Supporting Multi-User Interaction in Co-Located and Remote Augmented Reality by Improving Reference Performance and Decreasing Physical Interference

Ohan Oda

Submitted in partial fulfillment of the
requirements for the degree of
Doctor of Philosophy
in the Graduate School of Arts and Sciences

COLUMBIA UNIVERSITY
2016

ABSTRACT

Supporting Multi-User Interaction in Co-Located and Remote Augmented Reality by Improving Reference Performance and Decreasing Physical Interference

Ohan Oda

One of the most fundamental components of our daily lives is social interaction, ranging from simple activities, such as purchasing a donut in a bakery on the way to work, to complex ones, such as instructing a remote colleague how to repair a broken automobile. While we interact with others, various challenges may arise, such as miscommunication or physical interference. In a bakery, a clerk may misunderstand the donut at which a customer was pointing due to the uncertainty of their finger direction. In a repair task, a technician may remove the wrong bolt and accidentally hit another user while replacing broken parts due to unclear instructions and lack of attention while communicating with a remote advisor.

This dissertation explores techniques for supporting multi-user 3D interaction in augmented reality in a way that addresses these challenges. Augmented Reality (AR) refers to interactively overlaying geometrically registered virtual media on the real world. In particular, we address how an AR system can use overlaid graphics to assist users in referencing local objects accurately and remote objects efficiently, and prevent co-located users from physically interfering with each other. My thesis is that our techniques can provide more accurate referencing for co-located and efficient referencing for remote users and lessen interference among users.

First, we present and evaluate an AR referencing technique for shared environments that is designed to improve the accuracy with which one user (the *indicator*) can point out a real physical object to another user (the *recipient*). Our technique is intended for use in otherwise

unmodeled environments in which objects in the environment, and the hand of the indicator, are interactively observed by a depth camera, and both users wear tracked see-through displays. This technique allows the indicator to bring a copy of a portion of the physical environment closer and indicate a selection in the copy. At the same time, the recipient gets to see the indicator's live interaction represented virtually in another copy that is brought closer to the recipient, and is also shown the mapping between their copy and the actual portion of the physical environment. A formal user study confirms that our technique performs significantly more accurately than comparison techniques in situations in which the participating users have sufficiently different views of the scene.

Second, we extend the idea of using a copy (virtual replica) of physical object to help a remote expert assist a local user in performing a task in the local user's environment. We develop an approach that uses Virtual Reality (VR) or AR for the remote expert, and AR for the local user. It allows the expert to create and manipulate virtual replicas of physical objects in the local environment to refer to parts of those physical objects and to indicate actions on them. The expert demonstrates actions in 3D by manipulating virtual replicas, supported by constraints and annotations. We performed a user study of a 6DOF alignment task, a key operation in many physical task domains. We compared our approach with another 3D approach that also uses virtual replicas, in which the remote expert identifies corresponding pairs of points to align on a pair of objects, and a 2D approach in which the expert uses a 2D tablet-based drawing system similar to sketching systems developed for prior work by others on remote assistance. The study shows the 3D demonstration approach to be faster than the others.

Third, we present an interference avoidance technique (Redirected Motion) intended to lessen the chance of physical interference among users with tracked hand-held displays, while

minimizing their awareness that the technique is being applied. This interaction technique warps virtual space by shifting the virtual location of a user's hand-held display. We conducted a formal user study to evaluate Redirected Motion against other approaches that either modify what a user sees or hears, or restrict the interaction capabilities users have. Our study was performed using a game we developed, in which two players moved their hand-held displays rapidly in the space around a shared gameboard. Our analysis showed that Redirected Motion effectively and imperceptibly kept players further apart physically than the other techniques.

These interaction techniques were implemented using an extensible programming framework we developed for supporting a broad range of multi-user immersive AR applications. This framework, Goblin XNA, integrates a 3D scene graph with support for 6DOF tracking, rigid body physics simulation, networking, shaders, particle systems, and 2D user interface primitives.

In summary, we showed that our referencing approaches can enhance multi-user AR by improving accuracy for co-located users and increasing efficiency for remote users. In addition, we demonstrated that our interference-avoidance approach can lessen the chance of unwanted physical interference between co-located users, without their being aware of its use.

Table of Contents

List of Figures ..... **iv**

List of Tables ..... **x**

Acknowledgements ..... **xi**

Chapter 1. Introduction **1**
1.1 Research Questions and Dissertation Goals 4
1.2 Contributions 6
1.2.1 Design, implementation, and evaluation of a referencing technique for physical objects in co-located multi-user AR 8
1.2.2 Design, implementation, and evaluation of a referencing technique for supporting remote task assistance in AR 15
1.2.3 Design, implementation and evaluation of a physical-interference–avoidance technique in co-located multi-user AR 22
1.3 Structure of Dissertation 27

Chapter 2. Related Work **28**
2.1 Computer-Assisted Multi-User Interaction 28
2.1.1 Focuses 28
2.1.2 Categories 30
2.2 Multi-User Interaction in AR 31
2.3 Selection Techniques 33
2.3.1 Immediate Selection Techniques 34
2.3.2 Refinement-based Selection Techniques 39
2.4 Visualization Techniques 41
2.4.1 Assisting Interpretation of Object of Interest 41
2.4.2 Assisting Interpretation of Intended Action 43

Chapter 3. A 3D Referencing Technique for Shared Augmented Reality Environments. 45
3.1 Related Work 47
3.2 GARDEN 50
3.2.1 Interaction by the Indicator 53
3.2.2 Interaction by the Recipient 59
3.3 Comparison Techniques 60
3.3.1 Laser Pointer 60
3.3.2 Virtual Arrow 61
3.3.3 Video Share 62
3.4 Implementation 63
3.4.1 Software 63
3.4.2 Hardware 64

3.4.3 Calibration 65
3.5 User Study 66
3.5.1 Pilot Studies 68
3.5.2 Study Description 69
3.5.3 Hypotheses 71
3.6 Analysis 72
3.6.1 Quantitative Analysis 73
3.6.2 Subjective Analysis 81
3.7 Discussion 83

Chapter 4. A 3D Referencing Technique for Supporting Remote Task Assistance in Augmented Reality 85
4.1 Related Work 88
4.2 Our Approach 91
4.2.1 DEMO3D 94
4.2.2 POINT3D (Comparison Technique) 98
4.2.3 SKETCH2D (Comparison Technique) 100
4.3 Interaction Environment 100
4.4 Implementation 101
4.4.1 DEMO3D Implementation 103
4.4.2 POINT3D Implementation 106
4.4.3 SKETCH2D Implementation 107
4.5 Pilot Studies 108
4.6 User Study 110
4.6.1 Hypotheses 110
4.6.2 Methods 112
4.7 Results 118
4.8 Discussion 123

Chapter 5. A Physical-Interference–Avoidance Technique for Co-Located Multi-User Augmented Reality 126
5.1 Related Work 127
5.2 Redirected Motion 131
5.3 Test Application: AR Domino Knockdown 136
5.3.1 Software 137
5.3.2 Hardware 137
5.3.3 Pilot Study 139
5.4 User Study 141
5.4.1 Comparison Techniques 142
5.4.2 Study Description 144
5.4.3 Hypotheses 146
5.5 Analysis 147
5.5.1 Quantitative Analysis 148
5.5.2 Subjective Analysis 152
5.6 Limitations and Applications 155

Chapter 6. Conclusions and Future Work 157
6.1 Summary of Contributions 157

6.2 Lessons Learned 159
6.3 Future Work 161
6.3.1 Hand Tracking 162
6.3.2 Dynamic Scene Reconstruction 163
6.3.3 Applying Redirected Motion to Remote Assistance 163
6.3.4 Extending Remote Assistance for Instructing Multiple Local Users 164
6.3.5 Allowing Remote Expert to Define Constraints 166
6.3.6 DEMO3D and POINT3D in a Partially or Unmodeled Environment 169
6.3.7 Exploring Additional Opportunities for Redirected Motion 170
6.4 Final Thoughts 171

References 172

Appendix A. Goblin XNA Framework 187
A.1. Software Architecture 189
A.1.1 Scene Graph 190
A.1.2 6DOF and 3DOF Tracking and Input Device Abstraction 198
A.1.3 Video Acquisition and Vision-based Tracking 200
A.1.4 Physics Engine 201
A.1.5 Networking 203
A.1.6 2D User Interface 205
A.1.7 Shaders 206
A.2. Toolkits 207
A.2.1 Camera Calibration 207
A.2.2 Stereo Camera Calibration 207
A.2.3 Marker Layout 208
A.2.4 Graphical Scene Graph Display 211
A.3. Iteration History 213
A.4. Projects Implemented with Goblin XNA 214
A.4.1 AR Racing Game 215
A.4.2 AR Marble Game 218
A.4.3 AR Modeling on Microsoft Surface 220
A.4.4 ARmonica 221

Appendix B. Study Questionnaires 223
B.1. Questionnaire for Chapter 3 223
B.2. Questionnaire for Chapter 4 238
B.3. Questionnaire for Chapter 5 245

References (Appendix) 256

List of Figures

Figure 1.1. Examples of activities in which multiple people interact. (a) Children playing with Lego blocks (http://picklebums.com/2009/09/07/toys-we-love-lego/). (b) Plant workers assembling a machine (http://karenmarcelo.org/srlyard/2007/09/robodock-assembling-machines-and-prop.html). 1

Figure 1.2. Examples of computer-assisted multi-user interactions. (a) Two users try to point at a specific bone structure of a virtual skeleton that appears in 3D stereo through synchronized pairs of special shutter glasses [Agrawala et al., 1997]. (b) Two users manipulate different parts of a projected 2D map through their hand-held projectors. One user controls a large map with fewer details (only main streets are labeled) while the other user controls the focus region (the brighter region) in which more details are displayed (smaller streets are also labeled in addition to the main streets) [Cao et al., 2007]. 2

Figure 1.3. Examples of collaborative AR. (a) Two users collaborating on urban planning [Broll et al., 2004]. (b) One user examining a 3D virtual model of a ceramic vessel while the other user is looking at a 3D terrain model of an archaeological dig site where the vessel was excavated [Benko et al., 2004]. 3

Figure 1.4. The indicator's views through a stereo video–see-through head-worn display. (a) Indicator specifying a spherical volume with a tracked hand-held device. (b) A virtual representation of the physical region constructed from depth and RGB data is brought closer to the indicator. (c) Indicator pointing directly with her finger at the representation to indicate the physical object to which she intends to refer (in this case, the center purple cylinder)..... 9

Figure 1.5. The recipient's views through a stereo video–see-through head-worn display. (a) Indicator specifying an approximate region. (Here, the indicator is selecting a different region from that in Figure 1.4.) Only the infinite ray is visible to the recipient, but not the spherical volume. (b) A close-up representation of the spherical volume selected by the indicator. (c) Recipient seeing the selected volume and the reconstructed virtual representation of the indicator's hand. 11

Figure 1.6. DEMO3D interaction technique allows a remote expert to guide a local user to place the top of an aircraft engine combustion chamber relative to its bottom by interacting with a *virtual replica* of the top. As seen by the expert in AR, looking through the local user's cameras, the expert has placed the transparent virtual replica on the chamber bottom. Pairs of metaobject annotations on the virtual replica and physical chamber top are linked with color-coded rubberband lines to show the correspondence between the physical top and its virtual replica, and are also seen by the local user. 15

Figure 1.7. POINT3D, as seen by the expert in VR: The expert places 3D annotations (e.g., blue cube) on the transparent virtual replica, specifying contact points. The same annotations appear on the opaque *virtual proxy* corresponding to the virtual replica. In Point3D, once the

expert specifies corresponding contact points on both objects (*A* and *B*), a color-coded rubberband line connecting the points appears between the proxies. 17

Figure 1.8. A local user placing a chamber top on a Lego fixture as instructed by the remote expert..... 18

Figure 1.9. Redirected Motion in sequence from top to bottom. As the green player (on the right) moves toward the red player (on the left), the virtual location of the green player is shifted along the green player's direction of movement, while the (stationary) red player's virtual location is not affected. 22

Figure 1.10. Player's view of our test application. Virtual balls are fired at virtual dominoes by tapping on the screen of a hand-held computer. 24

Figure 2.1. A two-by-two matrix that illustrates four different categories of CSCW systems (a.k.a. *groupware*) based on whether the collaboration happens at the same or different place and whether at the same or different time [Johansen, 1991]. 30

Figure 2.2. (Top row) User grasps onto a virtual object with bare hand pinch gesture [Piumsomboon et al., 2014]. (Bottom row) User picks up a virtual object projected onto a table by swiping it into the user's own hand. Once picked up, the virtual object moves with the user's hand [Wilson and Benko, 2010]. 31

Figure 2.3. Points to an item displayed on a large screen with an index finger [Bolt, 1980]. 34

Figure 2.4. Go-go interaction technique extends the user's virtual hand to select an object at distance [Poupyrev et al., 1996]. 37

Figure 2.5. Selection in a scaled version of the virtual world (WIM) [Stoakley et al., 1995]. 38

Figure 2.6. Voodoo dolls technique [Pierce et al., 1999]. (a) User first makes a copy of a virtual object using an image plane selection technique [Pierce et al., 1997]. (b) Once copy is created, user performs precise selection or operation on the copy within arm's reach. The result of the selection or operation is reflected on the original counterpart. 39

Figure 2.7. Cutaways and ghosting visualization [Feiner and Seligmann, 1992] exposes the occluded internal battery by rendering the occluding object with (a) a clear cutaway or (b) a feathered cutaway. 41

Figure 2.8. Techniques for assisting interpretation of intended action. (a) A visual hint demonstrating a twirling action to perform [White et al., 2007]. (b) Instructions with 3D arrows are projected onto user's hand to demonstrate how the user should move her hand [Sodhi et al., 2012]. 43

Figure 3.1. A selecting user (the *indicator*) points at a close-up virtual representation of a portion of the physical environment sensed by a depth camera with her finger to indicate the target object. (The coarseness of the close-up representation and reconstructed virtual representation of the indicator's hand in this figure and the following figures is due to low depth-camera resolution.) 45

Figure 3.2. Interaction in GARDEN for the indicator. (a) The indicator's view through a stereo video–see-through head-worn display while initially specifying a spherical volume with a tracked Wii remote. The volume is placed at the intersection of a ray with the live depth map of the physical environment. Objects are overlaid with numerical annotations used only in the

user study. (b) The texture-mapped depth mesh contained within the selection sphere is brought closer to the indicator. (c) The indicator points directly with her finger at the representation within the close-up mesh of the physical object to which she intends to refer (in this case, the purple cylinder at the center). 51

Figure 3.3. First outlining technique. (a) Enclosing outline (green lines showing the initial portion of an outline being drawn) is formed by connecting a list of intersected points between the infinite virtual ray (red line) originating from the tracked hand-held device and the depth map. (b) A view with the overlaid depth map (not shown to users). (c) The 3D mesh inside the enclosed outline with red boundary lines. 52

Figure 3.4. Interaction in GARDEN for the recipient. (a) The recipient's view through a stereo video–see-through head-worn display of the indicator's infinite ray. (b) A close-up representation of the spherical volume selected by the indicator. (Here, the indicator has selected a different region from that in Figure 3.2). (c) The recipient sees the selected volume and the reconstructed virtual representation of the indicator's hand. 57

Figure 3.5. Animation in GARDEN for the recipient. (a) The virtual close-up representation at its initial virtual location. (b) A copy of the virtual representation and the indicator's hand are interpolated between the initial virtual location and the actual physical location. The virtual representation is rendered as transparent, while the virtual hand is rendered opaque. This allows the recipient to see the actual physical object being referenced when the copy arrives at the destination and the virtual hand pointing at the actual physical object. (c) The virtual representation and hand arrive at the destination. 58

Figure 3.6. Laser Pointer. (a) A view from the indicator's perspective, showing the laser pointed at the object numbered "3". (b) A laser pointer used for this technique (Wii remote itself is not used). 60

Figure 3.7. Virtual Arrow. Viewed from the indicator's perspective. 61

Figure 3.8. Video Share. Viewed from the recipient's perspective. 62

Figure 3.9. Hardware used for the techniques and the study. (a) The indicator, with tracked head-worn display, hand-held Wii remote, and Kinect mounted on shelf. (b) The numeric keypad used by the recipient. 64

Figure 3.10. The physical objects used in the user study, with overlaid virtual numeric identifiers. (a) Set 1. (The object under the red "0" is the one the indicator should select during the current study trial.) (b) Set 2. 66

Figure 3.11. The study setup. 67

Figure 3.12. Mean completion time in seconds with standard deviation for (a) both sets, (b) set 1 only, and (c) set 2 only. Set 1 contains physical objects for which both participants could see roughly the same side of each object. Set 2 contains objects for which the indicator and recipient see opposite sides of each object. 73

Figure 3.13. Mean accuracy in percentage for physical objects in (a) both sets, (b) set 1 only, and (c) set 2 only. Set 1 contains physical objects for which both participants could see roughly the same side of each object. Set 2 contains objects for which the indicator and recipient see opposite sides of each object. 75

Figure 3.14. Mean unanswerable rate in percentage for (a) both sets, (b) set 1 only, and (c) set 2 only. Set 1 contains physical objects for which both participants could see roughly the same side of each object. Set 2 contains objects for which the indicator and recipient see opposite sides of each object. 78

Figure 3.15. Histograms of ease of selection (blue bar on the left of each pair) and interpretation (orange bar on the right of each pair). Likert-scale ratings from 1–5 given by the participants for the four techniques with mode and median. 5 is best. 81

Figure 3.16. Ranked qualitative ease, speed, and accuracy of selection (left columns) and interpretation (right columns) with SEM (Standard Error of the Mean) for the four techniques. 1 is best. 82

Figure 4.1. DEMO3D interaction technique allows a remote expert to guide a local user to place the top of an aircraft engine combustion chamber relative to its bottom by interacting with a *virtual replica* of the top. As seen by the expert in AR, looking through the local user's cameras, the expert has placed the transparent virtual replica on the chamber bottom. Pairs of metaobject annotations on the virtual replica and physical chamber top (its physical counterpart) are linked with color-coded rubberband lines to show the correspondence between the physical top and its virtual replica, and are also seen by the local user. 85

Figure 4.2. DEMO3D: (a) The expert manipulates the virtual replica to demonstrate how to place the top chamber on the bottom chamber with a specific orientation. (b) An AR view seen by the local user through a head-worn display, in which pairs of metaobject annotations on the virtual replica and its physical counterpart are connected with color-coded rubberband lines. 93

Figure 4.3. Constraints. (a) When the expert places and fine-tunes the chamber top relative to the bottom, the two can interpenetrate or not fit properly, as shown here, because there is no force feedback. (b) This can potentially mislead the local user. 94

Figure 4.4. POINT3D: (a) Once the expert specifies corresponding contact points (red spheres) on both objects, a color-coded rubberband line connecting the points appears between the proxies. (b) An AR view seen by the local user through a video–see-through head-worn display, in which the color-coded rubberband line appears between the physical counterparts. 97

Figure 4.5. SKETCH2D. Screenshots from the application. (a) The expert sketches a red line to indicate one of the contact points on the chamber top with multi-touch interaction. (Inset shows third-person view.) (b) The expert has sketched all three corresponding contact points to indicate how the chamber top should align with a Lego fixture. 99

Figure 4.6. A representation of the manipulation device in cubic box rendered with different colors. The box is rendered in (a) *gray* when not intersected with a virtual proxy, (b) *green* when intersected, (c) and *red* when grabbing a virtual replica. 101

Figure 4.7. An example of our constraint-based approach with a Lego fixture (see Section 4.6). (a) The expert places the virtual replica of chamber top in VR so that it interpenetrates a white post at the bottom right of the Lego fixture. (b) Seen from the local user's perspective. (c) The chamber top is within a threshold distance and orientation from a predefined constraint, causing the top to snap to the constraint with a smoothly interpolated transition. (d) Seen from the local user's perspective. 103

Figure 4.8. POINT3D. (a) The expert specifies three contact points (red sphere, green cylinder, and blue box) on both the chamber top and bottom, which in turn define the 6DOF pose of chamber top relative to chamber bottom. The pointing device is represented as an arrow with a color corresponding to the currently selected metaobject. We also display an indication of the currently selected metaobject (blue box) above the arrow. (b) The arrow appears relative to the corresponding physical counterpart of currently selected virtual replica or proxy in the local user's environment. 105

Figure 4.9. An expert wearing a tracked Sony HMZ-T3W head-worn display interacts with (a) a tracked mouse (manipulation device) and lazy susan turntable in DEMO3D, and (b) a tracked Leonar3Do bird controller (pointing device) and mouse in POINT3D. (c) In SKETCH2D, the expert interacts with (d) an untracked Samsung tablet. (e) A local user wearing a tracked Canon HM-A1 head-worn display places an aircraft engine combustion chamber top on a Lego fixture. 113

Figure 4.10. A few examples of possible 6DOF poses that can be specified between the chamber top and the Lego fixture. A 6DOF pose can be defined by choosing three resting points of the chamber top on the 3-level pegs: two from two neighboring pegs out of a ring of eight 3-level pegs, and one from the 3-level, 4-sided peg at the center of the ring. In addition, the chamber top can be rotated along its "up" axis at the same resting points for a different 6DOF pose (excluding those rotation angles where the protrusions on the chamber top would touch the pegs). 114

Figure 4.11. Mean completion time in seconds for (a) Novice Experts and (b) Trained Expert. Error bars show 95% confidence intervals normalized to remove between-subject variability. 119

Figure 4.12. Overall technique preference by role (1 = most preferred, 3 = least preferred). Error bars show 95% confidence intervals normalized to remove between-subject variability. ... 120

Figure 4.13. Mean values from unweighted NASA TLX survey with ratings from 1–7 (1 = best). Error bars show 95% confidence intervals normalized to remove between-subject variability. 120

Figure 4.14. Task duration breakdown for Novice Experts and Trained Expert. Error bars show 95% confidence intervals normalized to remove between-subject variability. 123

Figure 5.1. Classification of multi-user AR applications. 127

Figure 5.2. Redirected Motion. (Row 1) Changes in physical and virtual locations under Redirected Motion. As the green player (right) moves toward the red player (left), the virtual location of the green player is shifted along the green player's direction of movement, while the (stationary) red player's virtual location is not affected. (Row 2) First-person screenshot views for the green player with Redirected Motion, corresponding to images on row 1. (Row 3) First-person views for the green player without Redirected Motion (i.e., no virtual shifting), corresponding to images on row 2. (Row 4) As the green player moves back beyond the distance threshold τ from the red player, the virtual location of the green player is shifted back to its physical location. 130

Figure 5.3. User *A*, moving with velocity *v*, is treated as moving toward user *B* only if the angle between *v* and the vector from *A* to *B* is less than 45° (the red area). For example, the solid vector *v* is classified as moving toward user *B* and the dashed vector v' is not. 132

Figure 5.4. Physical and virtual locations of user *A* over three frames. In each frame, *A* indicates the current physical location, and *A*v indicates its shifted virtual location. Primed labels indicate previous physical and virtual locations. Solid vectors indicate motion of the physical location of *A*. Dashed vectors are components of the virtual shifting vector $\vec{G}$, exaggerated for this figure. (a–b) Additional virtual shift is applied along the motion vector from A' to *A*, since the user moves toward *B*. (c) Additional virtual shift is not applied, since the user does not move toward *B*. 134

Figure 5.5. Test Application. (a) Player's view of two-person AR Domino Knockdown game. Virtual balls are fired at virtual dominoes by tapping on the screen. (b) Third-person view of game with AR visualization of *Redirected Motion*. Each hand-held UMPC is overlaid with a simplified geometric model to represent its actual physical location, and a more transparent offset geometric model to represent its shifted virtual location. Two small magenta cubes highlight the two points (on the models at the physical locations) that are currently closest to each other. 136

Figure 5.6. (a) Mean game duration with SEM (Standard Error of the Mean). (b) Mean distance between participants' devices with SEM. 148

Figure 5.7. (a–e) Histograms of perceived effectiveness (blue) and distractibility (orange) Likert-scale ratings from 1–5 given by participants for five conditions with mode and median. (f) Means of perceived effectiveness (blue) and distractibility (orange) with SEM (Standard Error of the Mean). 5 is best. 152

Figure 5.8. Ranked qualitative effectiveness (a) and distractibility (b) with SEM (Standard Error of the Mean) for the five technique conditions (including None). 1 is best (perceived to be most effective, least distracting). 153

Figure 6.1. An expert points at the top part of an aircraft engine combustion chamber with one of two virtual hands tracked using the 3Gear Systems [3GearSystems, 2011] hand tracker.... 162

Figure 6.2. Blue circle, red triangle, and green rounded rectangle represent different objects in the local user's domain as virtual proxies. (a) An expert views and interacts with multiple local users' environments side-by-side. (b) An expert views and interacts with one local user's environment at a time, and switches between different local user's environments. .. 164

Figure 6.3. (a) All local users' environments have the same domain objects. (b) Each local user's environment has different domain objects. 165

Figure 6.4. (a) All local users' environments have the same initial layout. (b) Each local user's environment has a different initial layout. 165

Figure 6.5. (a–b) To restrict the top to 1DOF rotation along its "up" axis, an expert can define a constraint by placing the top above the bottom and indicating it can rotate only about its "up" axis, which allows these configurations. (c) However, this configuration would be prevented. (d) In one alternative, the expert can constrain the top to slide along and rotate around the red cylinder, restricting it to 1DOF translation and 1DOF rotation. 166

List of Tables

Table 4.1. SKETCH2D multi-finger gestures..... 107

Table 5.1. Summary of parameters used in user study. 139

Table 5.2. Paired Wilcoxon test results on perceived effectiveness (top) and distractibility, where N=None, SD=Screen Dimming, SB= Sound Beeping, AD=Action Disable, and RM=Redirected Motion. 154

Acknowledgements

I would like to express my deepest gratitude for members of Computer Graphics and User Interface (CGUI) lab, especially my advisor Steven Feiner. Without his tremendous effort for securing research funding and support for my academic writing, I wouldn't have been at this stage of completing my Ph.D degree. I also greatly appreciate his understanding the substantial amount of research time I put into developing the Goblin XNA framework, and thus allowing me to postpone some of the Ph.D milestone deadlines.

I would also like to express my appreciation to my committee members, John Kender, Gail Kaiser, Barbara Tversky, and Steven Henderson. Especially, I owe John and Gail for enduring me rescheduling my candidacy exam at the last minute and providing insightful suggestions during my thesis proposal presentation.

I thank my colleagues in the CGUI lab for their mental support and friendship: Sinem Güven, Gábor Blaskó, Hrvoje Benko, Edward Ishak, Marc Eaddy, Sean White, Steven Henderson, Lauren Wilcox, Mengü Sükan, Nicolas Dedual, and Carmine Elvezio. I am especially grateful to Steven Henderson for his camera calibration software and refurbishing and modelling the aircraft engine combustion chamber and Rolls-Royce Dart 510 engine. I also enjoyed brain storming research ideas and chatting with the lab visitors from various foreign institutes and enterprises: Anette von Kapri, Christian Holz, Yoshitaka Tokusho, Yang Zhang, Dominikus Baur, Sangchul Ahn, and Claudia Lauschke.

I would also like to express my appreciation to Robert Wang for letting us use the 3Gear Systems 3D hand gesture tracking SDK from its early stage of the development.

I thank my project students for developing great tools for the Goblin XNA framework and implementing core functionalities for my research: Levi Lister, Mike Sorvillo, Wei Teng, Nikhil Ramesh, Fan Lin, Janessa Det, Ketaki Kulkarni, JeanHeyd Meneide, Lixing Dong, Seongwoon Ko, Pratheba Selvaraju, Colin MacAllister, and Semih Energin.

Last but not least, I appreciate the support of my wife Akiko while working on my dissertation, and her endurance to take care of our daughter Sarah without much of my help during my dissertation and defense preparation.

This material is based upon work supported in part by the National Science Foundation under Grants IIS-01-21239, IIS-0905569, and IIS-0905417, the Games for Learning Institute, and generous gifts and loans from Microsoft Research, Vuzix, Canon U.S.A Inc., and Leonar3Do.

Chapter 1. Introduction

Figure 1.1. Examples of activities in which multiple people interact. (a) Children playing with Lego blocks (http://picklebums.com/2009/09/07/toys-we-love-lego/). (b) Plant workers assembling a machine (http://karenmarcelo.org/srlyard/2007/09/robodock-assembling-machines-and-prop.html).

Multi-user interaction involves any form of interaction among multiple people, including direct or indirect physical interaction, and verbal or gestural communication. As we live in a community, it is natural that we interact with others across a variety of scenarios. These can range from children's play, such as Lego model construction, to professional activities, such as collaborative machine assembly, as depicted in Figure 1.1. As technology advances, multi-user interaction can be conducted ever more efficiently and over greater distances.

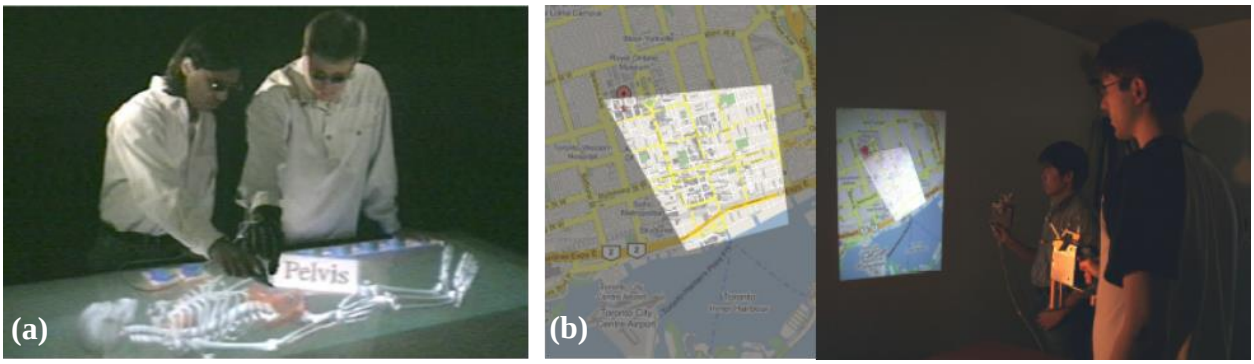

Figure 1.2. Examples of computer-assisted multi-user interactions. (a) Two users try to point at a specific bone structure of a virtual skeleton that appears in 3D stereo through synchronized pairs of special shutter glasses [Agrawala et al., 1997]. (b) Two users manipulate different parts of a projected 2D map through their hand-held projectors. One user controls a large map with fewer details (only main streets are labeled) while the other user controls the focus region (the brighter region) in which more details are displayed (smaller streets are also labeled in addition to the main streets) [Cao et al., 2007].

People have shown strong interest in improving and streamlining cooperative tasks, as most real-world human interactions are cooperative in nature. Recently, collaboration has been supported through technologies such as multi-touch surfaces [Agrawala et al., 1997; Izadi et al., 2003], projectors [Cao et al., 2007; Rekimoto and Saitoh, 1999], and virtual reality (VR) [Brown and Bell, 2004; Hindmarsh et al., 2000], as shown in Figure 1.2. Even though these technologies provide natural multi-user interaction, the focus of the interaction is on the display or projection surface. In the case of VR, the interacting users are completely hidden from each other or replaced with virtual avatars. Research has shown the importance of face-to-face interaction [Agrawala et al., 1997; Ishii et al., 1994; Kiyokawa et al., 1998], which provides enhanced awareness of other collaborators' gaze direction and gesture. Augmented reality (AR) [Feiner, 2002], in which geometrically registered virtual graphics and sound are interactively overlaid on the real world, has the potential to support natural face-to-face multi-user interaction. Studies have shown that users prefer collaboration in an AR environment to an immersive virtual environment, and can perform better on certain tasks because they can perceive each other's non-

verbal cues [Billinghurst et al., 1998]. AR provides seamless interaction between real and virtual environments, the presence of spatial cues such as gestures or gazes, and the support of tangible interfaces.

Figure 1.3. Examples of collaborative AR. (a) Two users collaborating on urban planning [Broll et al., 2004]. (b) One user examining a 3D virtual model of a ceramic vessel while the other user is looking at a 3D terrain model of an archaeological dig site where the vessel was excavated [Benko et al., 2004].

Collaborative AR is being explored in many fields, including entertainment [Barakonyi et al., 2005; Mulloni et al., 2008; Oda et al., 2008; Wagner et al., 2005], urban design [Broll et al., 2004; Ismail and Sunar, 2009; Wang et al., 2007], maintenance and repair [Bottecchia et al., 2010], assembly [Kirk and Fraser, 2005; Kiyokawa et al., 2000; Kurata et al., 2004; Wiedenmaier et al., 2003], surgery [Fuchs et al., 1998; Nicolau et al., 2005], site exploration [Benko et al., 2004; Hua et al., 2004], and education [Orozco et al., 2006; Pemberton and Winter, 2009]. One of the most significant challenges for collaborative AR is assuring the mutual understanding of speech and action among users [Azuma et al., 2001]. In face-to-face collaboration, people try to communicate their intentions to their collaborators in the clearest way possible using speech, gesture, gaze direction and non-verbal cues [Billinghurst and Kato, 2002]. However, it is difficult to guarantee that each user clearly understands what other users point at or refer to when each user has different overlaid graphics in AR.

Furthermore, in a co-located multi-user scenario, it is not unusual that one user's action may interfere with another user's activities. Solutions to such conflicts are explored in many different scenarios. In environments using table-top displays [Izadi et al., 2003; Morris et al., 2004] and projectors [Cao et al., 2007], the conflict resolution mechanism mainly focuses on limiting access rights and prohibiting manipulation in certain private areas defined by other users. In VR [Razzaque et al., 2001], collision avoidance with static physical obstacles such as walls is stressed using path-redirection techniques. Similarly in AR, it is essential to provide assistance to avoid physical collisions and interference for various reasons including security and performance.

This dissertation explores a novel class of AR interaction techniques for assisting multiple users in performing referencing tasks more accurately in shared environments and more efficiently in remote environments, while also preventing users from physically interfering with each other. Our exploration includes the development and evaluation of novel interaction techniques, as well as the implementation of an underlying software framework that accelerates the development of multi-user AR applications.

1.1 Research Questions and Dissertation Goals

Our thesis addresses the following research questions:

1. ***How can co-located users accurately communicate items of interest to each other in AR?*** For multi-user interaction, especially collaboration, to be successful, users must have the ability to generate and interpret effective reference cues [Azuma et al., 2001]. In the everyday environment, we can mostly rely on many of the nonverbal cues, such as gesturing and eye gaze. However, these nonverbal cues can be obscured in AR, since different participants may have different views and current head-worn displays prevent other users from seeing one's gaze direction. It can be especially challenging for a person to point at a specific part of a physical object at a

distance and to have another person understand the referenced part correctly. A referencing technique seeks to support users in coming to a mutual understanding of an object indicated by one of the users. The goal includes allowing a user to specify part or all of a physical object of interest in the shared scene, and ensuring that the other users understand which object is being referenced. What referencing techniques would provide better accuracy in manipulating physical objects at a distance for co-located users in a previously unmodeled environment?

2. ***How can remote users efficiently communicate items of interest and intents to each other in AR?*** It is even more challenging to refer to an object or an intent in the scene of a remote user and have the referenced object or intent understood by the remote user, since they cannot rely on direct gestures or eye gaze. This type of communication is quite common and critical for *remote task assistance,* in which two users are in physically separate environments. For example, a remote expert may need to explain to a local user at a different location the task procedures involving objects in the local user's environment, so that the local user can perform the task procedures. Thus, the efficiency and effectiveness of remote task assistance heavily depends on the referencing techniques used. How can we design a set of referencing techniques that would significantly improve remote assistance tasks in comparison with existing remote referencing techniques?

3. ***How can co-located users avoid physically interfering with each other in AR?*** In a situation where multiple users interact in a shared physical space, such as collaborative mechanical repair task, technicians may interfere with each other when they are examining and repairing parts that are physically near to each other. This type of undesired physical interference or interruption can reduce performance and can even cause serious injuries, not only in collaborative

domains, but also in competitive domains such as multi-player gaming. How can we design a system that could lessen the chance of physical interference among users?

1.2 Contributions

Our thesis is that our new techniques can provide more accurate referencing for co-located users and efficient referencing for remote users, and lessen physical interference among co-located users. This dissertation makes the following three contributions to the design of AR systems for supporting multi-user interaction with emphasis on referencing techniques and interference avoidance:

1. *Design, implementation, and evaluation of a referencing technique for physical objects in co-located multi-user AR* [Oda and Feiner, 2012]. We developed a referencing technique, GARDEN (Gesturing in an Augmented Reality Depth-mapped ENvironment), intended for use in an otherwise unmodeled physical environment in which objects in the environment, and the hand of the user performing a selection, are interactively observed by RGB and depth cameras. GARDEN was designed to be used in situations when it is difficult or inappropriate for the users to walk to the target object and point at it directly. We conducted a within-subject user study to compare our technique against existing AR referencing techniques, as well as the use of a physical laser pointer. The study showed that GARDEN performed significantly more accurately than all the comparison techniques in situations in which the participating users have sufficiently different views of the scene, and significantly more accurately than one of the comparison techniques in situations in which the participating users have similar perspectives.

2. *Design, implementation, and evaluation of a referencing technique for supporting remote task assistance in AR* [Oda et al., 2015]. We developed a 3D interaction and visualization approach, DEMO3D, in which a remote expert, using *virtual replicas* of physical objects, guides a

local user through demonstration to perform a 6DOF alignment task (a core component of many remote-guidance scenarios). Our approach allows the remote expert to switch seamlessly between a mirrored virtual reality (VR) representation of a local user's tracked environment, and shared live or frozen AR views from the perspective of the local user without affecting what the local user sees. We compared DEMO3D with another 3D approach (POINT3D) in which the remote expert identifies corresponding pairs of points to align on a pair of objects and a 2D sketch-based approach running on a tablet (SKETCH2D), similar to sketch-based systems used in previous work. A formal user study showed that DEMO3D performed faster than the other approaches. DEMO3D was preferred over SKETCH2D by local users, and remote experts generally felt DEMO3D was faster and better than SKETCH2D.

3. *Design, implementation and evaluation of a physical-interference–avoidance technique in co-located multi-user AR* [Oda and Feiner, 2009]. We designed and implemented *Redirected Motion*, an effective, yet nondistracting, interference avoidance technique, which transforms the 3D space in which a user moves their hand-held display, to direct their display away from other displays. We conducted a within-subject, formal user study to evaluate the effectiveness and distraction level of Redirected Motion compared to other techniques for avoiding physical interference. The study is based on an instrumented, two-player, first-person-shooter, AR game, in which each player interacts by using a 6DOF-tracked hand-held ultra-mobile computer. Comparison conditions include an unmanipulated control condition and three other software techniques for avoiding interference between the devices: dimming the display, playing disturbing sounds, and disabling interaction capabilities. Subjective evaluation indicates that Redirected Motion was unnoticeable, and quantitative analysis shows that the mean distance between users' devices during Redirected Motion was significantly larger than for the comparison conditions.

All the above techniques and test applications were implemented using Goblin XNA [Oda and Feiner, 2012] (Appendix A), an open-source AR framework we have developed. This framework incorporates a 3D scene graph, physics simulation, networking, a 2D UI system, and various tracking technologies, including marker-based 6DOF tracking, to support rapid development of multi-user AR applications.

We summarize these contributions further in the following sections.

1.2.1 Design, implementation, and evaluation of a referencing technique for physical objects in co-located multi-user AR

In Chapter 3, we present a 3D referencing technique, GARDEN (Gesturing in an Augmented Reality Depth-mapped ENvironment) [Oda and Feiner, 2012], that assists co-located users selecting and interpreting physical objects in previously unmodeled environment. Objects in the environment, as well as the hand of the user performing a selection, are interactively observed by RGB and depth cameras. GARDEN leverages the following observations from existing AR referencing techniques:

1. It is difficult for a user to perform precise selection of an object at a distance using a direct line-of-sight-based approach (e.g., pointing with a ray or 3D cursor) due to the often large effect on the point of intersection of a ray or 3D cursor with the scene that results from even a small angular change in the pointing direction [Chastine et al., 2007; Liang and Green, 1994; Poupyrev et al., 1996].
2. Pointing at an object using a line-of-sight-based approach can be misleading to other users because of the differences in the projections and occlusion relationships seen by the users [Chastine et al., 2007].
3. Outlining a region before performing a final selection can improve the accuracy with which other users can interpret which object was selected [Salzmann et al., 2009].

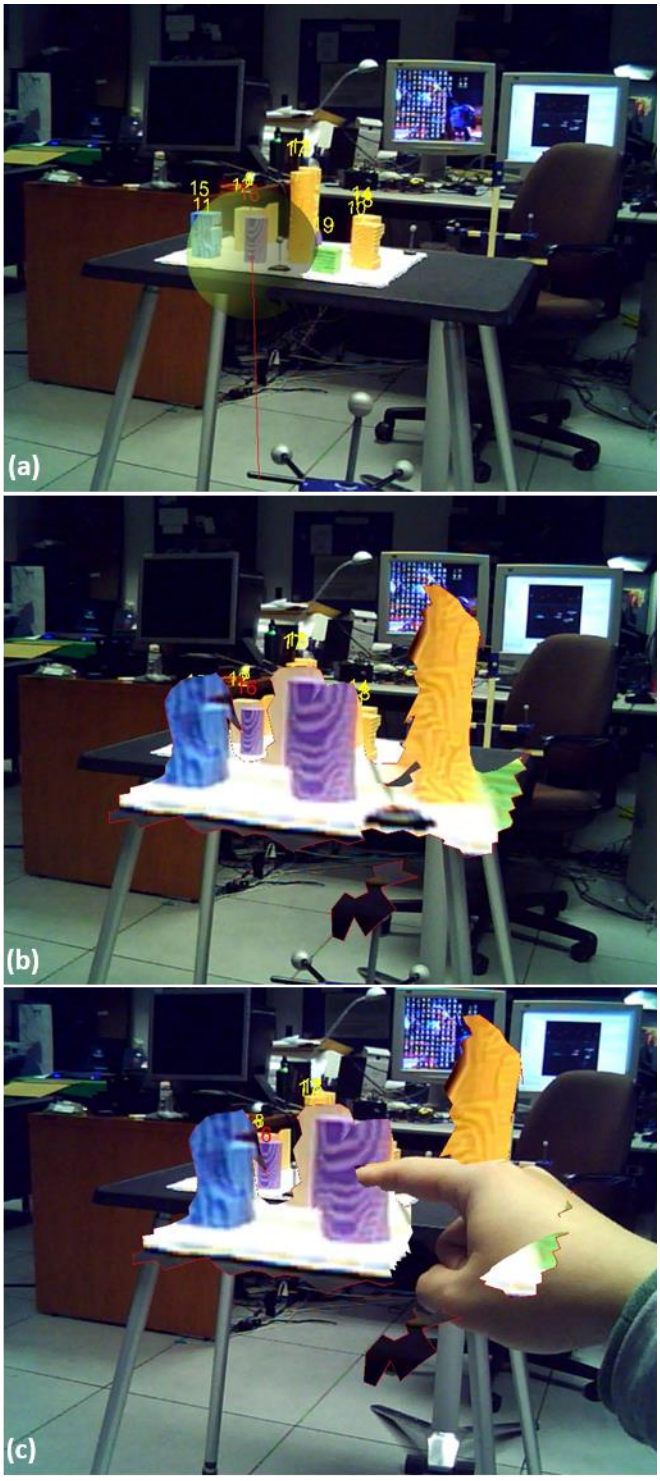

Figure 1.4. The indicator's views through a stereo video–see-through head-worn display. (a) Indicator specifying a spherical volume with a tracked hand-held device. (b) A virtual representation of the physical region constructed from depth and RGB data is brought closer to the indicator. (c) Indicator pointing directly with her finger at the representation to indicate the physical object to which she intends to refer (in this case, the center purple cylinder).

Initially, in a situation in which two users (a selecting user and an interpreting user) are involved, the selecting user (the *indicator*) specifies an approximate region that contains the target (Figure 1.4a). Once the indicator confirms the approximate region, a virtual representation of the physical region is reconstructed from data obtained from the depth camera and RGB camera. This representation is presented to the indicator at a closer distance than the actual object to make it easier to see and manipulate (Figure 1.4b). Finally, the indicator points at the close-up virtual representation with her hand to indicate the object to which she is referring (Figure 1.4c).

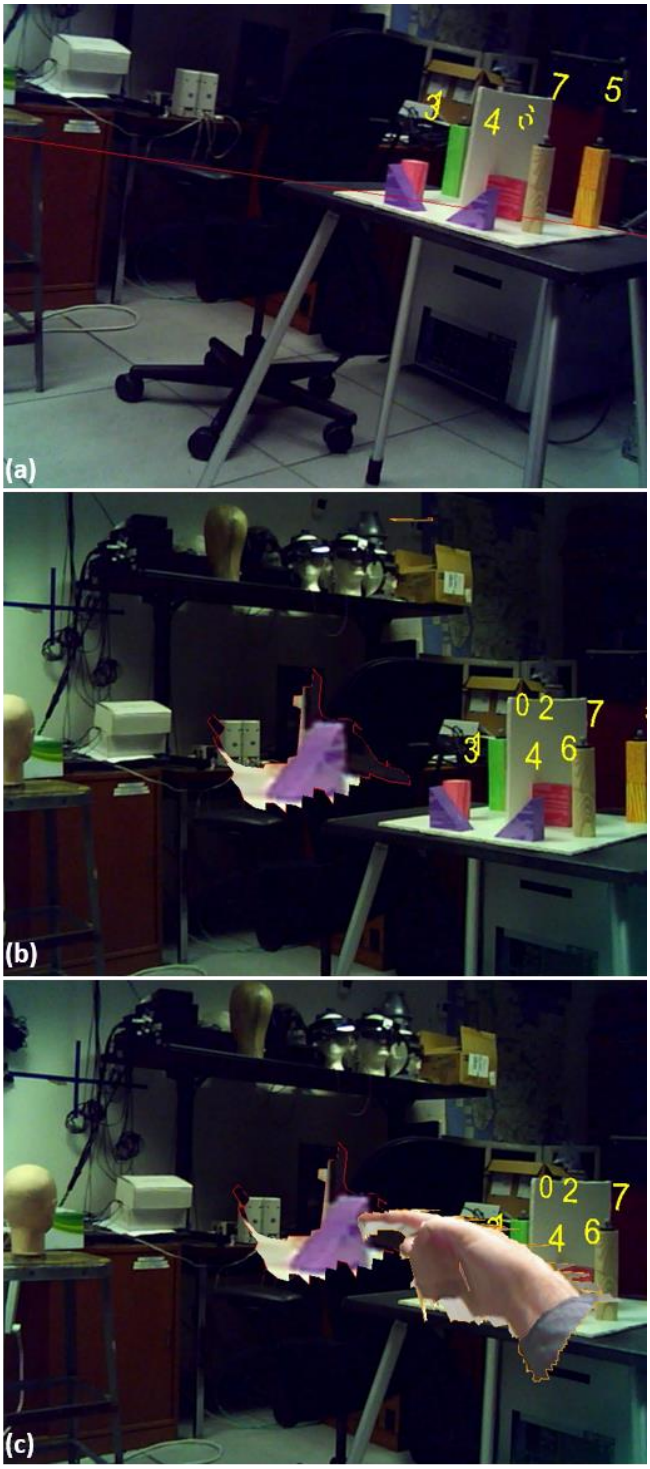

Figure 1.5. The recipient's views through a stereo video–see-through head-worn display. (a) Indicator specifying an approximate region. (Here, the indicator is selecting a different region from that in Figure 1.4.) Only the infinite ray is visible to the recipient, but not the spherical volume. (b) A close-up representation of the spherical volume selected by the indicator. (c) Recipient seeing the selected volume and the reconstructed virtual representation of the indicator's hand.

While the indicator is specifying an approximate region, the interpreting user (the *recipient*) sees an infinite ray cast to the physical environment from the indicator's tracked hand-held device (Figure 1.5a). Once the indicator finishes specifying the approximate region, the recipient sees a copy of the virtual representation of the enclosed physical region placed at a predetermined distance near the recipient and oriented along the recipient's view direction, similar to what was done for the indicator (Figure 1.5b). The recipient then sees a virtual representation of the indicator's hand pointing within the representation (Figure 1.5c). An optional user-controlled animation that interpolates a copy of the virtual representation and the indicator's hand seen by the recipient to the actual position and orientation in the physical environment is provided to the recipient to aid in understanding the mapping between the virtual representation and the real world.

We evaluated GARDEN against existing AR referencing techniques, as well as the use of a physical laser pointer (*Laser Pointer*). The comparison techniques included video sharing (*Video Share*) in which the indicator's view is displayed to the recipient while the indicator is selecting a target using a physical laser pointer, and a 3D virtual arrow (*Virtual Arrow*) with user-adjustable length. A within-subject, single-session user study examined 11 pairs of participants (22 participants) for a total of 64 referencing tasks for each pair. Each participant performed half the tasks as indicator and half as recipient. The referencing task involved an indicator selecting a pseudo-randomly assigned physical object from a predetermined set of objects using a specified technique and a recipient specifying an identifier associated with the physical object they think is being specified. Each selectable physical object has an overlaid unique numeric identifier floating above it in each participant's view.

A quantitative analysis of the accuracy of the techniques supported the following hypotheses we formulated prior to our study:

- H1: GARDEN will be more accurate than the comparison techniques when referencing objects for which the indicator and recipient do not share a similar perspective.
- H2: GARDEN will be more accurate than the Virtual Arrow when referencing objects for which the indicator and the recipient share a similar perspective.

We hypothesized that GARDEN would be more accurate than Virtual Arrow for both situations, because a line-of-sight-based approach may look perfectly intersected with the referenced object from the perspective of the indicator, but confusing or wrong from the perspective of the recipient. This occurs because of differences in the projections and occlusion relationships, even when the perspectives are close. In contrast, GARDEN provides the recipient with a close-up view of the selected region and the indicator's hand so that projection differences or occlusion should not be an issue.

We also expected that GARDEN would perform significantly more accurately than Laser Pointer and Video Share for situations in which the users have significantly different views of the scene. In these cases, it is not possible for the recipient to directly view the laser dot cast on the referenced object in the Laser Pointer technique, and Video Share requires the recipient to map the object designated with the laser in the perspective of the indicator's view to a corresponding object in her own perspective. In contrast, GARDEN provides an animation that aids the recipient in mapping between the referenced object in the close-up view and the corresponding physical object in the environment seen from her own perspective.

In an effort to evaluate the accuracy of the techniques, we treated the correctness of the recipient's answer as binary data and counted them for each technique. We used $\alpha = 0.0167$ (0.05/3) after Bonferroni correction as our criterion for statistical significance, since all tests involved three pairwise comparisons. A McNemar's chi-squared test revealed that GARDEN was significantly more accurate than the three comparison techniques (Laser Pointer: ($\chi^2_{(1,N=57)} = 20.024$, $p < 0.001$), Virtual Arrow ($\chi^2_{(1,N=55)} = 21.441$, $p < 0.001$), and Video Share ($\chi^2_{(1,N=57)} = 6.323$, $p = 0.0119$)) when referencing physical objects for which the indicator and recipient do not share a similar perspective. This result validated hypothesis H1. A similar chi-squared test showed that GARDEN was significantly more accurate than the Virtual Arrow ($\chi^2_{(1,N=65)} = 24.750$, $p < 0.001$), but found no significant difference between GARDEN and Laser Pointer or Video Share when referencing physical objects for which the indicator and the recipient share a similar perspective. This result validated hypothesis H2.

In addition to the correctness of the recipient's answer, we recorded the occasions in which the recipient could not even speculate as to which object was referred. We requested the recipients prior to the study to enter a specific answer if they thought it was impossible to identify the referred object. We also treated these occasions as binary data and counted them for each technique. A McNemar's chi-squared test indicated that GARDEN had significantly less occasions than the three comparison techniques (Laser Pointer: ($N = 122$, $p < 0.001$), Virtual Arrow ($N = 121$, $p < 0.001$), and Video Share ($N = 122$, $p < 0.0167$)).

1.2.2 Design, implementation, and evaluation of a referencing technique for supporting remote task assistance in AR

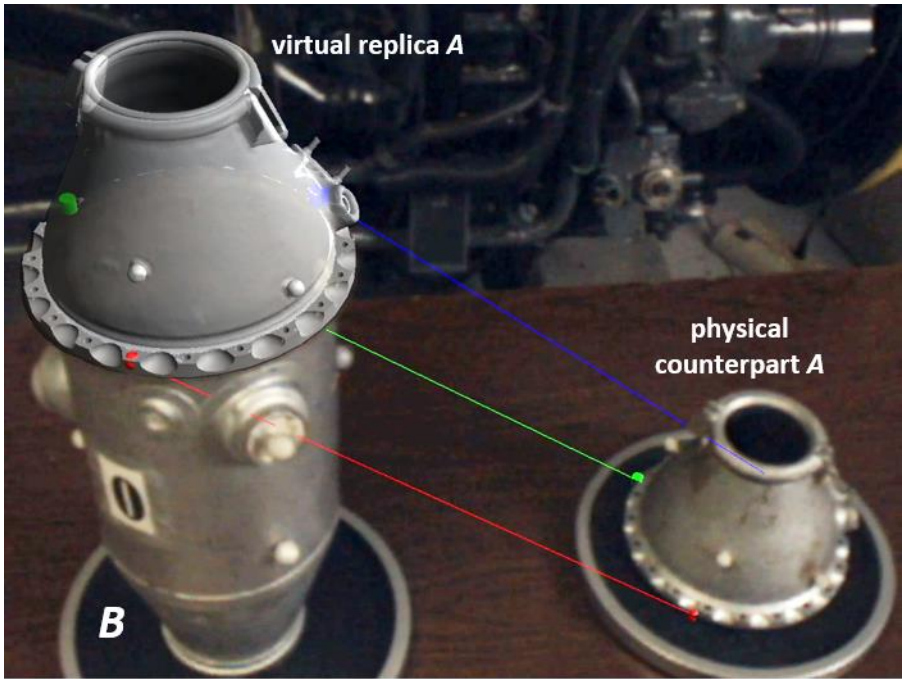

Figure 1.6. DEMO3D interaction technique allows a remote expert to guide a local user to place the top of an aircraft engine combustion chamber relative to its bottom by interacting with a *virtual replica* of the top. As seen by the expert in AR, looking through the local user's cameras, the expert has placed the transparent virtual replica on the chamber bottom. Pairs of metaobject annotations on the virtual replica and physical chamber top are linked with color-coded rubberband lines to show the correspondence between the physical top and its virtual replica, and are also seen by the local user.

In Chapter 4, we extend the idea of using virtual representation of physical objects to remote assistance scenarios. We address situations in which a remote expert advises a local user, both wearing head-worn displays. We present a 3D interaction and visualization approach, DEMO3D (Figure 1.6) using *virtual replicas* [Oda et al., 2015] for 6DOF alignment tasks between two physical objects, *A* and *B*. Our approach is based on the ways an expert guides a novice when co-located, by demonstrating appropriate actions. In our approach, the remote expert can create, annotate, and manipulate virtual replicas of physical counterparts in the local user's environment.

We assume that the physical objects with which the local user interacts have been modeled and are tracked in 6DOF. We also assume that the remote expert's environment contains a *virtual proxy* (Figure 1.7) for each relevant physical object in the local user's environment and that the position and orientation of each virtual proxy is determined by the corresponding position and orientation of the physical object in the local user's environment. The expert can create a virtual replica of a physical object by grabbing its virtual proxy. The virtual replica can then be manipulated in 6DOF, remaining where the grab is released, and grabbed again to manipulate it further. The virtual replicas are also displayed to the local user in context of their *physical counterparts* (Figure 1.6).

In DEMO3D, the expert uses a virtual replica of object *A* to directly demonstrate how to move *A* to its destination 6DOF pose relative to object *B*, as shown in Figure 1.6. However, since there is no force feedback when placing a virtual replica of *A* relative to *B*, the two can interpenetrate or not fit properly. To address this, we employ a constraint-based approach, which we explain in detail in Chapter 4. We add a set of landmark metaobjects (Figure 1.6) on the virtual replica, which are duplicated on its physical counterpart, with a connecting line between each metaobject and its duplicate in the local user's view. The local user can use these metaobjects as cues to match the 6DOF pose of the physical counterpart with the virtual replica placed by the expert.

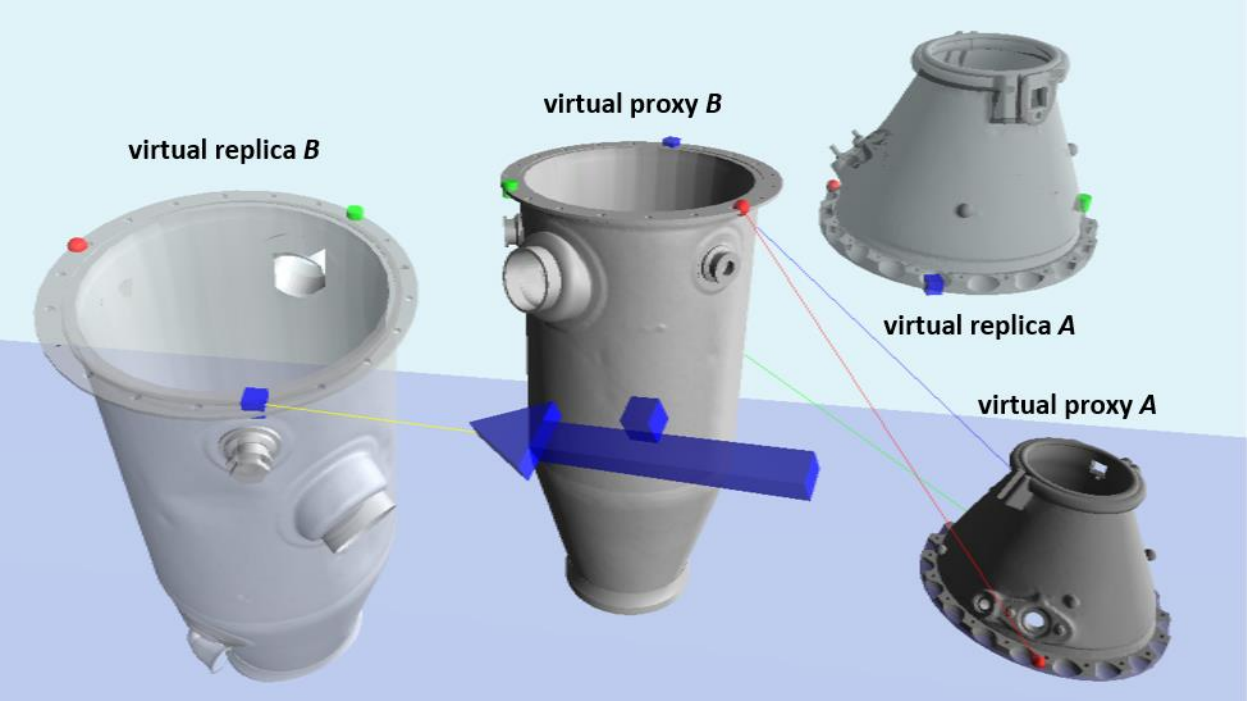

Figure 1.7. POINT3D, as seen by the expert in VR: The expert places 3D annotations (e.g., blue cube) on the transparent virtual replica, specifying contact points. The same annotations appear on the opaque *virtual proxy* corresponding to the virtual replica. In Point3D, once the expert specifies corresponding contact points on both objects (*A* and *B*), a color-coded rubberband line connecting the points appears between the proxies.

We compared DEMO3D with two other approaches: POINT3D and SKETCH2D. In POINT3D (Figure 1.7), which also uses virtual replicas, the expert manipulates a tracked pointing device that controls a ray whose intersection with an object defines a contact point as shown in Figure 1.7. After optionally creating one or more virtual replicas, the expert can point to a location on either a virtual replica or its virtual proxy to place annotations anywhere on the corresponding physical object. The expert can freely manipulate a virtual replica to find a suitable pose from which to place an annotation. Annotations appear on both the virtual proxy and the virtual replica in the expert's view and on the physical object in the local user's view. Once annotations for corresponding points have been placed on both *A* and *B*, a "rubberband" line will

appear between corresponding annotations on both objects to help the local user identify which annotated points should be aligned as shown in Figure 1.7.

In both DEMO3D and POINT3D, the expert can seamlessly alternate between two different perspectives on the local user's environment without affecting what the local user sees: a VR representation (Figure 1.7) using virtual models of 6DOF-tracked physical objects and an AR representation (Figure 1.6), seen through the local user's cameras (live or frozen).

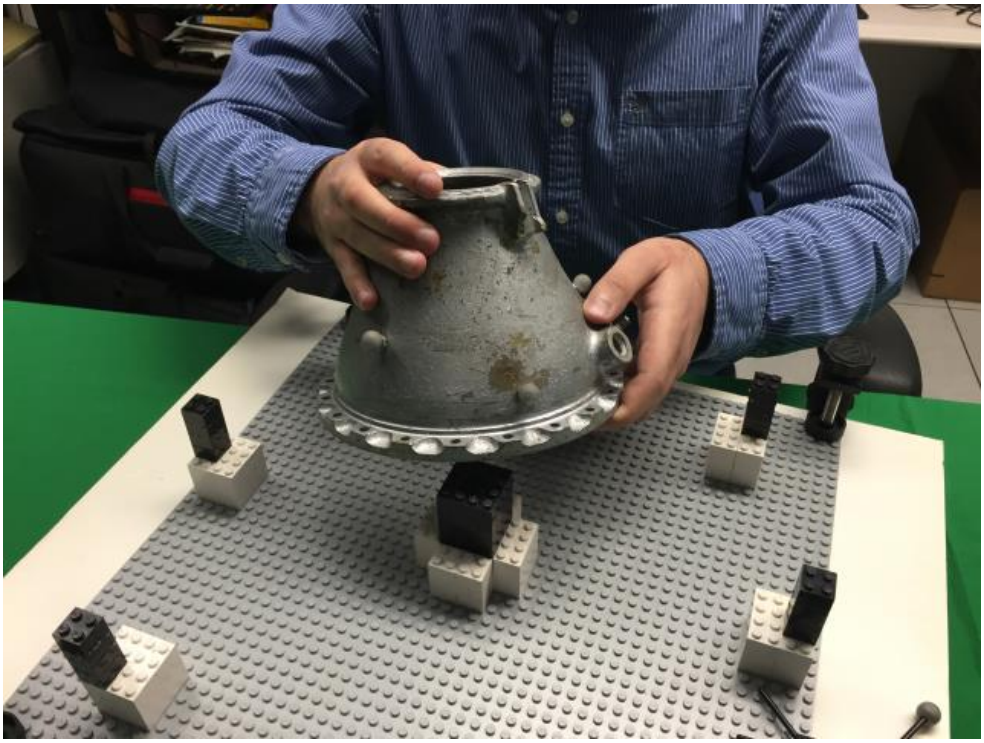

Figure 1.8. A local user placing a chamber top on a Lego fixture as instructed by the remote expert.

As a second comparison approach, we developed a 2D sketch-based system on a tablet (SKETCH2D) similar to those used in previous work. We compared the three approaches on speed of performance, ease of interaction by the remote expert, ease of interpretation by the local user, and preference. We required that the accuracy with which the local user performed each trial be within a small range of distance and orientation from correct pose for the trial to end; therefore, we compared only time, not accuracy. A within-subject, single-session user study ex-

amined 11 pairs (22 participants) for a total of 198 trials, one playing the remote expert role and the other playing the local user role. We also collected data with one of the researchers as a highly-trained expert for same number of trials. The 6DOF alignment task involved a remote expert instructing a local user placing a chamber top on a Lego fixture at one of its predefined configuration as shown in Figure 1.8.

Based on an analysis of the tasks and the pilot studies, we hypothesized the following:

- H1: DEMO3D should be faster than POINT3D. We expect POINT3D to take longer because it requires the remote expert to make six annotations, as opposed to DEMO3D, which only requires a single motion for demonstration. DEMO3D also allows for a quicker/less precise alignment because of its embedded 5-DOF constraint.
- H2: POINT3D should be faster than SKETCH2D. Similar to POINT3D, SKETCH2D also requires six annotations per alignment, but it additionally burdens the expert with 3D virtual camera navigation to be able to sketch on certain parts of objects. With POINT3D, the expert should be able to insert annotations relatively quickly using bimanual pointing. Both POINT3D and DEMO3D should require less interpretation time by the local user, compared to SKETCH2D, because of the assistance provided by the virtual rubberband lines that connect corresponding metaobjects.
- H3: Both remote experts and local users should prefer DEMO3D. We expect this because DEMO3D should be quicker and less cognitively challenging for the same reasons provided for H1 and H2.

We evaluated the results for both experts from the study participants (Novice Experts) and one of the researchers (Trained Expert) for significance using a Bonferroni-corrected α of 0.0167 (0.05/3). A Kenward-Roger corrected F-test showed significant main effects on completion time for both Novice Experts ($F_{(2, 173.106)} = 79.269$, $p < 0.0001$) and Trained Expert ($F_{(2,173.91)} = 153.82$, $p < 0.0001$). A pairwise least-squares means comparison revealed that the Trained Expert was significantly faster using DEMO3D than POINT3D ($t_{(178.44)} = -7.199$, $p < 0.0001$), validating H1. The Trained Expert was also significantly faster using POINT3D than SKETCH2D ($t_{(177.75)} = -10.23$, $p < 0.0001$), supporting H2. For the trials with Novice Experts, DEMO3D was significantly faster than POINT3D ($t_{(177.13)} = -11.944$, $p < 0.0001$) and SKETCH2D ($t_{(177.16)} = -9.551$, $p < 0.0001$), validating H1. However, completion times between POINT3D and SKETCH2D did not differ significantly ($t_{(177.12)} = 2.336$, $p = 0.0618$), failing to support H2.

A majority of participants (7 Novice Experts and 8 local users, 64% and 73% respectively) ranked DEMO3D as their most preferred technique, supporting H3. A Friedman test indicated a statistically significant differential preference between techniques for local users ($\chi^2_{(2)} = 9.46$, $p < 0.01$), and a Nemenyi's procedure showed that rankings of DEMO3D were significantly more favorable than those for SKETCH2D ($p < 0.01$). Novice Experts on average ranked DEMO3D first, SKETCH2D second, and POINT3D last, but the resulting Friedman test did not show significant difference, supporting, but not validating, H3 for Novice Experts.

The remote collaboration research was joint with colleagues Carmine Elvezio and Mengü Sükan. I conceived of the idea of using virtual replicas to demonstrate relative poses (DEMO3D) and the constraint-based approach and I proposed the user study setting in which the user places a chamber top relative to a base in one of a set of poses. I performed much of the implementation using the Goblin XNA infrastructure I developed, and I authored the majority of the paper [Oda

et al., 2015]. My colleagues suggested POINT3D as an alternative to DEMO3D that also uses virtual replicas, developed the final configuration of the base fixture used in the study, assisted with implementation, conducted and analyzed the study, and assisted with writing the paper.

1.2.3 Design, implementation and evaluation of a physical-interference–avoidance technique in co-located multi-user AR

Figure 1.9. Redirected Motion in sequence from top to bottom. As the green player (on the right) moves toward the red player (on the left), the virtual location of the green player is shifted along the green player's direction of movement, while the (stationary) red player's virtual location is not affected.

There are many situations in which multiple users working together in close quarters might physically interfere with each other. For example, when one maintenance technician is concentrating on task, he might not be paying too much attention to his surroundings including his fellow technicians. Therefore, it may be possible for nearby technicians to accidentally interfere with each other. Physical interference in a maintenance and repair task not only affects the user's performance, but can also cause serious injuries. Similarly, in a multiplayer game in which users are close to each other, physical interference can occur frequently if players focus on their game screens, without paying attention to their surrounding players.

In an effort to lessen this sort of unwanted and unintentional physical interference, we have developed and tested *Redirected Motion* [Oda and Feiner, 2009], an interference avoidance technique inspired by earlier work on redirected walking in virtual reality [Razzaque et al., 2001]. We were primarily interested in avoiding collisions between tracked devices, and, as a side effect, in avoiding collisions between users, who might otherwise not be tracked.

As shown in Figure 1.9, which was captured in a multi-player game testbed that we developed, Redirected Motion transforms the virtual location of a user (here, that of the user's hand-held device) as they approach another user to keep the users from colliding. This technique was designed assuming:

1. Users are most interested in the location toward which they are moving.
2. Users are aware of each other's approximate physical location.
3. Users are not intentionally trying to physically interfere with each other.

Using Interrante and colleagues' [Interrante et al., 2007] idea of scaling distance traveled by amplifying the user's velocity only in the direction in which they are walking, we apply an additional translation to the virtual location of a user in the direction in which they are moving. The translation is applied to a user (user *A*) when their distance to another user (user *B*) is under a specified distance and user *A* is moving toward user *B*. When the user is outside the distance threshold, the applied translation is gradually eliminated as the user moves.

Figure 1.10. Player's view of our test application. Virtual balls are fired at virtual dominoes by tapping on the screen of a hand-held computer.

We evaluated our technique against commonly used interference-avoidance techniques including an unmanipulated control condition (*None*) and three other software techniques for avoiding interference: dimming the display (*Screen Dimming*), playing disturbing sounds (*Sound Beeping*), and disabling interaction capabilities (*Action Disable*). The techniques were assessed under two criteria: effectiveness and distractibility. Effectiveness referred to how well the technique kept the two participants away from each other, and distractibility referred to how distracting the technique was. We developed a two-player game for our study, as shown in Figure 1.10,

in which each player attempts to knock all of the other player's virtual dominos off a real table by shooting virtual balls with a hand-held ultra-mobile personal computer (UMPC) through which they view the environment. The test application was designed to encourage players to move around frequently so that the effect of each technique becomes more prominent. A within-subject, single-session user study examined 9 pairs of participants (18 participants) for a total of 27 game rounds after removing outliers and mitigating learning curve bias. Players were allowed to move anywhere around the table and shoot from any angle they desire, as long as at least one marker in the optical-tracking marker array on the table is visible from and can be tracked by the device's embedded camera.

Prior to our study, we formulated several hypotheses based on preliminary pilot studies conducted with our lab members.

- H1: The control condition (None) will be least effective and least distracting.
- H2: Sound Beeping will be less effective, but less distracting than Screen Dimming or Action Disable, since the player can simply ignore the beep.
- H3: Screen Dimming and Action Disable will be most effective, but most distracting, compared to other techniques, since visibility or interaction capability are impaired.
- H4: Redirected Motion will be as effective as Screen Dimming or Action Disable and much more effective than Sound Beeping or None.
- H5: Redirected Motion will be the least distracting compared to Screen Dimming, Action Disable, or Sound Beeping, since the players do not perceive the shifting effect.

Analysis from a post-hoc questionnaire yielded several qualitative findings. First, players perceived both None and Redirected Motion to be significantly less effective and distracting compared to other three techniques, which invalidates H1. Second, Sound Beeping was assessed to be significantly less effective, but less distracting compared to Screen Dimming or Action Disable which qualitatively validates H2. Third, both Screen Dimming and Action Disable were evaluated as significantly more effective, but more distracting, than the other conditions, which qualitatively validates H3. Finally, participants thought that None and Redirected Motion were not distracting. This validates H5. We addressed hypothesis H4 in the quantitative analysis, since participants were not aware of the effect.

Analysis of player performance under different conditions revealed technique had a significant effect on game duration ($F_{(4,23)} = 5.382$, $p < 0.01$) and on distance between the participants over time ($F_{(4,23)} = 5.504$, $p < 0.01$). A paired-sample *t* test between Redirected Motion and Action Disable ($t_{(26)} = 3.611$, $p < 0.01$) shows that rounds played with Redirected Motion take significantly less time to complete compared to ones played with Action Disable, but there was no significant difference between Redirected Motion and other techniques. We believe that this occurs because when action is disabled, dominos could not be knocked off the board. This increase in game duration is consistent with the high level of perceived distractibility for Action Disable shown in the Likert-scale results and rankings.

Similar paired *t* tests between Redirected Motion and the other conditions showed that during Redirected Motion, participants maintained a significantly larger mean distance between them, especially comparing Redirected Motion and None, and Redirected Motion and Sound Beeping—None ($t_{(26)} = 3.946$, $p < 0.001$), Screen Dimming ($t_{(26)} = 3.414$, $p < 0.01$), Sound Beeping ($t_{(26)} = 4.462$, $p < 0.001$), Action Disable ($t_{(26)} = 2.854$, $p < 0.01$). This validates H4. There

was no significant difference among the comparison conditions for mean distance between users. We believe that Redirected Motion kept participants significantly farther away from each other compared to other conditions due to three reasons. First, the technique itself allowed the players to shoot at their targets without the necessity to get too close to another player. Second, we observed that some players appeared to try to use the other techniques strategically against their opponent to distract their play without knowing that the techniques only affected those who moved closer to the other player. Third, with the exception of Action Disable, the players could either ignore the beeping or plan a shooting at a safe distance and then move in briefly to execute it with a dimmed screen.

1.3 Structure of Dissertation

In the remainder of this dissertation, we first briefly review the history of computer-assisted multi-user interaction and previous work related to referencing techniques in Chapter 2. Next, Chapter 3 details the design, implementation, and evaluation of our 3D referencing technique for co-located environment. Following this, Chapter 4 discusses and assesses our referencing technique for remote environments. Chapter 5 then introduces our interference avoidance technique, *Redirected Motion*, and evaluates this technique against commonly used approaches. We also discuss the application of Redirected Motion to a remote assistance scenario discussed in Chapter 4 with multiple experts. Finally, Chapter 6 presents our conclusions and describes possible future work. In Appendix A, we describe our AR framework, *Goblin XNA*, which supports rapid development of multi-user AR applications, and which we use to implement the techniques described in Chapters 3–5. In Appendix B, we provide the study questionnaires given to the study participants for each of the user studies described in Chapters 3–5.

Chapter 2. Related Work

As briefly discussed in Chapter 1, researchers have expressed great interest in developing techniques and leveraging the latest technologies to assist multi-user interaction, especially in the form of collaboration. Their efforts span the fields of entertainment, urban design, site exploration, maintenance and repair, machinery assembly, education, surgery, and many more. In this chapter, we briefly summarize previous work on computer-assisted multi-user interaction and techniques that are an integral part of the referencing activity. In the subsequent chapters, we provide more in-depth prior literature directly related to each of our dissertation topics.

We first review the focuses and categories of computer-assisted multi-user interaction (Section 2.1). We then examine multi-user interaction designed for AR environments (Section 2.2). Finally, we survey a core set of techniques relevant to referencing techniques, which are mainly comprised of selection (Section 2.3) and visualization (Section 2.4) techniques.

2.1 Computer-Assisted Multi-User Interaction

Exploration for assisting multi-user interaction by means of computer systems has gained popularity since Greif and Cashman first coined the term *computer-supported cooperative work* (CSCW) in 1984 [Grudin, 1994]. Although the field is wide-ranging, we can see general focuses and categories in the existing literature.

2.1.1 Focuses

The research community in CSCW has mainly focused on *awareness* [Dourish and Bellotti, 1992], which signifies the understanding of the activities of others; *articulation work*

[Schmidt and Bannon, 1992], which indicates the distribution and integration of tasks among the group; and *appropriation* [Dourish, 2003], which denotes how people adopt and adapt technologies. This dissertation's concern for referencing effectiveness and interference avoidance aligns with the awareness domain. Dourish and Bellotti emphasize the importance of the awareness of individual and group activities for successful collaboration [Dourish and Bellotti, 1992]. Gutwin and colleagues introduced the concept of *workspace awareness* [Gutwin et al., 1996], an up-to-the minute knowledge about others' interaction, to support effective real-time remote interaction by leveraging the knowledge in face-to-face settings. Cooperative work depends on effective communication [Hiltz and Turoff, 1993; Olson and Lucas, 1982], and it is essential to establish a *grounding in communication* [Clark and Brennan, 1991], which comprises "mutual knowledge, mutual beliefs, and mutual assumptions" among the group. This requires the involved parties to coordinate both the content and process of their activities.

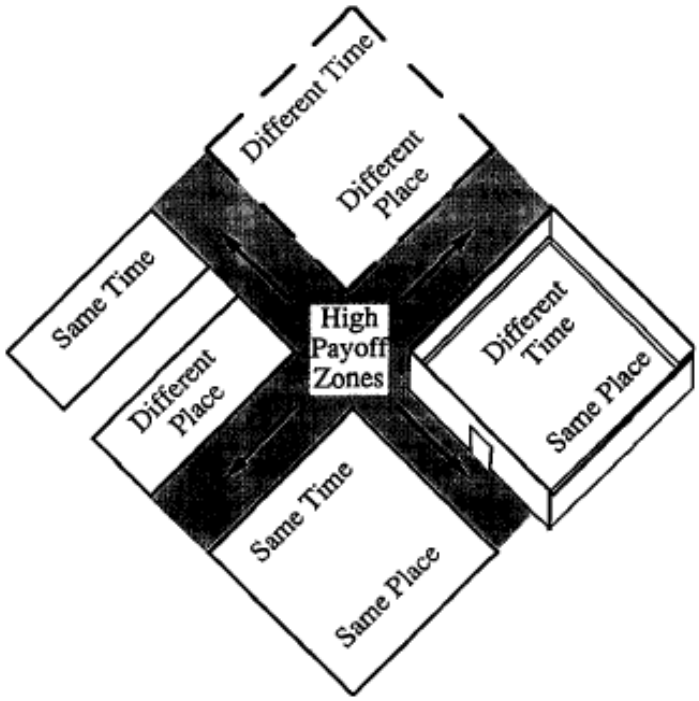

Figure 2.1. A two-by-two matrix that illustrates four different categories of CSCW systems (a.k.a. *groupware*) based on whether the collaboration happens at the same or different place and whether at the same or different time [Johansen, 1991].

2.1.2 Categories

The context of a CSCW system can be conceptually categorized into four groups [Johansen, 1991]. As shown in Figure 2.1, the classification is based on whether the collaboration is co-located or geographically distributed and whether the collaboration is synchronous or asynchronous. Co-located synchronous interaction confronts potential occlusion issues and physical interference [Billinghurst et al., 1998]. Remote synchronous interaction imposes a challenge because of absence of non-verbal cues such as gaze direction and gestures [Billinghurst and Kato, 2002]. Co-located or remote asynchronous interaction requires proper coordination of activities performed at different timings. Our dissertation focuses on addressing issues pertinent to synchronous multi-user interaction in both co-located and remote AR environments.

2.2 Multi-User Interaction in AR

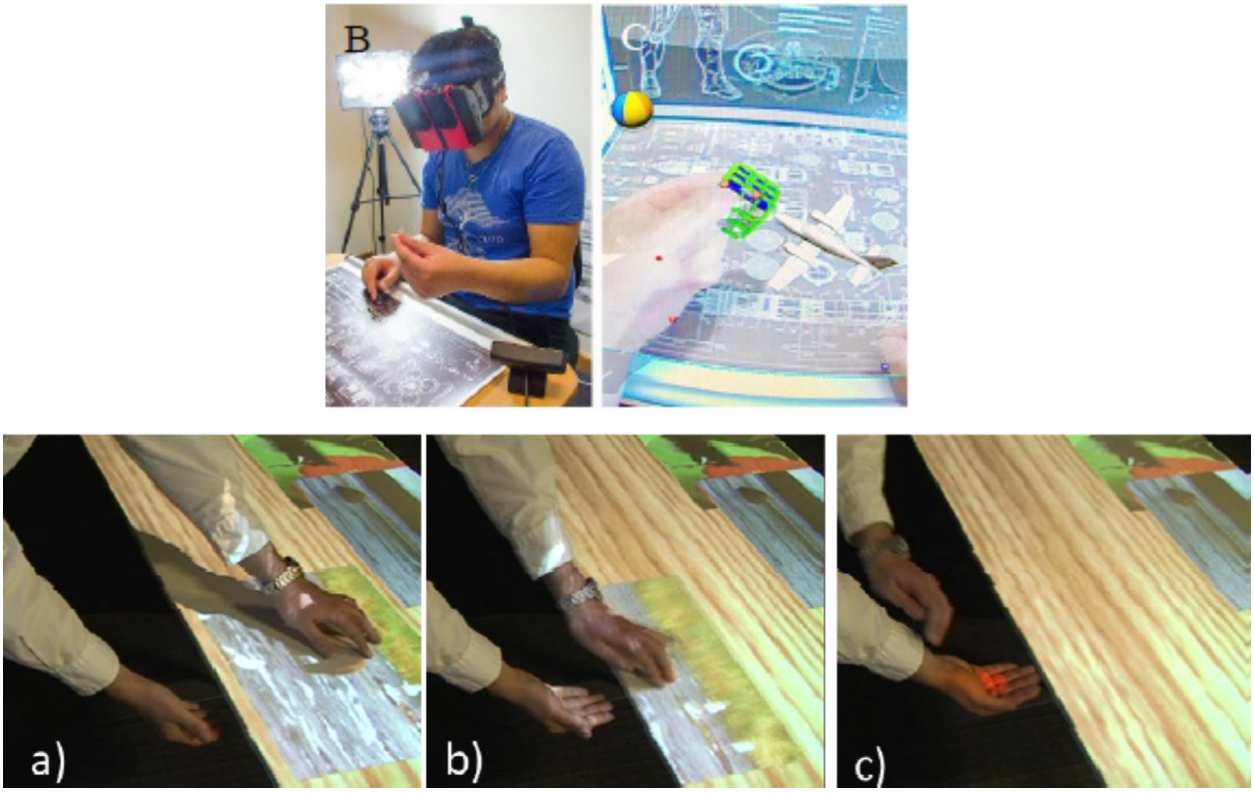

Figure 2.2. (Top row) User grasps onto a virtual object with bare hand pinch gesture [Piumsomboon et al., 2014]. (Bottom row) User picks up a virtual object projected onto a table by swiping it into the user's own hand. Once picked up, the virtual object moves with the user's hand [Wilson and Benko, 2010].

AR provides great potential to improve multi-user interactions with its capability for seamless interaction between real and virtual environments and enhanced awareness of other collaborators' activities through supplemental spatial cues. One of the earliest systems demonstrating collaborative multi-user interaction in an AR environment was *Studierstube* [Szalavári et al., 1996]. It allowed users to collaboratively view and interact with 3D virtual models through see-through head-worn displays. Its developers experimented with interaction with a "personal interaction panel" in which the user holds the panel in one-hand and interacts with a pen held in another hand. An evaluation showed that the system worked better than traditional screen-and-mouse based interaction for complex 3D model viewing and manipulation. Broll and colleagues

examined the affordance of physical objects as a mean to manipulate virtual contents registered with them in collaborative environment [Broll et al., 2000]. Ishii and colleagues facilitated collaborative urban planning by integrating 2D sketches, 3D physical models, and digital contents in a single information space [Ishii et al., 2002]. Projected augmentations of virtual shadows, wind patterns, and traffic are overlaid on top of the sketches and physical models placed in the interaction space through a ceiling-mounted projector.

While interaction through tangible interfaces can provide haptic feedback, it limits how and what interactions can be performed. To address this, researchers have sought to support natural and free-form interactions by means of bare-hand gestures. Lee and Höllerer demonstrated a technique to inspect 3D models using an open or closed hand gestures [Lee and Höllerer, 2007]. The interaction is limited to object selection and small degree of rotation restricted by nature of hand posture. Wang and colleagues extended this to full 6DOF tracking of bimanual hand gestures by leveraging commercially available depth sensors [Wang et al., 2011]. Piumsomboon and colleagues combined hand tracking with physics simulation to allow realistic manipulation of virtual contents through bare-hand interaction [Piumsomboon et al., 2014] (Figure 2.2 top row).

People have also expressed interest in supporting "cross-dimensional interactions." Regenbrecht and colleagues allowed users to seamlessly transfer data between 2D and 3D contexts using tangible user interfaces [Regenbrecht et al., 2002]. Their system makes it possible for users to transfer a model selected on a PDA (2D environment) to a physical turn table (3D AR environment) in which the users can rotate to view from their desired perspective. Benko and colleagues supported more intuitive interaction with hand gestures tracked by a wearable glove [Benko et al., 2005]. Users transfer data from a 2D tabletop display to an AR environment by

performing a grab gesture on the table and transferring the data back to 2D display by performing a push gesture on the table.

While display and interaction medium are usually separated in collaborative AR, researchers have presented ways to break through this limitation. Benko and colleagues cleverly combined a projector and depth camera to allow the users to interact with virtual content projected onto a body part or surface in a physically believable manner [Benko et al., 2012; Wilson and Benko, 2010] (Figure 2.2 bottom row). Baldauf and Fröhlich explored multi-user interaction with public displays through mobile devices [Baldauf and Fröhlich, 2013]. The user sees the AR overlay on her mobile device while using the mobile device to control the content displayed on the public display.

Although these techniques support interaction through 3D input devices, tangible interfaces, or body parts, the interaction mainly affects the overlaid virtual contents, but not any physical objects in relation to the virtual contents. We are interested in addressing interactions with virtual contents that will in turn affect their associated physical objects; which we believe can be more applicable in real world settings.

2.3 Selection Techniques

Object selection, which involves the specification of one or more objects by the user, is one of the most fundamental user interactions in a 3D environment. While numerous selection techniques have been introduced thus far, most of these techniques can be classified as immediate selection (e.g., ray-casting [Bolt, 1980]), in which the target is selected in single step with high precision, or refinement-based selection (e.g., SQUAD [Kopper et al., 2011]), in which the target is selected after several progressive refinement steps with looser precision for intermediate steps. Immediate selection techniques have the advantage of speed over refinement-based selec-

tion techniques when objects are large and sparsely distributed in the space. On the other hand, refinement-based selection techniques provide better accuracy with comparable speed for small objects in a cluttered environment. Our referencing techniques exploit refinement-based approaches with expectation that the increased accuracy will also improve the overall speed of the technique by allowing the interpreting user to easily and correctly understand the referenced object or intention.

2.3.1 Immediate Selection Techniques

Bowman and colleagues [Bowman et al., 2004] classified selection techniques into three categories by metaphors: selection by pointing (e.g., ray-casting [Bolt, 1980; Liang and Green, 1994]), selection by touching (e.g., virtual hand [Mine, 1995; Poupyrev et al., 1996]), and selection by scaling (e.g., WIM [Stoakley et al., 1995]).

2.3.1.1 Selection by Pointing

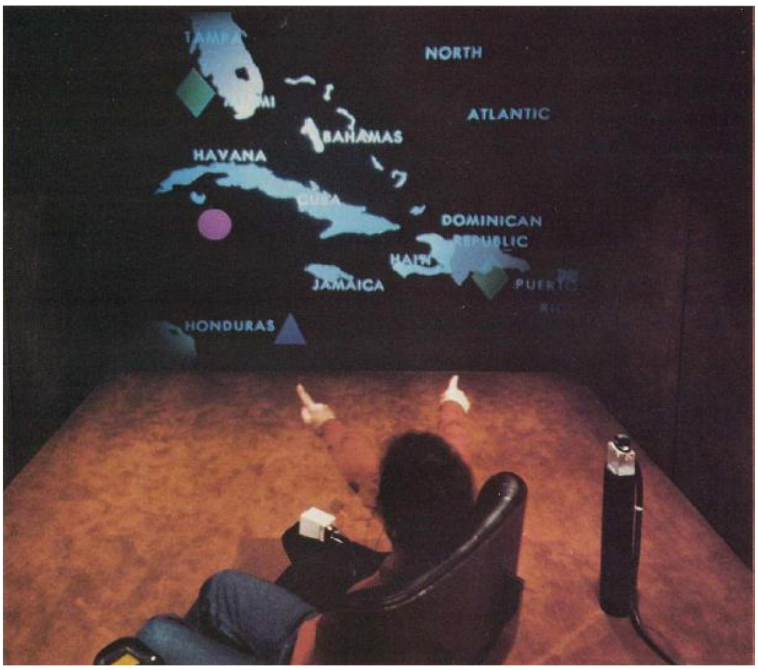

Figure 2.3. Points to an item displayed on a large screen with an index finger [Bolt, 1980].

Selection by pointing is by far the most widely used technique as it coincides with how we select objects naturally. Bolt is one of the earliest researchers to explore finger pointing techniques for selection in 3D environment [Bolt, 1980] (Figure 2.3). Liang and Green implemented a technique called “laser gun” [Liang and Green, 1994], resembling a physical laser pointer, in which the user points with a virtual ray extending from the tracked input device to specify an object that first intersects with the ray. However, a simple ray-casting technique suffers from two problems: instability and occlusion. Instability caused by hand tremor and poor tracking quality makes it difficult for users to aim at distant and small targets and can potentially increase the selection time. Occlusion happens when multiple objects exist along the line of the ray, and it can be difficult or impossible to aim at an occluded target.

To address instability, Liang and Green introduced a spotlight selection [Liang and Green, 1994] in which, instead of emitting a ray, the user emits a conic volume with a fixed apex angle originating from the tracked input device and any objects that intersect with this volume becomes a candidate. This allows the user to easily select a distant and small target without precise control of the tracked input device or user’s hand. Forsberg and colleagues extended spotlight selection technique to allow the modification of selection volume by controlling the apex angle [Forsberg et al., 1996]. The apex of a conic volume originates from the user’s dominant eye whereas the direction vector of this cone is controlled by a tracked device held by the user. A circle with a fixed radius is attached to the tracked device, and this circle determines the cross-section radius of the cone at the tracked device location. The user controls the apex angle by moving the circle towards or away from her dominant eye. A similar approach was proposed by Pierce and colleagues to perform selection on 3D objects at a distance by interacting with the 2D projection on the user’s image plane [Pierce et al., 1997].

However, the enlarged selection volume introduces ambiguity because more than one object may fall inside the volume, especially in a cluttered environment. Liang and Green formulated a metric to determine which object would be selected based on the distance between the target to the apex and central axis of the cone [Liang and Green, 1994]. Olwal and colleagues employed a statistical ranking calculated from multiple metrics including time (intersection time with the selection volume), stability (frequency of intersection with the selection volume), distance (distance from the origin of the selection volume in 3D space), visibility (occupancy percentage of the object within the selection volume), center-proximity (pixel distance from center of object to center of selection volume projected in 2D plane), the context of the selection based on speech commands (e.g., "this" vs. "that"), and gesture to determine the candidate [Olwal et al., 2003]. The bubble-cursor [Grossman and Balakrishnan, 2005] is another technique that intends to solve ambiguity by dynamically resizing a circular cursor so that it only contains one object at a time. Vanacken and colleagues extended bubble-cursor to 3D using spherical volume instead of circular area [Vanacken et al., 2007]. Even though these heuristic-based techniques resolve ambiguity, subtle movement of the tracked device may induce frequent changes of candidates, especially in a cluttered environment.

An alternative to these heuristic-based technique is to provide an explicit mechanism for users to disambiguate their intended target among other intersected objects [Hinckley et al., 1994]. Grossman and colleagues allowed the user to cycle through intersected objects with forward and backward hand movements [Grossman et al., 2004]. Grossman and Balakrishnan later extended this work to continuously move a depth marker along the line of the ray (depth-ray technique), so that the closest object to the marker is selected as an candidate [Grossman and Balakrishnan, 2006].

Other proposed techniques explore the change of control-display ratio to solve the instability issue. The PRISM technique [Frees et al., 2007] scales down the hand movement when moving slowly to increase the angular width of the target. The ARM technique [Kopper et al., 2010] allows the user to manually scale down the control-display ratio activated by a button press.

To address occlusion, Olwal and Feiner proposed the flexible pointer [Olwal and Feiner, 2003], which allows the user to bend the cursor using two 6DOF-tracked devices to point to fully or partially obscured objects. Wyss and colleagues used two simultaneously controlled rays to specify a target at the intersection of the two rays [Wyss et al., 2006]. A drawback of these techniques is that both hands are occupied for the selection task.

2.3.1.2 Selection by Touching

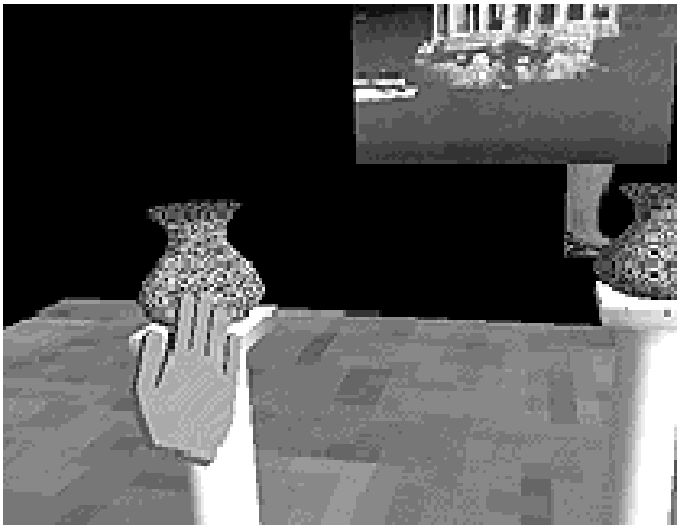

Figure 2.4. Go-go interaction technique extends the user's virtual hand to select an object at distance [Poupyrev et al., 1996].

A more direct method of interaction, in which ambiguity and occlusion are not an issue, is to use a 3D pointer to directly specify the 3D coordinate position for selection [Hinckley et al., 1994]. Mine describes a direct hand interaction for selecting objects within arm's reach [Mine,

1995]. The go-go technique [Poupyrev et al., 1996] explores a selection technique at a distance using nonlinear mapping between the actual distance the user's hand moved and the distance travelled by the 3D virtual hand to extend the range which the technique can cover (Figure 2.4). However, a number of studies [Bowman et al., 1999; Grossman and Balakrishnan, 2006; Poupyrev et al., 1998] showed that selection techniques that require direct specification of 3D coordinate position result in longer selection time. As an alternative to a simple 3D pointer, Zhai and colleagues developed the "silk cursor" technique [Zhai et al., 1994] which explores the use of a semi-transparent cubic volume cursor. Again, even though the increase of selection area leads to shorter selection time, the ambiguity issue arises in a cluttered environment and a dis-ambiguation mechanism has to be provided.

2.3.1.3 Selection by Scaling

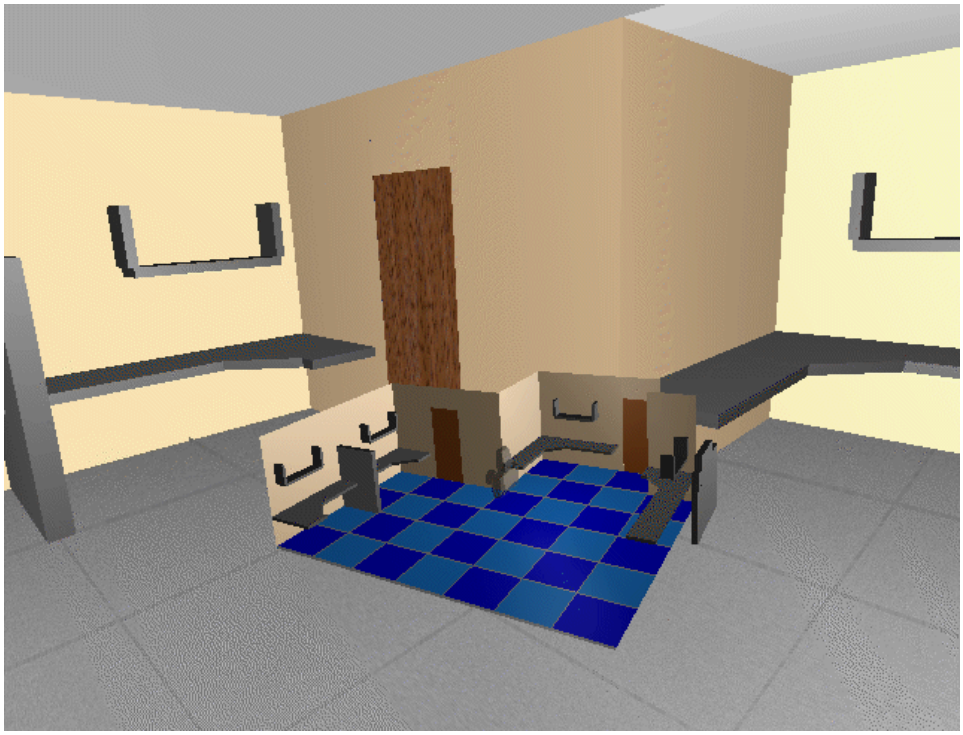

Figure 2.5. Selection in a scaled version of the virtual world (WIM) [Stoakley et al., 1995].

An alternative to extending the length of the user's arm is to scale the entire or part of the world and bring it within the user's reach. Stoakley and colleagues introduced a world-in-miniature (WIM) metaphor [Stoakley et al., 1995], in which an exact copy of the environment at a smaller scale is presented to the user at a distance within arm's reach (Figure 2.5). A selection can then be performed by pointing to its proxy on the WIM.

2.3.2 Refinement-based Selection Techniques

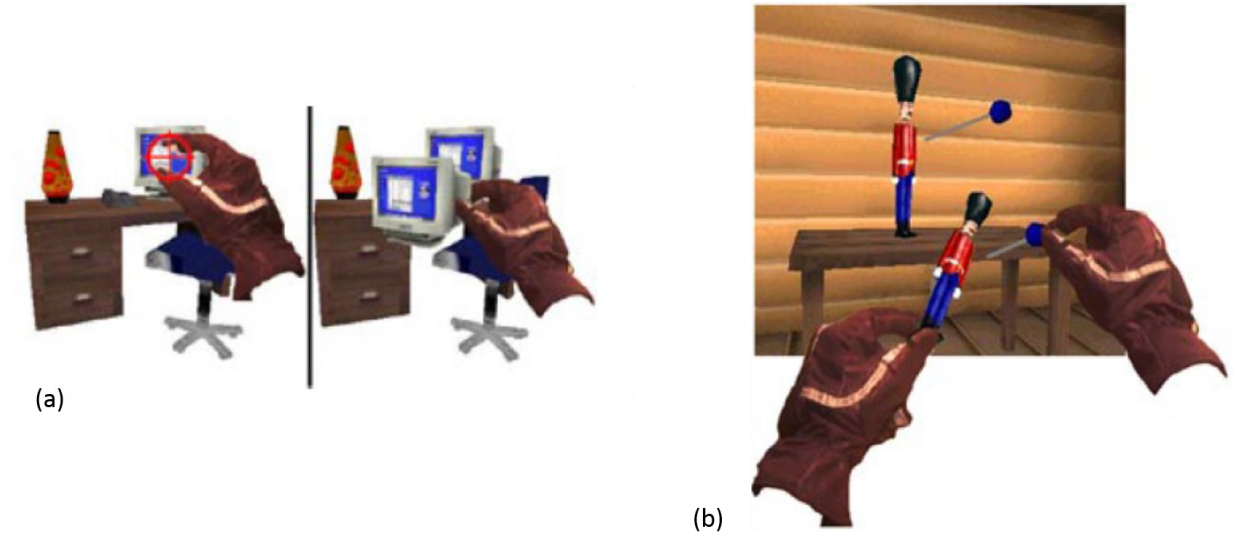

Figure 2.6. Voodoo dolls technique [Pierce et al., 1999]. (a) User first makes a copy of a virtual object using an image plane selection technique [Pierce et al., 1997]. (b) Once copy is created, user performs precise selection or operation on the copy within arm's reach. The result of the selection or operation is reflected on the original counterpart.

A typical refinement-based selection technique allows the user to roughly specify an area or volume that contains a potential target in the initial step, and perform more precise selection within the contained objects on the subsequent refinement phases. The voodoo dolls technique [Pierce et al., 1999] is a two-step selection technique, in which the user initially makes a virtual copy of an object in the environment and performs more precise selection on the duplicate presented within the arm's reach (Figure 2.6). Similarly, the augmented viewport [Hoang and

Thomas, 2010] captures a part of the scene at distance using a zoom lens or physical camera positioned near the target object, and the user performs selection on the zoomed image.

Grossman and Balakrishnan introduced variations of their depth-ray technique, lock-ray and flower-ray, that are executed in two phases [Grossman and Balakrishnan, 2006] in an attempt to avoid potential interference between the ray and depth marker manipulation. Both variations let the user fix the position and orientation of the ray after intersecting a list of objects along the ray. Then the final step is reduced to 2D linear (lock-ray) or radial menu (flower-ray) selection. However, these techniques do not work well for small and distant objects in a dense environment.

Kopper and colleagues proposed a progressive refinement technique, SQUAD [Kopper et al., 2011], in which the user initially casts a ray and creates a spherical selection volume, centered at the nearest intersecting surface. Upon completion of the first step, all objects intersected by the spherical volume are removed from the original context and evenly distributed among four quadrants on a separate context that only displays the quadrant view. The user then successively selects the quadrant that contains the target object with each refinement step, quartering the number of candidates until the target is the only one left in the quadrant. Due to the loss of spatial association between the scene and the quad-menu, they require the target to be visually distinctive from other objects. While this technique can be very accurate, it increases the number of iterative selections and disconnects the selected object from the original context. Cashion and colleagues extended SQUAD to present the initially selected set of objects in an evenly distributed grid in the original context [Cashion et al., 2012]. The grid presentation reduces the number of iterative selections and retains the connection from the original context, but the connection is still

ambiguous as to which object on the grid corresponds to which object in the original set, especially when multiple objects are visually indistinctive.

2.4 Visualization Techniques

A referencing activity is considered complete when the recipient interprets the referenced object or action correctly. However, it can be challenging sometimes to accurately interpret due to ambiguity, occlusion, or lack of depth cues. We examine several visualization techniques that can potentially assist the recipient to better understand the referenced object or intended action.

2.4.1 Assisting Interpretation of Object of Interest

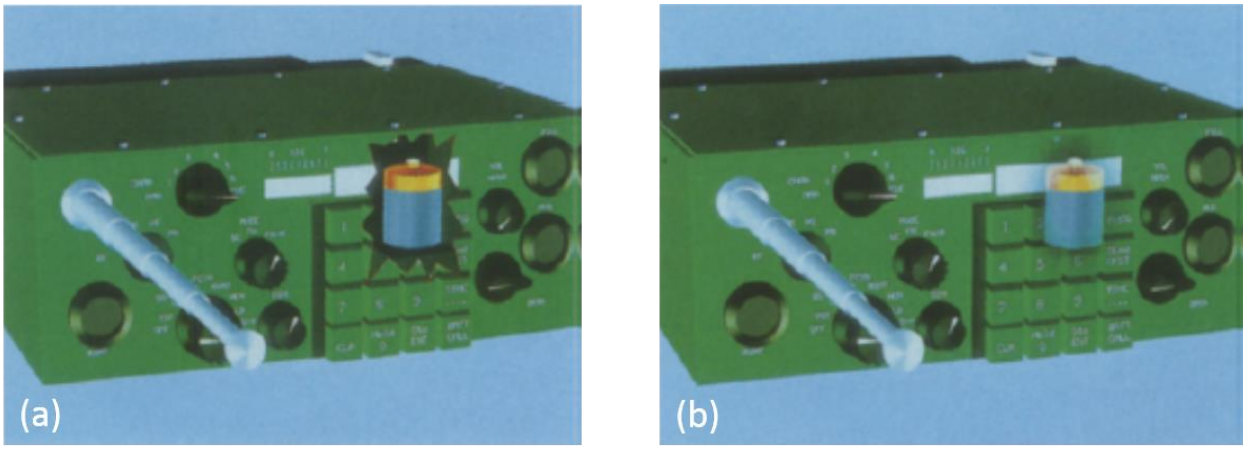

Figure 2.7. Cutaways and ghosting visualization [Feiner and Seligmann, 1992] exposes the occluded internal battery by rendering the occluding object with (a) a clear cutaway or (b) a feathered cutaway.

A classic and simple approach to disambiguate an object of interest from others is highlighting. Highlighting virtual content can be implemented in many ways, such as a change in color, change in transparency, change in size [Wang et al., 2005], addition of wireframe bounding box [Strauss and Carey, 1992], or mixture of these. While highlighting works well under the assumption that the recipient can see the highlighted object from her perspective, it does not work if there are occluding surfaces between the highlighted object and the recipient.

To address this issue, Feiner and Seligmann proposed a z-buffer based approach in which the occluding surfaces are substituted with a combination of cutaway holes and semi-transparent representations to disclose the object of interest [Feiner and Seligmann, 1992] (Figure 2.7). Li and colleagues extended this approach to support complex geometries and multiple objects of interest for static scenes [Li et al., 2007]. Burns and Finkelstein further extended to render view-dependent cutaways at interactive frame rates for dynamic scenes [Burns and Finkelstein, 2008]. Such "cutaways and ghosting" visualizations were shown to be effective in a collaborative VR environment [Argelaguet et al., 2011], and also deployed to an AR environment to reveal virtual objects behind physical occluding surfaces [Furmanski et al., 2002; Mendez and Schmalstieg, 2009]. Researchers have also experimented with overlays of 2D image or video stream on the occluding surface taken from a static camera placed behind it to provide X-ray vision [Avery et al., 2009; Barnum et al., 2009].

In an AR context, it is often necessary not to only understand the referenced virtual content, but also to correlate its positional association with the surrounding physical environment. In order to correctly understand the spatial arrangement between the virtual and physical elements, proper depth cues must be provided [Azuma, 1997]. Wither and Höllerer embedded a set of virtual markers in the scene as a means to aid depth cues [Wither and Höllerer, 2005]. Chastine and colleagues cast virtual shadows on the physical ground plane [Chastine et al., 2008]. People can even perceive the virtual content's correct position in 3D space without any additional visualizations under certain conditions. For example, if virtual objects lay on a shared surface with physical objects, people can perceive the relative position among them from the perspective view, assuming that the virtual camera's intrinsic and extrinsic parameters reflect the characteristics of a

real camera for video–see-through AR or the human eye for optical see-through AR [Kalkofen et al., 2011].

For correct depth perception, occlusion between the virtual and physical elements must be handled properly as well. Wloka and Anderson dynamically computed depth values for each pixel using a pair of stereo images to produce occlusion [Wloka and Anderson, 1995]. Breen and colleagues rendered invisible and registered virtual representation of each physical counterpart into the z-buffer for depth comparison [Breen et al., 1996].

2.4.2 Assisting Interpretation of Intended Action

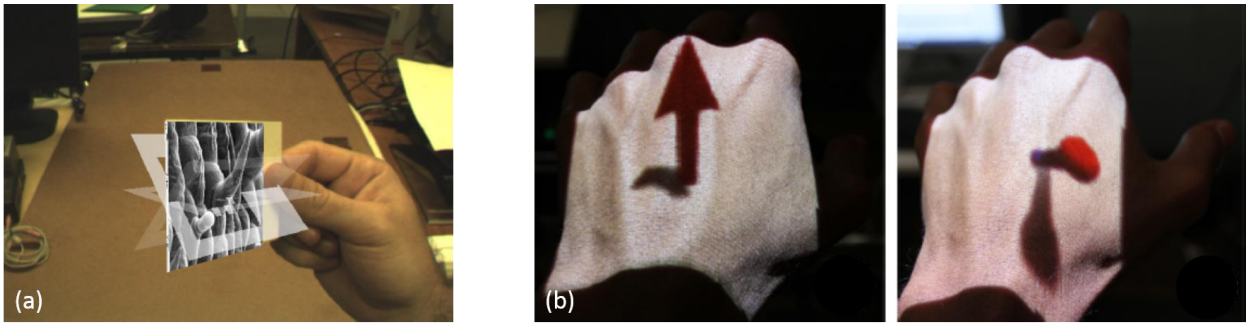

Figure 2.8. Techniques for assisting interpretation of intended action. (a) A visual hint demonstrating a twirling action to perform [White et al., 2007]. (b) Instructions with 3D arrows are projected onto user's hand to demonstrate how the user should move her hand [Sodhi et al., 2012].

Tversky and colleagues suggest that animation can be effective for conveying movement of an object [Tversky et al., 2002]. White and colleagues introduced *visual hints* [White et al., 2007] to help the user understand intended gesture or action (Figure 2.8a). The system displays a sequence of semi-transparent virtual representations (*ghosting*) of a manipulable target around the physical target with varied opacity. Ghosting is a hint to the user to follow the trajectory from more opaque to less opaque representations to complete the intended action. A study showed that a combination of ghosting and animation was most preferred over other combinations including textural and diagrammatic hints. Sodhi and colleagues projected visual hints in their system,

LightGuide [Sodhi et al., 2012], to guide the user to achieve desired motion of a body part (Figure 2.8b). The system projects incremental movement instructions such as 3D arrows in real-time onto the user's hand, and was shown to achieve an accurate trajectory. Tang and colleagues employed a multi-camera system to guide whole-body movement [Tang et al., 2015]. A top down and front view of the user are captured through cameras and displayed on a screen in front of the user with combined imagery of the user's body and dynamic visual hints overlaid on each part of the body. Others have explored 3D arrows [Henderson and Feiner, 2011], shadow [Freeman et al., 2009], or a ghost arm [Yang and Kim, 2002] to guide movement of body parts or physical objects.

Chapter 3. A 3D Referencing Technique for Shared Augmented Reality Environments

Figure 3.1. A selecting user (the *indicator*) points at a close-up virtual representation of a portion of the physical environment sensed by a depth camera with her finger to indicate the target object. (The coarseness of the close-up representation and reconstructed virtual representation of the indicator's hand in this figure and the following figures is due to low depth-camera resolution.)

One of the most important issues in cooperative multi-user interaction is awareness of interest [Fuchs et al., 1995]. When users interact in a shared environment, whether real or virtual, one user may wish to draw the attention of another to a specific object or region. A traditional

approach to selecting and communicating such a target is to point at it with a finger or hand. However, if the specific object or region of interest is farther away than arm's length and difficult to disambiguate verbally, it can be hard for others to understand what is being referenced. This can be even more difficult if the perspective of the person pointing is significantly different from the perspective of the person who is interpreting which object is being referenced.

In this chapter, we explore approaches to referencing objects in shared environments and introduce a new referencing technique suitable for previously unmodeled physical environments. Our technique, GARDEN (Gesturing in an Augmented Reality Depth-mapped ENvironment), uses RGB and depth cameras to sense the environment and AR to overlay representations of the physical environment that support selecting and communicating the object being referenced, as shown in Figure 3.1. These representations present close-up views of a depth-sensed region of the physical environment enclosed in a user-specified spherical volume to both selecting and interpreting users, and present the depth-sensed gesturing hand of the selecting user as well as an animation to the interpreting user. We have designed GARDEN to be used in situations in which it is difficult or inappropriate for the users to walk to the target object and point at it directly, and in which verbal communication is ineffective. For example, certain objects may be visible but not easily reachable (e.g., exposed ceiling beams or artwork behind a protective barrier) or may be time-consuming or dangerous to approach. As another example, the involved parties may speak different languages or have difficulty hearing (due to disability or a noisy environment).

We performed a user study in which we compared GARDEN against existing augmented reality referencing techniques as well as the use of a physical laser pointer. Our study confirmed that GARDEN was significantly more accurate than the techniques with which we compared it when used by participants who did not share a common view of the objects being referenced, and

significantly more accurate than one of these techniques when the participating users have similar perspectives. We also noted that while there were instances in which the interpreting user could not even provide a guess as to which object was being referenced, there were significantly fewer such occasions using GARDEN than with the comparison techniques.

In the remainder of this chapter, we first discuss related work in Section 3.1. Then, in Sections 3.2–3.4, we describe our technique and a set of comparison techniques, and discuss how we implemented them. Next, in Sections 3.5–3.6, we present the pilot and formal studies with which we evaluated the techniques. Finally, we discuss our study results in Section 3.7.

3.1 Related Work

Referencing has been described as a sequential process composed of the phases of selection, representation, and acknowledgement [Chastine et al., 2006]. First, one user (the *indicator*) *selects* an object; for example, by using her hands to form a 'frame' on the image plane [Pierce et al., 1997]. Then, the selected virtual object is *represented* visually using highlighting, perhaps combined with illustrative techniques such as cutaways and ghosting [Feiner and Seligmann, 1992], so that one or more other users (the *recipients*) can clearly understand what is being selected. Finally, the recipients *acknowledge* to the indicator that they have understood what has been referenced; for example, by nodding [Davis and Vaks, 2001].

Although there has been extensive research conducted on different approaches for selection, representation, and acknowledgement, there appears to be relatively little research focused on combining two or more of the three phases for shared environments. An approach that works well for an individual phase does not necessarily work well in context of the others. For example, virtual ray-based or 3D cursor-based selection techniques [Bolt, 1980; Grossman and Balakrishnan, 2006; Liang and Green, 1994; Olwal and Feiner, 2003; Poupyrev et al., 1996;

Wyss et al., 2006] can be simple and efficient ways for an indicator to select an object at a distance, but which object was selected may not be clear to a recipient viewing the environment from a different perspective. Consequently, we are interested in 3D referencing techniques that would allow a user to select a physical object at greater than arm's length, while making it easy for other users in the shared space to understand which object has been referenced.

Chastine and colleagues performed a comprehensive set of studies on the use of virtual pointing arrows as a medium for both referencing and interpreting referenced objects [Chastine et al., 2008; Chastine and Zhu, 2008; Chastine et al., 2007; Chastine et al., 2006]. However, they focused mainly on *virtual* objects in shared environments. For real-world tasks, such as mechanical assembly, users will often be referring to *physical* objects rather than virtual objects. Since the locations and shapes of virtual objects are known to the system and can be easily modified, there are many techniques for providing visual feedback about virtual objects to recipients, such as highlighting or enlarging selected objects [Carpendale et al., 1996] to emphasize them. In contrast, the locations and shapes of physical objects may be unknown to the system unless a full model of the scene is available; furthermore, a physical object can be occluded by another physical object in the scene.

Hirakawa and Koike [Hirakawa and Koike, 2004] applied vision algorithms to infer both the selection of a physical object and acknowledgement of the referenced object. Their system analyses a static camera facing the direction of the users to estimate the indicator's pointing direction using a ray that connects their dominant eye and the fingertip of their outstretched arm. They analyze the live video of another static camera to track the positions of moving physical objects. However, their selection and acknowledgement technique would not work if the indica-

tor points at a direction that is not along this ray, or if their fingertip is not clearly visible to the recipient, which are situations that we would like to address.

Salzmann and colleagues compared pointing at real objects to pointing at virtual objects in a two-user scenario in terms of the accuracy with which the recipient interprets the indicator's selection [Salzmann et al., 2009]. They compared three different selection techniques for referencing a relatively small object, such as a button on an automobile dashboard: touching an object, drawing an outline around an object, and pointing from a distance. They showed that it is less accurate to interpret a selection of a virtual object, as opposed to a real object, for all three techniques due to incorrect perception of depth as well as lack of occlusion. They concluded that it is more accurate to interpret an object at a distance that is outlined than one that is pointed at without drawing an outline for both real and virtual objects.

Finally, we note that there are many other selection techniques that assume that the system has a model of the environment that specifies which objects can be selected and supports geometric tests for selecting them [Grossman and Balakrishnan, 2005; Kopper et al., 2011; Pierce et al., 1997; Vanacken et al., 2007], which is typical of virtual reality (VR) systems. However, these techniques cannot be applied directly to select a physical object in an unmodeled environment, whose structure is unknown to the system.

3.2 GARDEN

Based on the previous work reviewed above, we make the following observations. First, it is difficult for an indicator to point precisely at a region or object at a distance when using a direct line-of-sight–based approach (e.g., a virtual arrow, 3D cursor, or infinite ray) [Chastine et al., 2007; Liang and Green, 1994; Poupyrev et al., 1996]. This is due, in part, to the often large effect on the point of intersection of a ray with the scene resulting from even a small angular change in the ray's direction when tracking the user's hand pose. This can be especially troublesome for small targets. Second, a line-of-sight–based approach may look correct from the perspective of the indicator, but be confusing or wrong from the perspective of the recipient, because of differences in the projections and occlusion relationships [Chastine et al., 2007]. Third, outlining a region before pointing can improve the accuracy with which the recipient can interpret which object was referenced [Salzmann et al., 2009].

We have designed GARDEN, a referencing technique that takes these observations into account. GARDEN uses AR to present an indicator and a recipient with overlaid representations of the real world within which the indicator can gesture. These representations are created through the use of an integrated RGB camera and depth camera (Microsoft Kinect for Windows), and viewed through tracked stereo video–see-through head-worn displays worn by the indicator and recipient.

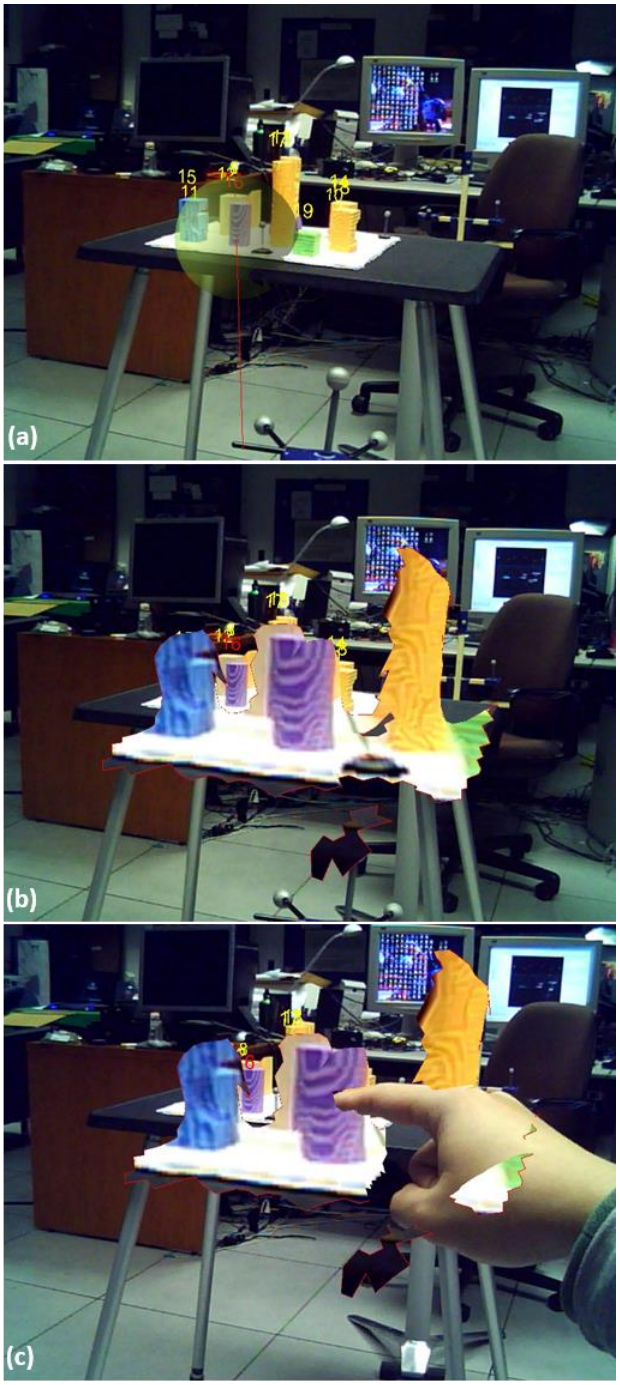

Figure 3.2. Interaction in GARDEN for the indicator. (a) The indicator's view through a stereo video–see-through head-worn display while initially specifying a spherical volume with a tracked Wii remote. The volume is placed at the intersection of a ray with the live depth map of the physical environment. Objects are overlaid with numerical annotations used only in the user study. (b) The texture-mapped depth mesh contained within the selection sphere is brought closer to the indicator. (c) The indicator points directly with her finger at the representation within the close-up mesh of the physical object to which she intends to refer (in this case, the purple cylinder at the center).

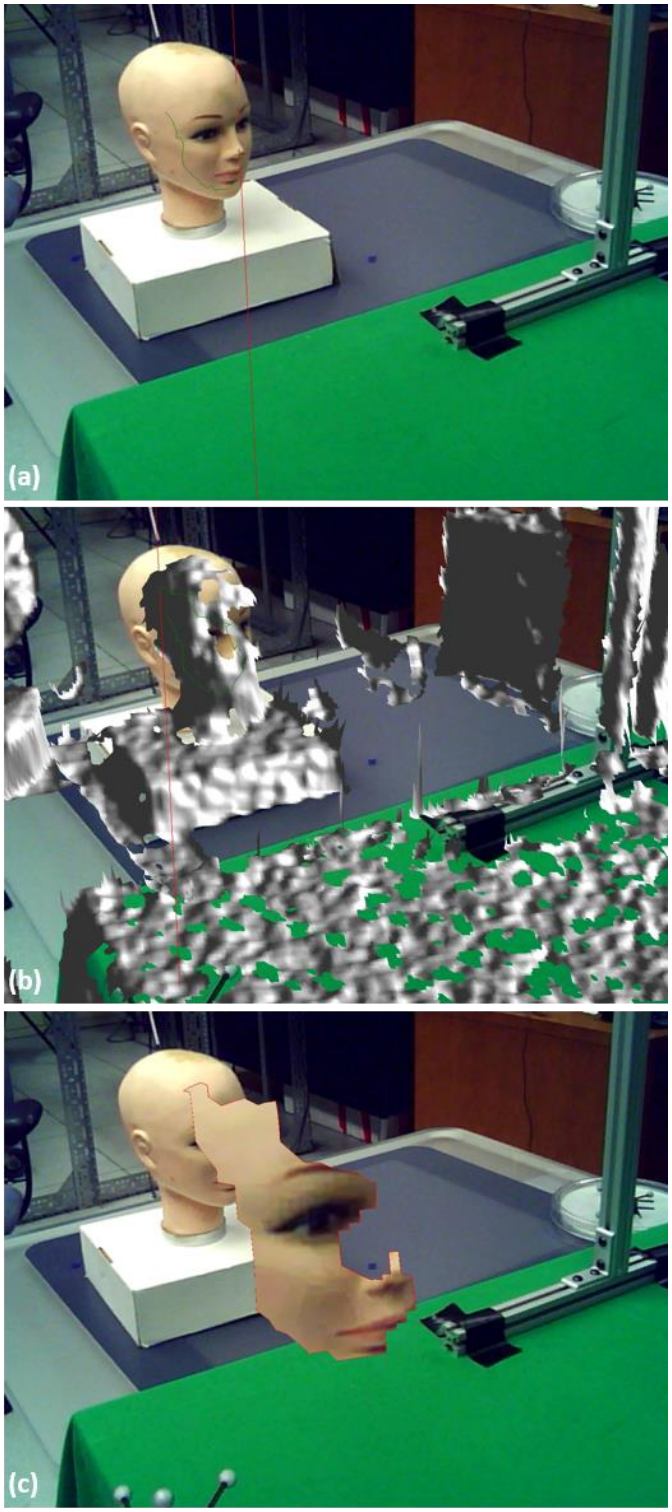

Figure 3.3. First outlining technique. (a) Enclosing outline (green lines showing the initial portion of an outline being drawn) is formed by connecting a list of intersected points between the infinite virtual ray (red line) originating from the tracked hand-held device and the depth map. (b) A view with the overlaid depth map (not shown to users). (c) The 3D mesh inside the enclosed outline with red boundary lines.

3.2.1 Interaction by the Indicator

Initially, the indicator specifies an approximate region that contains the target object (Figure 3.2a), similar to the outline technique of Salzmann and colleagues [Salzmann et al., 2009]. (For purposes only of our user study, part of the environment was pre-modeled and numerically labeled to help specify the target to which the indicator should point. The model and labels are not used by our technique itself.) We have implemented two methods for specifying this region, based on the intersection point between a mesh created from the depth map observed by a depth camera and an infinite virtual ray originating from an externally tracked pointing device (a Wii remote tracked with a 6DOF optical-tracking system in our implementation). Whenever the virtual ray intersects part of the physical environment captured by the depth camera, the portion of the virtual ray that should be obscured will be properly occluded.

In the first approach we tried, we had the indicator draw an enclosing outline on the physical scene around the desired object, viewed through a stereo video–see-through display. Figure 3.3(a) shows the initial portion of an outline being drawn. The enclosed triangle mesh inside the user-specified outline is computed as follows. We first recorded the 3D intersection points between the depth mesh and an infinite virtual ray every 50 milliseconds. These intersection points were then projected to the screen coordinate space of the indicator's head-worn display. (We cannot simply take the 2D coordinates of the depth map data since the depth sensor and the camera embedded in the indicator's head-worn display are at different locations, requiring the world-to-screen-space projection.) Next, we connected each pair of consecutive projected 2D points to form connected lines. The determination of whether the user-specified outline formed a closed loop was performed by testing whether the most recent line intersects with any of the preceding connected lines. We performed this check in 2D screen coordinate space rather than in the 3D world coordinate space, since the depth map is indexed in the 2D space of the screen, and if the

projected outline is a closed loop, so is the actual 3D outline on the mesh created from the depth map. The 2D computation is much simpler and less expensive compared to a similar 3D computation.

Once the outline becomes a closed loop, we then project the 3D depth map points to the screen coordinate space of the indicator's head-worn display and check which projected 2D points lie inside the closed loop on the screen space. Finally, we figure out which triangle vertices in the depth map mesh contain these enclosed points, and reconstruct the outlined 3D mesh using these triangles.

The outlining method requires careful manipulation of the Wii remote and the outline quality is very susceptible to jitter caused by the user's hand and the tracking system. It is also time-consuming to outline the region. To alleviate these issues, we implemented another method that encloses the region within a spherical volume centered at the intersection point. We constructed this region by first determining all vertices that lay inside the specified spherical volume. We then computed which triangles in the depth map mesh contain these vertices, and finally, created a mesh using the triangles that met the condition. We allow the indicator to change the radius of the sphere using buttons on the Wii remote. This method performed much faster than the outlining method, and was less susceptible to tracking instability; therefore, we decided to use the spherical volume method in GARDEN.

Once the indicator confirms their selection of a volume by pressing a button, GARDEN reconstructs a representation of the physical region within the volume from data obtained from the depth camera and RGB camera. This representation is presented to the indicator at a closer distance than the actual objects as shown in Figure 3.2(b). Based on a pilot study (Section 3.5.1), we determined that 50 cm would be an appropriate distance for the indicator to comfortably see

and point within it. The virtual representation is also world-fixed (the location is fixed in the 3D world) [Feiner et al., 1993], so that the indicator can move her head relative to it, which can also help differentiate the close-up representation from the indicator's view of the rest of the scene. (We originally intended for our implementation to use a helmet-mounted depth camera, rigidly connected to the head-worn display, but abandoned this approach because of the excessive bulk of the commodity components, which we did not want to disassemble. Instead, we use a stationary depth camera located near the indicator, so only small changes in head position will yield useful information.) Since the reconstructed mesh is texture mapped with a similar texture to that used for the background video, it was hard to visually discern the boundary of the reconstructed region from the background. To make the region stand out from the background, we added an outline around the reconstructed mesh (the red outline around the close-up representation shown in Figure 3.2b).

The indicator can redo the initial selection, if desired. When satisfied, the indicator can rotate the virtual representation about its up vector to view it from a different angle. The faces of the mesh created from the depth map of the spherical volume and the indicator's hand are rendered with a semi-transparent texture when seen from behind. This assists the recipient in determining which side of the mesh they are viewing. When ready to specify an object, the indicator points at the close-up virtual representation with her hand to indicate the object to which she is referring (Figure 3.2c).

Specifying an initial volume by casting a spherical volume around a point of interest has previously been used in the SQUAD technique of Kopper and colleagues [Kopper et al., 2011]. However, SQUAD works in a fully modeled virtual environment, making it possible for it to lay

out all objects intersecting the sphere for ray-cast selection through progressive refinement; thus, SQUAD could not be applied in our unmodeled environment.

Duplicating a portion of the world and moving it within arm's reach is used in the voodoo dolls technique of Pierce and colleagues [Pierce et al., 1999]. The voodoo dolls technique is a two-handed interaction technique in which the user creates a virtual copy of an original virtual object using an image-plane selection technique [Pierce et al., 1997]. The copied object is then attached to the index finger of the user's non-dominant hand, and the user can interact with the copy using her dominant hand while orienting and positioning the copy using her non-dominant hand. GARDEN is also related to the technique presented by Hoang and Thomas [Hoang and Thomas, 2010]. Their technique allows the user to interact with an enlarged view of a distant physical object in which either a distant remote camera, or a local camera with a zoom lens is used to obtain. While Hoang and Thomas capture and interact with a zoomed 2D camera view, we capture and interact with a zoomed 3D geometric representation. For example, this makes it possible to point at the side of an object, as seen from the depth camera. Furthermore, unlike these prior works, GARDEN allows one user to select an object and communicate that selection to another user, and do so in a previously unmodeled physical environment.

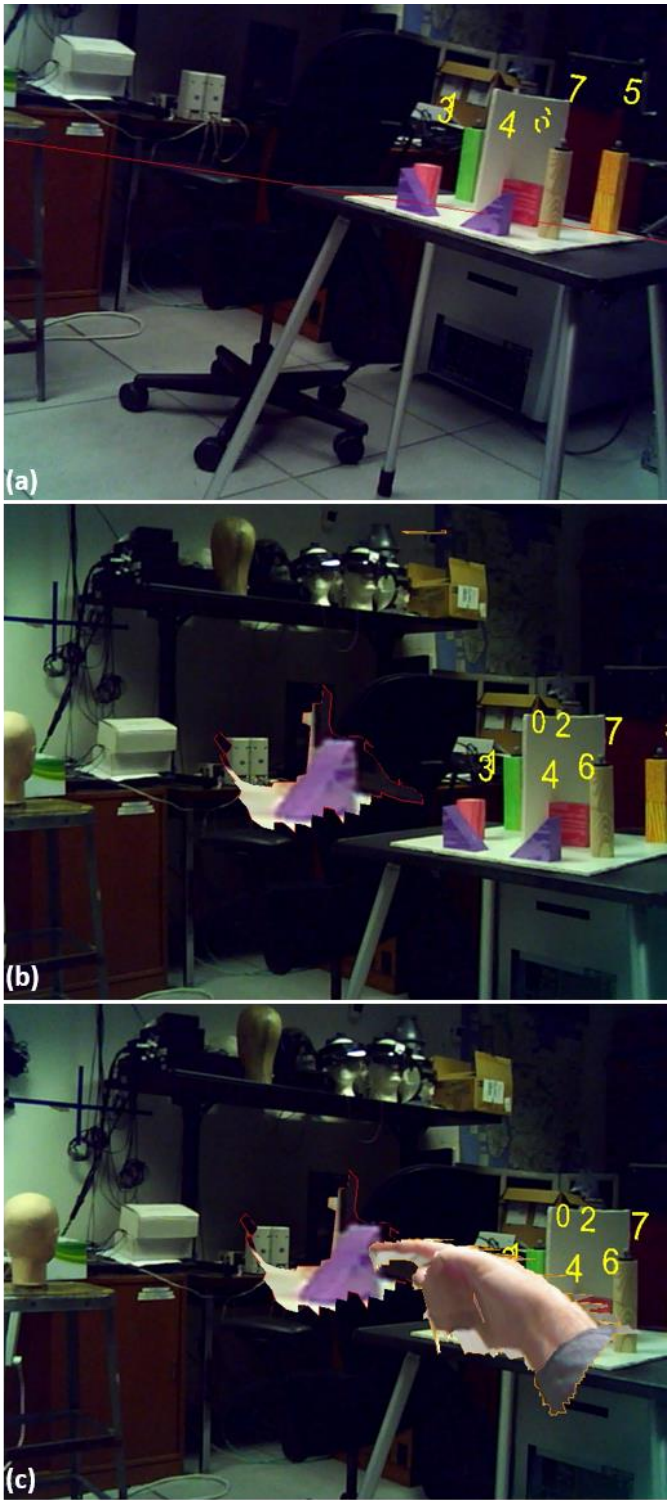

Figure 3.4. Interaction in GARDEN for the recipient. (a) The recipient's view through a stereo video–see-through head-worn display of the indicator's infinite ray. (b) A close-up representation of the spherical volume selected by the indicator. (Here, the indicator has selected a different region from that in Figure 3.2). (c) The recipient sees the selected volume and the reconstructed virtual representation of the indicator's hand.

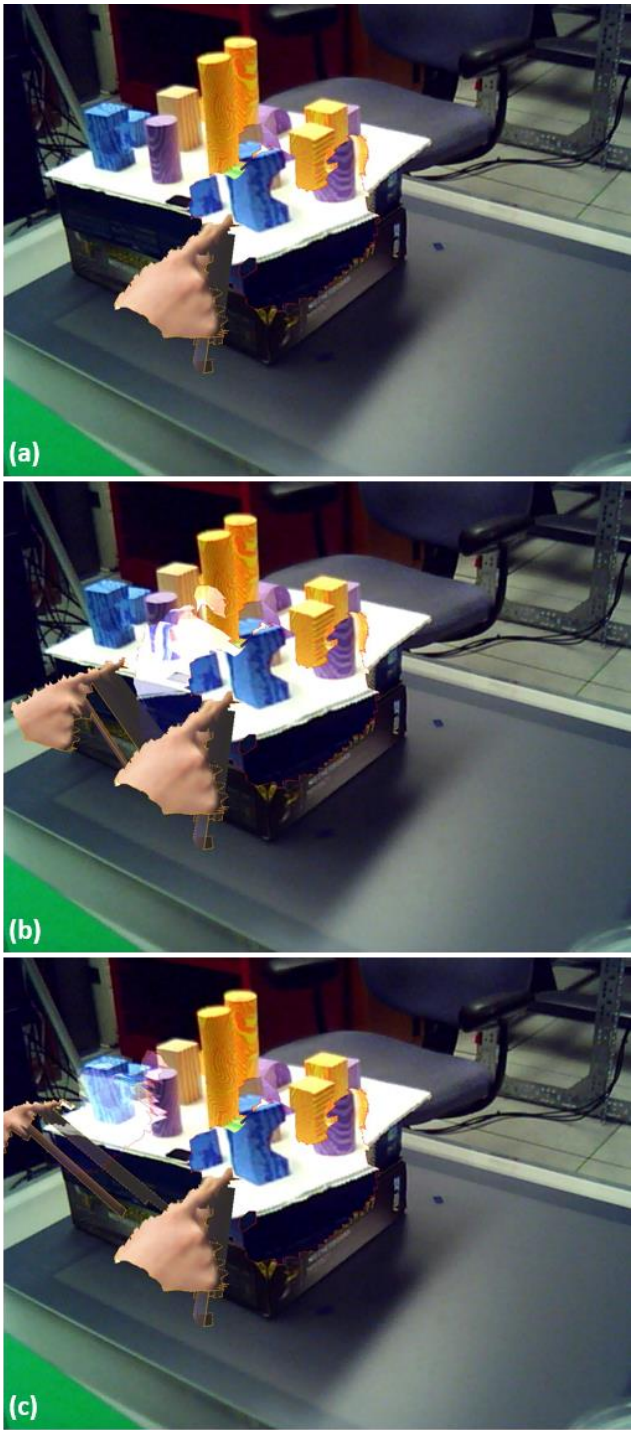

Figure 3.5. Animation in GARDEN for the recipient. (a) The virtual close-up representation at its initial virtual location. (b) A copy of the virtual representation and the indicator's hand are interpolated between the initial virtual location and the actual physical location. The virtual representation is rendered as transparent, while the virtual hand is rendered opaque. This allows the recipient to see the actual physical object being referenced when the copy arrives at the destination and the virtual hand pointing at the actual physical object. (c) The virtual representation and hand arrive at the destination.

3.2.2 Interaction by the Recipient

While the indicator is specifying the spherical volume, the recipient sees a properly occluded ray originating from the indicator's Wii remote (Figure 3.4a). Once the indicator finishes this initial selection process, a copy of the virtual representation of the enclosed region is presented to the recipient, oriented along the recipient's view direction and placed at a predetermined world-fixed [Feiner et al., 1993] distance near the recipient (Figure 3.4b), similar to what was done for the indicator.

The recipient then sees a virtual reconstruction of the indicator's hand pointing within the representation (Figure 3.4c). The virtual hand was reconstructed using the exact same method used for reconstructing the close-up representation by taking the triangles inside a spherical volume around the center of the close-up representation, based on the assumption that anything falling within this volume will be the indicator's hand. (We determined that 25 cm was an appropriate radius to enclose the pointing hand, while not including a large part of the wrist.) At this point, it can be relatively easy for the recipient to infer the portion of the virtual representation to which the indicator is pointing. However, the recipient would not know where the virtual representation maps to in the physical environment, since the virtual representation is not located at the real world location to which it corresponds. To assist the recipient in understanding the mapping between the virtual representation and the real world, GARDEN provides a user-controlled animation that interpolates a copy of the virtual representation and the indicator's hand seen by the recipient to the actual position and orientation in the physical environment of the spherical selection volume (Figure 3.5). This animation can be triggered by the recipient at any time while the indicator is pointing at the virtual representation of the enclosed region.

3.3 Comparison Techniques

We have implemented several existing techniques (Figure 3.6–3.8) that are commonly used for referencing tasks in physical environments to compare with GARDEN: pointing with a physical laser, pointing with a virtual arrow, and sharing video from the viewpoint of the indicator. The same 6DOF-tracked Wii remote used in GARDEN was used for these comparison techniques (Figure 3.6b).

3.3.1 Laser Pointer

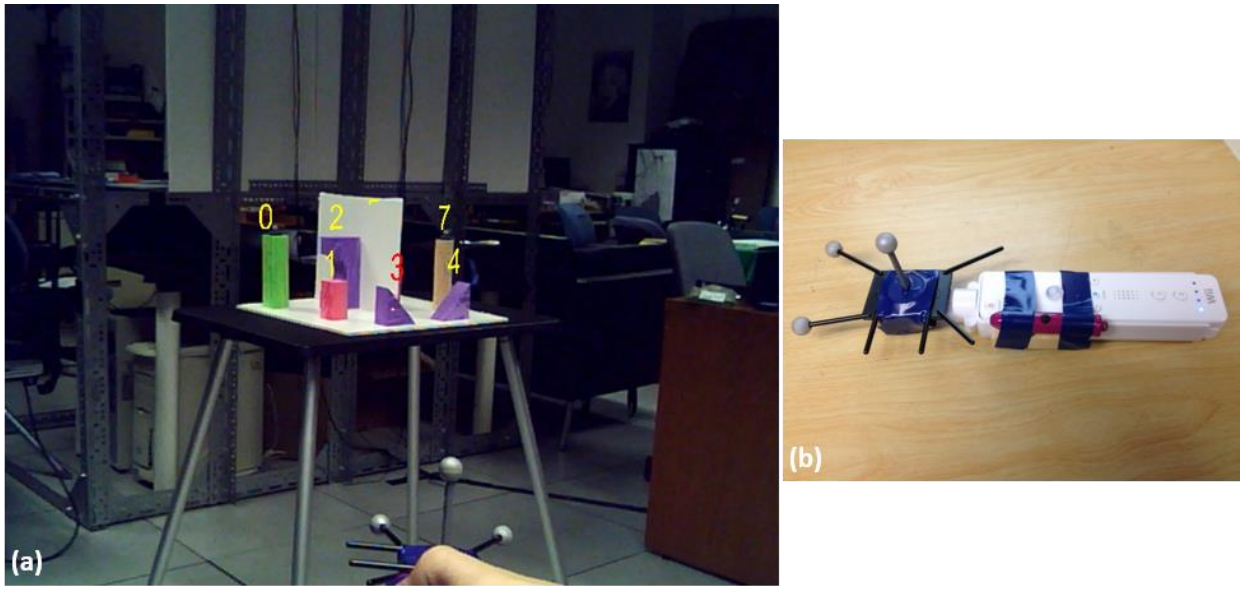

Figure 3.6. Laser Pointer. (a) A view from the indicator's perspective, showing the laser pointed at the object numbered "3". (b) A laser pointer used for this technique (Wii remote itself is not used).

A laser pointer (Figure 3.6a) is a popular means for pointing out distant physical objects to others. As shown in Figure 3.6(b), we attached a small laser pointer to the side of the Wii remote controller (The Wii remote controller itself is not used for this technique), which the user can operate with the button on the laser pointer's barrel to indicate an object of interest. We decided to use a laser pointer instead of finger pointing [Hirakawa and Koike, 2004; Salzmann et

al., 2009] because the laser pointer's clearly visible dot eliminates the ambiguity of finger pointing when selecting objects in cluttered environments at greater than arm's length.

3.3.2 Virtual Arrow

Figure 3.7. Virtual Arrow. Viewed from the indicator's perspective.

Line-of-sight–based approaches (e.g., virtual arrow, 3D cursor, depth cursor, and aperture-based selection) [Chastine et al., 2008; Forsberg et al., 1996; Grossman and Balakrishnan, 2006; Liang and Green, 1994] are often used for selection and referencing tasks in 3D virtual environments, including AR, as described in Section 2.3.1.1. Despite their differences, they are all based on the fundamental concept of a ray of adjustable length originating from a point near the user's hand. We have chosen a 3D virtual arrow (Figure 3.7) to exemplify this set of techniques. The arrow originates from the Wii remote. Its length can be changed with two buttons on

the Wii remote that grow and shrink it at a constant rate. The 3D arrow is properly occluded when any portion of it is blocked by the mesh within which it points.

3.3.3 Video Share

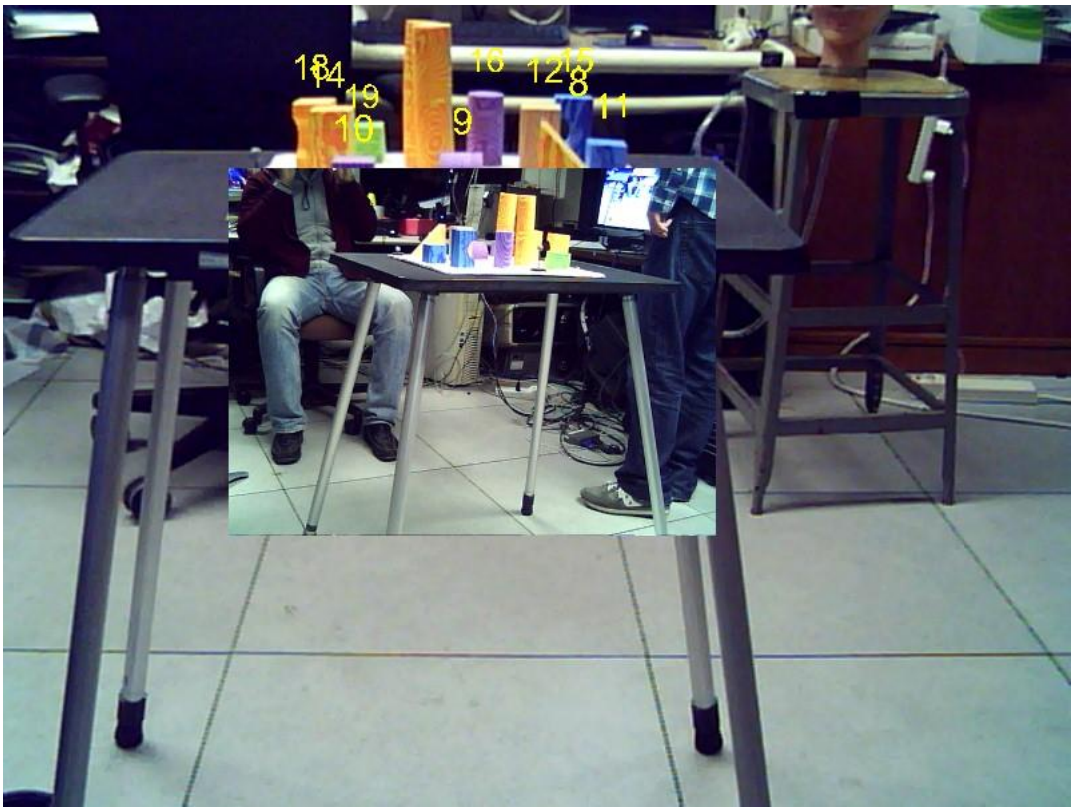

Figure 3.8. Video Share. Viewed from the recipient's perspective.

The laser pointer and virtual arrow can be quite effective for referencing tasks in which both the indicator and the recipient can see the laser dot or arrow tip clearly. However, there are situations in which the indicator can see the dot or arrow, but the recipient cannot. For example, when the indicator and recipient are facing each other and the physical object being referenced is between them, the laser dot may appear on a back side of an object relative to the recipient, and therefore not be visible to the recipient. One way to remedy the recipient's inability to see the real or virtual pointing geometry is to share the indicator's view with the recipient.

To accomplish this, while the indicator uses the laser pointer to select a physical object, we capture the video from the 640×480 resolution camera in one eye of the indicator's stereo video–see-through head-worn display. This video is streamed live to the recipient and texture-mapped onto a display-fixed (positioned at a fixed location and orientation relative to the display) [Feiner et al., 1993] 3D rectangle inset in both eyes of the recipient's view (Figure 3.8). Since the video texture is display-fixed, the recipient can move her head without losing the shared video view. (Since both users' video images are live, it is easy for the recipient to distinguish the recipient's video inset from the video background, which is somewhat difficult to appreciate in the still image of Figure 3.8.) The recipient can control the transparency of the textured 3D quad, so that she can see her own view clearly when desired.

3.4 Implementation

3.4.1 Software

Our experimental software is built using Goblin XNA [Oda and Feiner, 2012] (Appendix A). The Microsoft Kinect SDK [Microsoft, 2015] is used to obtain depth and color streams from a Microsoft Kinect for Windows v1. The depth maps of the environment and of the indicator's hand, captured on the indicator's side, are converted to 3D meshes, and textured with video capture from one of the cameras in the indicator's head-worn display. Copies of the textured meshes are viewed and manipulated by both the indicator and recipient when performing the referencing techniques, and allow proper occlusion of the pointing geometry (hand mesh, ray, or arrow) that intersects the mesh representing the contents of the selection sphere.

3.4.2 Hardware

Figure 3.9. Hardware used for the techniques and the study. (a) The indicator, with tracked head-worn display, hand-held Wii remote, and Kinect mounted on shelf. (b) The numeric keypad used by the recipient.

The application runs on Windows 7 for both the indicator and recipient. Each user is supported by a separate machine constructed with a 3.4 GHz AMD Phenom II X4 965 CPU with 8 GB memory, and an AMD Radeon HD 6990 GPU, connected through a Gbit public LAN. Each user wears a Vuzix Wrap 920 AR video–see-through stereo head-worn display, which displays a 640×480 side-by-side stereo pair (Figure 3.9a). The indicator uses a tracked Wii remote (Figure 3.6b) to perform the selection techniques, with a small cylindrical 5 mW red laser pointer attached on the side for easy activation. The recipient uses a numeric keypad (Figure 3.9b) to input answers and trigger actions in our user study.

A Microsoft Kinect for Windows v1 is rigidly attached above the indicator's head (top of Figure 3.9a) to capture both the physical environment and the indicator's pointing gestures. We initially tried to mount the Kinect on a helmet worn by the indicator to provide adjustable coverage of the depth-mapped area, but the weight of the Kinect made the helmet quite uncomfortable to wear, especially when moving the head around. We decided not to disassemble the Kinect to reduce its weight, and instead opted for mounting it in the environment. The Microsoft Kinect for Windows v1 has near (0.5m–3m) and default (0.8m–4m) modes. Our software automatically switches between the default mode to track the selection environment and the near mode to track the indicator's hand. The 3D position and orientation of the indicator's and recipient's head-worn displays, and the indicator's Kinect and Wii remote are tracked using a NaturalPoint OptiTrack V100 optical tracking system [NaturalPoint, 2015].

3.4.3 Calibration

To properly align and overlay the environment captured by the cameras in the Kinect, and Wrap 920AR displays, we first calibrated the intrinsic and extrinsic parameters of each camera using OpenCV [Intel]. We then calibrated the rigid transformation between the OptiTrack retroreflective spherical targets mounted on each of the devices and its camera view using the calibration technique of Fuhrmann et al. [Fuhrmann et al., 2001]. Finally, we calibrated the stereo separation for each Wrap 920AR and adjusted the field of view to better match each eye's 31° diagonal field-of-view display by scaling and cropping its video, using a tool we developed for the Goblin XNA framework (Appendix A.2.1).

3.5 User Study

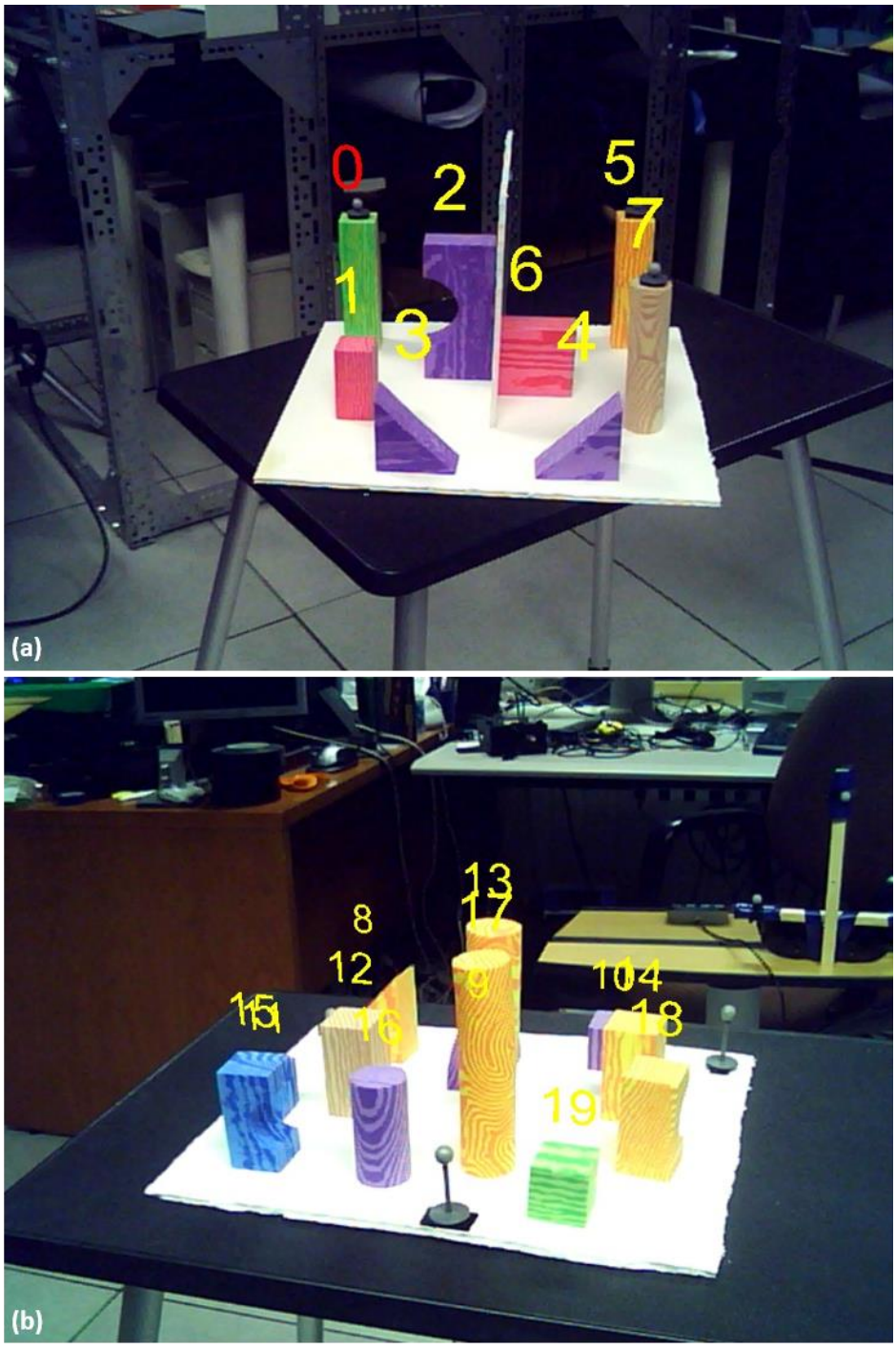

Figure 3.10. The physical objects used in the user study, with overlaid virtual numeric identifiers. (a) Set 1. (The object under the red "0" is the one the indicator should select during the current study trial.) (b) Set 2.

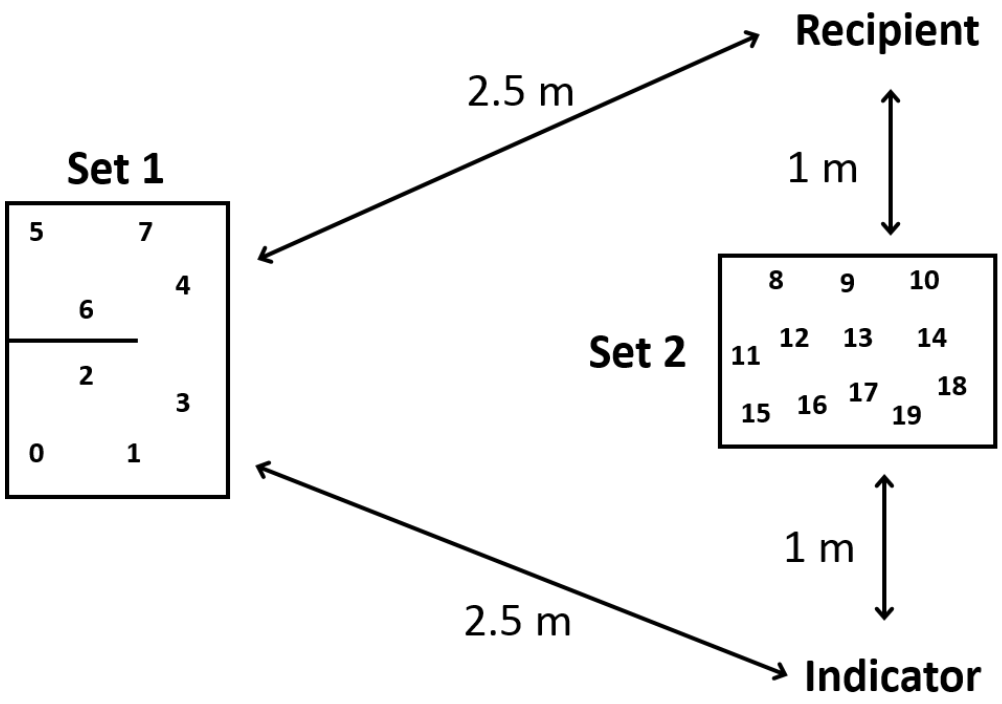

Figure 3.11. The study setup.

To understand how well GARDEN works relative to the three comparison techniques of Section 3.3 (“Laser Pointer”, “Virtual Arrow”, and “Video Share”), we conducted a formal user study. Within the study, we referred to GARDEN as “Sphere Select” to avoid naming bias and simplify the association that participants needed to make between the name and the technique. 22 paid participants were recruited by flyers and mass email requesting that users register as pairs (dyads). All participants were university students (8 female, 19–30 years old, average 23 years old). All subjects used computers daily, and none were members of our lab or had any familiarity with our technique. Ten indicated that they had heard of AR, most from articles in the popular press, although one had taken COMS W4172 (3D User Interfaces and Augmented Reality). All participants were right-handed.

We prepared two sets of physical objects, which were overlaid with a unique numeric identifier floating above each object in each participant’s view (Figure 3.10). We chose objects

with various colors and shapes to make it easier for the indicator to distinguish among the objects. Set 1 contained eight objects, as shown in Figure 3.10(a), and was placed roughly 2.5 meters away from both the indicator and the recipient, so that both participants could see roughly the same sides of the objects, except for those that were blocked by a separating wall placed between objects 2 and 6, as shown in Figure 3.11. Set 2 contained 12 objects, as shown in Figure 3.10(b), and was placed between the indicator and the recipient, roughly 1 meter apart from each, such that the indicator and recipient saw opposite sides of the objects, as shown in Figure 3.11.

The members of each dyad performed the experiment together, alternating roles as indicator and recipient, as described below. We measured effectiveness and accuracy both quantitatively and qualitatively. Quantitative measurement was assessed by computing the average completion time between when the indicator started specifying an object and when the recipient identified an object with the keypad. Quantitative measurement of accuracy was assessed by whether the recipient answered correctly. We also measured the un-answerable rate in which the recipient was not able to make a guess as to the object being indicated. Subjective measurements were obtained through a post-hoc questionnaire.

3.5.1 Pilot Studies

Prior to conducting the formal user study, we performed informal pilot studies with our lab members to refine our techniques and choose a set of hypotheses. From the pilot studies, we determined that 50 cm would be an appropriate distance from the indicator at which to display the virtual representation, to allow her to comfortably point within it. For the recipient, we determined that 80 cm would be an appropriate distance at which to present the virtual representation, so that it occupies less of the recipient's view, making it easier for her to see the animation that interpolates a copy of the representation to its actual location in the environment (Figure 3.5).

3.5.2 Study Description

Our user study was a within-subject, single-session experiment in which we compared each participant's performance on a single task using four referencing techniques. Since our experiment application runs in stereo to provide better depth perception, each participant took (and passed) the Stereo Optical Co. Inc. Stereo Fly Test [StereoOptical, 2015] to screen for stereo vision. Each dyad's session began with the study coordinator showing the two participants a prerecorded video on how to perform selection tasks under the different techniques. Participants were seated and asked not to communicate directly during the study, with one exception: We asked the indicator to verbally confirm that they had selected a target, so that the recipient knew when to determine the selected object. We prohibited verbal communication to isolate effects of the techniques and avoid a possible confound caused by verbal skill differences among pairs of participants. Because the laser pointer beam was difficult for the indicator to see through the head-worn display when it did not land on an object in the set, we allowed the indicator to peek under the head-worn display, if desired, while initiating selection in the Laser Pointer and Video Sharing conditions.

Participants performed a total of 64 tasks (trials), half as indicator and half as recipient, with each trial typically taking around 10–60 seconds. They switched their roles and seats after completing 32 trials. For a given role, a participant initially performed practice trials twice for each of the four techniques, for a total of eight practice trials, to get acclimated to the techniques, followed by three study blocks. Each study block contained four techniques presented twice in a row, for a total of 24 trials across the three blocks. Thus, across both roles, each participant experienced 16 practice trials = 2 (roles) × 4 (techniques) × 2 (trials) and 48 study trials = 2 (roles) × 3 (blocks) × 4 (techniques) × 2 (trials). Technique order within the study blocks was randomized using a Latin square.

At the start of a trial, the technique name ("Laser Pointer", "Virtual Arrow", "Video Share", or "Sphere Select" (the name used for GARDEN)) was displayed to both participants. All names were chosen to minimize naming bias, while remaining sufficiently descriptive to differentiate the techniques. Next, the label of the target object for the indicator to select was highlighted in red in the indicator's view (Figure 3.10a). Nine of the objects were never specified as targets because we determined in our pilot studies that they would be extremely difficult to select from the perspective of the indicator: 0, 5, 6, 8, 9, 10, 11, 13, and 15. However, all objects were labeled for both participants. The object to be selected was chosen randomly with as equal frequency as possible for each object from among the 11 objects used as targets.

During the pilot study, we also found that some participants had problems associating the labels with the correct objects, especially for objects with overlapping projections. To address this, the study coordinator reviewed the mapping of labels to objects during the practice trials until the participants indicated that they were satisfied.

When the recipient entered a number on the numeric keypad (Figure 3.9b), the corresponding label was highlighted in red on their view so that they could confirm that they entered the desired answer before finalizing it. Once the recipient finalized their answer, it was shown to both participants. The time spent displaying the technique name at the start of a trial and the answer at the end of a trial was not counted toward completion time.

Each participant filled out a post-hoc questionnaire (Appendix B.1). The questionnaire was designed to assess the difficulty, effectiveness, and accuracy of the four techniques using Likert-scale questions ranging between 1 (worst) and 5 (best), a request to rank the four techniques with ties allowed, and space for free-form comments.

The total time for the study was approximately one hour. During the study, the software recorded the 3D position and orientation of the head-worn displays, the user inputs for the indicator and the recipient, the completion time, the correct answer, the answer chosen by the recipient, and whether the recipient entered '0' as an answer to indicates that they could not make a guess. The software also recorded the 3D position and orientation of the Kinect and each set of selectable objects, even though these were intended to be stationary during the study.

3.5.3 Hypotheses

Prior to our experiment, we formulated two hypotheses:

- H1: GARDEN will be more accurate than the comparison techniques when referencing objects for which the indicator and recipient do not share a similar perspective (objects in set 2).
- H2: GARDEN will be more accurate than Virtual Arrow when referencing objects for which the indicator and the recipient share a similar perspective (objects in set 1).

We hypothesized that GARDEN would be more accurate than Virtual Arrow for objects in both sets, because virtual-ray–based or 3D-cursor–based pointing geometry may look perfectly intersected with the referenced object from the perspective of the indicator, but confusing or wrong from the perspective of the recipient. This occurs because of differences in the projections and occlusion relationships, even when the perspectives are close. In contrast, GARDEN provides the recipient with a close-up view of the selected region and the indicator's hand so that projection differences or occlusion should not be an issue.

We also expected that GARDEN would perform significantly more accurately than Laser Pointer and Video Share for objects in set 2. In these cases, it is not possible for the recipient to

directly view the laser dot cast on the referenced object in the Laser Pointer technique, and Video Share requires the recipient to map the object designated with the laser in the perspective of the indicator's view to a corresponding object in her own perspective. In contrast, GARDEN provides an animation that aids the recipient in mapping between the referenced object in the close-up view and the corresponding physical object in the environment seen from her own perspective.

Our informal pilot study indicated that GARDEN would take longer to complete the tasks than the other techniques. Therefore, we did not formulate any hypotheses about completion time.

3.6 Analysis

We analyzed a total of 66 study blocks (11 dyads × 2 roles × 3 study blocks) for each of the four techniques, where each block presented each technique twice in a row with different objects to be selected. We excluded data from trials that were interrupted by the participants asking questions during the study (less than 3%). One participant omitted the entire score and ranking section of the questionnaire and two other participants omitted this partially. We used the partial data from these participants in our qualitative analysis, and the entire data for the quantitative analysis since the quantitative data is not affected by these omissions. We analyzed our results according to Likert-scale ratings, subjective ranking, completion time, and accuracy. We used $\alpha = 0.0167$ (0.05/3) after Bonferroni correction as our criterion for statistical significance, since all tests involved three pairwise comparisons.

3.6.1 Quantitative Analysis

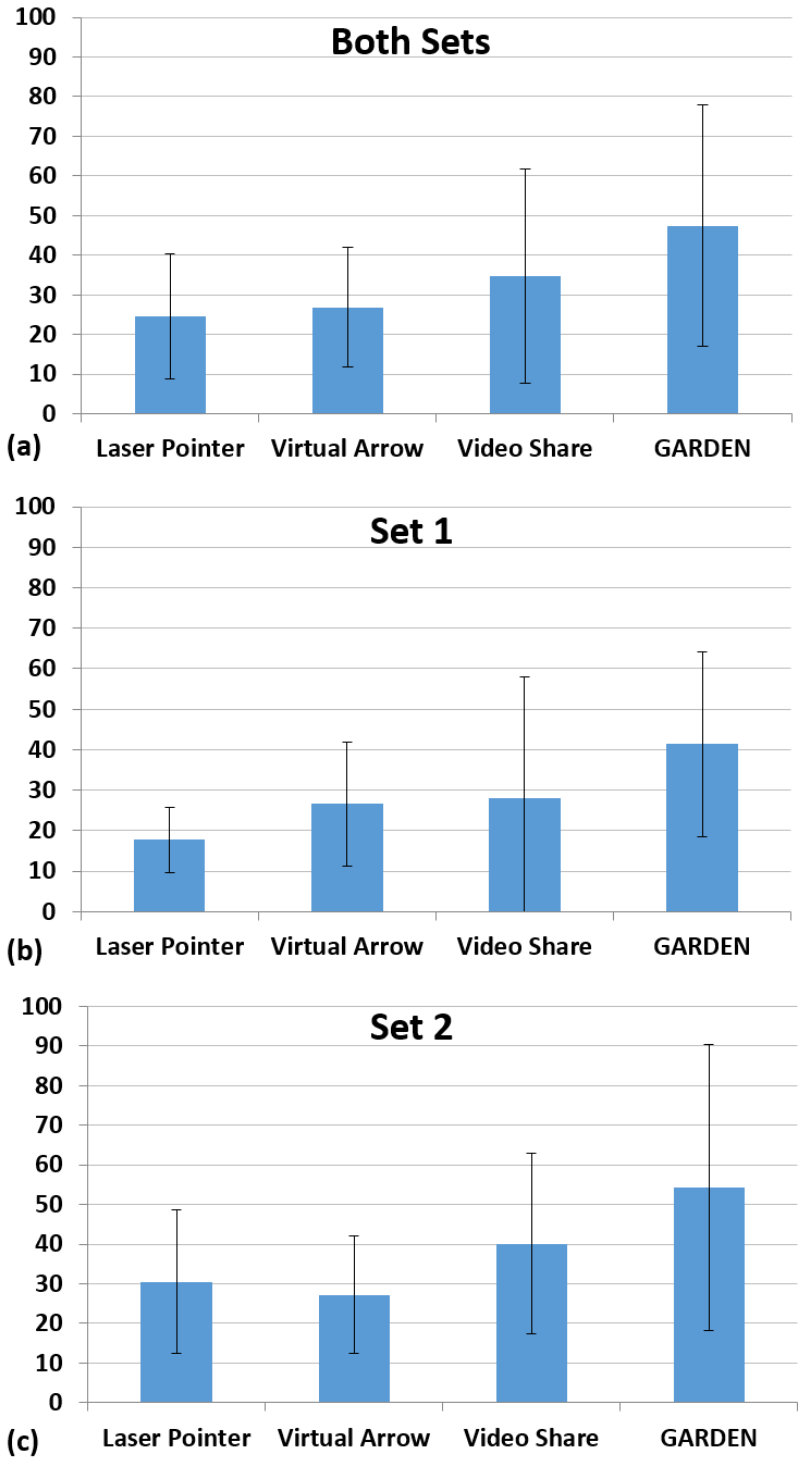

Figure 3.12. Mean completion time in seconds with standard deviation for (a) both sets, (b) set 1 only, and (c) set 2 only. Set 1 contains physical objects for which both participants could see roughly the same side of each object. Set 2 contains objects for which the indicator and recipient see opposite sides of each object.

A within-subjects one-way ANOVA shows that technique had a significant effect ($F_{(3,118)} = 25.331$, $p < 0.001$) on completion time (Figure 3.12a). A paired sample *t*-test between GARDEN and each comparison technique (Laser Pointer: ($t_{(121)} = 7.645$, $p < 0.001$), Virtual Arrow: ($t_{(120)} = 7.298$, $p < 0.001$), Video Share: ($t_{(121)} = 3.210$, $p < 0.01$)) shows that GARDEN takes significantly more time to complete the referencing task. Similar tests analyzed separately for both set 1 (Figure 3.12b) and set 2 (Figure 3.12c) exhibit the same significance result. Since GARDEN requires multiple steps for the indicator to select an object precisely and for the recipient to understand the correlation between the virtual and physical object using the animation, we expected it to take longer than the comparison techniques.

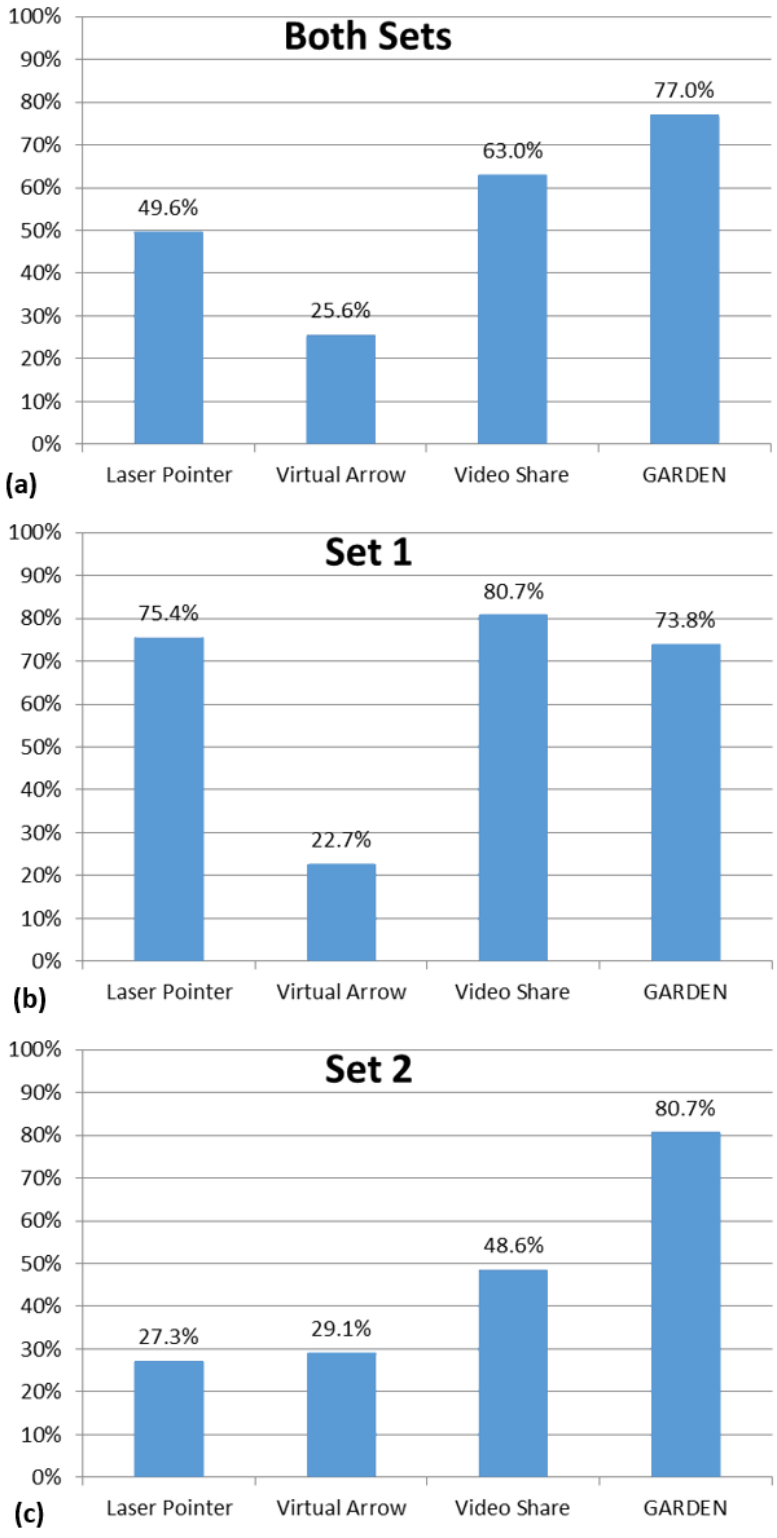

Figure 3.13. Mean accuracy in percentage for physical objects in (a) both sets, (b) set 1 only, and (c) set 2 only. Set 1 contains physical objects for which both participants could see roughly the same side of each object. Set 2 contains objects for which the indicator and recipient see opposite sides of each object.

To analyze selection accuracy (Figure 3.13), we treated the correctness of the recipient's answers as binary data and counted them for each technique. A McNemar's chi-squared test shows that GARDEN was significantly more accurate overall than both Laser Pointer ($\chi^2_{(1,N=122)} = 17.016$, $p < 0.001$) and Virtual Arrow ($\chi^2_{(1,N=121)} = 50.284$, $p < 0.001$), but there was no significant difference between GARDEN and Video Share ($\chi^2_{(1,N=122)} = 3.879$, $p = 0.0489$) (Figure 3.13a). However, if we analyze the data separately for the objects in set 1 (those for which both indicator and recipient share a similar perspective) and those in set 2 (those for which the indicator and recipient have significantly different perspective), a McNemar's chi-squared test shows that for referencing physical objects in set 2 (Figure 3.13c), GARDEN was significantly more accurate than the three comparison techniques (Laser Pointer: ($\chi^2_{(1,N=57)} = 20.024$, $p < 0.001$), Virtual Arrow ($\chi^2_{(1,N=55)} = 21.441$, $p < 0.001$), and Video Share ($\chi^2_{(1,N=57)} = 6.323$, $p = 0.0119$)). This validates H1.

Even though Video Share allowed the recipient to share the perspective of the indicator, several participants mentioned that it was hard to understand the corresponding object in the scene from the indicator's perspective, while GARDEN allowed the recipient to see how the selected virtual object corresponded to the physical environment. A few participants also complained that it was hard to see the red laser dot on a red object on the shared video in Video Share. In contrast, GARDEN is not affected by the colors of objects, as long as the depth camera can detect them. During the study, we observed that some users had problems selecting an object that was mostly blocked by another object from the indicator's perspective, and that the laser pointer was barely hitting the right object. While none of the participants mentioned this, we speculate, that this may have caused the recipient to misinterpret the object being selected. In contrast, with

GARDEN, an indicator could reach a partially visible object captured in 3D, and the recipient could interpret it correctly assisted by depth perception.

A McNemar's chi-squared test shows that for referencing physical objects in set 1 (Figure 3.13b), GARDEN is significantly more accurate than Virtual Arrow ($\chi^2_{(1,N=65)} = 24.750$, $p < 0.001$), but there is no significant difference between GARDEN and Laser Pointer ($\chi^2_{(1,N=57)} = 0.000$, $p > 0.999$), or between GARDEN and Video Share ($\chi^2_{(1,N=57)} = 0.450$, $p = 0.502$). This validates H2.

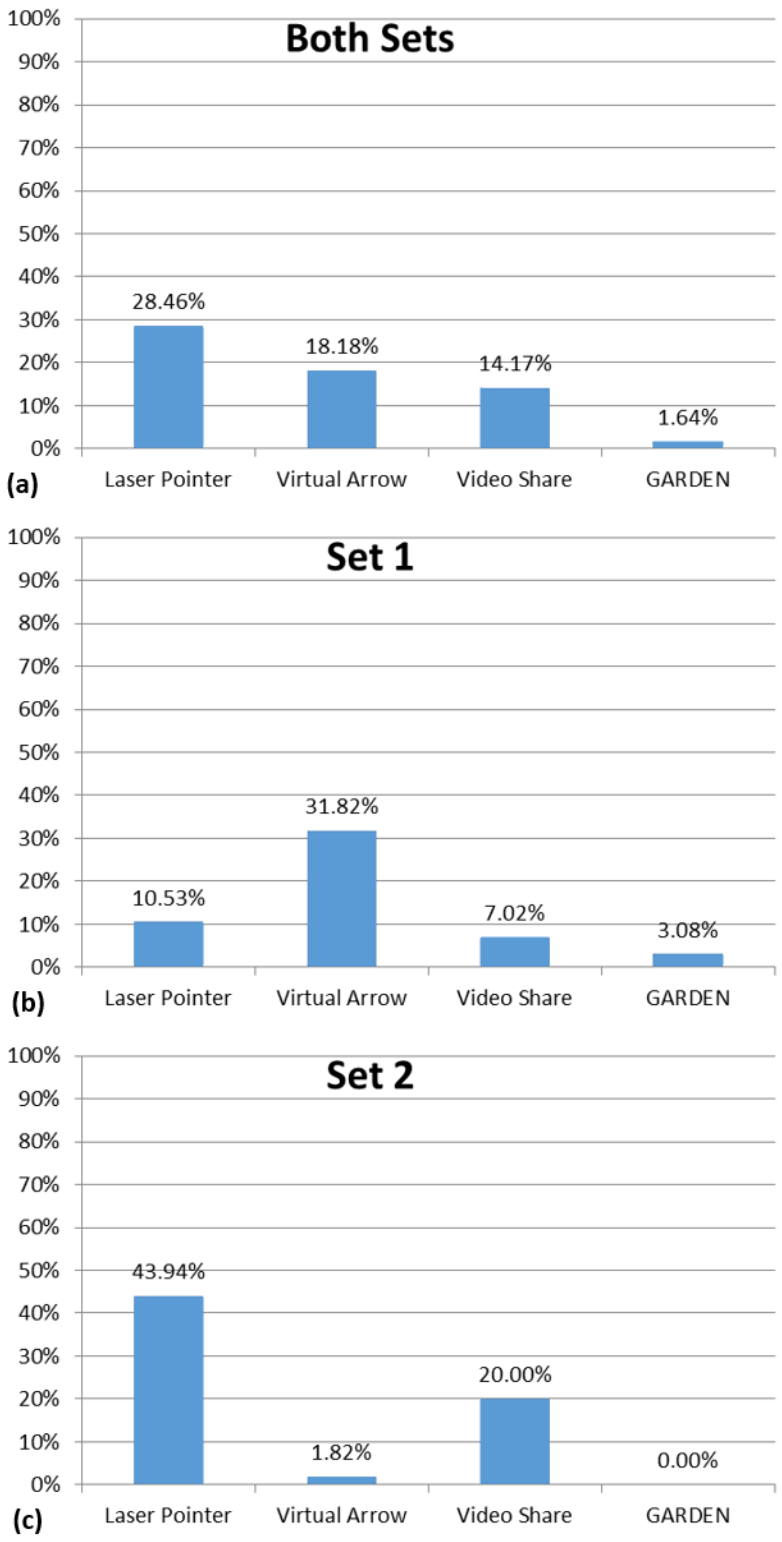

Figure 3.14. Mean unanswerable rate in percentage for (a) both sets, (b) set 1 only, and (c) set 2 only. Set 1 contains physical objects for which both participants could see roughly the same side of each object. Set 2 contains objects for which the indicator and recipient see opposite sides of each object.

In addition to the correctness of the recipient's answer, we recorded the occasions in which the recipient could not even speculate as to which object was referenced. We also treated these occasions as binary data and counted them for each technique. A McNemar's chi-squared test indicated that GARDEN had a significantly lower unanswerable rate overall than the three comparison techniques (Laser Pointer: ($\chi^2_{(1,N=122)} = 27.676$, $p < 0.0001$), Virtual Arrow ($\chi^2_{(1,N=121)} = 16.409$, $p < 0.0001$), and Video Share ($\chi^2_{(1,N=122)} = 9.389$, $p < 0.01$)) (Figure 3.14a). Analyzing the data separately for the objects in set 1 (Figure 3.14b) and those in set 2 (Figure 3.14c) using a McNemar's chi-squared test reveals that: for set 1, GARDEN had a significantly lower unanswerable rate than Virtual Arrow ($\chi^2_{(1,N=65)} = 15.429$, $p < 0.0001$), but there was no significant difference between GARDEN and Laser Pointer ($\chi^2_{(1,N=57)} = 2.286$, $p = 0.131$), or between GARDEN and Video Share ($\chi^2_{(1,N=57)} = 0.800$, $p = 0.371$); and for set 2, GARDEN had a significantly lower unanswerable rate than both Laser Pointer ($\chi^2_{(1,N=57)} = 23.040$, $p < 0.0001$) and Video Share ($\chi^2_{(1,N=57)} = 6.750$, $p < 0.01$), but there was no significant difference between GARDEN and Virtual Arrow ($\chi^2_{(1,N=55)} = 0.500$, $p = 0.480$).

Overall, we believe that GARDEN had a significantly lower unanswerable rate because of the close-up virtual representation and the animation that interpolates a copy of the virtual representation and the indicator's hand seen by the recipient to the actual position and orientation in the physical environment. We believe that the close-up virtual representation eliminated the occlusion and perspective issues and that the animation helped the recipient understand the correlation between the referred object within the virtual representation and the physical object at the actual location. On average, each recipient activated the animation 2.3 times per trial (one time is counted as a round trip from the virtual location to the actual location and from the actual location to the virtual location) with lower bound of 0 times and upper bound of 5 times.

We believe that Virtual Arrow had a significantly higher unanswerable rate than the other three techniques for physical objects in set 1 because the distance between the indicator and the objects in set 1 was relatively far. Consequently there was a projection mismatch (i.e., the target appeared to be properly selected from the perspective of the indicator, but looked wrong or ambiguous from the perspective of the recipient). This resulted from inaccuracies in the depth map, causing the arrow to terminate at a point that was not on the surface of the target. In contrast, the laser pointer (used for both Laser Pointer and Video Share techniques) always casts its beam on the objects so there were fewer occasions in which the recipient could not make a guess, except for those occasions in which the laser dot appears on the occluded sides of the objects.

In contrast, we believe that Laser Pointer and Video Share had a significantly higher unanswerable rate than Virtual Arrow and GARDEN for physical objects in set 2 because there were more occasions in which the laser beam was cast on the occluded sides of the objects. We also believe that Virtual Arrow had a lower unanswerable rate for set 2 than for set 1 because the distance between the indicator and the objects in set 2 was relatively close, leading to less projection mismatch.

3.6.2 Subjective Analysis

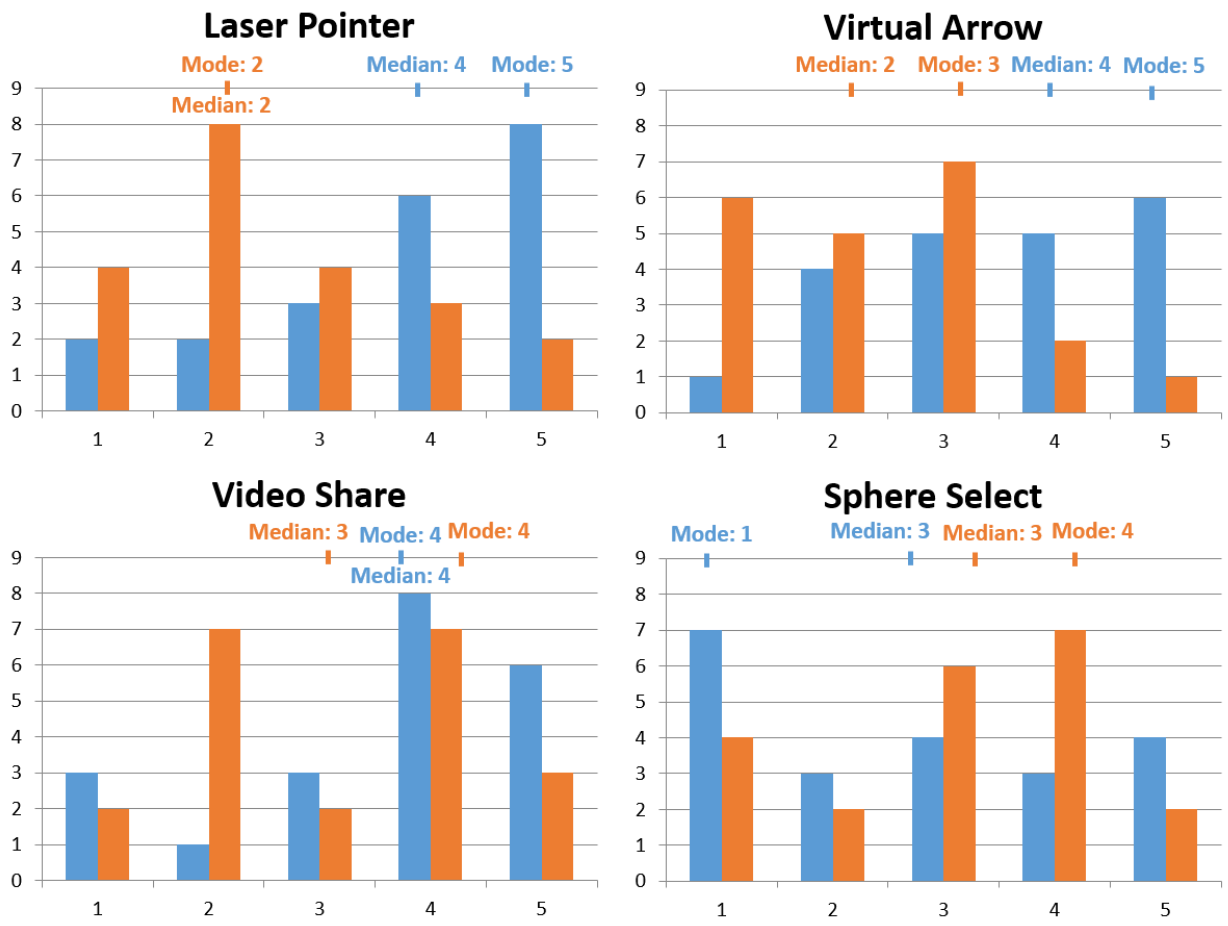

Figure 3.15. Histograms of ease of selection (blue bar on the left of each pair) and interpretation (orange bar on the right of each pair). Likert-scale ratings from 1–5 given by the participants for the four techniques with mode and median. 5 is best.

The Likert-scale results from the questionnaire show that participants found Laser Pointer and Virtual Arrow to be very easy for selection, but difficult for interpretation (Figure 3.15). One user commented that the laser beam was impossible to see when it appeared on an occluded side of an object, and the virtual arrow was sometimes pointed at an inappropriate target, such as somewhere in the air, even though it may have looked correct from the perspective of the indicator. Participants found that Video Share was easy for both selection and interpretation tasks, and GARDEN was difficult for selection, but easy for interpretation.

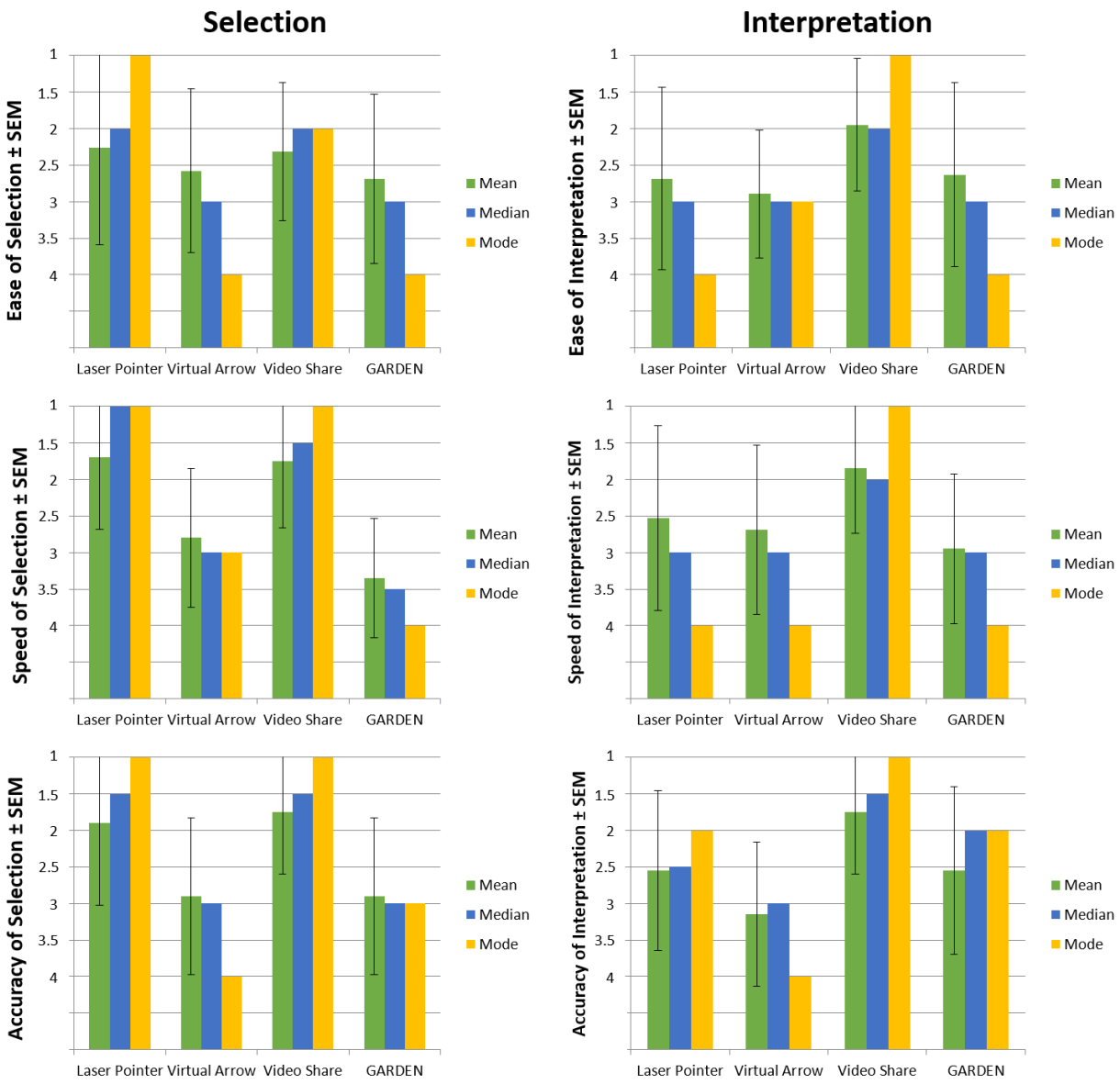

Figure 3.16. Ranked qualitative ease, speed, and accuracy of selection (left columns) and interpretation (right columns) with SEM (Standard Error of the Mean) for the four techniques. 1 is best.

A Friedman test shows that the distribution of the rankings for the perceived speed of selection ($\chi^2_{(3)} = 24.77$, $p < 0.001$), accuracy of selection ($\chi^2_{(3)} = 16.86$, $p < 0.001$), and accuracy of interpretation ($\chi^2_{(3)} = 11.71$, $p < 0.01$) between the techniques is significant, while the perceived ease of selection, ease of interpretation, and speed of interpretation was not significant (Figure 3.16). A Wilcoxon test indicates that GARDEN was perceived to be significantly slower for se-

lecting the target object compared to Laser Pointer ($Z = -3.27$, $p < 0.001$) and Video Share ($Z = -3.29$, $p < 0.001$), but there was no significant difference between GARDEN and Virtual Arrow.

A Wilcoxon test also indicates that GARDEN was perceived to be significantly less accurate for selection compared to Video Share ($Z = -2.9$, $p < 0.01$), but there was no significant difference compared to Laser Pointer ($Z = -2.2$, $p = 0.028$) or Virtual Arrow ($Z = -0.254$, $p = 0.80$). Even though the quantitative analysis (Section 3.6.1) showed that GARDEN was significantly more accurate than all three comparison techniques when referencing an object for which the participants do not share a similar perspective, several participants mentioned that they were not sure about whether their finger was pointing properly to the virtual representation. This issue may be due to the fact that most participants had never previously tried an AR application using a head-worn display. A Wilcoxon test shows that there is no significant difference between GARDEN and the other three techniques for perceived accuracy for interpretation (Laser Pointer: ($Z = -0.077$, $p = 0.939$), Virtual Arrow: ($Z = -1.590$, $p = 0.112$), Video Share: ($Z = -1.987$, $p = 0.047$)).

3.7 Discussion

The study showed that GARDEN is significantly more accurate than one technique (Virtual Arrow) for referencing physical objects for which the indicator and the recipient share similar perspectives, and is significantly more accurate than all the comparison techniques for referencing physical objects for which the indicator and the recipient have very different perspectives. Overall, our results suggest the following recommendations:

1. Laser Pointer would be better when users share a similar view, because Laser Pointer takes less time while providing accuracy comparable to that of Video Share and GARDEN.

2. GARDEN would be better when users do not share a similar view and accuracy is more important than speed, because GARDEN performs significantly more accurately than the comparison techniques even though it takes longer.

Since GARDEN requires multiple steps for the indicator to select precisely and for the recipient to understand the correlation between the virtual and physical object using the animation, we expected it to take longer than the comparison techniques, and the study showed that GARDEN indeed took significantly more time to complete the referencing tasks than the comparison techniques. We also speculate that the low resolution, noisy reconstructions we used caused recipients to take longer than necessary to reach an answer. However, we would like to emphasize that spending more time on referencing is not the cause of the improved accuracy for our technique, since there were significantly more occasions for the comparison techniques in which the recipient could not figure out the referred object at all. Thus, we argue that our technique itself improved the accuracy, rather than the longer referencing time.

We believe that the animation used in GARDEN helped the recipients make a guess as to what is being referenced when the correlation between the pointed target within the virtual representation and the physical object at the actual location is ambiguous, and as a result, it led to better accuracy overall. However, there was no correlation between the number of times the animation was activated and the accuracy. We believe that this is due to the fact that if the correlation between the target pointed within the virtual representation and the physical object at the actual location was clear after activating the animation once, the recipient could make a decision without needing to activate it additional times.

Chapter 4. A 3D Referencing Technique for Supporting Remote Task Assistance in Augmented Reality

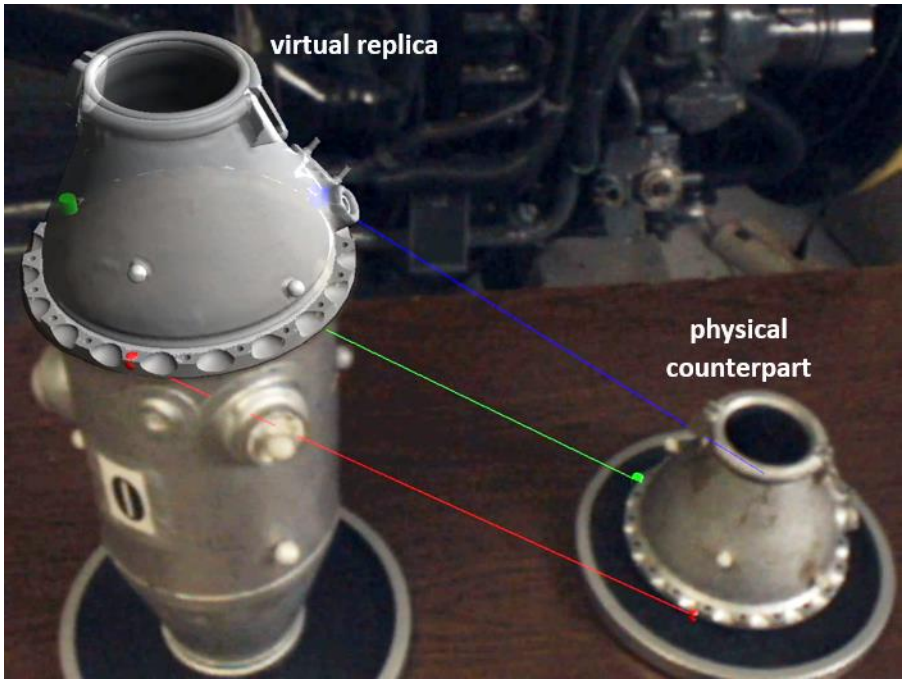

Figure 4.1. DEMO3D interaction technique allows a remote expert to guide a local user to place the top of an aircraft engine combustion chamber relative to its bottom by interacting with a *virtual replica* of the top. As seen by the expert in AR, looking through the local user's cameras, the expert has placed the transparent virtual replica on the chamber bottom. Pairs of metaobject annotations on the virtual replica and physical chamber top (its physical counterpart) are linked with color-coded rubberband lines to show the correspondence between the physical top and its virtual replica, and are also seen by the local user.

Task guidance has been an active topic in the field of AR, with applications to a wide range of domains, including the operation, assembly, maintenance, and repair of equipment (e.g., [Bottecchia et al., 2010; Gauglitz et al., 2014; Iyoda et al., 2008; Stevenson et al., 2008; Tecchia et al., 2012]). Seeing instructional graphics overlaid directly on the actual task environment can significantly improve a user's understanding compared to viewing instructions displayed on a nearby monitor or in a paper manual [Henderson and Feiner, 2011]. In many complex tasks, a

remote expert may need to assist a local user to guide actions on objects in the local user's environment. However, effective spatial referencing and action demonstration in a remote physical environment can be challenging [Chastine, 2007]. Traditional approaches to remote guidance using voice or video limit how the expert can instruct the local user. Language describing spatial locations and actions in space is frequently ambiguous or vague [Talmy, 2003], leading to confusion and error (e.g., [Heiser et al., 2004]). In contrast, AR enables a remote expert to directly interact with the local environment for 3D spatial referencing and action demonstration and allows a local user to visualize instructions directly overlaid on the environment.

Existing literature has explored approaches to enable remote experts to present instruction elements such as 3D arrows [Bottecchia et al., 2010; Chastine et al., 2008], to perform hand gestures [Stafford and Piekarski, 2008; Tecchia et al., 2012], and to place annotations such as 3D tags or sketches on physical objects [Adcock et al., 2014; Gauglitz et al., 2014; Kim et al., 2014; Lanir et al., 2013]. However, it can be challenging or even impossible for a remote expert to refer to a part of physical object in the local user's environment that is occluded or to demonstrate actions on it.

In this chapter, we explore a 3D interaction and visualization technique for tasks that require more than simple pointing or annotation. We developed DEMO3D, an approach that allows a remote expert to create a *virtual replica* of its physical counterpart present in the local user's environment as shown in Figure 4.1 [Oda et al., 2015]. The virtual replica can be used to demonstrate how to position and orient the physical counterpart (Figure 4.1). However, it can be time-consuming and error-prone to position and orient the virtual replica precisely due to interpenetration between the virtual replica and other objects in the local user's environment. To alleviate this problem, we use constraints to allow fast and precise placement of the virtual replica.

We also display a set of metaobjects on a virtual replica and its physical counterpart, as shown in Figure 4.1, to help the local user mentally map the position and orientation of the virtual replica to its physical counterpart.

In addition, we provide the remote expert with two different perspectives on the local environment within which to interact, similar to Tatzgern and colleagues, who allow the user to switch between a live video and synthesized views of a reconstructed physical scene [Tatzgern et al., 2014]. Our first perspective is a virtual representation that uses virtual models of the 6DOF-tracked physical objects. The expert can freely navigate among and interact with these virtual models, whose poses can be efficiently transmitted. However, some virtual models may not correspond exactly to their physical counterparts because of damage or modification. In addition, it is sometimes necessary for the expert to be able to see exactly what is taking place in the local user's environment, from the perspective of the local user. To address these situations, we provide a second perspective that allows the expert to see and interact from the local user's live or frozen stereo camera views.

We compared DEMO3D with two other approaches for a 6DOF alignment task: POINT3D and SKETCH2D. POINT3D allows a user to point at corresponding pairs of locations on a pair of objects and SKETCH2D supports 2D sketch-based annotation using a tablet. Our comparison examined speed of performance, ease of interaction by the remote expert, ease of interpretation by the local user, and preference. Our user study shows that DEMO3D was faster than both other approaches and POINT3D was faster than SKETCH2D with a highly trained expert. Experts generally felt that they performed faster and better with DEMO3D than with SKETCH2D. Furthermore, local users preferred DEMO3D to SKETCH2D.

This research was joint with colleagues Carmine Elvezio and Mengü Sükan. I conceived of the idea of using virtual replicas to demonstrate relative poses (DEMO3D) and the use of constraints, and I proposed the user study setting in which the user places a chamber top relative to a base in one of a set of poses. I performed much of the implementation using the Goblin XNA infrastructure I developed, and I authored the majority of the paper on the research [Oda et al., 2015]. My colleagues suggested POINT3D as an alternative to DEMO3D that also uses virtual replicas, developed the final configuration of the base fixture used in the study, assisted with implementation, conducted and analyzed the study, and assisted with writing the paper.

In this chapter, we first discuss related work in Section 4.1. Then, in Sections 4.2–4.4, we describe the concepts of our approaches, a comparison technique, the expert's interaction environments, and our specific implementations. Next, in Sections 4.5–4.6, we present the pilot and formal studies. Finally, in Section 4.7–4.8, we discuss our study results.

4.1 Related Work

Many researchers have explored approaches for supporting remote task guidance. Kuzuoka showed that a remote expert can use their finger to indicate regions of interest in the video sent from a local user, with the composite imagery of the finger on the video sent back to the local user's display [Kuzuoka, 1992]. Bauer and colleagues extended this work to AR by presenting a 2D mouse cursor operated by the expert on the local user's head-worn display [Bauer et al., 1999]. However, the 2D pointer becomes outdated if the local user moves. To address this, Bottecchia and colleagues allowed an expert to take a snapshot of the local user's view, and perform 2D interactions with the still image [Bottecchia et al., 2010]. However, pointing or gesturing on the local user's 2D view plane has limitations for conveying complex 3D actions. To allow demonstrations directly in the 3D environment, Kirk and Fraser embedded the gestural out-

put of an expert captured from an overhead video camera on a 3D plane where instructions are given in the local user's environment [Kirk and Fraser, 2005], while Goto and colleagues presented a similar technique using prerecorded video [Goto et al., 2010]. While these techniques may work well for tasks performed on a flat surface, they are not well suited for fully 3D tasks.

Chastine and colleagues allow a remote expert to manipulate a 3D virtual arrow on a local user's view [Chastine et al., 2008]; however, it is difficult and time-consuming to align the arrow. Bottecchia and colleagues make it possible for an expert to insert a precomputed 3D virtual animation into a local user's view to demonstrate how to perform a task [Bottecchia et al., 2010]; however, a precomputed animation is not flexible enough to document actions that change based on the situation. Stafford and colleagues devised a more flexible mechanism allowing an expert to reference a point of interest at a remote site [Stafford and Piekarski, 2008]. An indoor expert sees a virtual overhead view of a local user's world on a tabletop display and can point at the display directly to refer to a point of interest. This creates a virtual 3D representation of the indoor expert's hand from a texture-mapped visual hull determined by a set of cameras surrounding the tabletop. The representation of the hand is overlaid on the view of an outdoor user.

Later, Tecchia and colleagues use depth sensors to dynamically capture both the local user's environment and expert's hands [Tecchia et al., 2012]. The 3D scene of the local user and the 3D hands of the expert are then merged and presented to both sides, allowing the expert to provide gestural instructions directly in the local user's environment. Sodhi and colleagues applied this approach to a portable system [Sodhi et al., 2013]. Although it can be easy to demonstrate simple actions with these techniques, it can be challenging to show how to precisely ma-

neuver physical objects in the local user's environment: while the expert's hand gestures are captured, the expert cannot manipulate the objects.

Kurata and colleagues developed a system in which a local user wears a camera and a laser pointer on the shoulder [Kurata et al., 2004]. The remote expert sees the task space through the camera on the local user's shoulder, and references a point of interest in the physical environment with the laser. Sakata and colleagues built on this work by using a chest-worn display and the capability for the expert to add line drawings on a still image captured from the local user's view [Sakata et al., 2006]. Lanir and colleagues further extended this to a portable projector situated in the local user's environment, allowing an expert to add projected 2D drawings to the local user's physical environment [Lanir et al., 2013]. A video camera and a pico-projector are mounted at the end of a remotely-operated robotic arm so that they can be moved by an expert, who can interact with the live video feedback. In addition to sketching, Adcock and colleagues allow the expert to manipulate preconstructed 2D proxies of physical objects in the local user's environment [Adcock et al., 2014]. The expert manipulates these proxies on a multi-touch display with gestures specifying 2D translations and rotations projected onto the local user's environment with a ceiling-mounted projector. However, projection-based approaches have limitations on pointing or drawing on complex surfaces and surfaces that are not directly projected.

To overcome these limitations, Kim and colleagues presented a system that allows the remote expert to place annotations anywhere on the local user's physical environment constructed dynamically with a SLAM (Simultaneous Localization and Mapping) algorithm [Kim et al., 2014]. Gauglitz and colleagues supplemented this technique by allowing the expert to freely navigate in the constructed scene to place annotations from a separate viewpoint in the local user's perspective [Gauglitz et al., 2014]. The constructed scene is updated dynamically to reflect the

changes in the local user's environment. Although this technique would suffice for most guidance tasks, simple annotations on the physical environment cannot easily demonstrate actions such as 6DOF alignment, especially on complex surfaces.

Of the preceding approaches, only Adcock and colleagues [Adcock et al., 2014] allow the remote expert to manipulate proxies, albeit entirely in 2D. Tait and Billinghurst extend this so that an expert with a monoscopic desktop interface uses 2D input devices to specify a target layout for the local user by placing existing virtual copies of physical objects into a 3D model of the environment seen on the local user's head-worn display [Tait and Billinghurst, 2014]. In contrast, we wanted to allow the expert to view the local user's environment in a stereo head-worn display, while directly creating and manipulating, in 3D, replicas of tracked physical objects to demonstrate actions in the local user's environment.

4.2 Our Approach

Our approach is based on the way an expert guides a novice when co-located, by demonstrating appropriate actions. In our approaches, the remote expert can create, annotate, and manipulate in 3D virtual replicas of physical counterparts in the local user's environment. This is inspired by voodoo dolls interaction [Pierce et al., 1999], in which a user creates copies of existing virtual objects. The user can perform operations using the copies, which affect the original virtual objects. We employ a similar concept, adapted to multi-user remote assistance for ease of interaction by the expert and ease of interpretation by the local user.

As presented here, we assume that the physical objects with which the local user interacts have been modeled and are tracked in 6DOF. Even though it is possible to dynamically construct and track the local user's environment on the fly [Gauglitz et al., 2014; Newcombe et al., 2011; Tatzgern et al., 2014], this can require a wide range of perspectives, obtained while the task ob-

jects are being manipulated, to allow proper segmentation. Additionally, parts not already in the local user's environment cannot be modeled, preventing the remote expert from referring to them. Thus, our implementation assumes the existence of 3D models of the necessary parts and a suitable technology to track them. However, some of these constraints can be lifted, as we describe later in Section 6.3.6.

We assume that the expert's environment contains a virtual proxy for each relevant physical object in the local user's environment and that the position and orientation of each virtual proxy is determined by the corresponding position and orientation of the physical object in the local user's environment. The expert can create a virtual replica of a physical object by grabbing its virtual proxy. The virtual replica can then be directly manipulated in 6DOF, remaining where the grab is released, and grabbed again to manipulate it further. The virtual replicas are also displayed to the local user in context of their physical counterparts.

We developed DEMO3D, an approach that takes advantage of virtual replicas. We tested using this approach to guide placement of a physical object *A* relative to another physical object *B*, a task found in many domains using remote assistance, including assembly, maintenance and repair.

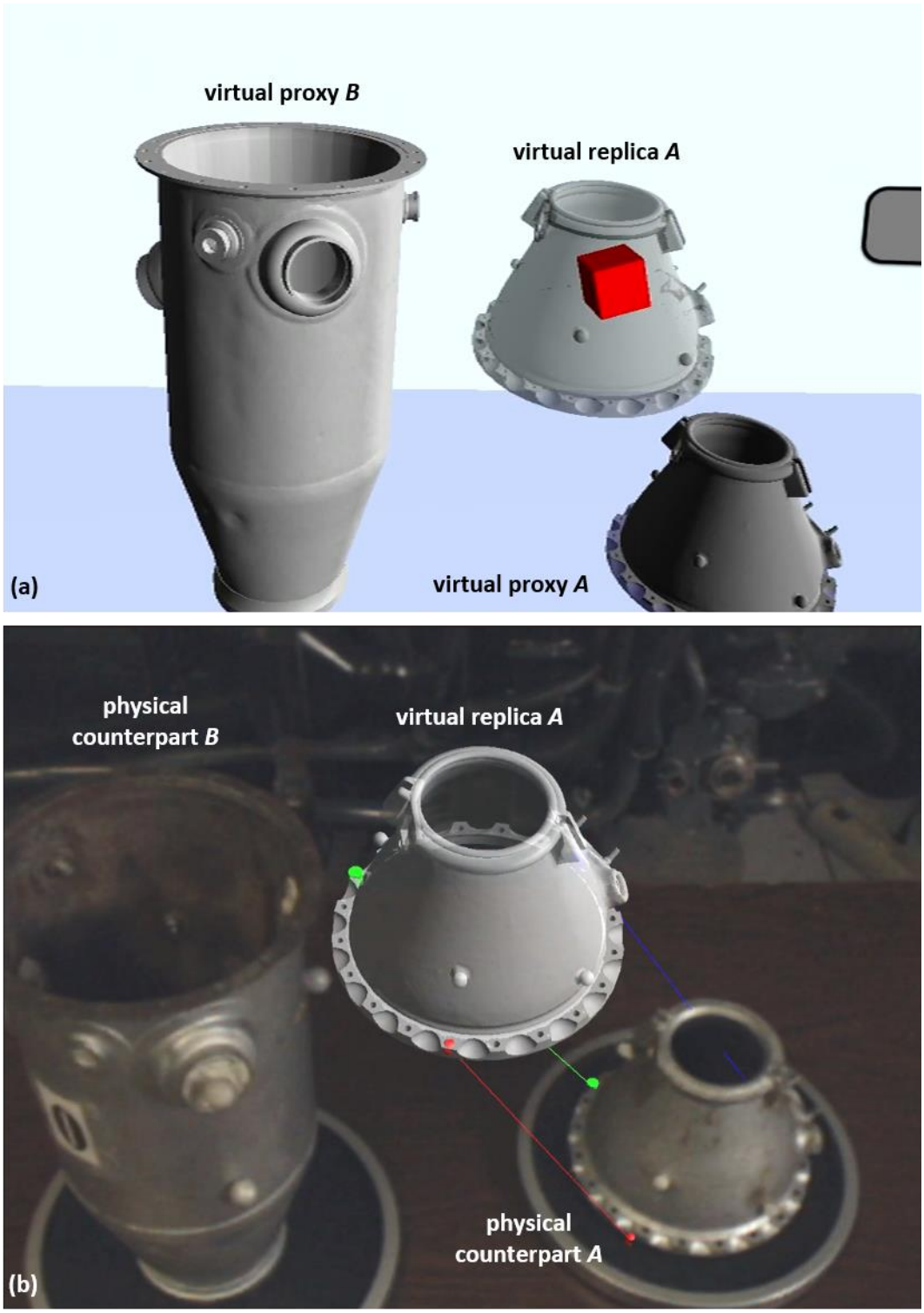

Figure 4.2. DEMO3D: (a) The expert manipulates the virtual replica to demonstrate how to place the top chamber on the bottom chamber with a specific orientation. (b) An AR view seen by the local user through a head-worn display, in which pairs of metaobject annotations on the virtual replica and its physical counterpart are connected with color-coded rubberband lines.

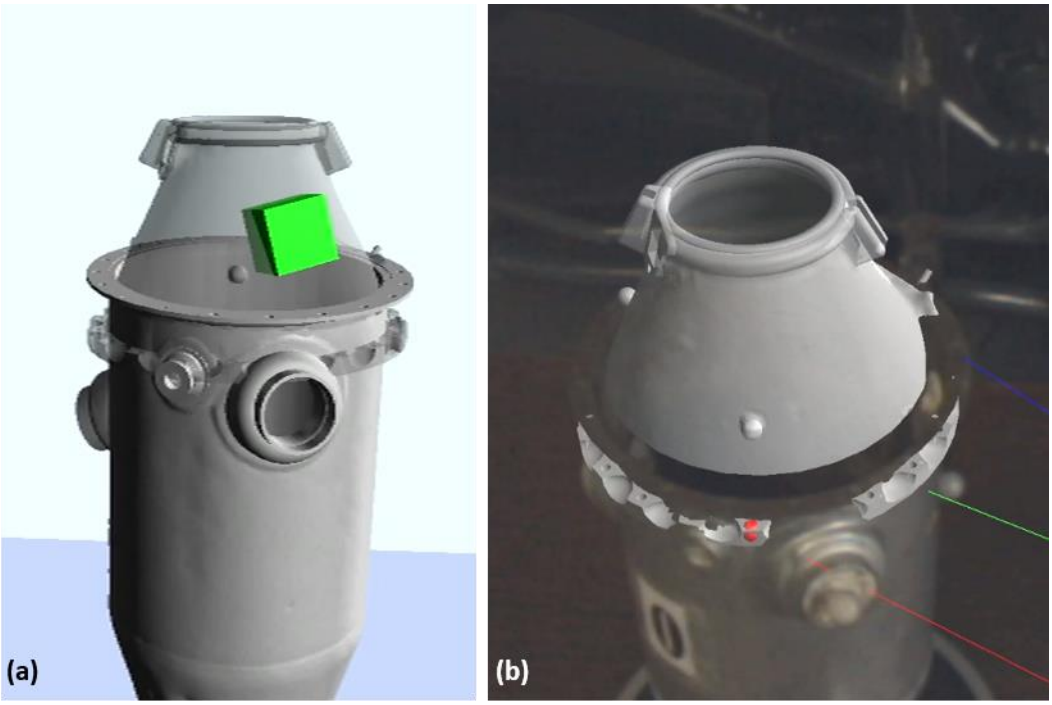

Figure 4.3. Constraints. (a) When the expert places and fine-tunes the chamber top relative to the bottom, the two can interpenetrate or not fit properly, as shown here, because there is no force feedback. (b) This can potentially mislead the local user.

4.2.1 DEMO3D

DEMO3D allows the remote expert to use a virtual replica to directly show how to achieve the final 6DOF pose of *A* relative to *B* in the local user's environment as shown in Figure 4.2. For example, when assembling furniture for a physically co-located expert, this approach would be equivalent to the expert actually picking up a part, physically aligning it relative to another part to show the local user how the parts fit, and then placing it back, so that the user can imitate the demonstrated action. One issue that arises in our case is that the virtual replica of *A* can interpenetrate *B* (Figure 4.3a), which can potentially cause misinterpretation (Figure 4.3b). Prior work has used a physics simulation to avoid interpenetration and allow realistic interaction [Piumsomboon et al., 2014]. However, this can be computationally expensive and even cause

unwanted repulsion to avoid interpenetration, as well as sliding and dropping in response to simulated gravity.

To address this, we employ a constraint-based approach similar to that of Adcock and colleagues [Adcock et al., 2014], who constrain physical objects to flat surfaces. However, we use constraints that can be more flexible and could be specified by the expert prior to giving guidance. Although the constraints used in our study were hard-coded, we describe how they might be specified by the remote expert in Section 6.3.5.

Once the remote expert has specified the final 6DOF pose of the virtual replica, the local user must determine how to place the physical counterpart to match the pose. This raises another issue: the local user may have difficulty understanding how to match the physical object to the virtual replica because of difficulties performing the mental rotation. Considerable research has shown that mental rotation is difficult to imagine and that physical rotation facilitates mental rotation (e.g., [Wexler et al., 1998]). Matching can also be difficult if the physical counterpart does not have enough prominent geometric features. To address this, we added a set of landmark metaobjects (Figure 4.2b) on the virtual replica, which are duplicated on its physical counterpart, with a connecting rubberband line between each metaobject and its duplicate to further simplify the matching process. The local user can use these metaobjects as cues to match the 6DOF pose of the physical counterpart with the virtual replica placed by the expert.

Furthermore, we fade out any virtual replica as its physical counterpart gets within a preset threshold distance to the virtual replica placed by the expert, while maintaining the visibility of the metaobjects. We decided to fade out just the virtual replica, but not the metaobjects, since there is general difficulty seeing one relative to the other when two objects are close and *z*-buffer conflicts can occur as the physical counterpart overlaps with its virtual replica, making it visually

confusing for the local user to fine-tune the final 6DOF pose. The appearance of the metaobjects themselves is intended to be sufficient for the local user to match the pose between the physical counterpart and its virtual replica when they are sufficiently near each other.

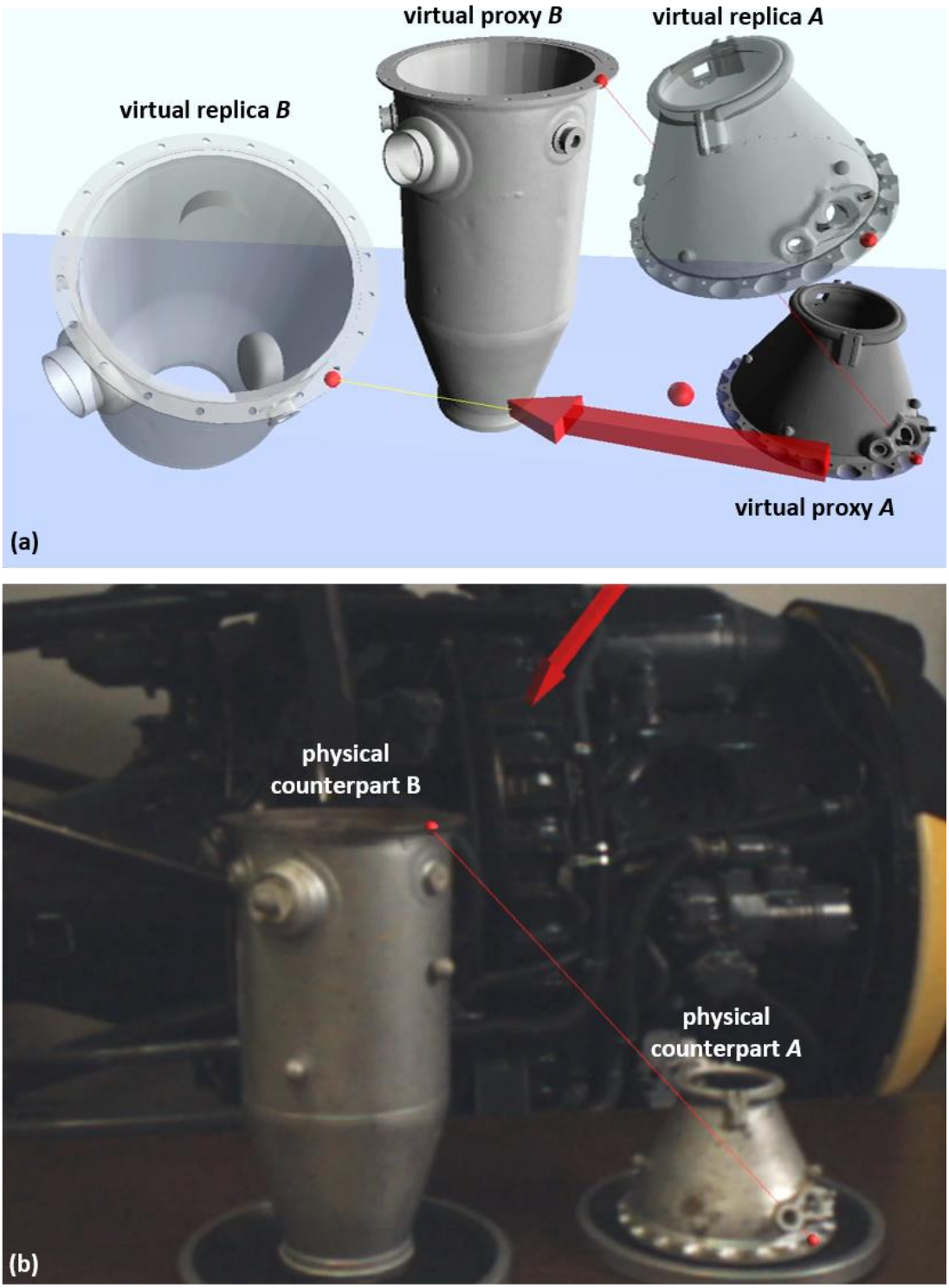

Figure 4.4. POINT3D: (a) Once the expert specifies corresponding contact points (red spheres) on both objects, a color-coded rubberband line connecting the points appears between the proxies. (b) An AR view seen by the local user through a video–see-through head-worn display, in which the color-coded rubberband line appears between the physical counterparts.

4.2.2 POINT3D (Comparison Technique)

Another way in which a physically co-located expert can guide a worker through an assembly task is by pointing to prominent landmarks on the objects and instructing the user to align them. Continuing our furniture assembly example, an expert might point to a peg on one part and a hole on another part, and say "That peg needs to be inserted into this hole." Specifying three pairs of contact points would allow the expert to convey the 6DOF pose of *A* relative to *B.* In our implementation, the remote expert manipulates a tracked pointing device that controls a ray whose intersection with an object defines a contact point. After optionally creating one or more virtual replicas, the expert can point to a location on either a virtual replica or its virtual proxy to place annotations anywhere on the corresponding physical object as shown in Figure 4.4. The expert can freely manipulate a virtual replica to find a suitable pose from which to place an annotation. Annotations appear on both the virtual proxy and the virtual replicas in the expert's view (Figure 4.4a) and on the physical object in the local user's view (Figure 4.4b). Since all annotations are attached to their objects, they are updated interactively as the objects are manipulated. Once annotations for corresponding points have been placed on both *A* and *B*, a "rubberband" line will appear between corresponding annotations on both objects to help the local user identify which annotated points should be aligned (Figure 4.4).

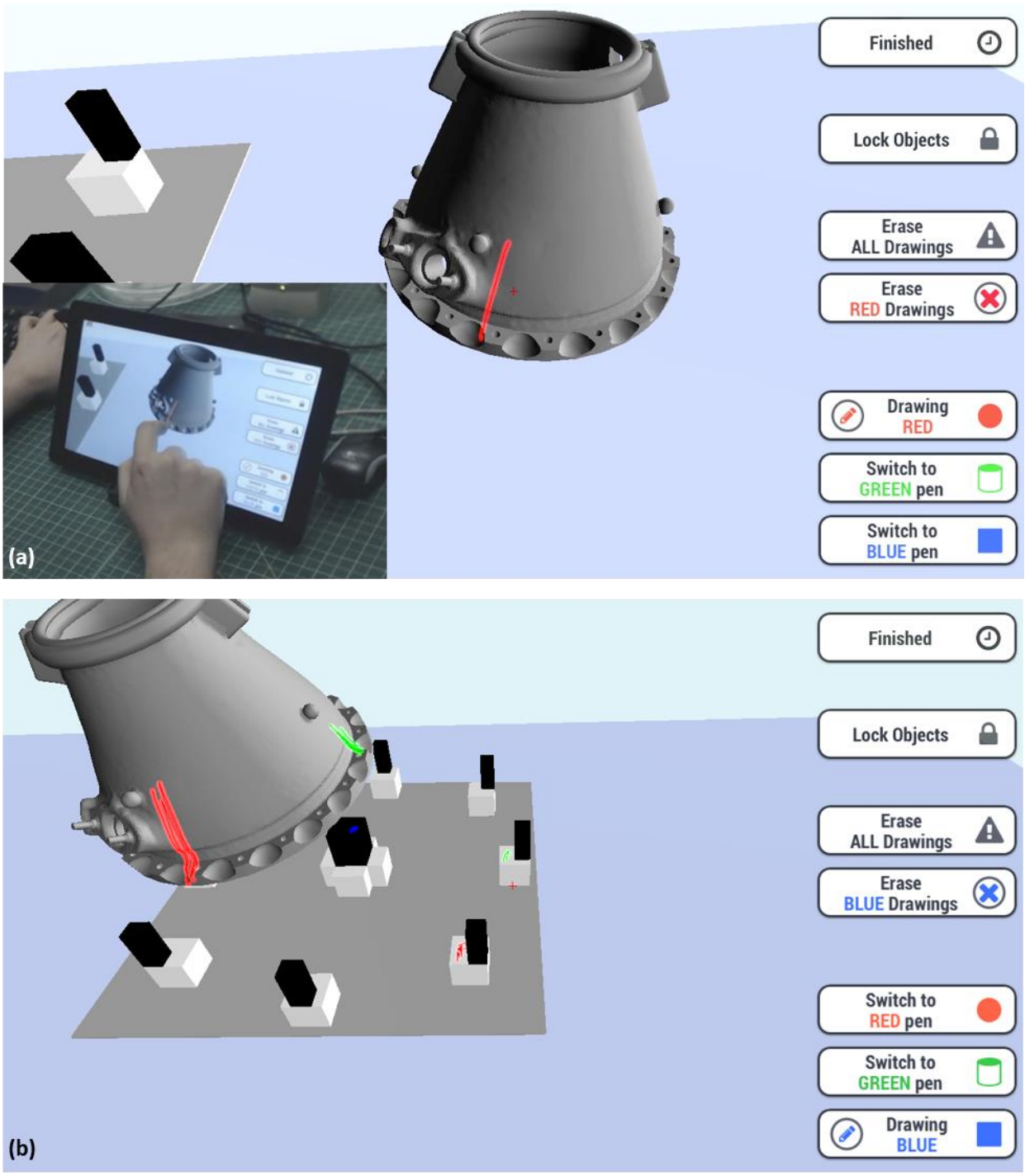

Figure 4.5. SKETCH2D. Screenshots from the application. (a) The expert sketches a red line to indicate one of the contact points on the chamber top with multi-touch interaction. (Inset shows third-person view.) (b) The expert has sketched all three corresponding contact points to indicate how the chamber top should align with a Lego fixture.

4.2.3 SKETCH2D (Comparison Technique)

As described previously, several recent remote task guidance systems provide the remote expert with a 2D annotation system [Adcock et al., 2014; Gauglitz et al., 2014; Kim et al., 2014; Lanir et al., 2013] based on a multi-touch tablet or PC, in some cases with a projector in the local user's environment to project the expert's instructions onto the environment [Adcock et al., 2014; Kim et al., 2014; Lanir et al., 2013]. We developed a similar system that allows sketch-based 2D annotations on a multi-touch tablet for the expert as shown in Figure 4.5, and displays the annotations on the local user's environment through a head-worn display. The expert uses multi-finger gestures to navigate among and draw annotations on virtual proxies. Each point sketched on the tablet screen is projected from the center of projection onto the closest point on the surface of the proxy object visible at that pixel, such that the sketches appear in 3D on the surfaces of the proxies.

4.3 Interaction Environment

For both the head-worn display and the tablet, the remote expert can transition seamlessly between two views without affecting what the local user sees: a VR view (Figures 4.2a, 4.3a, 4.4a, and 4.5) and an AR view (Figures 4.1, 4.2b, 4.3b, and 4.4b). In both views, the expert can grab (but not move) the virtual proxies to create virtual replicas, where the manipulation of the virtual replicas is presented to the local user. In the VR view, the expert is presented with a virtual environment, viewed from their own perspective, that includes virtual proxies of important tracked objects from the local user's environment, whose position and orientation are updated in real time as the local user manipulates them. Communicating geometric transformations, rather than video, between sites, minimizes network latency and avoids restricting the expert to views seen by the local user.

In the AR view, the expert interacts with camera imagery captured from the perspective of the local user. This can be useful when the expert wants to see the task environment from the local user's point of view, and when the task environment may contain unmodeled objects or objects that might not correspond to their models (e.g., if the objects have been damaged). The expert can also freeze the view (for the expert only) [Bottecchia et al., 2010] to support interaction with the physical objects at their positions and orientations when the view was frozen. This is especially useful when the local user is moving their head or a physical object in a way that confuses the expert.

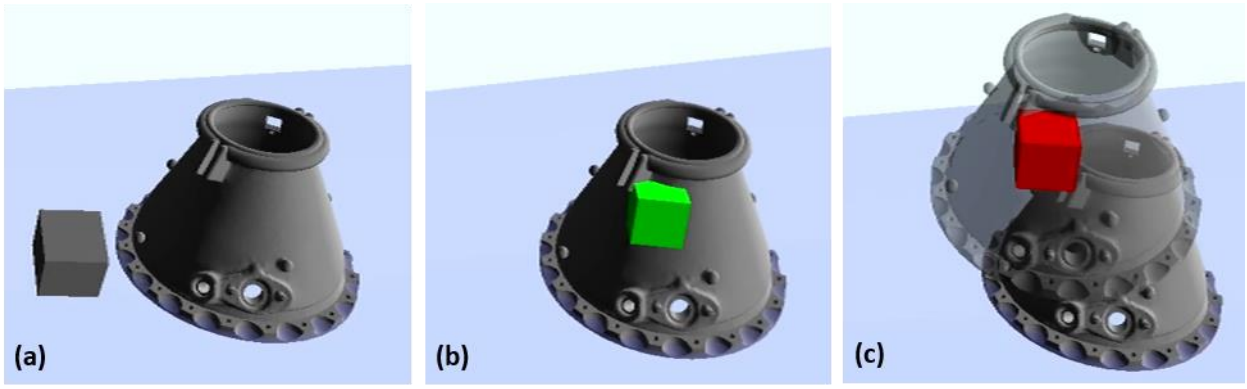

Figure 4.6. A representation of the manipulation device in cubic box rendered with different colors. The box is rendered in (a) *gray* when not intersected with a virtual proxy, (b) *green* when intersected, (c) and *red* when grabbing a virtual replica.

4.4 Implementation

Our experimental software is built using Goblin XNA [Oda and Feiner, 2012] (Appendix A). The remote expert creates the virtual replica of a virtual proxy by intersecting a 6DOF-tracked manipulation device with the proxy and pressing a button. The manipulation device is rendered as a cube in the VR environment as shown in Figure 4.6. As long as the device button is depressed, the virtual replica remains rigidly attached to the manipulation device and can be manipulated freely. When the button is released, the virtual replica stays at the position and orienta-

tion at the time of release. The expert can grab and manipulate the virtual replica as many times as they want. If the expert instead grabs the virtual proxy, the previously created virtual replica of that proxy will disappear and a new virtual replica will be created. (Our implementation allows only a single virtual replica for each virtual proxy to reduce visual clutter.) The virtual replica is removed if placed within a threshold distance and orientation from its virtual proxy.

The 6DOF position and orientation of the physical objects in the local user's environment are tracked and streamed to the remote expert's system to update the virtual proxies. In addition, the 6DOF position and orientation of the expert's manipulation and pointing devices are streamed to the local user's system, and the 6DOF position and orientation of each other's head are streamed to the other user's environment, where they can be visualized, as we describe later.

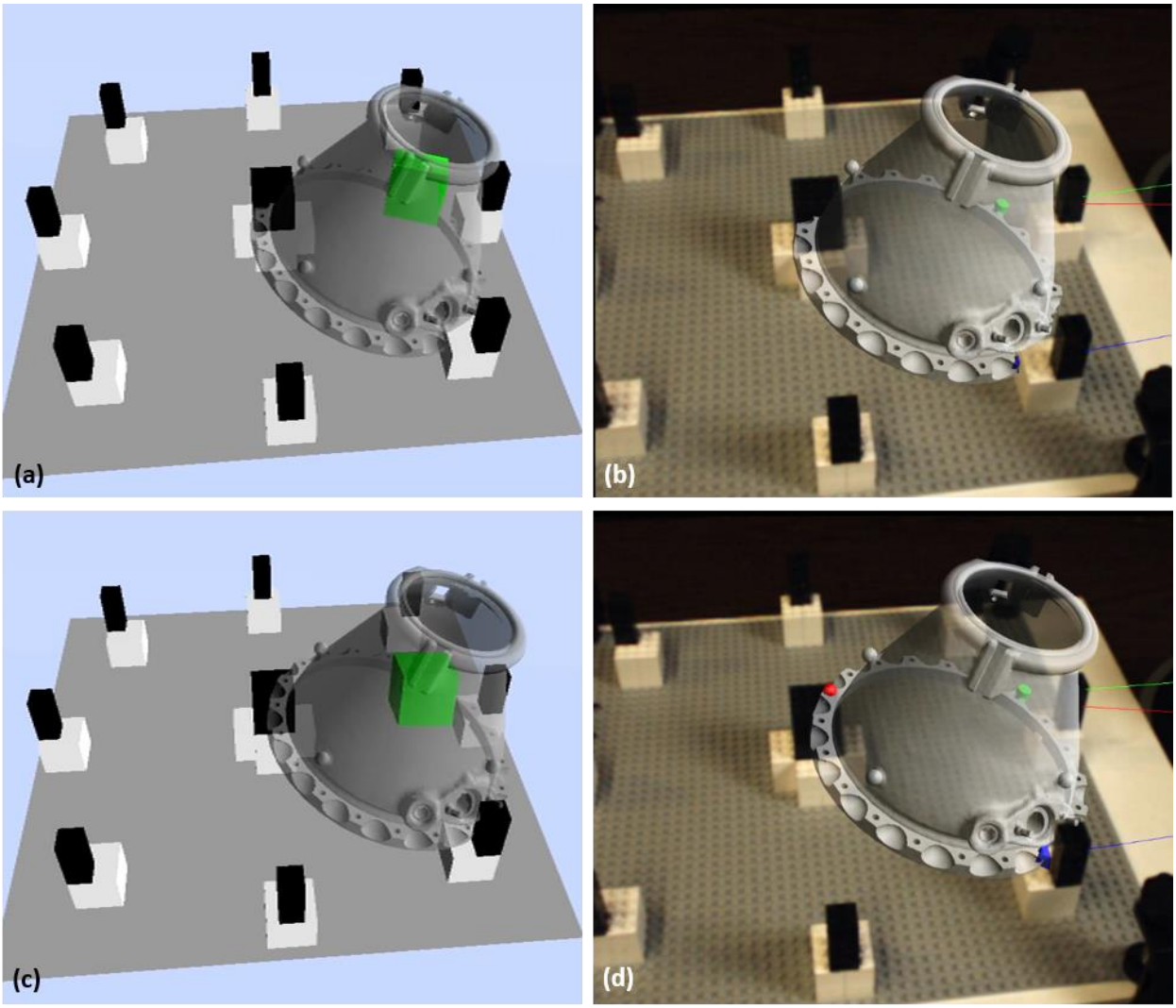

Figure 4.7. An example of our constraint-based approach with a Lego fixture (see Section 4.6). (a) The expert places the virtual replica of chamber top in VR so that it interpenetrates a white post at the bottom right of the Lego fixture. (b) Seen from the local user's perspective. (c) The chamber top is within a threshold distance and orientation from a predefined constraint, causing the top to snap to the constraint with a smoothly interpolated transition. (d) Seen from the local user's perspective.

4.4.1 DEMO3D Implementation

In DEMO3D, the remote expert uses the manipulation device in their dominant hand to create a virtual replica from its proxy and place the replica directly at the desired location. We make the virtual replica snap to the constrained location with a smoothly interpolated transition when the expert releases the virtual replica near a region where a constraint is specified, as

shown in Figure 4.7. The virtual replica will be left at the location at which it is released if there are no constraints within a threshold distance and orientation.

An arbitrary number of metaobjects can be defined by the expert prior to the task. Meta-objects can be added anywhere on the 3D model representing a physical object, preferably on prominent geometrical points, and will appear in the local user's environment as soon as the expert creates a virtual replica from a virtual proxy. The virtual replica fades out when the bounding boxes of the virtual replica and physical counterpart start to overlap, and fades in when the overlapping is resolved. To avoid fading out the replica while the expert is manipulating it, the fading behavior occurs only after the expert places the virtual replica.

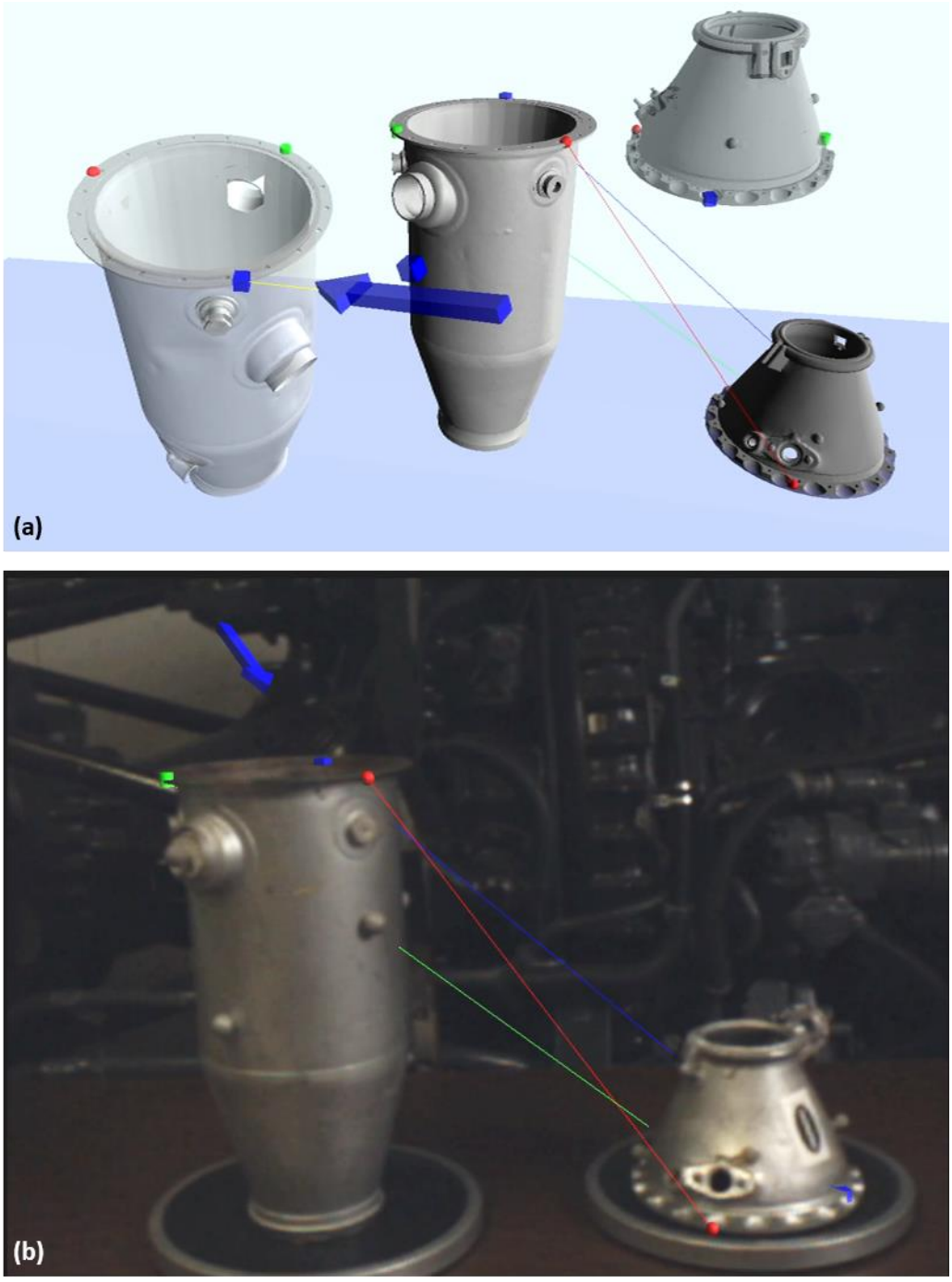

Figure 4.8. POINT3D. (a) The expert specifies three contact points (red sphere, green cylinder, and blue box) on both the chamber top and bottom, which in turn define the 6DOF pose of chamber top relative to chamber bottom. The pointing device is represented as an arrow with a color corresponding to the currently selected metaobject. We also display an indication of the currently selected metaobject (blue box) above the arrow. (b) The arrow appears relative to the corresponding physical counterpart of currently selected virtual replica or proxy in the local user's environment.

4.4.2 POINT3D Implementation

In POINT3D, the contact points on objects *A* and *B* are visualized as three metaobjects, each with a different shape and color, which protrude from the surface of each virtual proxy and virtual replica, as shown in Figure 4.8(a). We chose three contact points to support our rigid body manipulation tasks. Each pair of corresponding contact points on both objects is marked by metaobjects of the same shape and color. When instructing the local user, the remote expert first selects a metaobject, and then points to either a virtual replica or a virtual proxy to place it. The expert must place metaobjects on three contact points of each object to fully define the 6DOF pose of *A* relative to *B*. Selection of a metaobject and point on an object is done using a 6DOF-tracked pointing device, with two buttons to trigger pointing and choose a metaobject. We support bimanual interaction, in which the expert holds the manipulation device with her non-dominant hand, and the pointing device with her dominant hand.

Whenever the expert is pointing to an object, a 3D arrow appears in the local user's environment to help the local user identify the pointing pose. The arrow always appears relative to the physical object (Figure 4.8b), since the local user cannot see the corresponding virtual replica.

Gesture	*Function*
Drag with 1 finger	Sketch lines on the intersected virtual proxies.
Pinch with 2 fingers	Translate the camera along the ray emanating from the center of projection and the middle point between the two fingers' touch locations.
Drag with 3 fingers	Pan (when dragged horizontally) or tilt (when dragged vertically) the camera. *OR* (switched with a UI button) Orbit the camera around a point determined by the closest intersection between the central axis of the view volume and the virtual proxy or ground. Rotation is about the yaw axis (when dragged horizontally) or about the pitch axis (when dragged vertically) relative to the coordinate system of the intersected virtual proxy or ground.
Drag with 4 fingers	Translate the camera along the view plane.
Tap with 5 fingers	Reset the camera to a default position and orientation.
Double tap with 1 finger	Translate the camera towards a tapped location on a virtual proxy.

Table 4.1. SKETCH2D multi-finger gestures.

4.4.3 SKETCH2D Implementation

We implemented the following functionality with multi-finger gestures on the tablet: sketching lines, translating the camera in three dimensions, panning and tilting the camera, translating the camera towards a tapped location, and resetting the camera to a default position and orientation. The expert can also orbit the camera around a point determined by the closest interaction between the central axis of the view volume and the model. Table 4.1 details the gestures supported. In addition, we allow the expert to sketch using different colors and erase either all sketches or just those of particular color. To enhance visibility, lines are drawn with a glow effect, as shown in Figure 4.5. In the AR view, the expert can sketch, but not navigate, since the expert cannot control the local user's perspective (except to freeze the expert's copy).

4.5 Pilot Studies

Prior to conducting the formal user study, we performed informal pilot studies with our lab members and 12 compensated participants.

Initially, the expert interacted using natural hand gestures tracked by a depth camera [Wang et al., 2011]. However, the limited tracking range and lack of robustness in the recognition algorithms prevented smooth interaction with the virtual replicas. To address this, we switched to a Nintendo Wii remote [Nintendo, 2015], equipped with optical markers, which can be accurately tracked in a far larger volume. However, certain motions were difficult and time-consuming. In addition, one-handed interaction meant that POINT3D would take longer, due to the sequential process of pointing after grabbing and manipulating. Therefore, our implementation is now bimanual, using an easy-to-hold pointing device and a device for grabbing and manipulating virtual replicas, both outfitted with optical markers.

In POINT3D, we initially presented corresponding semi-transparent virtual replicas in the local user's environment. However, in early testing, we found that the virtual replicas usually cluttered the local user's limited field of view and often prevented the local user from seeing the metaobjects and rubberband lines clearly when the virtual replicas are placed between the local user and the physical objects. In addition, the technique placed the metaobject only when the expert placed the pointing device close enough to touch the surface without penetrating, but we found that without a physical obstruction preventing penetration of the object's surface, users would struggle to quickly place an annotation via ballistic pointing. To address this, we replaced the "touch and place" interaction with a raycast.

When testing the DEMO3D and SKETCH2D techniques, we found that if the local user moved the physical parts before the expert completed instructing, the movement of the virtual

proxy would interfere with completing the instructions. To avoid this, we introduced a "lock" feature that suppresses updating the expert's environment with the updated positions and orientations associated with objects moved by the local user until unlocked. We provided the expert with a foot-pedal to toggle between locking and unlocking for POINT3D and DEMO3D. For SKETCH2D, the expert taps on a button on the screen to toggle.

We initially showed an animation interpolating between the pose of a physical part and its virtual replica in DEMO3D to help the local user mentally map the 6DOF pose from the virtual replica to its corresponding physical part. However, we discovered that the animation was not helping the local user much. Instead, we found out that contact-point matching in POINT3D was easier for the local user to interpret. Therefore, we decided to replace the animation with a set of easily visible metaobjects on the physical part and virtual replica to further improve the performance of DEMO3D.

In SKETCH2D, we found people often dollied in too much, clipping the virtual proxies, requiring them to move back. To avoid this, we limit the amount the user can dolly in toward a point on a virtual proxy to a reasonable distance away from the point to avoid penetrating through the virtual proxy. We also limit how far away the user can dolly away from a point on a virtual proxy to avoid overshooting. We calculate this point by intersecting the ray emanating from the center of the two finger and the virtual proxies.

In certain trials, it was difficult for the expert to manipulate the virtual replica without reorienting themselves relative to the local user's environment. However, our setup sometimes made it difficult for the expert to physically move to a desired vantage point. To alleviate this, we provided the expert with an orientation-tracked physical lazy susan turntable to rotate their virtual view relative to the local user's environment. This feature was needed only for DEMO3D,

but not POINT3D, in which the expert can point at the virtual replica from a vantage point, making it unnecessary to physically move around. In DEMO3D, however, the expert needs to physically move to a viewpoint where she can place the virtual replica comfortably with precision.

The expert did not need to view the local user's environment in AR for the tasks in our study, since the physical objects were represented by accurate virtual models; further, we found that none of the participants in our pilot studies intentionally switched from VR to AR. Therefore, we removed the ability to switch between VR and AR for the user study to simplify the user interface.

4.6 User Study

We conducted a formal user study to compare the performance of DEMO3D, POINT3D, and SKETCH2D. We required that the accuracy with which the local user performed each trial be within a small range of distance and orientation from correct pose for the trial to end; therefore, we compared only time, not accuracy. To emulate the voice communications that would be supported in a remote collaboration environment (e.g., [Kuzuoka, 1992]), we allowed the two participants to communicate verbally during the trials to clarify misunderstandings or describe subtle adjustments. The experiment was conducted in the same room, so the participants communicated without a voice transmission device.

4.6.1 Hypotheses

Based on an analysis of the tasks and the pilot studies, we predicted:

- H1: DEMO3D should be faster than POINT3D. We expect POINT3D to take longer because it requires the remote expert to make six annotations, as opposed to DEMO3D,

which only requires a single motion for demonstration. DEMO3D also allows for a quicker/less precise alignment because of its embedded 5DOF constraint.

- H2: POINT3D should be faster than SKETCH2D. Similar to POINT3D, SKETCH2D also requires six annotations per alignment, but it additionally burdens the expert with 3D virtual camera navigation to be able to sketch on certain parts of objects. With POINT3D, the expert should be able to insert annotations relatively quickly using bimanual pointing. Both POINT3D and DEMO3D should require less interpretation time by the local user, compared to SKETCH2D, because of the assistance provided by the virtual rubberband lines that connect corresponding metaobjects.
- H3. Both experts and local users should prefer DEMO3D. We expect this because DEMO3D should be quicker and less cognitively challenging for the same reasons provided for H1 and H2.

The local user's task, which aligns the chamber top relative to a base explained in Section 4.6.2.3, is inherently 3D: to translate and rotate the chamber top to the specified position and orientation. For the expert, demonstrating the action to be performed is a more direct way of communicating than finding and marking the points of contact. For the local user, seeing the action is more easily comprehended and performed than finding and matching the points of contact. Research on the mirror neuron system in the brain suggests direct connections between perceiving an action and performing it (e.g., [Rizzolatti and Craighero, 2004]). Both methods are 3D, as is the task, so both methods should surpass the 2D tablet method.

4.6.2 Methods

4.6.2.1 Participants

We recruited 22 participants from our institution (5 male), 18–26 years old (average 22), through email and posted flyers. Participants were recruited as dyads who attended a single-session experiment together. Two participants had previous experience with AR, and none had any familiarity with our techniques.

Figure 4.9. An expert wearing a tracked Sony HMZ-T3W head-worn display interacts with (a) a tracked mouse (manipulation device) and lazy susan turntable in DEMO3D, and (b) a tracked Leonar3Do bird controller (pointing device) and mouse in POINT3D. (c) In SKETCH2D, the expert interacts with (d) an untracked Samsung tablet. (e) A local user wearing a tracked Canon HM-A1 head-worn display places an aircraft engine combustion chamber top on a Lego fixture.

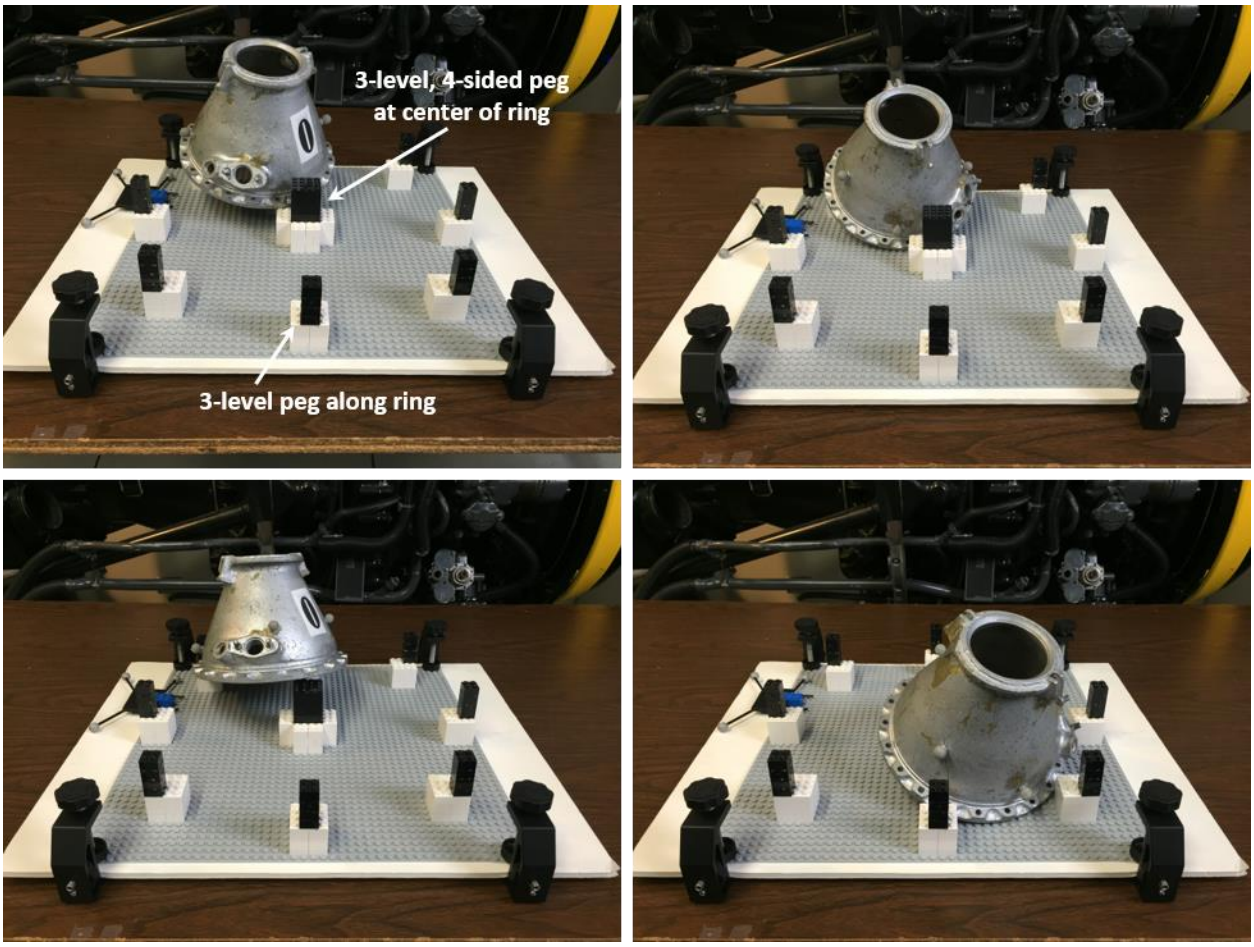

Figure 4.10. A few examples of possible 6DOF poses that can be specified between the chamber top and the Lego fixture. A 6DOF pose can be defined by choosing three resting points of the chamber top on the 3-level pegs: two from two neighboring pegs out of a ring of eight 3-level pegs, and one from the 3-level, 4-sided peg at the center of the ring. In addition, the chamber top can be rotated along its "up" axis at the same resting points for a different 6DOF pose (excluding those rotation angles where the protrusions on the chamber top would touch the pegs).

4.6.2.2 Equipment

In POINT3D and DEMO3D, participants assigned to the role of remote expert wore a Sony HMZ-T3W opaque stereo head-worn display. In DEMO3D, the expert held a tracked mouse (manipulation device) in their dominant hand and turned a tracked lazy susan turntable with their non-dominant hand (Figure 4.9a); in POINT3D, the expert held a tracked Leonar3Do bird controller (pointing device) in their dominant hand, and the tracked mouse in their non-

dominant hand (Figure 4.9b). The head-worn display, mouse, bird controller, and lazy susan turntable were tracked in 6DOF by a 10-camera NaturalPoint OptiTrack V100 tracking system, running on a computer powered by an Intel i7-4770k with 16GB of RAM and an Nvidia GeForce GTX 770. A footpedal was placed under a table to perform the "lock" feature described earlier. Experts used an untracked Samsung Series 7 Slate Tablet in SKETCH2D (Figure 4.9cd).

Participants assigned the role of local user wore a tracked Canon HMA1 stereo video–see-through head-worn display, as shown in Figure 4.9(e), running on a computer powered by an Intel i7-3770k with 16GB of RAM and an Nvidia GeForce GTX 780. Local users interacted with two physical objects that were replicated digitally for the expert. One was the top part of an aircraft engine combustion chamber, and the second was a fixture created for our study from 2x2 Lego bricks and a Lego 48x48 baseplate, as shown in Figure 4.9(e). (We used the fixture instead of the chamber bottom of Figure 4.1 to support a far wider range of possible 6DOF poses for the study, as shown in Figure 4.10.) The head-worn display, aircraft engine combustion chamber, and Lego fixture were tracked by a 12-camera NaturalPoint OptiTrack S250E tracking system.

4.6.2.3 Design

Since the participant playing the role of the remote expert is not a real expert, it was essential that the system provide instructions to the study participant to guide them in their role. To make the instruction easy to interpret and equivalent for all conditions, we added visual hints, similar to the POINT3D metaobject annotations, showing three contact points on both the top and the Lego fixture. In POINT3D and SKETCH2D, the expert would use these contact points as hints on how to place metaobject annotations or sketches. In DEMO3D, the expert would attempt to align the three corresponding contact points on each object. These contact points are visualized using the same shapes and colors used in POINT3D.

The instructional contact points for both objects are prepared before the study, to be shown by the system to the participant playing the role of the remote expert. In the informal pilot and the formal user study, the expert's goal was to place the top on the Lego fixture. The Lego fixture was organized as a ring of eight 3-level pegs with another 3-level, 4-sided peg at the center of the ring. Contact points for this fixture were defined as the exposed face of a Lego peg level. In total, there were 36 discrete contact points available for this fixture (Figure 4.10). Contact points on the top were defined in two phases. First, a set of contact point combinations are prepared for each unique, possible configuration relative to the Lego fixture. Three contact point combinations were chosen for this study. Second, the contact points are rotated about the center "up" axis of the chamber top. Since there were various protrusions on the top that would make certain poses difficult to match, we chose angles for the second phase that were achievable. In total, six angles were chosen for the second phase, representing ranges of acceptable angles truncated to 18° offsets, which aligned with generic holes on the chamber flange.

There were three within-subject interaction techniques × 8 ring pegs × 3 peg combinations × 6 relative yaw offsets = 144 unique 6DOF poses. Trials were blocked by technique and randomized by contact point combination and rotation angle for the top, and Lego fixture peg selection. Each block included three practice trials and six timed trials. Each technique described above was experienced first by one third of the dyads. The order of techniques was counterbalanced across dyads to minimize bias due to learning.

4.6.2.4 Procedure

Participants were welcomed by the study coordinators and given the PseudoIsochromatic Plate (PIP) Color vision test to screen for color blindness, the Stereo Optical Co. Inc. Stereo Fly Test (SFT) to screen for stereo vision, and the Vandenberg–Kuse Mental Rotation Test (MRT)

[Vandenberg and Kuse, 1978] to screen for spatial ability. The dyad member who scored higher on the MRT was assigned the role of remote expert, to reduce the effect of low spatial ability on instruction preparation. All participants passed the SFT; one (assigned as a local user) had some difficulty with the PIP, but did not perform worse than others in that role. After completing the tests and being assigned roles, participants were introduced to the study and given role-specific instructions. Both participants were seated in front of a table in the same room, but facing away from each other. Before each block, participants were given a detailed explanation of the interaction techniques. For SKETCH2D, an instruction sheet was provided to explain all possible controls. The participants were then shown a demo and allowed to explore the techniques.

The experiment had three segments, one for each technique, and each segment had two blocks. In the first block, one of the researchers served as expert while the participant designated to be the expert watched, and the participant assigned the role of the local user played that role. In the second block, the participant assigned the role of expert played that role and the other participant continued as the local user. The first block was necessary to demonstrate how to perform as an expert, but also allowed us to gather data from trials that used an experienced expert.

At the start of each trial, the expert was shown the virtual proxies and semi-transparent instructional metaobjects over the contact points on each object, and instructed to convey to the local user the 6DOF pose using the current interaction technique. The expert was instructed to press a dedicated “finished” button (by either tapping on a UI button on the tablet, or intersecting the manipulation device with a virtual button placed next to the Lego fixture proxy) to signal to the local user that the instruction had been prepared. However, the local users were encouraged to begin moving the chamber top as soon as they felt they understood the instruction. Once the pose had been acceptably matched, the virtual proxy turned green. The acceptable range was set

to be 25 mm in position and 7° in orientation (summed over all three axes of rotation) based on our pilot study. As soon as the pose had been acceptably matched, the expert confirmed by pressing the same button, now labelled "confirm." Once the trial was complete, the local user returned the top to the starting position, and all virtual annotations and replicas were removed in preparation for the next trial. Throughout the study, the 6DOF positions and orientations of the participants' heads, the manipulation device, the pointing device, the annotations, physical objects (and by extension, the virtual proxies), and virtual replicas were recorded.

Participants were asked to complete a three-part questionnaire (Appendix B.2) before, during, and after the study, assessing the three techniques. The questionnaire included an unweighted NASA TLX [Hart, 2006] and a request to rank the techniques from 1 ("Least Preferred") to 3 ("Most Preferred").

4.7 Results

We analyzed overall completion time separately by whether the expert was played by a researcher (Trained Expert) or a participant (Novice Experts). Each condition had a total of 198 trials (11 dyads × 3 conditions × 6 timed trials). We identified outliers using Tukey's outlier filter [Tukey, 1977], resulting in 5.6% (11 of 198 trials: four DEMO3D, four POINT3D, three SKETCH2D) of Trained Expert data and 6.1% (12 of 198 trials: six DEMO3D, two POINT3D, four SKETCH2D) of Novice Experts data being excluded from the rest of our analysis. We evaluated our hypotheses for significance using a Bonferroni-corrected α of 0.0167 (0.05/3).

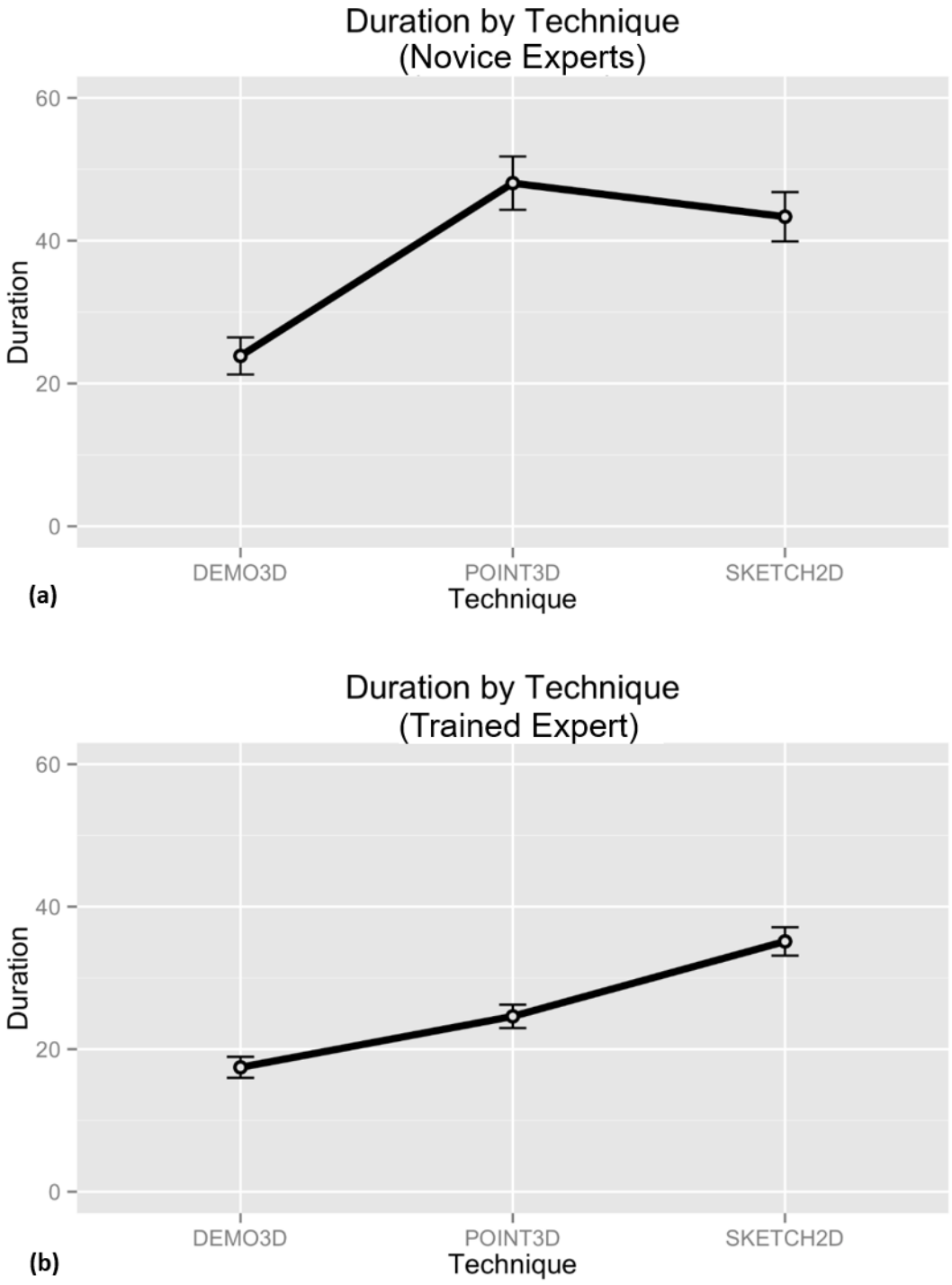

Figure 4.11. Mean completion time in seconds for (a) Novice Experts and (b) Trained Expert. Error bars show 95% confidence intervals normalized to remove between-subject variability.

Figure 4.12. Overall technique preference by role (1 = most preferred, 3 = least preferred). Error bars show 95% confidence intervals normalized to remove between-subject variability.

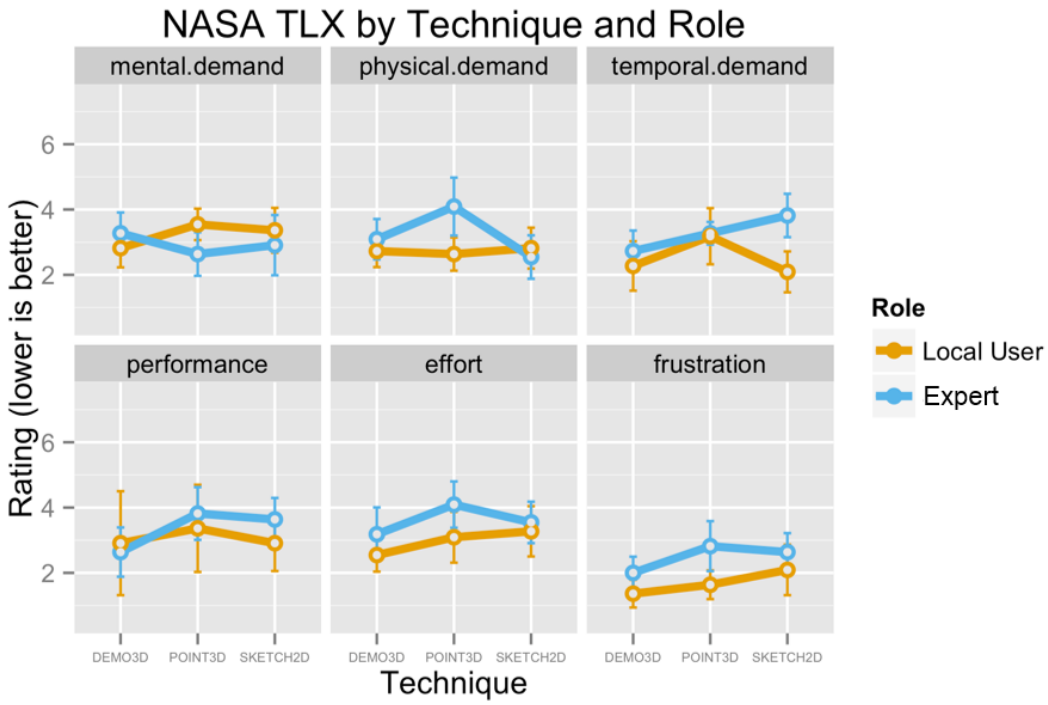

Figure 4.13. Mean values from unweighted NASA TLX survey with ratings from 1–7 (1 = best). Error bars show 95% confidence intervals normalized to remove between-subject variability.

We fit a linear mixed effects model to our data using R [R Core Team, 2015] to model completion time as a function of technique (fixed effect) and participant (random effect). Compared to a base model with only participant as a random effect, a Kenward–Roger corrected F-test showed that technique was significant as a fixed effect for both Novice Experts ($F_{(2,173.106)}$ = 79.269; $p < 0.0001$) and Trained Expert ($F_{(2,173.91)}$ = 153.82; $p < 0.0001$) (Figure 4.11). A pairwise least-squares means comparison revealed that the Trained Expert was significantly faster using DEMO3D than POINT3D ($t_{(178.44)} = -7.199$; $p < 0.0001$), validating H1. The Trained Expert was also significantly faster using POINT3D than SKETCH2D ($t_{(177.75)} = -10.230$; $p < 0.0001$), supporting H2. For the trials with Novice Experts, DEMO3D was significantly faster than POINT3D ($t_{(177.13)} = -11.944$; $p < 0.0001$) and SKETCH2D ($t_{(177.16)} = -9.551$; $p < 0.0001$), validating H1. However, completion times between POINT3D and SKETCH2D did not differ significantly ($t_{(177.12)} = 2.336$; $p = 0.0618$), failing to support H2.

Recalling that only recruited participants received the questionnaire, all expert ratings and rankings are for Novice Experts. A majority of participants (7 experts and 8 local users, 64% and 73% respectively) ranked DEMO3D as their most preferred technique (Figure 4.12), supporting H3. A Friedman test conducted to determine whether participants had a differential rank ordered preference indicated a statistically significant differential preference between techniques for local users ($\chi^2_{(2)} = 9.4545$; $p < 0.01$). A post hoc comparison using Nemenyi's procedure showed that rankings of DEMO3D were significantly more favorable than those for SKETCH2D, $p < 0.01$, partially validating H3 for local users. While local users on average ranked POINT3D higher than SKETCH2D and lower than DEMO3D, the difference between those two techniques was not significant. Experts on average ranked DEMO3D first, SKETCH2D second, and POINT3D

last, but the resulting Friedman test statistic was not significant ($\chi^2_{(2)}$ = 4:5455; p = 0.103), supporting, but not validating, H3 for experts.

Friedman tests applied to the unweighted NASA TLX survey results (Figure 4.13) revealed a significant difference between techniques in terms of perceived physical demand ($\chi^2_{(2)}$ = 9.1852; p = 0.01), temporal demand ($\chi^2_{(2)}$ = 6.75; p = 0.03), and perceived performance ($\chi^2_{(2)}$ = 6.8125; p = 0.03), by experts using an α of 0.05. Post hoc comparisons using Nemenyi's procedure revealed that experts felt POINT3D was physically more difficult than SKETCH2D. This may be because many participants are already comfortable interacting with 2D multi-touch displays using relatively common gestures, as opposed to attempting bimanual 3D pointing in immersive VR. (Several participants commented that they were used to touchscreen interactions.) Another possibility could be that the bimanual 3D pointing task of POINT3D is more physically demanding. In addition, POINT3D might have been at a further disadvantage because experts could not see their hands directly, occasionally resulting in their 3D controllers colliding with each other or their head-worn display. The rest of the pairwise comparisons between techniques for experts were not statistically significant; however, it is interesting to note that SKETCH2D had the lowest (best) average rank for physical demand, while DEMO3D had the lowest (best) average rank for performance and temporal demand. For local users, there was no significant difference between techniques in the TLX survey results, with generally low task load reported across all techniques.

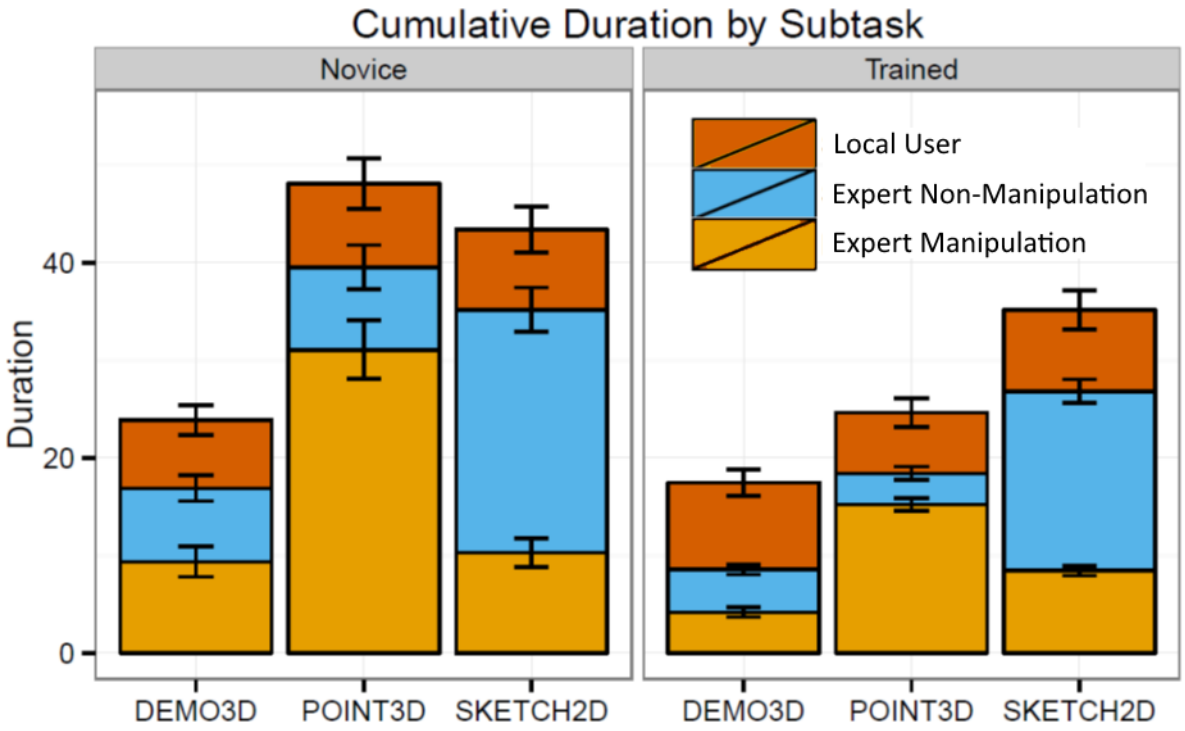

Figure 4.14. Task duration breakdown for Novice Experts and Trained Expert. Error bars show 95% confidence intervals normalized to remove between-subject variability.

4.8 Discussion

Most of the task time was used by the remote expert. While the approaches had a minimal effect on the time it took local users to place the top once instructed, several participants commented that the "(rubberband) lines connecting the shapes made alignment easier". Both Novice Experts and the Trained Expert were faster using DEMO3D. DEMO3D took little training; Novice Experts were barely slower than the Trained Expert. We believe DEMO3D was faster and more preferred because showing how to place the top at the correct position and orientation is more direct and natural than finding and annotating contact points. A participant commented about DEMO3D that "This one is more *human*. The grabbing feels like manipulating

things with my own hands with the gears", and another commented "Sometimes it is hard to put the object in the right place. But it is easy and entertaining to actually place the object." Many participants commented that DEMO3D was "easy". AR was key to creating transparent 3D virtual models that could be easily manipulated and visualized.

To understand why Novice Experts did not perform better using POINT3D than SKETCH2D, as we had hypothesized (H2), we analyzed usage patterns for each technique by breaking down the overall completion time into subtasks. One of the reasons we expected SKETCH2D to perform worse than DEMO3D and POINT3D was the amount of virtual camera navigation required by SKETCH2D. To understand how participants actually spent their time interacting with our system, we divided the overall completion time into expert and local user time (i.e., time between the start of the trial and when the expert pressed the "finished" button to signal to the local user that their instructions were ready). We further divided expert time into manipulation and non-manipulation time. For SKETCH2D, manipulation time was defined as the period during which the expert was touching the screen either to draw or to navigate with the virtual camera. For POINT3D, manipulation time was accumulated whenever the expert grabbed, held, or pointed at a virtual replica. For DEMO3D, manipulation time was simply the amount of time the expert grabbed and held a virtual replica while placing it onto the virtual fixture.

The results of this breakdown are shown in Figure 4.14. Surprisingly, expert manipulation time was much faster for SKETCH2D than POINT3D, even though both require adding six annotations. However, inspection of the videos revealed that most of expert time in SKETCH2D was spent with the expert's fingers hovering over the screen, but not actually touching it. Based on our observations, we believe that this hovering behavior occurred mainly because an expert (1) when navigating and drawing, touches near the screen center and taps on-screen buttons

placed near the screen edge to change sketch colors or proceed to the next step and (2) needs extra time to plan virtual camera navigation and recall the necessary gestures to execute those actions. For POINT3D, we were surprised by how much time and effort Novice Experts needed for bimanual pointing. A few participants commented that they felt they could do better on POINT3D after more practice trials. We believe this could be improved if experts could see their hands directly (in AR, rather than VR) and avoid accidental collision of tracked devices with one another and with the head-worn display.

Additionally, because holding the manipulation and pointing devices near the head or near each other resulted in degradation in tracking performance (causing overlap of retroreflective IR markers from the perspective of ceiling mounted cameras), we observed Novice Experts having to hold unnatural poses for longer periods of time, which may have also contributed to their perception of increased physical demand over SKETCH2D. A participant commented that "The antennas (trackable) fight sometimes and I need to stretch my arms out." Alternative tracking technologies, including ones that do not require line-of-sight (e.g., electromagnetic, which could work well in the expert's purely virtual environment), could alleviate this issue by allowing the expert to sit in a comfortable pose (e.g., leaning back, elbows resting) and fully leverage the affordances of bimanual interaction. Finally, participants' unfamiliarity with head-worn display and immersive AR environment might have contributed to the result as well, since several users commented that the head-worn display was somewhat taxing, uncomfortable, and causing dizziness after a while.

Chapter 5. A Physical-Interference–Avoidance Technique for Co-Located Multi-User Augmented Reality

As discussed in Chapter 1, it is often possible for users to physically interfere with each other while they interact in the same physical space and with the same virtual content. This is especially true when a user's attention is focused on virtual objects rather than on other users, or when the shared environment is sufficiently small. While some experiences encourage physical contact with other users or the environment (e.g., the Twister board game [Twister, 2009] or AR games such as Human PacMan [Cheok et al., 2004]), there are many multi-user AR applications in which it would be beneficial for the application to prevent undesirable physical contact. For example, the importance of safety in multi-user interactive application is evidenced by the protective jackets and straps provided by Nintendo for Wii remotes [Nintendo, 2015]. In contrast to these more extreme examples, we are interested in how the system itself can discourage unwanted and unintentional physical interference, possibly without making users aware that this is happening if such awareness might reduce user performance. In particular, we address hand-held AR, in which each user has a handheld device that displays information based on its tracked position and orientation. We are primarily interested in avoiding collisions between these tracked devices and, as a side effect, in avoiding collisions between users, who might otherwise not be tracked.

In this chapter, we first discuss related work in Section 5.1. Then, in Sections 5.2–5.3, we explain our approach, Redirected Motion, in detail and describe several alternative interference-avoidance techniques. Next, in Sections 5.4–5.5, we present the design and results of a formal user study that we conducted to measure how effective each of the techniques is at keeping users

apart and how distracting each technique is. Finally, in Sections 5.6, we discuss limitations and applications of Redirected Motion.

5.1 Related Work

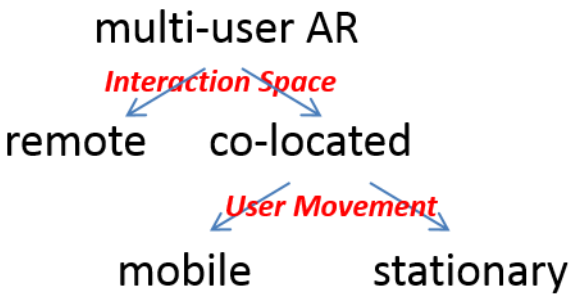

Figure 5.1. Classification of multi-user AR applications.

Existing multi-user AR applications can be roughly classified into two categories in terms of their interaction space, as shown in Figure 5.1: *co-located* in which the users are physically in the same space (e.g., [Barakonyi et al., 2005; Billinghurst and Kato, 2002; Björk et al., 2001; Cheok et al., 2002; Cooper et al., 2004; Hakkarainen and Woodward, 2005; Henrysson et al., 2005; Kiyokawa et al., 2002; Mulloni et al., 2008; Oda et al., 2008; Ohshima et al., 1999; Ohshima et al., 1998; Paelke et al., 2004; Reitmayr and Schmalstieg, 2001; Szalavári et al., 1998; Wagner et al., 2005]) and *remote* in which the users are physically in separate spaces (e.g., [Barakonyi et al., 2004; Diaz et al., 2006; Kato and Billinghurst, 1999; Leung et al., 2007]). We can further classify co-located environments into subcategories, as shown in Figure 5.1. When users' interaction spaces are remote, there can be no physical interference between the users. However, physical interference can occur when users are co-located and *mobile*, especially in situations in which the system does not impose rules to restrict each person to their own private space (e.g., [Barakonyi et al., 2005; Oda et al., 2008; Wagner et al., 2005]). For example, the

non-driver players in our AR racing game (Section A.4.1 [Oda et al., 2008]) can move arbitrarily around the gameboard to place tracked obstacles in desired places. In contrast, when users stay in their own separate location or never enter another user's interaction space (*stationary*) (e.g., [Cooper et al., 2004; Hakkarainen and Woodward, 2005; Henrysson et al., 2005;2006; Ohshima et al., 1998; Szalavári et al., 1998]), they cannot interfere with each other physically. For example, each of the two players in AR tennis [Henrysson et al., 2006] may move around on the side of their court, but is not allowed to cross the net, eliminating the possibility of physical interference. In this paper, we will focus on the case where users are co-located and are expected to move around in a shared application space (*mobile*).

Researchers have long explored how to prevent users from colliding with physical obstacles, such as walls, in virtual (and real) environments. One early method plays a warning sound when the user comes within a preset distance of an obstacle. Cheok and colleagues apply a more subtle version of this approach in their multi-user AR application, Touch-Space [Cheok et al., 2002]: The spatialized 3D sound of an airplane propeller is used to make each player aware of the other player's location, even when they cannot be seen.

Fitzpatrick and colleagues take advantage of the decisive role the balance mechanism of our inner ears play in directing walking [Fitzpatrick et al., 2006]; they show how applying an electrical current to a surface electrode placed behind the ear can be used to remotely "steer" a blindfolded user as they walk, without the user noticing the redirection if it is sufficiently subtle. In contrast, Razzaque and colleagues have developed "redirected walking" [Razzaque et al., 2001], in which the virtual location of a user in VR is transformed by injecting additional virtual rotation, especially while the user is physically moving. This can change the direction in which the user walks, cumulatively affecting the user's physical location and steering the user away

from obstacles—without the user realizing it if the incremental modifications are sufficiently small. While redirected walking assumes that the user is in VR, the new technique that we present in this chapter was inspired by the idea of transforming a user's virtual location to influence their physical location.

Other work has explored scaling interaction space in VR, For example, virtual motion can be scaled up from physical motion to allow the user to traverse virtual space faster [Bryson and Levit, 1991; Interrante et al., 2007] or extend their virtual reach [Poupyrev et al., 1996]. However, the use of these techniques is typically intended to be perceptible to the user. There is also an analogy to "mouse acceleration" in 2D desktop user interfaces, in which the mouse transfer function is dynamically modified to increase the speed of the cursor nonlinearly as the mouse is moved faster [Mackenzie and Riddersma, 1994]. Here, the tradeoff between precision and speed is of concern, not avoiding interference.

Finally, we note that collision avoidance is well-studied in multi-robot interaction [Asama et al., 1991; Cai et al., 2007; Fox et al., 1998]. However, the approaches adopted for that domain are quite different, typically involving direct control over the robot's motion without any concern for keeping the robot "unaware" of the change.

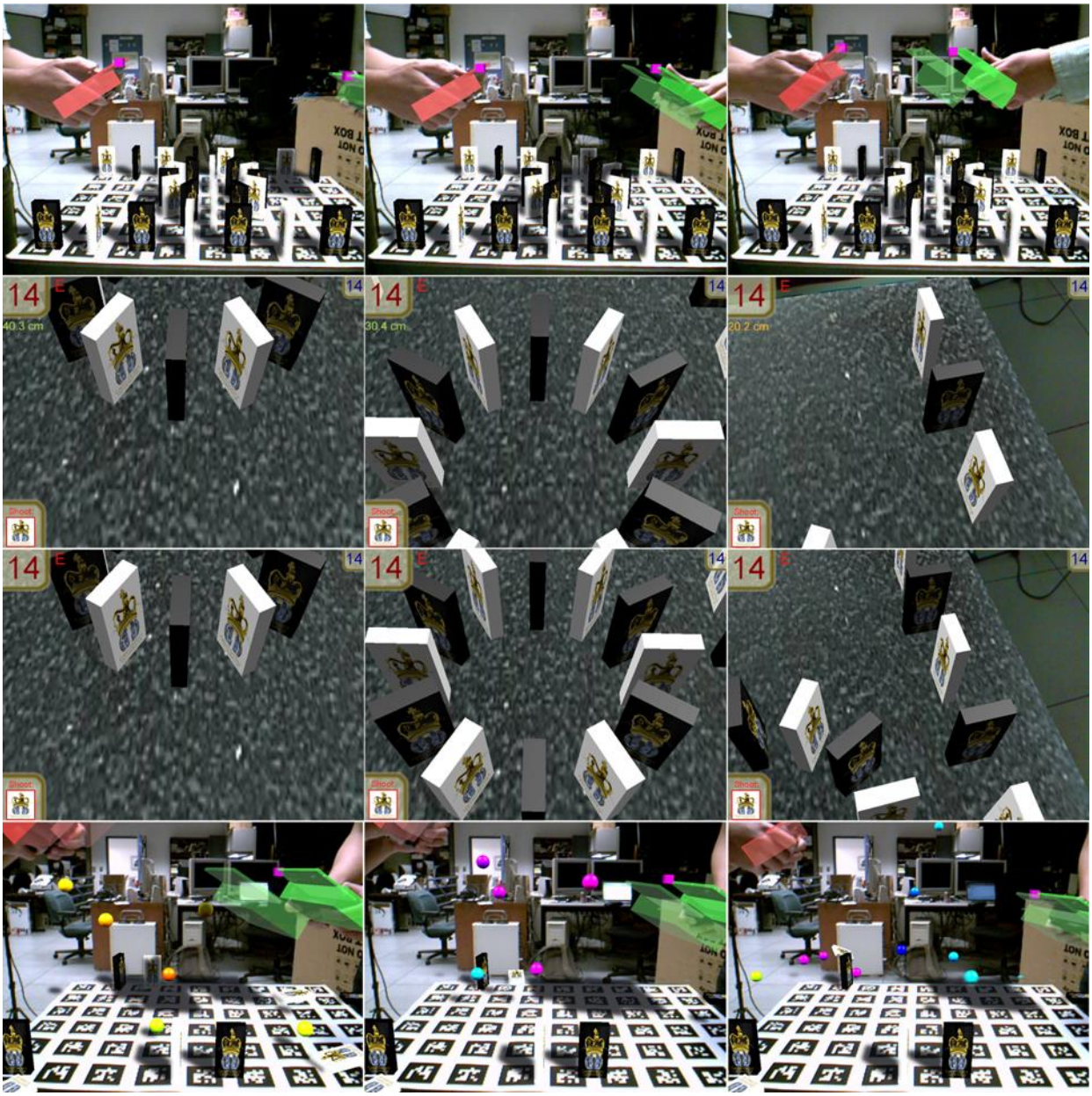

Figure 5.2. Redirected Motion. (Row 1) Changes in physical and virtual locations under Redirected Motion. As the green player (right) moves toward the red player (left), the virtual location of the green player is shifted along the green player's direction of movement, while the (stationary) red player's virtual location is not affected. (Row 2) First-person screenshot views for the green player with Redirected Motion, corresponding to images on row 1. (Row 3) First-person views for the green player without Redirected Motion (i.e., no virtual shifting), corresponding to images on row 2. (Row 4) As the green player moves back beyond the distance threshold τ from the red player, the virtual location of the green player is shifted back to its physical location.

5.2 Redirected Motion

We have developed an interference-avoidance technique, *Redirected Motion* that attempts to address the issue of unexpected physical interference among co-located mobile users. As shown in Figure 5.2, Redirected Motion transforms the virtual location of a user as they approach another user to keep the users from colliding, inspired by earlier work on redirected walking in VR [Razzaque et al., 2001].

We refer to the actual location (position and orientation) of a user's device in the physical world as the user's *location*. We will sometimes refer to the user's location as the user's *physical location* for emphasis, and will often make an implied reference to the user's location just by referring to the *user*. In contrast, we will use the term *virtual location* to refer to a possibly different location at which the system treats the device as being for purposes of rendering and interaction. We make three assumptions:

1. Users are most interested in the location toward which they are moving and the virtual objects with which they can interact at that location.
2. Users are aware of each other's physical location roughly, but not exactly.
3. Users are not intentionally trying to physically interfere with each other.

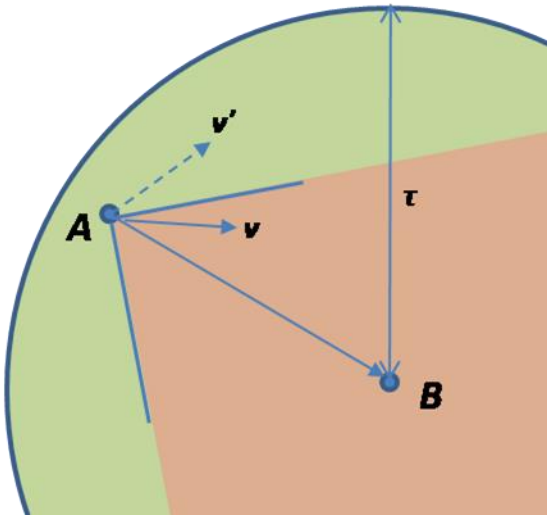

Figure 5.3. User *A*, moving with velocity *v*, is treated as moving toward user *B* only if the angle between *v* and the vector from *A* to *B* is less than 45° (the red area). For example, the solid vector *v* is classified as moving toward user *B* and the dashed vector *v'* is not.

When user *A* in Figure 5.3 reaches a set threshold distance from user *B*, then, as *A* continues to move toward *B*, *A*'s virtual location is shifted ahead of *A*'s physical location. The shift occurs in *A*'s instantaneous direction of travel, where the amount of shift is proportional to *A*'s velocity. The intent is that by exaggerating *A*'s travel virtually, *A* could be prevented from physically contacting *B* when *A*'s destination is sufficiently close. In other words, we hypothesize that shifting *A*'s virtual location ahead of *A*'s physical location could, in many cases, cause *A* to "stop short," and, thus, could keep the physical distance between *A* and *B* from being as small as it would be if this technique were not applied.

Interrante and colleagues developed "seven league boots" [Interrante et al., 2007] that scale the distance a user travels by amplifying the user's velocity only in the direction in which they are walking. We adapt this approach by adding an additional translation to the virtual location of a user in the direction in which they are moving. Note that this transformation will affect just the overlaid virtual scene. As shown in Figure 5.3, the translation is applied to user *A* only when their distance to another user *B* is under a specified threshold $\boldsymbol{\tau}$ and *A* is moving toward *B*.

To capture the notion of moving toward *B* as more than just getting closer to *B*, we include only motion in which the velocity component in the direction from *A* to *B* is greater than the component perpendicular to that direction.

We define $\vec{G}$ to be the cumulative 3D *virtual shifting vector* which is the translation from a user's physical location to their virtual location. $\vec{G}$ is initially (0,0,0), and is incremented at each rendering update (approximately thirty updates every second) by adding $\vec{g}$, which is calculated as:

$$\vec{g} = \frac{\delta \cdot \vec{v}}{M_1},$$

where $\vec{v}$ is the user's motion vector, and M_1 is a scalar constant to adjust the shifting distance. δ is 1 when the user moves more than a certain motion threshold ψ ($|\vec{v}| > \psi$) and the distance between the users is less than $\boldsymbol{\tau}$, otherwise, δ is 0. When the user is outside the distance threshold $\boldsymbol{\tau}$, $\vec{G}$ is gradually shifted back to (0,0,0) by μ cm each frame that the user moves more than ψ.

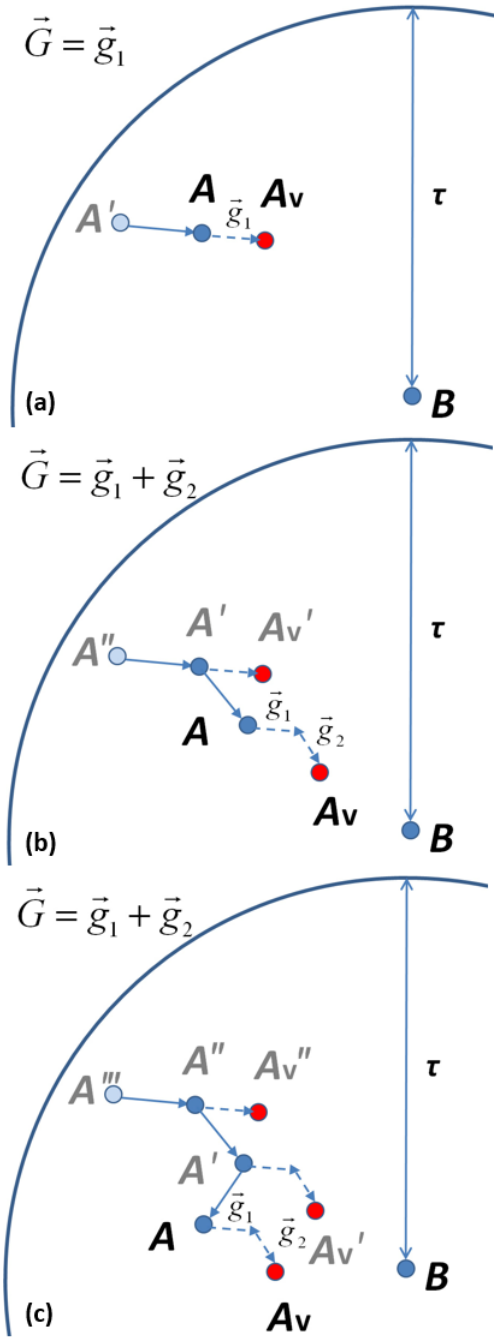

Figure 5.4. Physical and virtual locations of user *A* over three frames. In each frame, *A* indicates the current physical location, and A_v indicates its shifted virtual location. Primed labels indicate previous physical and virtual locations. Solid vectors indicate motion of the physical location of *A*. Dashed vectors are components of the virtual shifting vector $\vec{G}$, exaggerated for this figure. (a–b) Additional virtual shift is applied along the motion vector from *A′* to *A*, since the user moves toward *B*. (c) Additional virtual shift is not applied, since the user does not move toward *B*.

The elements *x*, *y*, and *z* of $\vec{G}$ are clamped between lower and upper shifting limits along each axis, such that $x_{\text{lower}} \leq x \leq x_{\text{upper}}$, $y_{\text{lower}} \leq y \leq y_{\text{upper}}$, and $z_{\text{lower}} \leq z \leq z_{\text{upper}}$. We do this to limit the amount of visual mismatch between the virtual view and physical view. In Section 5.3.3, we provide the specific values used for these parameters in our study application, as decided based on pilot studies. Figure 5.2 shows the technique as applied in our study, while Figure 5.4 shows schematic examples of how the physical and virtual locations of a user change over three frames.

Note that there is a clear distinction between this approach and a simpler one in which the virtual location is offset from the front of the device by a fixed vector throughout the application (a 3D analogue to Vogel and Baudisch's shifting of the 2D selection point on a touch-screen a set distance away from the user's finger [Vogel and Baudisch, 2007]). If the virtual location is always offset by the same amount, then there will be always a mismatch between the virtual and physical worlds, even when one is not needed. Furthermore, the offset will not reflect the direction in which the user is moving or the direction to the other user. For example, offsetting from the front of the device may not be useful when the users are standing shoulder to shoulder, with their devices facing in the same direction.

Since Redirected Motion changes the location from which virtual objects are rendered, it is not suitable for applications in which the offset between the virtual and physical locations will violate registration requirements. The challenge is how to avoid perceived misregistration. We address this in two ways. First, the application shown in Figure 5.2 is typical of many AR games in which there is a relatively loose connection between the virtual and physical worlds, with the physical world made visible largely to support social interaction. Like other examples of this genre [Henrysson et al., 2006; Nilsen and Looser, 2005], our game is played using a tracked board that is completely covered with a virtual texture, while allowing the rest of the environ-

ment to remain visible. In our case, this virtual texture prevents players from seeing virtual game objects appear to translate relative to the real board while the offset between the rendered virtual view and the real camera view changes. Second, as we discuss later, we also place an application-dependent limit on the magnitude of the offset.

5.3 Test Application: AR Domino Knockdown

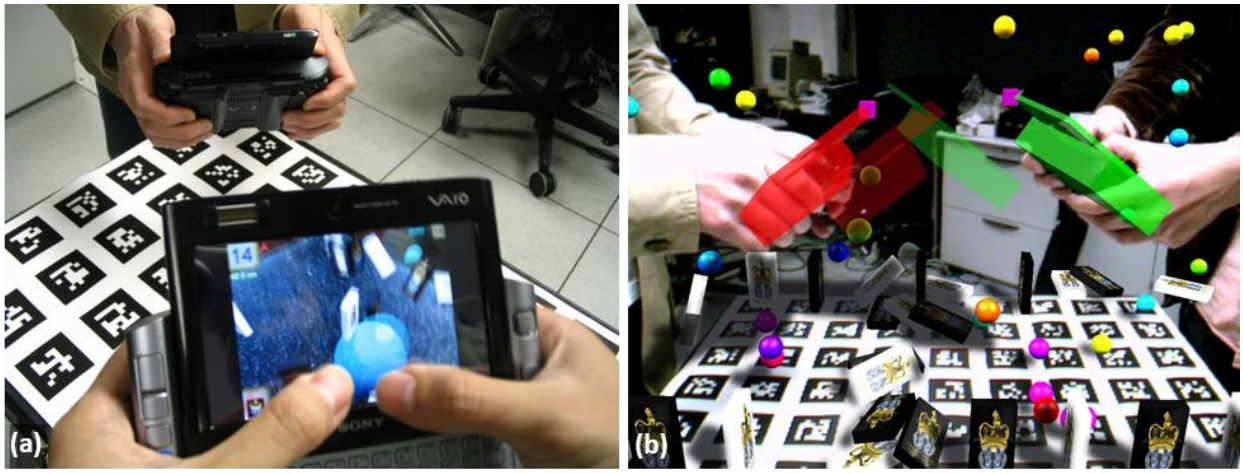

Figure 5.5. Test Application. (a) Player's view of two-person AR Domino Knockdown game. Virtual balls are fired at virtual dominoes by tapping on the screen. (b) Third-person view of game with AR visualization of *Redirected Motion*. Each hand-held UMPC is overlaid with a simplified geometric model to represent its actual physical location, and a more transparent offset geometric model to represent its shifted virtual location. Two small magenta cubes highlight the two points (on the models at the physical locations) that are currently closest to each other.

We developed a two-player game, AR Domino Knockdown, shown in Figure 5.5, in which to test our techniques. Each player attempts to knock all of the other player's virtual dominos off a real table (45 cm wide × 65 cm long) by shooting virtual balls with a hand-held ultra-mobile personal computer (UMPC) through which they view the environment. The table is covered with an array of fiducial markers for tracking, and we overlay a virtual texture resembling the table's surface on top of the marker array. Two sets of virtual dominos, each set having a distinctive color and texture, are initially placed in a startup configuration on the table.

Each player shoots virtual balls by tapping their fingers on the screen of their UMPC, which launches virtual balls from the tapped locations. Players can move anywhere around the table and shoot from any angle they desire, as long as at least one marker in the array on the table is visible from and can be tracked by the device's integrated camera. The device's position, orientation, and instantaneous velocity determine the direction in which a ball is fired and its speed. A rigid-body physics simulation is performed on the virtual balls and dominos. When a domino falls off the table, it is removed from the simulation. The numbers of dominos that each player has left is displayed on the screen, along with the distance between the players. (The color of the displayed distance is initially yellow green, and if the distance gets below the threshold τ', it turns orange.)

5.3.1 Software

Our game is built using Goblin XNA [Oda and Feiner, 2012] version 3.0 (Appendix A). Players communicate through a central server for which the player machines are clients. We use a server-client model instead of a peer-to-peer model to equalize and minimize the load on the clients. The physical simulation is performed on the server, and the simulated results are broadcast to the clients so that both see the exact same result. The central server broadcasts the game status in addition to the simulated results, and the client transmits to the server its camera transformation (which is the inverse transformation of the marker array it sees on the table) and the shifting transformation if Redirected Motion is used.

5.3.2 Hardware

The application runs on Sony VAIO UX-VGN-380N and UX-VGN-390N UMPCs, which are 1.2 lb., 5.9" (W) × 3.7" (H) × 1.3" (D) hand-held devices with a 4.5" diagonal touch-sensitive LCD screen and an integral camera in the back of the device. We used two of these for

the user study. Both run Windows XP on a 1.33 GHz Core Solo CPU with 1 GB memory, and an Intel 945GMS GPU, and communicate through a dedicated WiFi network with a laptop that acts as a central server. The laptop is a Sony VAIO VGN-SZ480 running Windows Vista with a 2.33 GHz Intel Core 2 Duo CPU, 2 GB memory, and an NVidia GeForce Go 7400 GPU.

5.3.3 Pilot Study

Symbol	Value	Description
τ	41 cm	Distance threshold between users *A* and *B*, below which Redirected Motion is triggered
τ'	25 cm	Distance threshold between users *A* and *B*, below which interference avoidance techniques other than Redirected Motion are triggered
$(x_{lower}, y_{lower}, z_{lower})$	(−4.5 cm, −6.5 cm, 0 cm)	Lower bounds of shifting vector elements for Redirected Motion
$(x_{upper}, y_{upper}, z_{upper})$	(4.5 cm, 6.5 cm, 0 cm)	Upper bounds of shifting vector elements for Redirected Motion
μ	0.01 cm	Shifting amount applied every ρ msecs when shifting $\vec{G}$ back to (0, 0, 0)
ψ	0.3 cm	Lower motion velocity threshold for enabling Redirected Motion when the physical distance between users *A* and *B* is under $\boldsymbol{\tau}$
ψ'	0.3 cm	Lower motion velocity threshold for triggering interference avoidance techniques other than Redirected Motion when the physical distance between users *A* and *B* is under $\boldsymbol{\tau'}$
M_1	2	Scalar constant to adjust shifting distance
M_2	3	Scalar constant that determines how rapidly brightness changes for Screen Dimming
ω	30	Average frames per second for application
ρ	33	Msecs between render frames

Table 5.1. Summary of parameters used in user study.

Prior to conducting a formal user study, we performed informal pilot studies with other lab and department members in order to choose suitable values for the variables used in the equations and thresholds specific for our experimental setup, as shown in Table 5.1. We determined that $M_1 = 2$ would produce unrecognizable, yet significant shifting for Redirected Motion, $M_2 = 3$ would render a visually smooth transition for dimming, and $\boldsymbol{\tau'} = 25$ cm (~10 inches) would be a reasonable distance for the effect threshold. As the player taps on the screen to shoot balls, it

moves the display slightly, which we observed to range up to ~ 0.3 cm. Therefore, we set both ψ and ψ' to be 0.3 cm. Shifting by 0.01 cm every ρ msecs when the user's movement speed was more than 0.3 cm/sec was fast enough, as well as unnoticeable, so we set $\mu = 0.01$ cm. Since our application was running around 30 frames per second, we set $\rho = 33$, and $\omega = 30$.

To determine the shifting distance threshold $\boldsymbol{\tau}$, we first needed to determine lower and upper shifting bounds for our game. Since we did not want the participants to realize that the view point was shifted, we decided to overlay the table with a texture size larger than the actual table. If the texture size is exactly same as the table size, then it would become obvious that the viewpoint was shifting. Either the dominos would need to shift relative to the texture (which would look odd, especially toward the edge of the table) or the texture would have to shift too, uncovering the table as it was translated, and clearly moving, as compared to the edges, since we cannot shift the physical view seen by the camera. To compensate for this anticipated mismatch between the virtual and physical view, we decided to overlay the tabletop with a texture that is larger than the actual tabletop, but not so big that the participant could easily realize the discrepancy. By setting the shifting limit to (texture size – actual size) / 2, the physical table will never be uncovered. Based on our pilot studies, we decided to use a texture 1.2 times larger than the actual size of the table (1.2 × (45cm × 65cm) = (54cm × 78cm)). Thus, $(x_{lower}, y_{lower}, z_{lower})$ = (−4.5cm, −6.5cm, 0), and $(x_{upper}, y_{upper}, z_{upper})$ = (4.5cm, 6.5cm, 0). Furthermore, we do not shift the virtual location along the z-axis, since we want to have the stationary dominoes to appear to be resting exactly on top of the physical table. (If the majority of virtual objects were not supposed to reside on the physical table, we would have shifted in z, as well.)

Once we decide on the shifting limits, we could compute the maximum shifting amount as $\sqrt{4.5^2 + 6.5^2} \approx 8$ cm. Since $M_1 = 2$, a motion distance of 8 cm × 2 = 16 cm is required for

full shifting. Since we wanted to maximize the effectiveness of Redirected Motion, we wanted the participants to be shifted for at least half the maximum amount in the worst case before they are considered to be too close to each other. In the worst case, when players move toward each other, the total distance they move will be approximately 16 × 2 = 32 cm before they get fully shifted. Thus, since we had chosen $\boldsymbol{\tau'}$ = 25 cm, we decided to set $\boldsymbol{\tau}$ = 25 cm + 32 / 2 cm = 41 cm.

5.4 User Study

To understand how well our avoidance techniques work in terms of effectiveness and distraction level, we conducted a formal user study. 18 paid participants were recruited by flyers and mass email requesting that users apply as pairs (dyads). All participants were university students (3 female and 15 male, 22–36 years old, average 26 years old). All subjects use computers multiple times per day, but only two had previous experience with AR, and only one subject had previously used a UMPC.

The members of each dyad were asked to play multiple rounds of the two-player AR Domino Knockdown game under different test conditions, as controlled by a test scaffold. We measured the effectiveness and distraction level both qualitatively and quantitatively. Qualitative measurement was assessed through a questionnaire. Quantitative measurement of effectiveness was assessed by computing the average distance between the two players during game play. The distance between the two players was recorded at each frame as the shortest distance between any point on one UMPC and any point on the other UMPC, computed by the physics engine using the 3D models of the UMPCs at their physical locations, as shown in Figure 5.1. We also recorded the average completion time of each game round as a potential measurement of distraction level.

5.4.1 Comparison Techniques

We were interested in comparing Redirected Motion with other techniques that are intended to persuade users to move away from each other when in physical proximity. The techniques we describe in this section modify what a user sees or hears, and what actions they can perform, but do not create an offset between the user's physical and virtual location. Instead, the goal is to alert the user to the impending interference, under the assumption that this will cause them to modify their trajectory.

The techniques are applied only when the distance between the users' physical locations is under a preset threshold $\boldsymbol{\tau}'$. Here, we choose $\boldsymbol{\tau}'$ to be lower than $\boldsymbol{\tau}$, to minimize distraction, since these techniques are intended to be noticeable. As with Redirected Motion, applying the techniques to a user is based on the absolute velocity of that user. (Thus, a user who is not moving toward another user will never experience any of these techniques, no matter how close the other user is or how fast the other user approaches.) The specific parameters used in our implementation are discussed in Section 5.3.3.

5.4.1.1 Screen Dimming

Our first alternative technique, *screen dimming,* modifies what the user sees by dimming their display. This alerts the user, but is intended to be punitive, since the dimmer the screen, the less well the user can see the virtual objects and perform effectively. The amount by which the screen is dimmed at each frame is computed as:

$$f = \frac{\delta \cdot |v|}{d \cdot M_2}, \qquad (0 \le F \le 1).$$

The current brightness of the screen, F, to which f is added at each frame, is clamped between 0 and 1, where 0 means normal brightness and 1 means totally off. $|v|$ is the movement speed, d is

the distance between the users, M_2 is a constant that determines how rapidly the brightness changes, and δ is either 1 in the case the user is moving toward the other user, or −1 if not. The screen brightness changes only if the user moves more than a motion threshold ψ' ($|v| > \psi'$) when the distance between the users is less than $\boldsymbol{\tau'}$. The screen gets dimmer and dimmer as the user moves their display toward the other player, until it is totally off. The screen gets dimmer quicker as *d*, the distance between the users, decreases. Since the technique is not intended to be imperceptible, *F* is reset to 0 once the user moves back beyond the $\boldsymbol{\tau}$ distance threshold.

5.4.1.2 Sound Beeping

There are many ways in which audio can be used to notify the user of the possibility of interference. A sound can be played if the user is within $\boldsymbol{\tau'}$, and various properties of the sound can be predicated on the user's velocity (e.g., modifying frequency, amplitude, or waveshape). In contrast to the more subtle use of spatialized 3D audio in Touch-Space [Cheok et al., 2002], our *sound beeping* technique use a simple "beep." While the user is within the distance threshold $\boldsymbol{\tau'}$, the number of beeps per second is incremented by one every ρ msecs if the user is moving toward another user and decremented by one every ρ msecs otherwise, if the movement speed $|\vec{v}|$ is greater than the movement threshold ψ'. The number of beeps per second is clamped between 0 and ω. The frequency is reset to 0 once the user moves back beyond the distance threshold $\boldsymbol{\tau'}$.

5.4.1.3 Action Disable

Disabling user interaction capabilities is also an effective way of modifying user experience. Inspired by the punitive response of a pinball machine that has been "tilted" by the user, the *action disable* technique prevents user interaction. This is a Boolean operation in our implementation of this approach: we completely disable user action (in this case, preventing balls from

being fired) as soon as the user crosses the distance threshold τ'. Action is immediately re-enabled after the user moves out of the distance threshold.

5.4.2 Study Description

Our user study was a within-subject, single-session experiment in which we compared each participant's behavior on a single task using four different interference avoidance techniques plus a control condition in which none of the techniques were used. We began each dyad's session by showing the two participants how to play the game.

Participants played a total of 25 rounds, with each round typically taking 30–120 seconds. Each round was initiated by the study coordinator so that the participants could rest or provide comments as desired between rounds. Participants initially played one practice block of five rounds with none of the techniques enabled to get acclimated to the game, followed by four blocks for the actual user study. Each of the four study blocks contained five rounds (one each for the four different interference avoidance technique and one in which no interference prevention techniques were used). Technique order within the study blocks was randomized using a Latin square.

Before participants started the first (practice) block, the study coordinator told each participant the color (white or black) of their designated target dominos, which remained the same throughout the study. The two participants were asked to position themselves at opposite ends of the table along the longer edge (65 cm) at the start of each round, and were reminded to avoid shooting their own dominos as much as possible and to move around the table to shoot more precisely. Before participants started the remaining blocks, the study coordinator informed them that they would see certain effects if they got too close to each other, and that they would need to move away from each other if they wanted the effects to disappear. They were also told that

some effects are quite obvious, but some are very subtle. To avoid biasing participants between control rounds and Redirected Motion rounds, they were not told that each block contained a control round in which no technique was used.

We found that the players during our pilot studies were not exactly sure when the new game round started after the study coordinator signaled the server to begin the next round. Therefore, for the formal study, we provided a countdown signal with sound and text message at the beginning of each round, and asked the players to start shooting after the countdown ended.

Each participant was asked to fill out a post-hoc questionnaire (Appendix B.3). The questionnaire was designed to assess the effectiveness and distraction level of the five conditions using Likert-scale questions ranging between 1 (worst) and 5 (best), a request to rank the five conditions with ties allowed, and space for free-form comments. To avoid the bias that could result from using descriptive technique names, we labeled each condition alphabetically (*A*–*E*). These letters were displayed on the UMPC screen for each game round during the actual user study, so that the participants could associate each effect with each letter. The study coordinator also emphasized which letter maps to which technique and explained the meaning of "effectiveness" right before each participant was asked to fill out the questionnaire.

Participants were told that the effectiveness of a technique referred to how well that technique kept the two participants away from each other. Since we expected that participants would not detect any effect for Redirected Motion because of our attempts to make it unrecognizable, the study coordinator also repeatedly asked the participants what effects they saw for the control condition and Redirected Motion. Participants were also asked to remember the difference between the control condition (*A*) and Redirected Motion (*E*) to avoid confusion when completing the questionnaire.

Total time for the study was approximately one hour. During the study, the software collected game play data including geometric relationships between the players' UMPCs (computed from the tracked 3D positions and orientations of the UMPCs), game scores and scoring profiles, duration of the time that players were within threshold $\boldsymbol{\tau'}$, and game duration.

5.4.3 Hypotheses

Prior to our experiment, we formulated five hypotheses:

- H1: The control condition (None) will be least effective and least distracting.
- H2: Sound Beeping will be less effective, but less distracting than Screen Dimming or Action Disable, since the player can simply ignore the beep.
- H3: Screen Dimming and Action Disable will be most effective, but most distracting, compared to other techniques, since visibility or interaction capability are impaired.
- H4: Redirected Motion will be as effective as Screen Dimming or Action Disable and significantly more effective than Sound Beeping or None, since the players can get close to their targets without the need to physically be close to the targets, and as a result, the players can be effectively far away from each other.
- H5: Redirected Motion will be the least distracting compared to Screen Dimming, Action Disable, or Sound Beeping, since the players do not perceive the shifting effect.

Overall, we hoped to show that Redirected Motion would be the most suitable interference avoidance technique, since it would be quite effective, yet not distracting.

5.5 Analysis

We ended up not using the data from the first study block (i.e., the second of the five blocks) because it was clear that several participants were perplexed by the perceptible techniques when they first encountered them. Removing the block mitigated learning curve bias for these first encounters. Therefore, we analyzed 27 game rounds (9 dyads × 3 study blocks) for each of the five technique conditions. We analyzed our results according to game duration (Figure 5.6a), mean distance between the participants (Figure 5.6b), Likert-scale ratings (Figure 5.7), and subjective ranking (Figure 5.8), using $\alpha = 0.01$ (0.05/5) after Bonferroni correction as our criterion for statistical significance.

5.5.1 Quantitative Analysis

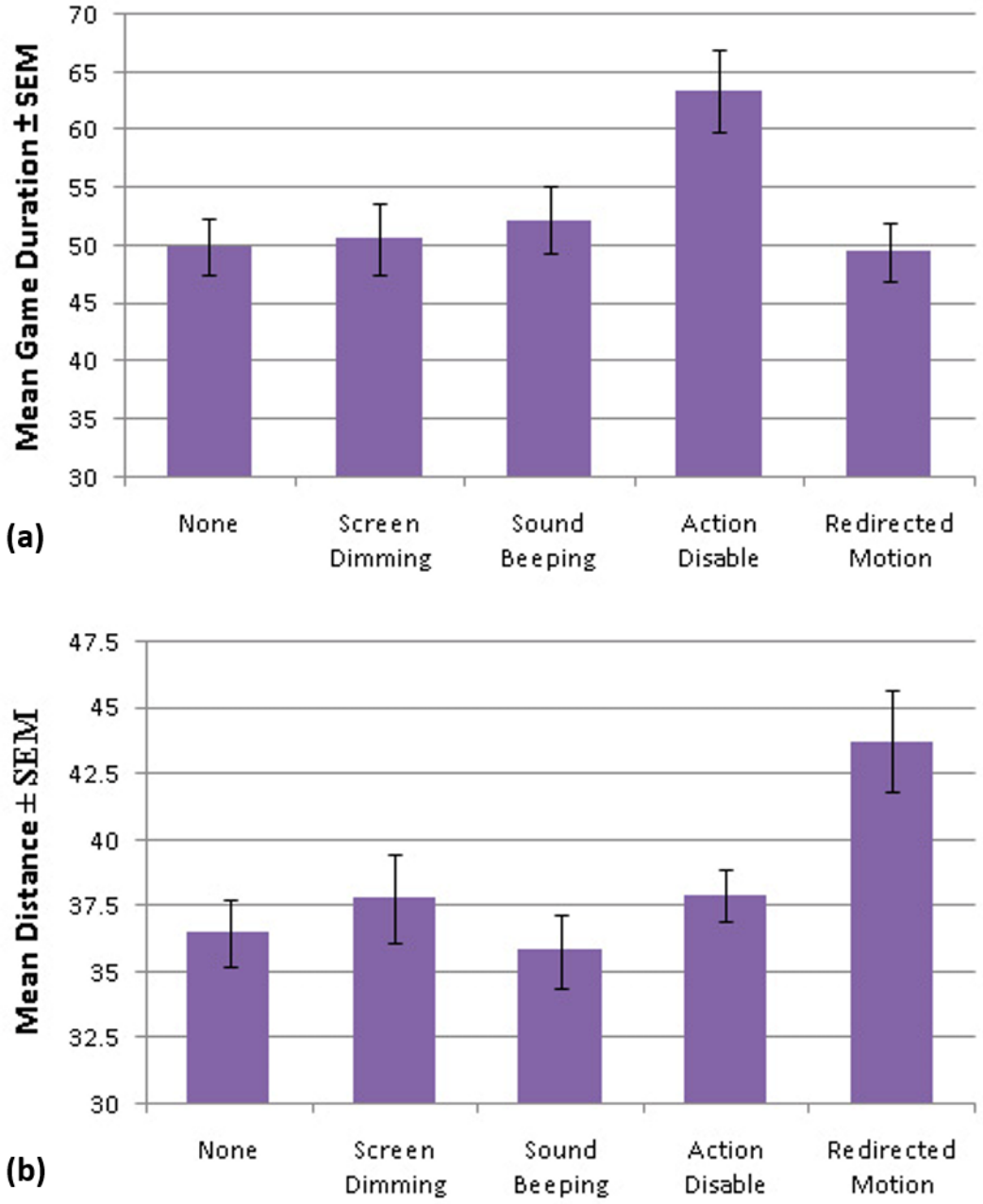

Figure 5.6. (a) Mean game duration with SEM (Standard Error of the Mean). (b) Mean distance between participants' devices with SEM.

A within-subjects one-way ANOVA shows that technique had a significant effect ($F_{(4,23)} = 5.382$, $p < 0.01$) on game duration (Figure 5.6a). A paired-sample *t* test between Redirected

Motion and Action Disable ($t_{(26)} = 3.611$, $p < 0.01$) shows that rounds played with Redirected Motion take significantly less time to complete compared to ones played with Action Disable, but there was no significant difference between Redirected Motion and other techniques. Paired-sample *t* tests confirm that rounds played with Action Disable took significantly longer than ones played with all other conditions—None: ($t_{(26)} = 4.628$, $p < 0.01$), Screen Dimming: ($t_{(26)} = 3.817$, $p < 0.01$), and Sound Beeping: ($t_{(26)} = 3.646$, $p < 0.01$). We believe that this occurs when action is disabled, because dominos could not be knocked off the board, effectively pausing game progress during that period. This increase in game duration is consistent with the high level of perceived distractibility for Action Disable shown in the Likert-scale results and rankings discussed in Section 5.5.2.

To quantitatively analyze technique effectiveness, we computed the mean distance between the closest points of the participants' devices during game play. Before analyzing the distance data, we cleared them by removing outliers and by ignoring the data collected during the four second countdown period at the beginning of each game round. Distances less than 1 cm or larger than 120 cm (1.5 times the diagonal of the table) were considered to be outliers because it is most likely that any distance outside of this range was caused by an (infrequent) tracking glitch. Tracking glitches could happen when the players were either so close to each other that a UMPC or body part could block the other player's UMPC camera or too far away from the marker array on the table to track it reliably. Outliers accounted for 0.7% of all distance data and were present for all five techniques: 0.21% of None, 0.21% of Screen Dimming, 2.7% of Sound Beeping, 0.31% of Action-Disable, and 0.6% of Redirected Motion. (There were more outliers for Sound Beeping than for other conditions because in one round played with Sound Beeping,

one of the participants moved quite far from the table, resulting in a relatively large number of tracking glitches.)

We asked the participants not to move from their initial positions at the opposite ends of the table until the four-second countdown finished, but several started to move closer to the board before the countdown ended, after they got accustomed to the game. Therefore, in order to remove this initial bias across the user study, we ignored the distance data collected during the countdown.

A within-subjects one-way ANOVA on the cleared data shows that technique had a significant effect ($F_{(4,23)} = 5.504$, $p < 0.01$) on distance between the participants over time (Figure 5.6b). Paired *t* tests between Redirected Motion and the other conditions showed that during Redirected Motion, participants maintained a significantly larger mean distance between them, especially comparing Redirected Motion and None, and Redirected Motion and Sound Beeping—None ($t_{(26)} = 3.946$, $p < 0.001$), Screen Dimming ($t_{(26)} = 3.414$, $p < 0.01$), Sound Beeping ($t_{(26)} = 4.462$, $p < 0.001$), Action Disable ($t_{(26)} = 2.854$, $p < 0.01$). This supports H4. However there was no significant difference among the comparison conditions.

That these other conditions did not significantly increase average distance between participants means that we did not quantitatively confirm the other hypotheses about effectiveness: H1–H3. Observations made during the user study indicate some possible explanations for the overall ineffectiveness of these perceptually obvious techniques. While some participants appeared to stay farther away from their opponent when the other techniques were used, other participants appeared to try to use these techniques strategically against their opponent. Although, we noted in Section 5.4.1 that the techniques only affected a participant who was actively moving (relative to the gameboard) toward their opponent, once both participants were within dis-

tance $\boldsymbol{\tau'}$, any motion toward the other by either participant would elicit the effect. Thus, an aggressive player could take advantage of the effect to try to limit the utility of their opponent moving in their direction. Furthermore, with the exception of Action Disable, a player could either ignore the beeping or plan an attack at a safe distance and then move in briefly to execute it with a dimmed screen.

5.5.2 Subjective Analysis

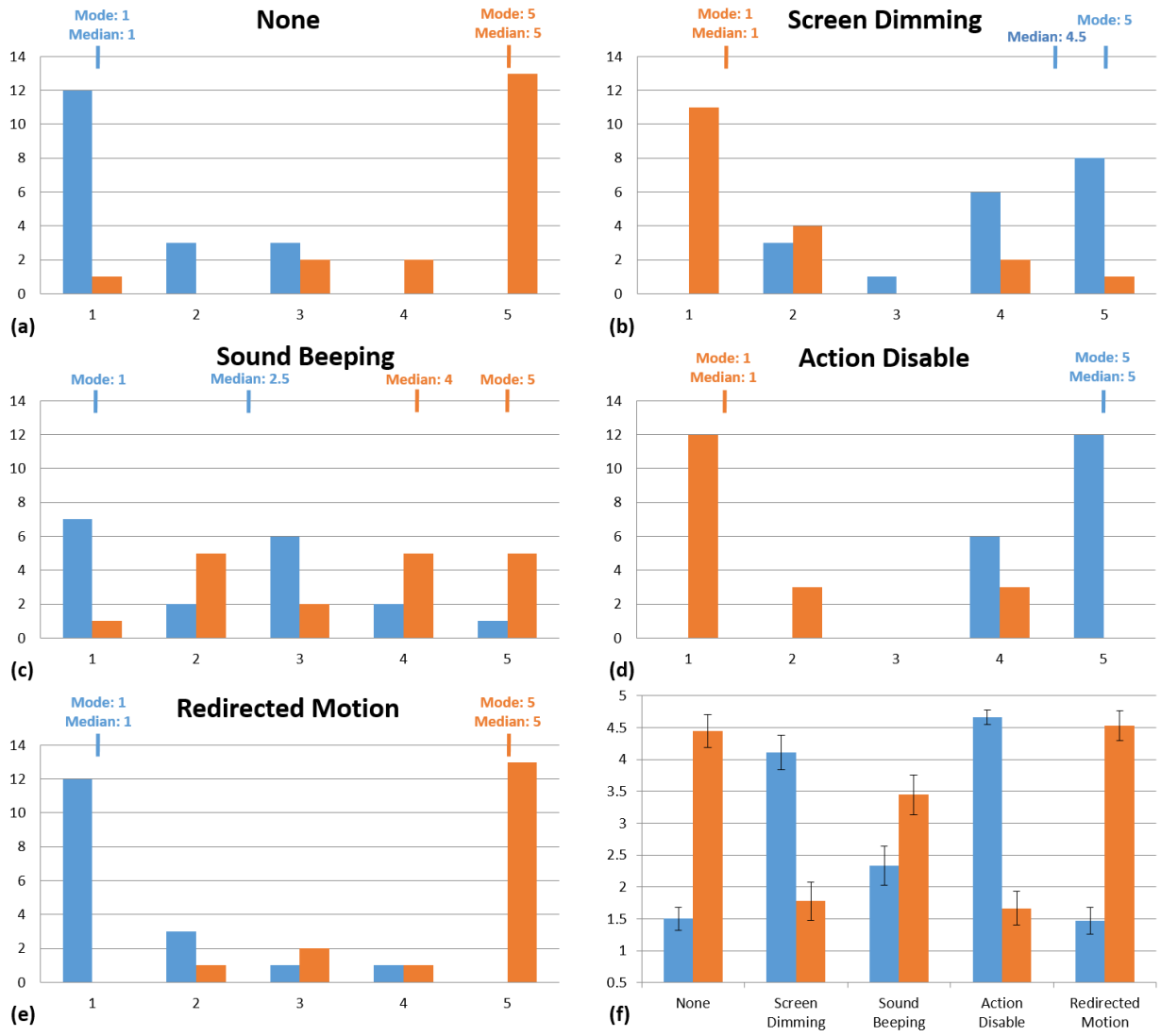

Figure 5.7. (a–e) Histograms of perceived effectiveness (blue) and distractibility (orange) Likert-scale ratings from 1–5 given by participants for five conditions with mode and median. (f) Means of perceived effectiveness (blue) and distractibility (orange) with SEM (Standard Error of the Mean). 5 is best.

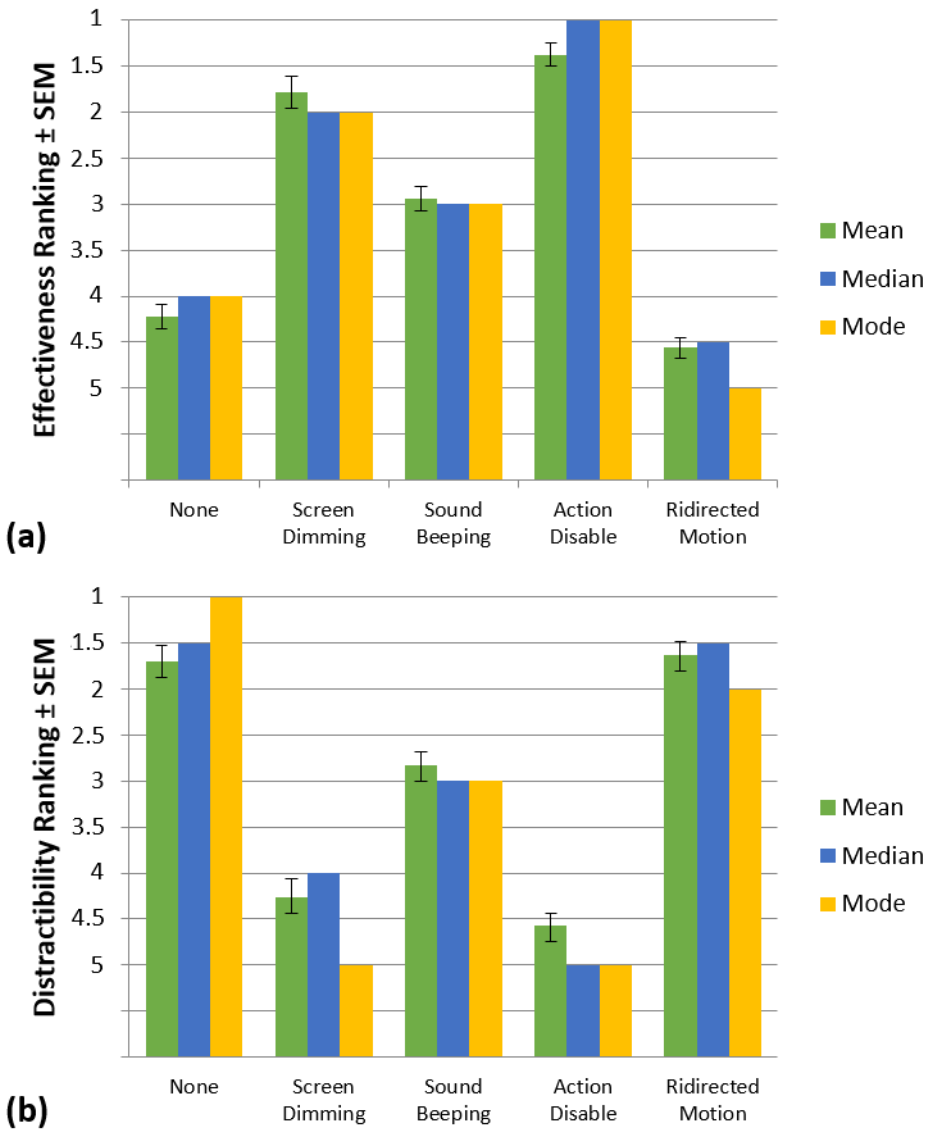

Figure 5.8. Ranked qualitative effectiveness (a) and distractibility (b) with SEM (Standard Error of the Mean) for the five technique conditions (including None). 1 is best (perceived to be most effective, least distracting).

	N-SD	N-SB	N-AD	N-RM	SD-SB	SD-AD	SD-RM	SB-AD	SB-RM	AD-RM
Effectiveness										
Z	3.55	3.05	3.55	1.60	2.97	1.44	3.53	3.59	3.53	3.57
p	.000	.002	.000	.109	.003	.151	.000	.000	.000	.000
Distractibility										
Z	3.42	2.80	3.44	0.27	2.89	0.91	3.37	3.45	3.25	3.43
p	.001	.005	.001	.791	.004	.361	.001	.001	.001	.001

Table 5.2. Paired Wilcoxon test results on perceived effectiveness (top) and distractibility, where N=None, SD=Screen Dimming, SB= Sound Beeping, AD=Action Disable, and RM=Redirected Motion.

A Friedman test shows that the distribution of the rankings (Figure 5.8) for perceived effectiveness ($\chi^2_{(4)} = 51.54$, $p < 0.001$) and distractibility ($\chi^2_{(4)} = 45.87$, $p < 0.001$) between the different techniques is significant. A Wilcoxon test (Table 5.2) indicates that None and Redirected Motion have significantly less perceived effectiveness and less distractibility compared to the other three techniques. Multiple participants mentioned "I didn't notice anything" for both techniques. Since most users did not perceive any difference between None and Redirected Motion, this does not confirm H1. Sound Beeping was evaluated to be significantly less effective, but less distracting compared to Screen Dimming or Action Disable, as shown in Table 5.2, which qualitatively supporting H2. Both Screen Dimming and Action Disable were evaluated as significantly more effective, but more distracting than the other conditions, as shown in Table 5.2, which qualitatively supports H3.

The Likert-scale results (Figure 5.7) show that participants found Screen-Dimming and Action-Disable technique to be quite effective, but very distracting. Sound Beeping was not perceived to be very effective, as participants mentioned that they could simply ignore the sound and keep playing. Participants thought that None and Redirected Motion were not distracting.

This supports H5. We addressed hypothesis H4 in our quantitative analysis (Section 5.5.1), since participants were not aware of the effect.

Even though most participants did not notice any difference between None and Redirected Motion, one participant said that he perceived a slight increase in gravity that caused the balls to land on the table closer to him compared to other techniques and another participant said that he noticed that the ball was fired slightly off from the location he tapped on the screen. These arguments are understandable, since dynamically changing the displacement of the virtual location from the physical location could cause these small discrepancies.

5.6 Limitations and Applications

As we pointed out in Section 5.2, Redirected Motion is not suitable for AR systems in which offsetting virtual objects from physical objects would violate requirements for accurate registration between those virtual and physical objects. These include applications in which virtual objects must be in close proximity to visible physical objects whose appearance would make the offset obvious. Since the application used in this experiment employed a patterned physical gameboard on which virtual objects were placed, we avoided this situation by using the well-known technique of overlaying the physical gameboard with a virtual texture. If the virtual objects were sufficiently distant from physical objects (e.g., if they were floating high enough above the gameboard), it would have been possible to avoid adding the virtual gameboard texture. Based on our pilot studies, we also established bounds on the amount by which a user's virtual location can be shifted relative to their physical location, and the speed with which the shift is made.

Even though we have experimented with Redirected Motion in a multi-user gaming scenario, we believe it is applicable to a remote task assistance scenario, such as that explored in

Chapter 4. In a scenario in which multiple co-located remote experts guide one or more local users, we can apply Redirected Motion to the experts' shared environment. In either DEMO3D or POINT3D, each expert interacts with virtual objects that do not correspond to physical objects in the experts' shared environment. Therefore, the virtual objects can be shifted without violating the requirements while maintaining a comfortable distance among the co-located experts. We will provide an example of applying Redirected Motion in a scenario with multiple experts in Section 6.3.3.

In addition, consider a collaborative AR environment in which co-located users design a product by constructing virtual parts and assembling them in a shared space. When multiple users work on the same part or assembly, Redirected Motion could make it possible for one user to view and interact with it from a virtual location that would be uncomfortably close to (or even physically blocked by) another user. In our user study, we intentionally tried to avoid making participants aware that Redirected Motion was being used. In a collaborative AR environment, however, we might want all users to know when a user's physical and virtual locations were different. This might be accomplished by offering the ability to view the other users' virtual locations as ghosted overlays similar to those used in Figure 5.2.

Chapter 6. Conclusions and Future Work

This dissertation has developed and evaluated novel AR interaction techniques to assist collaborating users in performing referencing tasks more accurately in physically shared environments and more efficiently in remote environments. Furthermore, we explored techniques for avoiding undesired physical interference among co-located users. The preceding chapters have provided detailed discussions of the design, implementation, and comparative study of our new approaches. In this chapter, we first summarize the contributions of our work in Section 6.1. We then provide some lessons learned for developing multi-user AR applications in Section 6.2. Finally, we conclude with a discussion of potential future work and some final thoughts about referencing and interference avoidance in multi-user environments in Section 6.3–6.4.

6.1 Summary of Contributions

We tried to answer the following research questions: How can co-located users accurately communicate items of interest to each other in AR? How can remote users efficiently communicate items of interest and intents to each other in AR? How can co-located users avoid physically interfering with each other in AR? In addressing these questions, we made the following contributions:

Design, implementation, and evaluation of a referencing technique for physical objects in co-located multi-user AR (Chapter 3). We developed a referencing technique, GARDEN, designed to be used in situations when it is difficult or inappropriate for users to walk to a target object and point at it directly. A formal study revealed that GARDEN performed significantly

more accurately than both existing AR referencing techniques and the use of a physical laser pointer in situations in which the participating users have sufficiently different views of the scene, and significantly more accurately than one of the comparison techniques in situations in which the participating users have similar perspectives.

Design, implementation, and evaluation of a referencing technique for supporting remote task assistance in AR (Chapter 4). We developed DEMO3D, a 3D interaction and visualization approach in which a remote expert, using virtual replicas of physical objects, demonstrates to a local user how to perform a 6DOF alignment task. A formal study showed that DEMO3D performed faster than an approach in which the remote expert specifies corresponding pairs of points to align on a pair of objects (POINT3D) and a 2D sketch-based approach (SKETCH2D) for both study participants playing the role of remote expert and a highly trained expert. A qualitative survey showed that DEMO3D was preferred over SKETCH2D by participants performing as local users, and participants acting as remote experts felt DEMO3D was faster and better than SKETCH2D.

Design, implementation and evaluation of a physical-interference–avoidance technique in co-located multi-user AR (Chapter 5). We developed a physical interference avoidance technique, *Redirected Motion*, which transforms the shared 3D space in which the user moves a handheld computer, to direct it away from those of others. We evaluated this technique within an instrumented two-player game in which co-located players used their handheld computers to shoot at a set of virtual objects in a shared volume. Our evaluation revealed that the mean distance between users using Redirected Motion was significantly larger than when using comparison conditions, which included an unmanipulated control condition, dimming the display, playing disturbing sounds, and disabling interaction capabilities. Subjective evaluation indicated that

Redirected Motion was essentially unnoticeable in comparison with the unmanipulated control condition. Participants found both of these conditions significantly less effective than the other conditions (despite the quantitative success of Redirected Motion) and significantly less distracting than the other conditions, and could not perceive any statistically significant differences in effectiveness and distractibility between Redirected Motion and the unmanipulated control condition.

6.2 Lessons Learned

We have learned several worthwhile lessons throughout the development and evaluation of our multi-user AR techniques for referencing and interference avoidance.

Interaction on a virtual copy is effective for both selection/instruction and interpretation. Inspired by voodoo dolls interaction [Pierce et al., 1999], our referencing techniques for local environments (Chapter 3) and remote environments (Chapter 4) leveraged the use of virtual copies. GARDEN (Chapter 3) successfully showed that the indicator was able to communicate a referenced physical object accurately to a recipient using a virtual copy, and that the same virtual copy can be used to help the recipient understand the relationship between the copy and its corresponding physical counterpart. POINT3D and DEMO3D (Chapter 4) further confirmed the effectiveness of using a virtual copy to instruct a remote user and help that user interpret the instructed 6DOF pose. We believe the use of virtual copies is effective because it allows the indicator/instructor to closely interact with and view a copy from a perspective that is otherwise impossible or very difficult to achieve with its virtual proxy or physical counterpart.

People prefer to interact from their own perspective rather than from a view through the eyes of another. We allowed the remote expert to instruct either from their own perspective or through the eyes of the local user (Chapter 4). Even though we did not conduct a formal study on

user preferences for perspective, our pilot studies revealed that the participants who played the role of expert tended to interact from their own perspective rather than through the eyes of the local user. Most of those participants who switched to the local user's perspective quickly switched back to their own perspective, meaning that they switched by accident rather than by intention. This may in part due to our environmental setting and technological limitation. We provided accurate models of the domain objects in the local user's environment so that the remote expert did not need to see through the eyes of the local user to check if there are any discrepancies between the virtual proxies and the real environment. Additionally, the technology we used had limitations in terms of video resolution (640×480 resolution for the AR view through the eyes of the local user compared to 1280×720 resolution for the remote expert's VR view), temporal latency of the video streaming, and brightness of the video. While these confounds may exist, the participants' behavior led us to infer that the participants preferred to interact from their own perspective. This observation suggests that, where possible, a multi-user AR application should provide the remote user a way to interact from their own perspective with approaches similar to that of Tatzgern and colleagues [Tatzgern et al., 2014] or Gauglitz and colleagues [Gauglitz et al., 2014].

3D direct interaction may not always yield better performance than simple 2D interaction for a 3D task. In Chapter 4, the study result showed that POINT3D was slower than SKETCH2D for recruited participants, even though both techniques required specifying six contact points. This is due in part to the bulky trackable attached to the interaction devices, but we also suspect that our well-designed 2D interaction condition and the familiarity of 2D interaction compared to direct 3D interaction for an ordinary person contributed to this result. In addition, our preliminary experimentation with remote referencing techniques [Oda et al., 2013] revealed

that if an expert can navigate the local user to a perspective where a 3D selection task becomes a 2D planar selection task, 2D interaction performs faster and more accurately than 3D interaction. When designing a 3D direct interaction technique in AR, a developer should first consider whether a 2D alternative may outperform it, depending on the nature of the task and environment.

Even if the virtual contents are not registered with the physical environment, it can still be beneficial to view them in AR. In our study of remote collaboration, recruited participants who played the role of expert had difficulties performing bimanual interaction while not being able to see their own hands and the interaction devices they are holding (Chapter 4). This leads us to believe that the participants would have been benefited from the ability to see an AR view through their head-worn displays even if the virtual contents with which they are interacting were not registered with their physical environment.

People are less sensitive to virtual shifting while moving. Redirected walking [Razzaque et al., 2001; Steinicke et al., 2008] proved that people are less sensitive to rotational injection while rotating their head in a VR environment. In Chapter 5, we successfully showed that people are less sensitive to virtual shifting while moving in an AR environment if we can mitigate the visual discrepancy between the virtual contents and the physical environment.

6.3 Future Work

We have successfully shown the effectiveness of our approaches for referencing and interference avoidance in multi-user AR environment, but there are potentials to further improve these approaches. We outline several of those possible future improvements below.

6.3.1 Hand Tracking

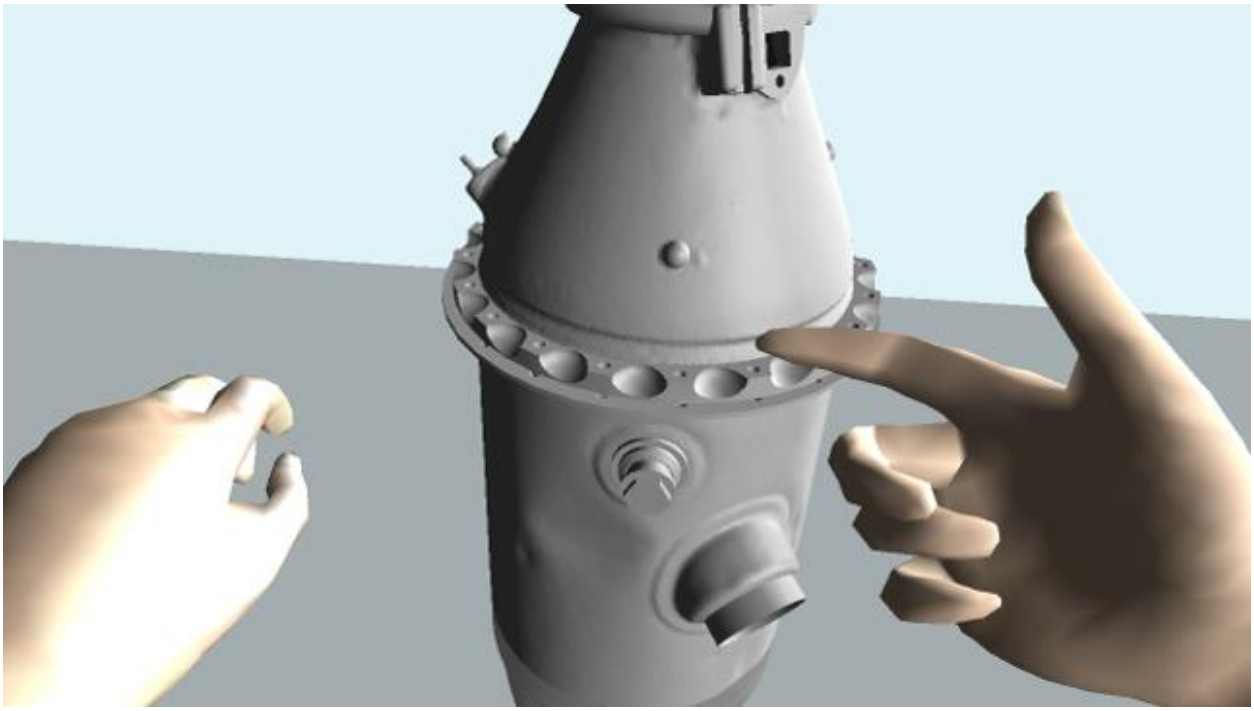

Figure 6.1. An expert points at the top part of an aircraft engine combustion chamber with one of two virtual hands tracked using the 3Gear Systems [3GearSystems, 2011] hand tracker.

GARDEN (Chapter 3) was shown to be more accurate than the comparison techniques, but failed to perform faster than or at least as fast as those techniques. We suspect that this is due in part to the poorly reconstructed hand model created from the depth mesh, which had an update rate of roughly 10 frames per second and included only the side of the hand that was visible to the depth camera at that point in time. These limitations significantly disadvantaged the speed of GARDEN for both selection by the indicator and interpretation by the recipient. As briefly discussed in Section 4.5, we have experimented with a state-of-the-art commercial hand tracker (3Gear Systems [3GearSystems, 2011]) for POINT3D and DEMO3D as shown in Figure 6.1, but it had limited tracking range and low accuracy for recognizing certain gestures. Once hand-tracking with improved stability and range becomes available, GARDEN will greatly improve both in accuracy and speed with its complete hand model estimation and both POINT3D and

DEMO3D can be easily adapted to use bare hands instead of hand-held interaction devices for more natural interaction.

6.3.2 Dynamic Scene Reconstruction

We are interested in employing a dynamic scene reconstruction approach similar to that of Gauglitz and colleagues [Gauglitz et al., 2014] both for improving the speed and accuracy of GARDEN (Chapter 3) with better reconstructed environmental models for selection and interpretation and for extending the DEMO3D approach (Chapter 4) to be applicable in an environment that requires less groundwork for modeling and tracking the domain objects. If a sufficiently good system for interactively capturing, tracking, and reconstructing scene geometry were available, a local user could scan the domain objects through a wearable depth sensor to generate virtual proxies and track the scanned objects.

6.3.3 Applying Redirected Motion to Remote Assistance

As discussed in Section 5.6, we see great potential in incorporating Redirected Motion in our remote assistance techniques (Chapter 4), especially in a scenario where there are multiple co-located experts, probably with different specialties, collaboratively instructing a remote user. In this scenario, we assume that the experts see the environment in AR so that they can see the virtual proxies and virtual replicas as well as their own hand for interaction. We also assume that the experts' hands are tracked with a wearable depth camera embedded in their head-worn displays for interacting with the virtual objects using their bare hands. Redirected Motion could be applied based on the position of each expert's tracked hands to avoid interference among them when interacting with the virtual proxies and replicas.

Recall our argument that people are less sensitive to virtual shifting when they are moving. Because the expert's view may not change when the virtual shift is applied, since the shift is

based on the position of their hands, but not their head-worn display or head position, the expert may become aware of the effect. Even if the effect is noticeable, we believe this won't be an issue as long as we keep the grabbed virtual replica static relative to the interacting hand and apply the shifting to virtual proxies and released virtual replicas. This will avoid registration discrepancies between the real hand and the grabbed virtual replica.

6.3.4 Extending Remote Assistance for Instructing Multiple Local Users

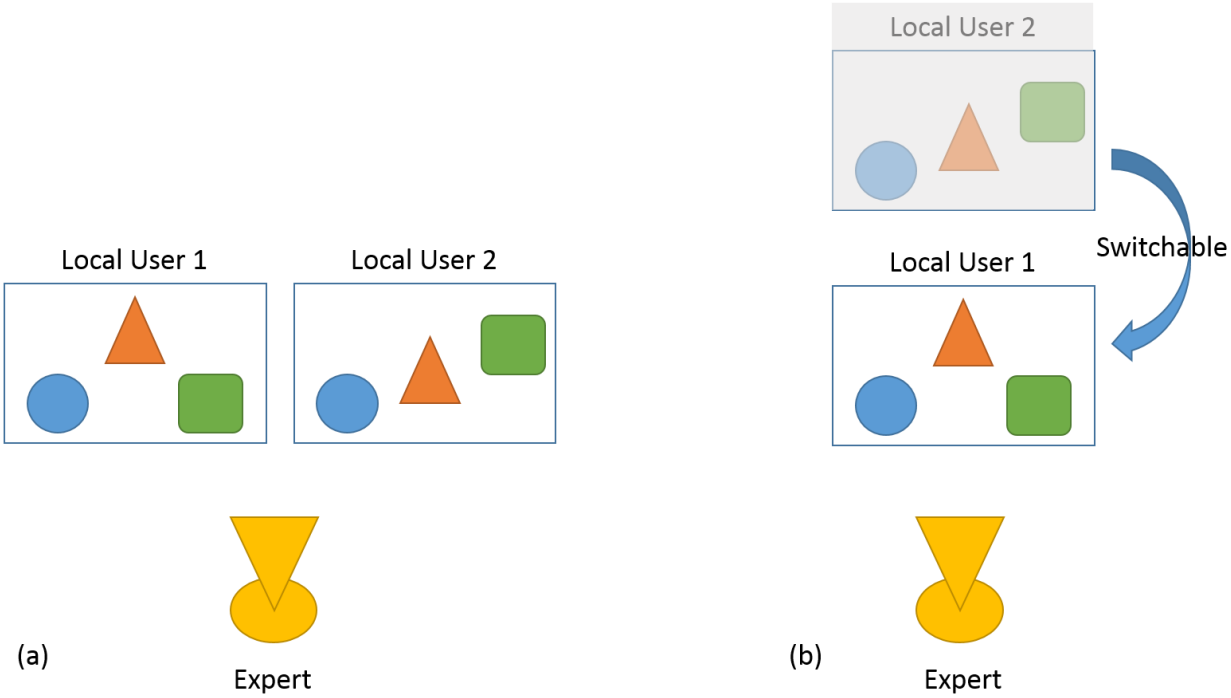

Figure 6.2. Blue circle, red triangle, and green rounded rectangle represent different objects in the local user's domain as virtual proxies. (a) An expert views and interacts with multiple local users' environments side-by-side. (b) An expert views and interacts with one local user's environment at a time, and switches between different local user's environments.

We see opportunities to extend our remote assistance technique to allow the remote expert to instruct more than one local user, potentially located at different sites. Unlike co-located assistance, which requires the expert to be physically present at the site, remote assistance provides the flexibility to support more than one local user at the same time. We propose possible

extensions of our techniques for two scenarios below. For each scenario, we assume that our system provides the expert with either a side-by-side (Figure 6.2a) or a switchable (Figure 6.2b) configuration of the local users' layout, which contains the virtual proxies of the domain objects.

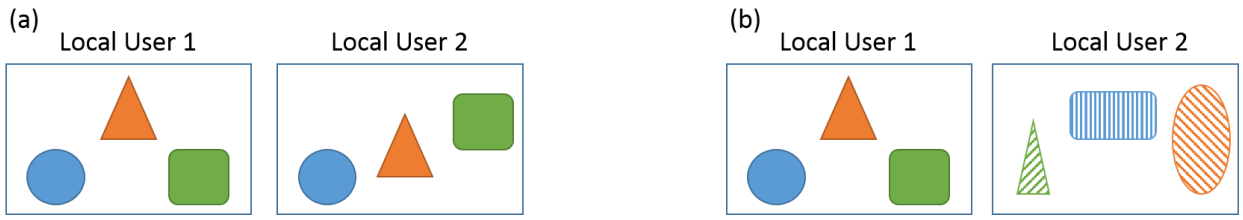

Figure 6.3. (a) All local users' environments have the same domain objects. (b) Each local user's environment has different domain objects.

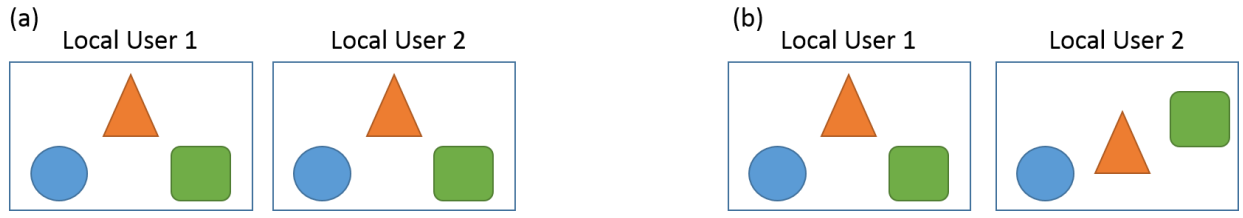

Figure 6.4. (a) All local users' environments have the same initial layout. (b) Each local user's environment has a different initial layout.

In the first scenario, the expert instructs the local users with the same domain objects (Figure 6.3a). If the initial layout is the same for each local user (Figure 6.4a), the expert can provide instructions with single layout, and each local user will see the instructions at their site. As a local user starts manipulating the physical counterparts, their layout will start to differ, so it will be necessary for the expert to provide additional advice for each local user, as needed. When the initial layout is different for each local user (Figure 6.4b), the expert will need to provide individual instructions from the beginning.

In the second scenario, the expert instructs the local users with different domain objects (Figure 6.3b). Since there are no correlations among the local users' environment, the expert will need to provide all instructions individually.

6.3.5 Allowing Remote Expert to Define Constraints

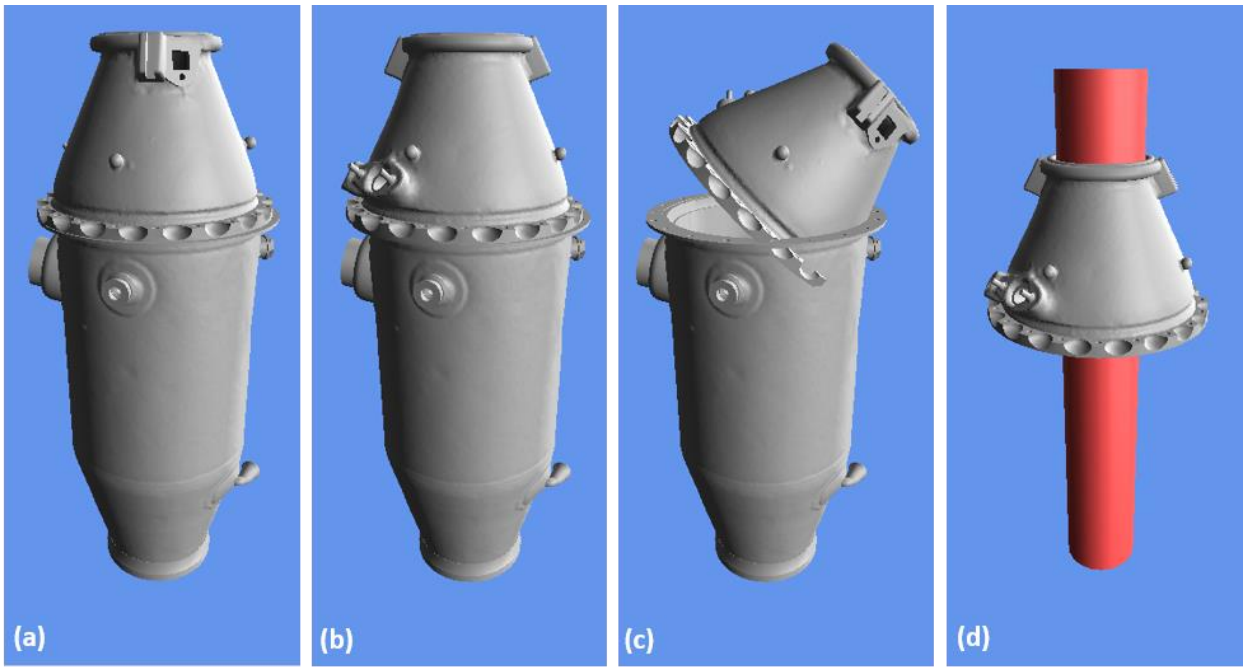

Figure 6.5. (a–b) To restrict the top to 1DOF rotation along its "up" axis, an expert can define a constraint by placing the top above the bottom and indicating it can rotate only about its "up" axis, which allows these configurations. (c) However, this configuration would be prevented. (d) In one alternative, the expert can constrain the top to slide along and rotate around the red cylinder, restricting it to 1DOF translation and 1DOF rotation.

We briefly explained the concept of how an expert can define constraints for our DEMO3D technique (Section 4.2.1), but did not provide an implementation—constraints were hardwired for our user study. In this section, we describe two possible implementations depending on the resources that are available in the expert's environment.

In many real assembly tasks, two rigid parts fit together in a constrained way with some leeway in translation or orientation (Figure 6.5a–c). The remote expert could orchestrate a set of rigid-body constraints, prior to guiding the local user, by placing the virtual replica at a location (or a region for a translational constraint) and specifying the DOFs the virtual replica can have (Figure 6.5d). When giving guidance, the expert could fine-tune the position and/or orientation within the DOFs allowed. This constraint-based approach would not only prevent unwanted in-

terpenetration, but would also reduce potential manipulation errors introduced while fine-tuning the final 6DOF pose near the constrained region (Figure 4.3a).

In a scenario in which the relevant physical objects in the local user's environment are available in the expert's environment, the expert can use them directly to specify the constraints assuming the 6DOF position and orientation of those physical objects are tracked. This system could be implemented in an AR environment in which the expert wears a head-worn display and holds an untracked controller. Continuing our example using the aircraft engine combustion chamber top and bottom, the expert would physically grab the top chamber with her hand and hold it above the bottom chamber so that the holes on the top chamber will align with the holes on the bottom chamber when rotated around the "up" axis of the top chamber. Once aligned properly, the expert will then use the controller to enable the 3D visualization overlaid on the 6DOF-tracked top chamber, showing a typical graphical representation of the coordinate system (*x*, *y*, and *z* translation axes and yaw, pitch, and roll orientation axes). The six axes are defaulted to be constrained, and the expert can specify which axes can be changed during the actual instruction using the controller. For example, to specify a constraint depicted in Figure 6.5(d) in which the chamber top is constrained to its "up" axis rotation as well as one degree of translation along the red cylinder, the expert would first fit the physical chamber top with the physical red cylinder, assuming there is enough friction between them or some latches that can hold them together. Once fit, the expert can define the up-vector rotation axis as flexible, and specify that it can translate along its up-vector axis, possibly with a range so that it won't translate beyond the height of the red cylinder.

In a situation in which the relevant physical objects are not available in the expert's environment, but the virtual proxies are available, the expert can specify constraints using virtual

proxies. This system could be implemented in a VR environment in which the expert wears a head-worn display and holds a 6DOF-tracked controller. Again, using the chamber top and bottom as an example, the expert would grab the virtual proxy of the chamber top using the tracked controller and place it above the virtual proxy of the chamber bottom. Recall the issues we encountered for DEMO3D due to lack of haptic feedback while aligning two virtual objects, namely interpenetration and difficulty aligning the touching face with sufficient accuracy. The expert will face the same issues in this situation.

While it is not desirable to use physics simulation during actual instruction, as discussed in Section 4.2.1, it can be used during constraint specification in which there is no time restriction. The virtual proxy will not be influenced by the physics simulation while held and manipulated by the expert, but will be affected as soon as the expert releases it. The physics simulation will resolve any interpenetration states and force the faces of the chamber top and bottom to correctly align under the influence of gravity. This process may require several trials until the expert reaches a stage where the holes between the chamber top and bottom are correctly aligned due to side-effects of physics simulation (e.g., when getting out of an interpenetration state, the virtual proxies are pushed away from each other causing the chamber top and bottom to no longer be correctly aligned). Once the virtual proxy of the chamber top and bottom are correctly aligned, the expert would perform the same steps as in the above scenario to define the constraints. Continuing on our example for Figure 6.5(d), the expert can turn off the physics simulation once the expert virtually fits the chamber top with the red cylinder to avoid the chamber top from sliding down. Then, the expert can specify its rotational and translation constraint as described in the above scenario.

6.3.6 DEMO3D and POINT3D in a Partially or Unmodeled Environment

As described in Section 4.2, our techniques assume that manipulable physical objects have been modeled and are tracked in 6DOF. This can be practical in many controlled scenarios, as most items are now designed and manufactured using 3D CAD models, which could be used as virtual proxies. When that is not feasible, a dynamic scene reconstruction algorithm could be used (e.g., [Gauglitz et al., 2014; Kim et al., 2014; Newcombe et al., 2011; Tatzgern et al., 2014]), as briefly discussed in Section 6.3.2. To ensure work objects can be distinguished from each other, they could be introduced one at a time and segmented by analyzing the differences between previous and current scans. Based on the particular setting, other tracking solutions could replace the NaturalPoint OptiTrack (e.g., feature-based or model-based vision, or SLAM).

Our approach can also be applied in a partially tracked and modeled environment. DEMO3D already allows an expert to demonstrate how to fit a single modeled and tracked part (e.g., the chamber top) into a *static* unmodeled environment, albeit without constraints. Because the expert can elect to see the local user's environment in stereo AR (Figure 4.1), they can direct the local user to look at specific places in the environment and demonstrate using the virtual replica of the single tracked and modeled part relative to the static environment, relying on depth cues from the stereo image pair. POINT3D could also be used in such an environment with a small modification: Adding annotations is currently done by pointing (with raycasting to the closest intersection), requiring a modeled environment; instead, raycasting could be easily replaced with direct 3D placement. Alternatively, if the local user's head-worn display includes a depth camera, POINT3D could work unmodified.

6.3.7 Exploring Additional Opportunities for Redirected Motion

In Chapter 5, we studied two-player hand-held AR interactions. A more general equation may need to be devised for more than two users. However, it would be interesting to see whether our current equation would work well for cases with more than two users, since the shifting equation depends only on whether a user is moving toward another user within a distance threshold. Furthermore, even though Redirected Motion is designed for avoiding interference between users, we also believe that it could be used to avoid passive obstacles.

For our distance thresholds $\boldsymbol{\tau}$ and $\boldsymbol{\tau'}$, we used a single constant value for the entire duration of the application. However, depending on the orientation of the handheld device (and its integral camera), the distance to be considered as too close may change because the player's physical body is asymmetric relative to the handheld device and its camera. For example, if the users are facing each other, then since their bodies are typically behind their cameras, $\boldsymbol{\tau'}$ can be smaller. However, if the users are next to each other with their displays approximately coplanar, then the players' arm lengths need to be considered in addition to the geometry of the handheld devices themselves, so $\boldsymbol{\tau'}$ should be larger. Thus, it may be more appropriate to make $\boldsymbol{\tau}$ and $\boldsymbol{\tau'}$ vary depending on the device orientation.

We did not explore rotational gain, in the spirit of redirected walking [Razzaque et al., 2001], since we speculated that it would not provide as much of a distance buffer as translational gain; however, for certain situations, especially when the players do not need to move much, rotational gain may work better than translational gain. For example, if each player is next to the other and required to rotate their body frequently and extend their arms to manipulate the overlaid virtual contents, then rotational gain might prevent the players from physically interfering with each other.

In our experimental setting, participants did not notice that they are being relocated, due to the subtlety of the translational gain. In future work, we would like to try to determine with more accuracy than our initial pilot studies, how much gain can be applied before users recognize the shifting, as Steinicke and colleagues [Steinicke et al., 2008] analyzed for redirected walking. The amount of unrecognizable gain may vary depending upon a variety of factors, including the task domain and aspects of the physical environment being augmented such as brightness; which can be an interesting research direction to investigate further. The more translational gain we can apply without the user noticing, the larger the distance we can shift the users. We may be able to keep users even farther apart from each other, assuming we also increase $\boldsymbol{\tau}$, as well as increase the range of shifting.

6.4 Final Thoughts

This dissertation has explored approaches for improving the effectiveness of referencing and reducing the risk of physical interference in multi-user AR environments. While our approaches did not excel on every metric with which they were evaluated against comparison techniques, we are confident that our approaches have many advantages, especially as the underlying software and hardware with which they are implemented improve. As the relevant technologies advance in AR, we expect to see more and more applications deploy in multi-user settings rather than single-user ones, in many domains, including, but not limited to, entertainment and task assistance. We hope that our work and findings will motivate other researchers to explore more ways to support and enhance effective multi-user AR experiences.

References

3GearSystems. (2011). http://www.threegear.com

Adcock, M., Ranatunga, D., Smith, R., and Thomas, B. H., "Object-based touch manipulation for remote guidance of physical tasks," *Proceedings of the 2nd ACM symposium on Spatial user interaction*, 2014, pp. 113-122.

Agrawala, M., Beers, A. C., McDowall, I., Fröhlich, B., Bolas, M., and Hanrahan, P., "The two-user Responsive Workbench: support for collaboration through individual views of a shared space," *Proceedings of the 24th annual conference on Computer graphics and interactive techniques*, 1997, pp. 327-332.

Argelaguet, F., Kulik, A., Kunert, A., Andujar, C., and Froehlich, B., "See-through techniques for referential awareness in collaborative virtual reality," *International Journal of Human-Computer Studies*, 2011, vol. 69, no. 6, pp. 387-400.

Asama, H., Ozaki, K., Itakura, H., Matsumoto, A., Ishida, Y., and Endo, I., "Collision Avoidance among Multiple Mobile Robots Based on Rules and Communication," *IEEE/RSJ International Workshop on Intelligent Robots and Systems*, 1991, vol. 3, pp. 1215-1220.

Avery, B., Sandor, C., and Thomas, B. H., "Improving spatial perception for augmented reality X-Ray vision," *Virtual Reality*, 2009, pp. 79-82.

Azuma, R., Baillot, Y., Behringer, R., Feiner, S., Julier, S., and MacIntyre, B., "Recent Advances in Augmented Reality," *IEEE Computer Graphics and Applications*, 2001, vol. 21, no. 6, pp. 34-47.

Azuma, R. T., "A Survey of Augmented Reality," *Presence: Teleoperators and Virtual Environments*, 1997, vol. 6, no. 4, pp. 355-385.

Baldauf, M. and Fröhlich, P., "The augmented video wall: multi-user AR interaction with public displays," *CHI '13 Extended Abstracts on Human Factors in Computing Systems*, 2013, pp. 3015-3018.

Barakonyi, I., Fahmy, T., and Schmalstieg, D., "Remote collaboration using Augmented Reality Videoconferencing," *Graphics Interface*, 2004, pp. 89-96.

Barakonyi, I., Weilguny, M., Psik, T., and Schmalstieg, D., "MonkeyBridge: autonomous agents in augmented reality games," *ACM SIGCHI International conference on Advances in computer entertainment technology*, 2005, pp. 172-175.

Barnum, P., Sheikh, Y., Datta, A., and Kanade, T., "Dynamic seethroughs: Synthesizing hidden views of moving objects," *Proceedings of the 8th IEEE International Symposium on Mixed and Augmented Reality*, 2009, pp. 111-114.

Bauer, M., Kortuem, G., and Segall, Z., ""Where Are You Pointing At?" A Study of Remote Collaboration in a Wearable Videoconference System," *Proceedings of the 3rd IEEE International Symposium on Wearable Computers*, 1999, p. 151.

Benko, H., Ishak, E. W., and Feiner, S., "Collaborative Mixed Reality Visualization of an Archaeological Excavation," *Proceedings of the 3rd IEEE/ACM International Symposium on Mixed and Augmented Reality*, 2004, pp. 132-140.

Benko, H., Ishak, E. W., and Feiner, S., "Cross-dimensional gestural interaction techniques for hybrid immersive environments," *Virtual Reality*, 2005, pp. 209-216.

Benko, H., Jota, R., and Wilson, A., "MirageTable: freehand interaction on a projected augmented reality tabletop," *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, 2012, pp. 199-208.

Billinghurst, M. and Kato, H., "Collaborative augmented reality," *Communications of the ACM*, 2002, pp. 64-70.

Billinghurst, M., Weghorst, S., and Furness, T., "Shared Space: an augmented reality approach for computer supported cooperative work," *Virtual Reality*, 1998, vol. 3, no. 1, pp. 25-36.

Björk, S., Falk, J., Hansson, R., and Ljungstrand, P., "Pirates! Using the Physical World as a Game Board," *Interact*, 2001, pp. 9-13.

Bolt, R. A., "Put-that-there: Voice and gesture at the graphics interface," *SIGGRAPH Computer Graphics*, 1980, vol. 14, no. 3, pp. 262-270.

Bottecchia, S., Cieutat, J.-M., and Jessel, J.-P., "T.A.C: augmented reality system for collaborative tele-assistance in the field of maintenance through internet," *Proceedings of the 1st Augmented Human International Conference*, 2010, pp. 1-7.

Bowman, D. A., Johnson, D. B., and Hodges, L. F., "Testbed evaluation of virtual environment interaction techniques," *Proceedings of the ACM symposium on Virtual reality software and technology*, 1999, pp. 26-33.

Bowman, D. A., Kruijff, E., Joseph J. LaViola, J., and Poupyrev, I., *3D User Interfaces: Theory and Practice*: Addison-Wesley, 2004.

Breen, D. E., Whitaker, R. T., Rose, E., and Tuceryan, M., "Interactive Occlusion and Automatic Object Placement for Augmented Reality," *Computer Graphics Forum*, 1996, pp. 11-22.

Broll, W., Lindt, I., Ohlenburg, J., Wittkämper, M., Yuan, C., Novotny, T., Schieck, A. F. g., Mottram, C., and Strothmann, A., "ARTHUR: A Collaborative Augmented Environment for Architectural Design and Urban Planning," *Journal of Virtual Reality and Broadcasting*, 2004, vol. 1, no. 1, pp. 1-9.

Broll, W., Meier, E., and Schardt, T., "The virtual round table - a collaborative augmented multi-user environment," *Proceedings of the third international conference on Collaborative virtual environments*, 2000, pp. 39-45.

Brown, B. and Bell, M., "CSCW at play: 'there' as a collaborative virtual environment," *Proceedings of the 2004 ACM conference on Computer supported cooperative work*, 2004, pp. 350-359.

Bryson, S. and Levit, C., "The virtual windtunnel: an environment for the exploration of three-dimensional unsteady flows," *IEEE Visualization*, 1991, pp. 17-24.

Burns, M. and Finkelstein, A., "Adaptive cutaways for comprehensible rendering of polygonal scenes," *ACM Transactions on Graphics*, 2008, vol. 27, no. 5, pp. 1-7.

Cai, C., Yang, C., Zhu, Q., and Liang, Y., "Collision Avoidance in Multi-Robot Systems," *IEEE International Conference on Mechatronics and Automation*, 2007, pp. 2795-2800.

Cao, X., Forlines, C., and Balakrishnan, R., "Multi-user interaction using handheld projectors," *Proceedings of the 20th annual ACM symposium on User interface software and technology*, 2007, pp. 43-52.

Carpendale, M. S. T., Cowperthwaite, D. J., and Fracchia, F. D., "Distortion viewing techniques for 3-dimensional data," *IEEE Symposium on Information Visualization*, 1996, pp. 46-53.

Cashion, J., Wingrave, C., and Jr., J. J. L., "Dense and Dynamic 3D Selection for Game-Based Virtual Environments," *IEEE Transaction on Visualization and Computer Graphics*, 2012, vol. 18, no. 4, pp. 634-642.

Chastine, J., Nagel, K., Zhu, Y., and Hudachek-Buswell, M., "Studies on the Effectiveness of Virtual Pointers in Collaborative Augmented Reality," *Proceedings of the 2008 IEEE Symposium on 3D User Interfaces*, 2008, pp. 117-124.

Chastine, J. and Zhu, Y., "The cost of supporting references in collaborative augmented reality," *Proceedings of graphics interface*, 2008, pp. 275-282.

Chastine, J. W., "On Inter-referential Awareness in Collaborative Augmented Reality," Computer Science, Georgia State University, 2007.

Chastine, J. W., Nagel, K., Zhu, Y., and Yearsovich, L., "Understanding the design space of referencing in collaborative augmented reality environments," *Proceedings of Graphics Interface*, 2007, pp. 207-214.

Chastine, J. W., Zhu, Y., and Preston, J. A., "A Framework for Interreferential Awareness in Collaborative Environments " *IEEE International Conference on Collaborative Computing*, 2006, pp. 1-5.

Cheok, A. D., Goh, K. H., Liu, W., Farbiz, F., Fong, S. W., Teo, S. L., Li, Y., and Yang, X., "Human Pacman: a mobile, wide-area entertainment system based on physical, social, and ubiquitous computing," *Personal and Ubiquitous Computing*, 2004, pp. 71-81.

Cheok, A. D., Yang, X., Ying, Z. Z., Billinghurst, M., and Kato, H., "Touch-Space: Mixed Reality Game Space Based on Ubiquitous, Tangible, and Social Computing," *Personal and Ubiquitous Computing*, 2002, vol. 6, no. 5-6, pp. 430-442.

Clark, H. H. and Brennan, S. E., "Grounding in Communication," *Perspectives on socially shared cognition*, 1991, pp. 127-149.

Cooper, N., Keatley, A., Dahlquist, M., Mann, S., Slay, H., Zucco, J., Smith, R., and Thomas, B. H., "Augmented Reality Chinese Checkers," *ACM SIGCHI International conference on Advances in computer entertainment technology*, 2004, pp. 117-126.

Davis, J. W. and Vaks, S., "A perceptual user interface for recognizing head gesture acknowledgements," *Proceedings of the 2001 workshop on Perceptive user interfaces*, 2001, pp. 1-7.

Diaz, M., Alencastre-Miranda, M., Muñoz-Gómez, L., and Rudomin, I., "Multi-User Networked Interactive Augmented Reality Card Game," *International Conference on Cyberworlds*, 2006, pp. 177-182.

Dourish, P., "The Appropriation of Interactive Technologies: Some Lessons from Placeless Documents," *Computer Supported Cooperative Work*, 2003, vol. 12, no. 4, pp. 465-490.

Dourish, P. and Bellotti, V., "Awareness and coordination in shared workspaces," *Proceedings of the 1992 ACM conference on Computer-supported cooperative work*, 1992, pp. 107-114.

Feiner, S., "Augmented Reality: A New Way of Seeing," *Scientific American*, 2002, vol. 286, no. 4, pp. 34-41.

Feiner, S., MacIntyre, B., Haupt, M., and Solomon, E., "Windows on the world: 2D windows for 3D augmented reality," *ACM Symposium on User Interface Software and Technology*, 1993, pp. 145-155.

Feiner, S. K. and Seligmann, D. D., "Cutaways and ghosting: satisfying visibility constraints in dynamic 3D illustrations," *The Visual Computer,* 1992, vol. 8, no. 5-6, pp. 292-302.

Fitzpatrick, R. C., Butler, J. E., and Day, B. L., "Resolving Head Rotation for Human Bipedalism," *Current Biology,* 2006, vol. 16, no. 15, pp. 1509-1514.

Forsberg, A., Herndon, K., and Zeleznik, R., "Aperture based selection for immersive virtual environments," *Proceedings of the 9th annual ACM symposium on User interface software and technology,* 1996, pp. 95-96.

Fox, D., Burgard, W., Thrun, S., and Cremers, A. B., "A Hybrid Collision Avoidance Method For Mobile Robots," *IEEE International Conference on Robotics & Automation,* 1998, pp. 1238-1243.

Freeman, D., Benko, H., Morris, M. R., and Wigdor, D., "ShadowGuides: visualizations for in-situ learning of multi-touch and whole-hand gestures," *Proceedings of the ACM International Conference on Interactive Tabletops and Surfaces,* 2009, pp. 165-172.

Frees, S., Kessler, G. D., and Kay, E., "PRISM interaction for enhancing control in immersive virtual environments," *ACM Transaction on Computer-Human Interaction,* 2007, vol. 14, no. 1, p. 2.

Fuchs, H., Livingston, M. A., Raskar, R., Colucci, D. n., Keller, K., State, A., Crawford, J. R., Rademacher, P., Drake, S. H., and Meyer, A. A., "Augmented Reality Visualization for Laparoscopic Surgery," *Proceedings of the First International Conference on Medical Image Computing and Computer-Assisted Intervention,* 1998, pp. 934-943.

Fuchs, L., Pankoke-Babatz, U., and Prinz, W., "Supporting cooperative awareness with local event mechanisms: the groupdesk system," *Proceedings of the 4th conference on European Conference on Computer-Supported Cooperative Work,* 1995, pp. 247-262.

Fuhrmann, A., Splechtna, R., and Prikryl, J., "Comprehensive calibration and registration procedures for augmented reality," *Eurographics Workshop on Virtual Environments,* 2001, pp. 219-228.

Furmanski, C., Azuma, R., and Daily, M., "Augmented-Reality Visualizations Guided by Cognition: Perceptual Heuristics for Combining Visible and Obscured Information," *Proceedings of the 1st International Symposium on Mixed and Augmented Reality,* 2002, p. 215.

Gauglitz, S., Nuernberger, B., Turk, M., and Höllerer, T., "World-stabilized annotations and virtual scene navigation for remote collaboration," *Proceedings of the 27th annual ACM symposium on User interface software and technology,* 2014, pp. 449-459.

Goto, M., Uematsu, Y., Saito, H., Senda, S., and Iketani, A., "Task support system by displaying instructional video onto AR workspace," *Proceedings of the 9th IEEE International Symposium on Mixed and Augmented Reality*, 2010, pp. 83-90.

Grossman, T. and Balakrishnan, R., "The bubble cursor: enhancing target acquisition by dynamic resizing of the cursor's activation area," *Proceedings of the SIGCHI conference on Human factors in computing systems*, 2005, pp. 281-290.

Grossman, T. and Balakrishnan, R., "The design and evaluation of selection techniques for 3D volumetric displays," *Proceedings of the 19th annual ACM symposium on User interface software and technology*, 2006, pp. 3-12.

Grossman, T., Wigdor, D., and Balakrishnan, R., "Multi-finger gestural interaction with 3d volumetric displays," *Proceedings of the 17th annual ACM symposium on User interface software and technology*, 2004, pp. 61-70.

Grudin, J., "Computer-Supported Cooperative Work: History and Focus," *Computer*, 1994, vol. 27, no. 5, pp. 19-26.

Gutwin, C., Greenberg, S., and Roseman, M., "Workspace Awareness in Real-Time Distributed Groupware: Framework, Widgets, and Evaluation," *Proceedings of HCI on People and Computers XI*, 1996, pp. 281-298.

Hakkarainen, M. and Woodward, C., "SymBall: camera driven table tennis for mobile phones," *ACM SIGCHI International conference on Advances in computer entertainment technology*, 2005, pp. 391-392.

Hart, S. G., "NASA-Task Load Index (NASA-TLX); 20 Years Later," *Proceedings of the Human Factors and Ergonomics Society 50th Annual Meeting*, 2006, pp. 904-908.

Heiser, J., Tversky, B., and Silverman, M., "Sketches for and from collaboration," *Visual and spatial reasoning in design III*, 2004, pp. 69-78.

Henderson, S. J. and Feiner, S. K., "Augmented reality in the psychomotor phase of a procedural task," *Proceedings of the 10th IEEE International Symposium on Mixed and Augmented Reality*, 2011, pp. 191-200.

Henrysson, A., Billinghurst, M., and Ollila, M., "Face to Face Collaborative AR on Mobile Phones," *Proceedings of the 4th IEEE International Symposium on Mixed and Augmented Reality*, 2005, pp. 80-89.

Henrysson, A., Billinghurst, M., and Ollila, M., "AR tennis," *ACM SIGGRAPH Emerging technologies*, 2006, p. 1.

Hiltz, S. R. and Turoff, M., *The network nation: human communication via computer*: MIT Press, 1993.

Hinckley, K., Pausch, R., Goble, J. C., and Kassell, N. F., "A survey of design issues in spatial input," *Proceedings of the 7th annual ACM symposium on User interface software and technology*, 1994, pp. 213-222.

Hindmarsh, J., Fraser, M., Heath, C., Benford, S., and Greenhalgh, C., "Object-focused interaction in collaborative virtual environments," *ACM Transactions on Computer-Human Interaction*, 2000, vol. 7, no. 4, pp. 477-509.

Hirakawa, M. and Koike, S., "A Collaborative Augmented Reality System Using Transparent Display," *Proceedings of the IEEE Sixth International Symposium on Multimedia Software Engineering*, 2004, pp. 410-416.

Hoang, T. N. and Thomas, B. H., "Augmented Viewport: An action at a distance technique for outdoor AR using distant and zoom lens cameras," *International Symposium on Wearable Computers*, 2010, pp. 1-4.

Hua, H., Brown, L. D., and Gao, C., "System and interface framework for SCAPE as a collaborative infrastructure," *Presence: Teleoperators and Virtual Environments*, 2004, vol. 13, no. 2, pp. 234-250.

Intel, "OpenCV: Open Source Computer Vision Library."

Interrante, V., Ries, B., and Anderson, L., "Seven League Boots: A New Metaphor for Augmented Locomotion through Moderately Large Scale Immersive Virtual Environments," *IEEE Symposium on 3D User Interfaces*, 2007, pp. 167-170.

Ishii, H., Ben-Joseph, E., Underkoffler, J., Yeung, L., Chak, D., Kanji, Z., and Piper, B., "Augmented Urban Planning Workbench: Overlaying Drawings, Physical Models and Digital Simulation," *Proceedings of the 1st International Symposium on Mixed and Augmented Reality*, 2002, p. 203.

Ishii, H., Kobayashi, M., and Arita, K., "Iterative design of seamless collaboration media," *Communications of the ACM*, 1994, vol. 37, no. 8, pp. 83-97.

Ismail, A. W. and Sunar, M. S., "Collaborative Augmented Reality: Multi-user Interaction in Urban Simulation," *Proceedings of the 1st International Visual Informatics Conference on Visual Informatics: Bridging Research and Practice*, 2009, pp. 382-391.

Iyoda, T., Abe, T., Tokai, K., Sakamoto, S., Shingu, J., Onuki, H., Shi, M., and Uchihashi, S., "LightCollabo: distant collaboration support system for manufacturers," *Proceedings of the 14th international conference on Advances in multimedia modeling*, 2008, pp. 369-379.

Izadi, S., Brignull, H., Rodden, T., Rogers, Y., and Underwood, M., "Dynamo: a public interactive surface supporting the cooperative sharing and exchange of media,"

Proceedings of the 16th annual ACM symposium on User interface software and technology, 2003, pp. 159-168.

Johansen, R., "Teams for tomorrow [groupware]," *Proceedings of the 24th Hawaii International Conference on System Sciences,* 1991, vol. 3, pp. 521-534.

Kalkofen, D., Sandor, C., White, S., and Schmalstieg, D., "Visualization Techniques for Augmented Reality," in *Handbook of Augmented Reality*, 1 ed: Springer New York, 2011.

Kato, H. and Billinghurst, M., "Marker tracking and hmd calibration for a video-based augmented reality conferencing system," *Proceedings of the 2nd IEEE and ACM International Workshop on Augmented Reality*, 1999, pp. 85-94.

Kim, S., Lee, G., Sakata, N., and Billinghurst, M., "Improving Co-Presence with Augmented Visual Communication Cues for Sharing Experience through Video Conference," *Proceedings of the 13th IEEE International Symposium on Mixed and Augmented Reality*, 2014, pp. 83-92.

Kirk, D. S. and Fraser, D. S., "The effects of remote gesturing on distance instruction," *Proceedings of the 2005 conference on Computer support for collaborative learning: learning 2005: the next 10 years!*, 2005, pp. 301-310.

Kiyokawa, K., Billinghurst, M., Hayes, S. E., Gupta, A., Sannohe, Y., and Kato, H., "Communication Behaviors of Co-Located Users in Collaborative AR Interfaces," *1st International Symposium on Mixed and Augmented Reality*, 2002, p. 139.

Kiyokawa, K., Iwasa, H., Takemura, H., and Yokoya, N., "Collaborative Immersive Workspace Through a Shared Augmented Environment " *International Symposium on Intelligent Systems and Advanced Manufacturing*, 1998, pp. 2-13.

Kiyokawa, K., Takemura, H., and Yokoya, N., "SeamlessDesign for 3D Object Creation," *IEEE MultiMedia,* 2000, vol. 7, no. 1, pp. 22-33.

Kopper, R., Bacim, F., and Bowman, D. A., "Rapid and accurate 3D selection by progressive refinement," *IEEE Symposium on 3D User Interface*, 2011, pp. 67-74.

Kopper, R., Bowman, D. A., Silva, M. G., and McMahan, R. P., "A human motor behavior model for distal pointing tasks," *International Journal of Human-Computer Studies,* 2010, vol. 68, no. 10, pp. 603-615.

Kurata, T., Sakata, N., Kourogi, M., Kuzuoka, H., and Billinghurst, M., "Remote Collaboration using a Shoulder-Worn Active Camera/Laser," *Proceedings of the Eighth International Symposium on Wearable Computers*, 2004, pp. 62-69.

Kuzuoka, H., "Spatial workspace collaboration: a SharedView video support system for remote collaboration capability," *Proceedings of the SIGCHI conference on Human factors in computing systems*, 1992, pp. 533-540.

Lanir, J., Stone, R., Cohen, B., and Gurevich, P., "Ownership and control of point of view in remote assistance," *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, 2013, pp. 2243-2252.

Lee, T. and Höllerer, T., "Handy AR: Markerless Inspection of Augmented Reality Objects Using Fingertip Tracking," *Proceedings of the 11th IEEE International Symposium on Wearable Computers*, 2007, pp. 1-8.

Leung, D. C. M., Au, P. S., King, I., and Yau, E. H. H., "Remote augmented reality for multiple players over network," *International conference on advances in computer entertainment technology*, 2007, pp. 220-223.

Li, W., Ritter, L., Agrawala, M., Curless, B., and Salesin, D., "Interactive cutaway illustrations of complex 3D models," *ACM Transaction on Graphics*, 2007, vol. 26, no. 3, p. 31.

Liang, J. and Green, M., "A highly interactive 3D modeling system," *Computer Graphics*, 1994, vol. 18, no. 4, pp. 499-506.

Mackenzie, I. S. and Riddersma, S., "Effects of output display and control-display gain on human performance in interactive systems," *Behaviour & Information Technology*, 1994, pp. 328-337.

Mendez, E. and Schmalstieg, D., "Importance masks for revealing occluded objects in augmented reality," *Proceedings of the 16th ACM Symposium on Virtual Reality Software and Technology*, 2009, pp. 247-248.

Microsoft. (2015). *Kinect SDK*. http://www.microsoft.com/en-us/kinectforwindows/

Mine, M. R., "Virtual Environment Interaction Techniques," UNC Chapel Hill Computer Science Technical Report, 1995.

Morris, M. R., Ryall, K., Shen, C., Forlines, C., and Vernier, F., "Beyond "social protocols": multi-user coordination policies for co-located groupware," *Proceedings of the 2004 ACM conference on Computer supported cooperative work*, 2004, pp. 262-265.

Mulloni, A., Wagner, D., and Schmalstieg, D., "Mobility and social interaction as core gameplay elements in multi-player augmented reality," *Proceedings of the 3rd international conference on Digital Interactive Media in Entertainment and Arts*, 2008, pp. 472-478.

NaturalPoint. (2015). *OptiTrack*. http://www.naturalpoint.com/optitrack/

Newcombe, R. A., Izadi, S., Hilliges, O., Molyneaux, D., Kim, D., Davison, A. J., Kohli, P., Shotton, J., Hodges, S., and Fitzgibbon, A., "KinectFusion: Real-Time Dense Surface Mapping and Tracking," *Proceedings of the 10th IEEE International Symposium on Mixed and Augmented Reality*, 2011, pp. 127 - 136.

Nicolau, S. A., Goffin, L., and Soler, L., "A low cost and accurate guidance system for laparoscopic surgery: validation on an abdominal phantom," *Proceedings of the ACM symposium on Virtual reality software and technology*, 2005, pp. 124-133.

Nilsen, T. and Looser, J., "Tankwar - Tabletop war gaming in augmented reality," *Proceedings of the 2nd International Workshop on Pervasive Gaming Applications*, 2005.

Nintendo. (2015). *Wii*. http://wii.nintendo.com/

Oda, O., Elvezio, C., Sükan, M., Feiner, S., and Tversky, B., "Virtual Replicas for Remote Assistance in Virtual and Augmented Reality," *User Interface and Software Technology*, 2015, pp. 405-415.

Oda, O. and Feiner, S., "Interference avoidance in multi-user hand-held augmented reality," *Proceedings of the 8th IEEE International Symposium on Mixed and Augmented Reality*, 2009, pp. 13-22.

Oda, O. and Feiner, S., "3D Referencing Techniques for Physical Objects in Shared Augmented Reality," *Proceedings of the 11th IEEE International Symposium on Mixed and Augmented Reality*, 2012, pp. 207-215.

Oda, O. and Feiner, S. (2012). *Goblin XNA*. http://goblinxna.codeplex.com

Oda, O., Lister, L. J., White, S., and Feiner, S., "Developing an augmented reality racing game," *INTETAIN*, 2008, pp. 1-8.

Oda, O., Sükan, M., Feiner, S., and Tversky, B., "Poster: 3D Referencing for Remote Task Assistance in Augmented Reality," *IEEE 3D User Interface*, 2013, pp. 179-180.

Ohshima, T., Sato, K., Yamamoto, H., and Tamura, H., "RV-Border Guards: A Multi-player Mixed Reality Entertainment," *Transactions of the Virtual Reality Society of Japan*, 1999, pp. 699-705.

Ohshima, T., Satoh, K., Yamamoto, H., and Tamura, H., "AR2 Hockey," *ACM SIGGRAPH 98 Conference abstracts and applications*, 1998, p. 110.

Olson, M. H. and Lucas, H. C., "The impact of office automation on the organization: some implications for research and practice," *Communications of the ACM*, 1982, vol. 25, no. 11, pp. 838-847.

Olwal, A., Benko, H., and Feiner, S., "SenseShapes: Using Statistical Geometry for Object Selection in a Multimodal Augmented Reality System," *Proceedings of the 2nd IEEE/ACM International Symposium on Mixed and Augmented Reality*, 2003, pp. 300-301.

Olwal, A. and Feiner, S., "The Flexible Pointer: An Interaction Technique for Augmented and Virtual Reality," *Conference Supplement of User Interface and Software Technology*, 2003, pp. 81-82.

Orozco, C., Esteban, P., and Trefftz, H., "Collaborative and distributed augmented reality in teaching multi-variate calculus," *Proceedings of the 5th IASTED international conference on Web-based education*, 2006, pp. 141-145.

Paelke, V., Reimann, C., and Stichling, D., "Foot-based mobile interaction with games," *ACM SIGCHI International conference on Advances in computer entertainment technology*, 2004, pp. 321-324.

Pemberton, L. and Winter, M., "Collaborative augmented reality in schools," *Proceedings of the 9th international conference on Computer supported collaborative learning - Volume 2*, 2009, pp. 109-111.

Pierce, J. S., Forsberg, A. S., Conway, M. J., Hong, S., Zeleznik, R. C., and Mine, M. R., "Image plane interaction techniques in 3D immersive environments," *Proceedings of the 1997 symposium on Interactive 3D graphics*, 1997, pp. 39-43.

Pierce, J. S., Stearns, B. C., and Pausch, R., "Voodoo dolls: seamless interaction at multiple scales in virtual environments," *Proceedings of the 1999 symposium on Interactive 3D graphics*, 1999, pp. 141-145.

Piumsomboon, T., Altimira, D., Kim, H., Clark, A., Lee, G., and Billinghurst, M., "Grasp-Shell vs gesture-speech: A comparison of direct and indirect natural interaction techniques in augmented reality," *Proceedings of the 13th IEEE International Symposium on Mixed and Augmented Reality*, 2014, pp. 73-82.

Poupyrev, I., Billinghurst, M., Weghorst, S., and Ichikawa, T., "The go-go interaction technique: non-linear mapping for direct manipulation in VR," *Proceedings of the 9th annual ACM symposium on User interface software and technology*, 1996, pp. 79-80.

Poupyrev, I., Weghorst, S., Billinghurst, M., and Ichikawa, T., "Egocentric Object Manipulation in Virtual Environments: Empirical Evaluation of Interaction Techniques," *Computer Graphics Forum*, 1998, vol. 17, no. 3, pp. 41-52.

R Core Team. (2015). *R: A Language and Environment for Statistical Computing*. http://www.R-project.org/

Razzaque, S., Kohn, Z., and Whitton, M. C., "Redirected Walking," *Eurographics*, 2001, pp. 289-294.

Regenbrecht, H. T., Wagnerb, M. T., and Baratoffa, G., "MagicMeeting- a Collaborative Tangible Augmented Reality System," *Virtual Reality - Systems, Development and Applications*, 2002, vol. 6, pp. 151-166.

Reitmayr, G. and Schmalstieg, D., "Mobile Collaborative Augmented Reality," *Proceedings of the IEEE and ACM International Symposium on Augmented Reality*, 2001, p. 114.

Rekimoto, J. and Saitoh, M., "Augmented Surfaces: A Spatially Continuous Workspace for Hybrid Computing Environments," *ACM SIGCHI*, 1999, pp. 378-385.

Rizzolatti, G. and Craighero, L., "The Mirror-Neuron System.," *Annu Rev Neurosci*, 2004, vol. 27, pp. 169-192.

Sakata, N., Kurata, T., and Kuzuoka, H., "Visual assist with a laser pointer and wearable display for remote collaboration," *CollabTech*, 2006, pp. 66-71.

Salzmann, H., Moehring, M., and Froehlich, B., "Virtual vs. Real-World Pointing in Two-User Scenarios," *Virtual Reality*, 2009, pp. 127-130.

Schmidt, K. and Bannon, L. J., "Taking CSCW Seriously: Supporting Articulation Work," *Computer Supported Cooperative Work*, 1992, vol. 1, no. 1-2, pp. 7-40.

Sodhi, R., Benko, H., and Wilson, A., "LightGuide: projected visualizations for hand movement guidance," *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, 2012, pp. 179-188.

Sodhi, R. S., Jones, B. R., Forsyth, D., Bailey, B. P., and Maciocci, G., "BeThere: 3D Mobile Collaboration with Spatial Input," *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, 2013, pp. 179-188.

Stafford, A. and Piekarski, W., "User evaluation of god-like interaction techniques," *Proceedings of the ninth conference on Australasian user interface*, 2008, pp. 19-27.

Steinicke, F., Bruder, G., Jerald, J., Frenz, H., and Lappe, M., "Analyses of human sensitivity to redirected walking," *ACM symposium on Virtual reality software and technology*, 2008, pp. 149-156.

StereoOptical. (2015). *Stereo Fly Test*. http://www.stereooptical.com/products/stereotests

Stevenson, D., Li, J., Smith, J., and Hutchins, M., "A collaborative guidance case study," *Proceedings of the ninth conference on Australasian user interface - Volume 76*, 2008, pp. 33-42.

Stoakley, R., Conway, M. J., and Pausch, R., "Virtual reality on a WIM: interactive worlds in miniature," *Proceedings of the SIGCHI conference on Human factors in computing systems*, 1995, pp. 265-272.

Strauss, P. S. and Carey, R., "An object-oriented 3D graphics toolkit," *SIGGRAPH Computer Graphics*, 1992, vol. 26, no. 2, pp. 341-349.

Szalavári, Z., Eckstein, E., and Gervautz, M., "Collaborative gaming in augmented reality," *Proceedings of the ACM symposium on Virtual reality software and technology*, 1998, pp. 195-204.

Szalavári, Z., Schmalstieg, D., Fuhrmann, A., and Gervautz, M., ""Studierstube" An Environment for Collaboration in Augmented Reality," *Collaborative Virtual Environment*, 1996.

Tait, M. and Billinghurst, M., "[Poster] View Independence in Remote Collaboration Using AR," *Proceedings of the 13th IEEE International Symposium on Mixed and Augmented Reality*, 2014, pp. 309-310.

Talmy, L., *Toward a Cognitive Semantics* vol. 1-2: MIT press, 2003.

Tang, R., Yang, X.-D., Bateman, S., Jorge, J., and Tang, A., "Physio@Home: Exploring Visual Guidance and Feedback Techniques for Physiotherapy Exercises," *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems*, 2015, pp. 4123-4132.

Tatzgern, M., Grasset, R., Kalkofen, D., and Schmalstieg, D., "Transitional Augmented Reality Navigation for Live Captured Scenes," *IEEE Virtual Reality*, 2014, pp. 21-26.

Tecchia, F., Alem, L., and Huang, W., "3D helping hands: a gesture based MR system for remote collaboration," *Proceedings of the 11th ACM SIGGRAPH International Conference on Virtual-Reality Continuum and its Applications in Industry*, 2012, pp. 323-328.

Tukey, J. W., *Exploratory Data Analysis*: Addison-Wesley, 1977.

Tversky, B., Morrison, J. B., and Betrancourt, M., "Animation: can it facilitate?," *International Journal of Human-Computer Studies*, 2002, vol. 57, no. 4, pp. 247-262.

Twister. (2009). http://www.hasbro.com/twister/default.cfm?page=Entertainment/History&src=endeca

Vanacken, L., Grossman, T., and Coninx, K., "Exploring the Effects of Environment Density and Target Visibility on Object Selection in 3D Virtual Environments," *IEEE Symposium on 3D User Interfaces*, 2007, pp. 117-124.

Vandenberg, S. G. and Kuse, A. R., "Mental Rotations, a Group Test of Three-dimensional Spatial Visualization.," *Perceptual and Motor Skills,* 1978, vol. 47, pp. 599-604.

Vogel, D. and Baudisch, P., "Shift: a technique for operating pen-based interfaces using touch," *SIGCHI conference on Human factors in computing systems*, 2007, pp. 657-666.

Wagner, D., Pintaric, T., Ledermann, F., and Schmalstieg, D., "Towards Massively Multi-User Augmented Reality on Handheld Devices," *Proceedings of the 3rd International Conference on Pervasive Computing*, 2005, pp. 208-219.

Wang, L., Zhao, Y., Mueller, K., and Kaufman, A., "The magic volume lens: an interactive focus+context technique for volume rendering," *Visualization*, 2005, pp. 367 - 374.

Wang, R., Paris, S., and Popović, J., "6D hands: markerless hand-tracking for computer aided design," *Proceedings of the 24th annual ACM symposium on User interface software and technology*, 2011, pp. 549-558.

Wang, X., Chen, R., Gong, Y., and Hsieh, Y.-T., "Experimental Study on Augmented Reality Potentials in Urban Design," *Proceedings of the 11th International Conference Information Visualization*, 2007, pp. 567-572.

Wexler, M., Kosslyn, S. M., and Berthoz, A., "Motor Processes in Mental Rotation," *Cognition,* 1998, vol. 1, pp. 77-94.

White, S., Lister, L., and Feiner, S., "Visual Hints for Tangible Gestures in Augmented Reality," *Proceedings of the 6th IEEE and ACM International Symposium on Mixed and Augmented Reality*, 2007, pp. 1-4.

Wiedenmaier, S., Oehme, O., Schmidt, L., and Luczak, H., "Augmented Reality (AR) for Assembly Processes Design and Experimental Evaluation," *International Journal of Human-Computer Interaction*, 2003, vol. 16, no. 3, pp. 497-514.

Wilson, A. D. and Benko, H., "Combining multiple depth cameras and projectors for interactions on, above and between surfaces," *Proceedings of the 23nd annual ACM symposium on User interface software and technology*, 2010, pp. 273-282.

Wither, J. and Höllerer, T., "Pictorial depth cues for outdoor augmented reality," *International Symposium on Wearable Computers*, 2005, pp. 92-99.

Wloka, M. M. and Anderson, B. G., "Resolving occlusion in augmented reality," *Proceedings of the 1995 symposium on Interactive 3D graphics*, 1995, pp. 5-12.

Wyss, H. P., Blach, R., and Bues, M., "iSith - Intersection-based Spatial Interaction for Two Hands," *Proceedings of the 3D User Interfaces*, 2006, pp. 59-61.

Yang, U. and Kim, G. J., “Implementation and evaluation of "just follow me": an immersive, VR-based, motion-training system,” *Presence: Teleoperators and Virtual Environments,* 2002, vol. 11, no. 3, pp. 304-323.

Zhai, S., Buxton, W., and Milgram, P., “"Silk Cursor": Investigating transparency for 3D target acquisition,” *ACM Computer-Human Interaction,* 1994, pp. 459-464.

Appendix A. Goblin XNA Framework

In this appendix, we present a software framework we have developed for streamlining development of AR applications. Development of a multi-user AR application can be quite challenging and time consuming because of the need for live video capture, tracking and registration, support for interaction devices, scene management, rendering of synthesized view, and synchronization among users on shared virtual contents. To rapidly prototype such applications, including ones for exploring the interaction techniques described in Chapters 3–5, we have developed an open-source framework, *Goblin XNA* [Oda and Feiner, 2012]. This framework is a successor of the Goblin framework developed by Marc Eaddy [Eaddy and Feiner, 2005]. Although Goblin XNA inherited only some of the AR related requirements specified in the original framework, it was designed and implemented from scratch with an emphasis on extensibility and flexibility.

The latest implementation of Goblin XNA was based on Microsoft XNA Game Studio 4.0, and written in nearly 40,000 lines of C# source code. Our framework supports 6DOF position and orientation tracking using several commercial tracking technologies, including optical marker-based tracking through ALVAR [VTT, 2015], in addition to providing a 3D scene graph, rigid body physics simulation (using Newton Game Dynamics [Newton Game Dynamics, 2015], Havok [Havok, 2015], and Matali [Komires, 2015]), networking (using Lidgren [Lidgren, 2015]), shaders, particle systems, and 2D user interface primitives. This framework has been tested and actively used to develop research prototypes by our lab members and for class assignments by students in the 3D User Interface and Augmented Reality course taught by Professor Steven Feiner from 2008 to 2013. It has also been released to the public since 2009 at

http://goblinxna.codeplex.com. The distribution includes not only the source code of the framework, but also tutorials, an installation guide, a user manual, and useful toolkits. The framework has been downloaded nearly 10,000 times world-wide since its initial release in 2009, and was continuously updated through 2012 with bug fixes and new features and supported through a discussion board. It has also been used in a variety of projects at other institutions (e.g., http://cs.utsa.edu/~jpq/Site/teaching/uiu-f11/team_ar_assignment.htm, https://prezi.com/i2cyneybxvfr/development-of-a-position-control-for-the-kuka/, http://cocuserguide.blogspot.com/search/label/clash%20of%20clans%20goblin%20xna).

In our discussion of Goblin XNA, we first describe its architecture and the high-level overview of its core components in Section A.1. Next, we describe several toolkits we have implemented outside of the framework to streamline the development of an AR application in Section A.2. We then explain the multiple iterations of the framework in Section A.3, and finally, present notable projects that were implemented in our lab using the framework throughout the span of my Ph.D career in Section A.4.

A.1. Software Architecture

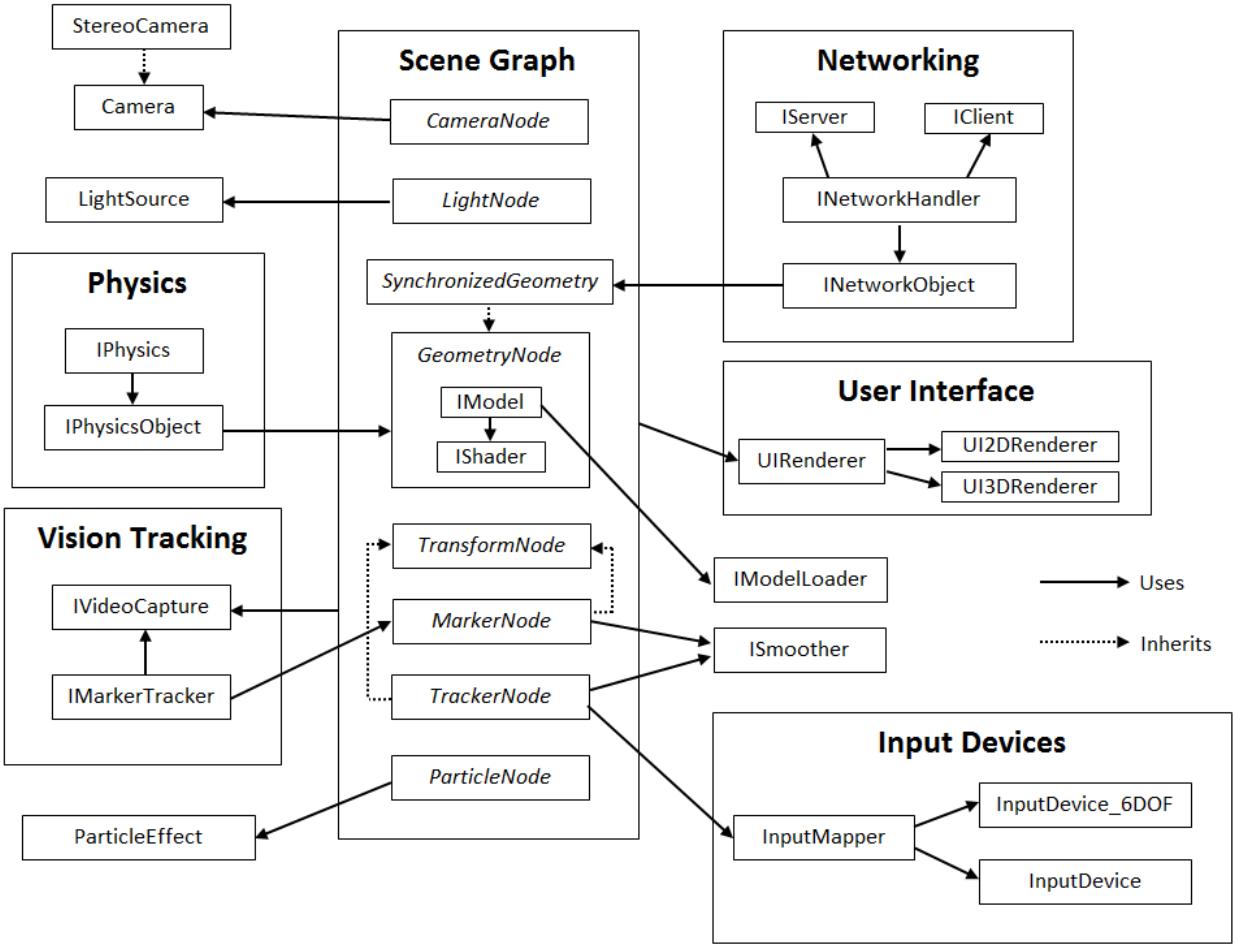

Figure A.1. An abstract view of the Goblin XNA software architecture.

Goblin XNA consists of several core components including scene graph, 6DOF and 3DOF tracking and input device abstraction, video acquisition and vision-based tracking, physics engine, networking, and user interface. As depicted in Figure A.1, the principal software component of Goblin XNA architecture is the scene graph, and all other core components are interfaced by the scene graph. We have designed many of our software components to be easily replaceable with another implementation so that the programmer can, for example, incorporate a physics engine of her choice by implementing the *IPhysics* interface to simulate the 3D objects in the scene

graph, if she is not satisfied with the physics engine we provide. We will briefly describe the high-level overview of each core component of the framework in the following sections.

A.1.1 Scene Graph

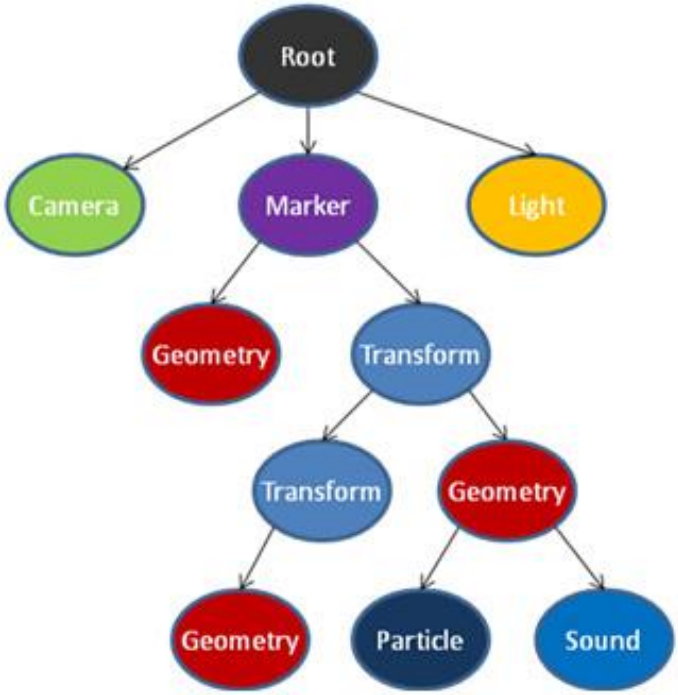

Figure A.2. An illustration of a typical Goblin XNA scene graph hierarchy.

A scene graph is a tree-like data structure that arranges the logical and spatial representation of a graphical scene. The design of our scene graph is similar to that of OpenSG [Reiners et al., 2015] in terms of the node relationship in the tree hierarchy. Our scene graph currently consists of ten node types: Geometry, Transform, Light, Camera, Particle, Marker, Sound, Switch, LOD (Level of Detail), and Tracker. An example scene graph hierarchy is illustrated in Figure A.2. The scene graph is rendered using preorder tree traversal. Detailed descriptions of all node types are provided in the following section.

A.1.1.1 Geometry Node

A Geometry node contains a geometric model (*IModel*) to be rendered in the scene using a specific *IShader* implementation. For example, a Geometry node might represent a 3D airplane model in a flight simulator game. A Geometry node can have only one 3D model associated with it. In case of rendering multiple 3D models, multiple Geometry nodes are required. A geometric model can be created by either loading data from an existing model file through an *IModelLoader* implementation or using a list of custom vertices and indices. Goblin XNA provides several simple shapes, such as Cube/Box, Sphere, Cylinder/Cone, ChamferCylinder, Capsule, Torus, and Disk, similar to those in the OpenGL GLUT library [SGI, 2015], as well as Billboard and Text3D. In addition to the geometric model itself, a Geometry node also contains other properties associated with the model. Some of the important properties include:

- Material properties, such as color (diffuse, ambient, emissive, and specular color), shininess, and texture, which are used for rendering.
- Physical properties, such as shape, mass, and moment of inertia defined in *IPhysicsObject* interface, which are used for physical simulation.
- Occlusion property, which determines whether the geometry will be used as an occluder that occludes the virtual scene, typically used in AR applications. When the geometry is set to be an occluder, the virtual geometry itself will not be visible, but it will participate in visible-surface determination, blocking the view of any virtual objects that are behind it relative to the camera. Occluder geometry is typically transparent because it corresponds to real objects (visible in the view of the real world in an AR application) with which the virtual object interacts.

A SynchronizedGeometry node, which extends the Geometry node, contains networking properties defined through an *INetworkObject* interface. With a distinction between the regular Geometry node and SynchronizedGeometry node, it makes it possible for the programmer to determine which Geometry nodes should be synchronized among multiple networked machines and which should be local.

A.1.1.2 Transform Node

A Transform node modifies the transformation (e.g., translation, rotation, and scale) of any object beneath it in the hierarchy that contains spatial information. For example, if a Camera node is added to a Transform node, and the transformation of the Transform node is modified, then the spatial information of the Camera node will also change.

We provide two ways to set the transformation of a Transform node. One way is to set each transformation component (pre-translation, scale, rotation, and post-translation) separately, causing the composite transformation matrix to be computed automatically from these values. No matter the order in which the components of the transformation are set, they will be applied such that any child of the Transform node will first be translated by the pre-translation component, then scaled by the scale component, next rotated by the rotation component, and finally translated by the post-translation component. An alternative is to directly set the transformation matrix. Note that the last approach used determines the transformation. For example, if the pre-translation, scale, rotation, and post-translation are set separately, and then later on a new transformation is assigned directly, then the node's transformation will be the newly assigned one. To switch back to using a transformation determined by the pre-translation, scale, rotation, and post-translation, a value needs to be assigned to one of these components, causing the node's trans-

formation to once again be composed from the pre-translation, scale, rotation, and post-translation.

A.1.1.3 Light Node

A Light node contains “light sources” that illuminate the 3D models. Light source properties differ based on the type of light, except for the diffuse and specular colors, which apply to all types. There are three types of simplified lights that have typically been used in “fixed pipeline” real-time computer graphics: directional, point, and spot lights.

A directional light has a direction, but does not have a position. The position of the light is assumed to be infinitely distant, so that no matter where the 3D model is placed in the scene, it will be illuminated from a constant direction. For example, the sun, as seen from the earth on a bright day, is often conventionally modeled as a directional light source, since it is so far away.

A point light has a position from which light radiates spherically. The intensity of a point light source attenuates with increased distance from the position, based on attenuation coefficients. For example, a small bare light bulb is often conveniently modeled as a point light.

A spot light is a light that has a position and direction, and a cone-shaped frustum. Only the 3D models that fall within this cone shaped frustum are illuminated by a spot light. As its name implies, a spot light can be used to model a simplified theatrical light.

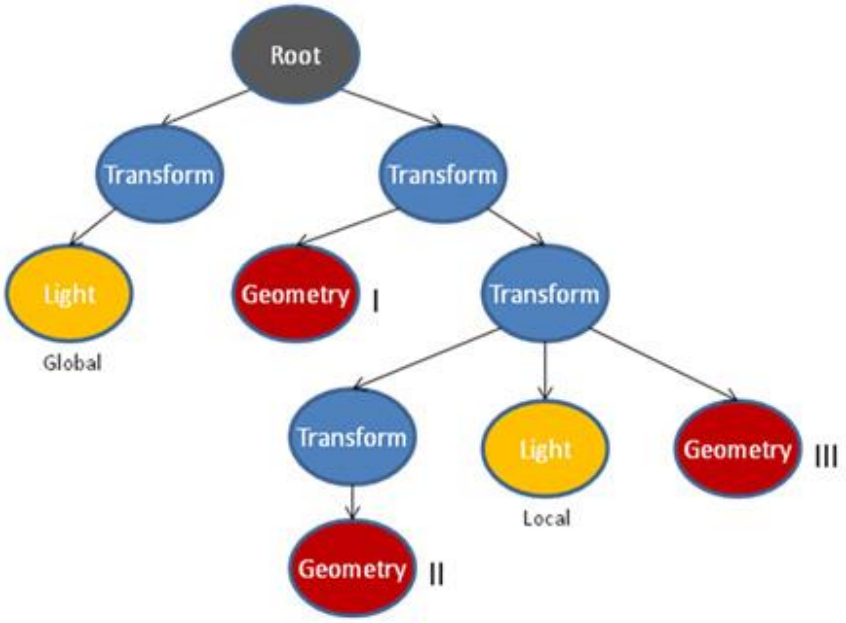

Figure A.3. An illustration of the difference between a global and local light.

A Light node can contain one light source and an ambient light color, and the entire node can either be global or local. A global Light node illuminates the entire scene in the scene graph. In contrast, a local light node illuminates only a part of the scene: the Light node's sibling nodes and all their descendant nodes. For example, in Figure A.3, the global Light node (marked "Global") illuminates all Geometry nodes in the scene (Geometry nodes I, II, and III), while the local Light node (marked "Local") illuminates only Geometry nodes II, and III (i.e., the siblings of the local Light node and their descendants). Furthermore, note that if the global Light node in Figure A.3 were a local Light node, then it would illuminate none of the Geometry nodes in the scene graph, because this Light node has no siblings.

A.1.1.4 Camera Node

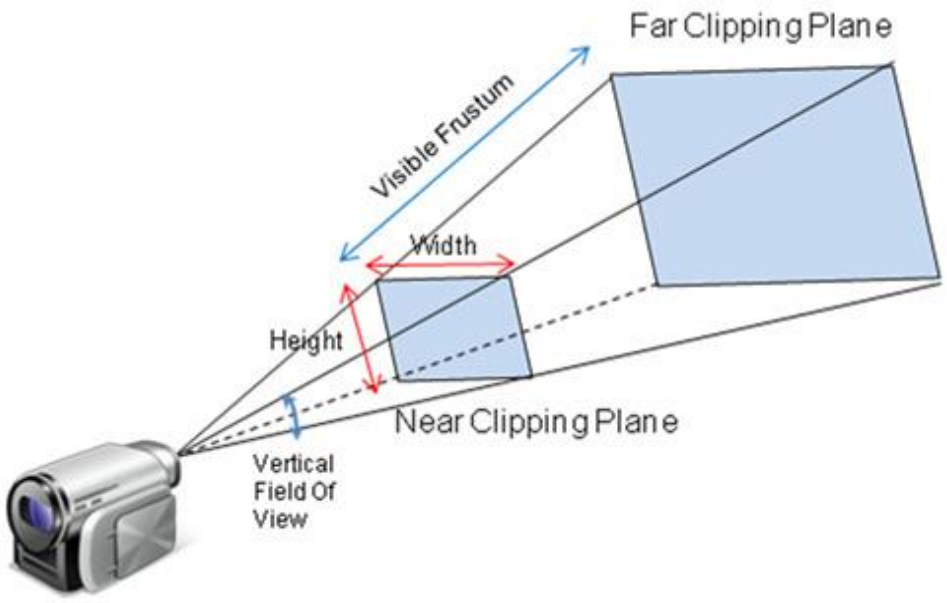

Figure A.4. An illustration of a visible frustum defined by a vertical field of view, aspect ratio (ratio of frustum width to height), near clipping plane, and far clipping plane.

A Camera node defines the position and visible frustum of the viewer (i.e., the *view volume* containing what you see on your display). The visible frustum of the viewer is defined by the vertical field of view, aspect ratio (ratio of frustum width to height), near clipping plane, and far clipping plane, as shown in Figure A.4. A view frustum that is a regular pyramid can be created by assigning values to these parameters, causing the view and projection matrices to be computed automatically. Alternatively (e.g., if a view frustum that is not a regular pyramid is desired), the view and projection matrices can be directly assigned. The initial view direction is toward the –z direction with an up vector of +*y*. The Camera node rotation property modifies this view direction by applying the given rotation to the initial view direction.

A.1.1.5 Particle Node

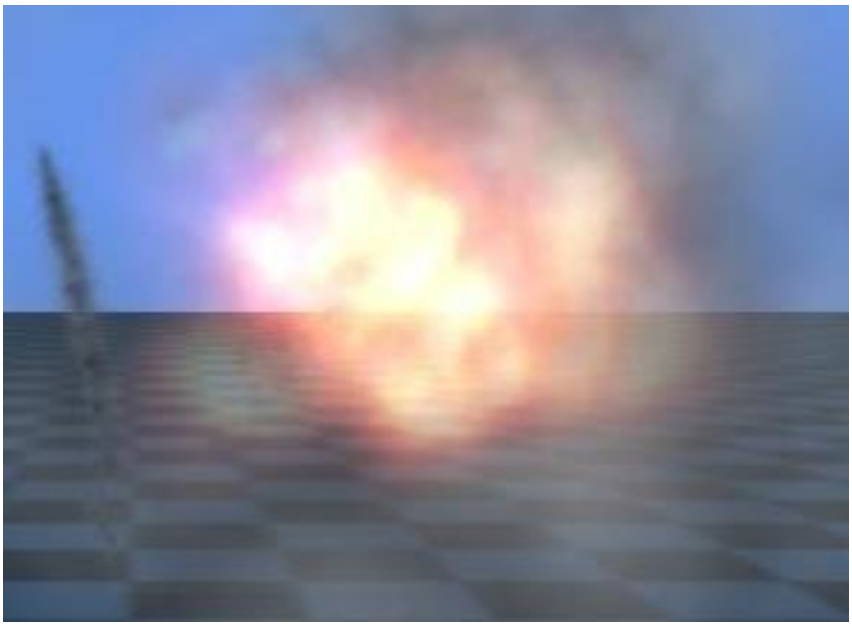

Figure A.5. An example of an explosion particle effect (courtesy of the XNA App Hub).

A Particle node contains one or more particle effects, such as fire, smoke, explosions (Figure A.5), and splashing water. A particle effect has properties such as texture, duration before a particle disappears, start size, end size, and horizontal and vertical velocities. Goblin XNA provides a small set of particle effects, including fire and smoke. The properties of an existing particle effect can be modified to create a customized particle effect.

A.1.1.6 Marker Node

A Marker node modifies the transformations of its descendant nodes, similar to a Transform node. However, the transformation is modified based on the 6DOF (six-degrees-of-freedom) pose matrix computed for an array of one or more *fiducial markers*. Fiducial markers are geometric patterns, typically printed out on paper or cardboard, that are viewed by a video camera. The video is processed interactively by computer vision algorithms that can find the image of each clearly visible marker and compute the position and orientation of that marker rela-

tive to the camera to determine its pose matrix. This node can be used to track a single marker or a specific marker array. A Marker node not only provides a computed pose matrix, but also supports smoothing out the sequence of pose matrices detected over time through an *ISmoother* implementation. We currently support double exponential smoothing [LaViola, 2003], but a custom filter can be implemented and used with a Marker node through our extensible interface class.

A.1.1.7 Sound Node

A Sound node contains information about a 3D sound source that has a position, velocity, and forward and up vector. This node can be attached to a Geometry node as a child to be associated with a specific 3D model so that a 3D sound can be produced at an appropriate location.

A.1.1.8 Switch Node

A Switch node is used to select a single one of its child nodes to traverse.

A.1.1.9 LOD (Level of Details) Node

An LOD (Level of Detail) node is used to select a single model to be rendered from a list of models, each of which is assumed to have a different level of detail. The LOD node is an extension of the Geometry node. This node can automatically compute the appropriate level of detail to use based on the distance between the model and the active camera.

A.1.1.10 Tracker Node

A Tracker node is quite similar to a Marker node. This node is associated with a specific 6DOF or 3DOF tracking device through the *InputMapper* class. The world transformation of this node is automatically updated based on the information passed from the tracking device, and any

nodes added below this node are affected as well. The world transformation of this node can be smoothed in a similar fashion to the Marker node described in Section A.1.1.6.

A.1.2 6DOF and 3DOF Tracking and Input Device Abstraction

Goblin XNA currently supports 3DOF (three–degrees-of-freedom) orientation tracking and 6DOF (six–degrees-of-freedom) position and orientation tracking using the 6DOF ALVAR [VTT, 2015] optical marker tracking systems, 3DOF and 6DOF InterSense hybrid trackers, and the 3DOF Vuzix iWear VR920 and Wrap Tracker 6TC (currently for 3DOF only) trackers. A vision-based tracking system (*IMarkerTracker*) is interfaced through the Marker node (Section A.1.1.6), and a physical tracking device such as InterSense trackers, GPS, and Vuzix orientation trackers (iWearTracker and Wrap Tracker 6TC) are interfaced through the *InputMapper* class, which provides hardware device abstractions for various input devices and allows simple access through a unique string identifier for each device. The other input devices such as mouse and keyboard are also supported through this input mapping class. All device classes are singleton classes to prevent multiple instantiations of an identical device.

A.1.2.1 6DOF Input Devices

A 6DOF input device provides position (*x*, *y*, *z*) and orientation (yaw, pitch, roll) information, and any classes that handles a 6DOF input device are implemented through the *InputDevice_6DOF* interface. The interface class provides a unified method to retrieve the position and orientation of a 6DOF input device.

A.1.2.1.1 InterSense Hybrid Trackers

We support two ways for obtaining the tracking data from an InterSense system. Any hybrid trackers associated with an InterSense system can be connected through a direct serial port

connection in the case only one machine accesses the tracking data or a network server connection in the case multiple machines access the tracking data. The driver software that comes with the InterSense system handles up to eight trackers, and any of the trackers can be accessed through the *InputMapper* class.

A.1.2.1.2 Vuzix iWear VR920 and Wrap Tracker 6TC Orientation Tracker

We provide an abstraction class on top of the Vuzix SDK so that the programmers can retrieve the tracker data just as any other 6DOF devices.

A.1.2.1.3 Simulated 6DOF Input Using the Mouse

We provide a simulated 6DOF input device based on the mouse dragging and wheeling for debugging or experimental purposes.

A.1.2.2 Non-6DOF Input Devices

Any input devices that do not support 6DOF input are in this category (e.g., mouse, keyboard, and gamepad). Unlike 6DOF input devices, these devices do not provide the same type of input, so there is no unified method like getting the position and orientation. However, all of the non-6DOF input device classes provide a unified method that can be used to programmatically trigger some of the callback functions specific to each input devices. For example, even if the actual keyboard key is not pressed, the programmer can use this method to trigger a key press event. These device classes implement *InputDevice* interface and are also managed through the *InputMapper* class, just like the 6DOF and 3DOF input device classes.

A.1.3 Video Acquisition and Vision-based Tracking

Two important functionalities that Goblin XNA addresses are combining rendered graphics with the real world image and 6DOF pose (position and orientation) tracking for registration. To support these features, we provide interface classes for capturing live videos and perform vision-based tracking on the captured frames. Since the vision-based tracking can be computationally expensive and it is crucial for an AR application to run at an interactive rate, we provided an option to thread both the video capturing and vision-based tracking processes from the main rendering thread.

A.1.3.1 Video Acquisition

We defined an interface class, *IVideoCapture* that specifies functionalities required for passing the decoded frame to the scene graph to be combined with the rendered graphics and to a vision-based tracking library. We currently support three different types of video-capture libraries: DirectShow, OpenCV, and PGRFly (from Point Grey Research, and used only with Point Grey cameras). Both DirectShow and OpenCV implementations can be used for regular webcams, but the DirectShow implementation provides better frame rate when multi-threaded from the rendering thread, while the OpenCV implementation streamlines the process of passing the decoded video frame to the OpenCV library for additional processing such as face or gesture recognition. In addition to the above implementations, we also provide an implementation that streams a series of still images to act as a fake video capture device for experimentation when a physical device is not available.

A.1.3.2 Vision-based Tracking

We defined an interface class, *IMarkerTracker* that specifies functionalities required for processing vision-based tracking on a given image and outputting the IDs of the markers found in the image as well as their transformations (3D position and orientation) relative to the video capture device. We currently support the ALVAR library [VTT, 2015], and provide a managed C++ wrapper to use the library in a managed environment. Useful tools such as camera calibration and marker layout described in Section A.2.1 and A.2.3 are also provided for streamlining application development.

A.1.3.3 Additional Features

To support stereo AR and optical-see through head-worn displays, we allow the programmer to acquire live videos from more than one capture devices and perform vision-based tracking on video frames from any of the attached devices. This feature also enables the programmer to use multiple capture devices in mono AR in a situation where a front-facing camera is used to display the physical environment while the other up-facing camera is used to perform vision-based tracking on markers attached on the ceiling.

A.1.4 Physics Engine

A physics engine is required for realistic rigid body physical simulation. We currently support three different types of physics engines: Newton [Newton Game Dynamics, 2015], Havok [Havok, 2015], and Matali [Komires, 2015], and provide managed C++ wrappers for accessing the methods implemented in Newton and Havok libraries. To support different implementations of physical simulation, we defined an interface class, *IPhysics* that specifies the initialization parameters for the simulation and methods for adding, modifying, and removing simu-

lation geometries specified by the *IPhysicsObject* interface. The *IPhysicsObject* interface class defines physical properties associated with geometry that are required for the specific implementation of the *IPhysics* class. An *IPhysicsObject* implementation will have the following properties:

Mass	The mass of the 3D object.
Shape	The shape that represents this 3D object (e.g., Box, Sphere, or Cylinder).
ShapeData	The detailed information of the specified shape (e.g., each dimension of a Box shape).
MomentOfInertia	The moment of inertia of the 3D object. (This can be computed automatically if not specified, but that will not guarantee the desired physical behavior.)
Pickable	Whether the object can be picked through mouse picking.
Collidable	Whether the object can collide with other 3D objects added to the physics engine.
Interactable	Whether the object can be moved by external forces.
ApplyGravity	Whether gravity should be applied to the object.

There are other physical properties that do not need to be set. However, setting them can, for example, allow an object to have initial linear or angular velocity when it is added to the physics engine. In addition to these properties, we also support material properties such as elasticity, softness, and static and kinetic frictions between objects depending on the implementation of the physics engine.

To integrate geometries in the scene to a physics engine seamlessly, we allow any Geometry nodes added to the scene graph to be added to a physics engine with a simple Boolean switch. Each Geometry node contains physical properties for its 3D model (i.e., it implements

IPhysicsObject), and the physics engine uses these properties to create an internal representation of the 3D model to perform physical simulation. If a Geometry node that is added to the physics engine is removed from the scene graph, then it is automatically removed from the physics engine.

Although physics simulation is quite common for a typical game engine, proper physics simulation in AR can be quite challenging, especially in the situation when we add simulation objects on a user-controllable marker [Buchanan et al., 2008]. Tracking errors and motion jitters can introduce unrealistic forces when markers or any vision-tracked objects are used to set the position and orientation of the virtual objects within the physics simulation. In order to avoid these issues and ensure reliable simulation, we have decided not to have the transformational changes on Marker or Tracker nodes affect the transformation of the virtual objects (Geometry nodes) attached to them by default. However, we provide a functionality to transform the virtual objects with appropriate force and torque within the physics simulation based on the transformation of a Marker or Tracker node using the inverse kinematic computation provided in Havok physics engine.

A.1.5 Networking

XNA Game Studio supports networking functionality specific to games, but it requires that the user log in to the XNA Live server, which can only connect to other machines through the XNA "lobby" system. While this works well for many kinds of games, it can be cumbersome if the programmer simply wants to connect a set of machines with known IP addresses or host names, and communicate efficiently among them, and will be unreliable if reliable internet access is not available. Therefore, Goblin XNA includes its own network interfaces that can be used for any application.

Our networking interface uses the client-server model in which a central server application communicates among multiple client applications. We provide four interface classes for handling network communication: *IServer*, *IClient*, *INetworkHandler*, and *INetworkObject*. An *INetworkHandler* implementation manages a list of *INetworkObject* with a unique identifier, which defines how an outgoing message will be encoded, how an incoming message will be decoded, and when or how often the message should be sent. An *INetworkHandler* implementation combines all messages into one message and sends the message through an *IServer* implementation if the application is a server or an *IClient* implementation otherwise. An *INetworkHandler* implementation also parses messages received from an *IServer* or *IClient* implementation and passes each parsed messages to appropriate *INetworkObject* identified by their unique identifiers. For example, a SynchronizedGeometry node, which extends the Geometry node, implements the *INetworkObject* interface and their transformations are synchronized among applications. Currently, our server and client implementations uses Lidgren [Lidgren, 2015] library.

A.1.6 2D User Interface

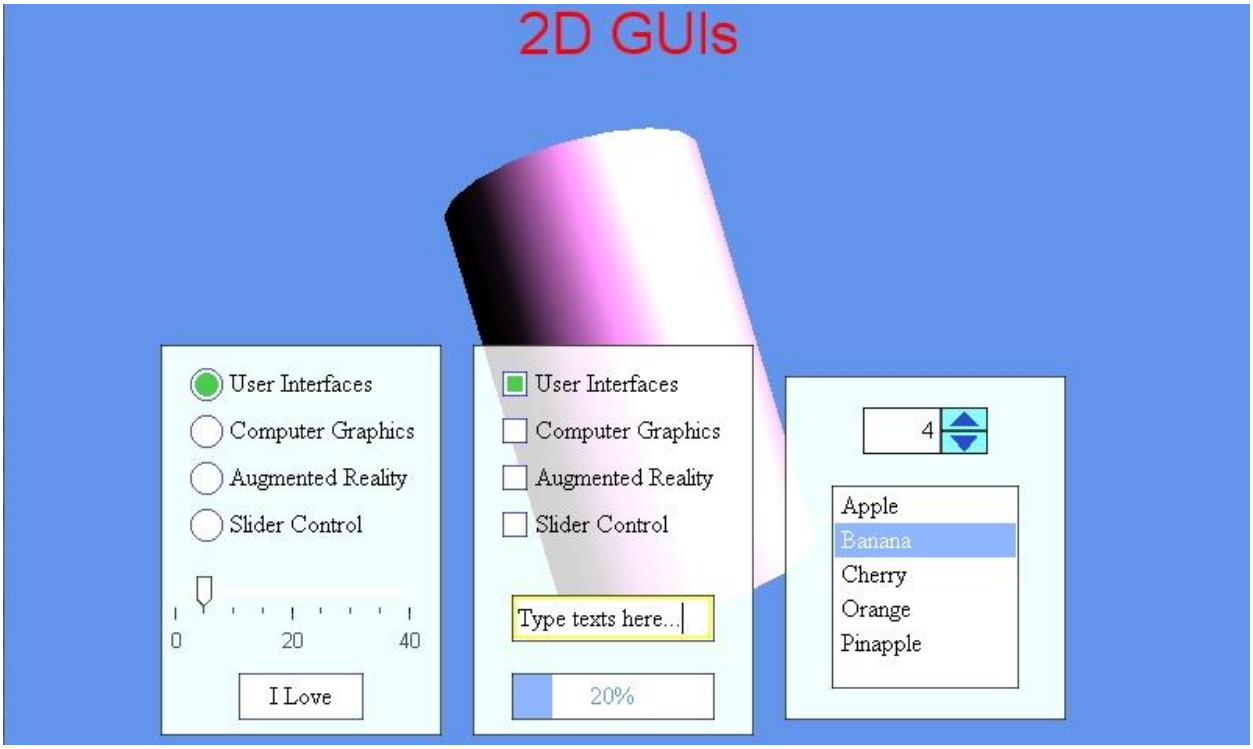

Figure A.6. A collection of 2D GUIs.

Goblin XNA provides a set of common 2D graphical user interface (GUI) components that can be overlaid on the scene, as shown in Figure A.6. We currently support basic 2D GUI components that include panel, label, button, radio button, check box, slider, text field, progress bar, list, and spinner. In addition to these basic ones, we have also provided GUIs that combine several of them such as media player controller and text field with suggestion list to demonstrate how a programmer can extend the basic elements. In Goblin XNA, 2D GUI components are overlaid on the 3D scene. They can be set to be transparent, so that both the GUI and the 3D scene behind it are visible. In addition, we support layers through callback functions so that the programmer can insert their own 2D drawings behind or above the 2D GUI layer by implementing the appropriate callback functions. The 2D GUI APIs are designed to be very similar to that of Java.Swing, including component properties and event handling (e.g., for button press actions).

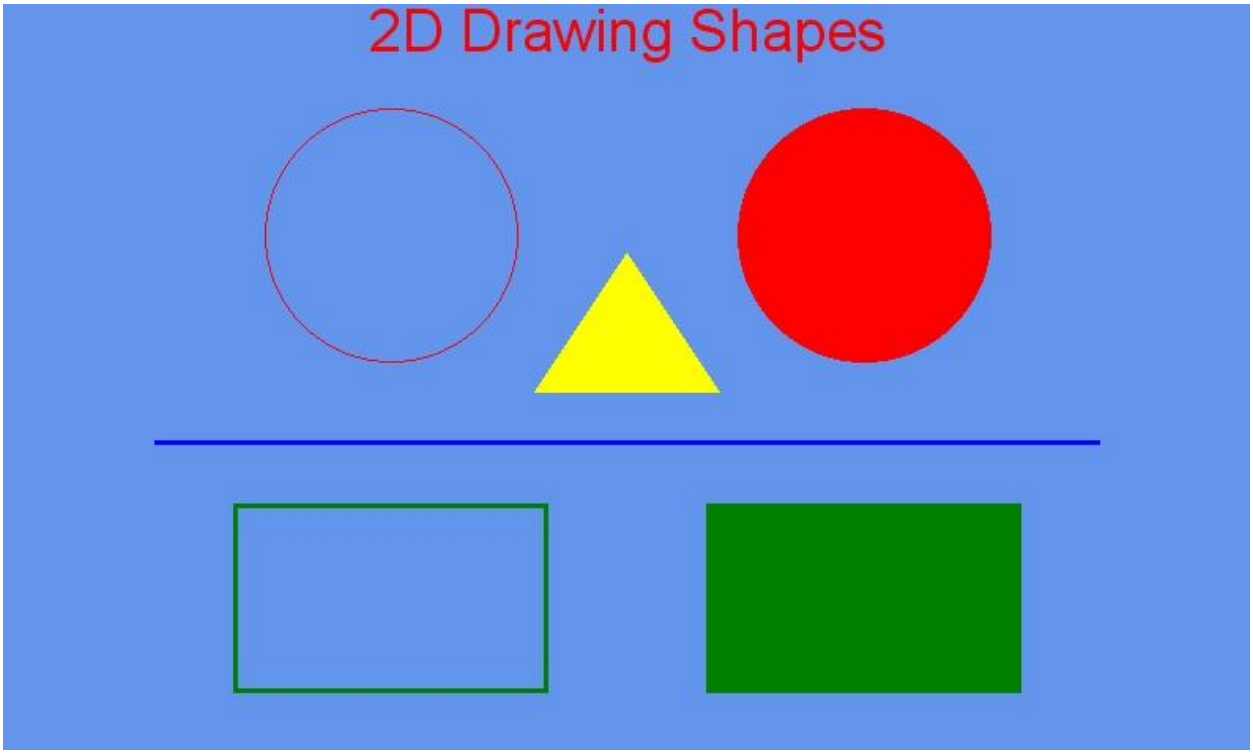

Figure A.7. A collection of 2D drawing shapes.

Additionally, Goblin XNA provides several functions for drawing simple 2D shapes such as line, rectangle, circle, and convex polygon as shown Figure A.7. Similar to Java2D, there are draw functions and fill functions.

A.1.7 Shaders

Instead of using a set of fixed-pipeline functions to render the 3D object and the scene, XNA Game Studio uses a programmable 3D graphics pipeline with the ability to create customized vertex shaders and pixel shaders. Even though this gives the programmer more power and flexibility in determining how the 3D objects in the scene are rendered, it means that the programmers will need to create their own shader to draw even a single triangle. Since many programmers new to the idea of a shader language do not want to spend time learning how to write shaders, we provide a set of shaders in Goblin XNA, including a simple effect shader that can

render many kinds of simple objects, a more advanced general shader, a particle shader, and a shadow mapping shader. All shaders implement the *IShader* interface class, and a programmer can implement their own shaders for rendering *IModel* objects.

A.2. Toolkits

While the framework provides a broad range of functionalities required for developing an AR application, there are still certain processes that the programmer has to undergo. To streamline these processes, we have developed several toolkits for calibrating the camera's intrinsic and extrinsic parameters, computing the stereo separation for a head-worn display, creating a custom marker layout, and debugging the Goblin XNA scene graph visually.

A.2.1 Camera Calibration

A vision-based tracking library usually requires camera-specific parameters such as lens distortion and focal length for accurate and stable tracking. A sample program for calibrating the camera parameters with a printed checkerboard is provided with the ALVAR distribution; however, it works only with regular webcams or point grey cameras. Since we also wanted to be able to calibrate cameras that cannot be connected to a computer, such as digital cameras with high resolution for photo shooting purposes, we have developed a camera calibration tool that can also calibrate a sequence of still images in addition to live video frames. The camera intrinsic parameters are computed from images of checkerboards captured from various angles and distances.

A.2.2 Stereo Camera Calibration

When viewing the world through a stereo head-worn display such as Vuzix Wrap920AR, the exact spatial and rotational difference between the two cameras need to be known in order to present the virtual contents properly overlaid on the physical environment. In addition to the ste-

reo separation, the field of each camera needs to be adjusted to match the field of view of the wearer's corresponding eye for proper stereoscopic AR experience. To address these issues, we developed a tool to automatically calibrate the stereo separation using the camera intrinsic parameters obtained from our camera calibration tool described in Section A.2.1 and a marker array. Once the stereo separation is computed, a collection of virtual contents are presented on the marker array to confirm the computed separation is correct. Finally, the wearer uses the keyboard to manually adjust the field of view and the vertical and horizontal shift seen through the head-worn display to more closely match the actual view seen with bare eyes.

A.2.3 Marker Layout

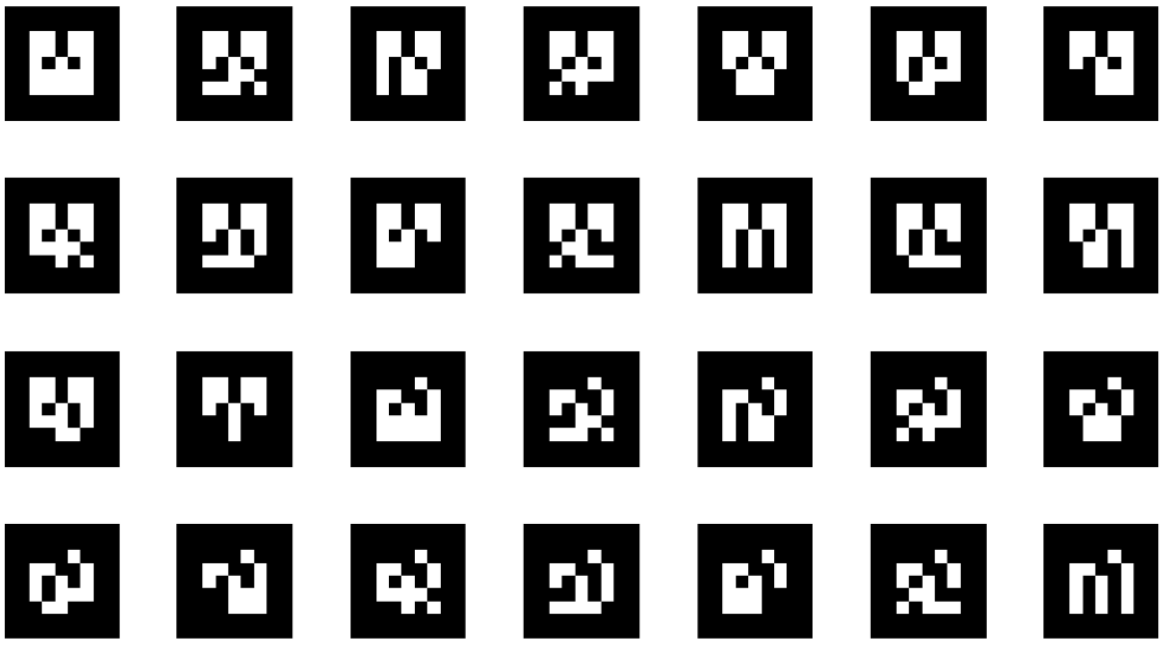

Figure A.8. A 4x7 marker array created using the Marker Layout tool.

Vision-based tracking libraries such as ALVAR, ARTag, and ARToolkit all provide programs for generating a single marker image with a given numerical identifier. However, none of them provide a tool for laying out an array of markers, even though all support multi-marker tracking. This is unfortunate, because an array of markers usually makes it possible for more sta-

ble tracking and covers a larger tracking area compared to a single marker. While tracking an array requires more computational power, current machines are quite powerful, so it is more reasonable and beneficial to track multiple markers rather than a single marker.

In order to track multiple markers, it is necessary to prepare an image that contains all markers and a configuration file that defines the size and 2D position of each marker. Even though it is relatively trivial to create a layout image manually by using image editing software such as Photoshop, it can be quite time consuming to create one for a large number of markers, such as the one shown in Figure A.8. It is also time consuming and error-prone to manually prepare a configuration file. Modifying an existing layout image or configuration file can be quite laborious as well.

```
// Create a layout manager with size 400x400 pixels, and actual marker size of 9 inches
LayoutManager layout = new LayoutManager(400, 400, 9);

// Create arrays of marker IDs we want to layout
// NOTE: Please use the SampleMarkerCreator project that comes with the ALVAR
// package to generate the raw marker images
int[] array1 = { 0, 1 };
int[] array2 = { 2, 3 };

int[][] marker_arrays = new int[2][];
marker_arrays[0] = array1;
marker_arrays[1] = array2;

// Layout the markers
for (int j = 0; j < 2; j++)
{
    for (int i = 0; i < 2; i++)
        layout.AddMarker(marker_arrays[j][i], new Point(60 + j * 172, 60 + i * 172),
            "raw_markers/ALVAR/MarkerData_" + marker_arrays[j][i] + ".png");
}

// Set the (0, 0) point in the configuration file to be at (60, 60) in the layout image
// In this case, it is at the left-upper corner of marker ID 0.
layout.ConfigCenter = new Point(60, 60);

// Compile the layout
layout.Compile();

// Output the layout image in gif format
layout.OutputImage("ALVARArray.gif", ImageFormat.Gif);

// Output the configuration file
layout.OutputConfig("ALVARConfig.xml", LayoutManager.ConfigType.ALVAR);
```

Figure A.9. An example program for generating a marker layout image and a configuration file.

To address these difficulties, we have implemented a tool that generates the marker layout image and the configuration file programmatically. As shown by the example in Figure A.9, the programmer simply needs to provide the image size in pixels, the size of each marker, the upper-left corner positions of each marker in pixels, and the center position (0, 0) of the marker array in pixels.

A.2.4 Graphical Scene Graph Display

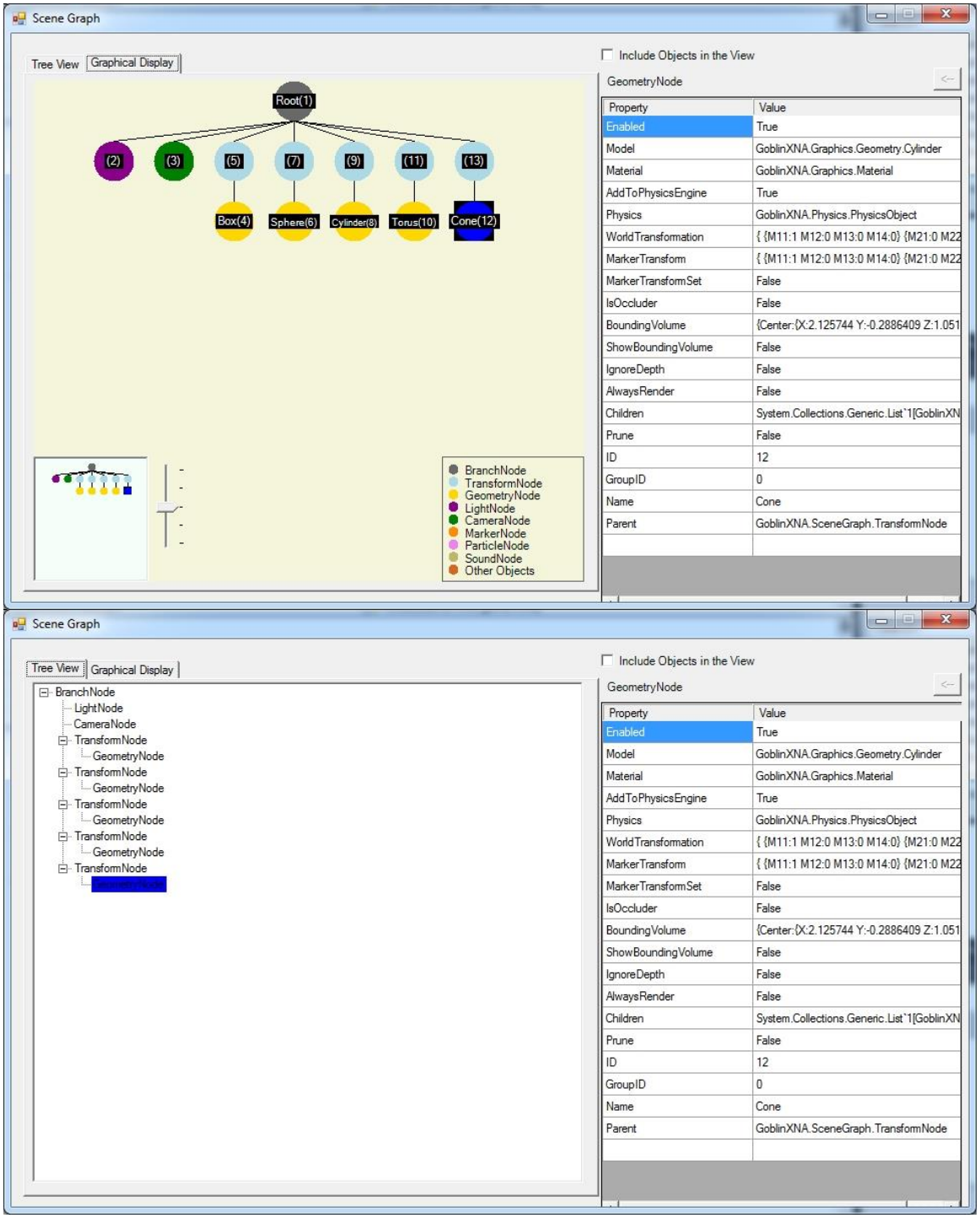

Figure A.10. A separate window from the Goblin XNA window showing the scene graph structure. (Top) A graphical view of scene graph structure is displayed on the left panel, and the selected node's (node 12) information is displayed on the right panel. (Bottom) A tree view of the scene graph structure.

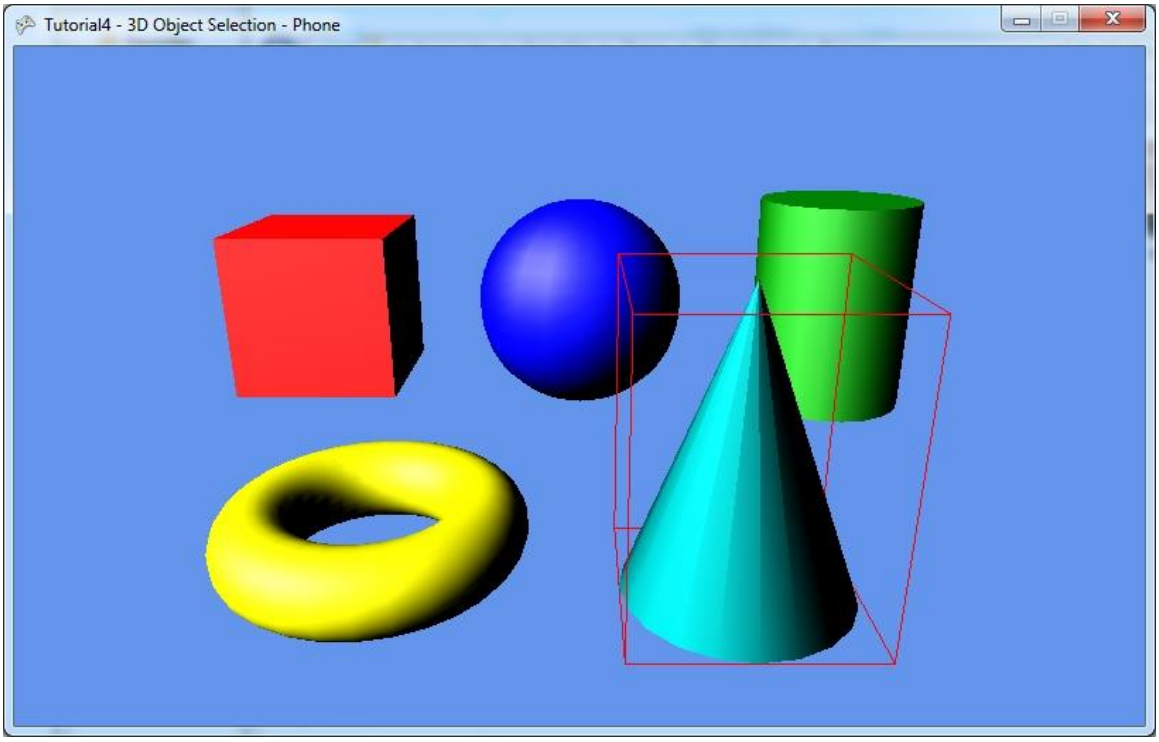

Figure A.11. The Goblin XNA application running with the scene graph display. A selected 3D model (node 12) is also highlighted in the 3D scene when the corresponding Geometry node is selected in the scene graph display view.

Even though it is relatively straightforward to manage a 3D scene with a scene graph structure, it can be quite challenging to identify a bug when the 3D objects do not behave as expected, especially for programmers who are unfamiliar with the concept of a scene graph or when nodes are deeply nested. A traditional way of debugging such as printing values on the console window or utilizing break points in the IDE can be inefficient especially when the programmer is unsure of which values to track.

To address these issues, we developed a tool to visualize the hierarchical tree structure of the scene of a running Goblin XNA application. As depicted in Figure A.10, the scene graph display runs on a separate window from the Goblin XNA application and renders the scene graph in 2D with different node colors depending on its type (Figure A.10, top). Each node contains its name if assigned and its numerical ID. Alternatively, a programmer can view the scene graph in

tree view (Figure A.10, bottom). When a node is clicked either on the graphical view or tree view, the node is highlighted and the property values are displayed on the right panel. These values get updated dynamically as the application runs. When a Geometry node is highlighted on the scene graph display, a corresponding 3D object in the Goblin XNA application is also highlighted with a bounding box as shown in Figure A.11. It is also possible to click the 3D objects in the Goblin XNA application and reveal its property values on the scene graph display. Any changes to the scene graph (e.g., node removal or addition) are reflected on the fly in the scene graph display.

A.3. Iteration History

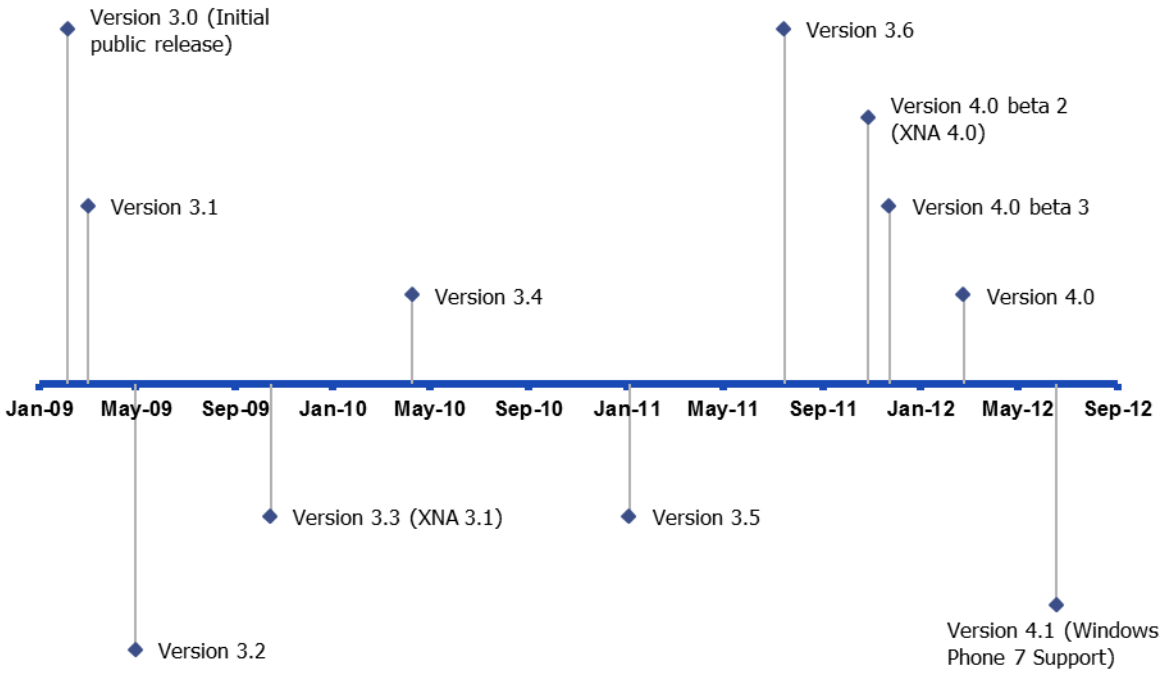

Figure A.12. Timeline of Goblin XNA public release.

Goblin XNA was initially built on top of a C# based commercial game engine, Truevision3D, which no longer exists, with additional features required for AR such as live video ac-

quisition and vision-based tracking in early 2007. However, due to several issues with their scene graph implementation, we decided to implement our own scene graph directly on top of the XNA framework and incorporate third-party libraries for supporting physics simulation and networking. The framework was released to the public in early 2009 at the CodePlex website under a BSD license. It was updated frequently with bug fixes and new features until 2012.

Prior to public release, the initial XNA-based version of the framework was built on top of XNA 2.0. XNA had been updated to version 3.0 by the time of the Goblin XNA initial public release followed by version 3.1 for Goblin XNA version 3.4, and finally, version 4.0 for Goblin XNA version 4.0 beta 2, as shown in Figure A.12. With a few thousand lines of additional code, we added support for Windows Phone 7 beginning in Goblin XNA version 4.1.

A.4. Projects Implemented with Goblin XNA

Several interesting AR applications have been developed using Goblin XNA framework in and out of our lab. I will briefly present in chronological order a few of the projects in which I was directly involved, but which are not part of my dissertation. These projects have all been presented at conferences.

A.4.1 AR Racing Game

Figure A.13. AR racing game. (a) User holding an optically tracked controller and controlling virtual car on tracked physical gameboard. (b) Car collides with virtual signpost.

The AR racing game [Oda et al., 2008] was the first project developed in the Goblin XNA framework and was presented at the *2007 Microsoft Research Faculty Summit* and at *INTETAIN 2008* (*2nd International Conference on Intelligent Technologies for Interactive Entertainment*) . This project demonstrates the process of transforming a non-AR game, XNA Racing Game [exDream, 2009], to a multi-player AR experience that supports direct spatial interaction. In our game, the driver wears a tracked video–see-through head-worn display, and controls the car with a passive tangible controller (Figure A.13a). Other players can participate by manipulating waypoints that the car must pass and obstacles with which the car can collide.

To emphasize the interaction with the physical world and remove the artificial boundaries of the game world, we eliminated the predefined environment of the original game, including the track. Instead of being restricted to the track, the driver can drive the car anywhere she wants on the ground plane, which is defined by a planar array of optical markers forming a gameboard of essentially arbitrary size. We then redefined the completion of a lap to require the driver to pass through a series of designated waypoints in sequence, potentially slowed by collisions with obstacles. Virtual objects that act as physical obstacles are overlaid on the gameboard (Figure

A.13b) and can be attached to separate movable markers that can be manipulated by additional players.

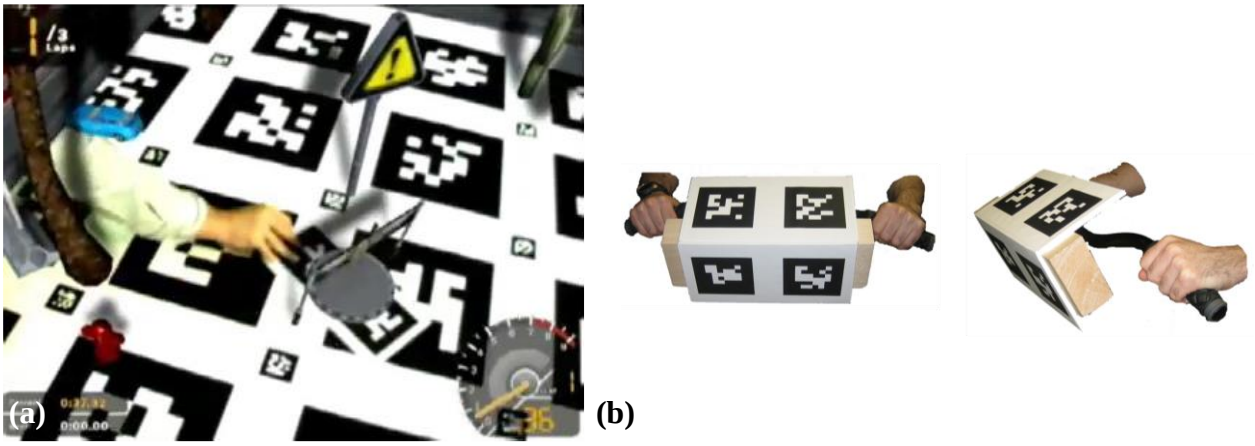

Figure A.14. (a) A non-driver player moving a waypoint attached to a marker array. (b) The controller used by a driving player to control the virtual car.

Unlike traditional console games, in which a limited number of players have access to controllers through which they can participate in the game, marker tracking makes it much easier to support additional players who can manipulate game objects. To take advantage of this, we attached environment objects to separate marker arrays. This enables observers to become non-driver players, who can work alone or together to assist or hinder the driver by dynamically modifying the environment during game play (Figure A.14a). These additional players view the game and the effect they have on it through a stationary display associated with a camera positioned and oriented arbitrarily in the environment, as long as it has an adequate view of the gameboard.

Rather than relying on conventional wired or wireless game controllers, or special-purpose driving controllers, we decided to use a passive tangible interaction device to function as the game controller in the spirit, for example, of Fiala's magic mirror device [Fiala, 2007]. Our driving controller (Figure A.14b) consisted of a fiducial marker array mounted on a two pieces of foamcore board, which are rigidly fixed to a pair of bicycle handlebars. (We chose the mixed

metaphor of a bicycle controller for a racing car because we found the handlebars to make a more comfortable unattached controller than a steering wheel.) Turning is accomplished by rotating the controller roughly parallel to the ground plane, while the car is accelerated or decelerated by tilting the controller forward or backwards.

A.4.2 AR Marble Game

Figure A.15. (Top) The AR marble game, viewed through a video–see-through, head-worn display. The virtual marble and dice roll and slide as if acted on by gravity. (Bottom) A player holding the optically tracked gameboard for the AR marble game in a demonstration in the Vuzix booth at the *2010 Consumer Electronics Show* (http://eyewearseeyou.blogspot.com/2010/01/vuzix-wrap-920ar-3d-video-glasses-with.html) .

AR marble game [Oda and Feiner, 2010] is the first gravity-aware AR game and has been demonstrated at numerous exhibits and conferences including *CHI 2010*. The game simulates a mixed reality version of Labyrinth [Labyrinth, 2010] experienced through a video–see-through head-worn display, as shown in Figure A.15, top. The player holds a gameboard covered with a printed array of fiducial markers (Figure A.15, bottom), and the markers are optically tracked by a camera in the head-worn display. The player must guide a virtual ball through a maze of static and dynamic obstacles that appear on the board by tilting and translating the board.

Gravity in the game always points in the correct direction, no matter how the player's head and board are oriented relative to each other. This is possible because, in addition to optical tracking, the game takes advantage of a separate head orientation tracker built into the head-worn display. The optical tracker determines the gameboard's position and orientation relative to the head-worn display. The built-in head tracker senses real gravitational acceleration to determine which way is "down," which we use to compute the direction of gravitational acceleration to be applied in the gameboard's coordinate system. Thus, the marble and other items on the game-board will roll or slide under the influence of gravity, which is modeled using a rigid body simulation.

A.4.3 AR Modeling on Microsoft Surface

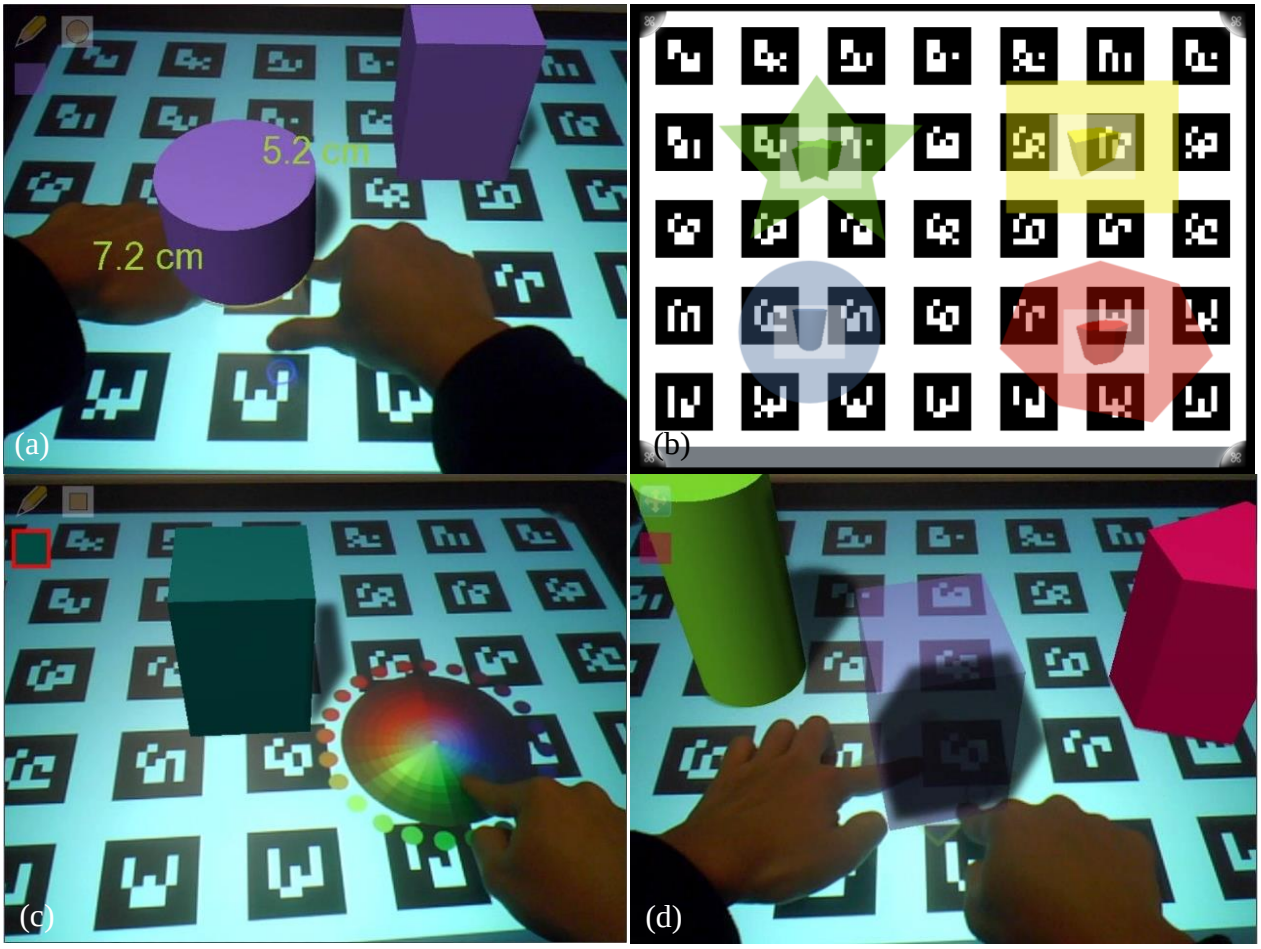

Figure A.16. 3D modeling using video–see-through head-worn display and Microsoft Surface. (a) User specifying a cylinder with 2D gestures on the Surface (b) Semi-transparent 2D footprint displayed on the Surface. (c) User modifying the color of a selected model. (d) User rotating a selected model.

We prototyped a simple 3D modeling system that runs on a video–see-through display combined with a 2D Microsoft Surface [Dedual et al., 2011] as shown in Figure A.16. The Surface shows semi-transparent 2D footprints of simple geometric models and other parts of the user interface, while the tracked video–see-through head-worn display presents the full 3D models overlaid on their footprints on the Surface. This makes it possible for a user wearing the head-worn display to view the 3D geometric models in AR, as if they were physically located on the Surface. Multi-touch gestures are used to create, edit, delete, and select the models. The head-

worn display's built-in camera tracks a marker array projected on the Surface. Figure A.16(a) shows a user creating a cylinder. A user who views the Surface without a head-worn display still sees the object footprints and construction interface, shown in Figure A.16(b), which are spatially registered with the models. Those users can also interact with the Surface and can point at the footprints within the same space viewed by users wearing head-worn displays.

A.4.4 ARmonica

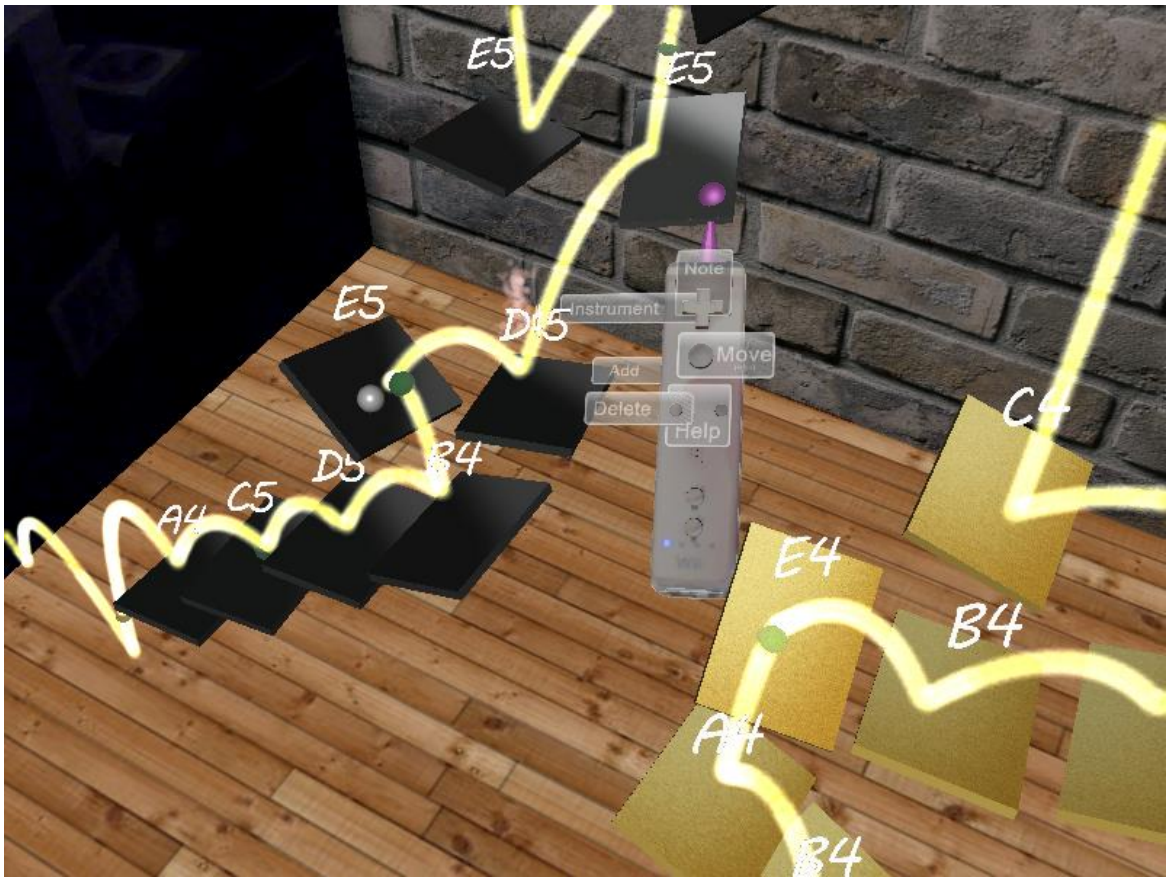

Figure A.17. ARmonica, a collaborative sonic environment, viewed through a video–see-through, head-worn display. A player ready to add musical notes (E5 note), represented as black bars, to the environment. Yellow translucent trail of ball's trajectory includes green translucent hemispherical tick marks at regular temporal intervals. This image shows the version of our system demoed at *UIST 2010*.

ARmonica [Sükan et al., 2010] is a physically-based 3D audiovisual AR environment inspired by several earlier systems such as Nimoy's BallDroppings [Nimoy, 2015] and Lytle's

physically-based animation [Lytle, 1991]. This application was developed to explore collaborative audio-visual play, using heterogeneous displays and interaction devices, and was demonstrated at *UIST 2010*. Players experience ARmonica through head-tracked head-worn displays and tracked handheld ultramobile personal computers, and interact through tracked Wii remotes and touch-screen taps.

The goal is for players to collaborate to create an AR environment in which virtual balls are fired at virtual bars that, when struck, emit sounds defined by musical instrument samples, accompanied by particle effect visualizations (Figure A.17). Players can place bars and ball launchers that hover over a table covered by an optical marker array. Ball launchers launch virtual balls into the scene based on player-set parameters, including ball generation frequency and initial velocity. Balls and bars interact in a physically realistic manner; each time a bar is hit, it generates a sound corresponding to a player-assigned instrument and note.

Players are provided with context-dependent visual overlays to assist in designing the environment. Each instrument is associated with a different texture (e.g., rosewood for marimba, metal for vibraphone, and black lacquer for piano). The Wii remotes are overlaid with tracked virtual labels that document each button's current function. To guide the placement of bars and ball launchers, we display a real-time trajectory of a ball as a yellow translucent trail as shown in Figure A.17. This real-time trajectory is computed ahead of time using physics simulation performed on a separate machine and changes as the user moves the bars or the ball launchers. Tick marks in the form of transparent hemispheres are displayed along a trail at locations that segment the trail into equal temporal intervals. These can visually assist a player in establishing a regular rhythm, if desired, by positioning bars tangent to the hemispheres.

Appendix B. Study Questionnaires

B.1. Questionnaire for Chapter 3

Participants were instructed to ignore the references to techniques E–F in the questionnaire, since only four techniques (A–D) were presented. Participants were also told the mapping between the technique names used during the study ("Laser Pointer", "Virtual Arrow", "Video Share", and "Sphere Select") and the technique labels (A–D) used in the questionnaire.

Comparative Study of 3D Referencing Techniques in a Shared Augmented Reality Environment

Participant ID: _____
IRB Protocol: IRB- AAAJ0655
Principal Investigator: Steven Feiner (skf1)
Co-Investigator: Ohan Oda (oo2116)

User Experience Survey

Date:

Age: Gender: F / M

I use a computer…

never
monthly
weekly
daily
multiple times per day

I am familiar with Augmented Reality

no
yes

For each question, we would appreciate any additional comments you have in the "Comments" section.

PART I. For the following questions, please circle a number from 1 through 5 to describe your experience with how easy it was to refer to an object at distance when you were performing the selection role, and how easy it was to interpret the referenced object when you were performing the interpretation role.

A:

	Selection:	hard				easy
		1	2	3	4	5
	Interpretation:	hard				easy
		1	2	3	4	5

Comments:

B:

	Selection:	hard				easy
		1	2	3	4	5
	Interpretation:	hard				easy
		1	2	3	4	5

Comments:

C:

	Selection:	hard				easy
		1	2	3	4	5
	Interpretation:	hard				easy
		1	2	3	4	5

Comments:

D:

	Selection:	hard				easy
		1	2	3	4	5
	Interpretation:	hard				easy
		1	2	3	4	5

Comments:

E:

	hard				easy
Selection:	1	2	3	4	5
Interpretation:	1	2	3	4	5

Comments:

F:

	hard				easy
Selection:	1	2	3	4	5
Interpretation:	1	2	3	4	5

Comments:

PART II. In the following questions, please place the number 1 (best) through 6 (worst) next to each choice. If you feel that multiple techniques perform roughly the same, then give them the same ranking.

Rank the techniques by how easy they are for referring to an object, from 1 (easiest) to 6 (hardest).

___ A

___ B

___ C

___ D

___ E

___ F

Comments:

Rank the techniques by how easy they are for interpreting which object is being referenced, from 1 (easiest) to 6 (hardest).

___ A

___ B

___ C

___ D

___ E

___ F

Comments:

Rank the techniques by how fast you think they are for referring to an object, from 1 (fastest) to 6 (slowest).

___ A

___ B

___ C

___ D

___ E

___ F

Comments:

Rank the techniques by how fast you think they are for interpreting which object is being referenced, from 1 (fastest) to 6 (slowest).

___ A

___ B

___ C

___ D

___ E

___ F

Comments:

Rank the techniques by how accurate you think they are for referring to an object, from 1 (most accurate) to 6 (least accurate).

___ A

___ B

___ C

___ D

___ E

___ F

Comments:

Rank the techniques by how accurate you think they are for interpreting which object is being referenced, from 1 (most accurate) to 6 (least accurate).

___ A

___ B

___ C

___ D

___ E

___ F

Comments:

Please provide any additional comments about or reactions to any of the techniques:

B.2. Questionnaire for Chapter 4

The three techniques (T, P, A) were presented and referred to by their letter names.

Remote Guidance Study

* Required

1. **Participant ID (Ask the study coordinator) ***

2. **Age ***

3. **Gender ***
   Mark only one oval.
   - Female
   - Male

4. **I use a computer... ***
   Mark only one oval.
   - Never
   - Monthly
   - Weekly
   - Daily
   - Multiple times per day

5. **I have experience using Augmented Reality systems ***
   Mark only one oval.
   - No
   - Yes

6. **If you answered "Yes" to the previous question, please explain your experience.**

7. **Which of the following roles have you been assigned?** *
Mark only one oval.

- ◯ Instructor
- ◯ Trainee

8. **How would you rate your mechanical ability?** *
Mark only one oval per row.

	1 (Poor)	2	3	4	5	6	7 (Excellent)
	◯	◯	◯	◯	◯	◯	◯

Part 1: Technique T

For the following questions, please choose a number from 1 through 7 to describe your experience with technique T.

For each question, we would appreciate any additional comments you have in the "Comments" section.

9. **Mental Demand** *
How mentally demanding was the task?
Mark only one oval per row.

	1 (Very Low)	2	3	4	5	6	7 (Very High)
	◯	◯	◯	◯	◯	◯	◯

10. **Physical Demand** *
How physically demanding was the task?
Mark only one oval per row.

	1 (Very Low)	2	3	4	5	6	7 (Very High)
	◯	◯	◯	◯	◯	◯	◯

11. **Temporal Demand** *
How hurried or rushed was the pace of the task?
Mark only one oval per row.

	1 (Very Low)	2	3	4	5	6	7 (Very High)
	◯	◯	◯	◯	◯	◯	◯

12. **Performance** *
How successful were you in accomplishing what you were asked to do?
Mark only one oval per row.

	1 (Perfect)	2	3	4	5	6	7 (Failure)
	◯	◯	◯	◯	◯	◯	◯

13. **Effort** *
How hard did you have to work to accomplish your level of performance?
Mark only one oval per row.

	1 (Very Low)	2	3	4	5	6	7 (Very High)
	◯	◯	◯	◯	◯	◯	◯

14. **Frustration** *
How insecure, discouraged, irritated, stressed and annoyed were you?
Mark only one oval per row.

	1 (Very Low)	2	3	4	5	6	7 (Very High)
	◯	◯	◯	◯	◯	◯	◯

15. **Please provide any additional comments about or reactions to technique T.** *

.....

.....

.....

.....

.....

Part 2: Technique P

For the following questions, please choose a number from 1 through 7 to describe your experience with technique P.

For each question, we would appreciate any additional comments you have in the "Comments" section.

16. **Mental Demand** *
How mentally demanding was the task?
Mark only one oval per row.

	1 (Very Low)	2	3	4	5	6	7 (Very High)
	◯	◯	◯	◯	◯	◯	◯

17. **Physical Demand** *
How physically demanding was the task?
Mark only one oval per row.

	1 (Very Low)	2	3	4	5	6	7 (Very High)
	◯	◯	◯	◯	◯	◯	◯

18. **Temporal Demand** *
How hurried or rushed was the pace of the task?
Mark only one oval per row.

	1 (Very Low)	2	3	4	5	6	7 (Very High)
	◯	◯	◯	◯	◯	◯	◯

19. **Performance** *
How successful were you in accomplishing what you were asked to do?
Mark only one oval per row.

	1 (Perfect)	2	3	4	5	6	7 (Failure)
	◯	◯	◯	◯	◯	◯	◯

20. **Effort** *
How hard did you have to work to accomplish your level of performance?
Mark only one oval per row.

	1 (Very Low)	2	3	4	5	6	7 (Very High)
	◯	◯	◯	◯	◯	◯	◯

21. **Frustration** *
How insecure, discouraged, irritated, stressed and annoyed were you?
Mark only one oval per row.

	1 (Very Low)	2	3	4	5	6	7 (Very High)
	◯	◯	◯	◯	◯	◯	◯

22. **Please provide any additional comments about or reactions to technique P.** *

.....

.....

.....

.....

.....

Part 3: Technique A

For the following questions, please choose a number from 1 through 7 to describe your experience with technique A.

For each question, we would appreciate any additional comments you have in the "Comments" section.

23. **Mental Demand** *
How mentally demanding was the task?
Mark only one oval per row.

	1 (Very Low)	2	3	4	5	6	7 (Very High)
	◯	◯	◯	◯	◯	◯	◯

24. **Physical Demand** *
How physically demanding was the task?
Mark only one oval per row.

	1 (Very Low)	2	3	4	5	6	7 (Very High)
	◯	◯	◯	◯	◯	◯	◯

25. **Temporal Demand** *
How hurried or rushed was the pace of the task?
Mark only one oval per row.

	1 (Very Low)	2	3	4	5	6	7 (Very High)
	◯	◯	◯	◯	◯	◯	◯

26. **Performance** *
How successful were you in accomplishing what you were asked to do?
Mark only one oval per row.

	1 (Perfect)	2	3	4	5	6	7 (Failure)
	◯	◯	◯	◯	◯	◯	◯

27. **Effort** *
How hard did you have to work to accomplish your level of performance?
Mark only one oval per row.

	1 (Very Low)	2	3	4	5	6	7 (Very High)
	◯	◯	◯	◯	◯	◯	◯

28. **Frustration** *
How insecure, discouraged, irritated, stressed and annoyed were you?
Mark only one oval per row.

	1 (Very Low)	2	3	4	5	6	7 (Very High)
	◯	◯	◯	◯	◯	◯	◯

29. **Please provide any additional comments about or reactions to technique A.** *

Part 4: Overall Preference

30. **Rank the techniques from 1 (Least Preferred) to 3 (Most Preferred)** *
Mark only one oval per row.

	1 (Least Preferred)	2	3 (Most Preferred)
Technique P			
Technique T			
Technique A			

31. **Please provide any additional comments below.**

B.3. Questionnaire for Chapter 5

Participants were instructed to ignore the references to technique F in the questionnaire, since only five techniques (A–E) were presented, referred to by their letter names.

Comparative Study of Interference Avoidance Techniques for 3D Multi-User Applications

Participant ID: _____
IRB Protocol: IRB-AAAD7189
Principal Investigator: Steven Feiner (skf1)
Co-Investigator: Ohan Oda (oo2116)

User Experience Survey

Date:

Age: Gender: F / M

I use a computer…

never
monthly
weekly
daily
multiple times per day

For each question, we would appreciate any additional comments you have in the "Comments" section.

COLUMBIA UNIVERSITY
INSTITUTIONAL REVIEW BOARD
IRB# AAAD7189 Approval Date: 12/06/08
IRB Initials: AV Expiration Date: 12/05/09

PART I. Interference Avoidance Techniques. For the following questions, please circle a number from 1 through 5 to describe your experience using the experimental systems.

A:

not effective					very effective
1	2	3	4	5	
very distracting					not distracting
1	2	3	4	5	

Comments:

B:

not effective					very effective
1	2	3	4	5	

very distracting					not distracting
1	2	3	4	5	

Comments:

C:

not effective				very effective
1	2	3	4	5

very distracting				not distracting
1	2	3	4	5

Comments:

4

D:

not effective				very effective
1	2	3	4	5

very distracting				not distracting
1	2	3	4	5

Comments:

5

E:

not effective				very effective
1	2	3	4	5

very distracting				not distracting
1	2	3	4	5

Comments:

6

F:

not effective				very effective
1	2	3	4	5

very distracting				not distracting
1	2	3	4	5

Comments:

PART II. For the following questions, please place a 1 (best) through 6 (worst) next to each choice.

Rank the techniques by how effective they are, from 1 (most effective) to 6 (least effective).

___ A

___ B

___ C

___ D

___ E

___ F

Comments:

Rank the techniques by how distracting they are, from 1 (least distracting) to 6 (most distracting).

___ A

___ B

___ C

___ D

___ E

___ F

Comments:

Please provide any additional comments about or reactions to any of the techniques:

References (Appendix)

Buchanan, P., Seichter, H., Billinghurst, M., and Grasset, R., "Augmented reality and rigid body simulation for edutainment: the interesting mechanism - an AR puzzle to teach Newton physics," *Proceedings of the 2008 International Conference on Advances in Computer Entertainment Technology*, 2008, pp. 17-20.

Dedual, N. J., Oda, O., and Feiner, S. K., "Creating hybrid user interfaces with a 2D multi-touch tabletop and a 3D see-through head-worn display," *Proceedings of the 10th IEEE International Symposium on Mixed and Augmented Reality*, 2011, pp. 231-232.

Eaddy, M. and Feiner, S., "Multi-Language Edit-and-Continue for the Masses," *Technical Report CUCS-015-05, Department of Computer Science, Columbia University*, 2005.

exDream. *XNA Racing Game*. (2009). http://create.msdn.com/en-US/education/catalog/sample/racing_game

Fiala, M., "Magic Mirror System with Hand-held and Wearable Augmentations," *Virtual Reality Conference*, 2007, pp. 251-254.

Havok. (2015). *Havok Physics*. http://www.havok.com/

Komires. (2015). *Matali Physics*. http://www.mataliphysics.com/

Labyrinth. (2010). http://en.wikipedia.org/wiki/Labyrinth_(game)

LaViola, J. J., "Double exponential smoothing: an alternative to Kalman filter-based predictive tracking," *Proceedings of the workshop on Virtual environments*, 2003, pp. 199-206.

Lidgren. (2015). *Lidgren Network Library*. http://code.google.com/p/lidgren-network/

Lytle, W., "More bells and whistles," *SIGGRAPH '90 Film Show*, 1991.

Newton Game Dynamics. (2015). http://www.newtondynamics.com/

Nimoy, J. (2015). *BallDroppings*. http://balldroppings.com

Oda, O. and Feiner, S., "Rolling and shooting: two augmented reality games," *Proceedings of the 28th of the international conference extended abstracts on Human factors in computing systems*, 2010, pp. 3041-3044.

Oda, O. and Feiner, S. (2012). *Goblin XNA*. http://goblinxna.codeplex.com

Oda, O., Lister, L. J., White, S., and Feiner, S., "Developing an augmented reality racing game," *INTETAIN*, 2008, pp. 1-8.

Reiners, D., Voss, G., and Neumann, C. (2015). *OpenSG*. http://www.opensg.org/

SGI. (2015). *GLUT*. http://www.opengl.org/resources/libraries/glut/

Sükan, M., Oda, O., Shi, X., Entrena, M., Sadalgi, S., Qi, J., and Feiner, S., "ARmonica: a collaborative sonic environment," *Adjunct proceedings of the 23nd annual ACM symposium on User interface software and technology*, 2010, pp. 401-402.

VTT. (2015). *ALVAR*. http://www.vtt.fi/multimedia