the lab.

The Augmented Reality chapter of this book includes a significant amount of work that I did while at the ATR Media Integration and Communication Research Labs in Kyoto. I am very thankful to all the amazing people I met and worked with there: Ryohei Nakatsu, Jun Ohya, Nobuji Tetsutani, Jun Kurumisawa, Tatsumi Sakaguchi, Keiko Nakao, Lew Baldwin, Desney Tan, Sidney Fels, Michael Lyons, Tal Shalif, Parham Zolfaghari, Christa Sommerer, and many others.

Finally, I am thankful to my family and friends in Russia, particularly my parents and brother Pavel Poupyrev; my first research advisers Vladimir Lischouk, Dinara Gazizova, and Vladimir Lukin; as well as to Sergei Tsimbalist, Vadim Malishev, Vitaly Lazorin, Constantin Guzovski, and Elena Krupskaya.

It is impossible to mention everyone in Japan, the United States, and Russia who helped, supported, and inspired me. Without this help, support, and inspiration I would not be able to take even my current modest steps toward designing and exploring future interaction technologies. My deepest gratitude and appreciation goes to all of you.

CHAPTER 7
Wayfinding

In this chapter, we discuss the 3D interaction task called wayfinding. As we saw in the introduction to Part III, wayfinding is the cognitive component of navigation. Here, we look at the psychological foundations of wayfinding, techniques and principles for supporting users' wayfinding in 3D environments, and the connection between wayfinding and travel techniques.

7.1. Introduction

We can define wayfinding as follows:

> Wayfinding is the cognitive process of defining a path through an environment, using and acquiring spatial knowledge, aided by both natural and artificial cues.

Wayfinding is a common activity in our daily lives. We move through real-world environments, such as cities, buildings, and roadways, for the purpose of reaching a destination or perhaps simply to explore. Wayfinding tasks can be as easy as visiting the bakery around the corner or as difficult as finding a specific address on a back road in an unknown neighborhood.

Wayfinding is often an unconscious activity—we move from one point to another without actively considering that we are finding our way through an environment. When we get lost, however, wayfinding may come to the forefront of our attention.

Many different types of information help us to perform wayfinding tasks. Landmarks such as a church, or specific items in a building such as the soda machine, may help us decide which way to travel. Routes that we have traveled before provide familiar surroundings. Signs, maps, and other directional information can help in unfamiliar situations.

It is important to provide as many different types of spatial information as possible when 3D virtual worlds are used for wayfinding-related purposes. These purposes can roughly be subdivided into two categories:

1. *Transferring spatial knowledge to the real world*: We can use a VE to obtain knowledge of the layout of an environment so we can use this knowledge in the real world. For example, firefighters have used VEs to quickly get an impression of the layout of a burning building to reduce the potentially hazardous effects of getting lost (Bliss et al. 1997; Waller et al. 1998).
2. *Navigation through complex environments in support of other tasks:* Large-scale, complex virtual worlds used for real-world work may require wayfinding support. For example, this may happen when someone is evaluating a large building and needs to relate structural information from different building locations to a colleague. With regards to complexity, an environment becomes increasingly difficult for us to comprehend when we cannot overview the complete environment at once, from one location.

Wayfinding in 3D UIs is difficult to support because of the differences between wayfinding in a real and in a VE. We are used to having solid ground under our feet, but in a VE, we may have to work without these natural constraints. Unconstrained movement can disorient people easily, and the absence of physical constraints may increase this feeling of disorientation. The lack of realistic motion cues due to virtual movement instead of physical walking makes wayfinding more difficult and may even lead to cybersickness (LaViola 2000a).

On the other hand, 3D UIs provide a wealth of opportunities for both natural and artificial wayfinding aids. We can classify these wayfinding aids in two groups: user-centered and environment-centered aids. *User-*

centered aids make use of the characteristics of human perception and can draw upon multiple human senses. Many of these aids relate to the topic of *presence,* the feeling of being in an environment. A detailed discussion of presence is beyond the scope of this text, but we provide some useful references in the recommended reading list at the end of the chapter. *Environment-centered* wayfinding aids refer to the conscious design of the virtual world to support wayfinding. The visual communication sciences and urban planning are an excellent source of information on real-world environment-centered aids.

7.1.1. Chapter Roadmap

The aim of this chapter is to describe the activity of wayfinding as a decision-making process and to identify techniques to support this process. This requires some background in psychology so the first part of this chapter introduces some basic cognitive principles on which wayfinding is built (section 7.2).

In the second part of the chapter, we explain how to apply the cognitive foundations to support wayfinding in VEs. Wayfinding aids are described in sections on user-centered support (section 7.3) and environment-centered support (section 7.4). Finally, we describe how aids can be evaluated (section 7.5) and provide some general guidelines for their use in 3D UIs (section 7.6).

7.2. Theoretical Foundations

Wayfinding is a decision-making process (Figure 7.1). This means that a user makes decisions (where am I? which direction should I go?) by mentally processing “input” (information obtained from the environment) and producing “output” (movement along a trajectory). Decision making is a very general cognitive process, and it can be represented in many ways. Golledge (1999) provides a good overview of cognitive processes and specific cognitive factors that affect wayfinding. Readers unfamiliar with cognitive psychology will find this and the other recommended readings (found at the end of the chapter) to be helpful.

Navigation in a 3D environment involves the processing of multiple sources of sensory information that we receive from the environment and the use of this information to execute a suitable travel trajectory. The en-

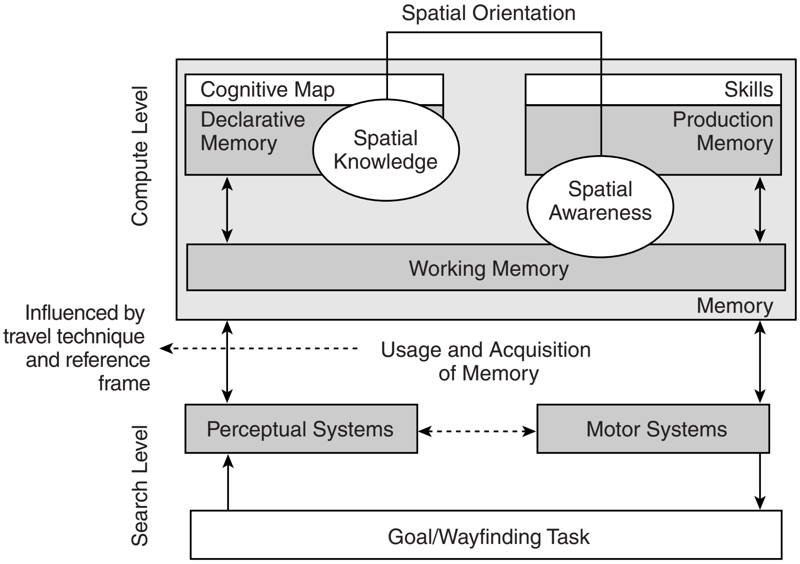
Spatial Orientation
Compute Level
Cognitive Map
Declarative Memory
Spatial Knowledge
Skills
Production Memory
Spatial Awareness
Working Memory
Memory
Influenced by travel technique and reference frame
Usage and Acquisition of Memory
Search Level
Perceptual Systems
Motor Systems
Goal/Wayfinding Task

Figure 7.1 *A representation of wayfinding as a decision-making process.*

vironmental information is stored in our long-term memory and is generally referred to as the *cognitive map,* the corpus of spatial knowledge we obtain from our environment. To be more precise, the cognitive map is a mental, hierarchical structure of information that represents spatial knowledge (Stevens and Coupe 1978; Downs and Stea 1977).

When we perform a wayfinding task, we make use of existing spatial knowledge, acquire new spatial knowledge, or use a combination of the two. The process of accessing, using, and building the treelike structures in the cognitive map is also called *cognitive mapping.*

Navigation is based on a tight feedback loop that continuously defines the relationship between the information we receive and our cognitive map of the environment, which enables us to understand our position and orientation. The knowledge of our location and viewing direction is called *spatial orientation,* while the combination of spatial orientation and spatial knowledge (cognitive map) is called *situation awareness* (a term generally used in aviation).

7.2.1. Wayfinding Tasks

In Chapter 6, we introduced three types of travel tasks. Wayfinding tasks are similar, but we present them here in the wayfinding context. We also consider a fourth wayfinding task with no analogous travel task.

Exploration involves browsing the environment. The user has no particular goal in mind, and perhaps a less structured movement pattern. However, exploration may be very effective in helping to build the cognitive map.

The second type of wayfinding task is *search.* Naturally, during search tasks, spatial knowledge is not only acquired, but also used. Naïve search is a task that is target-based, but in which the user does not know the exact location of the target. This means that the cognitive map does not contain enough information to allow direct movement toward the target. A primed search is also a target-based search task, but with a known target location. Therefore, the user's ability to use the cognitive map to understand the relationship between his present location and the target's location will define the success of finding the target.

In *maneuvering* tasks, a user performs many small-scale movements to reach a very specific position. During wayfinding, maneuvering may happen occasionally as a subtask of the search tasks mentioned above. It may, for example, be needed to identify a landmark from a specific point of view or to find a very small target. Maneuvering may also occur when the user is lost and needs to obtain more information from a specific location in order to decide which way to go.

The final wayfinding task is *specified trajectory movement.* In this task, the user is guided automatically through an environment along a predefined path in order to obtain a broad overview of the environment. This type of movement allows her to build a basic cognitive map in a short time, as long as the trajectory is defined effectively (with regards to movement pattern and viewpoints). Allowing the user to control viewpoint orientation during the movement provides for more effective gathering of spatial knowledge. This task was not identified as a travel task because it does not involve free motion through an environment.

7.2.2. Types of Spatial Knowledge

Search strategies and movement parameters influence the effectiveness of spatial knowledge acquisition. These factors affect not only the efficiency of building a cognitive map but also which of the qualitatively different kinds of spatial knowledge are acquired.

During wayfinding, people obtain at least three different kinds of spatial knowledge in the cognitive map (Thorndyke and Hayes-Roth 1982):

- *Landmark knowledge* consists of the visual characteristics of the environment. Visually prominent objects ("landmarks") form part of this information, but other visual features such as shape, size, and texture also play a role. In London, for example, Big Ben, the river Thames, and Heathrow airport are locations that many visitors immediately add to their landmark knowledge.
- *Procedural knowledge* (or *route knowledge*) describes the sequence of actions required to follow a certain path or traverse paths between different locations. Only sparse visual information is needed for procedural knowledge to be used properly (Gale et al. 1990). For example, a visitor to London will quickly memorize the route between her hotel and the nearest underground station.
- *Survey knowledge* can be described as the configurational or topological knowledge of an environment, consisting of object locations, interobject distances, and object orientations. This kind of knowledge is maplike and can therefore also be obtained from a map, even though the acquired knowledge from the map tends to be orientation-specific (Darken and Cevik 1999). Of the three kinds of spatial knowledge, survey knowledge represents the (qualitatively) highest level of knowledge and normally also takes the longest to mentally construct. Our fictitious visitor to London may attempt to obtain survey knowledge by studying the map of the underground (although she may not be successful!).

Building spatial knowledge requires visual, vestibular, and other motion information. For more information, refer to Henry and Furness (1993) and Steck and Mallot (2000).

7.2.3. Egocentric and Exocentric Reference Frames

During real-life motion, we feel as if we are in the center of space, a phenomenon that is called *egomotion.* During such motion, we need to match *egocentric* (first-person) information to the cognitive map, which commonly stores *exocentric* (third-person) information (Thorndyke and Hayes-Roth 1982). The differences between the egocentric and exocentric reference frames play a crucial role in wayfinding.

But what are these differences? Basically, an egocentric reference frame is defined relative to a certain part of the human body, whereas an exocentric reference frame is object- or world-relative.

During egocentric tasks, judgments are made according to the egocentric reference frame, which consists of the stationpoint (nodal point of the eye), retinocentric (the retina), headcentric (focused solely on the head), bodycentric (the torso), and proprioceptive subsystems (visual and nonvisual cues from our body parts, such as hands and legs), as shown in Figure 7.2. Details on these reference frames can be found in Howard (1991).

The egocentric reference frame provides us with important information such as distance (obtained from physical feedback like a number of strides or an arm's length) and orientation (obtained from the direction of the eyes, head, and torso). An object's position, orientation, and movement are related to the position and orientation of the eyes, head, and body.

During exocentric tasks, the position, orientation, and movement of objects are defined in coordinates external to the body. Namely, they are

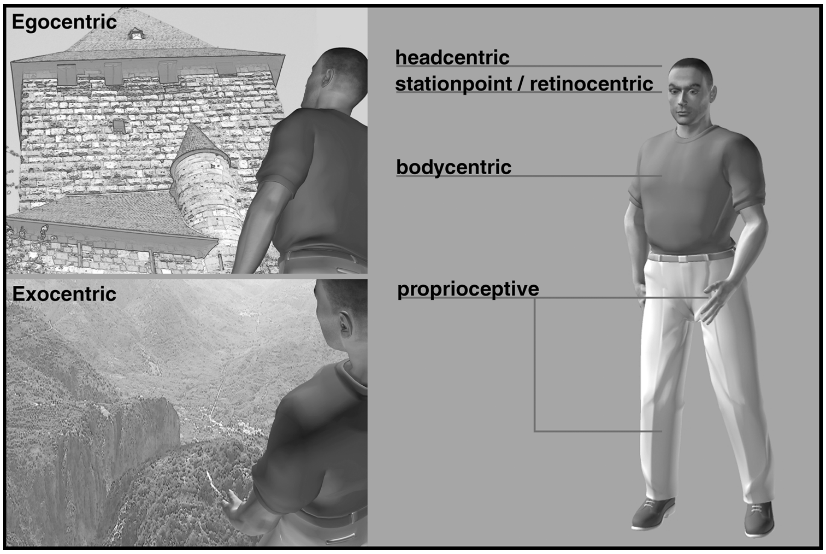

Figure 7.2 *Human reference frames (right) and associated views (left). In an egocentric view (top left), the user is inside the environment, while in an exocentric view (bottom left), the user is outside the environment, looking in. (Image courtesy of Ernst Kruijff)*

defined by an object's shape, orientation, and motion. Exocentric attributes are not affected by our orientation or position.

The reference frame is directly dependent on our viewpoint. The egocentric reference frame corresponds to first-person viewpoints, while exocentric reference frames are related to third-person (bird's-eye or outside-in) viewpoints. For example, in many video games, the user typically sees a first-person (egocentric) view of the environment as he navigates through it, but can also access an overview map of the environment showing his current location (exocentric).

When we find our way through an environment, we build up an exocentric representation (survey knowledge). However, when we enter an environment for the first time, we basically depend on egocentric information (landmark and procedural knowledge). Therefore, we often depend on landmarks at first, then develop routes between them, and eventually we generalize that egocentric spatial information into exocentric survey knowledge. It remains unclear, however, how the human brain determines the relationship between egocentric and exocentric spatial knowledge.

7.3. User-Centered Wayfinding Support

With the psychological foundations of wayfinding in mind, we now illustrate how to support wayfinding in a 3D environment. In general, the effectiveness of wayfinding depends on the number and quality of wayfinding *cues* or *aids* provided to users. This section (on user-centered cues) and the next (on environment-centered cues) present a number of different wayfinding aids and address questions such as, When and how should I include cues? and How does the design of my environment affect wayfinding?

Recall that user-centered wayfinding aids are targeted to human sensory systems. Thus, most user-centered support is technology-oriented. Since output devices still cannot deliver information that fully matches the capabilities of the human perceptual system (see Chapter 3), they can have a negative impact on wayfinding. There are certain strategies that developers can use, however, to lessen these negative effects. In this section we discuss

- field of view
- motion cues

- multisensory output
- presence
- search strategies

All of these wayfinding cues can be employed to some degree in desktop 3D UIs, but wide fields of view, physical motion cueing, and presence are typically associated with more immersive systems.

7.3.1. Field of View

A small field of view (FOV) may inhibit wayfinding. Since a smaller portion of the environment is visible at any given time, the user requires repetitive head movements to comprehend the spatial information obtained from the viewpoint. Using a larger FOV reduces the amount of head movement and allows the user to make spatial relationships easier. Some studies, such as Péruch, May, and Wartenburg (1997) and Ruddle, Payne, and Jones (1998), do not fully support these claims, showing little difference in the orientation capabilities of a user between several small FOVs (40, 60, and 80 degrees, or in desktop environments). However, they have demonstrated the usefulness of larger FOVs when environments become more detailed and complex. Furthermore, wide FOVs closer to the FOV of the human visual system (like those in some surround-screen displays) were not considered in these studies.

Another negative side effect of a small FOV is the lack of optical-flow fields in users' peripheral vision. Peripheral vision provides strong motion cues, delivering information about the user's direction, velocity, and orientation during movement. Finally, it has been shown that small FOVs may lead to cybersickness (Stanney et al. 1998).

7.3.2. Motion Cues

Supplying motion cues enables the user to judge both the depth and direction of movement and provides the information necessary for dead reckoning (backtracking of the user's own movement). Motion cues can be obtained from peripheral vision, as discussed above, but motion cues are not purely visual—it is important to supply the user with additional vestibular (real motion) cues if possible. A lack of vestibular cues causes an intersensory conflict between visual and physical information. This may cause cybersickness and can affect judgments of egomotion, thus negatively impacting the formation of the cognitive map.

The effect of real motion cues on the orientation abilities of users in VEs has been the subject of a range of studies. Slater, Usoh, and their colleagues (1995; 1999) compared a virtual travel technique based on pointing (Chapter 6, section 6.3.3) against walking in place and natural walking (using wide area tracking; Chapter 6, section 6.3.2). Walking techniques (physical motion) performed better than pointing techniques (purely virtual motion), even though some users preferred pointing due to ease of use. Although there were no large differences between walking in place and natural walking for the spatial orientation of the user, natural walking increased the sense of presence considerably (see section 7.3.4). Other studies of virtual and real travel have shown positive effects of real motion on spatial orientation (Klatzky et al. 1998; Chance et al. 1998).

Our understanding of the proper balance between visual and vestibular input is still being formed. Harris (Harris et al. 1999) performed tests matching visual and passive vestibular input, and concluded that developers should add vestibular information corresponding to at least one-quarter of the amount of visual motion.

Motion cues are difficult to implement for desktop 3D UIs. Peripheral vision cues cannot be provided due to the size of the display screen, and physical motion cues are limited for a seated user. The use of vestibular cue devices (see Chapter 3) can provide some motion cues directly to the brain, but their effects are not well understood.

7.3.3. Multisensory Output

In addition to the visual and vestibular systems, developers might want to experiment with other sensory systems to deliver wayfinding cues. Audio (see Chapter 3) can provide the user with useful directional and distance information (Davis et al. 1999). For example, the sound of trains can indicate the direction to the station, whereas the volume allows the user to estimate the distance to the train station. Audio for wayfinding support is still a largely open question.

Another form of multisensory support is the tactile map—a map whose contours are raised so they can be sensed by touch as well as sight. Initial experiments used tactile maps to fill in gaps in the spatial knowledge of visually impaired people. The tactile map was used as an additional cue, not as a substitute for another cue type (Jacobson 1996). Tan and colleagues (2002) showed that tactile cues can aid in the formation and usage of spatial memory, so tactile wayfinding aids are another area of great potential.

7.3.4. Presence

The sense of presence (the feeling of "being there") is a much explored but still not well-understood phenomenon that is assumed to have an impact on spatial knowledge. Briefly, the idea is that if the user feels more present in a virtual world, then real-world wayfinding cues will be more effective. Many factors influence the sense of presence, including sensory immersion, proprioception, and the immersive tendency of the user. The inclusion of a *virtual body*—that is, the user's own virtual representation—may enhance the sense of presence, which in turn has a positive effect on spatial knowledge acquisition and usage (Draper 1995; Usoh et al. 1999). A discussion of presence falls largely outside the scope of this book, but we do list some recommended reading on this topic at the end of the chapter.

7.3.5. Search Strategies

A final user-centered wayfinding technique is to teach the user to employ an effective search strategy. Using a search strategy often depends on user skill. More skilled users, like professional aviators, use different strategies than users with limited navigation experience. Not only do skilled users depend on other kinds of spatial knowledge, and therefore on other cues in the environment, but they often use different search patterns as well. Whereas novice users depend largely on landmarks, skilled users make use of cues like paths (for example a coastline).

Using a search strategy inspired by navigation experts can increase its effectiveness. For example, search patterns used during aviation search-and-rescue missions may aid a user during wayfinding (Wiseman 1995).

Figure 7.3 shows several possible effective search patterns. The basic line search follows a pattern of parallel lines along a specific line. The pattern search starts at a specific central point and moves further away from it, using quadratic or radial patterns. The contour search is designed to follow contours in a landscape, like a river or a mountain. Finally, the fan search starts from a center point and fans out in all directions until the target is found. Of course, the use of these search strategies is dependent

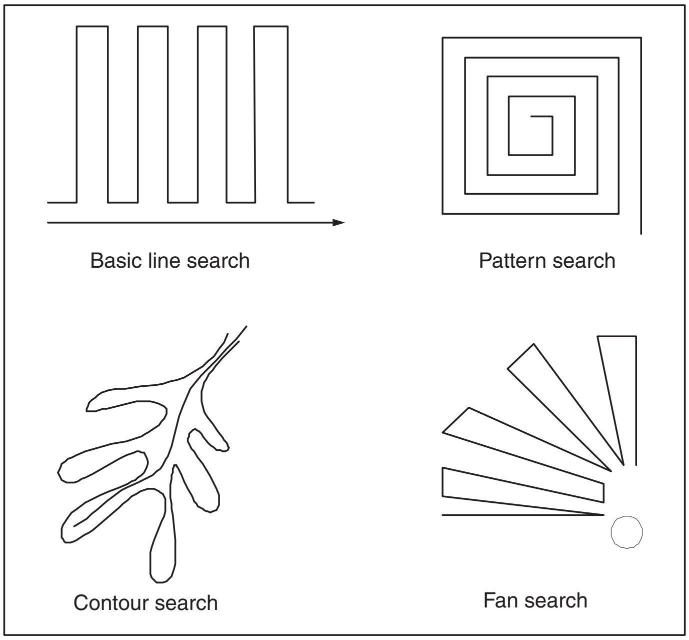

Figure 7.3 *Search patterns. (Figure adapted from Wiseman 1995)*

on the content of the environment—they might work well in a large outdoor environment, but would not make sense in a virtual building.

Another important search strategy is to obtain a bird's-eye view of the environment rather than performing all navigation on the ground. Users can be trained to employ this strategy quite easily, and it results in significantly better spatial orientation (Bowman, Davis et al. 1999). This can even be automated for the user. In the "pop-up" technique (Darken and Goerger 1999), users can press a button to temporarily move to a significant height above the ground, then press the button again to go back to their original location on the ground.

We assume that novice users can learn search techniques, even if the described pattern search strategies are seen primarily in expert navigators. Placing a rectangular or radial grid directly in the environment provides a visual path along which users can search. Although these grids may supply directional and depth cues, they do not necessarily force the user to maximize search effectiveness.

7.4. Environment-Centered Wayfinding Support

Beyond the technology and training support described above, most wayfinding aids for virtual worlds can be directly related to aids from the real world. These range from natural environmental cues like a high mountain to artificial cues such as a map. We discuss both *environment design* (the structure and visual aspects of an environment) and *artificial aids* (cues that can be added to an environment either as an environmental element or as a tool) as effective approaches to environment-centered wayfinding support.

7.4.1. Environment Design

The first type of environment-centered support relates to the construction of the environment itself. The world's structure and form can provide strong wayfinding cues without the need for cluttering the environment with added wayfinding aids. Of course, when the virtual world must accurately reflect a real-world location, such explicit environment design may not be possible. But in cases where the designer has some freedom, these techniques can be invaluable. Our discussion focuses on two types of environment design principles:

- Legibility techniques
- real-world wayfinding principles drawn from the natural environment and from architectural design

Legibility Techniques

Many of the structural rules applied to VEs are obtained from urban design principles. A book that has been the basis for many of those rules is *The Image of the City* (Lynch 1960). In this book, Lynch describes so-called *legibility techniques.* These techniques allow the user to quickly obtain an understanding of an environment by understanding its basic structural elements. For 3D environment design, we can summarize Lynch's theory as follows (Darken and Sibert 1996; Ingram et al. 1996):

- Divide large-scale environments into parts with a distinct character.
- Create a simple spatial organization in which the relationships between the parts are clear.
- Support the matching process between egocentric and exocentric reference frames by including directional cues.

Lynch identifies several basic building blocks that can be applied to achieve a legible environment: paths, edges, districts, nodes, and landmarks. *Paths* are elements or channels for linear movement, like streets or railways. People often view a city from the perspective of such paths. *Edges* are related to paths, but are focused on bordering spaces rather than on movement. These edges can be natural, like a river, or artificial, like a walled structure. *Districts* are areas that are uniquely identifiable because of their style (e.g., building style), color, or lighting. *Nodes* are gathering points, such as a major intersection of streets, or the "entrance" to a certain district. Finally, *landmarks* are objects that are easily distinguished and are often placed near a node (Darken and Sibert 1996).

A legible environment often has a repetitive structure; for example, think about how cities like New York are structured—they have a strong, regular pattern (see Figure 7.4). Most such structures deliberately make use of right angles. Research has shown that structures containing irregular corners (smaller or larger than 90 degrees) can lead to severe disorientation, even to the degree that survey knowledge can hardly be obtained. Users may then only be able to build the cognitive map to the level of route knowledge (Moeser 1988). Continuous change of the angles used in

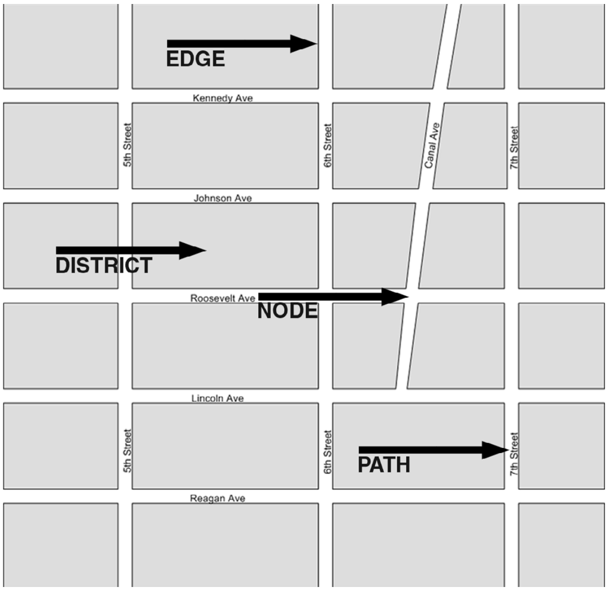

Figure 7.4 *A typical American city structure.*

corners (as in many European medieval towns) may also lead to disorientation (Ruddle et al. 1998).

These principles can be applied to virtual cities, buildings, and landscapes as easily as they can be applied to the real-world counterparts, and the virtual world provides even more flexibility in this regard. For example, Darken and Sibert (1996) placed large artificial landmarks in an environment that could be seen from any location in the environment, and he also found that the lines of a grid overlaid on the environment functioned effectively as edges or paths during search tasks. Even abstract information visualizations can be organized around these rules to improve users' spatial orientation. For example, data points can be clustered to create "districts" or visually linked to create "paths" (Ingram et al. 1996).

Real-World Design Principles

Many of the real-world design principles that can be applied to VE design are directly related to perceptual issues, like depth or distance estimation or visual communication. These techniques are often used to help us distinguish between structural elements.

Natural environment principles: We may draw upon the natural environment to supply the user with some basic wayfinding aids. A *horizon* gives us basic directional orientation information; *atmospheric* color methods or *fog* may provide better depth cues and help us to estimate distances more exactly.

Architectural design principles: Related to city planning techniques, architectural design principles can help in the acquisition and usage of spatial knowledge or in leading the user deliberately to specific parts of the environment. For example, correct *lighting* not only provides *shadows* as depth cues, but can also be used as a directional cue. By illuminating certain objects in an environment (like a doorway), these objects may be better recognized as landmarks. We often tend to move toward a light source, so lighting a target may also help users to find it more easily. In addition, the careful design of *closed and open spaces* can be used to direct a user to certain locations, since we tend to move toward openings (e.g., a door in the wall).

Color and texture: The useful application of *colors* is probably one of the most powerful real-world design techniques. Color provides us with landmark knowledge, but can also be used to identify certain types of objects. Structure can be communicated by making color groups. Specific objects (like landmarks) can be made more identifiable by giving them a contrasting color. Well-chosen *textures* not only provide us with depth cues, but can also provide landmark knowledge. In some cases, textures can also be used to visually lay out a path through an environment in order to support the formation of procedural knowledge. For example, a ribbon of colored carpet in a building can lead users along a path to important locations.

7.4.2. Artificial Cues

A second major way to support wayfinding in the environment is the addition of artificial wayfinding aids. With this approach, we enhance an existing environment rather than design the environment itself to support wayfinding. Artificial cues can be either placed directly in the environment or provided to the user as tools. Some of the artificial cues that have been used in 3D environments include:

- maps
- compasses
- signs
- reference objects
- artificial landmarks
- trails
- audio and olfactory cues

Maps

A map is a powerful tool for the acquisition of spatial knowledge. Because a map normally provides an exocentric representation of an environment, it can aid the formation of survey knowledge by a user. The knowledge obtained from traditional maps, however, tends to be orientation-specific (Tlauka and Wilson 1996; Darken and Cevik 1999). Looking south in the environment while trying to use a map with north at the top can lessen our wayfinding abilities. We need to be able to match the exocentric information from the map with the egocentric information from a first-person perspective.

Directional problems are just one example of difficulties with implementing a map in a 3D UI. Here are some other design guidelines for including maps in virtual worlds.

Use you-are-here maps.

You-are-here (YAH) maps combine a map with a YAH-marker. Such a marker helps the user to gain spatial awareness by providing her viewpoint position and/or orientation dynamically on the map. This means that the marker needs to be continuously updated to help the user match her egocentric viewpoint with the exocentric one of the map.

Consider multiple maps at different scales.

In some large-scale environments, a single map might not be sufficient. Ruddle, Payne, and Jones (1999) experimented with a combination of local and global maps. Global maps provide the world-reference positions of objects for easy location, whereas local maps communicate the direct surroundings of the user, allowing him to easily detect important objects in a scene. This is an application of the well-known "focus plus context" principle from information visualization (Ware 2000). Figure 7.5 shows the global and local map combination, with a YAH marker in the local map.

Carefully choose the orientation of the map.

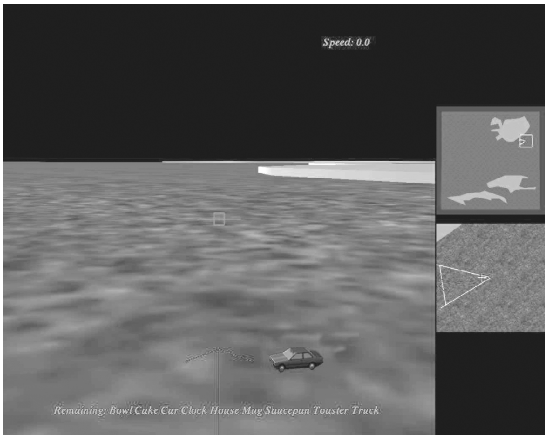
Speed: 0.0
Remaining: Bowl Cake Car Clock House Mug Saucepan Toaster Truck

Figure 7.5 *Local (bottom right) and global (upper right) maps.(Image courtesy of Roy Ruddle)*

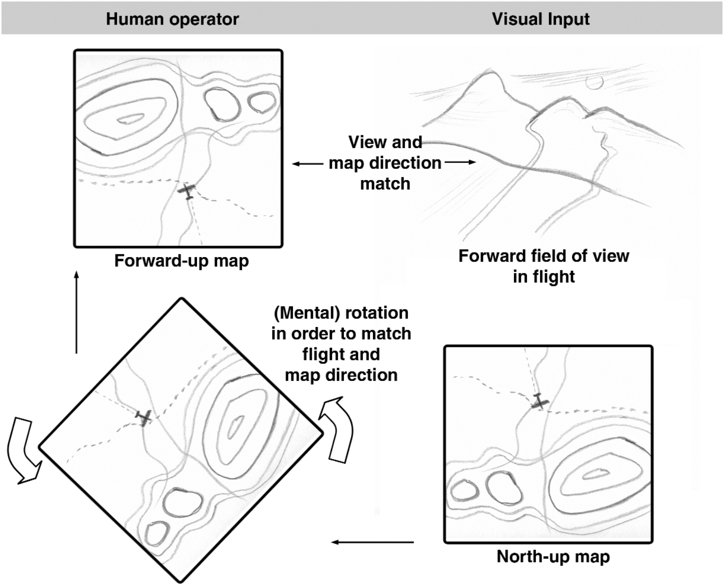
Human operator
Visual Input
View and map direction match
Forward-up map
Forward field of view in flight
(Mental) rotation in order to match flight and map direction
North-up map

Figure 7.6 *Rotation of a map to align viewpoint direction and map direction. (Figure adapted from Wickens and Carswell 1997)*

When a map is not aligned with the environment, users must mentally rotate the map information (Figure 7.6). Mental rotations can cause high cognitive load for a user.

One of the most famous studies on mental rotation (Shepard and Metzler 1971) shows that every 60 degrees of mental rotation will take a person approximately one second. Therefore, carefully choosing the orientation of a map may save the user time. Darken and Cevik (1999) compared a standard "north-up" map to a "forward-up" map (which rotates so that the map is always aligned with the environment; see Figure 7.7). He found that a forward-up map is preferable in egocentric search tasks (e.g., searching for an intersection in a city), whereas in exocentric search tasks (e.g., searching for an airport from the air) a north-up map seems to perform better.

Make the map legible.

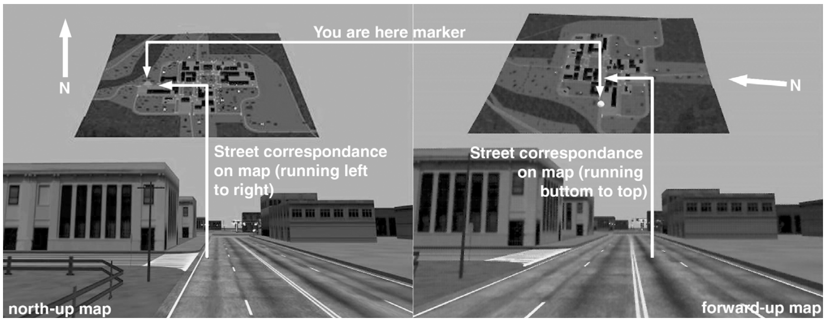
You are here marker
N
N
Street correspondance on map (running left to right)
Street correspondance on map (running buttom to top)
north-up map
forward-up map

Figure 7.7 *Forward-up versus north-up map. (Image courtesy of the MOVES Institute, Naval Postgraduate School)*

It is important for a map to communicate its contents clearly. The details of choosing a graphical representation for the environment is beyond the scope of this text, but other books on the design of graphical information (Tufte 1990) provide an excellent reference on the subject. A map should clearly show the organizational structure of the environment, for example, by combining a map with a grid. The grid may also support search strategies such as those described in section 7.3.5.

Use appropriate map size and placement to reduce occlusion of the environment.

Maps can fill a large portion of the display and thus occlude (block) the environment. A map must be large enough to communicate its details to the user. The size is dependent on the size of the environment, the resolution, and even the contrast of the display.

In an HMD-based system, for example, the resolution is typically quite low, meaning that a legible map might take up a quarter to a half of the display area! One solution is to allow the user to hide or show the map when needed, although this may lead to orientation difficulties. Another way to quickly access a map and reduce occlusion is to place the map on a tracked physical surface, such as a tablet (Figure 7.8), so that users can look at the map when needed and drop it out of their field of view at other times (Bowman, Wineman et al. 1998). In a surround-screen display, the map could be placed on the floor, as in Figure 7.9.

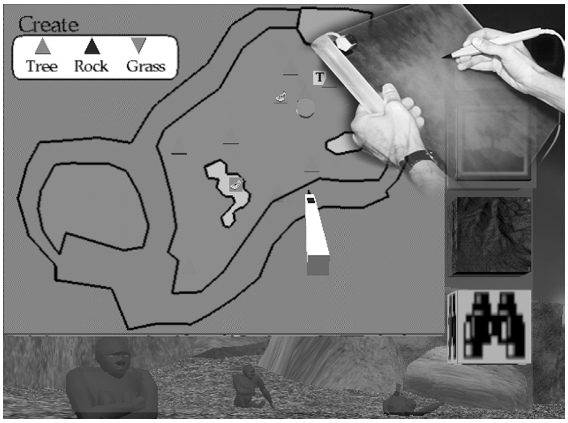

Figure 7.8 *Tablet-based map from the Virtual Habitat application. (Bowman et al. 1999)*

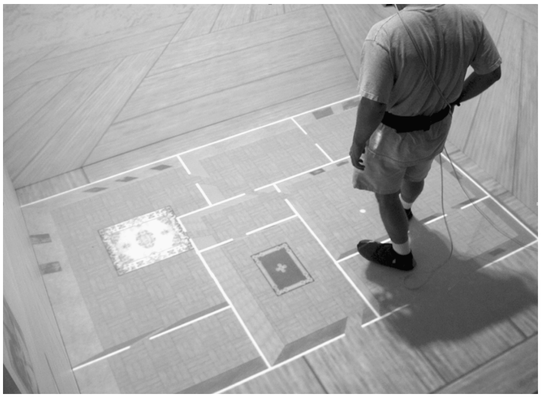

Figure 7.9 *A floor-based map called the Step WIM.*

Compasses

A *compass* (Figure 7.10) can also provide directional cues. For a trained navigator, a compass in combination with a map is an invaluable wayfinding tool. Most users of 3D UIs, however, will not be familiar with effective methods for using compass information. As a VE wayfinding aid, compasses are typically found in navigation training tools, such as those used in the military.

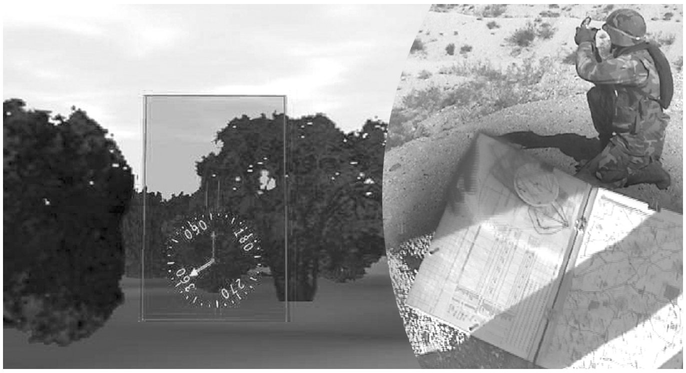

Figure 7.10 *Compasses in virtual and real world scenarios. (Images courtesy of the MOVES Institute, Naval Postgraduate School)*

Signs

Signs are used extensively in real-world environments to provide spatial knowledge and directions (Figure 7.11), but surprisingly there is little research on the use of signs as a wayfinding cue in VEs. Signs can be extremely effective because of their directness, but signs can also become confusing in complex environments (think about badly designed airports). Signs should be placed in an easily observable location, should supply clear directions, and should be spaced far enough apart that multiple signs do not confuse the user.

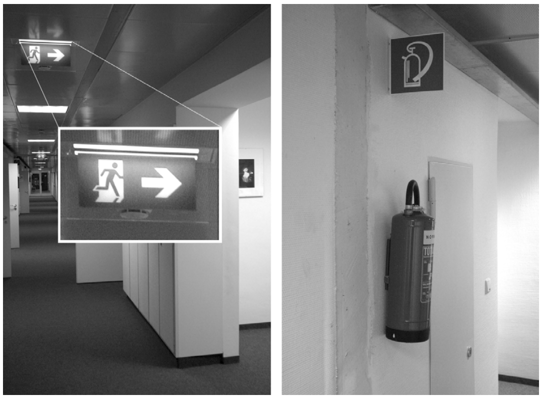

Figure 7.11 *Use of signs to direct wayfinders. (Photograph courtesy of Ernst Kruijff)*

Reference Objects

Reference objects are objects that have a well-known size, such as a chair or a human figure, and aid in size and distance estimation. Users often have difficulty judging distances in large, mostly empty environments. Since there are often no reference objects in these large spaces with which to compare the size of the room, distances are highly under- or overestimated. As soon as reference objects are placed in such a space, estimation of sizes and distances becomes easier.

Artificial Landmarks

Artificial landmarks are another group of cues that are similar to reference objects. These are easily distinguishable objects that can be used to maintain spatial orientation, develop landmark and/or route knowledge, and serve as foundations for distance or direction estimation. Although landmarks are naturally part of a legible environment design, artificial landmarks may be added to any environment to support users' wayfinding tasks.

Landmarks are most often implemented in the environment itself, although they can be used as tools. For example, in Worldlets , a desktop-based landmark application, the landmark becomes a tool like a compass. It is held in the hand or displayed on a surface (Figure 7.12). The user can

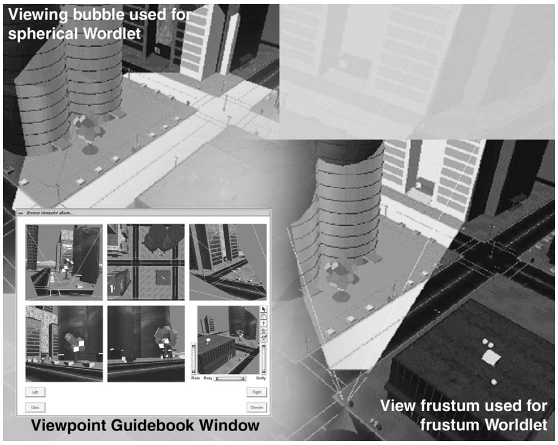
Viewing bubble used for spherical Wordlet
Viewpoint Guidebook Window
View frustum used for frustum Worldlet

Figure 7.12 *Spherical and frustum Worldlets, which can be viewed from within the Viewpoint Guidebook Window. (Image courtesy of T. Todd Elvins and David R. Nadeau, San Diego Supercomputer Center)*

turn the landmark and look at it from different sides to obtain knowledge of the environment around it and possibly use it to find the landmark's location in the full-scale environment.

We can identify two different sorts of landmarks: the *local* and the *global landmark.* Global landmarks are visible from practically any location, so they provide directional cues similar to a compass. Local landmarks help users in the decision-making process—when a decision point is reached, the available local landmarks provide useful information (Steck and Mallot 2000).

Several studies have investigated the use of landmarks in VEs (e.g., Vinson 1999), leading to two design guidelines.

Use clearly distinguishable visual characteristics.

It is important that a user be able to distinguish the landmark from other surrounding objects within the environment. Therefore, we should "contrast" its visual characteristics by using a contrasting color, different lighting, a contrasting form, or a different size.

Carefully pick the location of the landmark.

When placing the landmark, we can use the requirement of legibility to place it so it can easily be spotted, like a corner in a city structure, instead of placing it within a city block. Use the structure of the environment to support the detection of the landmark.

Trails

In order to help the user "retrace his steps" in an environment, or to show which parts of the world have been visited, *trails* can be included as an artificial wayfinding aid. A trail can be made up of a simple line or by using markers that include directional information, just like footprints in the real world. A trail can be placed directly into the environment, but can also be shown on a map (Darken and Peterson 2002; Grammenos et al. 2002).

Audio and Olfactory Cues

Audio is another artificial wayfinding cue. For example, speech can be used to explain the route to a user, as in modern car navigation systems.

As noted earlier, audio can also be coupled to objects to provide distance and directional cues or to uniquely identify an object. Imagine a dripping faucet to lead you to the virtual kitchen! Similarly, olfactory cues could be implemented—some objects have a distinct smell (a nasty smell like a factory or a pleasant one like a garden). Even though direction and distance can hardly be communicated via current olfactory interfaces, they, just like audio, can become a unique identifier.

7.5. Evaluating Wayfinding Aids

The performance of wayfinding aids and the usefulness of a VE for transferring spatial knowledge to the real world can be tested by a variety of methods (Darken and Peterson 2002). Figure 7.13 shows one setup that has been used to evaluate wayfinding performance. Although we discuss 3D UI evaluation in general in Chapter 11, here we discuss a few evaluation metrics that apply specifically to the evaluation of wayfinding.

First, *time-to-target* tests have been performed to analyze the user's movement time between two arbitrary points in a VE. This is a measure of the efficiency gains (or losses) provided by a wayfinding aid. Second, *path analysis* (see Figure 7.13, left) can be used to analyze the way the user moves through an environment. An ideal path should be predefined to

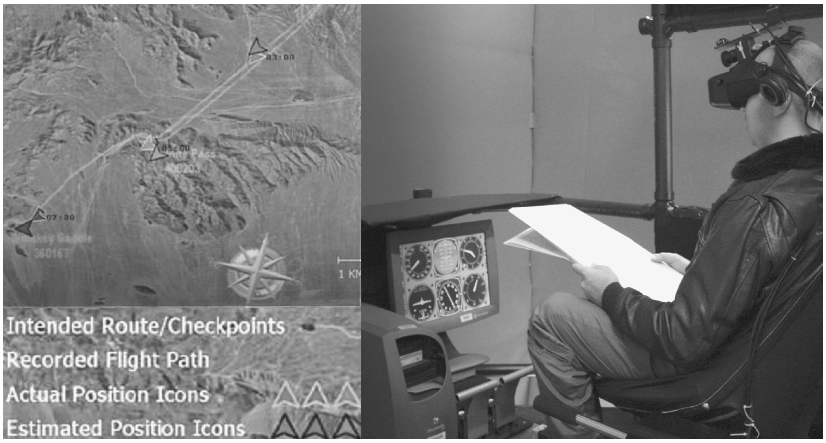

Figure 7.13 *Wayfinding test environment. Using chroma keying, the pilot sees the real map and the cockpit as well as the VE. On the left is an image of a map study's results. (Images courtesy of the MOVES Institute, Naval Postgraduate School)*

which the comparison can be made. If a user turns around and retraces his movements several times, for example, then obviously the existing wayfinding aids are not sufficient.

Asking the user to draw *layout sketches* (simple maps) of an environment can be a powerful method of defining the quality of the spatial knowledge acquired during movement through the environment. These sketches are also an indication of the likelihood of transferring spatial knowledge from the VE to the real world. The sketches can be analyzed for the user's perception of the overall structure of the environment, including relative and absolute sizes of objects and spaces, the location of objects like landmarks, and directional information.

Most evaluations of the performance of wayfinding aids have focused on visual cues. Research on the performance of cues using other sensory channels and on the relationships between sensory channels are fruitful areas for future work.

7.6. Design Guidelines

Although guidelines specific to certain types of wayfinding aids have appeared throughout the chapters, here we collect some general guidelines. One particular area of focus is the relationship between travel techniques and wayfinding—how travel techniques affect wayfinding and how travel techniques can be improved to support wayfinding tasks.

Match the cue to the task.

It is important that the characteristics of a wayfinding cue match the task requirements. For example, a map can be very powerful in a real-world setting like a landscape or building environment, but might simply be confusing in a more abstract environment.

Match the cue to users' skill.

Experience influences the way that people find their way through an environment. Therefore, it also influences the way that we need to use wayfinding aids in a VE. For example, implementing landmarks can be an effective way of supporting novice users, while experts may choose not to use such cues.

Don't make cues dominant features.

Subtlety is important, especially in environment design. When wayfinding cues become the dominant features in an environment, they may also turn out to be counterproductive. A cue should be seen as a tool to ease the user's navigation through an environment without being the sole point of information retrieval. For example, Darken and Sibert (1996) stated that the accentuated artificial landmarks in their environments led partly to users moving through the environment from one landmark to another. This may actually discourage the acquisition of spatial knowledge: a user may stick to a basic mental representation of the environment depending heavily on the dominant cue and missing other important spatial information about the environment.

Choose input devices providing real motion cues if possible.

The choice of the input device used for travel can affect wayfinding performance to a considerable extent. Vestibular cues (real motion cues) support the egocentric reference frame—real motion cues deliver important distance information. When users can walk naturally by using a wide-area-tracking system or a locomotion device such as a treadmill (Chapter 6, section 6.3.2), vestibular cues are present. However, when only passive travel techniques are used, real motion cues are limited to physical rotation, or they are missing altogether. The lack of real-motion cues may lead to false distance perception and therefore more difficulty in wayfinding tasks.

Avoid teleportation.

Travel techniques that teleport the user directly from one location to another are not recommended when applications require effective wayfinding. Users are often disoriented after teleportation, since they need to rebuild their spatial awareness (Bowman et al. 1997). Even with target-based travel techniques, smooth motion from one location to another is preferred. The same study, however, found that velocity does not have a noticeable effect on spatial orientation. Thus, the intermediate motion can be at high speed as long as the environment remains recognizable.

Integrate travel and wayfinding components.

Since travel and wayfinding are intimately linked, techniques for these two tasks should be integrated if possible. In some cases, devices like a treadmill allow this directly by coupling a method of travel with a vestibular-feedback component. Other techniques have inherent proprioceptive cues. Gaze-directed steering, for example, supplies directional information via headcentric cues. Wayfinding aids may actually be part of the travel technique. For example, the world-in-miniature technique combines a 3D map with a route-planning travel metaphor (Chapter 6, section 6.3.4). Finally, wayfinding aids can be placed in the environment near the focus of the user's attention during travel. For example, a small compass can be attached to the (real or virtual) tip of a stylus when the pointing technique (Chapter 6, section 6.3.3) is used.

7.7. Conclusions

Wayfinding is a complex issue. Due to the wide variety of different environments, with regard to both hardware and content, it can be challenging to support effective wayfinding. In this chapter, we looked at a variety of wayfinding aids, cues, and techniques, including solutions based on technology, training, environment design, and tools. VE wayfinding research is still relatively immature. We encourage readers to experiment with different combinations of aids that may yield superior wayfinding performance.

Recommended Reading

For further information on perception and cognition, we recommend the following reading:

Anderson, J. (1983). *The Architecture of Cognition,* Harvard University Press.

Newell, A., P. Rosenbloom, and J. Laird (1989). Symbolic Architectures for Cognition. *Foundations in Cognitive Science.* M. Posner (Ed.), MIT Press, 93–131.

Kosslyn, S. (1993). *Image and Brain,* MIT Press.

Marr, D. (1982). *Vision,* W.H. Freeman.

Johnson-Laird, P. (1993). *The Computer and the Mind: An Introduction to Cognitive Science.* London, Fontana Press.

Winston, P. (1993). *Artificial Intelligence,* 3rd ed., Addison-Wesley.

For an introduction to the effects of the sense of presence on wayfinding, we recommend the following:

Usoh, M., K. Arthur, M. Whitton, R. Bastos, A. Steed, M. Slater, and F. Brooks Jr. (1999). Walking > Walking-in-Place > Flying in Virtual Environments. *Proceedings of SIGGRAPH '99,* ACM Press, 359–364.

Regenbrecht, H., T. Schubert, and F. Friedman (1998). Measuring the Sense of Presence and its Relations to Fear of Heights in Virtual Environments. *International Journal of Human-Computer Interaction* 10(3): 233–250.

For an example of a study on the effects of wayfinding in training transfer, we recommend the following:

Darken, R., and W. Banker (1998). Navigating in Natural Environments: A Virtual Environment Training Transfer Study. *Proceedings of the 1998 IEEE Virtual Reality Annual International Symposium (VRAIS '98),* IEEE Press, 12–19.

CHAPTER 8

System Control

In 2D interfaces, UI elements such as pull-down menus, pop-up menus, toolboxes, palettes, toggles, radio buttons, and checkboxes are everywhere. These elements are examples of *system control* techniques—they allow us to send commands to an application, change a mode, or modify a parameter. Although we don't think much about the design of such techniques in 2D UIs, system control interfaces for 3D UIs are not trivial. Simply adapting 2D desktop-based widgets is not the ultimate solution. In this chapter, we discuss and compare various system control solutions for 3D UIs.

8.1. Introduction

The issuing of *commands* is a critical way to access any computer system's functionality. For example, with traditional desktop computers, we may want to save a document or change from a brush tool to an eraser tool in a painting application. In order to perform such tasks, we use graphical user interface (GUI) system control techniques like menus or function keys on a keyboard. Research in desktop system control has provided us with a wealth of techniques, such as those used in the WIMP (Windows, Icons, Menus, Pointers) metaphor (Preece et al. 2002).

Although much of the "real work" in a computer application consists of interaction tasks like selection, manipulation, and symbolic input, system control is critical because it is the "glue" that allows the user to control the interaction flow between the other key tasks in an application. For example, in writing this book with a word processor, the core activity is symbolic input, accomplished by typing on a keyboard. But this activity is interspersed with many small system control tasks—saving the current document by clicking on a button, inserting a picture by choosing an item from a menu, or underlining a piece of text by using a keyboard shortcut, just to name a few.

We can more precisely define system control as follows:

System control is the user task in which a command is issued to

1. request the system to perform a particular function,
2. change the mode of interaction, or
3. change the system state.

The key word in this definition is *command*. In selection, manipulation, and travel tasks, the user typically specifies not only *what* should be done, but also *how* it should be done, more or less directly controlling the action. In system control tasks, the user typically specifies only *what* should be done and leaves it up to the system to determine the details. In part 1of the definition, the system has a self-contained piece of functionality (e.g., for making plain text bold), and the user simply requests (commands) this function to be executed. Part 2 indicates a task such as choosing a new tool from a toolbox—nothing has changed besides the interaction mode. Changing the system state, part 3 of the definition, is exemplified by tasks such as clicking on a window to bring it to the front—that window becomes the "current window" in the system state, and subsequent actions are applied to it.

How can system control tasks be performed in 3D UIs? In 2D interfaces, system control is supported by the use of a specific *interaction style,* such as pull-down menus, text-based command lines, or tool palettes (Preece et al. 2002). Many of these interaction styles have also been adapted to 3D UIs (see section 8.3), which is certainly appropriate for desktop-based 3D UIs. In immersive and semi-immersive environments, however, WIMP-style interaction may not be effective in all situations. We cannot assume that simply transferring conventional interaction styles will lead to usability. In immersive VEs, users have to deal with 6-DOF input as opposed to 2 DOF on the desktop, and the input and output devices used in VEs differ considerably from the keyboard and mouse.

These differences create both new problems and new possibilities for system control. Therefore, in non-desktop 3D UIs, it may be more appropriate to use *nonconventional* system control techniques (Bullinger et al. 1997).

What determines the success (usability and performance) of a system control technique in a 3D UI? Before describing specific techniques, let's consider three sets of factors that influence the effectiveness of all techniques: human factors, input devices, and system- and application-level factors.

8.1.1. Human Factors of System Control

When designing system controls, we can learn much from the design of mechanical systems (Bullinger et al. 1997). In mechanical systems, control refers to the transfer of mechanical energy or information to a system for performing control actions. The interaction between the control device and the human's body is called the *control-body linkage.* As with all other interaction techniques, the user's physical characteristics, training, and experience level affect the operating effectiveness of the control-body linkage. Other factors that often affect user performance in traditional mechanical control systems include the shape and size of controls, their visual representation and labelling, methods of selection, and underlying control structures.

Many of these factors can be applied directly to the design of system control techniques for 3D UIs. For example, think about a menu in which the user rotates her hand to select an item (a "1 DOF menu," see section 8.3.1). When designing this menu technique, we need to consider how the user can rotate her wrist and the placement and size of the menu items that will lead to comfortable and efficient selection. When we design the menu poorly (e.g., when the user needs to turn her wrist to an uncomfortable position to select the items at the edges of the menu), the menu will not be very usable.

8.1.2. Input Devices

Many of the system control techniques described in this chapter are connected to particular input devices; for example, voice commands require use of a microphone. Hence, the properties of input devices influence the design of many system control interaction techniques. Some techniques, however, are device-independent.

The input devices used for system control techniques in 3D UIs can also have an effect on user performance and comfort. For example, a menu can be placed on a tracked physical surface, and the menu items can be selected with a tracked physical pen (the pen-and-tablet technique). The constraint provided by the physical surface will increase efficiency and accuracy. On the other hand, the user may tire more quickly when holding two physical devices.

The number and placement of buttons on input devices is a third factor that may influence the usability of system control techniques in 3D UIs. Multiple buttons allow "lightweight" mode switching and more flexibility of expression. The common context-sensitive menu in desktop interfaces, for example, is usually accessed via the right mouse button. However, multiple buttons can also lead to user confusion and error, especially if the mapping between buttons and functionality is inconsistent or unclear.

8.1.3. System- and Application-Level Factors

Factors related to the system's implementation can also influence the effectiveness of system control in a 3D UI. For example, recognition errors in speech or gesture interfaces can significantly reduce user performance and perceived usability.

The complexity of the application also plays an important role. With an increase in functionality, issuing commands becomes more difficult, since the user needs to know how to access all the functionality of the system. The system can be structured to help the user in this task. In addition, system control techniques that work well for accessing 10 commands may be completely unusable if the number of commands swells to 100.

8.1.4. Chapter Roadmap

We begin this chapter with a classification of system control for 3D UIs (section 8.2). Next, we describe each of the major categories in this classification (sections 8.3–8.6). In each of these sections, we describe representative techniques, discuss the relevant design and implementation issues, and provide guidance on the practical application of the techniques. In section 8.7, we cover multimodal system control techniques, which combine multiple methods of input to improve usability and performance. Section 8.8 describes some important design guidelines and the specific interrelationships between system control and other interaction techniques. Finally, a case study (section 8.9) serves a real-world example of system control design in a 3D UI.

Note that the techniques described in this chapter come mostly from work in immersive VEs. In many cases, however, the techniques or the principles they represent might also be used in augmented reality or desktop 3D UIs.

8.2. Classification

While there is a broad diversity of system control techniques for 3D UIs, many of them draw upon a small number of basic metaphors or their combination. Figure 8.1 presents a classification of techniques organized around these metaphors. This classification was also influenced by the description of nonconventional control techniques in (MacMillan et al. 1997). As was the case in the other chapters on 3D interaction techniques, this classification is only one of many possibilities, and alternatives have been proposed (e.g., Lindeman et al. 1999). However, we find this classification useful in understanding and discussing 3D system control techniques, and we use it as the organizing structure for the next four sections.

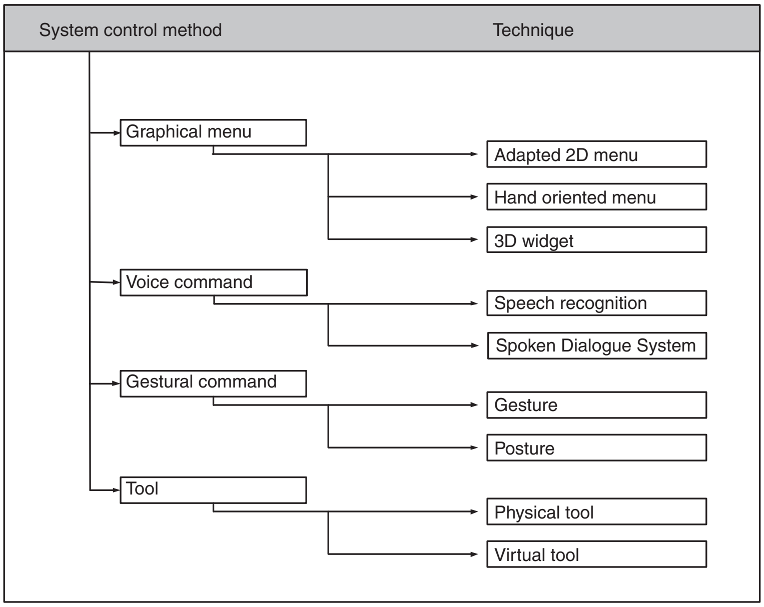
System control method
Technique
Graphical menu
Adapted 2D menu
Hand oriented menu
3D widget
Voice command
Speech recognition
Spoken Dialogue System
Gestural command
Gesture
Posture
Tool
Physical tool
Virtual tool

Figure 8.1 *Classification of system control techniques.*

8.3. Graphical Menus

Graphical menus for 3D UIs are the 3D equivalent of the 2D menus that have proven to be a successful system control technique in desktop UIs. Because of their success and familiarity to users, many developers have chosen to experiment with graphical menus for 3D UIs.

8.3.1. Techniques

In this section, we describe four techniques:

- adapted 2D menus
- 1-DOF menus
- TULIP menus
- 3D widgets

Adapted 2D Menus

Menus that are simple adaptations of their 2D counterparts have, for obvious reasons, been the most popular group of 3D system control techniques. These menus basically function in the same way as they do on the desktop. Some examples of adapted 2D menus are pull-down menus, pop-up menus, floating menus, and toolbars. Figure 8.2 shows an example of adapted 2D menu used in a Virtual Museum application in a

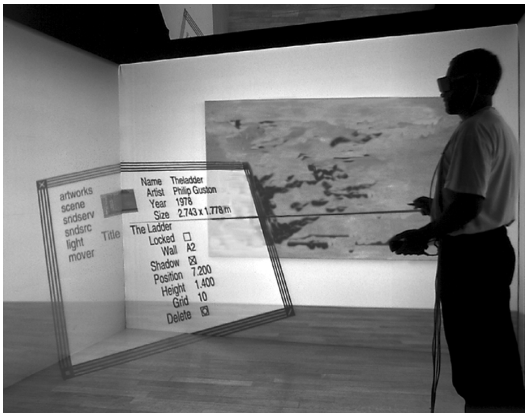
artworks
scene
sndserv
sndsrc
light
mover
Title
Name Theladder
Artist Philip Guston
Year 1978
Size 2.743 x 1.778 m
The Ladder
Locked ☐
Wall A2
Shadow ☒
Position 7.200
Height 1.400
Grid 10
Delete ☒

Figure 8.2 *A floating menu in the Virtual Museum application.(Photograph courtesy of Gerhard Eckel, Fraunhofer IMK)*

surround-screen display. It allows a user to plan an exhibition by finding and selecting images of artwork available for the exhibition. The menu is semitransparent to reduce occlusion of the 3D environment.

One adaptation of 2D menus that has been successful in 3D UIs is to attach the menus to the user's head—this way, the menu is always available, no matter where the user is looking. Another powerful technique is to attach the menu to a tracked physical surface (a tablet). Finding the menu is then as easy as bringing the physical tablet into view. The physical surface of the tablet also helps the user to select the menu items, and the menu can easily be put away as well.

The main advantage of adapted 2D menus is their familiar interaction style. Almost all users will instantly recognize these elements as menus and will understand how to use them. On the other hand, these menus can occlude the environment, and users may have trouble finding the menu or selecting items within it using a 3D selection technique (see the section on placement issues below).

1-DOF Menus

Selection of an item from a menu is essentially a one-dimensional operation. This observation led to the development of *1-DOF menus.* A 1-DOF menu is often attached to the user's hand, with the menu items arranged in a circular pattern around the hand (Figure 8.3); this design led to the name *ring menu* (Liang and Green 1994; Shaw and Green 1994). With this

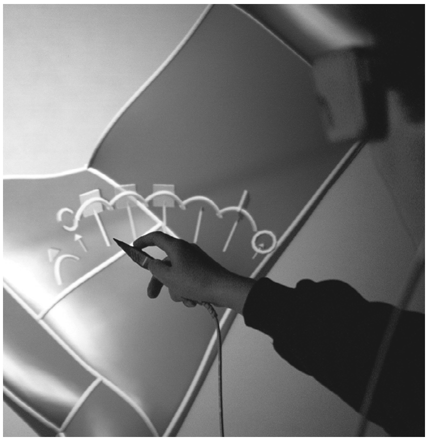

Figure 8.3 *A 1-DOF menu. (Photograph courtesy of Gerold Wesche, Fraunhofer IMK)*

design, the user rotates his hand until the desired item falls within a "selection basket." Of course, the hand rotation or movement can also be mapped onto a linear menu; it does not have to be circular. The performance of a ring menu depends on the physical movement of the hand and wrist, and the primary axis of rotation should be carefully chosen. Of course, hand rotation is only one possible way to select an item in a 1-DOF ring menu. For example, the user could rotate the desired item into position with the use of a button or buttons on the input device.

Handheld widgets are another type of 1-DOF menu. These do not use rotation, but instead relative hand position (Mine et al. 1997). By moving the hands closer together or further apart, different items in the menu can be selected.

In general, 1-DOF menus are quite easy to use. Menu items can be selected quickly, as long as the number of items is relatively small. Due to the strong placement cue, 1-DOF menus also afford rapid access and use—the user does not have to find the menu if it is attached to his hand and does not have to switch his focus away from the area in which he is performing actions.

TULIP Menus

Another method of attaching a menu to the user's hand in a 3D UI is to assign menu items to different fingers. Using Pinch Gloves (see Chapter 4), the system can interpret a pinch between a finger and the thumb on the same hand as a menu selection. If there are no more than eight menu items, this technique works very well. Up to 16 menu items can be accommodated if the items are organized into four menus with four items each—the non-dominant hand can be used to select a menu, and the dominant hand to select an item within the menu.

In many applications, however, there will be many more than 16 menu items. The TULIP (Three-Up, Labels In Palm) technique was designed to address this problem (Bowman and Wingrave 2001). In TULIP (Figure 8.4), when there are more than four items in a menu, the first three items are displayed, attached to the user's index, middle, and ring fingers. The pinky finger is labelled "more," and selecting this item moves the next three items in the menu onto the user's fingers. All of the inactive items are displayed, in groups of three, on the palm of the virtual hand. In this way, the selection of items is still direct, and users can see all of the items and how to access them. An empirical study has shown that this technique is moderately efficient and extremely comfortable and easy to use (Bowman and Wingrave 2001).

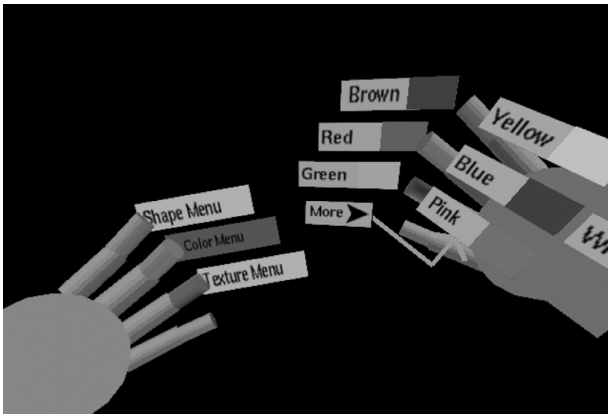

Figure 8.4 *TULIP menus. (Bowman and Wingrave 2001, © 2001 IEEE)*

3D Widgets

The most exotic group of graphical menu techniques for system control are 3D widgets. They take advantage of the extra DOF available in a 3D environment to enable more complex menu structures or better visual affordances for menu entries. We distinguish between two kinds of 3D widgets: colocated (context-sensitive) widgets and non-context-sensitive widgets.

In colocated widgets, the functionality of a menu is moved onto an object in the 3D environment, and geometry and functionality are strongly coupled. Conner and colleagues (1992) refer to widgets as "the combination of geometry and behavior." For example, suppose a user wishes to manipulate a simple geometric object like a box. We could design an interface in which the user first chooses a manipulation mode (e.g., translation, scaling, rotation) from a menu, and then manipulates the box directly. With colocated 3D widgets, however, we can place the menu items directly on the box—menu functionality is directly connected to the object (Figure 8.5). To scale the box, the user simply selects and moves the scaling widget, thus combining the mode selection and the manipulation into a single step. The widgets are context-sensitive; only those widgets that apply to an object appear when the object is selected. As in the example, colocated widgets are typically used for changing geometric parameters.

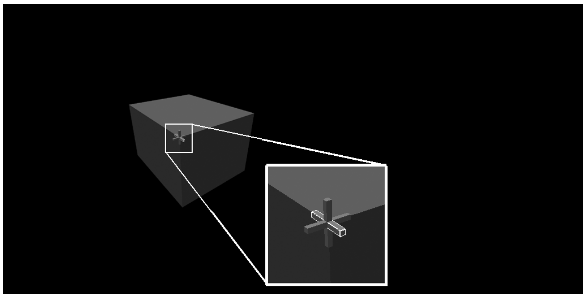

Figure 8.5 *A 3D colocated widget for scaling an object. (Image courtesy of Andrew Forsberg, Brown University Computer Graphics Group)*

A variety of colocated 3D widgets are shown in the context of a scientific visualization application in Figure 8.6: the widget near the front of the space shuttle is used to change the number of streamlines used to examine the flow field as well as their location in 3D space. The widget near the rear of the space shuttle is used to adjust the size and location of a color plane.

The command and control cube, or C^3, is a more general-purpose type of 3D widget (non-context-sensitive). The C^3 (Figure 8.7) is a $3 \times 3 \times 3$ cubic grid, where each of the 27 grid cubes is a menu item. The user

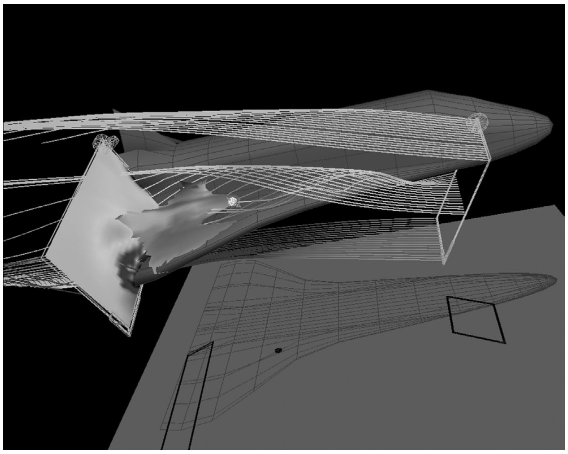

Figure 8.6 *A variety of 3D widgets used in a scientific visualization application. (Image courtesy of Andrew Forsberg, Brown University Computer Graphics Group)*

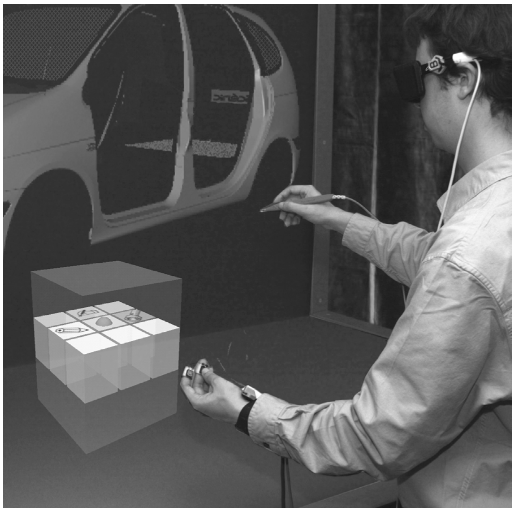

Figure 8.7 *The command and control cube. (i3D-INRIA. Data © Renault. Photograph courtesy of Jerome Grosjean)*

brings up the menu by pressing a button or making a pinch on a Pinch Glove; the menu appears, centered on the user's hand. Then the user moves her hand in the direction of the desired menu item cube relative to the center position, and releases the button or the pinch. This is similar in concept to "marking menus" (Kurtenbach and Buxton 1991) used in software such as Maya from Alias Wavefront.

8.3.2. Design and Implementation Issues

There are many considerations when designing or implementing graphical menus as system control techniques in a 3D UI. We discuss the issues of placement, selection, representation, and structure.

Placement

The placement of the menu influences the user's ability to access the menu (good placement provides a spatial reference) and the amount of occlusion of the environment. We can consider menus that are *world-referenced, object-referenced*, *head-referenced*, *body-referenced*, or *device-referenced* (adapted from the classification in Feiner et al. 1993).

World-referenced menus are placed at a fixed location in the virtual world, while object-referenced menus are attached to an object in the 3D scene. Although not useful for most general-purpose menus, these may be useful for colocated 3D widgets.

Head-referenced or body-referenced menus, such as TULIP menus (attached to the hand), provide a strong spatial reference frame: the user can easily find the menu. Mine, Brooks, and Sequin (1997) explored body-referenced menus and found that the user's proprioceptive sense (sense of the relative locations of the parts of the body in space) can significantly enhance menu retrieval and usage. Body-referenced menus may even enable eyes-off usage, allowing users to perform system control tasks without having to look at the menu.

The last reference frame is the group of device-referenced menus. For example, on a workbench display, menus may be placed on the border of the display device. The display screen provides a physical surface for menu selection as well as a strong spatial reference.

Selection

Traditionally, desktop menus make use of a 2D selection method (mouse-based). In a 3D UI, we encounter the problem of using a 3D selection method with these 2D (or 1D) menus. This makes interaction with the system control interface particularly difficult. In order to solve this problem, several alternative selection methods have been developed that simply constrain the DOF of the system control interface, considerably improving performance. For example, when an adapted 2D menu is shown, one can discard all tracker data except the 2D projection of the tracker on the plane of the menu. Two-DOF selection techniques such as ray-casting or image plane selection also address this issue (see Chapter 5). Finally, the menu can be placed on a physical 2D surface such as a screen or a tracked tablet in order to reduce the DOF of the selection task (see Chapter 10).

Representation and Structure

Another important issue in developing a graphical menu is its representation: how are the items represented visually, and if there are many items, how are they structured?

Because of the technology used in VEs, the size of items and the space between them is very important. Do not make items and inter-item distances too small, or the user might have problems selecting the items. The more complex the application gets, the more functions will be available. Make sure to structure the interface by using either functional grouping (items with similar function are clustered) or sequential grouping (using the natural sequence of operations to structure items), or by using context-sensitive menus so that only the applicable functions are displayed when the user accesses the menu (Salvendy 1997). Finally, control coding, which uses different colors, shapes, surfaces, textures, dimensions, positions, text, and symbols to differentiate items, can give an extra cue about the relations between different items and therefore make the structure and the hierarchy of the items more clear (Bullinger et al. 1997).

8.3.3. Practical Application

Menu techniques can be very powerful in 3D UIs when their limitations can be overcome. Selection of menu items should be easy, and the menu should not overlap too much with the workspace in which the user is working. Especially with applications that have a large number of functions, a menu is probably the best choice of all the system control techniques for 3D UIs. Finally, if the 3D graphical menus presented here are simply not usable enough for a particular application, developers may choose to move the menus to a dedicated 2D device like a PDA or Tablet PC (Figure 8.8). This approach works only when users can see the physical world (not in an HMD-based system), but it ensures usability. Of

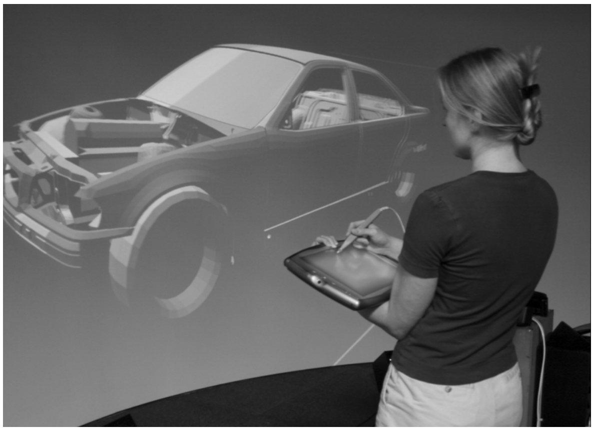

Figure 8.8 *User in the iCone display system using a remote interface. (Photograph courtesy of Fraunhofer IMK)*

course, this type of setup is also more expensive, more cumbersome, and perhaps more difficult to implement.

8.4. Voice Commands

The issuing of voice commands can be performed via simple speech recognition or by means of spoken dialogue techniques. Speech recognition techniques are typically used for issuing single commands to the system, while a spoken dialogue technique is actively focused on promoting discourse between the user and the system.

8.4.1. Techniques

A spoken dialogue system provides an interface between a user and a computer-based application that permits spoken interaction with the application in a relatively natural manner (McTear 2002). The most critical component of a spoken dialogue system (and of simple speech recognition techniques) is the *speech recognition engine.* A wide range of factors influences the speech recognition rate, such as variability among speakers and background noise. The recognition engine can be speaker-dependent, requiring initial training of the system, or speaker-independent, which normally does not require training. Systems also differ in the size of their vocabulary. The response generated as output to the user can confirm that an action has been performed or inform the user that more input is needed so as to complete a control command. In a spoken dialogue system, the response should be adapted to the flow of discourse (requiring a dialogue control mechanism) and generated as artificial speech.

Current speech recognition packages include IBM ViaVoice and HARK, while current spoken dialogue systems include CSLU toolkit, IBM voice server, Speechworks, and NLSA. More information on these systems, including references to academic work, can be found in (McTear 2002).

Many 3D UIs that use speech recognition also include other complementary input methods (e.g., Billinghurst 1998). These techniques are labelled *multimodal* and are discussed in section 8.7.

8.4.2. Design and Implementation Issues

The development of a 3D UI using simplified speech recognition or the more complex spoken dialogue systems involves many factors. A developer should start by defining which tasks need to be performed via voice interfaces. For an application with a limited number of functions, a normal speech recognition system will probably work well. The task will define the vocabulary size of the speech engine—the more complex the task and the domain in which it is performed, the more likely it is that the vocabulary size will increase. Highly complex applications may need conversational UIs via a spoken dialogue system in order to ensure that the full functionality of voice input is accessible. In the case of a spoken dialogue system, it should also be considered which input (vocal "information") is needed in order to determine the user's intentions.

Developers should be aware that voice interfaces are *invisible to the user.* The user is normally not presented with an overview of the functions that can be performed via a speech interface. Therefore, in order to grasp the actual intentions of the user, one of the key factors is verification. Either by error-correction via semantic and syntactic filtering (prediction methods that use the semantics or syntax of a sentence to limit the possible interpretation) or by a formal discourse model (question-and-answer mechanism), the system must ensure that it understands what the user wants.

Unlike other system control techniques, speech-based techniques initialize, select, and issue a command "all at once." Sometimes, another input stream (like a button press) or a specific voice command should be used to initialize the speech system. This disambiguates the start of a voice input and is called a push-to-talk system (also see Chapter 4). Error rates will increase when, for instance, the application involves direct communication between multiple participants. Here, a comment to a colleague can easily be misunderstood as a voice command to a system. Therefore, one should separate human–human and human–computer interaction when designing speech interfaces. Syntactic differences between personal communication and system interaction might be used to distinguish between voice streams (Shneiderman 2000).

8.4.3. Practical Application

Speech input as a system control technique in a 3D UI can be very powerful—it is hands-free and natural. Still, continuous voice input is tiring and cannot be used in every environment.

The user first needs to learn the voice commands before they can be issued. A user can easily learn simple commands for a limited number of functions. However, voice commands are probably not very useful in applications that allow only short learning times or no learning at all.

Finally, we note that similar interface issues have been studied in many different contexts. For example, speech commands for controlling a system via a telephone poses many of the same problems as using voice commands in a VE. Please refer to Brewster (1998) for a further discussion of issues involved in such communication streams.

8.5. Gestural Commands

Gestures were one of the first system control techniques for VEs and other 3D environments. Ever since early projects like Krueger's Videoplace (Krueger et al. 1985), developers have been fascinated by using the hands as direct input, almost as if one is not using an input device at all. Gestural commands can be classified as either *postures* or *gestures.* A posture is a static configuration of the hand (Figure 8.9), whereas a gesture is a dynamic movement. An example of a posture is holding the fingers in a V-like configuration (the victory sign), whereas waving and drawing are examples of gestures. The usability of gestures and postures for system control depends on the number and complexity of the gestural commands—more gestures imply more learning for the user.

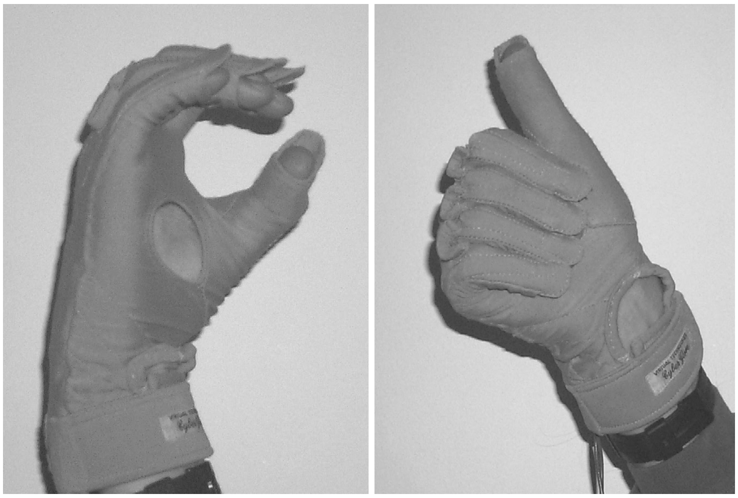

Figure 8.9 *Examples of postures using a DataGlove. (Photograph courtesy of Joseph J. LaViola Jr.)*

8.5.1. Techniques

One of the best examples to illustrate the diversity of gestural commands is Polyshop (later Multigen's SmartScene; Mapes and Moshell 1995). In this VE application, all interaction was specified by postures and gestures, from navigation to the usage of menus. For example, the user could move forward by pinching an imaginary rope and pulling herself along it (the "grabbing the air" technique—see Chapter 6). As this example shows, system control overlaps with manipulation and navigation in such a 3D UI. Consider the definition of system control as being the "change of mode of interaction." In Polyshop, the switch to navigation mode is lightweight and effective, since no "active" change of mode is performed.

In everyday life, we use many different types of gestures , and these categories also apply to the use of gestures in 3D UIs:

- *Speech-connected hand gestures*: Spontaneous gesticulation performed unintentionally during speech or language-like gestures that are integrated in the speech performance. Speech-connected gestures have been studied intensely in HCI and applied to multimodal interfaces (e.g., Bolt 1980).
- *Mimic gestures*: Gestures that are not connected to speech but are directly used to "describe" a concept. For example, Figure 8.10 shows a sweeping gesture in 3D space that defines a curved surface (Schkolne et al. 2001).

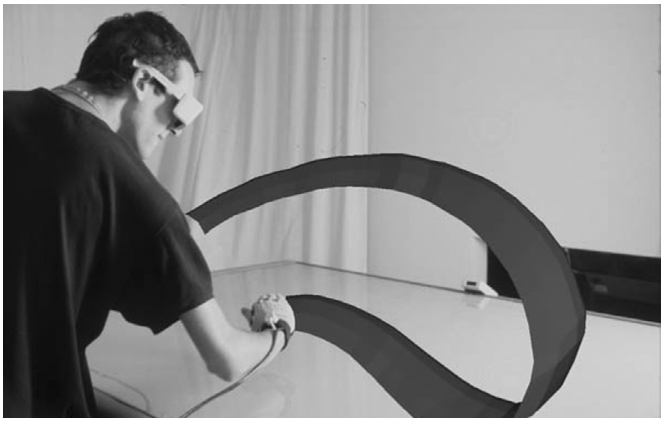

Figure 8.10 *Modeling gesture. (Photograph courtesy of Steve Schkolne)*

- *Symbolic gestures*: Gestures as used in daily life to express things like insults or praise (e.g., "thumbs up")
- *Sign language*: The use of a specified set of postures and gestures in communicating with hearing-impaired people. At least one 3D UI project has used sign-language-like gestures for communication (Fels 1994).

8.5.2. Design and Implementation Issues

The implementation of gestural input is usually tied to the input device used. Here are the major types of gesture input techniques:

- *Glove-based recognition*: Glove-like devices (see Chapter 4) analyze the raw data coming from their sensors with recognition algorithms such as hidden Markov models and neural networks. The hand has been used as button, valuator, locator, and pick device (Zimmerman et al. 1987; Sturman et al. 1989). Pinch Gloves can be used for limited postures, while DataGloves provide both postures and gestures using joint-angle measurements.
- *Camera-based recognition*: Video images of hand or finger gestures can be analyzed by using feature-recognition methods (computer vision methods) to recognize specific configurations of the hand (Starner et al. 1998).
- *Surface-based recognition*: Display screens, touch screens, or other flat surfaces can be used for gestures (Rekimoto 2002). Typically, a penlike device is used to make gestures on the flat surface. Here, the gestures do not involve the whole hand at all, but rather the strokes created by the pen (Figure 8.11).

Gesture-based system control shares many of the characteristics of speech input discussed in the previous section. Like speech, a gestural command combines initialization, selection, and issuing of the command. In addition, the available gestures in the system are typically invisible to the user. Finally, the user may have trouble remembering a large number of gestures. Thus, as with push-to-talk in speech interfaces, the UI designer should ensure that the user really intends to issue a gestural command via some implicit or explicit mechanism (this could be called a "push-to-gesture" technique). The number of gestures should be limited, and they should be highly learnable. The system should also provide adequate feedback to the user when a gesture is recognized.

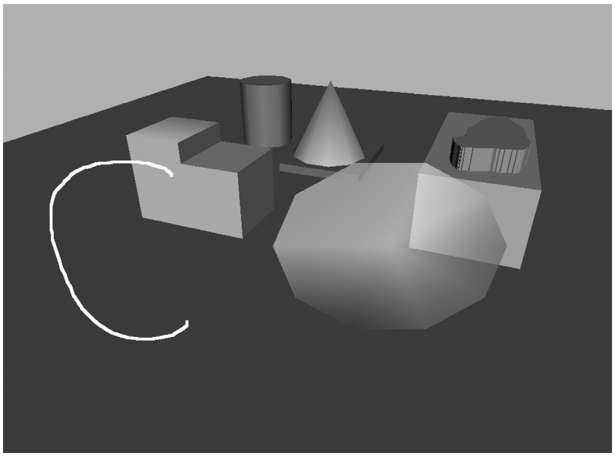

Figure 8.11 *A C-gesture used to select the color picker in the SKETCH application image courtesy of Brown University Computer Graphics Group)*

8.5.3. Practical Application

Gestural commands have significant appeal for system control in 3D UIs because of their important role in our day-to-day lives. However, with a few notable exceptions, such as the surface-based gestural interfaces of Teddy (Igarashi et al. 1999) and SKETCH (Zeleznik 1996), purely gestural system control interfaces have not been extremely successful. Choose gestural commands if the application domain already has a set of well-defined, natural, easy-to-understand, and easy-to-recognize gestures. In addition, gestures may be more useful in combination with another type of input (see section 8.7). For further reading on gestural interaction, please refer to Bordegoni and Hemmje (1993), Mapes and Moshell (1995), and LaViola (1999a).

8.6. Tools

In many 3D applications, the use of familiar (real-world) devices for 3D interaction can lead to increased usability. These devices, often called *tools,* provide directness of interaction due to their real-world correspondence. While individual tools may be used for selection, manipulation, travel, or other 3D interaction tasks, we consider a set of tools in a single

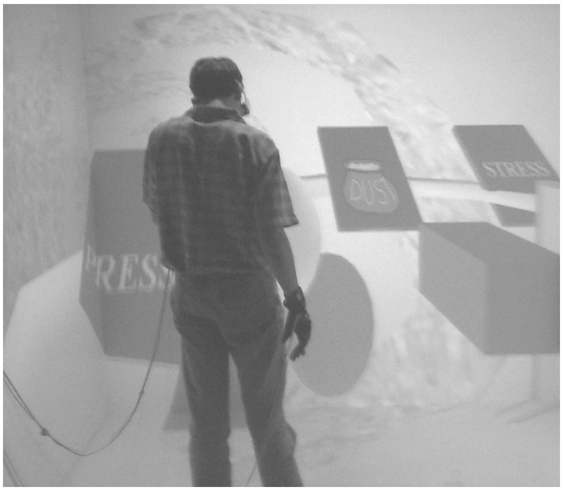

Figure 8.12 *Tool belt menu. (Photograph reprinted from Forsberg et al. 2000, © 2000 IEEE)*

application to be a system control technique. Like the tool palettes in many popular 2D drawing applications, tools in 3D UIs provide a simple and intuitive technique for changing the mode of interaction: simply select an appropriate tool.

We distinguish between two kinds of tools: physical tools and virtual tools. Physical tools are a collection of real physical objects (with corresponding virtual representations) that are also sometimes called *props.* A physical tool might be space-multiplexed (the tool only performs one function) or time-multiplexed (the tool performs multiple functions over time). A user accesses a physical tool by simply picking it up and using it.

Virtual tools have no physical instantiation. This can best be exemplified with a "tool belt" technique (Figure 8.12). Users wear a physical tool belt and can select various virtual tools by touching particular locations on the belt.

8.6.1. Techniques

A wide range of virtual tool belts exists, but they are largely undocumented in the literature. Therefore, we focus on the use of physical tools, as used for system control in 3D UIs, in this section. A more general discussion on props-based interaction can be found in Chapter 10.

Based on the idea of props, a whole range of tangible user interfaces (TUIs) has appeared. TUIs make use of real-world objects to perform actions in a VE (Ullmer and Ishii 2001; Fitzmaurice et al. 1995). A TUI uses physical elements that represent a specific kind of action in order to interact with an application. For example, the user could use a real eraser to delete virtual objects or a real pencil to draw in the virtual space.

Figure 8.13 shows a TUI for 3D interaction. Here, Ethernet-linked "interaction pads" representing different operations are used together with radio frequency identification (RFID) tagged physical cards, blocks, and wheels, which represent network-based data, parameters, tools, people, and applications. Designed for use in immersive 3D environments as well as on the desktop, these physical devices ease access to key information and operations. When used with immersive VEs, they allow one hand to continuously manipulate a tracking wand or stylus, while the second hand can be used in parallel to load and save data, steer parameters, activate teleconference links, and perform other operations. Functioning

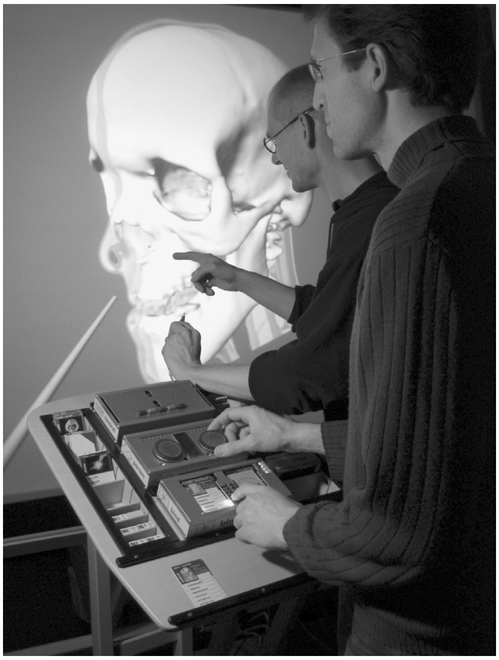

Figure 8.13 *Visualization artifacts—physical tools for mediating interaction with 3D UIs. (Image courtesy of Brygg Ullmer and Stefan Zachow, Zuse Institute Berlin)*

prototypes of this sort are beginning to be used by physicians and astrophysicists.

A TUI takes the approach of combining representation and control. This implies the combination of both physical representations and digital representations, or the fusion of input and output in one mediator. TUIs have the following key characteristics (from Ullmer and Ishii 2001):

- Physical representations are computationally coupled to underlying digital information.
- Physical representations embody mechanisms for interactive control.
- Physical representations are perceptually coupled to actively mediated digital representation.

These ideas can be applied to develop props-based "physical menus." In HMD-based VEs, for example, a tracked pen can be used to select from a virtual menu placed on a tracked tablet . In a projection-based VE, a transparent physical tablet can be used to achieve the same effect—the menu is displayed on the visual output device (projection screen) but is correctly aligned with the tablet so that the user sees the menu appear on the tablet. An example of this approach is the Personal Interaction Panel (Schmalsteig et al. 1999).

The principal advantage of displaying a menu on a tablet is the direct haptic feedback to the user who interacts with the menu. This results in far fewer selection problems compared to a menu that simply floats in the VE space.

An example of a slightly more sophisticated approach to using props and tools is the Virtual Tricorder (Wloka and Greenfield 1995). In this technique, a real physical 3D mouse is registered to its virtual copy inside the VE (Figure 8.14). By pressing buttons on the mouse, the user can access the menu and choose the desired tool from it. The functionality and virtual 3D appearance of the mouse then changes according to the selected tool. The strength of this approach is that it produces a single multipurpose tool for 3D interaction.

8.6.2. Design and Implementation Issues

The form of the tool communicates the function the user can perform with the tool, so carefully consider the form when developing props. A general approach is to imitate a traditional control design, for example, in

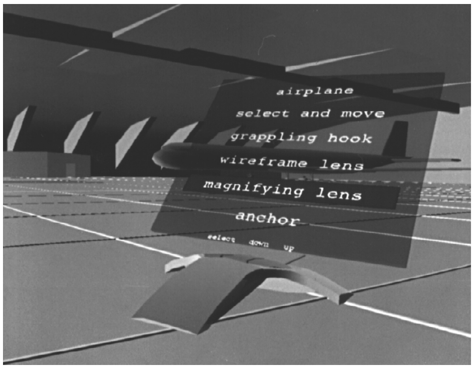

Figure 8.14 *The Virtual Tricorder. (Image reprinted with permission from van Dam et al. 2000; © 2000 IEEE)*

machinery design. Another approach is to duplicate everyday tools in the VE. The user makes use of either the real tool or something closely resembling the tool in order to manipulate objects in a VE.

Another important issue is the *compliance* between the real and virtual worlds (Hinckley et al. 1994). Other prop-based interfaces, like the Cubic Mouse (Fröhlich and Plate 2000), have demonstrated a need for a *clutching* mechanism. See Chapter 5 for more information on compliance and clutching in manipulation techniques.

The use of props naturally affords blind operation (the user can operate the device by touch), which may have significant advantages, especially when the user needs to focus visual attention on another task. On the other hand, it also means that the prop must be designed to allow tactile interaction. A simple tracked tablet, for example, does not indicate the locations of menu items with haptic cues; it only indicates the general location of the menu.

A specific issue for physical menus is that the user may want to place the menu out of sight when it is not in use. The designer may choose to put a clip on the tablet so that the user can attach it to his clothing, may reserve a special place in the display environment for it, or may simply provide a handle on the tablet so it can be held comfortably at the user's side.

8.6.3. Practical Application

Physical tools are very specific devices. In many cases, they perform only one function. In applications with a great deal of functionality, tools can still be useful, but they may not apply to all the user tasks. There is a tradeoff between the specificity of the tool (a good affordance for its function) and the amount of tool switching the user will have to do. Performing a simple user study will quickly reveal any problems with device switching.

Public installations of VEs (e.g., in museums) can greatly benefit from the use of tools. Users of public installations by definition must be able to use the interface immediately. Tools tend to allow exactly this. A well-designed tool has a strong affordance, and users may draw from personal experience with a similar device in real life. Many theme park installations make use of props to allow the user to begin playing right away. For example, the Pirates of the Caribbean ride at DisneyQuest uses a physical steering wheel and cannons. This application has almost no learning curve—including the vocal introduction, users can start interacting with the environment in less than a minute.

8.7. Multimodal System Control Techniques

The classification of system control techniques presented in Figure 8.1 does not contain the last group of system control techniques: *multimodal techniques,* which combine multiple input streams. In certain situations, the use of multimodal system control techniques can significantly increase the effectiveness of system control tasks.

Multimodal interaction is the use of multiple input channels (e.g., speech and gesture) to control a system. Following are some of the advantages for using multimodal system control in VEs:

- *Decoupling*: Using an input channel that differs from the main input channel used for interaction with the environment, and thus decreasing user cognitive load. If users do not have to switch between manipulation and system control actions, they can keep their attention focused on their main activity.
- *Error reduction and correction*: The use of multiple input channels can be very effective when the input is ambiguous or noisy, especially with recognition-based input like speech or gestures. The combination of input from several channels can significantly increase recognition rate (Oviatt 1999; Oviatt and Cohen 2000).
- *Flexibility and complementary behavior*: Control is more flexible when users can use multiple input channels to perform the same

task. In addition, different modalities can be used in a complementary way based on the perceptual structure of the task (Grasso et al. 1998; Jacob and Siebert 1992).

- *Control of mental resources*: Multimodal interaction can be used to reduce cognitive load (Rosenfeld et al. 2001); on the other hand, it may also lead to less effective interaction, since multiple mental resources need to be accessed simultaneously. For example, as Shneiderman (2000) observes, the part of the human brain used for speaking and listening is the same one as used for problem solving—speaking consumes precious cognitive resources.

Probably the best-known multimodal technique is the famous "put-that-there" technique (Bolt 1980). Using this technique, users can perform actions by combining pointing with speech. Many others have used the same combination of gesture and speech (e.g., figures 8.l5 and 8.16). In some cases' speech can be used to disambiguate a gesture, and vice versa.

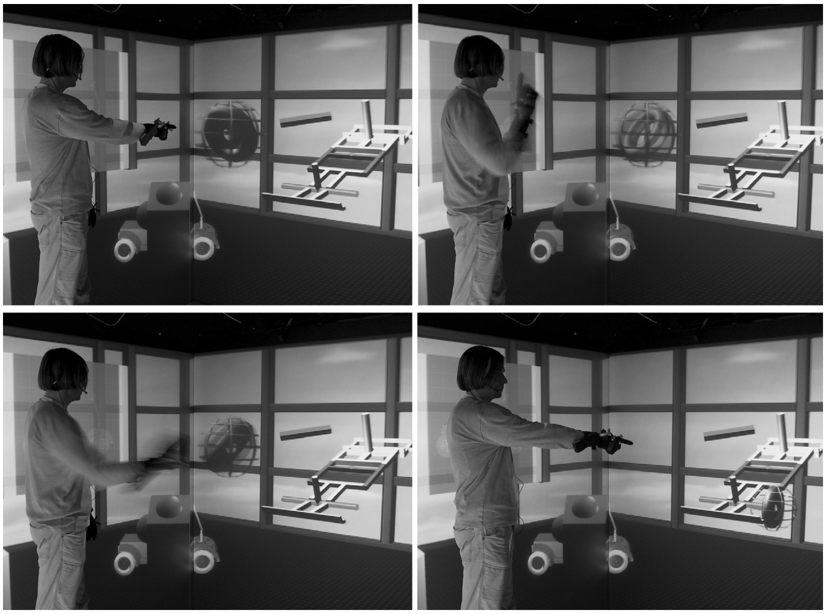

Figure 8.15 *A car wheel is selected, rotated, and moved to its correct position using voice and gestures. (Photographs courtesy of Marc Eric Latoschik, AI & VR Lab, University of Bielefeld; Latoschik 2001)*

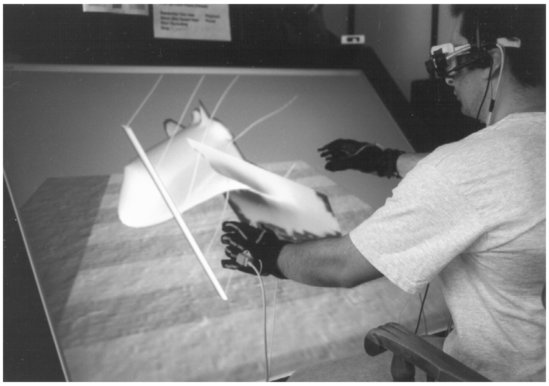

Figure 8.16 *A multimodal interface that combines hand gestures and speech used for scientific visualization. (Photograph reprinted from van Dam et al. 2000; © 2000 IEEE)*

Another possible technique is to combine gesture-based techniques with traditional menus, as in the "marking menus" technique. This means that novice users can select a command from a menu, while more experienced users can access commands directly via gestural input. This redundancy is similar to the use of keyboard shortcuts in desktop interfaces.

8.8. Design Guidelines

Throughout this chapter, we have presented many design guidelines for specific 3D system control techniques. In this section, we summarize some overall guidelines. Due to the relative lack of empirical evaluations of system control techniques for 3D UIs, however, these guidelines are primarily based on anecdotal evidence and personal experience. For now, therefore, most of the guidelines should be regarded as rules of thumb.

Avoid disturbing the flow of action of an interaction task.

System control is often integrated with another 3D interaction task. Such a task structure forms a "chain of actions." Due to this integration, system control techniques should be designed to avoid disturbing the flow of action of an interaction task. Lightweight mode switching, physical tools, and multimodal techniques can all be used to maintain the flow of action.

Prevent unnecessary changes of the focus of attention.

One of the major interruptions to a flow of action is a change of attentional focus. This may occur when users have to cognitively and/or physically switch between the actual working area and a system control technique, or even when they must look away to switch devices.

Avoid mode errors.

Always provide clear feedback to the user so that she knows which interaction mode is currently active.

Use an appropriate spatial reference frame.

Placing your system control technique in the "right position" can make a big difference in its usability. Users often get distracted when searching for a way to change the mode of interaction or issue a command. If the controls are not visible at all, placed far away from the actual focal area, or not oriented toward the user, the result will be wasted time. On the other hand, system controls attached to the user's hand, body, or a device are always available.

Structure the functions in an application.

There are several good techniques for structuring the functionality of an application, including hierarchical menu structures and context-sensitive system control. In cases where the number of functions is so large that these techniques are not effective, it can make sense to place some of the system control functionality on another device, such as a PDA, where resolution and selection accuracy are less of an issue.

Consider using multimodal input.

Using multimodal input can provide more fluid and efficient system control, but can also have its drawbacks.

8.9. Case Study: Mixing System Control Methods

System control in 3D UIs is a complex topic, and the techniques for it are wide ranging. Thus, it is useful to examine a case study of system control techniques in a complex 3D application in order to illuminate some of the important issues involved in system control design for 3D UIs.

8.9.1. The ProViT Application

The case study examines a VE application called ProViT . This application is a distributed engineering environment in which several remote collaborators cooperate on complex design review tasks. The application allows users to run simulations, manipulate 3D objects, access data from other applications, and communicate with one another via videoconferencing, all within the context of an immersive VE.

ProViT uses two display devices: a large stereoscopic L-shaped workbench (Figure 8.17) and a 12-inch tablet PC (Figure 8.18). It also uses three input devices: a tracked stylus, a Cubic Mouse (see Chapter 4, section 4.4), and the tablet PC.

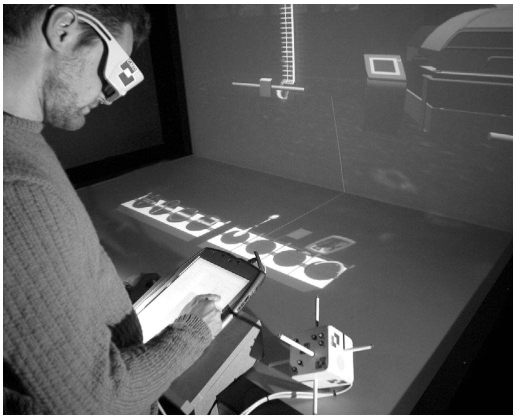

Figure 8.17 *A user controlling the distributed engineering application (Photograph courtesy of Ernst Kruijff)*

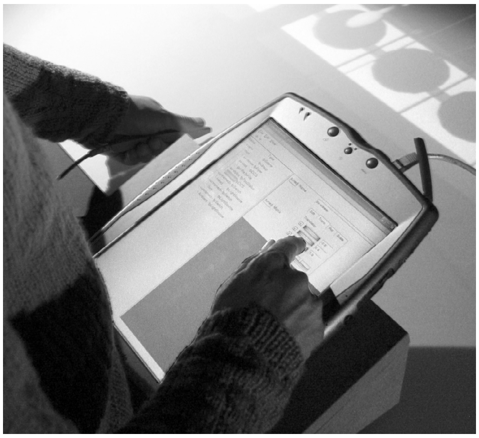

Figure 8.18 *Remote user interface. (Photograph courtesy of Stefan Conrad, Fraunhofer IMK)*

8.9.2. System Control Design Approach for ProViT

The high-level approach used in designing the 3D UI for this application was to separate the 3D and 2D interaction. Predictably, the inherently 3D actions are performed with the stylus or the Cubic Mouse, whereas 2D actions (like numeric input) are done via the tablet PC. This use of multiple devices to control the application could result in increased performance, but also creates a more complex input structure, which needs to be supported by techniques for maintaining the flow of action.

8.9.3. Mapping of Tasks to Devices

Considering the functions that this application must support, a 3D UI could have been designed that supported all of the functionality with a single device. In fact, this would have been similar to most other complex VE applications. The designers felt, however, that efficiency and usability could be improved with the use of multiple devices, even if users sometimes had to stop what they were doing to switch devices.

The mapping of tasks to devices has a direct effect on the usability of system control tasks. In this application, the three devices have different inherent features and limitations for system control. The tablet PC brings with it a wide range of highly optimized 2D system control techniques.

The Cubic Mouse has a large number of physical buttons, but these buttons have few natural mappings to functions. The stylus must make use of virtual system control techniques, since it has only a single button. In this application, the designers decided to use a hand-referenced 1-DOF menu (section 8.3.1) with the stylus.

Two of the devices can be used together: the stylus can be used to interact directly with the tablet PC. This eliminates the need to put down the stylus or pick up another device to access the 2D GUI.

Some functions, like navigation, were explicitly mapped to multiple devices to avoid device switching. For example, the user often needs to change the viewpoint in between manipulation actions. If the navigation function were mapped only to one device, this would automatically result in frequent device switching. In the current design, the user can use either the Cubic Mouse or the stylus for 3D navigation.

8.9.4. Placement of System Control

The second main issue in this application is the placement of its visible elements. The main working area and main focus of attention is the center of the workbench. In general, the designers wanted the user to be able to maintain focus on this area. In addition, this area should not be highly occluded by system control elements or widgets. These two goals are somewhat in conflict.

The hand-referenced 1-DOF menu addresses the first goal. It appears attached to the stylus, which is usually located in or near the working area. The user can always find the menu—no visual search is required—but this menu might occlude the VE content.

The GUI elements on the tablet PC meet the second goal. It places menus and complex information in one place (a display attached to the front of the workbench), and this information does not occlude the VE content. However, to use the tablet PC, the user must shift focus from one area to another, requiring significant head movements as the user performs an action in the GUI and then verifies that action in the 3D environment.

8.9.5. System Control Feedback

Finally, it is of utmost importance that feedback to the user is robust and understandable. Due to the multiple input devices used in this application, the designers provide feedback that does not depend on the input device in use. Instead, the feedback is attached to one of the displays and is consistent no matter what device or function the user happens to be using.

One very simple solution is a mode cue that is displayed in a small text bar in a corner of the workbench. The text bar shows the current interaction mode and is placed close to the user's working area. The second approach is to use the GUI on the tablet PC as a feedback method. Since the GUI is constantly synchronized with the VE application, the user can always look at the GUI in order to obtain a direct and detailed overview of the state of the system.

8.10. Conclusions

System control for 3D UIs is only in its infancy as a research topic. Although we have discussed a wide range of techniques, the design space for such techniques is virtually limitless. We expect to see many new and interesting 3D system control techniques, not only within the categories described here, but also in new categories that have not yet been invented. There is also a lack of good empirical evidence for the usability of various system control techniques at the present time—usability evaluations are desperately needed in order to validate the current design guidelines and develop new ones. Nevertheless, we hope this chapter has served to demonstrate the importance and complexity of system control interfaces and has provided some ideas for the design of novel 3D UIs for system control.

Recommended Reading

A general introduction to system control issues from a human factors background can be found in the following:

> Bullinger, H., P. Kern, and M. Braun (1997). Controls. *Handbook of Human Factors and Ergonomics.* G. Salvendy (Ed.), John Wiley & Sons, 697–728.