

**WITHINDOWS: A UNIFIED FRAMEWORK FOR THE DEVELOPMENT
OF DESKTOP AND IMMERSIVE USER INTERFACES**

BY

ALEX S. HILL
B.S., Trinity University, 1988
M.S., University of Texas at Austin, 1992

THESIS

Submitted as partial fulfillment of the requirements
for the degree of Doctor of Philosophy in Computer Science
In the Graduate College of the
University of Illinois at Chicago, 2007

Chicago, Illinois

I would like to dedicate this thesis to my grandfather, Alex S. Hill Sr., without whom I would have never had the opportunity to pursue it.

ACKNOWLEDGEMENTS

I would like to thank my thesis committee, Andrew Johnson, Daniel Sandin, Jason Leigh, Tom Moher, and Steve Jones, for their continued patience and support. I would also like to acknowledge the support of all the staff, faculty and students at the Electronic Visualization Laboratory and the School of Art and Design that have provided the support, emotional, technical and financial, that made this research possible.

ASH

TABLE OF CONTENTS

<u>CHAPTER</u>	<u>PAGE</u>
1. INTRODUCTION.....	1
1.1 The Ultimate Interface	1
1.2 Augmenting Human Ability	2
1.3 The Coming Augmented Reality.....	3
1.4 Efficient Augmented Reality Interfaces.....	4
1.5 The Withindows Framework	5
2.0 THREE DIMENSIONAL USER INTERFACES	8
2.1 Human Computer Interaction.....	8
2.1.1 Augmenting Human Ability.....	8
2.1.2 Virtual Reality	10
2.1.3 The Emergence of Two Themes	11
2.2 The Role of Haptics	12
2.2.1 Interaction with the Hands	13
2.2.2 Travel in Virtual Space.....	14
2.2.3 Virtual Travel Techniques	15
2.2.4 A Separation between Virtual and Physical Reality.....	16
2.3 Proprioceptive Interaction.....	17
2.3.1 Selecting and Manipulating Objects	18
2.3.2 The Separation between Egocentric and Exocentric Interaction.....	20
2.4 Manipulation at a Distance	22
2.4.1 Distance Manipulation Techniques.....	23
2.4.2 Corporealism.....	24
2.5 Relationship to Desktop Systems.....	26
2.5.1 System Control	27
2.5.2 Usability.....	31
3. THE COMING AUGMENTED REALITY	37
3.1 A Decontextualized Computer Revolution.....	37
3.2 The Emergence of Three Trends	39
3.2.1 The Geo-Referenced Internet.....	39
3.2.2 The Ubiquitous Internet.....	40
3.2.3 The Ubiquitous Display	41
3.3 A Truly Augmenting Reality	42
3.3.1 A Truly Universal Remote	43
3.3.2 The Application of Computation to the Physical	44
3.4 Future 3D User Interfaces	46
4. GUIDELINES FOR PRODUCTIVE 3D USER INTERFACES	48
4.1 Domain.....	48
4.1.1 Simulation	49
4.1.2 Augmentation	49
4.1.3 Augmentation Application Areas.....	50
4.1.4 A Superset of Two Dimensional Interfaces.....	51
4.2 Usability	51
4.2.1 Novice Use.....	51
4.2.2 Expert Use	52
4.2.3 Fatigue	52
4.2.4 Workflow	53
4.3 Efficiency.....	53
4.3.1 First Person Perspective.....	53
4.3.2 Third Person Perspective.....	54
4.3.3 Input Methods	55
4.3.4 Symbolic Manipulations	55

TABLE OF CONTENTS(continued)

<u>CHAPTER</u>	<u>PAGE</u>
4.4 Application Development.....	56
4.4.1 Canonical Methods	56
4.4.2 Single Authoring.....	57
4.4.3 Universal Design.....	57
4.4.4 Interface Management	57
4.4.5 A Unifying Framework.....	58
5. THE WITHINDOWS FRAMEWORK	59
5.1 Image-plane selection.....	60
5.1.1 Image-plane and the Desktop	62
5.1.2 The Advantages of Image-Plane	67
5.1.3 Transitional Configurations	74
5.1.4 Image-Plane Grab.....	83
5.1.5 Immersive Window Management.....	85
5.2 Through-the-Lens Techniques	90
5.2.1 Previous Through-the-Lens Techniques.....	90
5.2.2 The Combination of Image-Plane and TTL Techniques	95
5.2.3 The Role of 6 DOF Input.....	101
5.3 Shifting to an Exocentric Focus	104
5.3.1 Interaction with Egocentric Content	105
5.3.2 Image-Plane and Ray-casting Under Hand	108
5.3.3 Further Reductions in Fatigue.....	109
5.4 Satisfying Guidelines for Productive Applications	110
5.4.1 Usability.....	111
5.4.2 Workflow	112
5.4.3 Application Development	115
6. EXPERIMENTAL VALIDATION OF THE FRAMEWORK.....	121
6.1 Study Design	124
6.1.1 Hypotheses	125
6.1.2 Experimental Design	126
6.1.3 Task Details	127
6.1.4 Experimental Conditions	130
6.2 Study Results.....	133
6.2.1 Underhand Condition	133
6.2.2 At-A-Distance Condition.....	136
6.3 Study Discussion	141
6.3.1 Sub-Hypothesis 1.....	142
6.3.2 Sub-Hypothesis 2.....	142
6.3.3 Sub-Hypothesis 3.....	144
6.3.4 Sub-Hypothesis 4.....	144
6.3.5 Sub-Hypothesis 5.....	146
6.3.6 Sub-Hypothesis 6.....	146
6.4 Conclusions and Future Work	147
7. A PROOF OF CONCEPT APPLICATION	149
7.1 Motivations.....	149
7.1.1 Evaluate the Framework in a Realistic Environment	149
7.1.2 Create a Productive Immersive Application.....	152
7.1.3 Supporting a Community of Users	154
7.2 System Design.....	158
7.2.1 Execution Environment.....	158
7.2.2 Global Functionality and Viewpoint Management	160

TABLE OF CONTENTS(continued)

<u>CHAPTER</u>	<u>PAGE</u>
7.3 System Development.....	164
7.3.1 Simulator Improvements	164
7.3.2 Ygdrasil Architecture Improvements.....	165
7.3.3 Ygdrasil Node Improvements.....	167
7.4 System Evaluation	170
7.4.1 Classroom Evaluation	171
7.4.2 Expert Evaluation	173
8. CONTRIBUTIONS	179
8.1 Ideological Contributions	179
8.2 Theoretical Contributions.....	180
8.3 Empirical and Applied Contributions.....	181
CITED LITERATURE	183
APPENDIX	190
VITA	191

LIST OF TABLES

<u>TABLE</u>	<u>PAGE</u>
I ONE-WAY ANOVA ON TECHNIQUE FOR UNDERHAND TASK QUESTIONNAIRES	134
II ONE-WAY ANOVA ON TECHNIQUE FOR UNDERHAND TASKS	135
III ONE-WAY ANOVA ON TECHNIQUE FOR AT-A-DISTANCE TASK QUESTIONNAIRES.....	137
IV ONE-WAY ANOVA ON TECHNIQUE FOR AT-A-DISTANCE TASKS.....	138

LIST OF FIGURES

<u>FIGURE</u>	<u>PAGE</u>
1. Ray-casting selection	19
2. Image-plane selection	20
3. Selection of distant objects by occlusion is ambiguous with stereo vision.....	62
4. The typical desktop is a special case of image-plane selection.....	63
5. Placing a virtual cursor exclusively in the dominant eye	65
6. A reference cursor can be displayed in the non-dominant eye	67
7. The control-display ratio changes with distance for ray-casting.....	72
8. Approaching a surface using image-plane selection and ray-casting	74
9. A reinforced dominant eye cursor remains linear.....	75
10. Stabilizing the cursor during head motion.	76
11. Virtual mouse stabilization modulation	80
12. Camera based virtual mouse.....	82
13. Image-plane grab.	84
14. Proposed window icon for switching a window between coordinate systems	88
15. Proposed window icon to for switching a window between orientation states.	89
16. The relationship between alternate viewing window and the secondary scene.....	95
17. Proposed window icon to teleport the user to the viewpoint location.....	98
18. Image-plane selection on TTL windows and typical 2½D techniques.....	100
19. Image-plane grab within TTL windows avoids scaling problems	102
20. Proposed window icons for home and lock/unlock functions	104
21. Placing interface elements below the hand reduces fatigue.	106
22. Drag-and-drop task underhand with image-plane selection.....	128
23. Object selection task underhand with image-plane selection.	129
24. Questionnaire presented underhand with image-plane selection	130
25. Testing apparatus for the underhand condition.....	131
26. At-a-distance experimental condition drag-and-drop task with ray-casting.	132

LIST OF FIGURES(continued)

<u>FIGURE</u>	<u>PAGE</u>
27. At-a-distance object selection trial positions with ray-casting.	132
28. Pearsons correlation scatter plot matrix for the object selection task.	134
29. Underhand error and variability by direction.....	136
30. Pearsons correlation scatter plot matrix for the drag-and-drop task	137
31. At-a-distance drag-and-drop trial time by position.....	139
32. At-a-distance drag-and-drop trial error and variability by position.....	140
33. At-a-distance object selection trial time by position.....	141
34. The relationship between user viewpoint and window content.	151
35. Alternate viewing window presented in a desktop environment.....	161
36. Viewing window button to toggle the node hierarchy browser	162
37. Menus can be accessed both within and outside of the viewing window.....	163
38. Source and destination graph subtrees for advanced rendering	170

LIST OF ABBREVIATIONS

2D	Two Dimensional
3D	Three Dimensional
NLS	oNLine System
HMD	Head Mounted Display
CAVE	CAVE Automatic Virtual Environment
WIM	Worlds in Miniature
DOF	Degree of Freedom
C-D	Control-Display
CAD	Computer Aided Design
CRT	Cathode Ray Tube
WIMP	Window Icon Menu Pointer
RFID	Radio Frequency IDentification
GPS	Global Positioning System
VRD	Virtual Retinal Display
LCD	Liquid Crystal Display
UIC	University of Illinois at Chicago
AR	Augmented Reality
NCSA	National Center for Supercomputer Applications
API	Application Programming Interface
CAR	Center Axis Relock
OS	Operating System
PDA	Personal Data Assistant
GUI	Graphical User Interface
TTL	Through the Lens
3DUI	Three Dimensional User Interface
XGA	eXtended Graphics Array
CSCW	Computer Supported Collaborative Work
WYSIWIG	What You See Is What You Get

LIST OF ABBREVIATIONS(continued)

CPU	Central Processing Unit
SXGA	Super eXtended Graphics Array
ANOVA	ANalysis Of VAriance
SEM	Standard Error of Mean
VRML	Virtual Reality Modeling Language
VNC	Virtual Network Computing
IDE	Integrated Development Environment
EVL	Electronic Visualization Laboratory

SUMMARY

The focus of this research is the belief that it is more likely that desktop applications will make a gradual transition into three dimensional space than it is that there will be a paradigm shift into using three dimensional interfaces. The Withindows framework uses image-plane selection and through-the-lens techniques to create a single-authoring framework that optimizes the use of both traditional and immersive applications across transitional states such as desktop stereo, large-format displays and projector-based augmented reality. Because it is a generalization of the typical desktop fixed-viewpoint setup, image-plane selection allows head-tracking, stereoscopy and multiple degree of freedom input devices to be added and removed as needed. Image-plane selection can overcome ambiguity problems discouraging prior stereoscopic use by using a virtual cursor in the dominant eye and, thus, resolve non-linear control-display relationships inherent in some approaches to desktop stereo. When combined with through-the-lens techniques, image-plane selection allows immersive 2½D viewpoint management and object manipulation techniques analogous to those on the desktop that avoid scaling problems and restricted viewpoints. This approach resolves global search and control-display scaling problems inherent in prior through-the-lens implementations and allows extensions for 6 degree of freedom input devices that do not supersede the default interaction method. The problem of fatigue can be significantly reduced by replacing typical immersive interactions out in front of the user with image-plane selection on windows floating below the hand. This thesis describes the Withindows framework, the development of a proof of concept virtual world builder and a user study to validate some of the underlying assumptions it makes. The user study found evidence that a dominant-eye cursor is not well tolerated under all conditions and that image-plane selection and ray-casting produce comparable levels of user fatigue and ease of use when used underhand. The desktop development application was evaluated over a full semester in a classroom environment, and the immersive application was evaluated in three separate projects by expert users.

WITHINDOWS: A UNIFIED FRAMEWORK FOR THE DEVELOPMENT OF DESKTOP AND IMMERSIVE USER INTERFACES

Alex S. Hill, Ph. D.
Department of Computer Science
University of Illinois at Chicago
Chicago, Illinois (2007)

Dissertation Chairperson: Andrew Johnson

The focus of this research is the belief that it is more likely that desktop applications will make a gradual transition into three dimensional space than it is that there will be a paradigm shift into using three dimensional interfaces. The Withindows framework uses image-plane selection and through-the-lens techniques to create a single-authoring framework that optimizes the use of both traditional and immersive applications across transitional states such as desktop stereo, large-format displays and projector-based augmented reality. Because it is a generalization of the typical desktop fixed-viewpoint setup, image-plane selection allows head-tracking, stereoscopy and multiple degree of freedom input devices to be added and removed as needed. Image-plane selection can overcome ambiguity problems discouraging prior stereoscopic use by using a virtual cursor in the dominant eye and, thus, resolve non-linear control-display relationships inherent in some approaches to desktop stereo. When combined with through-the-lens techniques, image-plane selection allows immersive 2½D viewpoint management and object manipulation techniques analogous to those on the desktop that avoid scaling problems and restricted viewpoints. This approach resolves global search and control-display scaling problems inherent in prior through-the-lens implementations and allows extensions for 6 degree of freedom input devices that do not supersede the default interaction method. The problem of fatigue can be significantly reduced by replacing typical immersive interactions out in front of the user with image-plane selection on windows floating below the hand. This thesis describes the Withindows framework, the development of a proof of concept virtual world builder, and a user study to validate some of the underlying assumptions it makes. The user study found evidence that a dominant-eye cursor is not well tolerated under all conditions and that image-plane selection and ray-casting produce comparable levels of user fatigue and ease of use when used underhand. The desktop development application was evaluated over a full semester in a classroom environment, and the immersive application was evaluated in three separate projects by expert users.

1. INTRODUCTION

A significant portion of the developed world knows what a two dimensional user interface is. Many people have used one at one time or another, be it on a computer, a cell phone or at an automated bank teller. Yet, very few people have used a three dimensional computer interface. We are already familiar with one 3D user interface and it is marked by our relationship with the physical world. If we look at the 3D user interface through this lens then it is a relatively easy concept to understand. Such an interface to a computer might allow us to reach out and touch objects as we do in our own reality. This concept of a virtual reality that is not unlike our own has been a strong influence in framing the concept of what a 3D user interface is and can be. However, beyond mimicking the familiar 3D interface we already know, the subject of 3D user interfaces quickly becomes murky. Computer interfaces rarely involve moving objects around in physical space, they are most often vehicles for the manipulation of symbolic data. Symbolic data are the documents we move in a file manager, the numbers we plot and graph. Symbolic data are the commands we apply to our spreadsheets and scientific simulations, from calculating a standard deviation to computing the weather for next week. Our lives have been changed by the modern computer and its 2D user interface. This interface allows us to manipulate symbolic data within its 2D screen and 2D mouse input device. Many people believe that a 3D interface offers an opportunity to improve the efficiency with which we use our computers. Yet, how do we reconcile the acknowledgement that our productivity has been a reflection of the manipulation of symbolic data when a 3D user interface appears more suited to the physical manipulation of objects in space?

1.1 The Ultimate Interface

Much of the excitement around 3D user interfaces was initiated by Ivan Sutherland when he introduced the idea of an ultimate interface between man and computer in 1964. Sutherland proposed that a computer that can recreate and manipulate physical objects would allow us to learn and work in profoundly different ways. He proposed that this interface be used to allow students to “feel” the strong and weak forces characteristic of molecular bonds. This proposal changed that way that people conceived of the human to computer interface. Suddenly, the idea of interacting with a computer in a completely native physical manner appeared compelling to many and gave strong motivation to the fields of realistic computer graphics and computer controlled physical interfaces.

Throughout the late sixties and seventies, Fred Brooks pursued the very idea that Sutherland proposed with his GROPE project. Brooks found that delivering the sense of touch to a human, called haptics, is significantly harder than delivering realistic visuals. Unfortunately, even today we do not have the technology to adequately reproduce the sensation of feeling. Despite the inability to address the most important element in Sutherland's vision, researchers continued to pursue the idea of creating a completely virtual reality mediated by head mounted goggles and data gloves. Ironically, while Sutherland believed the ultimate interface would augment our ability to interface with computers, much of the result has been efforts to recreate or refashion the physical reality we already know. The motivation is understandable because conceptualizing a productive interface with a computer is significantly more difficult than recreating a virtual reality that is similar to our own. Unfortunately, this effort at creating a virtual reality has only increased productivity in limited domains. Forty years later we are not even close to realizing the ultimate interface. And, even if the technology were available to realize such an interface to the computer, few people would have any idea how such an interface should behave.

1.2 **Augmenting Human Ability**

Sutherland was motivated by an earlier thinker, Vannevar Bush, who is credited with inventing the idea of the hyperlink. Bush proposed a machine called the Memex in 1952 that would allow a person to store information, organize it via connections, and then retrieve that information efficiently. The Memex was proposed as a means to augment the human ability to process information. Bush strongly influenced Dick Engelberg, a contemporary of Sutherland, who almost single-handedly invented the modern computer interface. Engelberg is the father of the point-and-click interface and is famous for a demonstration he gave in 1967 known as "The mother of all demos" that included live video conferencing and the first mouse interface. Almost anyone can appreciate the augmentation of human ability that has resulted from this line of research into strictly 2D interfaces.

Useful 3D user interfaces that augment human ability have been much slower to arrive. In the seventies Fisher envisioned using virtual reality goggles and data gloves to create "Data Spaces" consisting of 2D computer screens arranged around the user and a virtual keyboard. However, the technology to interpret the movement of the fingers accurately enough to type in virtual reality was not available then nor is it today. The frustrating limitations of our ability to capture and reproduce physical interaction with the human body led many researchers to focus on the powerful results of generating a first person view of another reality. In contrast to haptic research,

technology to reproduce realistic graphics has grown exponentially to the point where practically every commercial a person watches today is a product of some form of computer graphics.

1.3 The Coming Augmented Reality

The explosion in our ability to render computer graphics is motivating technology to support the display of those graphics. Video games and high definition television are motivating the push towards ever larger display devices. Few people realize that the full resolution of the human eye is less than that of several hundred LCD displays put together. Increasingly, the availability of inexpensive projectors and large format displays is moving content previously confined to a small screen into the physical world around us. Moreover, a number of recent technological advances are making the dream of superimposing any graphical image over the view of the user a real possibility. Eventually the screen as we know it will disappear all together.

This trend towards ubiquitous display of computer graphics is happening in concert with an increasing interconnectedness between ourselves and the computing devices around us. The day when every appliance in our homes is connected and controllable via the internet no longer appears to be a fantasy of science fiction. Increasingly, devices such as our telephones and other remote controls are controlling the devices around us. Automobiles start and unlock themselves from afar; soda machines dispense their wares with the press of a button on our cell phone. These trends promise to completely change the way we interface with computers.

The future promises to truly augment our ability to interface with computers through the ability to overlay computer graphics onto the physical world. The term augmented reality traditionally has referred to the augmentation of our view of reality. However, the ability to augment what we see can radically augment the way we interact with the devices around us. Today an ordinary person can summon a cascade of machines, delivery men, and accountants to dispatch a product from across the world with a push of a button. Yet, this ordinary person must still walk over to their microwave to begin making dinner. The day is coming when the many different interfaces to devices we have to learn today will be abstracted by our own computer generated interface for them. This intersection of remote control and ubiquitous display will allow us to create our own customized interface to the world around us. Suddenly every microwave will operate the same way and be usable from anywhere.

This truly augmenting reality will change the world from one of computing confined within the computer to one that operates completely within the context of the physical world. The power of the computer and the physical world will marry to create completely new opportunities to improve our everyday lives. Information previously tied to maps and databases will suddenly be displayed in the world around us. Congestion and directions will become overlaid onto the road in front of us. Web searches for nearby gas stations will result in pop-ups on the horizon in front of us. These technologies will eventually change the nature of many vocations. Utility repair persons, police, and emergency responders will utilize the overlaying of information onto the physical world to increase their productivity. Consumers, too, will benefit with greater access to information about the products and services within their midst.

1.4 Efficient Augmented Reality Interfaces

With all of this information and computational power in front of us, the need to develop a means to interact with this new 3D interface becomes more urgent. The information in the coming augmented reality will be delivered to us in a manner that is natural to us, first-person perspective. First person perspective is an excellent way to appreciate the effects of new clothes, new windows, and other changes to our physical environment. First person perspective is also a powerful way to interact with objects and devices once we have moved close to them. However, in a future where we will interact with objects remotely, first person can be a severe limitation to our ability to select objects, manipulate objects, and to gain an overview of our surroundings. It was exactly these limitations that motivated the use of maps and other symbolic representations in the first place. Truly powerful interactions with the physical world cannot be restricted to our first person viewpoint. Consider the thought experiment of having magical powers. While it might be easy to move a cup across a table using first person perspective, how would one move that cup onto a table in the adjacent room without the ability to view that room? The ability to magically move from one location to another brings about the same problem. How does one magically appear somewhere else without the ability to see that location and decide exactly where to arrive?

Although magic may not be in our future, something close to it will be. Our future will give us great power over the environment surrounding us and we will need an interface that allows us to exercise that power efficiently. Beyond just overlaying information onto the world around us, the interface of the future will need to help us filter, query, and manipulate that information. The interface of the future will also need to accomplish the manipulation of symbolic information as efficiently as desktop systems do today. The interface of the future will

need to be precise and allow for long term work without fatigue. Interfaces popularized in film and television that have the user waving their arms in the air will never find use for any significant duration of productive work. The interface of the future must allow us to manipulate content at a distance. And, when our perspective on objects in our view is not adequate, we will need the ability to construct more appropriate viewpoints of those objects.

We also need a clear path forward from the display confined interfaces of today to the unconstrained interfaces of tomorrow. These interfaces should support the gradual transition that will take place from desktop displays to full wall displays and eventually to displays that completely surround the user. The interface of the future should be equally usable on a park bench, in front of a crowd or driving in a car. The interface of the future should support the use of contemporary applications in these more flexible environments. Not every application in the future will have a relationship with the physical world around us. Spreadsheets and text documents will likely still exist in fifty years, and we will not tolerate sitting in front of a computer to access them.

A clear path forward includes not only interaction schemes to accommodate the reality of the future but also application support for the transition from desktop to fully immersive interaction. Design methodologies that allow existing applications to migrate off of the desktop and into the physical space around the user without redevelopment are needed to smooth this transition. This same application development scheme should also support the ability to use those applications designed to work on the world around us on the desktop when desired. In short, a clear path forward is one that allows all applications to transition seamlessly not only back and forth from desktop to fully immersive but also through any transitional state in between. This holistic interface scheme also needs a holistic input device. Users need to make this eventual transition from the desktop to immersive interaction without changing the fundamental method they use to interact with the computer. Ideally, this input method has a consistency in all hardware configurations. This canonical input method needs to facilitate the transition from the mouse based input scheme used today. Ideally, both the input method and interface of the future should leverage the existing skill set of the millions upon millions of people already familiar with desktop interfaces.

1.5 The Withindows Framework

The focus of the research described here is an attempt to construct a set of guidelines for the design of coming augmented reality interfaces. These guidelines are informed by observations about the existing state of

3D user interface research and the trends that have dominated prior research. Realities about the advantages and disadvantages of first person interaction with computer generated content are a significant part of these guidelines. Guidelines in themselves do not offer a concrete solution to the problems that future user interfaces pose. A specific framework is proposed that addresses the unique needs of future user interfaces outlined by those guidelines. This framework, called the Withindows framework, supports the development of user interfaces that not only optimize fully immersive interaction but also facilitate the eventual transition from desktop interaction through transitional configurations.

The Withindows framework is derived from a more generalized interpretation of the desktop interface known as occlusion-based or image-plane selection. This generalization allows us to consider the inclusion of future technologies such as stereo display, user head tracking and 3D input in a consistent manner. This interpretation of the desktop metaphor facilitates a smooth transition into scenarios that are similar to desktop interaction in spaces surrounding the user. These spaces can include full wall screens, any surface within view of the user or interaction surfaces floating near the user. Image-plane selection not only allows interaction with traditional desktop content in a natural manner, but it also has advantages over prior interaction techniques used in 3D space. Image-plane selection is more accurate at a distance than the most frequently used technique, ray-casting, and it optimizes interaction on interfaces that are attached to flat surfaces in the environment whether from a distance or up close via simple touch.

Another main ingredient of the Withindows framework addresses the unique nature of interaction with the physical world around us. The framework acknowledges the need to manage alternate viewpoints for tasks such as selecting objects, moving objects and gaining an overview of the environment around the user. A set of techniques, known as through-the-lens techniques, are extended to form the second important part of the framework. These techniques allow for the simultaneous management of alternate viewpoints constrained within windows in an immersive environment. These techniques allow obscured objects to be selected, distant objects to be manipulated efficiently and expand the ability of the user to interpret the data around them. Both image-plane selection and through-the-lens techniques combine to create 3D applications that can also be used without modification on desktop computers. The final part of the Withindows framework involves allowing users to interact with the physical world around them from a more comfortable working position. Because the content surrounding users is available in alternate viewing windows, these windows can be placed in a comfortable position just below

the hand of the user. This allows an interaction style that is analogous to desktop mouse interaction and serves to minimize user fatigue.

The next chapter provides background about the history of 3D user interfaces and points out the themes that continue to influence their development today. Chapter three then describes the nature of the coming augmented reality in detail. This highlights the needs of the user interfaces that will interact with this future reality. This understanding of the coming augmented reality combines with a historical context of 3D user interfaces to motivate the guidelines outlined in chapter four. The full description of the Withindows framework is then presented in chapter five. The discussion in chapter six then turns to an experimental user study that was conducted in an attempt to validate some of the important assumptions about image-plane selection upon which the Withindows framework relies. Finally, a proof of concept application, designed using the Withindows framework, is described in detail in chapter seven. The motivations for choosing the application domain, some development details and some important details about the actual implementation are discussed. The chapter ends with the results of classroom and expert user evaluation of the resulting application. Finally, chapter eight offers concluding remarks about the contributions that this work hopes to make to the field of 3D user interface research and its implications for future work.

2.0 THREE DIMENSIONAL USER INTERFACES

In order to fully appreciate the challenges facing the development of three dimensional user interfaces, some background information is required. This chapter is devoted to giving a historical perspective on research into 3D user interfaces. In addition, this chapter will point out some important themes that have influenced the style and direction of research in the field. The first section will discuss the split between augmenting human ability and the pursuit of creating a virtual reality. The next section discusses the important role of haptics in 3D user interface research and points out how the inability to create haptic output has created a separation between physical and virtual reality. The following section introduces the subject of proprioceptive interaction with 3D content. Here, the differences between exocentric and egocentric viewpoints are highlighted. The discussion then moves to a product of egocentric interaction, manipulation at a distance. This section introduces the concept of corporealism and the inefficiencies inherent in first person manipulation at a distance. The final section deals with the relationship between 2D and 3D interfaces and the differences in productivity and user fatigue that typically distinguish the two.

2.1 Human Computer Interaction

The beginnings of user interface development were motivated by a desire to augment the efficiency with which individuals interact with the computer. While this theme motivated the development of 2D interfaces that would eventually become the personal computer, research into 3D user interfaces has not been as fruitful for the individual. A combination of technical limitations and the unique nature of head tracked visuals pushed efforts towards realizing a virtual reality. While a virtual reality is a worthwhile goal in itself, the dual goals of augmenting human ability and achieving a sense of presence have arguably prevented some research into more practical applications.

2.1.1 Augmenting Human Ability

At the end of the Second World War, Vennavar Bush published an article, entitled “As it may be”, giving what is considered to be the first description of hypermedia (1). Bush described a machine called the Memex that would allow a person to store various media within it and recall that media at a later time. He imagined a person inserting content, such as newspaper clippings, into the machine and then making connections or links between

disparate pieces of information. Although he had no idea how to build such a machine, Bush described how a user might store these linked stories and play them back at a later date. Most importantly, Bush proposed that one might then follow these links into another tangential story and facilitate the exploration of another direction of thought. The proposed Memex device is credited with foretelling a number of future innovations including speech recognition, personal computers and online libraries such as Wikipedia. The ideas proposed within “As it may be” were important because they helped frame technology in the context of augmenting the individual. The idea that technology could augment an individual was groundbreaking because many people perceived technology, especially computers, as having no utility for the average person. Ken Olsen, the founder of Digital Equipment Corporation, is famously quoted in 1977 as having said, “there is no reason for any individual to have a computer in his home”.

Even if the computer was not a personal one, the idea that a user might have an interactive relationship with it was also groundbreaking. Prior to the nineteen sixties, computers ran in a batch mode because of limited computational power and slow peripherals. The computer was configured with a program to run, the program was started, and the output was then printed out and read by the user. The advent of multi-tasking, dividing the computer processor between tasks such as reading keyboard input, displaying to the screen and doing computations, ushered in a completely new understanding of how a user might interact with a computer. In what is considered a seminal moment in human computer interaction, Ivan Sutherland, in 1963, produced an interactive sketching program for his doctoral dissertation called Sketchpad (2). Sutherland revolutionized the concept of what the relationship between man and machine might be. Sketchpad allowed a user to draw shapes, rotate and move them and even copy and paste them by working directly on the computer screen with a light pen.

In the mid-sixties, Doug Engelbart would further revolutionize the role of the human and the computer with the work he did at the Augmentation Research Center at Stanford University. Engelbart invented the computer mouse, and his oNLine System (NLS), demonstrated for an audience in 1968, was the first working example of hypertext shown to the public (3). The NLS system was effectively a word-processor, but it was a powerful one that allowed the user to organize their document in useful ways. Engelbart believed in the importance of collaboration, and his demonstration featured two people making simultaneous changes to a single document linked up via real-time video conferencing. This demonstration, often referred to as “The mother of all demos”, was especially important because it highlighted the potential of a computer to augment the everyday

abilities of individuals. There is little doubt that the work of researchers like Engelbart has had a profound influence on the productivity of the individual. By contrast, the field of 3D user interfaces has arguably had little effect on the augmentation of the individual.

2.1.2 Virtual Reality

Morton Heilig is often cited as a major influence in the field of virtual reality. Heilig decided in the fifties that he wanted to bring theatergoers an experience as close to reality as possible with what he termed “reality machines”. He dubbed his invention the experience theater and sought funding for a theater with 180 degree viewing angles, wind, vibrating seats, smells and anything possible to recreate the experience of actually experiencing another reality. Although influential in motivating thinking about virtual reality, Heilig had difficulty securing funding for his ideas because they were expensive to construct and maintain. He eventually managed to build an arcade style machine called Sensorama in 1962 that allowed the user to experience a motorcycle ride complete with stereo viewing, stereo sound, wind, vibration and even the smell of exhaust.

While the brute force methods of Heilig may not have been particularly sexy, the ideas of Ivan Sutherland on computer synthesized artificial reality ignited significant interest. In his 1965 paper, “The Ultimate Display”, Sutherland proposed a computer interface that could manipulate not only light and sound but also physical matter. He suggested that this interface could radically transform the way we interact with and learn from a computer. Sutherland used an example to illustrate his point. A computer could allow a student to “feel” the forces between molecules of exaggerated size and thereby come to a greater understanding about the weak and strong molecular forces.

Sutherland followed up his proposal with the first demonstration of a head mounted display (HMD) system that allowed a user to view synthesized 3D content superimposed onto the real world (4). This stereo viewing system projected an image onto half-silvered mirrors in front of the user that changed in accordance with the position and orientation of the head. The system was nicknamed the “Sword of Damocles” because it would certainly kill you if the bulky device were to fall on you. Regardless of the clumsy appearance, the one-two punch of “The Ultimate Display” and the first working HMD created a completely unique train of thought in the computer science world; if we can now synthesize the visual sense then we may be on the verge of realizing a completely

artificial reality. It wouldn't be until the nineteen eighties that the phrase virtual reality would be coined by Jaron Lanier, but the idea of creating a synthesized reality is still attributed to Sutherland.

2.1.3 The Emergence of Two Themes

The very ability to track the motion of the head and deliver first person perspective computer graphics allowed virtual reality technology to deliver a compelling sense of immersion. The position and orientation of the human head are only a combined 6 variables but together they account for one of the most personal and uniquely human sensory perceptions of the physical world. This sense of first person visual immersion is arguably the most salient feature of any virtual reality device and the technology of the time was capable of delivering it. The systems for tracking the position of the head relative to the surroundings were crude. But, covering up the real world alleviated a significant part of the problem and left the most important part of the experience intact. This led to a significant amount of research that involved no display of the physical world around the user. Ironically, the HMD first proposed by Sutherland was an augmented display and put forth as a means of augmenting the information available to the user. One of the two applications he demonstrated included a superimposition of compass directions onto the walls of the room in which the user was located.

In the early seventies, Scott Fisher proposed that virtual reality technology could be used to place users in a completely artificial "Data Space" composed of multiple artificially rendered display screens and a virtual keyboard (5). Ideas like those of Fisher and Sutherland did not evolve into practical ways for first person 3D user interfaces to augment the human productivity. Fisher and his contemporaries did not have the ability to track the motions of the fingers with the crude tracking technology of the day. This forced researchers to focus on gross motions and their use of the technology mirrored these technological limitations. Grabbing objects and moving them in a physically intuitive manner offered more immediate satisfaction than trying to manipulate the very symbolic interfaces of 2D desktop computers. The HMDs Fisher used also did not have the resolution to display readable text at anything but very large sizes. These technical limitations motivated researchers to place the user within content instead of placing content in front of the user as was the norm. Instead of the fine grained representation of information common on the desktop and physical media, the technology provoked a focus on the kinds of content that worked well with visual immersion. There has been a natural tendency towards recreating reality because the success of such an effort is easily qualified. We have a good idea what the

necessary qualities of a successful virtual reality are. On the contrary, envisioning exactly what type of productive work one would do in 3D artificial space has been more difficult to conceive of.

As researchers worked towards increasing the resolution and accuracy of the technology they did not discourage the idea that virtual reality technology could create a synthetic reality. The increasingly high technical requirements of the medium and the all-or-nothing notion of a completely synthetic virtual reality culminated in the nineties with the invention of the Cave Automated Virtual Environment (CAVE). The CAVE is a projector based environment that makes experiencing virtual reality a significantly less intimidating and isolating experience (6). Multiple users can view a CAVE virtual world from within small enclosure formed by the rear projected display walls wearing nothing more than special glasses. The CAVE delivered a significant increase in effective resolution and graphic fidelity with an associated price tag of over 1 million dollars. This technology coincided with an increase in public interest in virtual reality and significantly contributed to the success of a journal devoted to medium. Fittingly the Journal of Teleoperators and Virtual Environments was given a name, Presence, that describes the strong influence that creating compelling immersive experiences has had on the research field.

The technical limitations of 3D user interfaces may have lessened over the years, but the early influence of Sutherland's Ultimate Display and the draw of first person perspective have contributed to some ambiguous goals in the field. Many have felt that the end-game of the technology is realism, and, as a result, the focus on augmenting human ability has not seen the same focus. Today the technology is now allowing research into augmented reality to supplant virtual reality and return the focus to that of empowering the individual. However, the lack of a clear separation between research devoted to augmenting the human and that of creating a virtual reality has arguably led to some isolationism. The high technical requirements and high fidelity required to impart a sense of presence has likely caused some more practical uses for the underlying technology to receive less attention.

2.2 The Role of Haptics

The sixties were a time of a paradigm shift in the way that we perceive of machines and the work they do. The age of mechanisms was ending and the age of silicon was beginning. Suddenly the work done in machines with no moving parts was beginning to eclipse the power and mystic of the mechanical world. This shift is an important influence in computer science and the development of virtual reality. For more than a decade after

Sutherland first proposed it, Fred Brooks worked to coax computers into delivering synthesized haptic content to the user. Brooks used the ability to feel the interactions between molecules as the main motivation for his GROPE project. Brooks and many other researchers found that delivering haptics was significantly more difficult than delivering synthesized visuals. Today the reasons appear clear in hindsight. The visual organ is in a fixed location, gets its input from a relatively fixed location in front of the head and is perfectly happy getting input at only 24 frames a second. By contrast, the sensory organ of the skin is the largest sensory organ we have, takes on a multitude of shapes and locations and is sensitive to input frequencies in the thousands of Hertz.

If these challenges were not enough, a significant problem with delivering haptics in the mid to late twentieth century was that it was using technology from the old paradigm. No solid-state machine could deliver haptics, and as a result the same bulky, expensive to build, and expensive to maintain machines of the past were employed to realize the effort. As difficult as it was to realize the physical part of Sutherland's dream, the visual and auditory part was much more appropriate for the new paradigm of silicone transistors. Not only could visual imagery be generated and delivered completely by computer, it could also be done repeatedly and with minimal maintenance. The difficulties of delivering on the promise of haptic interaction created a disconnect between what synthesized reality can deliver and the physical reality it is supposed to mirror. This separation forced compromises between maintaining a sense of immersion and allowing immersed users to accomplish normal tasks with some efficiency. In the context of developing productive 3D applications, the inability of technology to bridge the gap between virtual and physical reality raises questions about the utility of continuing to use interface elements developed with immersion as a goal.

2.2.1 Interaction with the Hands

Haptics are a critical part of almost every physical task we undertake. Writing and typing on a keyboard both require the physical touch of the surface under our fingers. The process of reaching to grab an object relies on feeling the object to accurately gauge the distance and subsequent force required to retrieve it. Even setting an object down relies on subtle cues about the interaction between object and surface. The pressing of buttons, turning of dials and activation of levers all rely on the constraints provided by haptic feedback. Without haptic feedback many ordinary activities in the physical world become more difficult to accomplish. Unfortunately, accurately tracking the motions of individual fingers and delivering haptic sensation to them has been difficult to accomplish without bulky and expensive mechanical equipment. Even in such an environment, the computations

required to simulate a natural interaction between hand and object are expensive to perform. As a result, many researchers found themselves resorting to the press of a physical button to indicate activities such as pressing a virtual button, grabbing an object or opening a door.

Even tasks such as reaching to grab a virtual object are made more difficult by the lack of haptic feedback. Visual and auditory cues such as shadows and intersections between the hand and the virtual object become the main cues for gauging distance. In the presence of visual resolutions and fidelity that are considerably lower than in real life, the task of estimating distance becomes a challenge under even the best circumstances. Realizing these problems, researchers began incorporating what are called pseudo-haptic interactions into their work. Doug Bowman at Georgia Tech used a flat surface that was registered and tracked within the virtual environment for user interaction with his work on designing animal habitats in virtual reality (7). Users used a tracked stylus to move objects around on the virtual notepad and viewed the results within the surrounding environment. These physical surfaces have come to be known as physical props and researchers have incorporated them into rear projected environments like the CAVE by using clear Plexiglass panels (8). Later, Robert Lindeman found empirical evidence of the benefits of pseudo-haptic interaction when he compared interactions on surfaces fixed in the virtual environment and held in the non-dominant hand (9). In either case, users were able to touch virtual buttons on a surface much more efficiently when they had a physical prop providing haptic feedback.

2.2.2 Travel in Virtual Space

Although tracking the user viewpoint delivered a compelling sense of immersion with objects close to the user, it quickly became apparent that another technology, virtual travel, could not only increase the utility of 3D spaces but also increase the sense of immersion within them. Both a user moving their head and a user moving through a space enjoy the effect of visual content changing due to perspective distortion. This change is known as motion parallax and it is perhaps the most important aspect of visual immersion in a first person virtual space. Virtual 3D spaces have the advent that they are potentially unlimited in space and it became desirable to exploit this space. Not only does allowing the user to explore a space increase immersion, but it also creates the conditions for exploring 3D objects and spaces in ways that may not be possible in the physical world. The user can be reduced to the size of an atom to explore a molecule or the user can be scaled to the size of a building to

explore a city. Thus, travel within virtual reality promises to augment the user by allowing them to learn and explore information in ways that have not been previously possible.

Behind the natural movement of the head and body, walking is one of the most natural methods for moving through the space we inhabit. Unfortunately, technological limitations and the different configurations of physical and virtual space prevent users from physically walking in virtual spaces. As a technical challenge, delivering the ability to walk artificially should be one of the first places to look for success. Walking is a relatively simple haptic activity when compared to grabbing an object with the hand. Walking only involves gross sensory interaction between the foot, gravity and a flat surface in a predictable manner. A big challenge in delivering haptic feedback is delivering the necessary force to the point of interaction. The floor, by its nature, is physically grounded and therefore works well with gravity to deliver the majority of the sensor perception the feet need.

Unfortunately, artificial walking is actually very difficult to deliver. Even if a device that can move a flat surface in all directions can be developed, synthetic walking presents significant challenges in determining how to move that surface in anticipation of the next footfall. The last two decades have seen a number of highly complex and expensive solutions to the problem. Rudy Darken is well known for his Omni-Directional Treadmill developed in the late nineties for the military (10). The military is an end user of virtual reality technology that seeks accurate simulation of reality. The armed forces are also one of the only consumers of virtual reality technology that could afford to develop and maintain the devices necessary to accomplish synthesized travel. Treadmills and other virtual travel technology are examples of technology from the old paradigm of the mechanical. Their many bulky moving parts, motors and electronics are notoriously expensive to build and require constant maintenance and calibration.

2.2.3 Virtual Travel Techniques

The natural result of a lack of haptic walking systems has been the development of techniques that are known as virtual travel techniques. A number of techniques have been developed to move the user through the virtual space artificially. The Point-and-Go technique which moves the user in the direction the hand is pointing has proven the most popular technique. Gaze directed versions of the same technique have also been attempted with mostly limited success (11). Keeping the head pointed in the direction of travel restricts the ability to view the

surroundings as a user moves through them. The Pulling-Rope or Grabbing-at-Air technique encourages that the user reach out in the desired direction, indicate the intent to grab and pull their arms back towards their body (12).

These techniques tend to move the user smoothly through the environment. At times this continuous movement is augmented with teleportation techniques. Studies have found, however, that teleportation is disorienting to users and diminishes the sense of immersion (13). This knowledge resulted in time-based techniques that automatically move the user smoothly between destinations over a short period of time. Another response to teleportation disorientation is techniques such as Worlds-in-Miniature (WIM) by Pausch (14). This technique displays a miniature model of the surrounding environment floating in front of the user. The user grabs this model with the hand and moves the model over their head to form a new viewpoint that corresponds with that of the desired teleportation location. With this method, there is significantly less disorientation when the travel event occurs because the user is already viewing a version of the new viewpoint. The main drawbacks of the WIM technique are twofold. First, developing a miniature model of the surrounding environment requires a separate development task and may not be appropriate for some environments that are large or sparsely populated with content. Second, the presentation of a miniature model floating in front of the user can create visual confusion between it and the surrounding scene.

2.2.4 A Separation between Virtual and Physical Reality

The inability to deliver accurate haptic interaction between the virtual world and the hands and feet of the user forces interface designers into compromises. These compromises create a tension between maintaining immersion and supporting efficient interaction with 3D virtual spaces. In the case of grabbing objects, pressing a button is likely to be more efficient than attempts to mimic the real world task. In the case of virtual travel, physically intuitive methods increase such qualities as immersion at the expense of speed, accuracy and efficiency with which we move through that space. Physically intuitive methods do increase the spatial awareness and wayfinding ability of users. Yet, these are skills forced onto us by the constraints of the physical world and are not of obvious utility in a productive application. Often the most important goal associated with spatially organized data is the ability to formulate a mental model of that structure and search globally for content. Travel techniques that preserve immersion are often inefficient for global search tasks. More efficient techniques such as teleportation do reduce the efficiency with which users become spatially aware in a virtual space, but there are

also complementary techniques such as maps and overviews that can serve to compensate for this loss of immersion when searching for content in that space (15).

User studies on virtual travel often conflate wayfinding ability and spatial awareness with an ability to perform a global search of the environment. Bowman used spatial awareness, the ability to point blindfolded at a virtual object, as a metric in evaluating travel techniques (16). Yet, spatial awareness is actually a better indication of immersion in a space than it is a measure of the ability to do a global search. Darken evaluated the ability of users to execute global searches in virtual environments, but only evaluated a single point-and-go travel technique (17). These studies are useful if a space must be navigated in a first person manner but they do not answer the question of why one would do so to efficiently find anything. A true evaluation of the efficiency of a virtual space would compare global search techniques with other non-immersive tools. In fact, some studies have found evidence that desktop exploration is just as effective at imparting configurational knowledge about 3D space as immersion (18).

The reality is that immersion is greatest when users use physical interactions such as natural walking motions to travel virtual space (19). Yet, true immersion is not a practical goal for 3D applications unless they are attempting to simulate reality. The development of productive 3D applications would benefit from standardized or canonical methods of interacting with content and managing user viewpoint within virtual spaces. Unfortunately, a desire to maintain immersion often casts contemporary 3D interaction techniques as compromises on the path to a truly physically intuitive interface to virtual spaces. Metaphors such as first person perspective and virtual hand interaction are certainly applicable to the simulation of reality but should not be taken as granted if they become an impediment to the development of efficient 3D interfaces.

2.3 Proprioceptive Interaction

Proprioception is the innate ability to interact with objects efficiently when they are near to or attached to the body. It is reinforced by a spatial awareness of our body and a familiar visual feedback when we move it. We can not utilize proprioception when working on objects away from the body. This dichotomy between near and far interaction can be expressed in terms of exocentric and egocentric viewpoints. An egocentric view is one looking outward from the self. Moving through a space that surrounds us is an egocentric interaction with that space.

Egocentric viewpoints are the way we are accustomed to viewing the majority of the world and the most powerful method of delivering a sense of immersion to an immersed user.

An exocentric view is one that is from the outside or above. Looking at an object held in the hand is an exocentric interaction with that object. While exocentric inspection and manipulation of content is very well supported by first person immersive technology, interaction at a distance has also been influenced by attempts to maintain physically intuitive methods that are commonplace in a physical reality. Unfortunately, the advantages of first person perspective in an exocentric condition become liabilities in an egocentric condition. Immersive user interfaces would benefit from a clear appreciation of this dichotomy and an acknowledgement of the inherent inefficiencies in contemporary efforts to select and manipulate objects at a distance.

2.3.1 Selecting and Manipulating Objects

Selecting objects within reach of the user is a trivial task in both physical and virtual realities. Despite the lack of haptic interaction, selecting objects around the body by reaching for them is a relatively efficient task due to the proprioceptive feedback provided by the combination of user viewpoint and the motion of the virtual hand. Like the icons on a traditional interface, moving the hand within the volume of an object indicates the intention to select it. Virtual hand manipulation is also an effective way to manipulate objects when they are close enough and small enough to create an exocentric condition. Typically only the position and orientation of the hand itself is measured by tracking equipment. Yet, this combination of what is called 6 degree of freedom (DOF) hand tracking and 6 DOF head tracking still creates a compelling sense of interactivity with even medium sized objects such as a chair. The tracked motion of the hand and ability to manipulate user viewpoint combine to create parallax motion cues and a strong sense of proprioceptive feedback.

Virtual hand manipulation has obvious limitations when attempting to reach objects that are out of reach. One solution to the physical travel problem has been to eliminate the need to travel in order to select or manipulate objects. The first technique for extending the reach of the user was arguably the pointing based technique introduced in the seminal MIT paper “Put that there” by Bolt (20). This technique, known today as ray-casting, selects the first object along a ray extended from the hand into the environment (Figure 1). Ray-casting allows any object within view of the user to be selected and is usually accompanied by a visible ray extending from the hand in 3D environments. When objects are near the user, the technique benefits from proprioception

because users are already familiar with the hand-eye coordination of holding and moving similar objects in their hands. When all objects are sufficiently far from the user, the technique effectively becomes a 2D selection tool on a sphere surrounding the user (21).

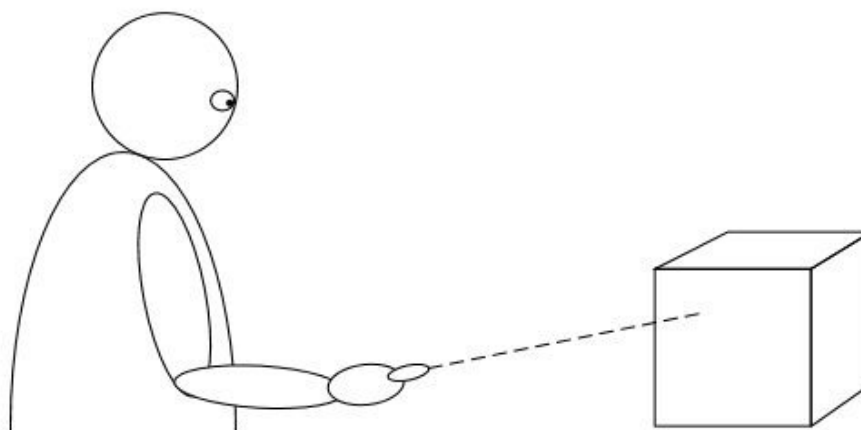


Figure 1. Ray-casting selects the first object intersected by a ray projected from the hand of the user.

Ray-casting does have some well documented problems. The first problem with ray-casting is known as the lever problem. Although objects can be selected at a distance, once they are attached to the end of the ray, they can only be manipulated in an arc around the hand and are, thus, difficult to move closer or farther away. A second problem with ray-casting results from the heavy reliance on the angular input of the hand and is known as the arc angle problem. As objects are selected farther away from the user, the same angular changes in orientation result in increasingly large movements of the object. The arc angle problem creates difficulties in selecting and moving objects at a distance because the resolution of the tracking device and unsteadiness of the hand begin to dominate the activity. Another problem with ray-casting that has not been covered in the literature is the alignment problem that results from the fall-off of proprioceptive feedback as distance increases. As distance increases, only relatively large movements of the ray allow the visual and physical feedback produced to create proprioceptive feedback. Users making small manipulations at a distance have a tendency to bring their hand and ray into alignment with their view. This user technique makes the accuracy of these smaller object movements

more dependent on the translational relationship between hand and the target. As a result, as the hand moves farther out of alignment with the line of sight, fine movements at a distance become less efficient.

Another object selection technique is known as gaze-based selection. Because tracking systems typically only track the position of the head, this technique requires that the user stare in the direction of the desired object. Gaze based techniques have not proven popular because holding the head fixed on an object is not a natural task. In 1997, Pierce introduced an occlusion-based method called image-plane selection (22). Image-plane selection operates in a similar manner to ray-casting, but the ray is always formed down the line of sight and through a position manipulated by the finger-tip (Figure 2). Image-plane selection has not proven popular in the past because it forces the user to raise their arms significantly more than ray-casting and therefore results in more user fatigue.

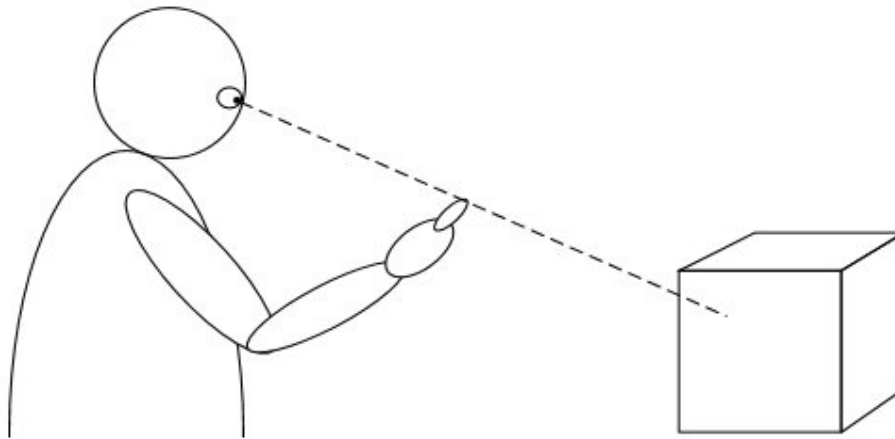


Figure 2. Image-plane selection casts a ray from the eye through a point controlled by the hand.

2.3.2 The Separation between Egocentric and Exocentric Interaction

Exocentric viewing is the optimal way that humans come to fully understand the spatial arrangement of content. Objects small enough to be held are done so at waist level while larger objects are preferably kept below the level of the head. These arrangements allow one to easily move their head around the object to create a parallax view of the object. This parallax view, whether produced by head motion or manipulation of the object, is

what allows the spatial nature of the object to be appreciated. The waist level is also an efficient location for working with the hands. When resting on a desk, chair or even folded legs, the arms have the best opportunity to work for long periods. In contrast, egocentric viewing of content is the optimal way that humans appreciate what it is like to actually be immersed inside spatial content. There is much insight to be gained from egocentric viewing of space. Exploring space in a first person manner can give significant insight into how that space would accommodate a human.

These distinctions between egocentric and exocentric viewpoints are important because they relate strongly to the difference between immersion and forming an efficient mental model of a space. The majority of the utility derived from egocentric viewpoints comes not from a unique use of 3D technology but from the ability to simulate reality. A person egocentrically immersed in a 3D space is unlikely to have any greater understanding or efficiency within that space than what is already afforded by the physical world. First person perspective is not an optimal way to appreciate the spatial arrangement of objects. First person perspective creates occlusions and distortions at a distance that cannot be resolved without moving. As objects move away from us, our ability to either move them or change our viewpoint decreases with distance. Whether it be in the physical world or a virtual world, moving to another location to change viewpoints becomes more troublesome the farther away an object moves. First person perspective projection also makes objects increasingly difficult to see clearly at a distance and it also compresses depth, making the estimation of the distance to objects and between them difficult.

The shortcomings of first person egocentric perspective are the very reasons that tools such as tabulations, graphs, charts, maps and elevations were developed. These tools arrange and present information from an undistorted orthogonal viewpoint and allow a person exocentric viewpoints that may not be otherwise physically possible. Parallel perspective representations of objects allow distance relationships to be more easily understood. Several approaches have been used to try and compensate for the deficiencies of first person perspective. Johnson and Leigh allowed users to assume either a Mortal or Deity role in their CALVIN architectural design program (23). Deity users were scaled up to give them an exocentric viewpoint of an architectural design during construction. Pierce exaggerated the size of landmarks in the distance and eliminated the natural occlusion of other objects in between to make landmarks easier to find (24). Other researchers such as Viega have placed virtual lenses in the hand of the user to allow seeing through occlusions or a zoomed viewpoint (25). A rather poignant example of the limitations of first person perspective involves a recent interface

that allows the user to literally grab the horizon and tilt the surrounding horizon upwards in order to gain perspective on the spatial arrangement of landmarks beyond the user (26).

First person perspective can create inefficiencies that exocentric viewpoints can alleviate. The examples above highlight the lengths to which researchers have gone to avoid incorporating alternate perspectives into spatial environments. Other researchers have exploited the differences between egocentric and exocentric by finding ways to introduce alternate viewpoints directly into virtual environments. Darken was instrumental in introducing maps that either float in front of the user or appear on the floor below them (13). Fukatsu introduced a superimposed birds-eye view that allowed users to maintain an overview while traveling in a virtual environment (27). Paush and Stoakley used their WIM technique to allow users to select and manipulate objects at a distance by manipulating them exocentrically within the hand held miniature. Mine was influential in introducing orbital viewing into virtual environments. Instead of forcing the user to travel around an object for inspection, he mapped head motion directly onto a viewpoint that moves around the object (28). These examples are practical ways that first person perspective can either be augmented or even completely abandoned in order to improve the efficiency with which a user can gain spatial information. Developers of user interfaces need to remain aware of the strengths and weaknesses of both egocentric and exocentric viewpoints when developing 3D applications.

2.4 Manipulation at a Distance

A number of techniques have been developed to allow a user to manipulate objects from an egocentric viewpoint. Most of these methods are inefficient because the user cannot escape the limitations of both first person viewing and first person manipulation from a distance. Methods that avoid the limitations of first person perspective, however, often impose new constraints on the relationship between input and the resulting object motion.

The limitations of first person have motivated desktop and tablet based tools that are significantly more efficient because they remain isomorphic and utilize more appropriate alternate perspectives. The reluctance to avoid the inefficiencies associated with first person manipulation at a distance raises the possibility that developers have been prejudiced in favor of physically orientated interactions in 3D user interfaces.

2.4.1 Distance Manipulation Techniques

The first technique designed to avoid the lever problem of ray-casting belongs to a class known as arm-extension techniques. The Go-Go technique, introduced by Poupyrev, simply extends the reach of the arm by applying a non-linear mapping between hand distance and virtual hand distance (29). The mapping to the virtual hand remains linear until roughly two-thirds of the user arm length at which point the hand moves away from the user based on a squared function. Once an object has been selected and grabbed, normal rotations of the hand allow it to be positioned at any point in a range around the user. An obvious shortcoming of the Go-Go technique is the inability to adjust the scaling between arm motion and object motion. The scaling problem makes the Go-Go technique useful for gross movements to and from distant locations but a poor method for making fine adjustments at a distance.

Bowman developed the HOMER technique to address the lever problem associated with ray-casting (30). The HOMER selection and manipulation technique combines ray-casting and virtual hand manipulation at the location of the object. Once an object is selected with the standard ray-casting technique, the virtual hand moves directly to the object to establish a scaling between physical and virtual hand distance. Movement half of the distance towards the user results in the virtual hand and object moving half the distance to the user. With some practice this technique allows a user to move an object freely within a large range around their body. Although the HOMER technique gives users the ability to reset what is known as the control-display (C-D) relationship each time an object is grabbed, it does not solve the scaling problem. The scaling established for distant objects only allows coarse adjustments to their position, and the scaling for nearby objects prevents them from being moved a significant distance away from the user.

Mine offered an interesting solution to the problem with his Scaled-World-Grab technique (31). Upon selecting an object, instead of scaling only the hand input relative to the object position, Mine scaled the entire world. Selecting an object twice the distance from the hand scales the world around the user down so that the object appears at the same depth as the virtual hand. This technique works best with an object selection technique such as image-plane selection because the selected object appears to move directly into the hand of the user without changing apparent size. The significant advantage of re-sizing the world is that the range the user can move their viewpoint around the object is scaled relative to the object distance. This ability to move the head and appreciate the world from alternate viewpoints is a benefit of scaled-world techniques. Another benefit

of the scaled world technique is that the control-display ratio maps hand motion directly onto object motion. This type of control-display relationship is known as an isomorphic relationship. The drawbacks of the technique are the same as those of the HOMER technique. Large and small scaling factors limit the granularity of distant object movements and the range of nearby objects respectively.

Pierce solved the scaling problem by utilizing Mine's scaling and introducing a reference object into the manipulation task (32). His technique, Voodoo Dolls, uses image-plane selection to place a scaled copy of the object in the hand of the user along with a reference object selected with the non-dominant hand. The primary object can then be positioned relative to the reference object until it is released. This scheme allows the user to determine the desired scaling between hand motion and resulting object motion through the choice of a reference object either distant from or near to the object. The interaction with both objects remains isomorphic and can be done exocentrically near the waist of the user. One drawback of the Voodoo Dolls method is that, like the WIM technique, users sometimes become confused about the separation between real objects and their smaller copies.

2.4.2 Corporealism

The accurate positioning of objects in 3D space requires the ability to manage user viewpoint during the process. With the exception of the Voodoo Dolls technique, all of the above techniques introduce inefficiencies because the user viewpoint is restricted by first person egocentric perspective. The restricted viewpoints of these techniques are not a significant problem when objects are being moved in an arc around the user. However, the combination of non-linear depth control-display ratios and the depth compression of first person perspective make most practical manipulation exercises an inefficient process.

The limitations of first person egocentric perspective have motivated tools such as those found on modern 3D workstations. Modeling and computer aided design (CAD) programs create viewpoints that remove perspective distortion and allow the user to view the manipulation task from an optimal viewpoint. These desktop tools usually do not suffer from scaling problems because the effective control-display relationship between mouse and content is scaled by the zoom factor of the viewing window. The same limitations of egocentric manipulation at a distance motivated researchers like Bowman to move manipulation tasks onto tablets held by

the user. The work by Bowman and others acknowledges that a top-down view is a more appropriate viewpoint from which to manipulate the position of objects in the horizontal plane.

In light of the limitations involved, it is somewhat surprising that significant efforts have gone into first person manipulation at a distance. The effort to preserve immersion is not a convincing explanation for these efforts because they are built upon compromised notions of reality. The techniques described above routinely dismember the hand from the body, shrink and grow the surrounding environment or involve manipulating surrogates of objects outside their environmental context. A more likely explanation for these efforts is a prejudice that there are efficiency gains to be found by re-liberating information processing from the confines of the desktop computer. This prejudice is grounded in a form of materialism, or corporealism, which assumes that the physical world and our methods of interacting with it have a natural efficiency. The problem with corporealism is that it conflates the ease with which we interact with the physical world with a perceived ease of interacting with more abstract information.

The very concept of a physical interface to the computer first put forward by Sutherland is rooted in a belief that such an interface would yield significant benefits. While humans do work with great efficiency with physical objects, our interactions rely on a very high fidelity haptic interface. Human computer interfacing expert Donald Norman is well known for his admonitions about the positive role of constraints in efficient user interfaces (33). Without the ability to realize a physical interface and the constraints provided therein, the utility of pursuing a physical metaphor becomes suspect. Corporealism is the motivation behind one of the most frequent subjects of 3D user interface research, 3D modeling. Many efforts have been pursued in the belief that giving the user an input device with 6 degrees of freedom will allow them a greater level of expression. However, what researchers usually find is that providing position and orientation without haptics merely confounds the ability of the user to control individual variables with any precision.

In the realm of information presentation, first person perspective has no natural advantage for the efficient organization of information. Yet, physical materialism has motivated the development of 3D file managers and numerous other attempts to map information into first person Cartesian space (34). There are undoubtedly situations where first person interactions with data offer advantages, but only limited domains provide advantages significant enough to warrant a move from desktop 3D representations to full immersion. A further problem with

physical materialism is that it is usually at odds with the notion of reducing physical exertion. Modern computer applications would not be as productive if it were not for the relatively low fatigue associated with mouse driven interfaces. While moving around and reaching for objects is natural behavior in the physical world, encouraging such behavior in a computer mediated interface requires significant benefits to justify the additional physical effort.

2.5 Relationship to Desktop Systems

The desktop computer interface succeeded in creating an environment within which productive applications could be developed and used. By contrast, three dimensional user interfaces are well known for their low usability, low productivity and high fatigue. As a result, it is natural to consider the relationship between 2D and 3D user interfaces in the context of improving the productivity and usability of 3D interfaces.

Like desktop interfaces, system control functions, such as those typically found in application menus, are a necessity for 3D applications. And, as with desktop systems, discrete and accurate input methods are a necessity. Unfortunately, to date, typical desktop interaction schemes such as menus, sliders and icons have not translated efficiently into 3D space. Traditional methods such as virtual hand, ray-casting, and image-plane selection each have their own drawbacks that appear to have prevented the development of a single canonical input scheme. In fact, the prevailing wisdom is that no single 3D input method is appropriate under all conditions and that user interface decisions must be made on a case by case basis (35).

This different context within which 3D user interface development resides is in sharp contrast to desktop systems which have benefited from a stability of hardware and software. The significantly more brittle hardware environment for immersive display systems and a lack of a true killer application to drive commercial use has created an environment where basic usability concerns such as fatigue and productivity have often taken a back seat to experimentation. Although applications that simulate reality have immediate utility, those that truly augment the productivity of the individual have been hard to come by. The result today is a climate where the once highly hyped area of 3D user interfaces now must concentrate on finding practical applications that justify its broader use.

2.5.1 System Control

Beyond the more physically based tasks of virtual travel and object manipulation, the need frequently arises to initiate system level functions within 3D applications that are not strict simulations of reality. The desire to initiate functions in an accurate and timely manner tends to push interaction schemes towards the type of point-and-click elements such as menus and icons used on desktop systems. However, limited display resolution and the design space of first person perspective present difficulties for incorporating efficient desktop interface elements.

A desire to simultaneously increase the resolution dedicated to virtual environments and their immediate usability spurred a movement towards physically intuitive interfaces. Unfortunately, methods that eschew traditional system control do not scale well up to complex interfaces of real-world applications and may not even be appropriate in the context of current technological limitations.

2.5.1.1 Input Devices

The two dimensional space of desktop displays has contributed to the relatively constrained input parameters of X and Y position. The much less constrained design space of 3D input has led to a number of different input methods and input devices. In pursuit of a more natural interface, the Data Glove was first introduced in 1987 to try and capture the pose of the human hand. In addition to a lack of haptics and computational difficulties in modeling interactions with surfaces, the main problem with this type of input is that the plethora of information provided by such an input device does not immediately aid efforts to accurately control a computer interface. Actions such as stroking or striking an object might be well modeled by an accurate representation of the hand, but discrete inputs and accurate positioning are not the strengths of such a device. The need to create discrete inputs such as those provided by physical buttons motivated research into interpreting the specific gestures made with the hand. Again, computational complexity has kept these efforts from finding broad usage until more recently. However, regardless of when gesture based hand recognition will find mainstream usage, the end result is that a complex multidimensional input device such as the human hand ends up being reduced to discrete inputs.

The lesson of whole hand input also applies to computer vision based interfaces. Just like the Data Glove, images of the user must be converted into useful information for the computer. Because video images of

the user must be turned into discrete information such as head position and hand position, they offer few unique advantages over other less computationally intensive tracking technologies. Still, there is much useful information available by processing images of the user, but that information, such as facial expression, body posture and the location of eye gaze, is often more applicable to mirroring the physical embodiment of the user, known as the avatar, than it is to controlling a computer interface. There are uses for technologies that capture facial expression and eye gaze on the desktop. The computer can respond to changes in user mood or modify the screen content in response to what part of it appears interesting to the user. However, it is hard to argue that these input methods have a significantly different utility in 3D user interfaces.

Another input method that is often brought up in connection with 3D user interfaces is voice input. Like vision based input, voice input is unlikely to have significantly more utility to a 3D user interface than it has to a desktop interface. Voice input is an attractive input means because it is challenging to provide a fully immersed user with a keyboard. Despite its broad availability, voice input has not supplanted the keyboard or mouse on desktop systems. Voice input is a slow input method. In contrast to a mouse click, a voice command must be formed from memory, takes time to utter, must be heard completely by the system, and runs the risk of being misunderstood. These shortcomings are likely to keep voice input from supplanting the mouse and keyboard for some time. While voice is likely to find more use in a 3D user interface, it is likely to be for a small set of frequently used commands or undirected text dictation. Regardless of the level of use in virtual environments, voice is unlikely to supplant the need for accurate, timely and discrete interaction with a user interface.

2.5.1.2 Interaction Schemes

Beyond physically based interactions such as manipulating objects, any useful 3D application is likely to require a number of system level functions. When adding system control to an immersive application, it is natural to turn to constructs such as menus and icons that have been used successfully on the desktop. When used in first person immersive applications, traditional menus and icons suffer from readability problems not only because of low display resolution but also because of perspective projection distortions. These distortions, known as affine distortions, occur when a flat surface is presented at an oblique angle to the user viewpoint. The typical response to this problem has been to display system control elements directly on the screen of a HMD display device. This configuration maximizes the readability of text and ensures that the user does not lose track of the information. In

CAVE systems, this same strategy translates into placing content untransformed directly on one of the display walls surrounding the user.

One concern with display fixed menus and icons is the interaction scheme used to interact with them. Because the content is not a part of the 3D environment surrounding the user, interaction schemes such as ray-casting that rely on proprioception may not work well with them. Bowman and Wingrave have reported that users prefer an occlusion or image-plane based scheme when selecting view-fixed interface elements (36). However, this strategy can potentially cause confusion unless the same interaction scheme is also used for more general object selections in the surrounding environment. Other common approaches to display-fixed menus have been voice recognition or simple button presses to sequence through menu items. Unfortunately, keeping interaction items such as menus and icons in view of the user also takes up valuable display real-estate and can interfere with the ability of the user to view the surrounding content. In fact, Billinghurst found that users not only preferred information screens that float with their body over view-fixed content but also were able to perform information searches faster when content was body-fixed (37).

Interface elements that are allowed to remain fixed in the world or with respect to the body pose other challenges for interaction. The proper determination of selection areas around selection elements in a 2D interface is a trivial task. However, when using a virtual hand selection method, choosing the proper volume within which to indicate a selection is not as straightforward. Theoretically, the proper volume for a flat text item is a viewpoint aligned parallelogram that allows the user to select the item at the same depth in front of and beyond the text. However, in practice, selection volumes are usually of a fixed arrangement and combine with poor depth perception to create significant selection difficulties. As a result, interface elements often have to be oversized in order to make their selection efficient. An increase in interface size contributes to display real-estate usage and increases the range of motion required of the user.

The limitations of virtual hand selection have contributed to the popularity of ray-casting for selection. However, ray-casting comes with its own set of problems on traditional interface elements. Because it relies on intersections between a ray and objects, intersections with menus and icons can become more difficult the closer those elements move to the user. As a flat surface moves closer to the user viewpoint, the angle between the ray and the surface becomes increasingly oblique. Another issue that tends to force ray-casting interfaces away from

the user is the fact that many HMD systems do not provide enough binocular overlap to provide good stereoscopy at close distances. Without the depth cues of stereo vision, users have trouble using ray-casting at close distances (34). In fact, researchers often run their HMD systems in strictly monoscopic mode in order to avoid rendering twice and, therefore, increase the effective display frame rate. Moving interface elements farther away from the user does simplify the ray-casting task into one that is more two dimensional. However, once interface elements move away from the user, significant issues of how to prevent interference between the user interface and surrounding objects arise.

Seeking to blend the strengths of both desktop and 6 DOF hand input, Coninx and others have created hybrid interfaces that feature traditional desktop mouse interaction on 2D interface elements coupled with immersive interaction techniques (38). Systems such as these, attempt to leverage the strengths of both mediums. However, the problem with such hybrid systems is that they force the user to move between two very different interaction schemes. The user might literally have to set down a 6 DOF control device to use the mouse based interface elements in an efficient manner. This inconsistency in interface design is the same reason that many novel input devices such as dials and pen interfaces for CAD systems never found broad usage on the desktop. A single canonical input method such as the desktop mouse prevents user confusion and inefficient transitions between input devices. Whether presented near, far or mixed with desktop input devices, the results of directly bringing 2D interface elements into 3D have been disappointing. On most counts, 2D interface elements have not managed to provide the efficiency of system control that they routinely provide on desktop systems.

2.5.1.3 Physically Intuitive Interfaces

High learning curves, poor usability and a feeling that traditional interfaces perform less efficiently in 3D space motivated researchers to seek more appropriate interfaces to 3D applications. In a 1992 paper, Kay Stanney argued that virtual environments were “in need of design metaphors uniquely suited to their characteristics and requirements” (39). These sentiments echoed those of researchers who felt that the techniques we routinely use to interface the physical world can be leveraged as a natural starting point for interacting with 3D applications. This approach seeks to incorporate metaphors from the physical world into 3D interactions in ways that avoid the display real-estate and learning curves associated with traditional interfaces.

The teleportation scheme used by the WIM technique is one example of a physically intuitive interface. Grabbing the model and moving it to over the head to generate a viewpoint requires very little in the way of traditional interface elements. However, strict adherents to the philosophy even seek to avoid such contrived constructs as disembodied maps in an effort to prevent interface clutter. Another example involves the use of a virtual magnifying glass to view content at a distance or the use of a virtual knife instead of more traditional interface elements to divide a geometric model. There is some irony in the physically intuitive movement. The typical hardware environment that most users encounter has no haptic feedback or high-fidelity input. In the context of pressing buttons to “grab” objects and pushing a joystick to “move” through a 3D space, one could easily argue that the native environment of most 3D user interfaces is not that of the physical world but one of obvious mediation with a computer.

A similar movement known as tangible computing has also taken place in the physical world. This movement seeks to remove the dependence on traditional device interfaces in favor of more intuitive physical metaphors. A good example is the Topobo assembly system with kinetic memory (40). The user builds a robotic entity out of small interconnecting pieces similar to LEGO blocks. The user then manipulates the robot manually to program the various behaviors of the robot. The tangible computing movement and physically intuitive 3D interface movements are appealing because they seek to simplify user interfaces in novel and intuitive ways. Yet these principles should already be a consideration in all good interface design. Just as physical walking will always be the preferred locomotion metaphor; there will be other circumstances where physically intuitive interaction metaphors are optimal. Unfortunately, there are limits to how far these techniques can go towards eliminating the need for more traditional system control. The majority of real-world applications are simply too abstract in their context and too complicated to benefit substantially from such an approach (36). As a result, it is likely that both 2D and 3D applications will continue to require interface structures such as the traditional menu that organize large numbers of functions into an easily accessible format.

2.5.2 Usability

The desktop computer has been notable for its relatively consistent input devices, input schemes and hardware configuration. These factors combined with the true utility of early spreadsheet programs such as Lotus 1-2-3 to set the stage for the gradual increases in usability and productivity that we now take for granted with desktop computers. The much less constrained and more technically demanding field of 3D user interfaces has

not benefited from any of these factors. More often than not, 3D applications are vehicles for technological and user interface experimentation instead of actual augmentations of user productivity.

The usability and utility of 3D applications has not been sufficient to motivate their broad usage in the public domain. The very nature of the medium, its technical complexity and a lack of canonical interaction schemes have combined to create user experiences that are usually fatiguing, cumbersome and frustrating. Moreover, a lack of applications that merit the additional effort associated with using 3D technology has contributed to a somewhat unclear picture about its utility beyond strict simulation of reality.

2.5.2.1 Technology

One of the complaints lodged against Engelbart about his famous demonstration in 1968 was that his method of making on screen selections was inefficient. The system, as demonstrated, required a four-step process to choose and acknowledge the selection of an on screen item. Engelbart later pointed out that he implemented the scheme to prevent errors because his pointing device was very inaccurate. The accuracy of mouse pointing quickly developed to obviate such compromises, but some choose to take the accuracy of the contemporary mouse for granted when looking backwards at the work of Engelbart. Technological concerns have played a significant role not only in helping to make immersion a goal of 3D technology, but also in defining the user relationship to the technology itself.

For many years the HMD systems used to create immersive interactions were too heavy to allow long term use. The technological limitations were so significant that a CRT based system supported completely by a counterweighted metal boom was popular as recently as ten years ago (41). The technology used to create 3D user interfaces has been expensive, cumbersome to use and notoriously difficult to maintain. Tracking systems and display systems often had to be recalibrated after each use and the many devices that have to work together increase the probability that something will go wrong. This contrasts sharply with the relatively contained desktop devices that were being built as early as the mid-seventies when the term "personal computer" was first coined.

The technological limitations associated with 3D technology have limited user access to the devices. A relatively short usage window and cumbersome headgear and tracking devices put a burden on the application to make the time spent worthwhile. Unfortunately, outside of novelty and entertainment, the results delivered by 3D

applications have rarely been worth the effort. One significant problem has been the lack of a canonical interaction scheme. More often than not, 3D applications are merely testbeds for novel interaction schemes and interaction devices instead of attempts to create a productive work environment. As a result, many users spend the majority of their time trying to learn a new user interface or input device. In contrast, the success of desktop systems has been founded on two input devices, the keyboard and mouse, both of which have hardly changed in forty years. Moreover, the relative consistency of the desktop Windows Icons Menu Pointer (WIMP) metaphor has created a society of computer literate people that can easily leverage their existing skills to interact with new applications.

2.5.2.2 Physical Fatigue

One of the main factors that motivated the design decisions of Doug Engelbart and his colleagues was a desire to optimize the physical effort put into operating the computer. Engelbart developed a specialized keyboard called the Keyset that used chording, or combinations of keys, to allow the user to enter text while using the mouse simultaneously. Although these innovations were not pursued energetically by his team when they moved to Xerox Parc, they illustrate the extent to which Engelbart was concerned with minimizing even the need to move the hands or the eyes from the task at hand. The same physical considerations have been notably absent from 3D user interface research.

The idea of minimizing the physical motion of the user is at odds with efforts to deliver a compelling sense of immersion in a 3D virtual space. A user that is standing and motivated to move around is more likely to appreciate the novelty of synthesized first person perspective in a virtual space. In addition to standing, typical virtual hand interactions such as reaching for and manipulating objects around the user are significantly more taxing than analogous “drag-and-drop” tasks on a desktop computer. The physical nature of immersive 3D interaction also combines with heavy head mounted displays and cumbersome tracking equipment to increase fatigue. The development of the CAVE in the early nineties represented a unique opportunity to reduce the fatigue associated with immersive interaction. Instead of a bulky HMD, a CAVE display system allows the user to wear special shutter glasses that are significantly lighter. Still, until recently, even the minimum tracking configuration of 6 DOF head and 6 DOF hand input has required encumbering wires attached to each. While a significant amount of HMD research has been done in a sitting position to reduce fatigue, CAVE interactions often involve several persons at a time and thus encourage standing and the resulting fatigue associated with it.

Minimizing user fatigue has been a secondary concern in 3D user interfaces more often than not. The virtual reality application developed by Bowman for building animals habitats allowed users to travel through the environment and manipulate objects in a first person manner with the Go-Go technique. The system also used the tablet interface to allow the user to both travel to other locations and move objects on the horizontal plane. This combination allowed users to initiate gross object movements or virtual travel actions and then use first person techniques to make fine object adjustments or viewpoint adjustments at the local of interest. This example illustrates that virtual reality applications need not completely abandon first person interaction techniques to reduce fatigue.

2.5.2.3 Application Domain

One application domain that has received significant attention is the viewing of scientific data. Immersing a user within a 3D representation of data has the potential to deliver some insights not available elsewhere. The petroleum industry has found that immersive technology such as the CAVE is very useful for the analysis of geological sonar data. While the cost and maintenance of immersive display systems has prevented more widespread use for scientific visualization, another problem has been the user interface provided to that data. Aside from simply moving around within the data and viewing it egocentrically, 3D applications rarely provide any significant ability to control or manipulate the data because 3D user interfaces are notoriously difficult to build, program and use.

Exocentric interaction has proven valuable for surgical and dental training while egocentric viewing has proven valuable for design analysis of architectural content and other similar tasks. Virtual reality technology has also proven useful in other areas where simulation of reality helps to significantly defray costs. Military training, flight simulation, industrial training, product prototyping and psychological treatments that involve real-world scenarios have found significant utility. However, what most of these applications of the technology have in common is that their utility lies in simulating the physical world.

Beyond the challenge of assembling and maintaining an immersive 3D system, there is a significant amount of effort required to program 3D applications that have non-simulation user interfaces. A certain hype, or irrational exuberance, has surrounded the medium because of its compelling re-creation of first person space.

This excitement has led to virtual reality applications in a number of domains that have little to gain from it. Virtual reality applications developed for domains such as teaching petroleum engineering, understanding gorilla social structure and learning how to style hair are just some of the examples that have brought little practical benefit to users (42,43). Given the effort required, one would expect that the choice of application domain would be made carefully. However, finding domains that benefit from 3D applications has been one of the most significant challenges for 3D user interface development.

Consider the example of 3DM, a system for creating 3D models in an immersive environment. The system, developed in 1992 by Butterworth, was an ambitious system that would appear to be an appropriate use of immersive technology (44). The system borrowed the use of a 2D interaction panel from desktop interfaces along with many of the same features that contemporary desktop modeling packages included. However, because it used virtual hand interaction techniques, selecting items on the interaction panel was more difficult than using a desktop system. Also, depth ambiguity made it difficult to accurately position the vertices of a model without relying on grid or vertex snapping. Efforts to inspect the accuracy of vertex placement required the user to physically walk around the content to change viewpoints. From the standpoint of a practical application, many other problems kept 3DM from achieving high usability. The limited user interface left many functions available on desktop systems unavailable to the user. Moreover, the models created within the system could not be exported to other modeling packages for further development. Finally, there was no opportunity for users to learn the user interface without donning the immersive equipment themselves. As a result, a significant portion of the energy that users devoted to the system had to be spent learning to use it. In the end, it is understandable why users would not find it a compelling alternative to desktop based development applications.

Applications that leverage 3D environments to create work environments that are more productive than desktop systems for symbolic manipulation tasks have been hard to come by. Three examples demonstrate some of the attempts to develop what Bowman has called “information-rich virtual environments” (45). Feiner developed a see-through display system that allowed a user to view a very large XWindow screen by changing the orientation of their head (46). Individual XWindows could also be attached to individual trackers so that they would always appear at the same location in the surrounding environment or move with physical objects. Fenier mapped the XWindow bitmap directly onto the low resolution display device so that text and other information was always

perpendicular to the viewer. Links could be created between windows that would present as a solid line connecting them.

Angus created a Mosaic browser completely out of graphical elements instead of using a bitmap approach (47). Instead of creating a 3D object browser within the 2D browser, links to 3D objects presented actual 3D models floating above the browser window. At Georgia Tech, Bowman created a virtual venues application that imbedded contextual information at various locations throughout the virtual space (48). Users used a tablet-based menu to request additional information related to various locations. Some selections played back scenes within the environment while other menu selections provided textual or auditory information. Bowman implemented what he called spatial hyperlinks that took the user to another location along a time-based path.

The three applications described above are all examples of work that found little practical value for persons at the time. Having a browser in a virtual environment, attaching XWindows to objects in the environment and context sensitive information in virtual environments were unlikely to cause users to return repeatedly. Yet, as we will see in the next chapter, each of these applications acknowledges an important part of what will be eventually be a revolution in the way people interact with information and the applications they are familiar with on the desktop.

3. THE COMING AUGMENTED REALITY

The term augmented reality is typically used to describe the superimposition of computer generated content over the contents of the physical world. However, augmented reality has the potential to take on a more dramatic meaning by also serving to significantly augment the relationship between man and the physical world around him. This augmentation of the individual will involve the creation of a new reality that exists not in a physical space or a cyber space but in a new space forged at their intersection. This new reality will finally extend the significant power and productivity of modern computers into the physical world immediately around us.

Three significant trends, the geo-referencing of the internet, the internet connectivity of everyday objects and the ubiquity of display technology are helping to realize this future. These three trends are being pushed forward by demands for increased efficiency and productivity from consumers and business people alike. The result of these trends will be an augmented reality that allows individuals to control the devices around them with the same level of control we currently wield over more symbolic information. This extension of computational power will extend beyond individual device control to include the application of techniques that have formerly been confined to the desktop directly onto the objects in our midst. A significant part this coming augmented reality will be the power that arises from the ability to view the environment around us from arbitrary viewpoints. This technique, a mainstay on desktop environments, has not been an important aspect of prior virtual and augmented reality interfaces.

This vision of a coming augmented reality highlights the need to develop efficient and productive interaction techniques for use in the general 3D space around us. In the future we will need techniques for interfacing devices from a distance, using traditional 2D application interfaces and manipulating viewpoints and objects in 3D space. While many 3D applications do require specialized interfaces, the coming augmented reality will require a canonical interaction methodology that smoothly integrates a broad spectrum of needs.

3.1 A Decontextualized Computer Revolution

Prior to the computer revolution that occurred in the middle of the last century, our technological achievements worked directly on the physical world. Beginning with the era of the computer, a separate space

developed where our actions began to play out. This space, commonly referred to as cyberspace, existed long before networks began connecting computers together to create the World Wide Web. The most basic example of this cyberspace is the computer simulation. The simulation of physical processes within a computer creates a proxy that can be manipulated, observed and repeated much more efficiently than is possible in the physical world. Today, proxies for real-world items have greatly increased the efficiency with which we do business and lead our lives. Inventory is bought and sold, letters and documents are exchanged and people meet and exchange ideas more productively than the physical world allows. This parallel cyberspace now allows any person with an internet connection and a credit card to initiate the manufacture of a customized part, have it gift wrapped and hand delivered to another person across the world with the mere press of a button.

Despite our significant power to efficiently manipulate, organize and exchange information in this cyberspace, it has, to date, remained distinctly separate from the physical world we inhabit. The information and proxies represented within cyberspace frequently have had no context within the physical world. While we may easily identify an item in a computer inventory, we may not be able to actually find the physical object in our midst. We could easily have a strong relationship with a person in cyberspace and yet not even recognize them walking down the street. And the reverse holds; we often have little means of taking physical objects in our midst and finding their corresponding proxy in cyberspace. This decontextualization has created the conditions where technology has empowered the individual in significant ways but remains distinctly separated from our physical existence.

This lack of context is such that we can find a rare antique from the comfort of the couch but remain ignorant about where the next gas station lies down the road. We can marshal the forces of our whole neighborhood to a cause but must still physically get out of our chair to operate the devices in our home. This environment often forces us to painstakingly convert the qualities of the physical world into a format that is appropriate for the computer world before any more powerful work can be done on them. Measurements must be taken, databases must be updated and contacts must be entered into mediating devices in order to yield the benefits of computation. The conditions become even starker the closer to our everyday activities the tasks move. The mere receipt of a business card creates the dilemma of how to introduce it to our digital world. And, the computer frequently acts like a blind man that must be manually introduced to peripherals that may be lying right next to it.

3.2 The Emergence of Three Trends

The separation between the physical world and cyberspace can be overcome by approaching the problem from three directions. First, the representation of the physical world within the computer must become more concrete and more detailed. Objects that have proven useful as mere symbolic representations must now be manifest in more physical detail. Secondly, ordinary objects and devices within the physical world need to begin to communicate directly with their proxies in the computer realm. These objects need to take a more active role in defining their status within cyberspace. Finally, objects in the computer realm need to gain the ability to appear more readily in the physical world. As would have it, these are exactly the three trends that are moving forward at an ever increasing pace. Our desire to digitize and query information in the world around us is rapidly creating a detailed representation of the world around us. The day is also rapidly approaching when every item in our midst will be connected to and will routinely update its status with that database. And, our ability to alter the visible world with digital content delivered from the realm of cyberspace is increasing every day.

3.2.1 The Geo-Referenced Internet

A revolution is taking place that is increasingly mapping the information held in cyberspace more concretely to the physical world around us. Information that previously might have been abstracted in a list or diagram is today being represented more directly in its physical context. A list of homes in the area is now a map showing those homes in relation to one another. A query of department stores is now based on the location of the user and presented in the form of a map. Google, Inc. has been a leader in this trend by allowing end users to create their own maps on the fly with what are known as Mashups.

Over time the accuracy of the geo-referencing is increasing. What today is often a 2D representation of objects in latitude and longitude will soon be actual 3D representations of the homes, buildings and other structures at that location. Again, Google is leading the way with its Google Earth software that does what Google maps does for 3D. Users can easily create their own 3D representations of objects with included software (formerly known as Sketchup). The trend is not confined to consumer information. Productivity and efficiency are quickly moving such paper mainstays as building surveys and utility schematics online. Many areas from law enforcement and homeland security to urban development and utility maintenance are beginning to benefit from these geo-referenced databases.

Increasingly, the once disparate databases that stored information such as demographics, housing prices, retailers, medical information, etc. are being integrated. This trend is not happening because the government and other sources are determined to have unfettered access to our lives. The trend is proceeding because we, ourselves, want access to that information. On an everyday basis, people are asking for greater access to aggregated information about the environment around them. Retailers are increasingly integrating the physical dimensions of their stores and warehouses with their inventory to increase the efficiency with which they can organize and access their products.

3.2.2 The Ubiquitous Internet

When combined with ever more accurate satellite, aerial and, even, ground based acquisition of geographic information, the trend of geo-referenced data promises to create an ever more concrete representation of our physical world. What this mirror image of the physical world will also require is systems to automatically update that information on a timely basis. This significant second trend, a direct and constant connection between objects and their cyberspace proxies, is already happening as processing power and imbedded intelligence become ubiquitous.

Already, Radio Frequency IDentification (RFID) tags allow an item to be identified and its database proxy to be queried immediately. From the other direction, these tags are moving from indicating not just their product identification but also their exact location in the world (49). This opens the door for databases to automatically know not just when an item arrives but also exactly where it can be found. Yet, even this level of interconnection relies on existing database entries to identify the full functionality of a given item. Increasingly, more complicated objects such as appliances and consumer electronics are becoming internet aware. Inexpensive internet servers on a chip have made it feasible to connect common appliances such as security cameras, printers and refrigerators directly to the internet (50). The motivation for this trend is the convenience and flexibility of being able to access and control these devices from any internet location. This embedding of intelligence into ordinary objects allows them to be queried directly to determine their operating state and even a complete list of the functions they can carry out. Furthermore, the advent of Bluetooth technology is now allowing the recognition of and connection with peripherals when they enter within range of the computer. Devices are becoming more aware of one another and are beginning to automatically create connections amongst themselves.

Wireless communication between devices allows them to communicate directly without centralized control. The day will soon arrive when asking the air conditioner to automatically draw the blinds during the day will be a reasonable request. This increased interconnectivity also opens the door for these devices to use their wireless connections to determine their position in the physical world via standard radio triangulation similar to that used by modern Global Positioning Systems (GPS) (51). This internal ability of a device to determine its position and update the geo-referenced representation of its self is a crucial part of creating the coming augmented reality. Furthermore, wireless access points currently only expand internet access as far as their reach alone. By forming mobile ad-hoc networks that relay internet communications, everyday devices are also likely to play a significant role in expanding the coverage of internet access (52).

3.2.3 The Ubiquitous Display

Just as objects in the physical world are increasingly making their presence in cyberspace felt, the results of the flow of information in that alternate world are flooding into the physical space around us. The third significant trend, the ability to interject computer generated content directly into the physical world is taking place all around us. Already, ordinary people are accustomed to the routine interjection of digital content into pre-recorded video content. The trend is now becoming commonplace for live video content as the superimposition of graphical content is now routinely superimposed onto the environment during televised sports broadcasts.

The same trend is also taking place in the physical world. Commodity projectors, formerly confined to screens and dimly lit rooms, are increasingly being used in much more realistic environments. Algorithms already exist that allow a combination of camera and projector to scan the projection surface and use compensatory techniques to present undistorted images in real-world conditions (53). Currently these algorithms are capable of automatically estimating the surface shape and lighting conditions every few seconds. The coordination of multiple, un-stabilized hand-held projectors to create a single composite image has also been demonstrated in the literature. In fact, a recent industry survey estimated that 8% of all cell phones would have an embedded portable projector by the year 2012 (54).

A number of other technologies are also beginning to make the prospect of ubiquitous stereographic display much more realistic. The most exotic of these technologies is the Virtual Retinal Display (VRD), a device

that projects a low power laser from a small aperture to paint images directly onto the retina of each individual eye (55). The VRD, already being used in the medical industry and by the military, is especially attractive because of its ability to provide bright images even in sunlight and modulate the light wave of individual pixels to match the focal depth at which it appears. The literature is also seeing an explosion in the number of techniques that can deliver stereo content with no need for glasses or other equipment. The Varrier display technique developed at UIC, uses parallax barriers to dynamically direct independent left and right eye viewpoints to a user via face tracking technology. This technology has been deployed on composite 65 LCD panel systems to deliver over 25 Megapixels of display resolution to each eye (56).

While the Varrier is currently only found in laboratories, commodity projectors are also capable of generating stereographic imagery through the use of shutter glasses. When auto-calibrated in real-world environments, current consumer projectors are capable of displaying 60Hz stereo display to users wearing lightweight shutter glasses. It is not unreasonable to imagine an environment where the ubiquitous MP3 player earplugs are supplemented with designer shutter glasses for the ubiquitous display of stereo content. The maximum size of LCD flat panel displays has doubled in the last 4 years and research environments are pushing the effective resolution of display devices into the hundreds of Megapixels (57). This is most significant in light of the fact that the effective resolution of the human eye has been estimated at just over 500 Megapixels¹. Consumer demand and technological innovation are increasing the resolution, size and flexibility of display technologies so rapidly that is it not unreasonable to expect the computer screen as we know it to disappear altogether.

3.3 A Truly Augmenting Reality

At the intersection of a concrete representation of the physical world and the independent ability of objects in our midst to communication with and be controlled by the internet lies the potential for a completely new reality. This new reality lies in the ability to manipulate and control not only remote processes from the desktop but also the ability to master the physical space around us without leaving our seats. Anyone who has ever owned a remote control television set understands the significance of this power. To be able to remotely control any electronic device, accurately and efficiently, from any location has the potential to dramatically change our lives as we know it. This powerful ability, combined with the ability to display computer generated content at will within the

¹ <http://www.clarkvision.com/imagedetail/eye-resolution.html>

surrounding environment, sets the stage for humans to completely abandon our current reliance on hardware to mediate our relationship to cyberspace.

3.3.1 A Truly Universal Remote

We are already beginning to see the beginnings of a change in how we interface with the devices around us. The cell phone is quickly becoming the Swiss Army Knife of devices with built in cameras, video playback, music playback and voice control. It is already becoming relatively common for cell phone users to purchase items from vending machines and control music software on desktop computers via Bluetooth connections (58). Internet connectivity on cell phones is also becoming commonplace and with it comes the ability to interface directly with additional devices around us. The nature of this interface between personal computing devices and other devices is of critical importance to a truly augmented reality. Not only will we be able to control devices without leaving our seat, but we will also be able to modulate their interface to suit our own needs. By making their functionality public, internet enabled devices allow devices such as a cell phone to reconstruct an interface to that device that is both customized to the user and creates familiar patterns of use for that user. This means that every microwave, every printer and every television will operate with a familiar interface and become immediately usable even to those unfamiliar with its specific brand.

The ubiquitous display of computer generated content promises to allow us to eventually abandon the cell phone in favor of more context sensitive interactions with devices. Interfaces, formerly constrained to LCD displays, can be presented next to or floating in the air above the device in question. This interaction in context is often more appropriate because the interface and the device that is being controlled are collocated. One concrete example of how such interfaces might be realized has been demonstrated by Ramesh Raskar and his group at MERL. By using standard computer vision techniques, a hand-held projector can both project a cursor down its centerline and simultaneously display a stabilized image of an augmented interface onto a flat surface (53). The result is a hand-held device that both renders the augmented interface and acts as the input device itself.

Many people are familiar with the quote from science fiction writer A. C. Clarke, "Any sufficiently advanced technology is indistinguishable from magic²." A common scene in the Star Trek television series was that of an engineer approaching, diagnosing and reconfiguring alien technology without the slightest difficulty. To

² http://en.wikipedia.org/wiki/Clarke's_three_laws

most people this would appear impossible. Yet, the future augmented reality provides a plausible explanation for how such a process might unfold. The device in question simply makes its functions available to the personal computer system of the engineer. And, his system then overlays familiar functionality onto the alien controls and displays in front of him, allowing him to work almost as easily and effectively as if he were back on the Enterprise.

3.3.2 The Application of Computation to the Physical

The real power of the coming augmented reality lies in the significant ways that geo-referenced computing will affect our productivity. The ability to control devices efficiently from across the room is certainly compelling, but consider the more subtle yet powerful ability to work magic outside the realm of what is visible. Imagine accurately controlling a lawn mover from the comfort of your seat, initializing the analysis of a faulty device by merely looking at or knowing exactly what lies around the next corner. In addition to manipulating devices and viewing their associated information, the coming augmented reality will bring a sea change in our ability to leverage the power of computation directly onto the physical environment around us.

The contemporary vision of augmented reality already includes many useful concepts that will be a part of a truly augmenting reality. An augmented reality system can be expected to overlay the results of a gas station query over the viewpoint of the user, highlighting those buildings either visible or occluded that are of interest. Visions of augmented reality already include the ability to select buildings and objects within the physical environment and retrieve information about them from databases or web pages. Augmented reality is also predicted to allow users to create powerful subjective views of their surroundings. Whether using strict visual overlays or more modest methods such as the screen of a tablet computer, future systems will add important information to user controlled viewpoints to aid productivity in fields such as utility work, mechanical repair, device training, law enforcement, urban planning and surgery.

What separates contemporary views of augmented reality from a truly augmenting reality is a more significant integration between existing productive desktop software and a geo-referenced mirror of the physical world. Augmented reality research predominantly takes for granted that its power is gained from the registration of information within the viewpoint of the user and an ensuing interaction within that same context. However, the productivity of desktop systems is often associated with much more flexible and powerful viewpoints of information. Many of the strengths of desktop systems lie in their symbolic manipulation of data in ways that have

no correlate in physical space. Furthermore, architects, planners, surgeons and engineers routinely leverage the flexibility of multiple and varied viewpoints of 3D structures to increase their productivity. In a truly augmenting reality, the user will use their relationship to the physical world to initialize the application of traditional computation onto the environment around them.

Let us use contemporary GPS mapping systems as an illustrative example of this subtle but powerful integration. Contemporary GPS systems are capable of generating 3D viewpoints similar to those presented in front of the user. When queried for content such as a gas station, a mapping program is likely to provide some context by presenting the information from above or at an elevated position. An augmented reality system, on the other hand, is likely to overlay that information directly onto the user viewpoint. A more powerful system is one that initializes the query based on what direction the user is looking, generates an AR composite viewpoint and then allows the user to smoothly manipulate that viewpoint from their current perspective into one that is more useful. Such a transition allows the user to gain valuable insight into the relationship between their current position and the location of queried content around them. The resulting information can then be available for search and manipulation with more traditional symbolic tools such as distance, price or alphabetical orderings in a familiar 2D application window.

Consider another example related to the architectural design of buildings. Current augmented reality research efforts focus their attention on techniques within first person for the interactive design or redesign of buildings (59). Again, restricting the user viewpoint to that provided by user position is an inefficient approach. The available model of a current building may not be sufficiently accurate enough to enable the user to fully appreciate the relationship between it and their current viewpoint. As a result, the current viewpoint of a building is a more appropriate point from which to initiate the manipulation of any model of that building. A subset of the existing viewpoint can literally be framed by the user, exchanged with a digital version and then manipulated in a traditional application at that very location. This process does not preclude the use of what is probably the most important aspect of traditional augmented reality, the registration of digital information with the real world. After making design changes to the model, the superimposition of those changes over the existing viewpoint is of significant value to both the user and others on the scene, yet forcing all work to take place in that context is not.

Some incarnations of a more concrete representation of the physical world within cyberspace are more easily realized than others. Examples that include details about utilities, roads, buildings and connected devices might appear more feasible than those that include real-time features such as people, animals and plant life. However, even when the information about the user is limited to location and viewpoint, many aspects of human pose and even lip movement can be simulated by relatively straight forward means. This simulation of human features for virtual reality avatars has already been a research topic for many years. Simple video cameras can play a significant role in interpreting the state of the world in the absence of other more discrete sensors. Current techniques exist for projecting the video representation of a person directly onto an existing model of the user. In fact, techniques for determining the structure of objects at real-time using multiple or even single camera silhouettes have already been demonstrated in the literature (60). Well established image-based modeling techniques can reconstruct the physical dimensions of a space with a minimum of two camera views (61). These and other techniques, along with the inexpensive proliferation of internet enabled security cameras, will enable the ready realization of arbitrary viewpoints of the physical world by aggregating multiple sources of information available in cyberspace.

The ability to generate arbitrary viewpoints originating from outside the spaceship was always a curious aspect of the Star Trek television series. In light of the current discussion and our present ability to visualize images billions of light years from the earth, perhaps generating such views within outer space will not raise eyebrows in the near future. Consider some of the implications of this technology for the individual. Combinations of wireless devices on our body and security camera imagery have the potential to literally teleport a representation of ourselves to any location at any time. This virtual or holographic conferencing is likely to find broad public support for the sharing of such information. This same reconstruction of reality can also allow a person to render the walls of their home invisible or even zoom their vision beyond those walls in real-time to inspect the commotion in the street outside (62). From this perspective, the idea of remotely controlling a lawn mower from within one's home no longer seems unfeasible when arbitrary viewpoints of the task are available.

3.4 Future 3D User Interfaces

An important aspect of the possible scenarios described above that has gone unaddressed is the methods we will use to satisfy the demanding interactivity that such a reality calls for. The abstraction of device interfaces and their presentation within the environment will require techniques for efficiently interacting with those

interfaces in the 3D space around us. The ability to create and utilize alternate viewpoints of our surroundings will require methods to easily manage those viewpoints. And, the incorporation of existing applications like those already available on the desktop will require significant improvements to prior techniques for manipulating symbolic information in 3D space. The potential of this future reality puts a significant burden on technologists to develop practical and efficient interfaces for use in a more generalized 3D space.

From the perspective of a coming augmented reality, the importance of integrating current desktop computing power with the space around us becomes more obvious. Despite the fact that it may be less intuitive to tackle, research that seeks ways to imbed, integrate and access computer generated content within our environment is critical to realizing such interfaces. There are many questions to be answered about how to approach developing such 3D interfaces. Can existing applications be used in 3D space with efficiencies on a par with desktop systems, or will they require significant reformulation to operate in such an environment? If we seek to abandon mediating devices like the cell phone, what display and interaction methods are appropriate for interfacing the devices in our midst? What input devices should future 3D user interfaces make use of; the hands, a stylus, a laser pointer, physical props? While managing our own viewpoint is a natural physical process, managing disembodied alternate viewpoints may prove a difficult task. What sort of techniques will allow users to manage alternate viewpoints such that their benefits are not outweighed by the overhead of their management? The first step to answering these questions is to outline a set of guidelines for the development of productive and efficient 3D user interfaces informed by the discussion here and that of the previous background chapter.

4. GUIDELINES FOR PRODUCTIVE 3D USER INTERFACES

The prior two chapters covered both the past and future of 3D user interfaces. Along the way, a number of points have been made about what techniques do and do not work well for 3D user interfaces. This chapter will condense those observations into development guidelines of productive 3D user interfaces. The guidelines presented here are not intended to cover all aspects of 3D user interface design, but they do seek to make new contributions to a growing dialectic on the subject.

The guidelines presented here cover four main topic areas; domain, usability, efficiency and application development. The first section argues that there are two separate domains within the field of virtual reality; simulation and augmentation. The following sections attempt to show that many concepts of user interface design and application development associated with desktop interfaces are appropriate for 3D applications. The chapter ends with an argument for a unified framework that allows 2D and 3D user interfaces to transition smoothly between desktop and fully immersive use.

4.1 Domain

The domain of 3D user interfaces has for many years been associated with the term virtual reality. However, the term virtual reality has likely outlived its usefulness. Today it is readily associated with the hype and subsequent disappointment attributed to it during the last decade of the last century. The term has a connotation of neither efficiency nor productivity and thus holds back the potential of the underlying 3D technology. The Gartner Group, a technology consulting firm, developed the Hype Curve in the mid-nineties to describe the effects of hype on emerging technologies³. The hype curve describes a surge in interest followed by what is called the “trough of disillusionment”. This dip in the enthusiasm for a technology precedes what is called the “slope of enlightenment”, during which the benefits of practical application of the technology come to be understood. Today, 3D user interfaces are most certainly in the trough of disillusionment and poised to find more practical applications than the all or nothing expectations that have been heaped upon them.

³ http://en.wikipedia.org/wiki/Hype_cycle

4.1.1 Simulation

The contemporary notion of virtual reality would benefit from separation into two domains; simulation and augmentation. There are a great number of useful applications in the domain of reality simulation. The early success stories in flight simulation, design prototyping, industrial training, surgical training and the more recent success of psychological evaluation and treatment fall squarely into the category of simulation. Architectural walkthroughs and immersive product design reviews also fall under the category of simulation. The criteria for such applications are straight forward, to simulate the physical conditions with as much fidelity as is necessary to accomplish the goal of the application. The utility of simulation environments is directly tied to their return on investment. In most cases where simulation is used, no physical setup could be created and maintained that would achieve the same goals for less investment. Immersion is normally deemed a useful goal for simulation environments, and the use of immersion is a way of differentiating between simulation and other more traditional methods of training and evaluation.

Simulation applications generally have little use for user interfaces as they are normally understood and are only concerned with efficiency and productivity in the context of the tasks they simulate. Simulation environments are not concerned with reducing the fatigue of their users. In fact, simulation environments for exercise, sex and virtual travel are all very likely future applications of the technology. Simulation environments are appropriate places for the exploration of interaction schemes that attempt to compensate for limitations in current technology. Not all aspects of a task need be simulated perfectly. Virtual travel and button oriented object manipulation techniques that seek to retain immersion and avoid user frustration can be appropriate in the context of simulation. Furthermore, implicit interface elements such as humanoid agents and voice control are probably more appropriate in simulation environments because avoiding discrete instrumentation helps to maintain immersion.

4.1.2 Augmentation

The domain of augmentation has a completely different set of goals from simulation. Augmentation seeks to use 3D technology to augment the ability of humans to accomplish tasks, and, as such, the effectiveness of augmentation should be judged with respect to all other methods available to accomplish the same task. An augmented reality system for enforcing building codes should be evaluated against more traditional methods of doing the same task. This makes calculating the return on investment for augmentation a more straight forward

task. In contrast to simulation, there is no place for undue physical exertion in augmentation applications. Immersion should be completely ancillary to the primary goal of efficiency and productivity with which tasks can be accomplished. There is no justification for using an analogous physical task in an augmentation application unless it can be justified through increased efficiency or productivity. First person methods may be appropriate at times but should not be imposed at all times. Interfaces focused on augmentation need to maximize system control and therefore are more likely to need discrete interfaces that allow explicit system instructions.

The suggestion that simulation and augmentation become separate disciplines does not in any way preclude their use in the same context. It is merely a useful delineation that helps to avoid the conflicted goals of many prior efforts. For every simulation environment, there is a candidate augmentation environment that is used for its development, commissioning and maintenance. Also, fully immersive display environments are appropriate places to simulate the use of augmentation applications. Or put another way, what we often call virtual reality today is really just simulated augmented reality. Furthermore, augmentation should not be strictly associated with the superimposition of content over reality. A fully immersive environment for the development of 3D models or even 3D games is an augmentation environment. A prime example of such an application is the Virtual Director application by Donna Cox, Bob Patterson and Marcus Thiebaut (63). This application helped choreograph camera paths through computer generated astronomical simulations from within a CAVE display environment. In contrast, it is also becoming increasingly popular to purchase equipment that allows a person to virtually film a virtual scene with equipment that appears and functions like normal camera equipment. Ironically, this trend is concerned with augmenting the ability of the human but attempts to do so by simulating reality.

4.1.3 Augmentation Application Areas

When considered against the backdrop of their return on investment, the immediate application areas of augmentation are narrower than the ambitions placed upon virtual reality. Their most obvious use involves situations where the surrounding context plays a significant role in the task involved. Utilities management, interactive urban planning and disaster management are domains where the need to integrate digital information with the physical environment are of obvious benefit. These applications involve the application of computational power to the surrounding environment and require the development of efficient input techniques. Those same techniques are likely to apply to the second most powerful use of the technology, the extension of computing into the surrounding environment. Efforts to create mobile computing platforms and create augmented office

environments are good examples of this application domain. Presentation environments, collaborative work environments and sophisticated visualization environments like the CAVE are environments where users need access to traditional computing resources but where typical desktop interaction is impractical.

4.1.4 A Superset of Two Dimensional Interfaces

The two major applications areas of augmentation both represent movement from two extremes; the desktop and the physical environment. As a result, research into 3D user interfaces should not be considered as separate from desktop applications. It is probably more appropriate to think of 3D interfaces as a superset of 2D interfaces; increasing the ability of desktop interfaces to work in more flexible conditions while offering the increased productivity of contextual interaction with the environment. Often research into 3D user interfaces has assumed an abrupt transition into fully immersive environments. However, it is more likely that desktop and hand-held interfaces will make a gradual transition into 3D environments via the trends described in the prior chapter. From this perspective, the domain of 3D user interfaces includes any number of transitional states between desktop and fully immersive use. Embracing 3D user interfaces as occurring along a continuum between desktop and immersion highlights the need to hold those interfaces to the standards of usability that are currently applied to the desktop.

4.2 Usability

The days when novelty and hype can sustain enthusiasm for 3D user interface technology have likely passed. If 3D user interfaces are to move up a slope of enlightenment then they must rely on high levels of usability to justify the high investment in equipment and development that 3D user interfaces require. This will require improvements to novice usability, expert expressiveness, the minimization of fatigue and more flexible workflow scenarios.

4.2.1 Novice Use

The personal computer is a common denominator; most people in the developed world have access to one. Because access to 3D technology is more limited, novice users must be supported without requiring a significant learning curve. In video game design, novices are typically introduced to the application with a tutorial. This type of training is appropriate when the user has made a commitment to the application but is not appropriate for general applications. The best route to making 3D applications usable for novice users is to leverage their

existing skill set. For simulation applications, the skill set used in the physical world is appropriate. However, the existing skill set of tools based on the desktop metaphor is a more appropriate starting point for the more symbolic and system level controls of augmentation applications. Personal computer users are familiar not only with traditional interface elements but also with the methods desktop interfaces use to provide important feedback. Techniques such as context sensitive help, grayed out buttons or menus and dialog boxes are still likely to deliver their message effectively in a 3D context.

4.2.2 Expert Use

Jay Bolter and Larry Hodges argued in a 1995 paper that 3D user interfaces will have to provide the same level of symbolic manipulation as desktop interfaces if they are ever to find broad adoption (64). Expert users and novice users alike need explicit and discrete system control. This requires access to a rich set of system level functions similar to those provided by desktop systems. This access will require organizational tools analogous to desktop menus that organize a large number of available functions into a compact and unobtrusive data structure. Techniques such as gesture and voice interfaces that allow experts to operate more effectively in 3D environments are certainly of value. Yet, as is the case on desktop systems, these shortcut functions should not supersede more structured input methods such as menus that remain usable when more expert level functions are unfamiliar or forgotten.

4.2.3 Fatigue

Three dimensional user interfaces for augmentation will never gain broad acceptance if they continue to assume that users will expend more effort to use them than other competing technologies. Images of users waving their arms in the air or bending to reach for items are simply unrealistic when compared to desktop experiences. Even though it is counterintuitive to our normal experience in 3D space, the physical motion and the fatigue that comes with it must be reduced to a minimum. This does not preclude the use of more physically intuitive techniques, but it does mean that they cannot be the sole method of interaction with an application if anyone is to be expected to use that application for a significant period of time. Not all techniques that use high DOF input devices or even whole hand input need be physically taxing. The set of gestures that can be mapped to the human hand alone is likely enough to accommodate expert needs while keeping fatigue at a minimum.

4.2.4 Workflow

Providing a sophisticated visualization environment does not justify the significant expenses of and effort to use fully immersive projection environments. These applications need to provide significant levels of user functionality within those immersive settings if they are to be part of a more efficient workflow scenario. Access to a significant subset of the functions available on the desktop allows users to remain productive while taking advantage of the benefits of immersion. These applications should provide a level of consistency and portability with desktop systems that blurs what is now a rather distinct line between the two environments. This level of consistency not only applies to the appearance and functionality of interfaces but also to the interaction metaphors that they use. There are a number of intermediate states such as desktop stereoscopic display systems, large-format displays, multiple-display environments and even projector-based augmented reality environments that are effectively transitional points between the desktop and full head-tracked immersion. A consistent interaction methodology between platforms allows users to concentrate more on the benefits that each brings rather than the idiosyncrasies of their use.

4.3 Efficiency

The often conflicted goals of immersion and productivity have prevented 3D user interfaces from fully embracing efficiency. Three dimensional user interfaces for augmentation must be prepared to dismiss with aspects of typical interaction with the physical world when they threaten to interfere with the efficiency with which users accomplish their tasks.

4.3.1 First Person Perspective

The main utility of first person perspective is for the simulation of egocentric space and the exocentric inspection of 3D structure. First person perspective in egocentric settings should be employed when its benefit to the user is obvious. Applications where the end result will be viewed in first person egocentric conditions, such as architecture, urban planning and virtual world design, have obvious uses for first person perspective viewpoints. Moreover, augmented reality applications appropriately use first person perspective to register digital information directly within the viewpoint of the user.

Exocentric viewing conditions are a powerful way for users to gain a holistic appreciation of 3D structures. For gaining an understanding of overall structural features, the objects involved should be scaled appropriately

such that the user can manipulate the structure with the hands and move their head a significant distance around the object. For more detailed structural features and their interrelationship, it is also appropriate to scale structures up to human size and fix it with relation to the 3D space. Users physically moving around a structure benefit from the positional consistency of that structure with respect to their bodies. An excellent example of this use is the CRUMBS application for protein folding from NCSA (65). Users are able to visualize volume renderings of proteins and place “crumbs” along a path through the structure that helps them analyze how the proteins have folded. In this context, virtual travel techniques only serve to disrupt the physical relationship between user and data.

4.3.2 Third Person Perspective

Augmentation applications should not force users to accomplish tasks in first person that are better suited for other representations. When a task requires that the user appreciate relationships between objects, a spatially distributed parallel perspective viewpoint is almost always more appropriate than a depth compressed first person perspective. Alternate viewpoints also give the user the flexibility to overcome object occlusions and distance problems that make object selections difficult. Users should have the flexibility of manipulating alternate viewpoints as is commonly done on the desktop. This is not an unreasonable requirement since any augmented or virtual reality application requires its own model of the surrounding environment.

Because of the potential for confusion, alternate viewpoints should be initialized to the current first person viewpoint when possible to help keep the user aware of their context. Alternate viewpoints should also make use of frames, filters or other design features in order to disambiguate them from the surrounding content and avoid confusion. The management of third person viewpoints should be object-centric instead of user-centric. Viewpoints that orbit objects or pan along planes of interest are more easily manipulated from a third person perspective. These techniques are commonly employed for the management of viewpoints on desktop systems. Viewpoints that pan in a parallel perspective view or rotate around a focal center are more appropriate for maintaining perspective on specific content while those that employ first person techniques (i.e. point-and-go and driving metaphors) are more appropriate for undirected travel or wayfinding through spaces.

4.3.3 Input Methods

A lack of haptics in most 6 DOF input devices means that opportunities to use them should be considered very carefully. Multiple DOF devices are best employed when the relationship between control and display is isomorphic and proprioception can be maximized. Exocentric manipulation near the body is an example of one such case. In cases where this is not possible, techniques that use a user-centric coordinate system are more appropriate. The HOMER technique is a good example of how non-isomorphic interactions work best when registered to the user perspective. Hand motions that move up, down, left, right, in and out with respect to the user are a good way to concurrently control multiple variables that are may not be mapped directly to the physical motion of an object.

Depth compression and a lack of haptics do cause problems for object manipulations with 6 DOF input devices requiring any significant accuracy. For optimal control, object manipulations should be coupled with optimized viewpoints of the object. For manipulations at a distance from the user, a viewpoint that is appropriate to the task should allow the object to be viewed in its original context. Techniques that disembody objects from their original context discard information that may be important to the manipulation task. As is the case with desktop systems, when alternate viewpoints are available, they also present a good opportunity to constrain object motion. For example, a top viewpoint of an object is a good opportunity to restrict manipulation to the horizontal plane.

4.3.4 Symbolic Manipulations

Interfaces that manipulate symbolic data need constraints to work effectively regardless of whether they are 3D or 2D. This means that sliders, buttons and other constrained inputs remain useful in 3D. When an interface is close to the user, more physical metaphors may be appropriate such as pulling a lever or turning a dial. However, as interfaces move away from the body and the ability to interpret and manipulate in depth falls away, the opportunity for natural 3D interaction diminishes. These interfaces are only made less efficient by introducing unnecessary degrees of freedom to their use. Virtual hand techniques that add a depth requirement to what is already a 2D interface should be avoided in favor of those such as ray-casting or image-plane that effectively eliminate depth.

Traditional interface elements are likely to find a place in future augmentation interfaces. Organizing interface elements along depth is unlikely to be successful except in instances where interface elements are truly exocentric. When presented at any significant distance, it is hard to imagine that another construct will supersede the spatial information density that 2D interface elements routinely deliver. The flat surface of desktop interface elements naturally translates to a spherical surface in first person perspective. However, using a flat design may prove a useful compromise given the pseudo-haptic utility of flat surfaces and their frequent presence in our environment. The long history of design practice and human interaction research is a good opportunity to immediately introduce well designed interface elements into 3D space. Another argument in favor of traditional interface elements is the consistency they provide. Forcing users to transition from a flat file layout on the desktop to one that presents elements in on a 3D sphere is hard to justify.

4.4 Application Development

In the context of application development, introducing significantly different interaction methods increases the difficulty of programming for both environments. Interaction devices that rely on high DOF input devices also run the risk of limiting the range of users that can effectively use them. The adoption of useful ideas such as canonical methods, single-authoring and universal design points to the need for a holistic framework for both 2D and 3D user interfaces.

4.4.1 Canonical Methods

In the general context of 3D user interfaces, it is often argued that interaction schemes and interaction devices should be chosen based on a case-by-case basis. However, such an approach will never lead to the broad adoption of 3D user interfaces by the general public. Just as the desktop computer has done, 3D technology needs a canonical hardware setup, a canonical input device and canonical methods of interacting with applications. This consistency is the key to creating an environment where other parties can begin to produce content for the medium. The stability of video game consoles and their associated hardware is a critical part of the environment that allows companies to make significant software investments for those platforms. Without canonical methods, both the personal computer and video game console would never have succeeded. Canonical methods need not close the door to innovation. Just as right clicking and scrolling buttons on the mouse have evolved, useful improvements to usability can still develop within their context.

4.4.2 Single Authoring

The proliferation of different computing devices and their attendant hardware configurations puts a strain on application developers. Increasingly, developers are using techniques that abstract applications and interfaces in ways that allow their rapid deployment to multiple hardware configurations. Three dimensional user interfaces should not be excluded from efforts to reduce the development time and effort by employing single authoring solutions. Traditional 2D applications should be usable in 3D environments without redevelopment. And, since applications routinely handle 3D data on the desktop, methodologies should be established that allow natively immersive applications to be used on desktop computers without significant redevelopment. The possibility of reusing content in both desktop and immersive settings is also another argument in favor of using traditional interface design elements.

4.4.3 Universal Design

There is a movement towards designing user interfaces so that they can be used by a broader segment of the population. This movement stands in sharp contrast to accessibility tools that allow those with less ability to use tools designed for the fully able. The addition of depth to what are effectively 2D interfaces reduces the chances that low-mobility users can use them. Universal design is not incongruent with good design. When possible, unnecessary degrees of freedom should be eliminated. One easy way to make future 3D interfaces more accessible is to eliminate the fine motor control associated with multiple DOF devices as a requirement for their effective use. Universal design does not preclude the use of 6 DOF input as long as other input methods that are less reliant on multiple DOF control are a default feature. Tablet based interfaces such as those used by Bowman are one example of how virtual travel can be reduced from a complex multidimensional task to one in only two dimensions.

4.4.4 Interface Management

Despite the increase in flexibility associated with 3D space, augmented and virtual reality user interfaces typically have far less flexibility than those used on the desktop. Desktop user interfaces can typically be resized, repositioned, reprioritized and even reorganized, in the case of web-browser based interfaces. Constructs analogous to those available on desktop systems are needed in 3D environments where the need to effectively manage display real-estate is even more important. Not only are the objects surrounding the user likely to play a part in computational tasks but obscuring the surroundings may pose serious safety risks to the user. Similar to

the static presentation of information on maps, there is active research into techniques for automatically reorganizing information with respect to the content displayed beyond it (66). However, as on the desktop, the priority of information and its relevance to the task at hand is often impossible to predict by artificial means.

4.4.5 A Unifying Framework

When effectively separated from simulation applications, the close relationship between the requirements of augmentation applications and existing desktop applications begins to appear. The guidelines presented here have a consistent theme that many desktop concepts retain their appropriateness even in the context of 3D user interfaces for augmentation. The somewhat counterintuitive notion of reducing fatigue by avoiding real-world interactions in 3D user interfaces also supports this theme. Single-authoring, canonical methods and universal design all encourage a consistency of methodology between desktop and immersive interaction. A holistic framework that embraces these guidelines to enable efficient transitions between desktop and immersion is needed.

Experience has suggested that the many different physical and hardware configurations of 3D technology require that user interface decisions be made on an individual case. However, the same thing could easily be said about 2D user interfaces in the context of the many configurations they inhabit. Despite the need for individuated designs for handheld devices, information kiosks and home electronics, 2D user interfaces still benefit from a consistent methodology and input scheme where it is most important; on the ubiquitous desktop. Given a similar unified framework, the domain of 3D user interfaces for augmentation can also be an environment where productive software that augments human computation can proliferate.

5. THE WITHINDOWS FRAMEWORK

The guidelines developed in the prior chapter do not provide a specific framework for meeting the goals of productive augmented and virtual environment applications. The Withindows framework is a concrete strategy that realizes those guidelines. This chapter will introduce the principles of the Withindows framework and show how they satisfy the guidelines for developing productive augmented and virtual interfaces. After a brief chapter overview, the following sections will discuss the two principle components of the framework, image-plane selection and through-the-lens techniques. The last sections will then show how this combination creates the conditions for usable, efficient and easily developed immersive applications.

The most important part of developing a framework for the development of productive augmented reality interfaces lies in the acknowledgement that 3D interfaces do not need to use 3D concepts, they merely need to operate in a 3D context. Existing desktop interfaces already compensate for a lack of haptic interaction by incorporating useful constraints that allow an ordinary mouse to accurately control a great number of system control variables with a relatively small set of interface widgets. Simply using these interfaces within a 3D environment is not enough to satisfy the needs of coming augmented reality applications. The most important part of any successful framework is a smooth transition from existing desktop functionality into the primary tasks associated with exploring and manipulating 3D space. The primary tasks of selection at a distance, manipulation at a distance, global search, and travel, in the case of virtual reality, must be optimized in this framework if it is to have any utility.

The Withindows framework is built around two important techniques. The first technique is image-plane selection. The typical desktop environment is actually a special case of the more general image-plane selection technique. This formulation gives image-plane selection an advantage when transitioning between desktop and immersive interaction. Because of its natural relationship to desktop interaction, analogous interactions are easily created within an immersive setting. Image-plane also has a number of advantages over other selection techniques such as ray-casting in immersive environments.

In addition to facilitating transitions between usage scenarios, any successful framework has to incorporate the tasks normally associated with 3D interaction. The primary tasks of selecting objects, manipulating objects, searching the global environment and traveling to other locations must be optimized by this framework. The key to optimizing these tasks is decoupling the visualization part of the task from the limitations of 1st person perspective. The Withindows framework uses a class of techniques called through-the-lens techniques to manage alternate viewpoints within windows controlled by the user. Primary immersive interaction tasks can be accomplished completely within these windows in an optimized manner. In addition, a number of novel interaction techniques utilize these windows and 6 DOF interactions to create the opportunity for increased efficiency.

A critical part of increasing efficiency involves minimizing the fatigue of working in immersive environments. The key to reducing fatigue is avoiding the typical reliance on direct manipulation with objects at a distance from the user. Encapsulating the primary immersive tasks within a window allows the user to avoid fatiguing selection, manipulation and global search by working primarily on windows located in a comfortable position below their hands. In combination with image-plane techniques, these windows can emulate the highly constrained and efficient selection and manipulation techniques commonly found on workstation computers.

The tight integration with workstation techniques is the final part of the Withindows framework that helps it meet application development guidelines that foster broad adoption. The direct analog of image-plane selection with desktop interaction means that existing applications can be ported to immersive settings without any redevelopment effort. Furthermore, the reliance on alternate viewpoint windows creates immersive applications that are analogous to those currently used on the desktop for creating and managing 3D content. This means that immersive applications can be ported to desktop environments with no redevelopment effort.

5.1 Image-plane selection

Introduced in the second chapter, generalized image-plane selection is an object selection technique that reduces the selection problem to one of occluding a visible object with either the finger or some other virtual or real object. Image-plane selection effectively reduces the selection problem to one that is purely two dimensional within the plane of the image presented to the user. For a user in the physical world, the image-plane is defined as what they can see from any single position and orientation of their head. For users of head mounted displays the image presented on the LCD is also the image-plane that the user sees at any given time. In projection based

systems such as the CAVE, the term image-plane is more ambiguous. There may be as many as 6 different display surfaces being rendered in order to surround the user in visual content. Because these display surfaces are the focal point of our visual accommodation, they are sometimes referred to as the image-plane. For the purposes of this document, as in the physical world, I will refer to the image formed by these display surfaces within the user field of view as the image-plane.

In practice, the image-plane selection technique is implemented by casting a ray from the viewing position of the user through a single point on the hand of the user. The first selectable object that is found along the ray is chosen and all objects behind it are ignored. The point controlled by the hand, which will be referred to as the virtual cursor, is often a fingertip in physical or augmented reality. In a virtual reality, the virtual cursor is usually a part of the virtual hand or some other virtual object moving with the hand. Because the depth of the virtual cursor is of no influence, image-plane selection can be considered a two dimensional interaction technique that only requires the specification of 2 points in a spherical coordinate system around the user. Because traditional ray-casting from the wrist is implemented in a similar fashion, it is also considered a two dimensional task when objects are sufficiently far from the user.

It should be obvious that occluding an object with the finger is an ambiguous way of selecting anything but larger objects in a stereo environment (Figure 3). When focused on an object at a distance away from the finger, the finger will be covering a different part of the image-plane in each eye. This is easily demonstrated by alternately closing the left and right eyes. Because of this ambiguity, the majority of image-plane implementations have used a monoscopic setup that presents the same image in each eye of the user (20). The desktop WIMP interface is one such monoscopic implementation of image-plane selection.

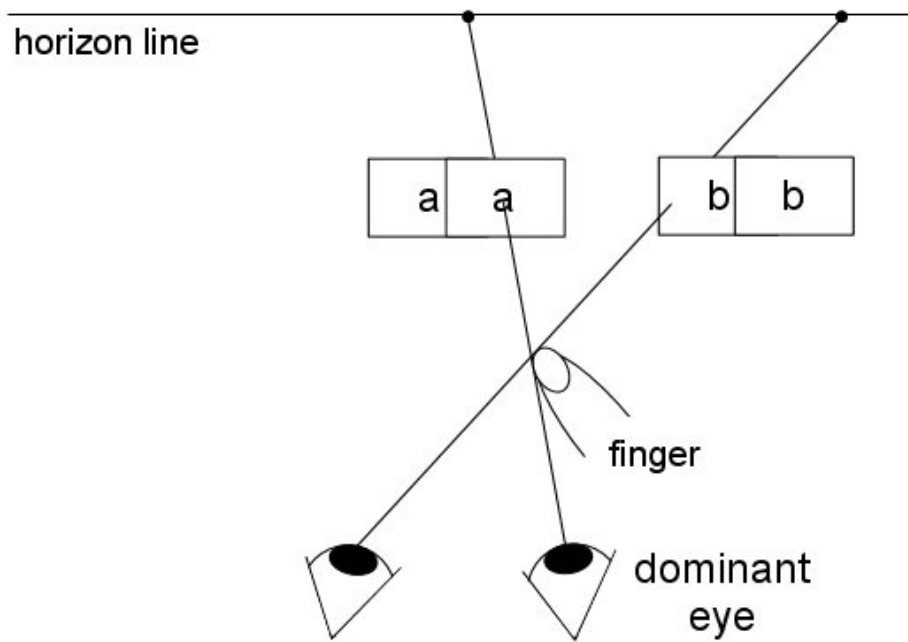


Figure 3. Selection of distant objects by occlusion is ambiguous with stereo vision.

5.1.1 Image-plane and the Desktop

The typical desktop Windows, Icons, Menus, and Pointer (WIMP) interface is a special case of image-plane selection. Although not a true three dimensional representation of a desktop, common operating systems such as Microsoft Windows and the XWindows system typically display windows in a depth ordered stack with the currently active window on top. The mouse cursor in these systems almost always remains visible in front of the desktop content. If we consider the mouse as residing somewhere in between the user viewpoint and the desktop windows, then we can see that selecting the first visible object with a ray cast from the viewpoint through the tip of the cursor emulates the familiar behavior of the mouse. The desktop situation is a simplified case of image-plane selection where the display is monoscopic, the user viewpoint remains fixed and the virtual cursor is restricted to moving in a plane a fixed distance between the viewpoint and desktop content (Figure 4).

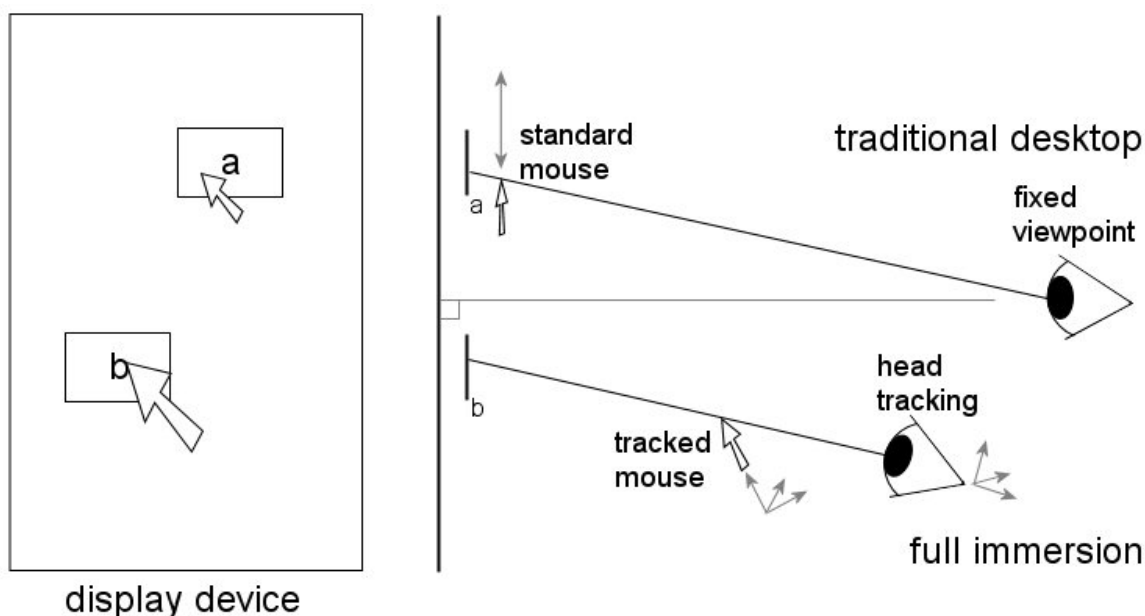


Figure 4. The typical desktop is a special case of image-plane selection where viewpoint is fixed and the mouse is constrained to a plane.

One important aspect of the desktop setup is the projection matrix applied to the scene. If we think of the desktop content as residing completely within a single plane, then the cursor need only be placed a small distance closer to the user in order to always remain in front of the content. If the cursor itself is a flat object, then such a setup could be rendered using the 1st person perspective projection we see in the physical world with no noticeable distortion of the cursor. However, in practice the desktop situation is completely absent such distortions and would require that the user viewpoint be rendered using a parallel perspective. Using a parallel perspective rendering is a more appropriate choice because it allows each desktop window to reside at a different depth behind the cursor without perspective distortion affecting its size.

5.1.1.1 Stereoscopic Use

If image-plane selection is to find any use in more realistic display settings then it must be usable in a stereoscopic display mode. One solution to the ambiguity problem with stereo displays is to use a virtual cursor that is located at the depth of the selected object. Desktop stereo displays have traditionally been a combination of monoscopic interface elements (i.e. windows, icons, mouse) and stereo content contained within windows. Because these systems have usually been focused on stereo display rather than stereo selection, they have typically used a monoscopic cursor that resides at the depth of the desktop. This type of implementation has been

easily accomplished because the standard operating system application programming interface (API) provides no facility for rendering the mouse at a specified depth. This solution leads to some stereo confusion when the cursor remains at a distance when moved over content that appears in front of or in back of the screen.

Another solution to the stereo display problem that has recently been demonstrated involves rendering the cursor at the same depth as the content underneath it (67). In a desktop stereo display this means that the cursor remains fixed at the desktop depth until it is moved onto a stereo rendered object. The depth of the mouse is then fixed to the selection point on the 3D object. This way the cursor appears over the same selection point on the object in both eyes. For an object, such as a sphere, the mouse would appear to move off of the scene towards the user. For more discontinuous objects, such as a cube, the depth transition would be more abrupt. Since the depths involved on a desktop stereo display are relatively small and the rest of the desktop content resides at a fixed depth in front of the user, this kind of setup is unlikely to create significant confusion for the user. However, in a more unrestricted situation such as an augmented or virtual reality environment the same constraints do not apply.

The closest analog to a desktop in immersive environments is the floor or the surrounding background. Placing the depth of the virtual cursor at the actual depth of these elements would cause the depth of the cursor to move between two extreme depths as the user moves it from the horizon to just in front of their feet. This extreme fluctuation in cursor depth is likely to provide visual confusion and even eye strain as the cursor drops off to a depth at the horizon as it is moved between nearby objects. A compromise solution that places the cursor at a constant depth somewhere in between the user and the horizon when traveling between objects would appear like an appropriate solution. However, the depths involved between objects in augmented and virtual environments can also be as extreme as those in everyday life, with objects of interest residing both close to the user and at great distances. As a result, such a compromise is likely to be distracting to the user.

5.1.1.2 Dominant Eye Cursor

Another solution to the ambiguity problem is to place the virtual cursor only in the dominant eye of the user when moving between selectable objects (Figure 5). In the physical world we routinely use our dominant eye to resolve the same ambiguities created by stereo viewing. Most people are unaware that even the simple task of pointing at a distant object is modulated by our chosen dominant eye. This can easily be demonstrated by

pointing at a distant object and then alternately opening and closing each eye. Other tasks, such as archery, firearms and billiards routinely require us to either ignore our non-dominant eye or close it altogether. The human brain only uses the dominant eye when an ambiguity is presented. Either the dominant or non-dominant eye can be closed during aiming tasks with equivalent results. The Center Axis Relock (CAR) system, taught by law enforcement agencies throughout the world, is a firing technique that relies on our natural ability to use a sight presented strictly in the non-dominant eye⁴. The CAR system trains law enforcement personnel to turn their dominant eye away from the target so that its view down the gun sight is blocked by their nose. This firing technique allows the shooter to accurately sight their gun in the non-dominant eye without having to close either eye.

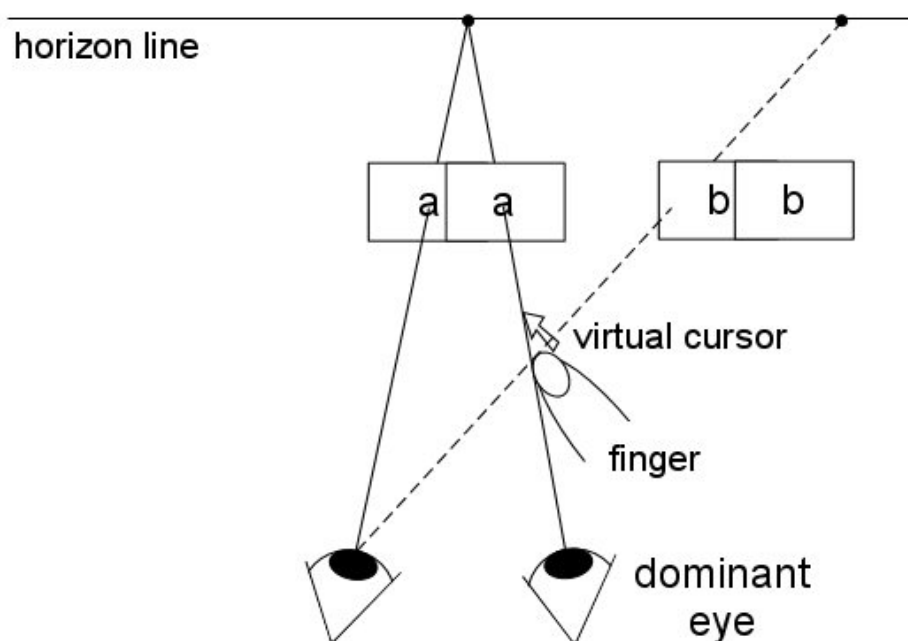


Figure 5. Placing a virtual cursor exclusively in the dominant eye resolves ambiguities associated with image-plane selection in stereo environments.

Our ability to control the display rendering delivered to each eye in stereo display environments presents an opportunity to apply these principles to image-plane selection tasks. Using a desktop stereo display, Ware conducted a study to verify that users would be able to select objects floating at various depths in front of them

⁴ Sabre Tactical Training Resource and Research, Paul Castle, <http://www.sabretactical.com>

faster using an image-plane technique than a virtual hand technique (68). The most obvious reason that users were faster using the image-plane technique was that they had to move a smaller distance to select objects at a different depth. The study used a cursor presented in both eyes for the virtual hand treatment and a cursor presented solely in the dominant eye for the image-plane treatment. Ware reported that the dominant eye cursor was well tolerated by his subjects and found no indication that it prevented them from performing the task efficiently.

5.1.1.3 Reinforcing Cursor

While placing the virtual cursor exclusively in a single eye resolves the ambiguity problem, there is still a possibility that prolonged use will cause the user to experience eye strain. A compromise solution is to reinforce the depth of the object being selected by presenting the cursor at object depth in both eyes and then drop the non-dominant eye cursor during transitions between selectable objects (Figure 6). When objects are being selected the user is most likely focused at the depth of the object and should have little problem with the appearance of the cursor in the non-dominant eye. The dominant eye cursor does not require any change to its size or position in order to make it appear at the depth of the selected object. Contemporary graphical interfaces routinely change the color of targets to indicate that they are selectable and that the mouse is over them. The non-dominant eye cursor can also serve to reinforce object selection because it only appears when the cursor is over a selectable item. The rest of this document will refer to the non-dominant eye cursor as a reinforcing cursor.

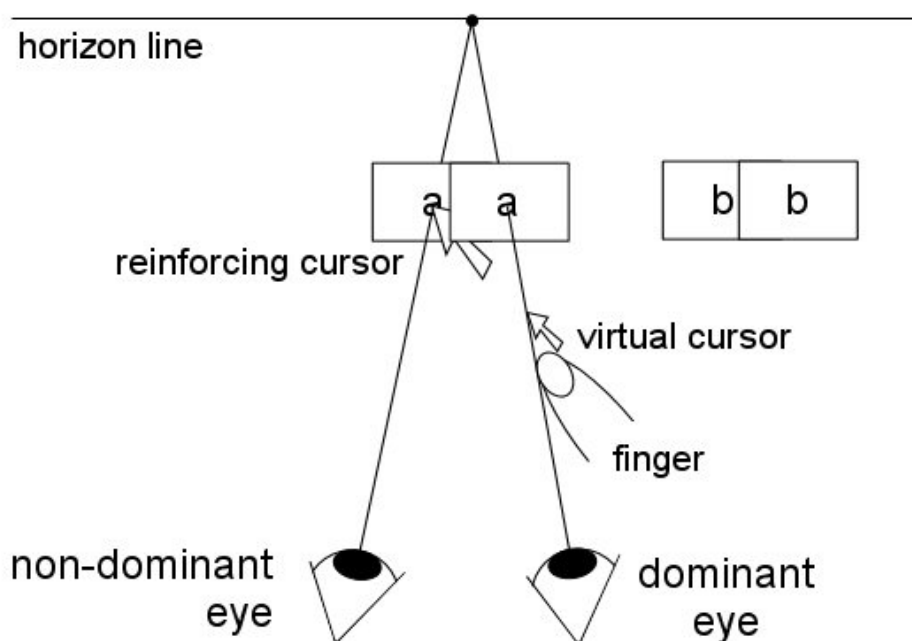


Figure 6. A reinforcing cursor can be displayed in the non-dominant eye to make the virtual cursor appear at the depth of the selected content.

5.1.2 The Advantages of Image-Plane

Now that we have a strategy for using image-plane selection in stereo display environments we can consider its utility as the primary means of selection in an augmented or virtual environment. Image-plane selection has a number of advantages over prior techniques that use either the virtual hand or a ray cast from the wrist. The advantages of depth insensitive techniques such as image-plane and ray-casting over virtual hand techniques have been discussed in prior chapters. Ray-casting is the most widely used technique for selection at a distance and its popularity is often attributed to its relative ease of use and low fatigue. However, as we will see, ray-casting is only easier to use and less fatiguing than image plane selection when used in constrained situations. When used in more generalized cases, image-plane selection is not only more efficient and less fatiguing but also has a number of advantages that make it ideal for interfaces that transition between desktop and immersive use.

5.1.2.1 A Canonical Input Scheme

Any successful framework needs a canonical input scheme that requires the minimum degrees of freedom necessary to achieve efficient input. The 2 degrees of freedom and 3 button input of the WIMP scheme are an excellent example of an input method that is powerful yet uses a minimum of inputs. The canonical input device for any canonical input scheme should work in a robust manner with a minimum of technical complexity. Again, the standard mouse is technologically simple and robust enough to be used in a number of settings and configurations. Image-plane selection is a natural choice for a canonical input scheme because it can be implemented efficiently with only 2 degrees of freedom and its input can be acquired in a manner that is more robust and efficient than that of ray-casting. Because image-plane selection is implemented ideally with the hands, a simple 3 DOF mouse is an appropriate canonical input device for immersive interaction.

On the desktop, the cursor within the image-plane is controlled by a mouse which is a 2 degree of freedom input device. In the more general case of immersive image-plane selection, the location of the virtual cursor is determined by the position of the hand in front of the user. In practice, image-plane selection is usually a 3 degree of freedom tracking problem because hand position is specified by the Cartesian coordinate system. However, this is somewhat misleading because the actual position of the hand will only be required in virtual environments to render the virtual hand. The only required input to the system is actually the 2 DOF location of the virtual cursor within the image-plane presented to the user. Although using the hands to position the cursor in the image-plane is a natural proprioceptive task, image-plane selection does not require their use to specify cursor position. The means by which image-plane input is prescribed is flexible enough to accommodate any number of 2 DOF input devices such as a mouse, a joystick, eye tracking or even brain-body interfaces such as those being developed for persons that have lost the use of their arms (69).

In theory, ray-casting from the wrist can also be fully specified by a 2 DOF device by fixing the position of the wrist and specifying the zenith and azimuth angle of the wrist angle. However, because it is a function of hand position and orientation, the transfer function between a ray cast from the wrist and the resulting object intersection is significantly more complex than that of image-plane selection. Ray-casting is highly dependent on proprioception when used at close distances. A 2 DOF input scheme for ray-casting is unlikely to prove an efficient input means because the technique relies so heavily on the familiar hand-eye coordination used in the physical world (70). Image-plane selection on the other hand, only involves occluding the desired object with the

virtual cursor in the image-plane. Image-plane selection, while also aided by proprioception, relies more heavily on the visual feedback provided within the image-plane itself.

In practice, ray-casting requires a 6 DOF input device to specify the position and orientation of the hand. As a tracking problem, determining the position of an object in space is less difficult than acquiring the additional orientation information about that object. The position of three points is usually required to determine orientation in tracking systems that use either visual tracking or some other form of point triangulation. Because the only required input to an image-plane setup is the location of the virtual cursor within the image-plane, this information can be easily captured by a single video camera located at the viewpoint of the dominant eye. Determining the location of a point on the hand is a relatively simple computer vision task that can be handled by modest mobile computing devices. This is best exemplified by the Tinmuth system for outdoor augmented reality modeling (59). In this system, the location of the virtual cursor is determined by tracking the location of a fiducial marker within the image of the camera capturing the environment. When using a camera based approach is not appropriate, as in the case of a CAVE setup, the 2 DOF projection onto the image-plane of a 3 DOF tracked point becomes the logical approach. Regardless of whether it is derived from a 3 DOF point in space, captured directly within the image-plane or directly input by a 2 DOF device, image-plane selection remains effectively a 2 DOF input technique.

The minimal degrees of freedom associated with image-plane selection make it a favorable choice for a canonical input technique. The preferred input device in fully tracked immersive environments then becomes a single tracked point attached to the hand of the user. Selection and other tasks frequently need discrete inputs like those provided by buttons. While the means of capturing this information are flexible, a simple 3 DOF mouse appears as a natural choice for a canonical image-plane input device.

5.1.2.2 Remains Isomorphic

The nature of the transfer function between input and output is a significant factor in making image-plane a superior selection technique to ray-casting. A control-display transfer function remains isomorphic when the change in the output display is consistent with the change in input control. Isomorphism helps the user to coordinate their input in a consistent manner. A good example of non-isomorphism is the relationship between the steering wheel and the direction of travel when driving in reverse. From the viewpoint of the driver, turning the

wheel to the right appears to send the car to the left. Some level of non-isomorphism can easily be tolerated. The desktop mouse is a non-isomorphic device, yet the simple relationship between forward and up is easily tolerated. The isomorphic nature of image-plane selection is one of its strongest attributes.

The reason that image-plane selection is isomorphic is quite obvious. Because the resulting selection point remains under the hand or virtual cursor, there is always a one-to-one relationship between hand motion in the image-plane and the resulting location of the selection point. The direct relationship between control action and display result allows the user to accurately control the trajectory of the selection point. Furthermore, the occlusion of the object underneath the cursor visually reinforces the selection task. By contrast, ray-casting relies on the ability of the user to determine the intersection point between the ray and an object. When objects become smaller and less differentiated by their surface properties, the ability of users to discern a ray intersection is diminished. Although these selections can be reinforced by highlighting and other means, it has been shown that ray-casting users rely primarily on intersection for their visual feedback (71).

5.1.2.3 Robust to Affine Distortions

The complexity of ray-casting becomes a problem when it is used on flat surfaces that are not parallel to the image-plane of the user. Such surfaces are known as oblique or affine distorted surfaces. Widgets confined to a plane, such as sliders and cascading menus, are a hallmark of desktop interfaces and have frequently been used in virtual environments. When a ray is moved along a surface at an oblique angle to the user the resulting motion of the intersection point on that surface moves in a non-linear manner. This is best illustrated when walking with a flashlight in the dark. As the floor moves away from the user the ratio between hand orientation and the resulting location that is illuminated increases non-linearly. This kind of non-linear behavior is called a control-display non-linearity and is more likely to cause usage problems than a linear control-display ratio.

There are situations where control-display non-linearities do not introduce significant problems. In the case of the Go-Go technique, a non-linear relationship can be tolerated because the non-linearity is only along the depth axis and remains body-centric. In the case of rotating objects with the hands, a non-linear relationship between the orientation of the object and the hand position can help overcome the angle limitations of the human wrist without introducing significant control problems (72). However, in the case of controlling the selection point on an oblique surface, ray-casting presents a more significant control problem because the non-linearity is a

complex function of hand position and surface orientation. In contrast, oblique surfaces have significantly less effect on the control-display relationship when image-plane selection is used. Consider the case of a slider widget that is positioned on the floor in front of the user. Even though the scale may appear distorted by distance, moving the slider to the desired location only requires moving the hands over that location. A ray-casting user must deal also with the non-linear control-display ratio of the intersection point on the slider.

Ray-casting is a good technique for making gross movements between destinations. The majority of ray-casting studies that involve moving objects between locations have paid little attention to the path followed between the origin and the destination. The non-linear nature of ray-casting presents problems when the task requires that the cursor follow a constrained motion path. Cascading menus are an example of a situation where the path followed between origin and destination is critical. Path following on cascaded menus is already a difficult task that has motivated a significant amount of research on desktop systems (73). Cascading menus floating in an immersive environment present even more usability challenges. Even if a world-fixed menu is initially oriented directly towards the user, as submenus move away to the side the effective surface of the menu becomes affine distorted. With the advent of projection based augmented reality and the utility of pseudo-haptic feedback, it is likely that future interfaces will make use of oblique surfaces located within the physical environment. The isomorphic nature of image-plane selection has a good chance of improving the usability of widgets confined to these affine distorted surfaces.

5.1.2.4 Insensitive to Depth

The more complex control-display transfer function of ray-casting does act isomorphic under certain conditions. When objects are all sufficiently far away from the user, the motion of the hand and the resulting object selections are almost directly related and ray-casting becomes effectively a 2D selection task (21). However, when the distribution of objects near and far is more disparate, the three dimensional task complexity of intersecting the ray with objects increases. In these more realistic environments ray-casting becomes a truly 3D task because the user must also consider selection depth. Image-plane selection, on the other hand, remains a 2D task regardless of the depth of objects in the scene. Recent studies have shown that image-plane selection times are significantly faster than ray-casting in environments where objects are distributed evenly in depth (74).

Two dimensional interface widgets such as menus and icons are often used for 3D user interfaces. When located in 3D environments, these interface elements may change distance and angle with respect to the user viewpoint. Regardless of its distance from the user, if an interface widget has the same appearance to the user then its control-display ratio remains a constant when using image-plane selection. By contrast, two interface widgets located at different distances may appear the same to the user but require significantly different control actions to achieve the same display result. This effect increases as the distance between user and user interface elements decreases (Figure 7).

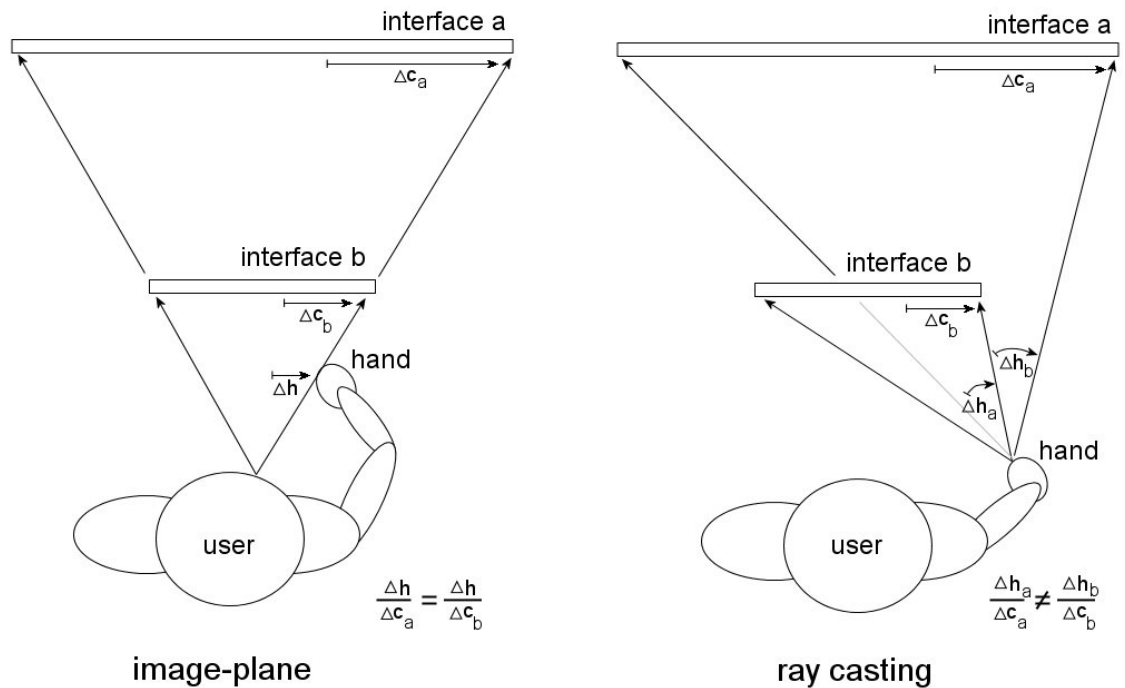


Figure 7. The control-display ratio between hand motion Δh and cursor movement Δc changes with distance for ray-casting but not for image-plane selection.

Another problem associated with ray-casting at close distances is the necessity of a stereo display. Because of its heavy reliance on proprioceptive spatial cues, the usability of ray-casting is strongly correlated with distance when using a monoscopic display. When objects are located a significant distance from the user, the two dimensional nature of the task allows it to be used effectively without stereo. However, when trying to select objects closer to their body, subjects become unable to use ray-casting effectively without stereo (36). The

usability of image-plane selection under both stereo and mono display conditions contributes to its most important quality, the ability to transition smoothly from desktop to immersive environments.

5.1.2.5 Smooth Transitions to Ray-Casting or Touch

Irrespective of the theoretical advantages of image-plane selection, it is not the first choice of all users. Because it requires more arm motion than ray-casting, experimental settings that involve object selection on distant objects have usually lead to a preference for ray-casting (75). In studies where objects are at more varied depths closer to the user, Wingrave has found an almost even preference for either technique (74). As the study results presented in chapter 6 demonstrate, each technique has benefits in different settings. Ray-casting is faster for selecting relatively large objects that are all at the same distance from the user. However, when accuracy at a distance is required, research has shown that hand unsteadiness prevents pointing based techniques from achieving high accuracy (76). Image-plane techniques, on the other hand, are more accurate at a distance and faster when objects are at various depths. For these reasons, it may be desirable to accomodate hybrid strategies that allow a transition between image-plane and ray-casting techniques.

Transitioning into ray-casting from image-plane is easily accomplished by aligning the initial ray down the sight line of the hand. Once the transition has been initiated, changing the orientation of the hand slowly reveals the ray that was collinear with the virtual cursor view. Making the transition from ray-casting into image-plane cannot be accomplished as smoothly. The intersection of the ray is unlikely to be near the hand in the image-plane since ray-casting is most frequently used with the hands outside the viewer line of sight. As a result, a switch into image-plane selection will cause a sudden change in selection location and is likely to confuse the user. One option is to animate the reorientation of the ray into a collinear angle with the hand sightline, but the time consumed during such a transition is likely to motivate users to forgo the transition altogether. Once the interaction using the alternate technique has been completed, the transition back into the primary technique is less likely to be disruptive to the user.

The orientation of the ray extending from the hand is also a reason that image-plane is a superior technique when transitioning into touch. Both image-plane and ray-casting devolve into touch when they are used to interact with a physical surface. A ray-casting implementation is more likely to cause confusion because of the rapidly changing nature of the ray intersection point. Even if the ray is emitted directly from the finger, as it

approaches the physical surface, the intersection point will be changing rapidly unless the ray remains pointed directly at the eventual touch location (Figure 8). By contrast, the relationship between the virtual cursor and the location touched on the physical surface is more likely to remain stable with an image-plane selection technique. As a result, the user can focus on the trajectory of their finger without visual interference from an unstable selection point.

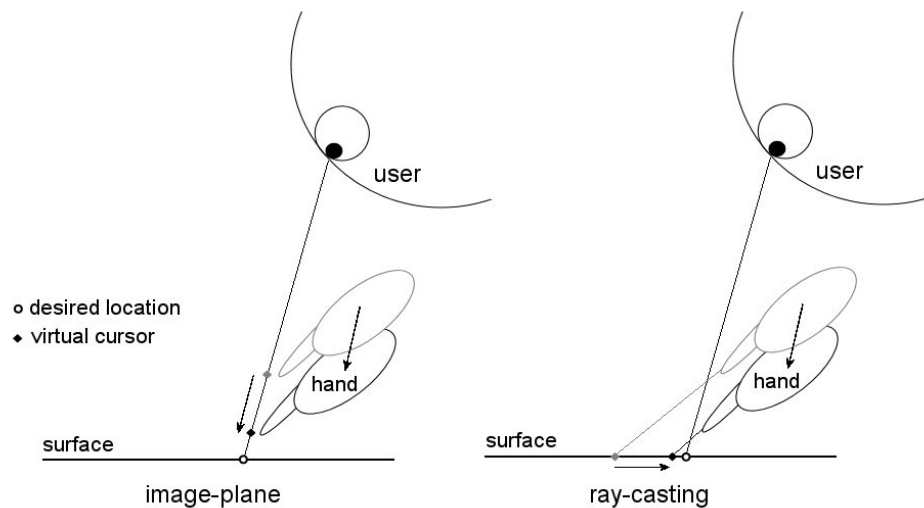


Figure 8. The virtual cursor is more likely to move relative to the desired selection location when transitioning to touch using ray-casting.

5.1.3 Transitional Configurations

Developing productive and efficient interfaces for the augmented reality of the future requires that applications, knowledge and experience generated on desktop systems be able to transition smoothly into augmented and virtual realities. This transition not only concerns moving into fully immersive settings but also must accommodate transitional setups that include any combination of the ingredients of full immersion. The direct relationship between image-plane selection and the WIMP interface makes it ideally suited to making these transitions. With image-plane concepts, the main ingredients of stereo display, 6 DOF or 3 DOF tracked input devices and full head tracking combine to easily transition between best in practice implementations in multiple settings. Traditional desktops, large format displays, multiple display setups and projector based augmented reality systems can all be described in terms of image-plane selection. Image-plane concepts not only facilitate

transitions in between these transitional configurations but can also bring new insight into ways to improve their usability and flexibility.

5.1.3.1 The Traditional Desktop

The relationship between image-plane techniques and desktop stereo has already been introduced. Stereo display can be added to desktop systems by either fixing the cursor depth to the desktop background or moving it to the depth of the stereo content it resides above. Naively changing the depth of content to match that of stereoscopic content below it presents a problem when the content is viewed from an oblique angle. When the content is viewed straight on, the control-display relationship between mouse input and resulting mouse position in the image-plane remains linear (Figure 9). However, when viewed from an off angle, a cursor that is changing depth will move across the image-plane of the user in a non-linear fashion. To date, researchers that have implemented this strategy did not address this contingency because they were using an autostereoscopic display system that only provided a small viewpoint range in front of the screen (67). The proper solution to maintaining isomorphic cursor motion involves utilizing the properties of image-plane selection.

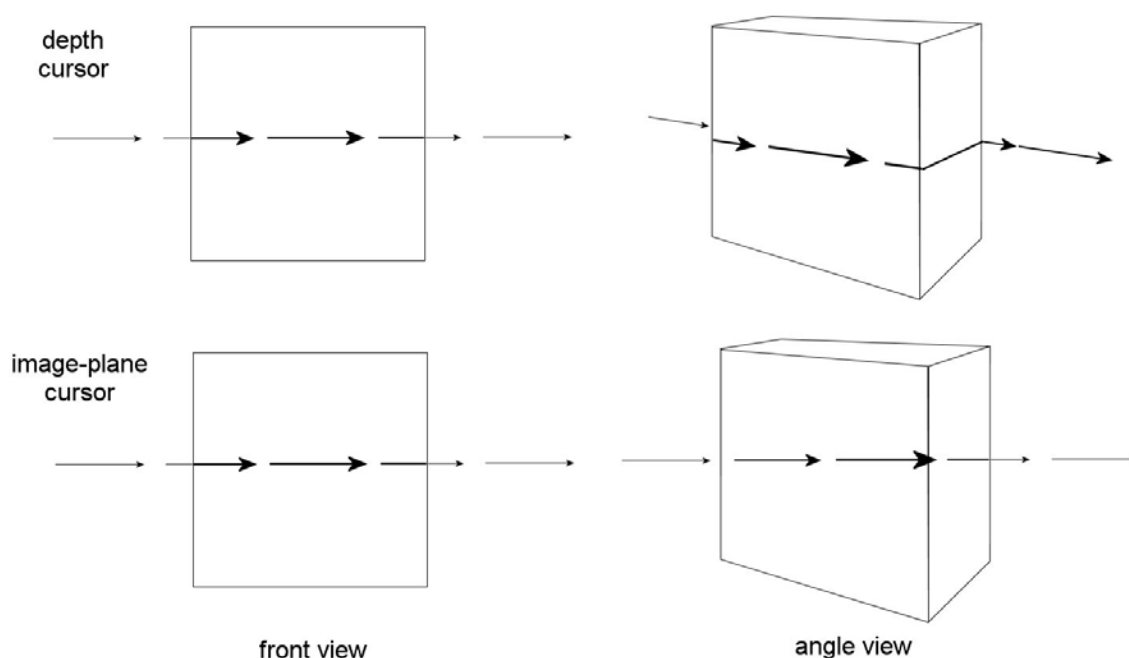


Figure 9. A standard mouse placed at content depth viewed at oblique angles has non-linear movement while a reinforced dominant eye cursor remains linear.

A second option for desktop use involves moving the cursor movement plane to the depth of the object under the cursor to stabilize it during head motions. While this strategy would likely appear consistent to the user during head motions and during mouse movement, it remains to be seen whether it would produce any problems when doing both simultaneously. The strategy is implemented by adjusting the control-display relationship between cursor and mouse to create a consistent effective cursor plane position. The gain Δc applied to the actual image-plane control-display relationship is the ratio of the depth of the actual cursor plane d_a to the depth of the effective cursor plane d_e both with respect to the user (Figure 10).

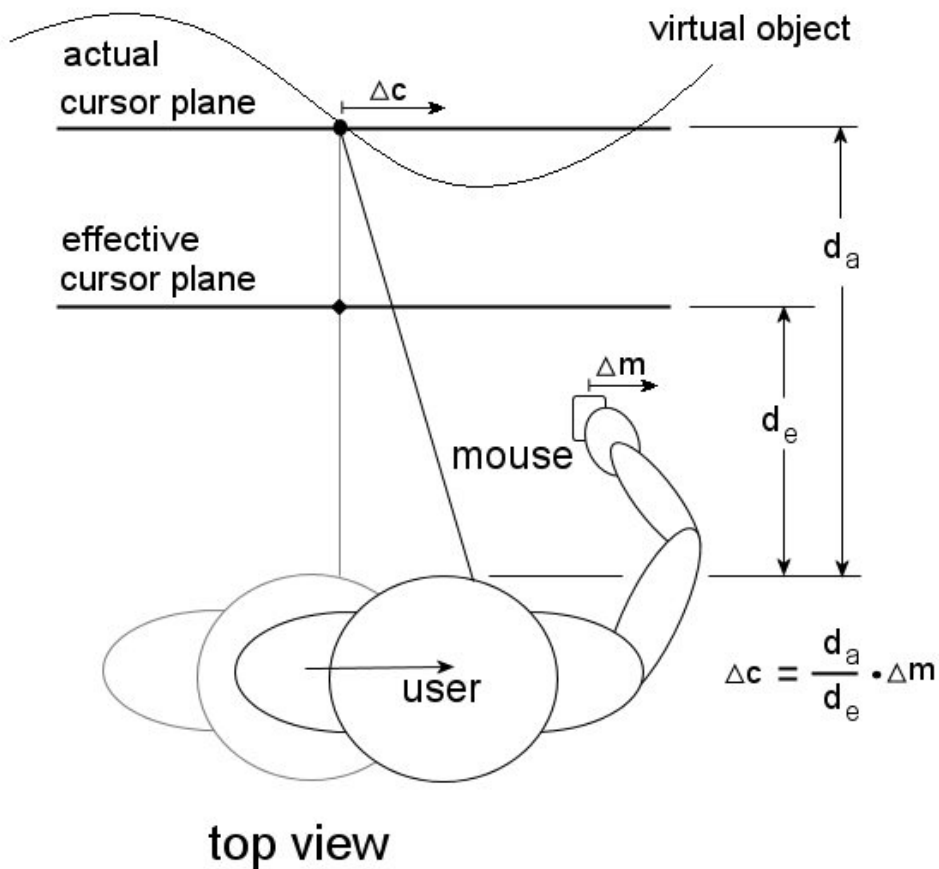


Figure 10. Placing the desktop stereo cursor plane at content depth and modulating the mouse-cursor control-display ratio stabilizes the cursor during head motion.

Given head tracking and stereo display, a higher degree of freedom input device can be added in any one of several ways. A third degree of freedom in mouse movement may serve some utility in allowing the user to lift

the mouse to adjust some depth related properties. For instance, the user may adjust a clipping plane depth or the depth of OS windows by lifting the mouse off of the flat surface. The relationship between a 3D mouse and the desktop content can also be rotated so that the mouse motion and resulting cursor motion are isomorphic. This condition is known as a virtual mouse since the mouse action now occurs in a plane in free space and is no longer constrained by a flat surface. The depth of the mouse from the screen can either be ignored to increase ease of use, be incorporated into the resulting virtual cursor position or utilized in some other manner as described above. A typical mouse is only a relative positioning device. In a manner similar to lifting the mouse and relocating it, virtual mouse implementations typically allow the user to hold a button while re-centering the mouse coordinate system. This disengagement of the mouse from cursor position is often referred to as clutching. Absolutely registering the mouse position with the position of the cursor in the image becomes strict image-plane selection.

The most important aspect of using image-plane principles with desktop stereo is that each ingredient of fully immersive stereo can be added or removed without compromising the usability of the technique. A 3D mouse, relative or absolute, can be used with or without stereo display. When head tracking is removed, and the viewpoint becomes fixed, the result merely degrades to a virtual mouse configuration. By contrast, ray-casting does not operate on the desktop without a full 6 DOF input device and a stereo display to accommodate interaction with content at or near the depth of the display.

5.1.3.2 Large Format Display Environments

Increasingly, large format display systems are finding use for the visualization of complex high resolution data sets and multiple asynchronous data streams. Research has shown that these display environments not only improve group work tasks but also improve the performance of spatial tasks for individuals (77). These display systems are often constructed out of several dozen LCD panels or multiple rear projection screens arranged in a grid. Typically the content on such systems is managed from a desktop using a desktop mouse and keyboard. However, these systems are envisioned as more interactive devices, and there is active research into interaction methods that allow users to use them more directly. Current best in practice interaction techniques for these systems follow directly from an image-plane selection strategy.

The same principles that apply to the traditional desktop case apply to large format displays. The most important aspect of interaction with these display devices has to do with cursor operation while standing in proximity to them. A number of researchers have investigated using touch to interact with these devices. However, as Sutherland observed over forty years ago, holding the arm up to a screen quickly becomes fatiguing. Ray-based techniques such as laser pointers have been a popular choice for interacting with large format systems at a distance, but these techniques have proven much less accurate for pixel level accuracy than a traditional mouse. One of the reasons cited for ray-casting inaccuracy is hand unsteadiness. Two recent studies have found that a virtual mouse implementation allows more accuracy than ray-casting techniques in these large format setups. The authors of these studies argue that users are able to steady their hand motion by holding their upper arm next to their body (78). The research done at the University of Toronto has found that a hybrid strategy using pointing to relocate the coordinate system for relative virtual mouse action offers similar performance to a strategy that uses virtual mouse alone (79). The transition time between virtual mouse and ray-casting compensates for a reduction in the number of arm re-centering actions required to move the cursor over large distances. This research uses a strategy similar to the one mentioned in a prior section to transition from image-plane virtual cursor directly into a ray-based technique that moves the mouse with hand orientations. Once the cursor has been moved to a new location, the relative virtual mouse behavior is restored at the new cursor location.

The transition from desktop interaction to virtual mouse interaction with a 3D mouse is a natural and smooth one that can be accomplished simply by standing up. The system can be configured to make the transition to virtual mouse as soon as the mouse leaves the vicinity of the desktop work area. As with the traditional desktop, stereo display and head tracking can be added without changing the fundamentals of an image-plane based interaction scheme. A framework that is based on image-plane concepts fits naturally within the context of current research results on interaction with large display systems at a distance.

5.1.3.3 Multiple Display Environments

Another active research area involves using multiple displays of different size and location to create a contiguous virtual workspace. These working environments allow the user to drag-and-drop content between display surfaces as disparate as Personal Data Accessories (PDAs) and large format displays. In addition to the technological challenges of creating a single large workspace from multiple devices, cursor management across

devices is an important research area. The predominant interaction method in these environments is a standard desktop mouse. Some of the latest research into effective cursor management in multiple display environments fits naturally into an image-plane based framework.

When display surfaces are not spatially continuous, a strategy for moving the cursor between surfaces in an intuitive manner needs to be developed. A common strategy that is used for desktop systems is called display stitching. The different edges of the displays are arranged into a virtual desktop that allows the cursor to smoothly transition between displays. Other proposed techniques introduce virtual space between displays to mirror the actual physical arrangement (80). However, one condition that screen stitching techniques cannot solve is when users or displays change location such that one display is overlaying a portion of another. In these circumstances, the questions of how to transition the cursor between displays becomes significantly more complicated.

Recent research into addressing these cases uses the position of the head with respect to the display environment to make the cursor smoothly transition across displays that are partially occluding one another (81). This technique, called perspective cursor, uses a 6 DOF head tracker and requires a complete database of each display location and size. This technique combines several best in practice techniques such as different mouse speed over different types of displays and halos to keep the user aware of off screen cursor position (82). However, the main aspect of this technique is actually a more general virtual mouse implementation similar to the one previously mentioned for a stabilized desktop cursor. The mouse position does not change during head motion, but the mouse moves linearly across surfaces at different depths from the user. A strict image-plane cursor cannot be stabilized because it remains attached to the hand. In this more general case of head motion cursor stabilization, the actual cursor plane remains perpendicular to the ray cast between the user and the cursor. The stabilization can be modulated between strict virtual mouse selection and cursor stabilization by placing the actual cursor plane a fraction k_m of the distance d_c between the effective cursor plane and the object under the cursor (Figure 11). Although the simplicity of the actual implementation of perspective cursor may not require the use of actual image-plane calculations, such an implementation would increase the flexibility of the system. An image-plane derived implementation would allow smooth cursor movement across stereo content within displays and facilitate a transition into either strict image-plane selection or a virtual mouse implementation.

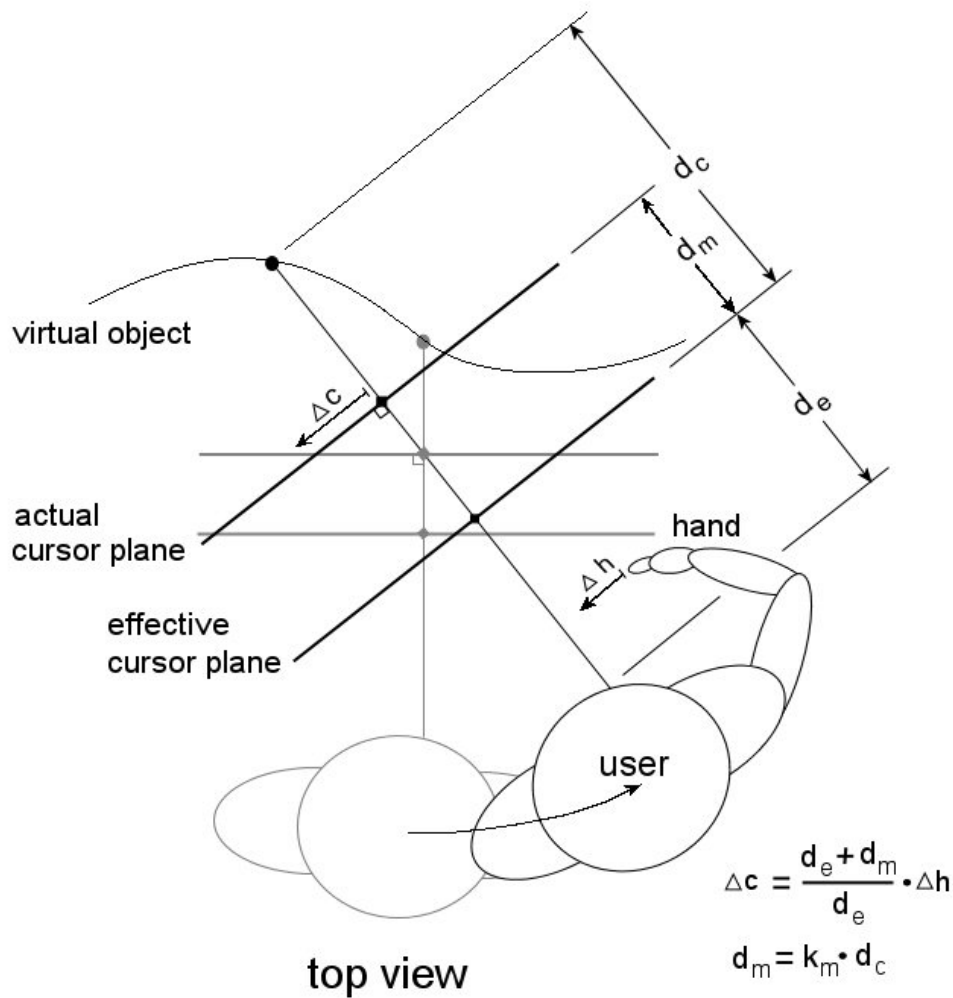


Figure 11. Virtual mouse stabilization is modulated by placing the actual cursor plane a fraction of the distance d_c between effective cursor plane and content underneath the virtual cursor.

5.1.3.4 Projection Based Augmented Reality

The expense and technical requirements of augmented reality systems have motivated a research area that uses inexpensive projectors to superimpose imagery onto flat surfaces. A single camera and relatively modest computational power can readily determine the location of objects within the image-plane. When camera and projector are collinear and the user is touching the surface, determining which interface elements are being occluded creates a simplified case of image-plane selection. However, the requirement that users touch the surface directly can present problems in augmented office environments. Physically touching the display surface is likely to disrupt the visual imagery and impair overall usability. Another problem with touching these surfaces has to do with user resistance. A recent study, conducted with Wizard of Oz techniques, used operators in

another room to emulate a system that can adapt to and learn any number of interaction schemes proposed by study participants (83). One significant result the authors found was that subjects were very reluctant to leave their seat to interact with surfaces. Users avoided touching any surface more than two feet away and in doing so would even expend more energy than touch would otherwise require. The authors even found a general unwillingness to touch even those surfaces that were within reach, with subjects preferring pointing based techniques that allowed them to minimize the movement of their arms and wrists. This is further evidence that hybrid solutions that use both ray-casting for gross selection and high pixel accuracy selection techniques may be appropriate.

While typical ray-casting cannot be implemented within these environments because of the discontinuous display surface, laser pointer based solutions do offer some potential for camera based tracking systems because the resulting laser light on the surface can be tracked with standard computer vision techniques. However, aside from their inaccuracy, ray-based techniques in projection based augmented reality environments are also likely to require motion along surfaces presented at an oblique angle to the user. As mentioned before, image-plane selection has advantages over ray-casting on these surfaces. The relative ease of acquiring the location of the hand via head mounted camera provides an opportunity to add interaction at a distance to projector based augmented reality. Given the known locations of the projection surfaces in a database, the hand position within the image-plane of a 6 DOF camera can be used to calculate a cursor position on the surface in front of the user (84). This extracted virtual cursor location can then be displayed in the virtual workspace via the projector. It should be obvious that such an image-plane scheme would still allow users to approach the display surface and interact with it via touch.

While the accuracy of this technique would be strongly affected by the accuracy of the 6 DOF tracking setup and the position of the camera on the head, such a setup may still prove usable even if the virtual cursor location does not track the hand perfectly. A relative image-plane interaction technique such as a virtual mouse may prove just as usable in practice. As long as the hand remains visible within the camera view, a virtual mouse implementation would only require the 3D position of the camera in order to accurately move the cursor in a plane between the user and current cursor position (Figure 12). In the absence of an input device with buttons, virtual mouse clutching can be accomplished using computer vision techniques and simple gestures (79). This type of virtual mouse implementation is effectively equivalent to the perspective cursor condition described above.

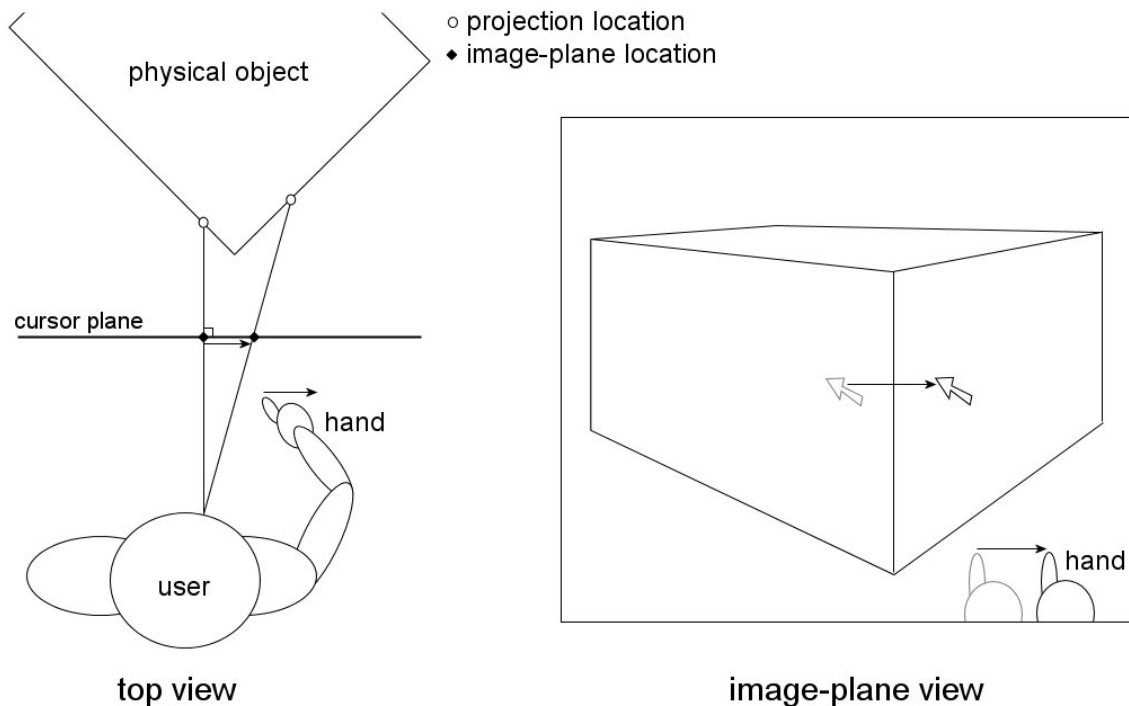


Figure 12. Camera based virtual mouse requires no orientation information to move the cursor on a plane between camera position and the content under it.

The nature of surface projected augmented reality can also increase the ease with which users position a virtual cursor. The most salient feature of projection based environments, the restriction of the display onto flat surfaces, can help create more isomorphic interactions when using image-plane selection. The subjects in the Wizard of Oz study mentioned above were given a virtual mouse option to manipulate objects on a surface. By placing their hand flat on the surface of interest, subjects could move a cursor relative to their hand motion. This pantograph technique, which used a high ratio of hand control to cursor display, proved unpopular among subjects in the study. It is reasonable to assume that the technique would have found greater acceptance had users been able to use a one-to-one control-display ratio without having to touch the surface. When a relative virtual mouse interaction is available, the ability of users to estimate the angle of the surface can aid the interaction. Instead of moving strictly with respect to the image-plane, hand motion that is approximately coplanar with the surface will closely mirror the resulting cursor motion presented on the interaction surface. This type of simple transfer function between hand and cursor motion should be relatively easy to use in practice because the control-display relationship remains almost completely linear.

The ability to interact with surfaces from a distance is an important ingredient in a successful augmented office environment. The low computational and hardware requirements of image-plane selection can potentially provide interaction at a distance while allowing these systems to remain economical and easily deployed. Furthermore, conceptualizing projection based augmented reality in terms of image-plane selection eases the transitions to and from desktop stereo, large format display and multiple display setups. By using image-plane as the basic selection technique in these environments, a common thread eases transitions between each of them and eventually straight into completely immersive environments.

5.1.4 Image-Plane Grab

The guidelines presented in the last chapter argue that manipulation at a distance is an inefficient way to manage objects in 1st person virtual environments. Although the framework presented here seeks to avoid manipulation at a distance, useful manipulation techniques that use image-plane selection can be derived. Image-plane grab is a manipulation at a distance technique similar to Bowman's HOMER technique that can use 3 DOF input devices for object positioning. Image-plane grab is an improvement to HOMER because it is more isomorphic and provides more consistent feedback to the user. When 6 DOF input is available, objects can be both positioned and oriented in an intuitive manner.

The Voodoo Dolls technique, developed by Pierce, is an example of a manipulation technique that utilizes image-plane selection. The technique uses Mine's Scaled-World-Grab metaphor to bring objects within the operating range of the user. The distinct advantages of the technique have already been discussed. One advantage that has not been mentioned is that the technique can also be used to position objects when only 3 DOF input is available. However, the Voodoo Dolls technique violates two important guidelines that make it a poor candidate for inclusion in this framework. First, it requires two hands, and as a result significantly increases the DOF requirements for the input device. Second, it can be inefficient because objects do not remain in their context during manipulation.

The image-plane grab technique is very similar to the HOMER technique and was first proposed by Pierce (22). As with the HOMER technique, the user is free to select an object with their hand at an arbitrary distance from their body (Figure 13). Also, as with Bowman's technique, the object can be moved to any location in a spherical coordinate system around the body of the user. The object distance from the user is relative to the

initial distance between the user hand and user viewpoint. Moving the hand half the distance to the eye point moves the object half the distance to the user. Furthermore, given full 6 DOF input, a selected object can be oriented directly at any location.

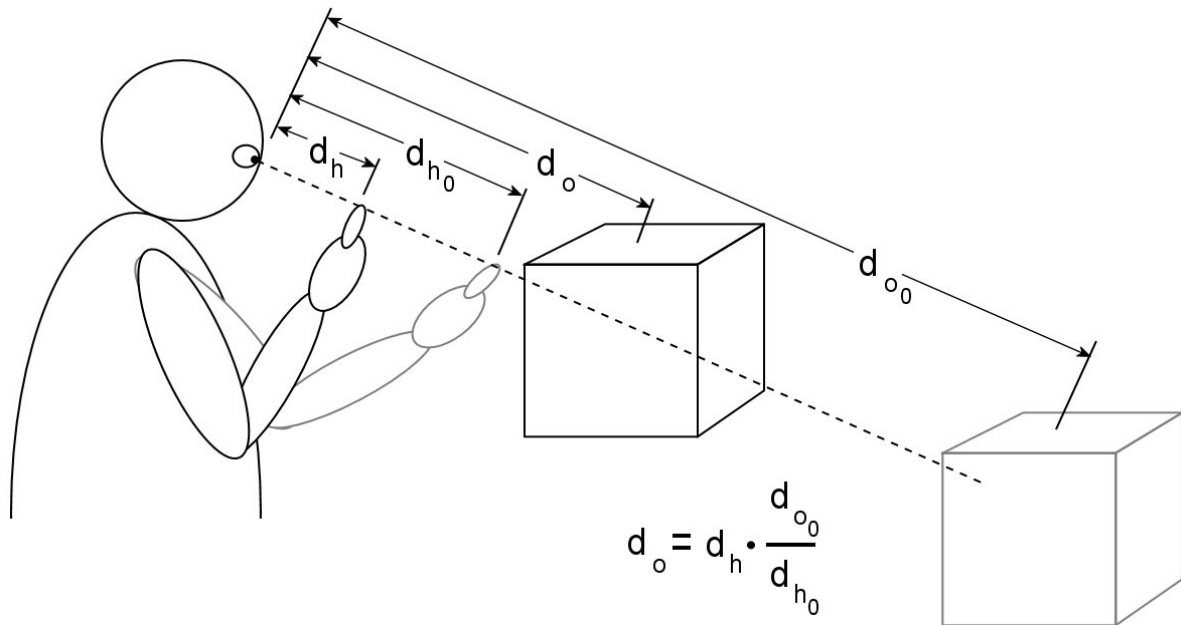


Figure 13. The initial hand and object depths from the user determine the hand and object depth relationship of image-plane grab.

The advantages of using a manipulation technique based on image-plane selection over ray-casting are two-fold. First, the manipulation action is significantly more isomorphic than that of the ray-casting technique because the object remains under the virtual cursor at all times. This makes orientation changes and movement to other locations within the image-plane much more natural actions. The second advantage of image-plane grab has to do with uncertainty about the actual control-display transfer function. The HOMER technique uses an estimated position of the user torso as the axis point for the spherical coordinate system that both the hand and selected object move within. Users can easily handle the offset between hand motion and object rotations about their vertical axis because they can estimate the location of the pivot point. However, it is impossible for the user to fully appreciate where the pivot point is along their vertical axis. As a result, this uncertainty makes it more difficult for a HOMER user to accurately control the vertical position of an object.

One obvious problem with using image-plane grab instead of a technique like HOMER is the potential for user fatigue. However, in some contexts, the tradeoff between increased fatigue and significantly more isomorphic interaction with content may be warranted. One such context involves the management of application windows and other user interface elements in a fully immersive environment. Ray-casting techniques such as HOMER are best suited for 2D content that is some distance from the user, but tasks such as window management cannot reasonably be expected to operate under such restrictive conditions.

5.1.5 Immersive Window Management

A common theme to desktop and other transitional configurations is that there is a reference surface on which graphical interface elements reside. Large format, multiple-display and projection based augmented office setups all typically confine user interface elements to the plane of the display surface. The unique nature of completely immersive environments such as augmented and virtual reality is that objects can appear floating in free space around the user. In order to use image-plane selection with traditional GUI widgets in these environments, some techniques for managing these surfaces need to be developed. The Withindows framework supports these common interface elements through a combination of the image-plane selection techniques described above, extensions to existing window management rules and image-plane grab. This combination of techniques makes interaction with 2D interface widgets within immersive environments easier to manage and use.

The advantage of using pseudo-haptic interaction on physical surfaces such as tablets and tables has already been discussed before. However, an appropriate physical device may not always be available or convenient to use. Tablets, however light they may be, are limited by their a priori size and cannot be scaled up when the needs of the interface demand it. However, the smooth transition between touch and distant interaction makes image-plane selection a good choice when a combination of physical surface and floating interface elements are used. Floating windows can be snapped to physical surfaces and automatically resized as they are on desktop systems when resolutions are decreased. Interface elements such as cascading menus can run off of a physical prop and still be used effectively with image-plane techniques because of the smooth transition away from touch. When 2D interface widgets such as windows are not fixed to a physical surface they tend to fall into one of three categories. Feiner introduced the categories of surround-fixed, view-fixed and world-fixed windows in

his publication regarding the display of XWindows in virtual environments (46). The orientation of view-fixed windows will always be parallel to the image-plane and, as a result, they have proven very usable with image-plane selection. World-fixed windows either have a fixed position within the virtual world or are attached to objects and move with them. Mine was influential in introducing body-centric principles to virtual reality research that utilize surround-fixed interface elements (31). Surround-fixed windows remain in the same position relative to the body of the user.

5.1.5.1 Standard Functionality

Regardless of the positioning of windows in the environment, they all benefit from attributes common to those on the desktop. Window locations should be adjustable. Window size should be adjustable by selecting a corner or edge and stretching the frame. Windows need some affordances that hide them, retrieve them and normalize their size for easy viewing. And, window focus should be selectable either by selecting the window or by bringing it in front of other windows. The techniques that follow are those proposed for the management of windows within the Withindows framework. Window management remains an active research area and the techniques presented here are not meant to solve fundamental windowing problems on the desktop. Instead, they are an attempt to extend and transition existing desktop functionality into 3D space in a consistent manner.

Regardless of the method used to select and manipulate their attributes, the addition of 3D position gives windows some unique traits. First, whereas windows in desktop environments are presented in a simulated depth ordering that only allows windows to be brought to the front, windows in 3D environments can be organized in depth with more flexibility. Windows that move backwards in depth are naturally reduced in size while those coming forward are increased in size. This trait allows the contents within windows to be easily scaled in size. This can be useful because virtual and augmented displays are often limited in resolution. With respect to depth, windows are unlikely to find much utility when they are occluding the input device, which in virtual and augmented reality is likely to include the hands. As a result, windows should, regardless of their actual depth, be rendered behind some indicator of the virtual cursor location. The relationship between windows and content in the surrounding environment is not as straightforward. Because windows and the information contained within use up valuable display real estate, rendering them in front of content is not always appropriate. However, like windows on the desktop, an option should be available to keep a window “always on top”. Otherwise, world content and other windows in between a window and the user should be allowed to occlude it.

Just like on the desktop, windows need to be minimized in order to reduce the amount of real-estate they occupy. Mene proposed that menus reside in a location just out of view from the user and be “grabbed” and pulled into view via virtual hand techniques. Techniques that allow selection at a distance give more flexibility to move windows down and away from immediate view. Windows fixed to the body can be minimized to a location around the feet. From this position they are easily restored by a single button click and returned to their former position with little physical effort. Windows fixed to the viewpoint can follow rules directly analogous to desktop windows, moving to the edge of the image-plane when minimized. Windows fixed to the world or objects can minimize to locations either below them or in a location dictated by the environment developer. As such, a minimize button attached to 3D windows, similar in appearance to those on desktop windows, is appropriate. Aside from the close button, the other button ubiquitous to desktop application windows is maximize. The results of maximizing a view-fixed window are analogous to the desktop case, the window fills the complete image-plane. Because of the potential for disorientation with completely immersive technology, it seems more appropriate to restrict the size of the window so that some peripheral vision is still maintained. The intended purpose of maximizing a surround-fixed or world-fixed window is to make interaction with it the focus of attention. To this end, moving the window to a pre-defined and prominent position within the viewpoint of the user is an appropriate action. The behavior of the window once maximized need not change; surround-fixed windows will remain so, while world and object fixed windows may be designed to revert to their prior position within the viewpoint when the user looks or moves away.

The ability to move windows between a surround-fixed, world-fixed and view-fixed state is appropriate. The coordinate system to which windows are fixed should be adjustable within the Withindows framework. An additional button added to the window frame can allow windows to toggle between view, body and world fixed settings (Figure 14). As in desktop environments, not all windows need allow such changes to occur. Some world fixed windows may be designed to remain permanently in place, while users should be allowed to leave their own windows attached to arbitrary world locations. Selecting an object before toggling the coordinate system button is a convenient way to fix the location of a window to that object instead of its location in the world.

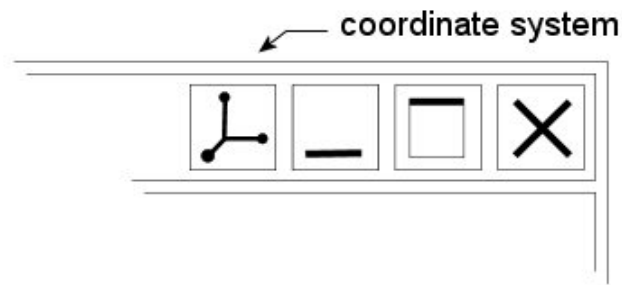


Figure 14. Proposed window icon for switching a window between view-fixed, surround-fixed and world-fixed coordinate systems.

5.1.5.2 Position and Orientation

In addition to managing the coordinate system of windows, the position within the user field of view and their orientation are also of concern. Windows in 3D environments have the possibility of residing in any location with respect to the user. Given the reduced resolution within immersive environments and the added difficulty of working on affine distorted windows, it appears appropriate that windows remain oriented to the user when feasible. Bowman realized the utility of keeping windows oriented to the user when he implemented his VEWL windowing API. Using a ray-casting selection technique, windows move around the user, maintaining tangency with the surface of a user centered sphere (85). This same drag-and-drop behavior works naturally with image-plane selection; its depth insensitive nature makes moving windows around the user a natural exercise. Using the same image-plane grab techniques described earlier, windows can be easily moved closer and farther from the user. Windows at arms reach will change depth in isomorphic fashion while those at twice the distance will change depth at twice the rate. With window orientation fixed to face the user, image-plane grab allows windows to be intuitively managed in an orbit around the viewpoint.

Keeping windows oriented towards the user is not always feasible or desirable. There may be world fixed windows that are designed to force the user to view them from a specific direction. There may also be utility in allowing users to adjust the orientation of their own world or surround fixed windows. As we will see later, there are also likely situations where pointing a window in a particular direction offers benefits. For this purpose, yet another window frame button that toggles the orientation state of a window is appropriate (Figure 15). A single click to this orientation button allows windows to be oriented freely via the image-plane grab technique. Another click re-oriens a window towards the viewpoint of the user. For windows that do not remain viewpoint oriented,

clicking on this same button can temporarily orient the window for easier viewing while the mouse button remains depressed or until the user looks/moves away.

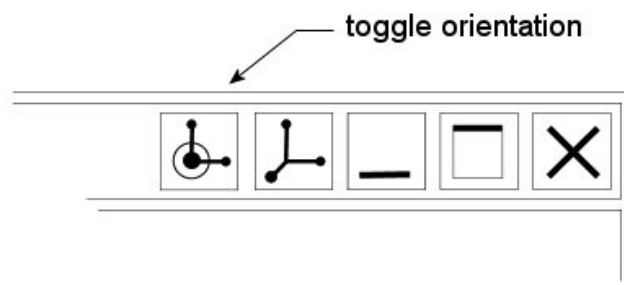


Figure 15. Proposed window icon to for switching a window between viewpoint oriented and free oriented states.

In keeping with the transitional focus of the Withindows framework, windows that migrate to immersive environments bring all of the same functionality from the desktop. Some functionality, such as minimization and maximization behave in different ways that are appropriate within a 3D environment. Additional window functions also become available in 3D that are appropriate to a display environment with more flexible concepts of what the desktop is; view-fixed, surround-fixed or world-fixed fixed. Moreover, other functionalities such as window orientation become available when hardware such as a 6 DOF input device is available.

Image-plane selection is a natural input method for manipulating 2D application windows in an immersive environment. It is also reasonable to ask if there are other situations that need to be considered when choosing a canonical input method for our framework. While the mouse input has proven capable of satisfying the requirements of desktop systems, a canonical input method and associated device need to support the full spectrum of tasks common to immersive environments. As we will see, image-plane selection is also the foundation of selection, manipulation, search and travel methods that are efficient and low in fatigue.

5.2 Through-the-Lens Techniques

The second major component of the Withindows framework is the use of through-the-lens techniques for primary immersive tasks. This class of techniques uses alternate viewpoints constrained within windows in the immersive environment to carry out immersive interaction tasks with high efficiency. These tasks include object selection, object manipulation, global search and virtual travel. Although image-plane selection is the primary selection technique of the framework, so far, we have only discussed its use for selection and manipulation with respect to a 1st person viewpoint. The efficiency of object selection and manipulation are significantly increased when they are coupled with the ability to control viewpoint. Furthermore, when combined with techniques for managing 3rd person viewpoints, alternate viewing windows allow efficient global exploration of the environment and preview of travel in virtual environments. A significant part of the Withindows framework is the combination of image-plane selection and through-the-lens techniques. This combination creates the conditions for constrained object manipulation and viewpoint management techniques that are analogous to those currently used in desktop modeling and CAD applications. This tight integration between desktop 3D applications and the development of applications to manage 3D space from within is the key that allows full featured immersive applications to be ported directly to desktop environments without redevelopment.

5.2.1 Previous Through-the-Lens Techniques

The limitations of first person perspective are the main motivation behind through-the-lens (TTL) techniques. Those limitations include difficulty selecting objects that are distant or occluded by objects in view, the inability to effectively manage object manipulations when depth is compressed by perspective projection, the inability to preview teleportation events and the inability to easily develop an overview of an environment while exploring it egocentrically. The term through-the-lens was introduced by Stoev and Schmalstieg in their paper introducing techniques for selecting and manipulating occluded objects by looking at them through a viewing window (86). Stoev later went on to describe a complete taxonomy of window interaction techniques for managing viewpoints within these windows that included search and travel techniques (87).

Through-the-lens techniques combine 3D magic lenses with the WIM technique (14,25). The WIM technique is both a gross object manipulation technique and a global overview technique. One of the drawbacks of the WIM technique is that the miniature model of the environment is superimposed directly onto the surrounding environment. This superposition can lead to confusion between the miniature and surrounding

environment. Also, as it is usually implemented, the WIM technique requires that a second, low resolution, model of the environment be developed. When the environment is large, features in the miniature model can become too small to effectively view or select. By encapsulating a 3rd person viewpoint of the scene within a framed window, as with 3D magic lenses, Stoev and Schmalsteig avoided ambiguity problems and produced a viewpoint of the primary model that could be translated or rotated for easier object selection and zoomed outward to provide overviews or details of the environment. This novel introduction of a completely secondary viewpoint allowed users to manage a global overview of the environment without having to change their virtual location. The complete taxonomy of possible relationships between the user viewpoint, the content in the alternate scene and the position of the viewing window includes many variations. For the purposes of the following discussion, we will assume that these alternate viewing windows are floating in the immersive environment in a fixed position and that moving them or user viewpoint does not change the view displayed within them.

5.2.1.1 Object Selection and Manipulation

Alternate viewing windows co-located within the immersive setting allow the user to view and select occluded objects from another perspective. Reaching into the frustum of the alternate viewing window allows the user to select objects with a virtual hand technique. These alternate viewpoints also allow the user to zoom into distant objects and manipulate them within the window as if they were actually close at hand. The power of this technique is obvious; the user can create a viewpoint for the manipulation task that is optimized to the task at hand. An object that must be aligned with the floor can be viewed from a side view during manipulation. This technique solves the most severe problem associated with the Voodoo Dolls technique; that objects are not viewed in their true context. And, TTL object manipulation techniques also solve isomorphism problems associated with techniques like HOMER and Go-Go because the object remains attached directly to the virtual hand.

Most importantly, these viewing windows can solve the scaling problem that Voodoo Dolls solves with a reference object in two different ways. The primary method involves scaling the scene viewed within the alternate viewing window in order to bring objects within reach of the hand. Views from higher above the horizon will scale the secondary scene down so that objects on the ground can be reached. As the viewpoint approaches ground level, the scaling approaches one-to-one and normal isomorphic interaction applies. This scaling of the scene allows objects to be moved over a distance that is reflective of the field of view visible within the window. A second method of managing the scaling problem involves drag-and-drop movement of objects between the

secondary scene and the surrounding scene. This technique allows the user to move objects long distances while maintaining a uniform scaling between object and hand motion. Although two viewpoints must be arranged in advance, the source object viewpoint and a viewpoint of the destination location, this technique allows a user to move an object over an arbitrary distance and position it accurately with a single action.

As with the Voodoo Dolls technique, TTL users can simultaneously manipulate the position of an object and control their view of that object. By using two hands, a user can control the viewpoint with one hand while manipulating the position of the object with the other hand. In the same way that two handed manipulation is a synchronous task in Voodoo Dolls, any change in the camera position within the secondary scene while the object is held will result in a change in its position with respect to that secondary scene. However, in contrast to Voodoo Dolls, there are opportunities to asynchronously manipulate an object and its associated viewpoint. By grabbing and manipulating an object within the surrounding environment, the user can both manage an alternate viewpoint within a window and position the object in an asynchronous fashion. As powerful as the TTL technique is, the tools originally provided to manage TTL viewpoints are unfortunately not only inefficient but also the source of inconsistent scaling relationships between the hand and objects within viewing windows.

5.2.1.2 Viewpoint Management

Through-the-lens relies heavily on virtual hand based techniques for the management of the alternate viewpoint, sometimes called viewpoint navigation. A combination of three techniques, eyeball-in-hand, scene-in-hand and TTL-WIM are used to manage the viewpoint within a window. The eyeball-in-hand technique allows the position of the hand to directly determine the position and orientation of the viewpoint shown in the window. The scene-in-hand technique allows the user to reach out and effectively grab the scene within the window and translate and rotate it arbitrarily. This technique, similar to a virtual mouse, allows the user to use clutching to re-center the hand in order to cover more distance. In addition to these two techniques, a TTL-WIM technique allows the user to define a rectangular area on the surface of the viewing window and zoom the window viewpoint to fill the window. This technique always produces a viewpoint that is orthogonal to the horizontal axis. By zooming out and selecting areas from an overview a user can cover large distances easily.

As originally conceived, a TTL user has to use a combination of at least two very different techniques in order to accomplish many routine viewpoint management tasks. For example, if the user wants to view an object

seen in the distance close up, they have to use TTL-WIM to zoom out to a viewpoint that includes that object then zoom into the region containing the object. The same user would then most likely use the grab-and-drag technique to zoom into and center the object in question within the window. This sequence of tasks not only requires the use of two techniques that are fundamentally different in nature to accomplish similar tasks but also discards the original viewpoint that contained the object within it.

The second problem with the TTL techniques involves the scaling of the secondary scene to aid object manipulation. As mentioned before, TTL windows solve the scaling problem by scaling the scene within the window in order that objects can be reached and manipulated within the view frustum. However, the circumstances that determine this scaling factor are very inconsistent because the scaling factor is only determined by the frustum between the viewpoint and the ground. Only the TTL-WIM technique scales the secondary scene when zooming into an object from above. This means that movement of objects from one ground location to another will involve scaling that allows their reach. However, if the scene-in-hand or eyeball-in-hand techniques are used to bring a distant object within view, there is no longer any guarantee that it will be within arms reach. The lack of an object-centric approach to viewpoint management makes TTL object manipulation tasks inefficient. If distant objects could be selected and focused upon, then the secondary scene could potentially be scaled appropriately to make them reachable within the viewing frustum.

5.2.1.3 Global Search and Travel

In addition to object selection and manipulation, TTL windows and WIM techniques allow the user to get a global overview of the environment. This overview capability allows the user to search for objects and locations without moving from their original location. This ability is extremely valuable in traditional augmented reality environments where the user cannot physically move to an overview position. In response to this, some recent work allows the user to manipulate the orientation and position of the information overlaid onto their viewpoint. When lifted up and rotated, information about landmarks in their view can be more easily interpreted (26). Another benefit of alternate viewpoints in virtual environments is that users can preview a teleportation viewpoint before initiating a teleportation event. For either global search or teleport previews, the scene-in-hand viewpoint management technique of TTL makes it difficult to generate orbital viewpoints of content. Seeing an object from multiple angles requires reaching out to the object and rotating the scene from that pivot point. Given that TTL-

WIM is the only technique that scales the secondary scene, it is likely that the user will have to use a combination of both techniques to accomplish this task.

The relationship between TTL and 3D Magic Lenses is also an important one. Alternate viewing windows are effectively subjective views of the surrounding world. The system developed by Stoev and Schmalsteig allowed users to preview object manipulations within windows before implementing them in the surrounding environment. These same windows are readily used for all manner of subjective viewpoints such as wiring diagrams, finite-element analysis, X-ray vision, wireframe overlays and others. Up until this point, we have assumed that moving the alternate viewing window does not change the view within the window. However, by registering the position of the secondary scene directly over the surrounding scene during window movement, the viewing window can appear as nothing more than a transparent window. When used with X-ray vision or other subjective viewpoint tools, the viewing window then becomes a subjective lens into the surrounding world.

This behavior, directly analogous to 3D magic lenses, can also be exploited for the purposes of managing the viewpoint within the window. By registering the two scenes, the user can move the viewing window over content in the surrounding scene and then fix the secondary scene to the viewing window as before (Figure 16). The viewing window can then be moved back to its original position with the captured viewpoint intact. Finally, managing a viewpoint within a window is potentially disorienting because it is easy to lose track of the position and orientation of the virtual camera relative to the surrounding environment. By registering the viewing window with the surrounding environment, a user can better appreciate the starting point for their viewpoint manipulations.

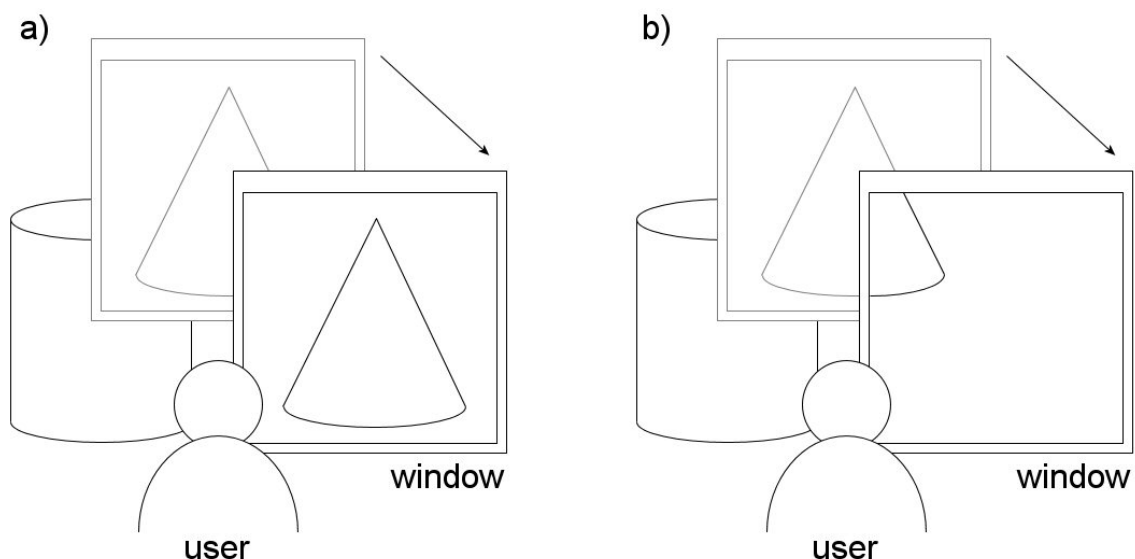


Figure 16. The alternate viewing window can either a) remain locked to the secondary scene or b) allow its movement to change the viewpoint into it.

5.2.2 The Combination of Image-Plane and TTL Techniques

The basic principles of through-the-lens techniques provide a novel solution to the most significant impediment to efficient interaction with immersive environments; the limitations of first person perspective. Global search, travel previews, object drag-and-drop, appropriately scaled object manipulations and asynchronous two-handed viewpoint and object manipulation are among the significant contributions the technique brings. However, viewpoint management techniques that are not object-centric keep TTL methods from being as efficient as possible. And, like other virtual hand based techniques, users potentially have to move their hands over a wide and sometimes awkward range to accomplish basic tasks. Virtual hand manipulation techniques also complicate any attempts to render viewing windows in parallel perspective. When combined with a depth insensitive technique like image-plane selection, TTL techniques can be implemented with significantly less effort. This combination of image-plane selection and TTL techniques makes it possible to use constrained low DOF viewpoint management and object manipulation techniques that are directly analogous to those found on the desktop. These focal point centered methods generally known as 2½D techniques are an important element of the Withindows framework.

If it is not already obvious, the relationship between image-plane selection and the desktop extends to the techniques commonly used for interacting with 3D content in 3D desktop modeling and CAD software. The

predominant interaction technique on workstation 3D modelers and CAD software, often called 2½D, is merely an implementation of image-plane selection on the desktop. As on the desktop and in immersive settings, 3D objects can be selected by placing the cursor directly over them and clicking. As with desktop applications, a natural extension to image-plane selection is the ability to select multiple objects by dragging a selection rectangle over a portion of the image-plane. Workstation modeling applications have adopted a number of techniques over time that are directly applicable to the alternate viewpoints in TTL windows. These techniques are predominantly object-centric and have been developed for both the management of viewpoints and the manipulation of the 3D objects within them. The Withindows framework uses these 2½D interaction techniques to manage window viewpoints and manipulate the objects within them.

5.2.2.1 Viewpoint Management

Viewpoint management in workstation modeling packages such as Maya⁵ typically controls viewpoint by asynchronously rotating, panning and zooming the viewpoint with respect to a coordinate origin (88). These focal point centered techniques easily allow a user to zoom and rotate the viewpoint with respect to the origin of objects in the scene. This is usually accomplished by clicking some portion of the viewing window and dragging the cursor horizontally or vertically. This same technique can naturally be extended to TTL windows; depending on the current mode, clicking the virtual cursor over some portion of the window and dragging alters one aspect of the viewpoint. Some applications such as CosmoWorlds⁶, strictly use buttons on the window to change modes, while others, such as Maya, use combinations of keyboard keys and mouse buttons to quickly change the mode that is applied to the drag action.

The other major attribute of desktop modeling packages is either multiple viewpoints that include top, side and front views or the ability to easily toggle the current viewpoint between these orthonormal viewpoints. An important aspect of orthonormal viewpoints is that they are rendered in a parallel perspective projection; objects remain the same size regardless of their depth from the camera. Parallel perspective eliminates the distortions of first person perspective projection and makes it possible to view multiple collinear relationships within the same viewpoint. Because the predominant model on the desktop involves multiple viewing windows, transitions between parallel and perspective projection are typically initiated abruptly without concern for any disorientation

⁵ [http:// www.autodesk.com/maya](http://www.autodesk.com/maya)

⁶ <http://www.sgi.com/products/software/cosmo>

that they may cause. However, an object-centric focus allows viewpoints to transition smoothly between perspective and parallel perspective without significantly distorting the area of attention. By simultaneously narrowing the field of view and zooming out from the center of focus, a perspective projection asymptotically approaches a parallel perspective. Without a center of focus, there is no reference with which to modulate the viewpoint field of view and the area of attention will grow or shrink in an unpredictable manner. With this technique, the forward clipping plane is established by the original viewpoint and adjusted by subsequent zooming operations.

5.2.2.2 Global Search and Travel

Desktop style viewpoint management solves the problems associated with the original eyeball-in-hand, scene-in-hand and TTL-WIM TTL techniques. A combination of an object-centric design and more consistent viewpoint management techniques allow the user to obtain global overviews without losing their focus and implement more flexible virtual travel schemes. Two useful functions that are commonly available to workstation users are view-all and select-focus. The view-all function zooms the viewpoint out to show all currently selected objects or, if no objects are selected, the entire scene. The select-focus function changes the rotation and zoom focal point from what is initially the world origin to that of the currently selected object or group of objects. This function typically changes the orientation of the viewpoint in order to bring the selected object to the center of focus. These two functions play an important part in the increased efficiency of global search and travel within the Withindows framework.

The advantages of being able to smoothly zoom out from an object should be obvious. Because of its reliance on selecting a rectangular section of the ground plane, the original TTL-WIM technique does not allow the user to maintain either their focus on an object or the viewpoint orientation with respect to the object. By using a combination of the view-all function and a top-view function, a user can easily zoom out from an object, move to a top view, zoom to view the whole scene or create a top view of the whole scene in two or less button clicks without shifting their focus. While these actions can be emulated with scene-in-hand and eyeball-in-hand techniques, their use would no doubt be more fatiguing and less accurate.

From any overview position, regardless of location or angle, objects can be selected and zoomed into with a couple of mouse clicks. This object-centric focus solves the first major shortcoming of the original TTL

viewpoint management techniques. An object that is visible within an alternate viewing window can be easily selected regardless of its distance from the viewpoint with image-plane selection. The select-focus function can then change the viewpoint to center on that object. At this point, the user can zoom into the object and adjust their viewpoint around the object as desired. Alternately, the user can utilize the view-all function to zoom directly into the object. This technique of managing viewpoint allows a user to move their viewpoint easily between visible targets while keeping their destination within focus at all times.

Through-the-lens and WIM teleportation techniques that involve moving the viewpoint to cover the user field of view reduce disorientation but may prove time consuming and fatiguing to more advanced users. The Withindows framework includes two teleportation methods that increase the speed and accuracy with which users can teleport themselves in virtual environments. The first method is similar to existing techniques but does not require moving the window over the user image-plane. An additional icon located along the top of the window frame allows the user to teleport immediately to the viewpoint of that window (Figure 17). To ease disorientation, instead of moving the user viewpoint exactly to the window viewpoint, the user can be teleported to the relative position of their current viewpoint to that of the window. The result is that the view within the window does not change, but the surrounding view changes. Or put another way, the surrounding scene becomes registered with the secondary scene. It is worth noting that this technique cannot always be implemented without discontinuity between the primary and secondary scenes because object-centric viewpoints allow the camera to rotate into positions that are not level with the horizon.

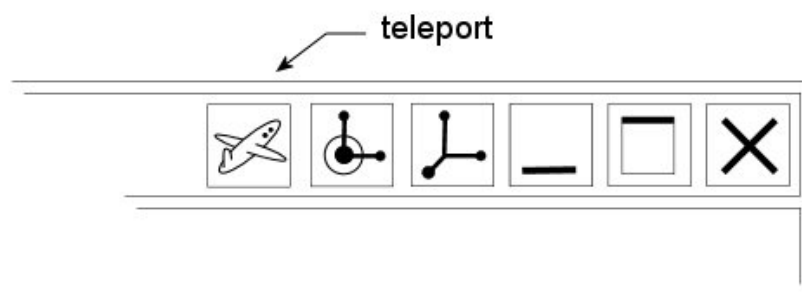


Figure 17. Proposed window icon to teleport the user to the location that coincides with the current window viewpoint.

Although the above technique is useful for teleporting to arbitrary locations in a virtual world, it is not an efficient way to accurately establish the end position of the teleportation. Whether moving the window over the image-plane or teleporting to a window relative viewpoint, it is difficult to appreciate the exact horizontal and vertical position the user will move to. The second teleportation scheme in the Withindows framework uses image-plane selection to move the user to the first applicable surface under the virtual cursor. This method is similar to but more accurate than other ray-casting teleportation methods that have been implemented before, and can be used to select locations either within a window or in the surrounding environment. As a result, it is appropriate that the initiation of this teleportation is not tied to a particular button on a window but is instead a function available globally to the user. The actual method of invocation will be discussed below in the section on global versus viewpoint functions.

5.2.2.3 Object Manipulation

The original through-the-lens implementation was limited by its reliance on virtual hand techniques. The authors of the original paper on the subject alluded to the potential use of manipulation techniques that work at a distance (86). The limitations of their viewpoint management techniques and a lack of an object-centric focus contribute to the inefficiency and inconsistency of the implementation. The addition of traditional 2½D constrained motion techniques to TTL windows allows for more highly constrained object manipulations that consistently create the proper control-display scaling for the task. The Withindows framework uses these image-plane oriented techniques to allow efficient object manipulation in the absence of either haptic feedback or 6 DOF input devices.

Desktop 2½D object manipulations are so named because they do not involve true 3D object manipulation. This class of techniques constrains a single axis of a three dimensional manipulation task in order to allow a 2D input device to specify the remaining two axes efficiently. These techniques are closely related to image-plane selection. A typical example involves selecting an object and moving it along a flat plane displayed in perspective projection (Figure 18). Because the object remains fixed under the cursor, the resulting location on the plane is dictated by the projection from the viewpoint through the cursor and onto the plane. The user often chooses the desired fixed axis by selecting a different location on what is known as the manipulation handle or 3D widget. When top, side or front viewpoints are displayed in parallel projection, the selection of the constrained axis becomes automatic. Often the user can choose to constrain all but a single manipulation axis. The manipulation of orientation and scale typically benefit from this ability. These 2½D techniques are almost universally used and

allow users to accurately specify the locations, orientations and dimensions of objects in a Cartesian coordinate system. Efficient desktop interaction techniques remain an active research area and better methods will likely be developed. However, this thesis is more concerned with the relationship between best in practice desktop techniques and immersive interaction than it is with the merit of those techniques.

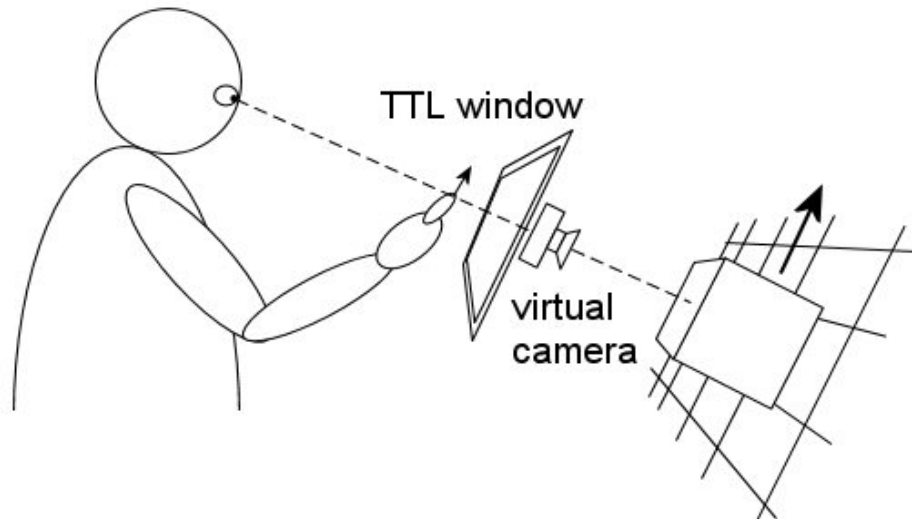


Figure 18. Image-plane selection on TTL windows works naturally with typical 2½D interaction techniques that constrain object movement to a plane.

Despite their ability to address the control-display scaling problem, the reliance on virtual hand techniques makes the original through-the-lens implementation no less fatiguing and no more accurate than typical virtual hand interaction on local objects. Objects can be moved with high precision and low effort from a window to the surrounding scene via a drag-and-drop. However, the added effort of managing two viewpoints and the inability to scale the surrounding scene limits the range of situations that benefit from it. As a result, object manipulations that require a different scale, those on very large or small objects, will be done exclusively within TTL windows. As mentioned before, the solution to the scaling problem provided by the original TTL concept only ensures that objects in the frustum between the viewpoint and the established ground plane can be selected and manipulated. This limitation arises because, in the more general case, the system does not know the intended object origin and destination. Image-plane selection on TTL windows handles this shortcoming naturally because all objects are selectable and windows automatically establish the relationship between control and display. The isomorphic

relationship between the object and the virtual cursor will naturally scale the distance and accuracy of constrained motion in a window that is either zoomed out to reveal a large area or zoomed in to a small object. In addition to 2½D constrained manipulations, 6 DOF techniques such as image-plane grab can also be used in TTL windows with appropriate control-display relationships.

5.2.3 The Role of 6 DOF Input

The advantage of using 2½D techniques on TTL windows is that they only require low DOF input devices in order to accomplish selection, search and manipulation tasks. As we have seen, there are limitations to the effectiveness of 6 DOF techniques when they are used to manipulate objects in depth from egocentric viewpoints. However, this does not mean that 6 DOF or higher input devices have no utility. The Withindows framework is focused around low DOF input requirements in order to allow simple canonical input means and facilitate transitions between desktop and immersion. But, when available, 6 DOF input devices can add greater flexibility and expressive power to viewpoint management and object manipulation tasks. The role of 6 DOF and higher input devices within the Withindows framework is to extend the basic functionality without replacing its reliance on low DOF input methods.

5.2.3.1 Object Manipulation

The image-plane grab technique introduced in the section on image-plane selection is an example of how 6 DOF input can be useful in the Withindows framework. This technique allows the user to move objects freely over a large range with a single action. The discussion above showed how 2½D constrained object manipulations within windows address the scaling problem. Image-plane grab can also be used within alternate viewing windows in a manner that sets up an appropriate control-display scaling. Obviously, since image-plane grab remains isomorphic, the range of motion orthogonal to the line of sight will be scaled appropriately to the viewpoint established by the window. The effect of manipulating the distance of objects is not as obvious. Image-plane grab establishes the same relationship between object and hand motion when working on objects either in front of the user or within windows. The object remains under the virtual cursor and its depth remains relative to user viewpoint. However, the depth scaling of objects within viewing windows is more appropriately established with respect to the location of the virtual camera (Figure 19). A virtual camera to object distance that is the same as the distance between user viewpoint and their hand will produce a one-to-one relationship between object and hand depth. A virtual camera that is twice as far from the object will allow the object to be moved over a depth

range that is twice as large. As a result, zooming the window out to reveal a larger field of view within which to move the object scales the range of depth manipulation accordingly.

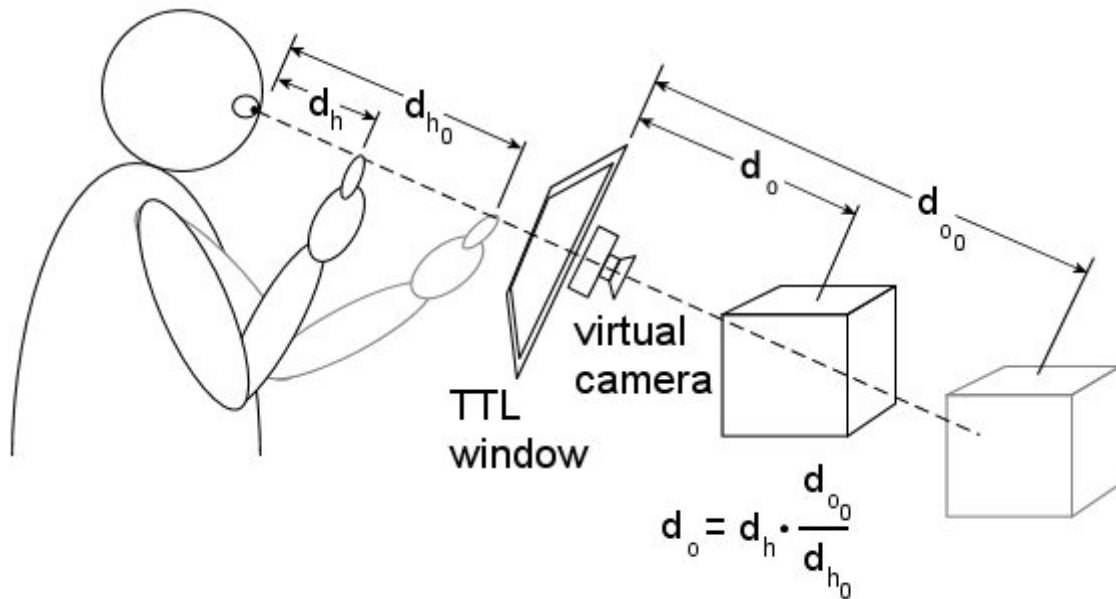


Figure 19. Image-plane grab within TTL windows avoids scaling problems by using the object depth from the virtual camera location rather than from the user.

5.2.3.2 Viewpoint Management

The 2½D based viewpoint management techniques described above only allow the zoom, pan or rotation around an object to be adjusted independently. Usually, this is a good use of two dimensional input devices, with pan adjusting both the horizontal and vertical position of the camera and rotation adjusting both the azimuth and zenith of the camera with respect to the object. A natural use of a 6 DOF input device is the simultaneous adjustment of both rotation and zoom with respect to an object. By tying the zoom parameter to the user viewpoint, in a manner similar to image-plane grab, the user can intuitively manage the camera viewpoint in a body-centric manner. Bringing the hand half the distance towards the user viewpoint, zooms the camera half the distance to the object in question. When initiated with the hand in between the user viewpoint and viewing window, the control-display relationship between hand orientation and viewpoint rotation remains almost isomorphic. Rotating the hand towards the right of the user viewpoint rotates the viewpoint to the right around the

object. Because the rotations are all body-centric, this viewpoint management technique can be initiated with the hand located in any position relative to the user without significant loss in efficiency. As other researchers have demonstrated, a non-linear relationship between wrist rotation and viewpoint orientation is a practical option (72). This 6 DOF viewpoint manipulation technique remains an additional function and does not supplant the normal viewpoint manipulation techniques. The details of invoking 6 DOF viewpoint management are covered in the next section.

5.2.3.3 Window Management

The initiation of 6 DOF viewpoint management and the capturing of the surrounding scene within the viewing window can both be accomplished with the addition of two buttons to the window frame; a home button and a lock button (Figure 20). The lock button toggles the relationship between window motion and the secondary scene. By default, the button indicates the window and secondary scene are locked and will move together. Pressing the home button immediately registers the window viewpoint with the surrounding scene and unlocks the secondary scene from the window. The window can then be moved over objects to frame them within the window. Once a desired window viewpoint has been established, the lock button can be toggled back to the lock state and the window and the viewpoint captured within will move together. At any point, unlocking the secondary scene allows the user to change the viewpoint in a manner similar to the eyeball-in-hand technique. Once unlocked from the secondary scene, moving the viewing window changes the position and orientation of its viewpoint into the secondary scene. Because it also effectively unlocks the viewpoint, it is natural to tie the 6 DOF viewpoint management technique to the lock button. Clicking and holding the lock button unlocks the secondary scene and initiates 6 DOF viewpoint management until the button is released and the viewpoint is locked in its new position.

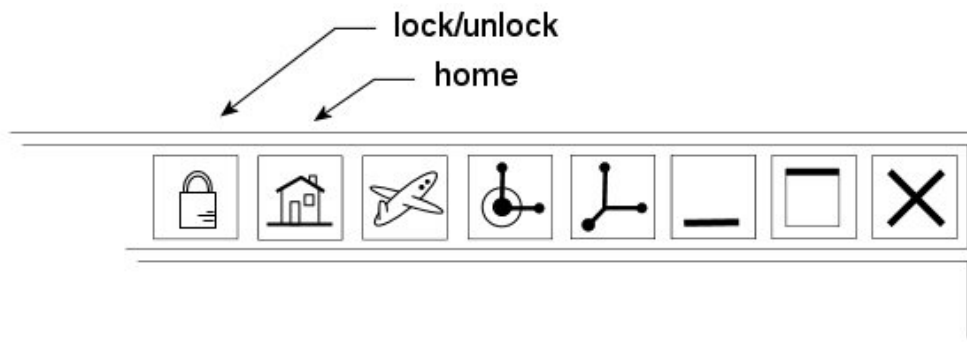


Figure 20. Proposed window icons to register window viewpoint with the surroundings (home) and lock/unlock the window viewpoint from the widow.

Although 6 DOF input is not a requirement for the management of windows, it can add flexibility to how window contents are displayed and how content visible within their viewpoint is captured. While information is often re-organized with respect to window size on desktop interfaces, the ability to compress window content in a horizontal or vertical manner is often limited. In a 3D environment, changing the orientation of the vertical or horizontal axis of a window allows a user to easily compact the information displayed within the window along that axis. This may allow a user to reduce the amount of display space the information in a window consumes. In addition, as it was mentioned before, there are additional ways that viewpoints can be defined when using through-the-lens techniques. Although framing content within the viewing window can still be accomplished with windows that are always oriented towards the user, the ability to more fully control the angle of the viewing window can increase the flexibility of this task. One might use the secondary window like a viewfinder in a camera, merely pointing it at the desired content instead of having to align the content with the user viewpoint. This ability to accomplish tasks while keeping the hands at waist level is a powerful way to reduce user fatigue.

5.3 Shifting to an Exocentric Focus

The third major part of the Withindows framework addresses the subject of user fatigue. Any interaction that requires users to lift their arms from their side for long periods of time is going to prove fatiguing. The key to reducing fatigue is reducing the dependence on interactions with content out in front of the user. Because interaction with 2D content and primary immersive tasks can take place efficiently within floating windows, users can interact with these windows in a comfortable position below their hands. Image-plane interaction with windows under hand should be similar to ray-casting in fatigue and efficiency. The ability to efficiently accomplish

the primary immersive tasks within TTL windows creates the conditions that allow the Withindows framework to shift the focus of 3D interaction to an exocentric one that keeps fatigue at a minimum.

The difference between exocentric and egocentric interaction plays a central role in the nature of fatigue in most physical activities. The human body is designed to do long term interactions in an exocentric context. Placing objects at or about waist level allows a person to work on them with their upper arms at their side and supported by their upper torso. This working position also increases the ability of a person to move their head around the object for different views of the task. Once interactions move beyond reach of the resting forearm or above the shoulder they become fatiguing. In short, any long term interaction that does not allow the upper arm to remain at rest will prove fatiguing. There are a number of ways that the focus of 3D interactions can be moved from an egocentric viewpoint that is in front and away from the user to one that is inherently exocentric. The primary vehicle for this transition is 2D windows floating under the hand of the user.

Windows floating under hand are windows oriented towards the user and located beyond arms reach. The optimal location of the center of these windows is along the line between user viewpoint and the natural starting position for hand related tasks. When standing and using a single hand, the natural position for hand tasks has the upper arm at the sides with the forearm bent downward about 45 degrees at the elbow and pointed about 45 degrees away from the user centerline. When sitting, the natural working position of the hand tends to be with the elbow bent almost 90 degrees and the forearm facing almost forward. These hand positions are not different than those a user would gravitate towards when using a mouse standing or sitting respectively.

5.3.1 Interaction with Egocentric Content

The single greatest problem with image-plane selection technique has been the fatigue associated with using it. For selection tasks away from the user, it should not be used for long periods at a time. In contrast to what has been portrayed in films like *The Minority Report*, future interfaces will not involve long periods of interaction with the arms raised or extended⁷. Content that does not explicitly interact with the physical world should be placed in a comfortable working position when used for any significant duration of time (Figure 21). With image-plane selection on floating windows, traditional programs such as e-mail and web browsers can be used for long periods in an augmented or virtual reality. There are two obvious ways that traditional content can

⁷ <http://www.imdb.com/title/tt0181689>

be extended to interact with the physical world. The first, the mere display of content superimposed onto the physical world, has little effect on user fatigue. The second, interaction between user and the surrounding environment involves acting on content in the physical world. The most elementary hypothetical situation involves a user seeking information about something in their environment. A user can potentially make a single click on a building, sign or product in order to bring up detailed information about it. At this point, if the interaction is going to be of any duration, the user can move the information down to a comfortable working position and continue their perusal.

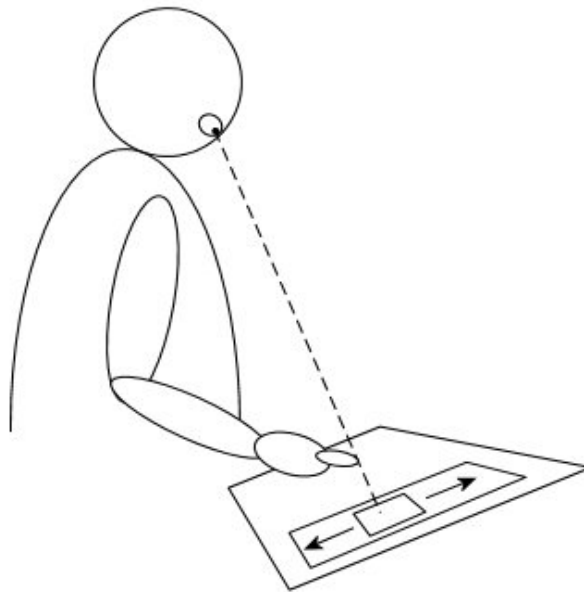


Figure 21. Placing interface elements below the hand lets the user keep their hand in a comfortable working position in fully immersive environments.

A somewhat more complicated scenario might involve repeated interactions with a portion of the environment such as a building. An integral part of any future augmented reality interface will be a thorough model of the environment. This means that a user should be able to create an alternate viewpoint window, frame the building in their view, lock the view and bring the window down to a more comfortable working position. Obviously, from this secondary viewpoint, the user now has more freedom to inspect the building from other viewpoints. The results of their interaction, perhaps a new addition to the building, can remain visible in both

views, and the process becomes an interactive one between exocentric interaction and appreciation of the visible egocentric results.

We have already seen how techniques like ray-casting and Voodoo Dolls avoid fatigue by allowing the user to avoid raising their arms. Even though it uses image-plane selection, Voodoo Dolls keeps fatigue low by allowing the user to bring objects to a comfortable working position for manipulation. Because immersive tasks can be accomplished within TTL windows, users also have an opportunity to move these windows into a comfortable working position to accomplish their tasks. As was described in the previous section, once an object in the environment has been selected, zooming out for a global overview or previewing virtual travel is easily accomplished in TTL windows. In a similar manner, object manipulations in TTL windows can also happen in a comfortable position. The use of image-plane 2½D techniques prevents the user from having to reach into the interaction. The elimination of reaching in manipulation tasks can play a significant role in reducing fatigue.

In the case of more direct manipulation tasks using image-plane grab, TTL windows allow the user to manipulate an object without concern for its distance. A hypothetical object manipulation might proceed as follows. The user selects the object directly in the surrounding environment and then presses the select-focus button followed by the view-all button on a viewing window. This frames the object selected in the surrounding environment within the viewing window. The user presses and holds the lock button to rotate and zoom the viewpoint for optimal manipulation and then proceeds to use image-plane grab or another 2½D manipulation on the object. During the object manipulation, the user can simultaneously view the results of their actions in the surrounding environment. This object manipulation scenario only requires lifting the arm a single time if the object is not already visible within the viewing window. Alternatively, also consider that an object visible in the surrounding environment can be grabbed and dragged into any location within the viewpoint of a viewing window with a single action. Although these interactions only presume the use of a single hand, bi-manual interaction offers the potential for simultaneous manipulation of viewpoint and object. A 6 DOF viewpoint manipulation (most likely with the non-dominant hand) and a 6 DOF or even 2½D object manipulation can take place simultaneously with a two-handed input.

The key to efficient interaction with content at a distance is the ability to manage both object manipulations and the viewpoint from which they are undertaken. Through-the-lens techniques on floating

windows allow the focus of object interactions to move away from egocentric interaction at a distance. While these techniques do not completely eliminate more fatiguing interactions with objects at a distance, they do offer the opportunity for efficiently undertaking primary immersive tasks for long periods of time with minimal fatigue. Both image-plane and ray-casting on floating windows under hand closely emulate desktop mouse use. However, there will be some differences between using image-plane selection and ray-casting under these circumstances that must be considered.

5.3.2 Image-Plane and Ray-casting Under Hand

When interacting with content under hand, either a ray-casting or image-plane technique can be used. The main differences between image-plane and ray-casting in this context are the required working range and the issue of hand occlusion. Because the center of interaction is along the line between the hand and user viewpoint, image-plane selection and ray-casting become very similar techniques as the working range decreases. The study described in the next chapter seeks in part to determine if this assumption holds up. As content moves farther away from the natural working position of the hand, the difference between the physical working range of image-plane and ray-casting increases. Because a ray-casting user need only use their wrist to cover a large range, it might appear that the working range of image-plane becomes significantly larger than that of ray-casting. However, in actual practice, ray-casting users use a combination of wrist orientation changes and hand motion when working on 2D tasks. This behavior is most likely an acknowledgement of the difficulties of executing precise actions in a Cartesian coordinate system with an input device that uses a spherical coordinate system.

The second main difference between the two techniques, occlusion, can have an affect on the ability of the user to execute certain tasks accurately. In the general case, when selecting objects in a 3D environment, the user is not likely to be occluding the target destination with their hand. This means that moving an object to a new location will probably not be affected by the presence of the hand. However, in the case of cascading menus and other drag-and-drop tasks in a smaller area relative to hand size, occlusion can potentially reduce the efficiency of the task. The user study, described in the next chapter, found some evidence that image-plane users were less accurate during drag-and-drop tasks when moving in the direction of the hand. One potential solution to this problem involves increasing the offset between the virtual cursor and the hand. In virtual environments, another solution to the problem is to make the virtual hand semi-transparent in order to minimize its affect on occlusion.

While this subject will be discussed in more detail later, this discussion highlights the potential need for hybrid strategies that allow transitions between either image-plane or ray-casting.

5.3.3 Further Reductions in Fatigue

In addition to the aforementioned advantages of image-plane selection over ray-casting, the flexibility of image-plane can also contribute to reductions in fatigue. Because it is fundamentally a two dimensional task, the selection point within the image-plane can be established with great flexibility. In complement to image-plane techniques that keep the cursor stabilized when the head moves, the cursor position can be adjusted by head orientation and then only changed relative to the image-plane by other means. This technique, first proposed by Forsberg and Zeleznick, allows the gaze of the user to be the main determinant in object selection (89). The position of the virtual cursor within the image-plane can then be determined directly by mouse input or by the absolute position of the hand within its comfortable working range.

It remains to be seen which transfer function between user hand and virtual cursor would provide the greatest efficiency in a virtual mouse condition. A standard mouse transfer function would have the hand moving in a horizontal plane. However, it is likely that the user would be able to mirror the orientation of the image-plane with their hand or finger motions. In this way, a user could easily control not only a cursor that moves with the viewpoint but also the more general clutched virtual mouse cursor in the image-plane. Up until this point, the implementation details of a virtual mouse have been somewhat overlooked. In addition to a mechanism to clutch the re-centering of the input device, a virtual mouse that is working within the full potential visual range of a user would likely require some techniques to aid its use. For example, the current field of view of the user can be used to quickly move the virtual cursor from its original position to the nearest position on the edge of the user field of view. This technique would allow the user to quickly bring the virtual cursor within view and move it over the content of choice.

The clutching action of a virtual mouse can also be used in a more generalized image-plane selection setting. In an immersive environment with 6 DOF tracking, the position of the virtual cursor relative to the hand can also be clutched in three dimensions in order to allow low fatigue interaction with objects out in front of the user. Moving an interaction surface into a position below the hand is not always convenient or even possible. The interaction surface may be world fixed and unalterable or the display environment may be limited to a large

screen in front of the user. In these circumstances, the virtual cursor can be placed over the interaction surface, a clutching function enabled and the hand moved to a comfortable working position. Once clutching is disabled, the motion of the virtual cursor can be made to move relative to the hand in either a strictly translated Cartesian coordinate system or a spherical coordinate system with respect to the user.

5.4 Satisfying Guidelines for Productive Applications

The three main parts of the Withindows framework, image-plane selection, 2½D interaction on TTL windows and moving interactions under hand, combine to address many of the guidelines established in the previous chapter. The Withindows framework directly addresses the need for a low DOF interaction scheme that transitions between desktop and immersive interaction. The framework blends traditional desktop applications with geo-referenced interaction with the physical world. The final guidelines that have not been directly addressed by the framework involve usability, workflow and application development. The unifying nature of the Withindows framework addresses these areas by creating applications that are immediately usable by novices, facilitate cross platform and collaborative workflow, and can be single-authored for either environment.

Applications developed under the Withindows framework will have several traits that directly satisfy the guidelines for productive 3D user interfaces. The most important aspect of the framework is reinforcement of the appropriate roles of 3D immersive viewing and interaction. The framework does nothing to prevent users from viewing 3D content in a first person manner; whether it is egocentric or exocentric, superimposed or completely virtual. The framework does not prevent interaction with objects and interface elements at a distance. What the framework does do is acknowledge the limitations of trying to do productive work on content at a distance with existing techniques. Users can select objects, manipulate objects and view content with easily managed third person viewpoints. This decoupling of viewpoint and interaction allows the user to move their work into a more comfortable working position and keep fatigue levels at a minimum. The second most important aspect of the framework is the ability to transition smoothly between desktop and immersive interaction. Existing desktop applications can easily transition between any number of transitional configurations along the path to full immersive interaction. And, the tight analog between desktop WIMP interfaces and image-plane selection helps them remain usable during this transition. Not only can existing desktop applications be used with minimal fatigue in immersive environments, but immersive applications can also be used on the desktop without any significant

redevelopment. As a result, applications developed within the framework are more usable, facilitate better workflow and are more easily developed.

5.4.1 Usability

It is a certainty that more efficient interaction techniques than those proposed here will eventually be developed. It is a common belief in 3D user interface (3DUI) research that each application interface must be tuned to the specific task of the application. However, the Withindows framework seeks a holistic approach to 3DUI development that can be applied over a broad category of uses. One of the most obvious benefits of using traditional desktop interaction in immersive applications is that computer users will be familiar with the general interaction scheme. As a result, users can focus on the novel aspects of 3D interaction instead of spending time learning basic interactions. The familiar concepts of icon selection and cascading menus allow advanced users to access the large number of system commands that modern applications require. By adding logical extensions to existing desktop window management, the framework makes 3D interaction a superset of typical 2D interaction.

The familiarity with desktop applications also extends into the tools provided for primary immersive tasks. Using 2½D interaction techniques within 3D environments will be familiar to any user that has done 3D modeling or scene development on the desktop. This interaction scheme not only makes interacting in 3D more efficient, it also creates the conditions where immersive applications can be easily introduced to users on the desktop. Because the primary immersive tasks of object selection, object manipulation and global search happen completely within a window using 2½D interaction, that window can be presented unaltered on the desktop. A user can then select, manipulate and view content just as they would immersively while still on the desktop. By directly mapping the mouse to the location of the virtual cursor over the alternate viewing window, a desktop user can easily accomplish the same interactions on that window while on the desktop. This symmetry between desktop and immersive interaction allows novice users to become acclimated to the immersive application on the desktop. As a result, the menu structure, functionality and idiosyncrasies of the immersive application can be familiar to the user before they ever use the application immersively. This desktop exposure strategy has several advantages. First, immersive equipment such as HMD and CAVE projection systems are in short supply. A large number of persons can become familiar with an immersive application without the need for increased immersive resources. When users are exposed to the immersive application they will then be able to spend more time focused on the idiosyncrasies of immersive interaction than on the functionality of the application.

The second benefit of this strategy involves the resolution limitations of current immersive technology. Head mounted displays with resolutions greater than XGA, 1024 x 768 pixels, are currently very expensive. As a result, when interface elements must compete with other environmental content the resolution allotted to them becomes quite limited. Although interface elements such as icons and sliders can often be rendered in reduced resolution, the rendering of menu and button text becomes especially problematic in low resolution environments. Further complicating the problem is the fact that these interface elements may not be completely aligned with the display device. Text and other interface elements that are affine distorted become even harder to display clearly in low resolution environments. It is already well understood that our brain routinely compensates for the limitations of our own vision system. Even though we can only see color in a limited part of our eye, the world around us appears in colors not actually seen but generated within our mind (90). The potential advantage of being able to expose users to immersive interfaces on desktop systems is that user familiarity with menu and button text will increase the ability of users to perceive them under low resolution conditions.

5.4.2 Workflow

The Withindows framework facilitates productive workflow for both individual and collaborative work in several ways. The typical workflow pattern for immersive applications has been very inefficient. Immersive application designers cannot develop content directly within the virtual environment, changes made immersively cannot be accessed on the desktop and the desktop cannot be used to offload any portion of the immersive workflow. Not only can users become familiar with Withindows based applications on the desktop, but they can also do any portion of their work in either environment. This ability to move between environments also creates opportunities for unique collaborative work environments with both desktop and immersive users working at the same time. Furthermore, the unique nature of keeping interface elements within floating windows addresses many of the important criteria for successful computer supported collaborative work (CSCW).

In order to talk about workflow with regards to immersive applications, we first have to consider the different domains of immersive applications. There are basically three application domains for such computer applications; simulation, visualization and development. All immersive applications centered around simulation and visualization must be developed themselves at some point. The workflow for building a simulation or visualization application typically involves a cycle of using desktop tools to build the application followed by testing

the application immersively. The problem with this process is that the developer must move back and forth from desktop to immersion in order to evaluate their work. Obviously a more efficient development cycle would allow the user to evaluate their work immediately. And, once they have found a necessary change, the user should be able to execute the change while evaluating the application. This is effectively the definition of What-You-See-Is-What-You-Get (WYSIWYG) development environments. The actual use of simulation and visualization applications often involves both desktop and immersive tools. However, the functionality of these tools and their interfaces are almost always different between desktop and immersion. Because of the difficulty of designing and implementing usable 3D interfaces, the user typically has more functionality on the desktop than when using the immersive application. The domain of immersive development involves applications for creating 3D content; either 3D models or interactive worlds. Because these applications have usually been research applications, they rarely have any close correlate on the desktop. In cases where there has been a corresponding desktop application, none have provided either the same functionality or interface.

Applications built within the Withindows framework are guaranteed to have the same interface in both environments. A 3D modeling package, a virtual world builder or a scientific visualization tool that uses alternate views within TTL windows will be immediately usable on the desktop and allow access to the same functionality. Conversely, a desktop application for creating 3D content can be easily extended for use within an immersive environment. Increasingly, desktop modeling packages like Maya are allowing for the immediate visualization of their content in a stereo head tracked immersive environment. A fully realized Withindows implementation is not significantly more than the appearance of the desktop Maya application window within that immersive environment. Given the proper display technology, a user of such a system could conceivably work at their desk on the desktop interface and simultaneously view the results in the environment surrounding them. Then, when desired, the user could literally stand up, grab the desktop interface and begin working immersively.

The advantages of this type of workflow extend beyond a single user work environment. Through the use of a collaboration server, applications can allow two persons to work on the same data set simultaneously. The opportunities for such collaboration are not often available on the desktop. It is hard to imagine the need for two users to work on the same excel or word document at the same time. However, in the case of developing 3D environments, when the results of the work are to be experienced immersively, the benefits of having one person on the desktop and one person within the content become obvious. The desktop person will likely have the

greater ability to make detailed changes to objects and their behaviors while the immersive user will likely benefit from being able to directly engage those objects and suggest modifications to their behaviors. The only system that has attempted this sort of collaborative workflow used a much simpler and fundamentally different interface for the immersive user (91). As with any collaborative application, keeping each user aware of the actions of the other user is a priority. The Withindows framework also delivers significant advantages with respect to mutual awareness.

In addition to having a single user on the desktop, another possible collaborative configuration involves two users working simultaneously in a tele-collaborative virtual environment. In these types of environments each user needs to remain aware of the actions of each user. This is difficult to do when interface elements are overlaid directly onto the display device instead of residing directly within these worlds. These tele-collaborative environments have typically been implemented using what is called global WYSIWYG. In this configuration, confusion about the state of the world and the user are avoided because every change made by the user on the world is visible to the other users in the environment. Users typically are represented by an avatar which gives other users an idea of the viewpoint of the user. This implementation has also made good use of ray-casting. A virtual hand with a ray emanating from it allows other users to see what the user is trying to interact with. However, there are several problems with global WYSIWYG. First, an important element in CSCW environments is the ability of individual users to have subjective viewpoints that may or may not be the same as their other collaborators (92). The second problem with typical solutions for tele-collaboration is that users routinely find themselves working in different locations within the environment. Therefore, when separated or even turned in another direction, users cannot remain aware of the viewpoints or actions of their counterparts. As a result, much of what passes for work in these environments involves trying to ascertain what other users are seeing and doing (93). Finally, a global WYSIWYG setup has problems with 2D interface elements such as menus because their readability becomes compromised from large distances and oblique angles.

The Withindows framework simultaneously solves many of these problems. First, because they are working interface elements within the world, the full system state of windows and their widgets will remain visible for other users to see. Even in a situation where there is a desktop user and an immersive user, the interface window of the desktop user and virtual cursor actions on it can potentially be viewed by the immersive user and vice versa. Secondly, because users do not need to move to another virtual location in order to search the

environment, they can actually work side by side or in circular arrangements. Working in close proximity allows users to easily view and remain aware of the actions of one another. While not utilizing the virtual travel mechanisms may seem counter to the individual freedom associated with virtual environments, multiple users can still take turns teleporting the group to among locations or even executing virtual travel techniques for the group as a whole (i.e. point-and-go). Finally, if not already obvious, the Withindows framework solves the problems of subjective viewpoints. Each user can maintain their own subjective view of the content in front of them while a separate global viewpoint of the world can be maintained for the group in the surrounding environment.

5.4.3 Application Development

The final set of guidelines addressed by the Withindows framework involves the development of applications that can transition between immersive and desktop use. There are obvious benefits to using established 2D graphical interface elements within virtual environments. There is already a rich lexicon of input means devoted to capturing symbol input. These elements are the product of a long history of development and refinement. Creating an infrastructure that supports 2D interface elements and window management within immersive settings is a natural stepping stone for desktop operating systems that are increasingly becoming aware of 3D content. Yet, the most important part of application development using the Withindows framework is the ability to single author both desktop and immersive applications for use in any transitional configuration in between. Finally, another benefit of using a low DOF input method like image-plane selection is support for the principles of Universal Design.

5.4.3.1 Symbolic Input

Along with the established gamut of interface elements for desktop environments comes a very well established set of design rules. Important topics such as menuing, cursor management, color selection, visual reinforcement and sonic notification all contribute to the productive success of current graphical user interfaces. Design rules for graphical interfaces have been combined with a rich set of tools for specifying symbolic information. The buttons, menus, sliders and 2D point selection used today are the results of several decades of design iteration. If it can be executed successfully in immersive environments, the existing lexicon of input methods is a natural starting point for the manipulation of symbolic information.

There will continue to be substantial differences between the usability of an operating system window on the desktop and one floating in free space. However, the adoption of existing desktop standards can only help to highlight where those differences lie. One obvious divergence point is with the desktop keyboard. Regardless of whether finger motion can be captured, keyboard input requires some form of physical surface. Yet, in the context of the ability to use menus, sliders and file managers immersively, the role of speech recognition becomes more obvious. As it has on the desktop, speech recognition will continue to find success in executing learned system commands and producing text from unorganized dictation.

5.4.3.2 Window Management

Leveraging existing GUI standards and the use of application windows in 3D opens the door for desktop application developers to easily make the transition into creating content for these environments. With Windows extends the functionality of windows within virtual environments to include not only traditional minimization and full-view functionality but also unique viewpoint related functionality with geo-referenced content. These extensions give developers a framework within which they can intuitively augment the functionality of their own applications.

The extensions to window management proposed here are logical steps forward for window management. It is only a matter of time before desktop operating systems will have to deal with windows that are assigned 3D positions in space. Already, the latest version of the Windows operating system, Vista, includes functionality for presenting windows in a 3D arrangement and for projecting their content onto 3D objects⁸. Increasingly, desktop systems are becoming aware of the devices around them. Bluetooth already allows a computer to recognize when a device is near and begin communication with it. It is only a logical step forward for the operating system to understand exactly where in physical space it and the devices attached to it reside. The principles of the Windows framework serve as a template of how extending windows into three dimensions can increase the flexibility within which they are used. Adding 3D position to desktop windows and a generalized image-plane model would be a relatively easy addition to frameworks such as XWindows and the Windows Foundation Classes. With 3D positions incorporated into existing applications, there is the potential that applications that were never designed for immersive use can find use in future 3D environments without any modification whatsoever.

⁸ <http://www.microsoft.com/windows/products/windowsvista/features/experiences/aero.mspx>

5.4.3.3 Global Versus Viewpoint Functionality

There is a natural dividing line between functionality that should remain completely associated with viewing windows and functionality that also applies outside of windows. The dividing line is between functionality related to viewpoint management and all other application functionality. Functions such as teleportation to a surface, object selection and manipulation apply to objects in the environment. Functions such as pan, rotate, zoom and teleportation to a viewpoint apply only to the viewpoint within the window. This separation helps guide the design of the viewing window interface. Icons and menu items related to viewpoint management are appropriately located on the viewing window. This group of functions includes pan, rotate, zoom, view-all, select-focus, view-top, view-right, etc. Global functionality such as teleportation to a surface and adjusting object properties are more appropriately associated with the objects in question.

The separation between viewpoint and global functionality is by no means a strict one. Our hypothetical immersive Maya application would have a mixture of object oriented functionality, viewpoint management functionality and numerous icons and drop-down menus offering global functionality. The object oriented functionality in Maya involves right-clicking to bring up a menu at the current cursor position. When it appears in the main application viewing window, it is appropriate to align this menu with the window surface. Because it is context sensitive and unrelated to the viewpoint, there is no reason this menu cannot appear outside of the viewing window. When it does appear outside, the menu can appear oriented towards the user just as any window would be.

In addition to context sensitive menus, modeling applications such as Maya often have portions of the application window dedicated to presenting object related information or some form of object hierarchy browsing. These display areas are presenting global information and in a strict sense are not associated with any viewpoint. Contemporary desktop application design often allows these areas to be moved off of the main application window and become separate application windows whose location and size can be individually managed. As a result, this sub-window would become a global object in our hypothetical application that is associated neither directly with the viewing window nor the surrounding environment. In this context, as is typical on desktop systems, it is appropriate that the sub-window lack any functionality associated with viewpoint management.

5.4.3.4 Single Authoring

The ability to move applications from one computing environment to another without redevelopment is known as single authoring. Single authoring is usually a term applied to the development of desktop and mobile applications. By creating generalized representations of user interfaces, a mobile application can more easily transform the interface into one appropriate for a device with limited screen space or reduced functionality (94). While a desktop application might have the CPU power to drive a voice recognition system, a mobile device may have to forgo such affordances in favor of simple menu selections. While a desktop application can present multiple GUI widgets and associated animations on the same page, a mobile device may have to resort to presenting each widget in succession on a small black and white screen. The discipline of single authoring has never been applied to the transition between desktop and immersive applications. The development of 3D user interfaces has always been considered too specialized for such consideration. Yet, just as interfaces can be reconfigured for lower resolution devices, they can also be reconfigured for the different capabilities of immersive display environments. While HMD configurations rarely have more than XGA resolution, projection and multi-LCD panel solutions frequently have significantly higher resolution. Furthermore, under the Withindows framework, in addition to the desktop affordances, applications can also find themselves with additional capabilities like 6 DOF input. Borrowing from the efforts of prior single authoring work, the Withindows framework can alter interfaces in response to different device characteristics and takes advantage of additional resources to increase functionality instead of replacing core functionality.

One way that Withindows facilitates single authoring has already been discussed above. Given the proper development environment and sufficient resolution, desktop applications can be brought into an immersive environment without any redevelopment. The framework also supports single authoring for applications that are designed for both desktop and immersive work. The example above of how to potentially use Maya in both environments is an example of this type of single authoring. Moreover, Withindows provides a general framework for the development of immersive applications that can also be used on the desktop. Simply encapsulating all of the immersive functionality within an alternate viewing window allows that functionality to be presented and exercised on the desktop. This design does not restrict the ability of users to exercise functionality outside of the viewing window, it merely calls for all functionality to be available via those windows. The ability to select objects can easily occur both within windows and in the surrounding environment. Techniques like image-plane grab and even 2½D object manipulation can still be applied to objects out in front of the user. And, an image-plane

teleportation scheme that operates on surfaces within windows is easily used outside in the surrounding environment.

5.4.3.5 Universal Design

Universal Design is a broad movement to include features in products and devices that allow the greatest number of persons to use them. The movement is focused on the design of products that are accessible to disabled persons without the need for assistive devices. Universal design principles encourage the design of products that are flexible, require low physical effort and are simple and intuitive. By addressing these three guidelines, the Withindows framework encourages the development of augmented and virtual reality (VR) applications that can be accessed by the widest range of persons. The unique low DOF nature of image-plane selection allows users to control virtual cursor position by any number of means. The use of floating windows placed under the hand creates conditions that allow even low mobility users to interact with them. And, the familiar desktop interaction metaphor extends the concept of accessibility not only to the disabled but also to those who lack prior exposure to 3D user interfaces.

The Universal Design movement is a response to the segmentation that can occur when products and devices are altered to allow persons with abilities outside the norm to use them. For instance, although a ramp might allow wheelchair access to a building, it might result in disabled persons entering the building through a separate entrance. The Universal Design movement seeks to avoid the stigmatization of disability by applying a doctrine of inclusiveness to design. The Universal Design movement has a set of principles that includes seven guidelines for product and device design. Supporters of Universal Design believe that these guidelines apply equally to accessible design and to sensible design. An interface to virtual and augmented reality that is accessible by persons of limited physical ability makes perfect sense when we consider the where the technology is going. The ability to interact with and control the devices around ones self with a single interaction modality could be of significant value to disabled persons. Of the seven guidelines, three are addressed directly by the Withindows framework; Flexibility in Use, Low physical effort, and Simple and intuitive.

Reliance on a low degree of freedom input mechanism helps the Withindows framework foster the development of immersive applications that are easily used by a broad range of users. Because it is inherently a two dimensional input method, the virtual cursor used by image-plane selection encourages flexibility in use by

hand, mouse or any other 2D input method. In addition to brain-body interfaces, the cursor selection methods described above that use a combination of viewpoint and 2D input could be used by users that have limited body control. In addition to the flexibility of input method, the flexibility of the presentation medium can also bring advantages. Because they do not rely on any fixed location with respect to the body or the environment, 2D floating windows can be located in any number of locations or even placed on large format devices for users with limited eyesight. Users without sight might even find it convenient to move 2D content onto haptic rendering surfaces similar to those available today (95). Of course such an advantage would be obviated by a true haptic interface, but then again, the technology for such a device has not been developed.

The means by which the Withindows framework fosters low physical effort have been covered in previous sections. It is important to recognize that keeping fatigue at a minimum has implications not only for long term work with immersive applications but also for the very usability of such applications by users with less than normal physical ability. The Universal Design movement would likely reject the corporealism that has influenced prior 3D interfaces as based on insensitive assumptions about the physical abilities of their users.

Beyond opening up immersive use to the physically disabled, the Withindows framework also opens the door to a broad class of users already familiar with desktop computers. It has been argued that 3D user interfaces will become more simple and intuitive when they mirror the physical world that we already know (39). The argument can just as easily be made that any 3D user interface is most naturally understood as a computer interface instead of an interface to any true physical world. Giving users affordances in these environments that they are already familiar with can go a long way towards increasing their adoption. It is common knowledge in the field that the two main factors holding back broader adoption of immersive technology come down to hardware and software. The technological hurdles of displays and tracking will be solved in time. And, immersive environments focused on entertainment and simulation of the real world will continue to thrive on their intuitive interaction. But, the fundamental usability limitations that have plagued productive immersive environments will require a different kind of novice usability that is simple and intuitive if they are ever to find broad adoption.

6. EXPERIMENTAL VALIDATION OF THE FRAMEWORK

The Withindows framework proposed in the last chapter has a number of theoretical advantages. However, many theoretical frameworks have been proposed that do not find eventual implementation. The success of the Withindows framework will ultimately dependent on whether developers and users feel it actually improves their experience. However, some individual aspects of the framework can also be evaluated formally. This chapter describes the design of a user study to evaluate the validity of two major assumptions made by the framework. The first assumption is that image-plane selection will improve user interaction with traditional 2D widgets presented at an arbitrary position and orientation with respect to the user. The second assumption is that image-plane selection on 2D widgets presented under hand is roughly equivalent to ray-casting under the same conditions. This chapter describes the design and results of a user study to address these two assumptions. After an introduction to describe the assumptions the study is based upon, the following sections will describe the study design followed by the empirical results. The last two sections discuss the results of the study and the future research that those results indicate.

The Withindows framework is composed of three main ideas, each of which have individual components that can be evaluated in an experimental setting. The framework consists of image-plane selection on traditional interface elements, alternate viewing windows for immersive tasks and the shifting of these tasks to an exocentric viewpoint. One of the most important implications of the Withindows framework is that users will be using image-plane selection for most interactive tasks. This puts a premium on evaluating whether the claimed advantages of image-plane selection actually translate into increased usability. The user interactions proposed by the Withindows framework can be broken down into two main scenarios, both of which involve traditional 2D widgets. The first scenario involves interacting with traditional desktop content and TTL viewing windows that are located under the hand of the user. The framework relies on the assumption that users can use image-plane selection in these conditions efficiently and for long durations. The second major scenario places selectable objects and their associated widgets on surfaces some distance away from and at oblique angles with respect to the user. The Withindows framework argues that image-plane selection has a number of advantages over ray-casting on these types of surfaces.

The Withindows framework proposes that placing immersive application windows in a working position below the hand will lead to interactions that are efficient and low in fatigue. It is reasonable to assume that windows floating under the hand of the user should allow an interaction style that is analogous to desktop mouse operation. Ideally, given adequate display resolution, these windows can be kept small enough that they allow the user to keep their hand within a comfortable working area. The framework argues that image-plane techniques on these windows are no different in fatigue and efficiency than ray-casting. It appears obvious that placing windows under the hand will lead to a user experience that minimizes the fatigue of image-plane techniques. And, it is also reasonable to assume that the ability to stabilize the arm next to the torso will contribute to maximizing the accuracy of image-plane interaction under hand. If this style of interaction with 2D widgets is to prove usable then it must cause no more user fatigue and be no less efficient than any other possible configuration that uses ray-casting. The optimal location to place windows using ray-casting is not as obvious. The view alignment problem states that in the absence of proprioceptive cues the difficulty of manipulating a target with ray-casting increases as the alignment between hand and eye decreases. This would suggest that aligning the hand with the eye offers the best opportunity for efficient ray-casting operation under all conditions. For this reason, it is reasonable to assume that using ray-casting on windows below the hand is no less accurate than on windows located at any other location. Furthermore, ray-casting users tend to use a combination of both wrist movement and arm movement when interacting with 2D interface elements. For the same reasons cited for image-plane selection, this suggests that placing the content below the hand of the user will minimize user fatigue when using ray-casting.

The efficiency of both desktop and immersive user interfaces is typically evaluated in the context of object selection and object manipulation. Timed selection tasks are of obvious utility as they give a good indication of the efficiency of the input device. For 2D interfaces, object manipulation tasks typically resolve to drag-and-drop tasks where the dependent variables are the time to completion and the accuracy with which the object can be positioned at the destination. However, the ability to accurately follow a trajectory throughout the task is often important when using common interface elements such as cascading menus. As a result, a path following task that requires accuracy throughout is a better measure of the efficiency of an input technique for general 2D widget use. The study described in this chapter uses a simple object selection task and a path following task on 2D surfaces floating under the hand of the user. The Withindows framework proposes that image-plane techniques on 2D widgets are an effective interface even when those widgets are positioned on surfaces in the environment.

The framework argues that image-plane selection is insensitive to changes in depth and robust to affine distortions. This suggests that image-plane is more efficient than ray-casting when interactions occur on oblique surfaces at various depths with respect to the user. In order to compare image-plane and ray-casting under these conditions, the study described here places 2D interaction surfaces at varying depths and orientations with respect to the user in a separate experimental condition.

The Withindows framework relies heavily on assumptions about the superiority of image-plane selection over traditional ray-casting. While many of the remaining advantages of image-plane are theoretically sound, there has been little empirical evaluation to determine their validity. Bowman found that image-plane selection was somewhat slower than ray-casting in a testing environment that did not include occluding objects (75). Later, Wingrave found that image-plane was faster than ray-casting for object selection in more spatially dense environments but did not publish evidence that the difference was statistically significant (74). Bowen also found that object manipulation techniques using image-plane were faster than similar techniques that used ray-casting as a selection technique. Pierce established that his Voodoo Dolls technique, and its use of image-plane selection, was overall no slower or more fatiguing than the HOMER technique when both selection and manipulation of objects was considered. Pierce also reported that users found selecting distant objects less difficult with image-plane selection than with ray-casting. While Wingrave found no clear preference for either technique, most researchers have reported that users prefer ray-casting because of its low fatigue. Mulder found that image-plane selection was faster than and preferred over ray-casting when used for menu selections but did not publish a statistical analysis of the significance (96). Wingrave evaluated menus using image-plane selection, but the menus were fixed to the image-plane and ray-casting was not involved in the study (36). Lee reported that image-plane was less accurate and slower than ray-casting, but the study did not track head motion and therefore was not a strict implementation. Lee, McKenzie and Vogel have all found evidence for the superior speed and accuracy of mouse-based techniques over ray-casting (78,79,97). There is no research to date regarding the unique strengths of image-plane selection on oblique surfaces or the accuracy of the technique at greater distances. Ware found that users tolerated using a virtual cursor exclusively in their dominant eye during an image-plane selection task (68). Forsberg also presented a cursor exclusively in the dominant eye but left his aperture circle visible in both eyes for his selection technique (98). MacKenzie introduced the terms movement variability, movement error and movement offset for measuring the accuracy of path following during tasks such as menu use (73).

This chapter will describe an experiment conducted to verify some of the claims made in the last chapter about image-plane selection. The goal of the study was to determine if both image-plane selection and ray-casting can be used with the same efficiency, fatigue and ease of use when used under hand. Under separate conditions, the study also sought to further verify the speed advantage of image-plane in dense environments and to determine if path following on surfaces at oblique angles and varied depths is superior to ray-casting under the same conditions. The following sections will describe the hypotheses formulated for the study along with the experimental setup. The discussion will then turn to results that showed the hypotheses are not valid under all conditions. While the results found that image-plane and ray-casting do produce similar fatigue levels, there was some eye-strain associated with the selection task that skewed the ease of use scores. The eye strain results suggest that using a dominant eye cursor was not well tolerated under all conditions. The study results on surfaces located at a distance from the user did not establish a significant efficiency advantage for image-plane. The study did not find a significant increase in accuracy or speed due to using image-plane on oblique surfaces presented at various depths and angles with respect to the user. A post-hoc analysis of the data did show some significant increases in efficiency on some groupings of surfaces. However, the results were not statistically valid due to the lack of proper balancing between subjects. Overall, the study validated the assumption that image-plane fatigue and ease of use on surfaces under hand is roughly equivalent to ray-casting under the same conditions. Furthermore, while it did not demonstrate an improvement in efficiency on 2D surfaces away from the user, the study did find that image-plane is at least no less accurate or efficient than ray-casting under the same conditions.

6.1 Study Design

The study was designed to test two main hypotheses; that the image-plane selection technique is not significantly different than ray-casting when used on 2D surfaces under hand and is superior to ray-casting when surfaces are located at various distances and orientations with respect to the user. The study used a within subjects design under two separate conditions, interaction under hand and interaction at-a-distance, each independently testing one of the main hypotheses. Twenty four subjects performed 20 simple object selection trials and 20 path following trials using each selection method under each condition for a total 3840 total trials. The object selection task measured selection time while the path following task measured task completion time, the average deviation from the path and the standard deviation of that error. Each subject answered a

questionnaire after each task block to assess their perceived arm fatigue, eye strain, motion sickness and ease of use.

For the simple selection task, the subject was presented with one of sixteen different icons and asked to select the matching object from four choices on the interaction surface. For the path following task, the subject was presented with an icon and its outline connected by a path of the same width. The study used a 1 wall rear projected passive stereo display system with an Ascension tracking system and an 8x10 foot screen. In order to present the interaction surface below the hand of the subject, each subject performed the entire study on top of a three foot platform. The at-a-distance condition used the same two tasks but moved the interaction surface between eight different positions in the environment. The following sections describe the study design in more detail.

6.1.1 Hypotheses

The study sought to address two main hypotheses, each of which can be broken down into sub-hypotheses. The first main hypothesis is that image-plane selection on 2D widgets floating below the hand of a user is not significantly different than ray-casting under the same conditions. This hypothesis is a null hypothesis (H_0), known as the skeptic's hypothesis, because we are seeking to reject the finding that image-plane is significantly different than ray-casting. The other main hypothesis of the study is that image-plane selection is more efficient than ray-casting in environments that present 2D widgets at various distances from the user and at various orientations with respect to the user. Various qualitative and quantitative measures contribute to break the main hypotheses down into six sub-hypotheses.

The first sub-hypothesis is that subjective user arm fatigue is not significantly different when using either image-plane or ray-casting underhand. The second sub-hypothesis is that eye strain and motion sickness are not significantly different when using either technique. This sub-hypothesis is related more to the visual effect of using image-plane selection in a stereo display environment than to the physical execution of the task. These two sub-hypotheses support the third sub-hypothesis that overall ease of use is not significantly different between the two techniques. The fourth sub-hypothesis is that quantitative measures of efficiency such as time and accuracy are not significantly different for either interaction method when used under hand. The last two sub-hypotheses are related to the second main hypothesis. There are two primary tasks that allow efficiency to be measured; simple

object selection and path following on 2D surfaces. The fifth sub-hypothesis is that simple object selection with image-plane is faster than ray-casting in environments where object selections occur at varied distances. The sixth sub-hypothesis is that path following on surfaces at oblique angles to the user viewpoint and at varied distances from the user is more efficient when using image-plane than ray-casting.

6.1.2 Experimental Design

The study was a full factorial design within subjects study with repeated measures using either image-plane or ray-casting as the selection technique. The two techniques were presented under two different experimental conditions, with the interaction surface placed below the hand of the user and with the interaction surface positioned at various distances and angles with respect to the user. The two techniques and two conditions combined for a total of 4 treatments that each subject was exposed to. For each of the 4 treatments the subject completed a single block of trials doing a drag-and-drop task followed by a single block of trials doing an object selection task. Each block of trials began with 5 practice trials followed by 20 measured trials. As a result, the total number of data points collected for each of the two conditions was 1920. The study population consisted of 24 paid subjects, 4 of whom were female, with an average age of 23.7 years recruited by web postings and campus flyers. Eight subjects reported having prior VR experience. All subjects were tested for color blindness, dominant handedness and dominant eye. All of the subjects were right handed and one subject was color blind. Right eye dominance was found in 13 of the subjects.

Each subject was presented two treatments of the underhand condition followed by two treatments of the “at-a-distance” condition. The underhand and at-a-distance treatments presentation was balanced using a Latin squares arrangement so that each treatment ordering was repeated each 4 subjects for a total of 6 times over the 24 subjects. Each subject was presented with the same 5 object selection or drag-and-drop practice trials before each block. The 20 measured trials in each trial block were randomized into 4 orderings, each of which was presented to the subject over the course of the 4 treatments. Both the underhand and at-a-distance experimental condition trial orderings were arranged in a Latin squares arrangement such that each ordering was presented twice over the course of 24 subjects. Both the object selection task and the drag-and-drop task were confined within a framed interaction window.

The study was conducted in a large screen stereo display environment using a 6 DOF magnetic tracking system. The display system used a passive stereo rear projected SXGA resolution setup with circular polarized filters. The screen from Stewart Filmscreen was 10' wide and 8' high and located directly on the floor. The tracking system used was an Ascension Flock of Birds magnetic tracker. A WANDA™ controller with three buttons was used to capture button inputs and the position of the user hand⁹. The Ygdrasil VR development environment was used to develop the testbed environment on a 1.5Ghz dual Pentium computer running the SUSE 10.0 operating system.

6.1.3 Task Details

The drag-and-drop task was designed to simulate the type of path following common to cascading menu interaction. The subject selected a colored shape by pressing and holding a controller button and moving the shape along a marked path until it rested within a white outline of the shape (Figure 22). The subject clicked on a play button to begin each trial and pressed a stop button after completing the trial. Subjects were allowed to rest between trials and the total time between pressing play and stop was measured for each trial. The mean cumulative error, or movement error, between the indicated path and the actual trajectory was calculated until the target first entered the destination outline. In addition, the standard deviation of the error along the path, known as the movement variability, was measured for each trial. Subjects were asked to complete the trial as accurately and quickly as possible. As an added incentive, a running total of earned money was maintained for the user. Each successful object selection trial garnered \$0.10 while trial time and accumulated error combined to subtract from the total monies.

⁹ WANDA is a trademark of the Board of Trustees of the University of Illinois.

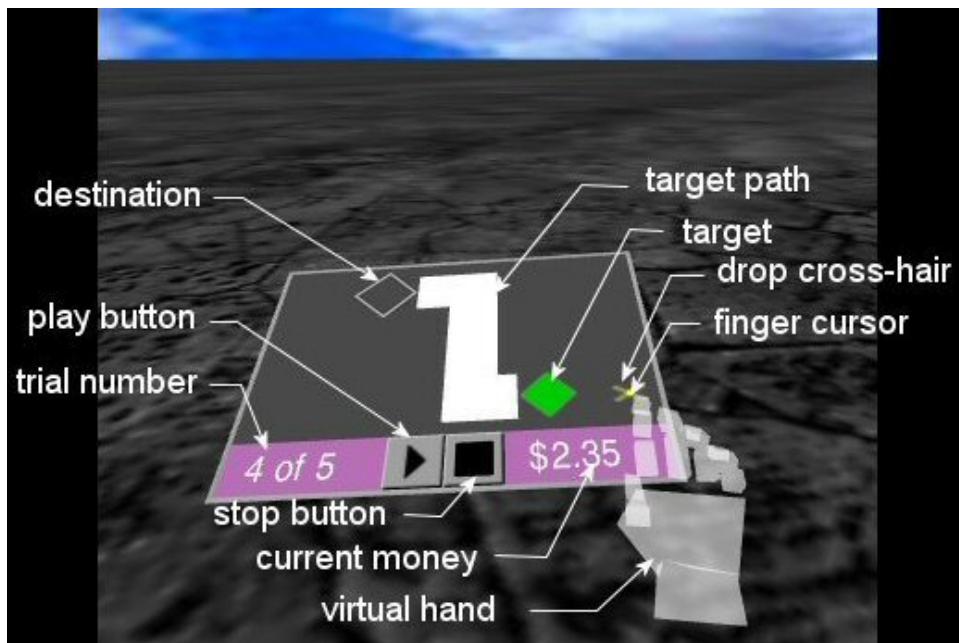


Figure 22. Drag-and-drop task underhand shown with the image-plane technique.

The implementation of the image-plane technique rendered a yellow sphere at the tip of the virtual finger in the dominant eye only. The virtual hand was presented with a transparency of sixty percent. Each subject was tested for eye dominance prior to participation by asking them to look at a target through a small aperture. Instead of reinforcing the yellow sphere on the surface, a drop cross-hair was rendered on the task window when the finger cursor was over it. Because the drop cursor was located at the same distance as the selection surface, it could be rendered in both eyes without causing selection ambiguity. The object selection task presented 4 colored shapes in the interaction window and a target shape located on the ground several feet in front of the user (Figure 23). When the user clicked on one of the 4 shapes, a white outline indicated the selection. Each successful selection that matched the target shape garnered \$0.05 while trial time subtracted from the total monies.

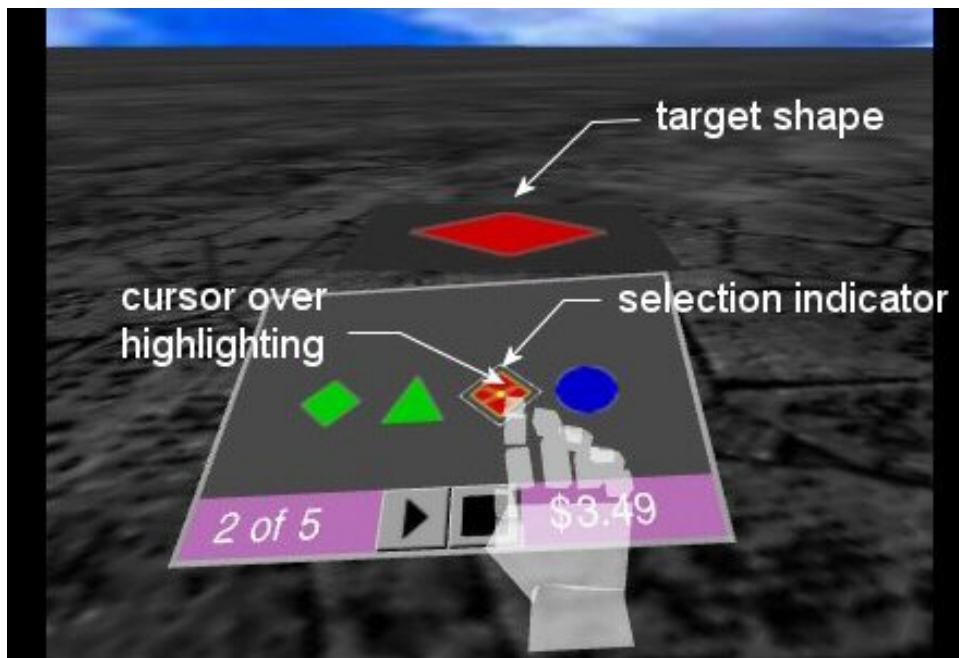


Figure 23. Object selection task underhand shown with the image-plane selection technique.

After each block of either object selection or drag-and-drop trials, the subject completed a 4 question survey administered using the object selection technique of the current treatment. The following questions were presented on a 5 point Likert scale within the interaction window: arm fatigue, eye strain, motion sickness and ease of use. Lower numbers corresponded to lower fatigue, eye strain, sickness and difficulty of use respectively (Figure 24). The selected number button was grayed out to indicate a successful selection. The play, stop and continue buttons did not use a drop cursor, but indicated their selection by highlighting the button background during cursor over events.

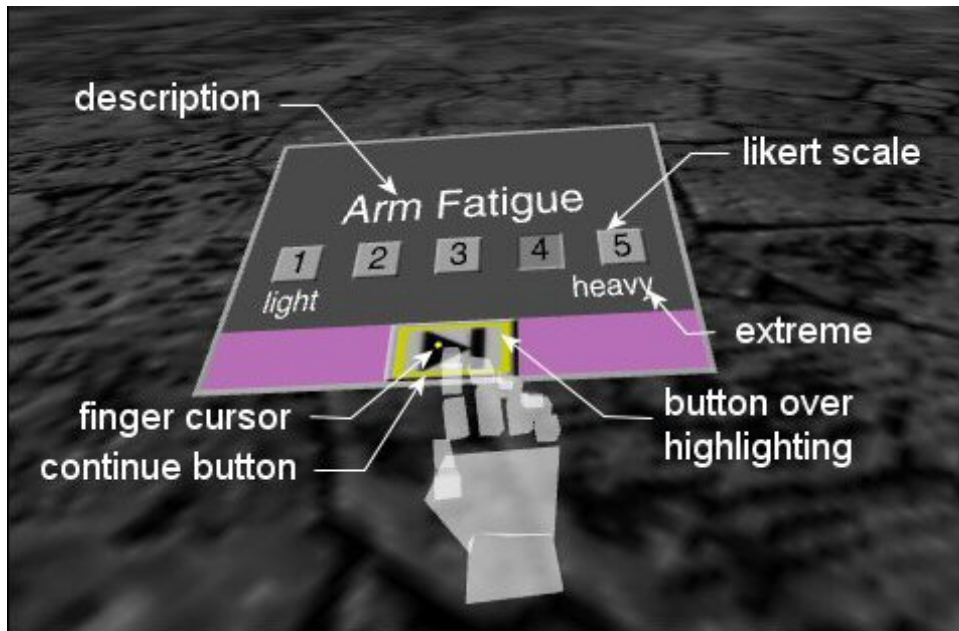


Figure 24. Questionnaire presented underhand shown with the image-plane selection technique.

6.1.4 Experimental Conditions

The study was conducted under two separate experimental conditions, under hand and at-a-distance. The virtual image was presented approximately 4 feet underneath the hand of the user in a fixed location during the first two underhand treatments. In order to generate a virtual image under the hand of the user, a 3' x 4' x 3' stand was built (Figure 25). The user was turned approximately 20 degrees away from their dominant hand in order to ensure that their legs did not interfere with the image presented on the screen or their ability to drop their hand into a comfortable position. This allowed the given screen space to be optimized for interactions with content displayed bellow the dominant hand.

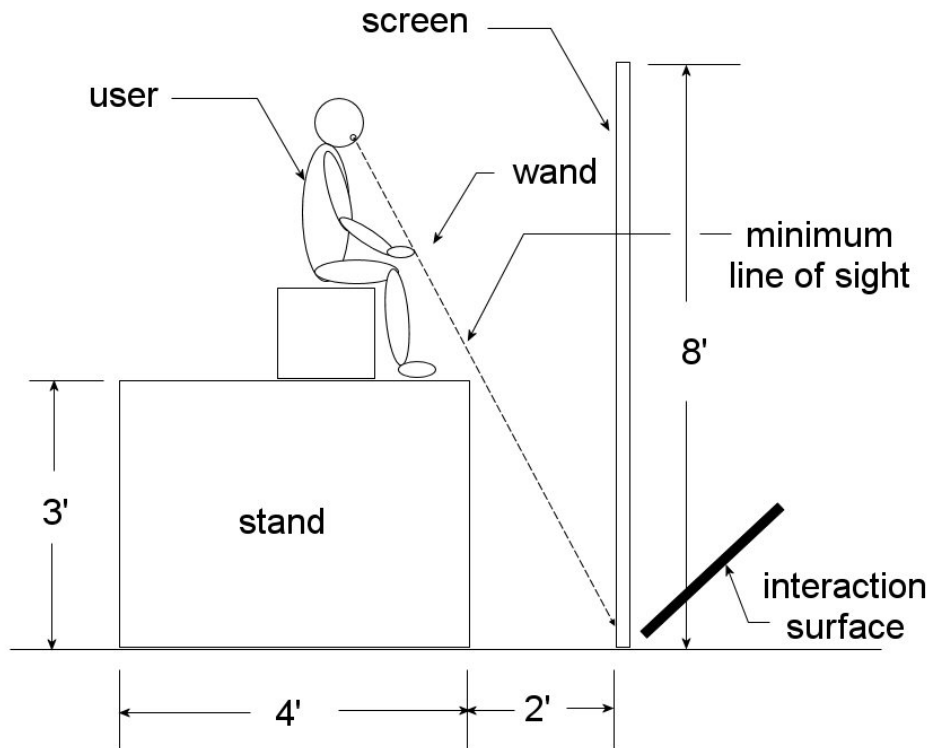


Figure 25. Testing apparatus for the underhand condition showing the approximate location of the interaction surface relative to the user.

The at-a-distance condition is a separate experimental condition designed to simulate the varied distances and orientations that might be encountered in a realistic augmented reality environment (Figure 26). Four virtual boxes were placed in the environment; one under and in front of the user, a small one floating in front of the user, a larger one 22 feet in front and to the left of the user, and another one 36 feet in front of the user. The left cube was oriented 20 degrees around the horizontal axis while the front cube was oriented 25 degrees around the vertical axis. To guarantee that all trials started from the same position, the play and stop buttons remained in a fixed location 12 feet in front of the user. The task window was moved to one of 8 different positions for each trial. Each of the 5 practice trials were presented at the same positions (#s 1,3,4,7,8), while the 20 measured trials used one of 4 randomized position orderings (Figure 27). The position orderings were arranged in a Latin squares arrangement such that each position ordering was presented twice over the course of 24 subjects.

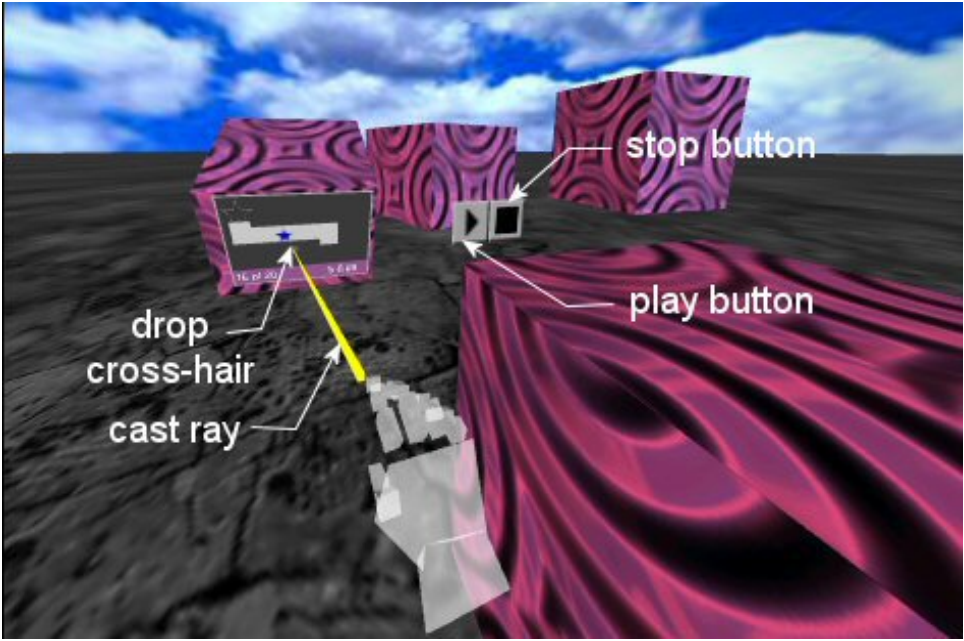


Figure 26. At-a-distance experimental condition drag-and-drop task shown with the ray-casting technique.

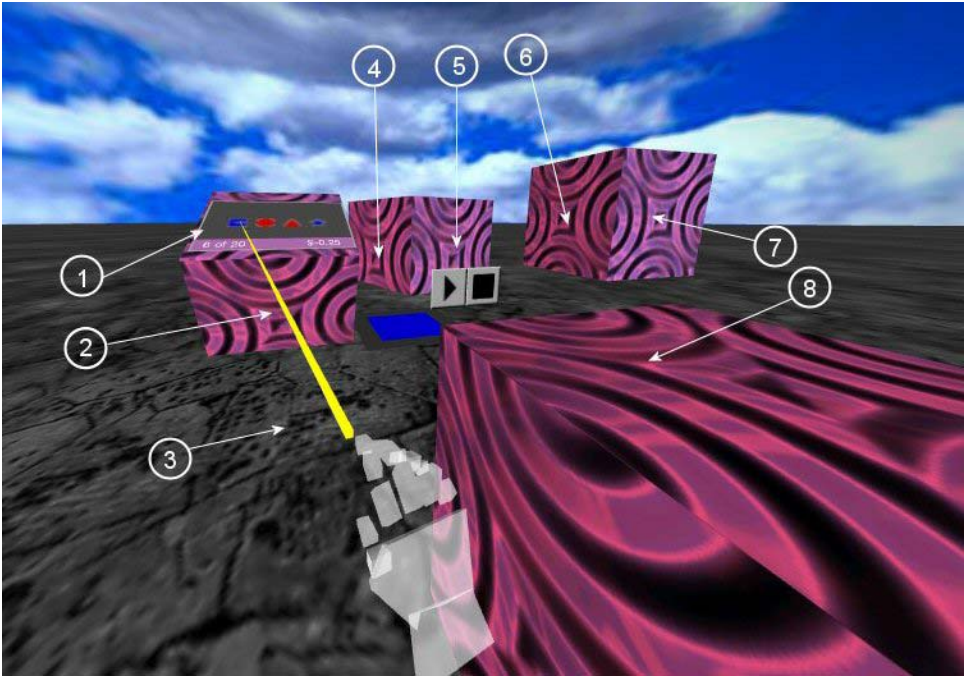


Figure 27. At-a-distance object selection trial positions shown with the ray-casting technique.

During the image-plane trials, subjects were asked to keep their arm straight in order to maximize the distance to the wand and the resulting accuracy of the technique. Early evaluation showed that users became very fatigued when trying to execute all 20 trials sequentially. In order to try and prevent fatigue from polluting the results, subjects were forced to wait 1 minute between each group of 5 image-plane trials. The implementation of the ray-casting technique used a yellow cylinder of fixed radius projecting from the virtual hand to a distance of 10 feet beyond the farthest surface. The ray exited the hand downward at a 45 degree angle in order to facilitate interaction with surfaces below the hand without discomfort. For this experimental condition, the subject was seated on the stand directly facing the screen approximately 5 feet away. The drop cross-hair was displayed at the intersection of the ray and the task window.

6.2 Study Results

Total task time, movement error and movement variability were calculated for each subject during the drag-and-drop task. Task movement error and variability were calculated in units of feet, and the width of the target path was fixed at 1 foot for all trials. Total task time and selection accuracy were recorded for each subject during the object selection task. To account for outliers, the standard deviation of each trial block was calculated for each user and those values not within 3 standard deviations were rejected.

6.2.1 Underhand Condition

The results of a one-way repeated measures analysis of variance (ANOVA) on each of the questionnaire results using technique presentation order as a between subjects factor gave the results shown in Table I. The F value is a measurement of the distance between population distributions and the p value represents the probability that the F measurement is incorrect. A p value of 0.05 is typically used to indicate a 95% confidence level in the results. For the drag-and-drop task, there was no significant effect on technique between image-plane and ray-casting. However, the mean values for arm fatigue ($M_{ip}=2.33$, $M_{rc}=2.08$) and ease of use ($M_{ip}=2.33$, $M_{rc}=2.00$) do indicate that ray-casting was somewhat better tolerated. The results for the object selection task indicate a significant increase in eye strain ($p=0.032$) and ease of use ($p=0.024$) for the image-plane technique. A Pearson's correlation of the object selection questionnaire results showed that eye strain and ease of use were strongly correlated ($r=0.803$) (Figure 28). Including eye-dominance in the analysis showed that it was a significant factor in drag-and-drop ease of use ($F_{1,22}=6.0$, $p=0.024$) with right eye dominant subjects giving more favorable scores for both techniques.

TABLE I

ONE-WAY ANOVA ON TECHNIQUE FOR UNDERHAND TASK QUESTIONNAIRES				
question	drag-and-drop		object selection	
	$F_{1,22}$	p	$F_{1,22}$	p
arm fatigue	4.21	0.052	0.049	0.83
eye strain	0.29	0.59	5.21	0.032
motion sickness	0.48	0.50	0.00	1.00
ease of use	3.29	0.083	5.90	0.024

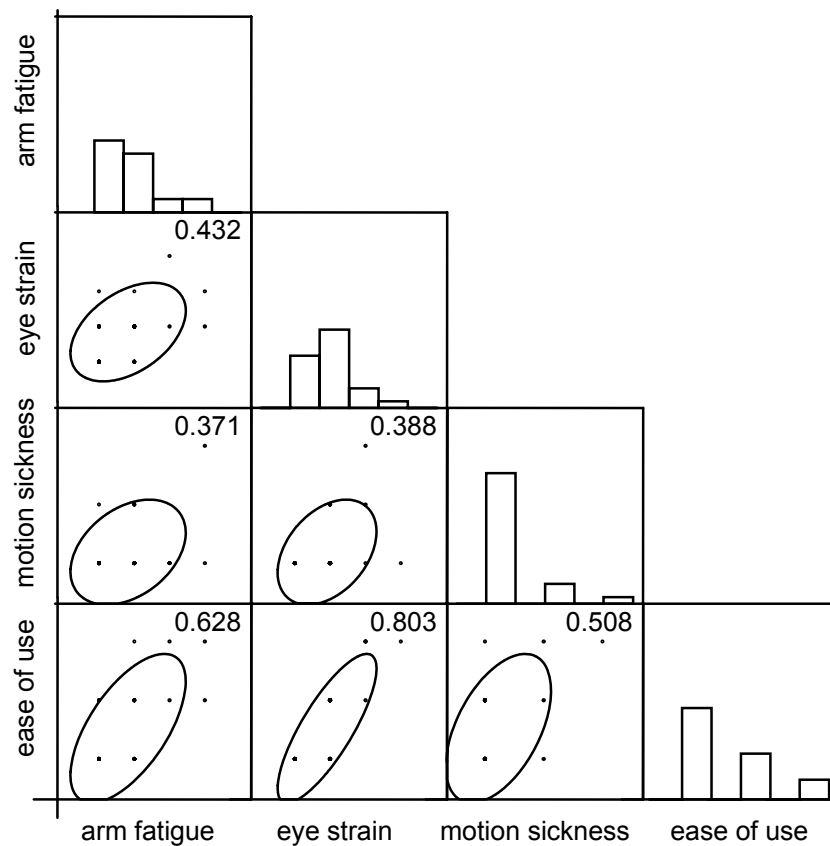


Figure 28. Pearson's correlation scatter plot matrix for the object selection task showing strong correlation between eye strain and ease of use.

The results of separate one-way ANOVA on time, error and variability using technique presentation order as a between subjects factor are shown in Table II. Trial times were consistently faster for ray-casting for both the drag-and-drop ($M_{ip}=8.43$, $M_{rc}=7.99$) and selection tasks ($M_{ip}=2.62$, $M_{rc}=2.40$). Four of the twenty trials were discarded due to an error in the calculation of the movement error and movement variability for those trials. For

the remaining 16 trials, the drag-and-drop task results indicate that both error ($M_{ip}=0.083$, $M_{rc}=0.076$) and variability ($M_{ip}=0.061$, $M_{rc}=0.056$) were significantly higher for the image-plane technique. Including eye-dominance in the analysis showed that it interacted significantly with technique ($F_{1,22}=6.0$, $p=0.024$) for object selection time. Right eye dominant subjects performed the image-plane selection task faster as a group.

TABLE II

ONE-WAY ANOVA ON TECHNIQUE FOR UNDERHAND TASKS

measure	$F_{1,22}$	p
drag-and-drop time	3.62	0.070
drag- and-drop error	7.84	0.010
drag- and-drop variability	4.78	0.040
object selection time	13.86	0.001

A post-hoc analysis segregated the trials into two groups; 6 trials requiring left-to-right path following and 9 trials requiring right-to-left path following with the remaining trials involving both directions. The results of separate two-way repeated measures ANOVA on time, error and variability indicated a strong interaction between technique and direction for both time ($F_{2,44}=6.14$, $p=0.021$) and error ($F_{2,44}=5.50$, $p=0.028$). Figure 29 shows that ray-casting trials moving from left-to-right generally had less error and variability. All error bars presented in this document indicate the standard error of the mean (SEM). Pair-wise one-way ANOVA on error showed that ray-casting on left-to-right trials was significantly more accurate than right-to-left trials ($F_{1,22}=15.75$, $p=0.001$) and image-plane trials in the same direction ($F_{1,22}=10.44$, $p=0.004$). No significant difference was found between image-plane and ray-casting error when only right-to-left trails were considered. A pair-wise one-way ANOVA showed similar results for trail variability.

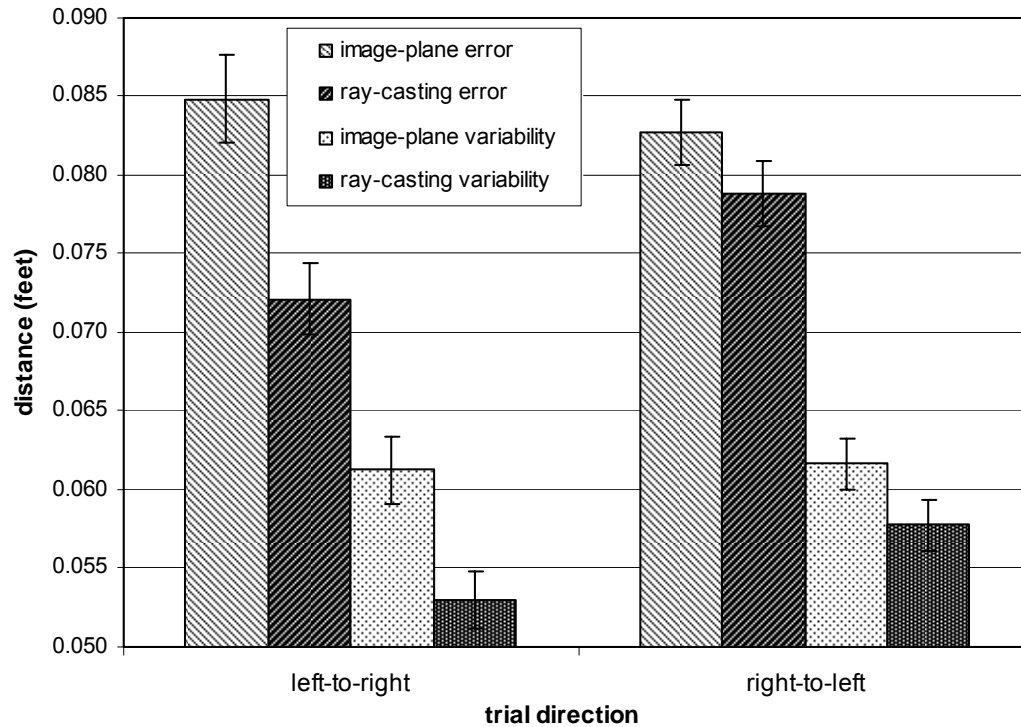


Figure 29. Underhand error and variability by direction suggests that ray-casting may be significantly more accurate on left-to-right trials.

6.2.2 At-A-Distance Condition

The results of a one-way ANOVA on each of the questionnaire results using technique presentation order as a between subjects factor gave the results shown in Table III. For both the drag-and-drop and object selection task there was a significant effect of technique on arm fatigue. For the drag-and-drop task, there was a significant main effect on both eye strain ($M_{ip}=2.33$, $M_{rc}=2.0$) and ease of use ($M_{ip}=3.46$, $M_{rc}=2.75$). A Pearson's correlation of the drag-and-drop questionnaire results showed a moderate correlation ($r=0.618$) between eye strain and ease of use (Figure 30). For the selection task, there was a significant main effect on ease of use ($M_{ip}=2.63$, $M_{rc}=2.04$) along with a significant presentation order effect ($F_{1,21}=5.21$, $p=0.033$). Subjects using image-plane first gave more favorable scores as a group for both techniques.

TABLE III

ONE-WAY ANOVA ON TECHNIQUE FOR AT-A-DISTANCE TASK QUESTIONNAIRES				
question	drag-and-drop		object selection	
	$F_{1,22}$	p	$F_{1,22}$	p
arm fatigue	18.76	0.00	36.62	0.00
eye strain	8.19	0.009	0.49	0.13
motion sickness	2.00	0.17	0.31	0.58
ease of use	12.87	0.002	8.29	0.009

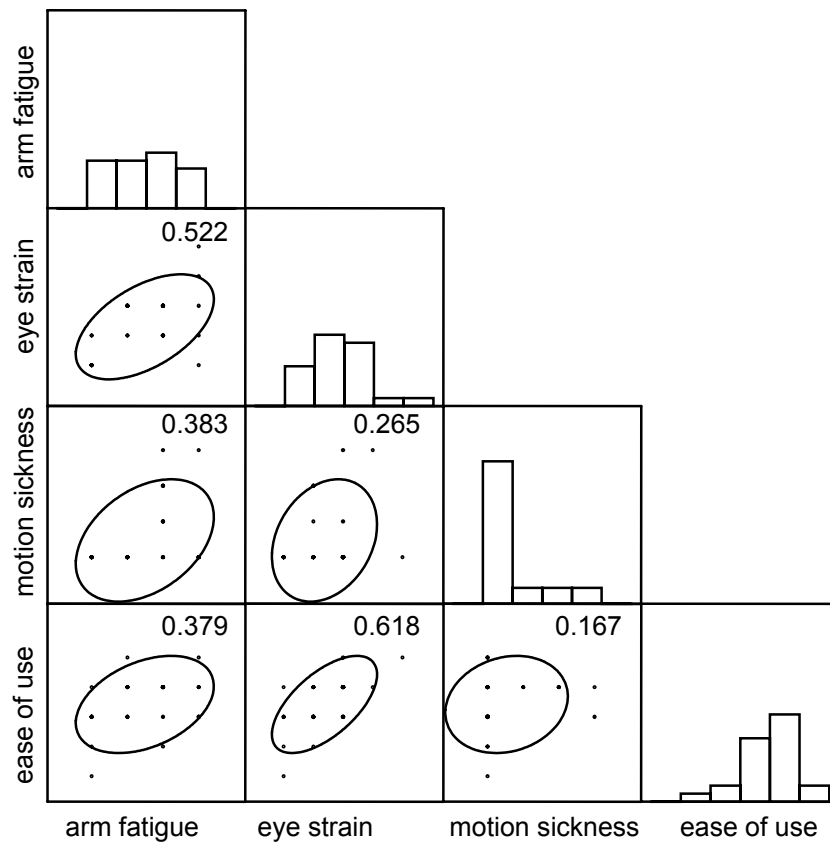


Figure 30. Pearson's correlation scatter plot matrix for the drag-and-drop task showing strong correlation between eye strain and ease of use.

The results of separate one-way repeated measures ANOVA on time, error and variability using technique presentation order as a between subjects factor are shown in Table IV. Four of the twenty trials were discarded due to an error in the calculation of the movement error and movement variability for those trials. No

significant effect due to technique or presentation order was found. A post-hoc analysis segregated the trials into three groups; 6 trials occurring to the left of the user (#s 1,2,3), 5 trials positioned in front of the user (#s 4,5), and 9 trials positioned close to and to the right of the user (#s 6,7,8). The results of separate two-way ANOVA on time, error and variability indicated a strong interaction between technique and position for drag-and-drop time ($F_{2,44}=29.10$, $p=0.0$), error ($F_{2,44}=27.34$, $p=0.0$), variability ($F_{2,44}=18.82$, $p=0.0$) and selection time ($F_{2,44}=21.96$, $p=0.0$).

TABLE IV

ONE-WAY ANOVA ON TECHNIQUE FOR AT-A-DISTANCE TASKS

measure	$F_{1,22}$	p
drag-and-drop time	0.71	0.41
drag- and-drop error	0.034	0.86
drag- and-drop variability	0.066	0.80
object selection time	0.45	0.51

Figure 31 shows that ray-casting trial times were faster than image-plane for trials to the left of the user ($M_{ip}=9.22$, $M_{rc}=8.28$) while they were slower than image-plane for trials in front of the user ($M_{ip}=8.50$, $M_{rc}=9.31$). A pair-wise one-way ANOVA on trial time showed that both trials to the left ($F_{1,22}=39.75$, $p=0.0$) and trials in front ($F_{1,22}=10.01$, $p=0.005$) were significantly different with regards to technique. Figure 32 shows that ray-casting trials in front of the user had higher error ($M_{ip}=0.11$, $M_{rc}=0.15$) and variability while they had lower error ($M_{ip}=0.14$, $M_{rc}=0.12$) and variability for trials to the right and near the user. A pair-wise one-way ANOVA on trial error showed that the error difference for both trials in front ($F_{1,22}=24.65$, $p=0.0$) and to the right ($F_{1,22}=9.79$, $p=0.005$) was significant while only trial variability in front of the user ($F_{1,22}=16.39$, $p=0.001$) was significantly different.

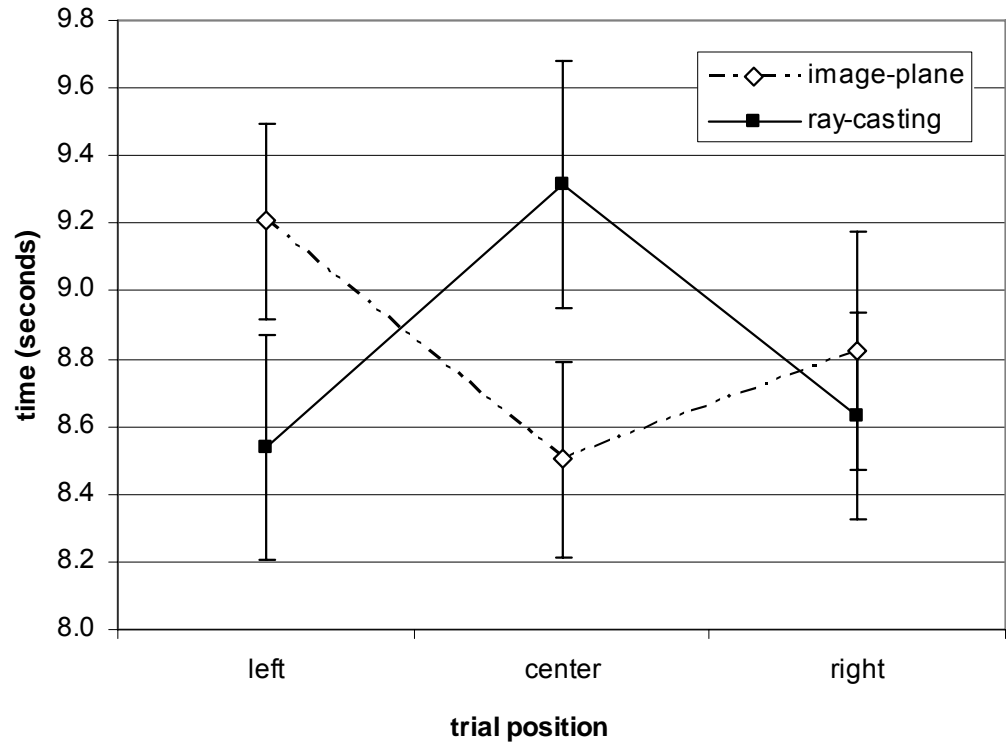


Figure 31. At-a-distance drag-and-drop trial time by position shows a significant difference between left and center trials.

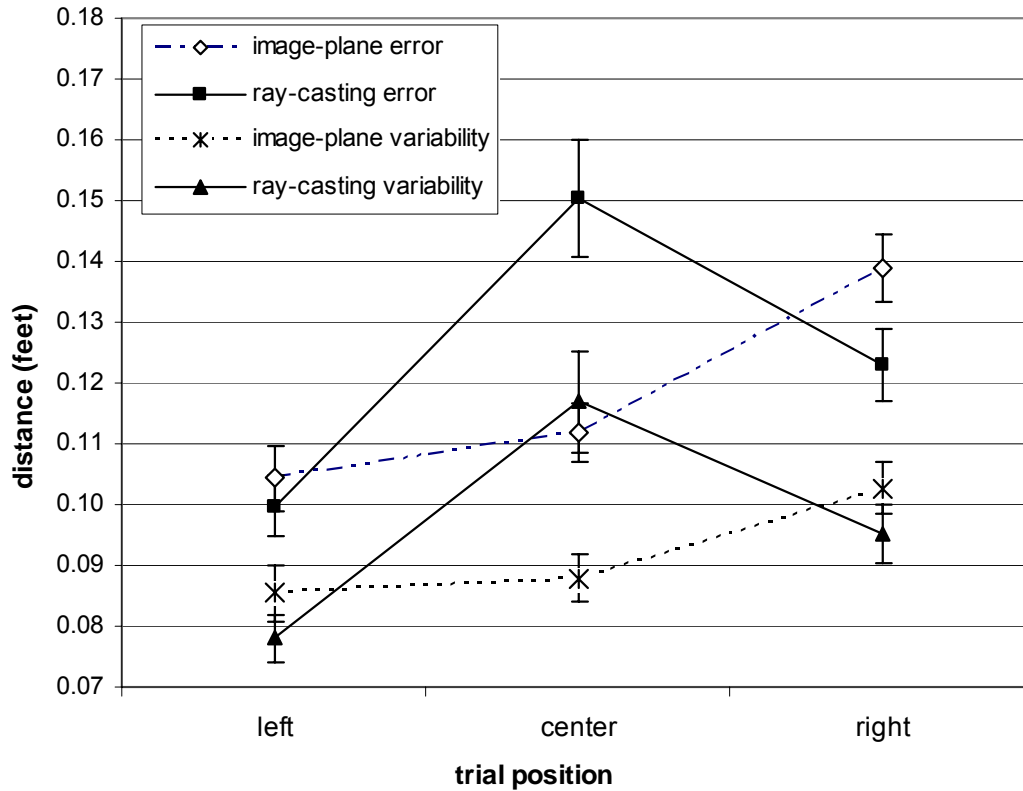


Figure 32. At-a-distance drag-and-drop trial error and variability by position shows a decrease in ray-casting accuracy on center trials.

For the object selection task, Figure 33 shows that ray-casting trial times were faster than image-plane for trials to the left of the user ($M_{ip}=3.74$, $M_{rc}=3.22$) while they were slower than image-plane for trials in front ($M_{ip}=3.30$, $M_{rc}=3.47$) and to the right of the user ($M_{ip}=3.56$, $M_{rc}=3.75$). A pair-wise one-way ANOVA on trial time showed that only trials to the left ($F_{1,22}=40.35$, $p=0.0$) were significantly different with regards to technique. However, a two-way ANOVA including only trials in front of and to the right of the user showed that image-plane was significantly faster than ray-casting ($F_{1,22}=4.39$, $p=0.048$).

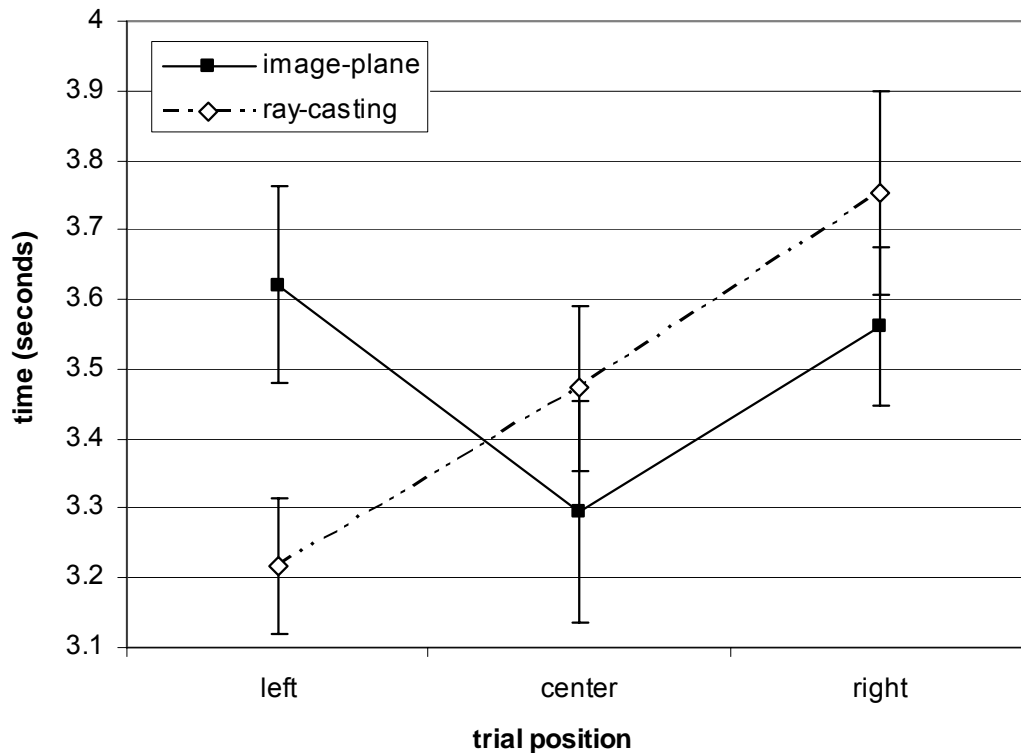


Figure 33. At-a-distance object selection trial time by position shows faster times for center and right trials.

6.3 Study Discussion

The study failed to find statistical evidence in support of five of the six sub-hypotheses. The study did find evidence in support of the hypotheses that image-plane arm fatigue is similar to ray-casting when used under hand. The results for using a dominant eye virtual cursor were mixed with significant eye strain reported during the underhand object selection and at-a-distance drag-and-drop tasks. This eye strain appears to have contributed to a significant difference in ease of user reporting from the subjects during the underhand condition. The significant difference in underhand accuracy between methods may be related to physiological differences between the two tasks. While there was no overall difference in time and error results for the at-a-distance condition, the results suggest that image-plane selections were moderately faster than ray-casting when not reaching across the body. The accuracy results at a distance also suggest that image-plane selection was more accurate than ray-casting at greater distances.

6.3.1 Sub-Hypothesis 1

The questionnaire results for both tasks of the underhand condition support the first sub-hypothesis that arm fatigue is similar when using either ray-casting or image-plane selection. The statistics do show that the study subjects came close to reporting a significant difference between the two techniques. However, these results are encouraging because the size of the interaction surface and its distance from the user gave subjects the opportunity to significantly reduce the required arm motion when using ray-casting. When using the image-plane technique, subjects had to move the wand a full foot to reach the left or right extreme of the interaction surface. A similar action could be accomplished with only a 35 degree wrist rotation when using the ray-casting technique. The majority of subjects took advantage of this difference in technique and used wrist rotations almost exclusively during the ray-casting object selection task. The effort reduction during the drag-and-drop task was not quite as significant because most subjects used an even mix of wrist and arm motion during path following. In addition, the physical setup also contributed to an increase in effort for subjects using the image-plane technique. The single wall system restricted the ability to position the interaction window directly below the hand. The resolution of the system also placed limitations on how small the interaction window could be rendered. Given a high resolution head mounted display or a CAVE system with floor projection, the interaction window could potentially be reduced in size or positioned closer to a natural working position for the subject. It is reasonable to expect that improvements in the physical setup would only reduce the difference in fatigue reporting between the two techniques.

6.3.2 Sub-Hypothesis 2

The second sub-hypothesis, that the use of a virtual cursor in the dominant eye would be well tolerated by subjects, was not proven to be true under all conditions. The results suggest that eye strain was caused when subjects had to follow the virtual cursor at a distance or transition quickly between distant and near viewpoints. The eye strain caused by the underhand object selection task was not reported during the drag-and-drop task or during the object selection task at a distance. This suggests that looking away from the virtual hand to view the target object in front of the user contributed to the eye strain. This eye strain was likely related to a difficulty in re-acquiring the position of the virtual cursor. The portion of the ray-casting ray emanating from the hand position likely serves as a visual clue to orient the user towards the intersection point. Several strategies could be employed to keep the user aware of the virtual cursor position. Halos or lines emanating from the virtual cursor may help orient the user. These visual cues could even be modulated by the relationship between viewing

orientation and the hand. The visual cues can increase when the user is looking away from the virtual cursor and then recede as the user orients their viewpoint with the hand position.

The significant eye strain observed during the at-a-distance drag-and-drop task cannot be attributed to an inability to track the virtual cursor. The low eye strain scores and the fast object selection times suggest that tracking the virtual cursor and making selections was not related to the eye strain. The lower resolution and smaller size of the interaction window cannot be attributed to the eye strain because the ray-casting task would have also shown similar results. One possible source of eye strain is the size of the virtual cursor. The ray-casting ray naturally appears smaller as the intersection point moves away from the user, while the image-plane cursor does not change size when it is placed over distant targets. Although it may not affect gross selection, the increased size of the cursor may interfere with attempts to follow the path accurately. An obvious solution to this problem is to decrease the size of the virtual cursor to reflect the depth of the object being selected. This size change could compliment traditional selection referent color and shape changes in providing cues to help the user appreciate what they are selecting.

Another possible cause of eye strain during distant drag-and-drop tasks could be related to the very nature of image-plane selection technique. The drag-and-drop task using the image-plane technique at a distance is potentially visually confusing. The subjects were faced with a semi-transparent virtual hand, their own physical hand, a physical screen and a distant rendered object all provoking different disparity relationships within the brain. In the physical world, it is a more natural condition to have already moved the hand to an object that is being manipulated. This resolves the disparity problems between hand and selected object during the task. If these disparity issues are contributing to eye strain, then it is possible that with time users can adjust to the disparity between their physical hand and the object being manipulated. If exposure does not resolve the problem, there may also be strategies to ameliorate the affects of a disparity mismatch. In an augmented reality environment, obviously there is no need for a virtual hand. And, in an HMD environment there is no need to render the physical hand. For the remaining disparity cues of the virtual or physical hand, it may be possible to increase the transparency of the hand in the non-dominant eye during manipulation tasks. Another possible strategy involves using the scaled world grab technique to bring a selected object to the depth of the hand. The main drawback of scaled world techniques involves the manipulation of object depth and the alteration of perspective. Distant objects are difficult to bring closer to the user, and objects depths are limited by the scaling

relationship between world and the user. However, image-plane tasks that involve 2D widgets are constrained in both depth and location and are therefore not subject to the major drawbacks of the scaled world grab technique. While not appropriate for general selection and manipulation tasks, the scaled world technique could aid 2D widget use at a distance by effectively bringing only the widget itself to the position of the hand.

6.3.3 Sub-Hypothesis 3

The questionnaire results for ease of use under hand only support sub-hypothesis three for one of the two tasks. Subjects reported similar ease of use for the underhand drag-and-drop task, but gave significantly poorer ease of use scores to image-plane during the object selection task. The ease of use question is capturing the aggregate of all subjective factors, two of which are the physical motion required during the task and the ability to see the cursor accurately. There was a moderate correlation between arm fatigue and ease of use ($r=0.628$), but the ease of use scores for the image-plane object selection task were strongly correlated with the eye strain scores ($r=0.803$). The increase in eye strain appears related to the unique nature of the object selection task performed under hand. Users had to repeatedly look out in front at the target shape and then back again at the interaction window. The questionnaire results for the object selection task in the at-a-distance condition suggest that eye strain was not a problem when the target and the interaction windows were both out in front of the subject. Although correlation does not imply causation, the eye strain results suggest that it was the unique nature of the underhand object selection task that resulted in the low ease of use for the image-plane technique. Therefore, it is reasonable to assume that a selection task that was designed with the target shape within the interaction window would have produced comparable ease of use scores in the underhand condition.

6.3.4 Sub-Hypothesis 4

The fourth sub-hypothesis was that subjects would show similar task efficiency when using both techniques. The time and error results for the underhand condition show a difference in the ability of subjects to use image-plane as efficiently as ray-casting. The faster object selection times for ray-casting are not surprising given that ray-casting subjects were able make selections by merely rotating their wrist when using ray-casting. Assuming that Fitt's law applies to both techniques, physiological differences in the two techniques guarantee that the slope constant describing image-plane selection will be higher. The task times during the underhand drag-and-drop task were faster for ray-casting than image-plane but not statistically significant. Again, Fitt's law helps to explain the inability of subjects to complete the path following task as quickly. It is reasonable to assume that

an improved physical setup, with a smaller interaction window would have allowed image-plane users to produce faster task completion times. However, it is also safe to say that image-plane selection will never be a faster technique than ray-casting on surfaces below the hand. Ray-casting is an efficient selection technique on 2D surfaces oriented towards the user. And, the distance and size of the content are not likely to create accuracy problems for ray-casting.

The movement error and movement variability measures for the underhand condition do not support the fourth sub-hypothesis. A post-hoc analysis was undertaken in an attempt to segregate the trials by direction of movement. Unfortunately, the study was not designed to test the hypothesis that path following in a specific direction would produce different efficiencies. The most consistent grouping of trials formed two sub-groups; one with trials moving predominantly from the bottom left of the interaction window to the top right and another with trials requiring movement in the opposite direction. This grouping found that ray-casting was significantly more accurate in trials from left to right. This grouping also found that ray-casting was significantly faster than image-plane in trials from right to left.

There are a number of physiological factors that might contribute to these differences in task execution. Prior research has shown that mouse operations in the horizontal direction are slower than those in the vertical direction (99). And, this same research found that up mouse movements were significantly slower than down movements. This physiological difference has influenced the design of menu systems as menus typically drop down and require the greatest accuracy when moving horizontally. It is reasonable to assume that path following on windows under the hand is similar to mouse movement on the desktop. On the other hand, it is generally accepted that up and down movement, known as flexion and extension, of the wrist is more efficient than movement in the left to right range (100). It is therefore possible that the significant increase in ray-casting accuracy during left/down to right/up trials was a reflection of purely motor physical differences between the image-plane and ray-casting techniques. Another possible explanation for performance differences during left to right movements is that right handed users found that their hand or virtual hand occluded the intended path. Subjects made no mention of this phenomenon, but it still needs to be a consideration when designing for image-plane selection on cascading menus.

6.3.5 Sub-Hypothesis 5

The fifth sub-hypothesis states that image-plane selection is faster than ray-casting in environments where selections are made over varied depths from the user. The initial results of the study do not support this hypothesis, but analysis of the results does suggest that it is true when the user is not reaching across their body. A post-hoc analysis grouped the eight positions of the interaction window into three groups; those to the left of the subject, those directly in front of the subject and those to the right of the subject. The selection times on trials using image-plane selection to the left of the right handed subjects were significantly slower than those using ray-casting. This result is not surprising and would likely be predicted by Fitt's law. An ANOVA analysis excluding trials to the left of the subject did show a significant increase in the speed with which subjects performed the object selection task. Unfortunately, the claim that image-plane is faster than ray-casting cannot be made here because subjects did not complete the same task when only the front and right position trials are considered. Both the trial ordering and the trial position were subject to a Latin Squares arrangement. The study was not designed to allow the trials to be segregated by position as subjects were not presented the same object selection trials at each location.

6.3.6 Sub-Hypothesis 6

The sixth sub-hypothesis states that image-plane is more accurate than ray-casting on surfaces that are oblique to the user. The results of drag-and-drop trials during the at-a-distance condition do not show an increase in accuracy for image-plane selection. The results of the post-hoc analysis for drag-and-drop trials to the right of the user reflect the findings of the underhand condition as users were more accurate with ray-casting. Overall, given the varied orientations of the surfaces to the right of the subject, one would have expected image-plane selection to perform better. It appears that subjects using the ray-casting technique were able to compensate for the oblique angles on these close surfaces by using a strategy that relies heavily on translational movements. Subjects tended to move their hand into a position between within their viewpoint and rely more on translational movements than rotational movements of the ray. This strategy obviously trades off fatigue for increased accuracy. The same analysis of left, center and right trial groupings did show that image-plane was more accurate than ray-casting on trials located directly in front of the user. The two center interaction window positions represent the greatest distance from the user. One would expect the accuracy limitations of ray-casting to appear at such a distance. However, again, this type of analysis is not statistically valid because subjects did not perform the same trials at the center locations with each technique.

The center trials were located the farthest distance from the user and presented a small target. It is worth considering the possibility that the accuracy differences were related to the accuracy allowed by each technique. The officially published tracker jitter for the Ascension tracking system is 0.5mm translation and 0.1 degree angular rotation¹⁰. Assuming that the subject holds the wand 2 feet from their head, an image-plane implementation should produce an effective angular jitter of 0.9 degrees. Actual measurements of the tracking system revealed a standard deviation of approximately 0.3mm for position and 0.02 degrees for orientation. These results are consistent with other published reports of measured Ascension tracker jitter (101). Again, assuming that the wand is held approximately 2 feet from the head and that both head and wand can potentially contribute to image-plane jitter, the effective image-plane jitter would be approximately 0.056 degrees. This suggests that tracker jitter should benefit the ray-casting technique and does not explain the difference in accuracy during drag-and-drop trials in front of the user.

6.4 Conclusions and Future Work

The study established that there was no difference in arm fatigue based on technique when the interaction window was used below hand. While the statistics did not support the hypothesis, it is reasonable to assume that image-plane and ray-casting can both be used with equal ease of use under hand as long as frequent reacquisition of the cursor is not required. The study failed to show that placing the cursor exclusively in the dominant eye was well tolerated under all conditions. Eye strain appeared to be a problem with the dominant eye technique when the subjects were doing path following at a distance or frequently reacquiring the virtual cursor. More investigation is warranted to determine if eye strain can be avoided with the technique. One way to study this is to make small changes to the study and repeat it. The target shape during underhand object selection can be moved onto the surface in order to alleviate the need to look away from the interaction window. A reinforcing cursor at the interaction surface can be used instead of the drop cursor that was used in this study. The non-dominant eye rendering of the virtual hand can also be hidden during manipulation at-a-distance to investigate its effect on eye strain. And, the size of the virtual cursor can be adjusted relative to the distance of the selection referent.

¹⁰ <http://www.ascension-tech.com/products/flockofbirds.php>

The study also failed to show that subjects could do drag-and-drop and selection tasks with equal efficiency on interaction windows held under hand. This result raises many questions about what factors contribute to the difference in efficiency between the two techniques. The design of the study was not prepared to answer these questions. A future study is warranted to investigate the effect of path following direction and virtual hand transparency on image-plane efficiency. Such a study would benefit from a more simplified path following task that can easily be characterized by direction. These results suggest that the ability to transition smoothly into ray-casting may prove useful in implementations based on image-plane selection.

While the study did not establish an increase in efficiency due to the image-plane technique on surfaces at a distance, it did establish that image-plane was at least as fast and accurate as ray-casting and therefore a viable alternative to it. However, these results are not likely to encourage broader use of image-plane selection because subjects reported higher ease of use and lower arm fatigue ray-casting scores for the at-a-distance condition. Given that users do not find it easier to use, a clear performance advantage over ray-casting is needed to justify using the technique. The study conducted here did not adequately isolate the independent variables that would allow a more thorough investigation of image-plane efficiency.

An obvious change to the present study would be the matching of trial identifiers by position instead of the double Latin Squares ordering that was used here. This would have allowed the analysis to exclude those trials that required reaching across the body without compromising the validity of the analytical results. The study presented here was not adequately designed to explore the relationship that surface distance and orientation have on path following efficiency. A more appropriate study would isolate depth and orientation independently in a more systematic manner. Another potential problem was the effect of fatigue on the results. In order to control for fatigue, this study forced users to wait 1 minute between each group of 5 trials. An alternative methodology is to eliminate fatigue by doing the evaluation of orientation in the underhand condition. Both the angle of the interaction window and the distance of the window from the user can be adjusted independently in the underhand position. Another approach to the fatigue problem acknowledges the tendency of ray-casting users to use translational movements when possible at the cost of higher fatigue. A study that compares ray-casting to a virtual mouse implementation may help keep fatigue for both techniques at similar levels for tasks at-a-distance.

7. A PROOF OF CONCEPT APPLICATION

The results of the user study presented in the previous chapter suggest that some of the most critical assumptions of the Withindows framework hold up empirically. However, the evaluation of individual parts of the framework cannot substitute for the real world evaluation of applications based on the framework. The development of working applications offers the opportunity to address implementation and usability issues that only come to light when actual users are involved. Yet, the choice of domain for such applications is as important as the development effort itself. Any immersive application that is going to lead to productive work must satisfy an established need for that effort. At the same time, the domain in question should be one that will benefit from the interactivity that the Withindows framework offers. The design and evaluation of virtual worlds is a domain that satisfies these requirements. This chapter describes the development of a graphical interface for the Ygdrasil virtual reality development system. The first two sections of the chapter describe the motivations and design for the application. The last two sections give some details about the development of the application and discuss the results of that development effort.

7.1 Motivations

The motivations for building a working application based on the Withindows framework are threefold. The first reason is an opportunity to demonstrate how the pieces of the framework fit together to create immersive applications. A working application offers the opportunity to evaluate the practicality of framework details that have heretofore been derived out of theoretical utility. The second reason for such a development effort is to make a contribution to the field of 3D user interfaces by creating an application that satisfies the high goals set forth here. Despite three decades of research in the field, very few applications have been developed that are truly an improvement in the productivity and usability of existing tools. The final reason, and perhaps the most important reason for creating any software application, is the desire to support an existing community of users by responding to an established need for improved software tools.

7.1.1 Evaluate the Framework in a Realistic Environment

Whether it is for a physical device, an organizational structure or a design guideline such as Withindows, a proof of concept application in a real environment is an invaluable tool for any theoretical design. The pieces of

the Withindows framework are motivated by theoretical considerations that suggest they will achieve their goals. However, one cannot fully appreciate whether the sum of the parts will yield the intended results without putting them together in an actual application. The usability of doing 2½D selection and manipulation of objects within alternate viewing windows needs to be evaluated. The efficiency gains, if there are any, of the 6 DOF input methods proposed need to be established. The proposed rules for window management need to be tested under real world conditions. And, the appropriate relationship between user viewpoint and the content visible within a viewing window needs to be established. With each of these topics unpredictable implementation issues are likely to be uncovered.

One important subject that has not been fully discussed is the relationship between viewer and the content within those windows. In their original work, Stoev and Schmasteig described a taxonomy that included three possible states for the viewing window and three possible states for the content within those windows. The states of the viewing window are effectively the same conditions proposed for the application windows in the Withindows framework; surround-fixed, view-fixed and world-fixed. Surround-fixed windows are those floating windows whose position and orientation with respect to the user are under user control. The following discussion will focus on surround-fixed windows because the implications are easily translated to the other conditions. The three possible states of content within viewing windows are secondary world fixed to the surrounding world, secondary world fixed to the viewing window and secondary world fixed to the user viewpoint. The first case has been described before as unlocking the window from the secondary world in order to frame objects within its view. In the second case, where the secondary world is fixed to the viewing window, moving the user viewpoint allows the user to see different parts of the secondary world as if looking through a window (Figure 34). In the final case, the secondary world is fixed to the user viewpoint and, as a result, the image within the window remains the same as the user moves their viewpoint around the content. However, in contrast to the second case, changing the orientation and position of the viewing window changes the viewpoint into the secondary world.

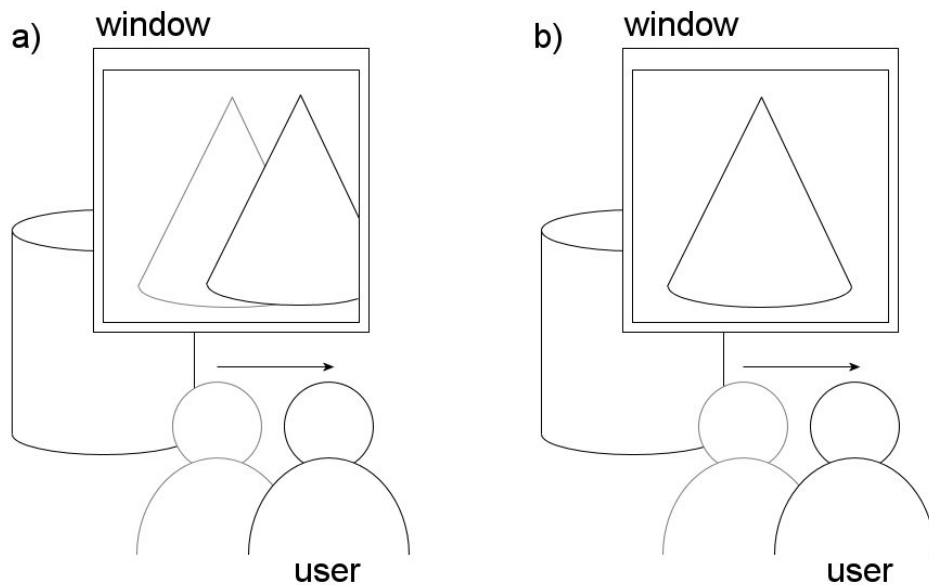


Figure 34. The relationship between user viewpoint and window content is similar to either a) looking through a picture window or b) using a camera LCD viewer.

The combinations of viewing window coordinate system and the relationship between the secondary world and that window create many possible configurations that must be investigated for their practicality and usability. On the whole, the states of the secondary world correspond to two real world situations. The first case is analogous to looking through a lens into another world. The second case is analogous to looking into a world through a picture window. And, the final case is analogous to looking at the LCD viewing window of a digital camera. It is difficult to determine if users will find utility in the ability to see different parts of the secondary world by moving their head position. On one hand, it would appear that having such ability would enhance their understanding of the content within the window. Yet, it is also likely that small changes in user viewpoint could cause objects of interest to move out of sight, thereby frustrating the user. Also, consider the example given in the last chapter where a user might merely point the viewing window at content within the scene in order to capture a viewpoint. If we allow orientation and position changes to the viewing window to determine the content within it then what should the user expect to see through the window when shooting from the hip? If we apply case one above then the content the user is interested in may only be visible from an awkward angle with respect to the viewing window.

The utility of a picture window or viewing window approach is further complicated by the subject of collaborative work environments. If we allow user viewpoint to determine what is seen within a window, then two users will not see the same content within a window. Under this condition, how do we maintain situational awareness between users when other users cannot see the objects that a user is interacting with? While these comments may suggest that abandoning the picture window in favor of using a viewpoint independent model is desirable, there may be ways to provide both options to the user. In a collaborative work environment, the view presented within the window could be fixed to the user that is using it, while other users can be given a viewpoint into that window that is viewpoint independent. This suggests that the original taxonomy may need to be expanded to include the potential viewpoints of secondary viewers. Most importantly, this discussion highlights the difficulty of fully appreciating what design choices are actually appropriate for a generalized framework. A proof of concept application is a practical environment within which to evaluate the practicality and usability of these different scenarios for actual users.

7.1.2 Create a Productive Immersive Application

The Withindows framework is not only a guide for the development of immersive applications, but it also serves as a guide to determining which domains will benefit from immersive applications. The framework stresses the relationship between first person egocentric appreciation of 3D space and the need to interact with that space in some fashion. There are a number of application areas outside of the simulation of reality that are already benefiting from egocentric immersion. These success stories include protein analysis, geophysical analysis, engineering heat and fluid analysis, engineering design review, architectural and virtual environment design analysis. Any one of these tasks is likely to benefit from an increased ability to interact with the underlying data while still experiencing the content from within. Yet, the most obvious choice for the types of interactive tasks that are considered primary in virtual environments involves the arrangement of physical space. For this and several other reasons, an application for building and evaluating virtual worlds is a perfect fit for the Withindows framework.

The task of building virtual worlds typically involves designing a 3D space and assigning behaviors to objects within that space. This process involves not only the programming of complex interactivity, but also the placement and arrangement of interaction volumes and surfaces. As in game design, the designer of a virtual world must ensure that interaction locations are sized and positioned appropriately to be activated by actual

users. Designers of virtual worlds frequently need to understand the relationship between viewpoint and the physical limitations of the human body. Simply moving through a virtual world in a simulation mode can easily lead to inappropriate assumptions about what users can reach and how their body motion will behave. For these reasons, virtual world design is somewhat like architectural design except that the traditional workflow of both disciplines is completely different.

Because they did not have the resources to actually test their environments, architects adopted a variety of ergonomic and design guidelines to help them create spaces that humans can work in effectively. And, recently Architects have used a workflow that only utilizes immersive applications for late design reviews. In contrast, being technologically oriented, game and virtual world designers frequently test their designs directly in within the target environment in a tighter loop. However, unlike game developers, virtual world builders must go through more effort to fully evaluate their work in its target environment. This frequent need to place the finished product directly within the eventual usage scenario is analogous to the WYSIWIG interfaces that have been developed for HTML editing. For this reason, the ability to not only evaluate but also make design changes to virtual worlds is of obvious benefit to developers. Immersive design changes require the ability to manipulate objects accurately from a distance while maintaining the current user perspective. The evaluation process in virtual worlds is also likely to require that users travel frequently between different parts of the environment. These two primary immersive tasks, along with the need to access the same object behavior and property attributes from within the immersive environment, make virtual world building an ideal application domain for the Withindows framework.

Several systems for building immersive virtual worlds have been developed. Some aid the development process on the desktop and allow the user to interact with their design immediately in simulation. Recently, a number of desktop virtual reality applications that integrate the construction of world content into the primary user interface have appeared¹¹. Others, mostly in research environments, have attempted to create immersive virtual world development environments (23,102). However, none attempt to offer the same level of productivity in both contexts. A number of collaborative workflow scenarios have been described in a previous chapter. A workflow that includes one designer working on the desktop and another person evaluating the world is of obvious use for virtual world designers. This author is only aware of a single system that has attempted to create a simultaneous desktop development and immersive evaluation tool (91). This system used a highly functional interface on the

¹¹ <http://secondlife.com/whatis/building.php>

desktop for designing the world and allowed the collaborative immersive user to make minor changes via a virtual hand interaction with objects. The situational awareness in this collaborative system was addressed to some extent by representing desktop mouse clicks as interactions with an oversized hand in the immersive setting. While this method may give the immersive user an indication of user intent during object manipulations, this leaves the user with no awareness of high level functionality such as behavior editing. Stenius developed a collaborative 3D modeling system based on the DIVE system (103). This system was designed for desktop use and never extended to an immersive context. A combination desktop and immersive world builder based on the Withindows framework should allow for a comprehensive desktop interface that can be also be accessed asynchronously or collaboratively within an immersive setting. A single application that allows both productive desktop development and effective immersive access to that functionality would be a significant contribution to the field.

7.1.3 Supporting a Community of Users

Many prior efforts at developing immersive applications have been primarily focused on building a proof of concept to experiment with the usability of such systems. Many of the most ambitious systems have been directed at 3D object modeling (44,104). The majority of these applications have been attempts to exploit the unique nature of 6 DOF or higher input devices. Unfortunately, for reasons that have been covered here extensively, the specification of input in free space without haptic feedback limits the utility of these interaction methods at this time. Yet, even if these technical limitations can be overcome; it is hard to imagine that the 3D modeling community would either trade the comfort of the desktop for more physical interaction or be able to justify the improvement in design that true immersion can bring. The same cannot be said for the designers of virtual environments. Virtual environment designers are more likely to have access to virtual reality equipment and are likely to be motivated to use it to evaluate their content in the target environment. There is one user community of virtual world builders associated with the Electronic Visualization Laboratory (EVL) that is likely to benefit from such a development effort. This community of electronic artists is not only in need of a graphical interface but is also in need of the kind of support that improved tools can bring.

7.1.3.1 The Ygdrasil Development System

The Ygdrasil virtual reality development environment is a script-based tool for rapidly creating tele-collaborative virtual reality worlds. Developing virtual reality applications has been a notoriously difficult process.

The technical requirements of creating such worlds can make individual artistic expression prohibitive. The Ygdrasil development environment is a script based system that allows non-technical users to create virtual reality worlds with a minimum of technical expertise. Users can create virtual worlds with typical functionality by combining any number of predefined object components. When additional functionality not provided by the basic system is needed, a dynamic shared library system allows additional components to be added to the system without recompilation of the core system. The worlds created by Ygdrasil are unique because they seamlessly provide functionality that allows them to be networked together to create a single tele-collaborative virtual reality between multiple sites.

The Ygdrasil development environment grew out of the realization that developers of virtual reality applications frequently require a similar set of functions. Developers frequently reused the same template that provided basic functionality of such as the creation of the main application loop and code for navigating within the environment. Beyond this basic functionality, the development of a virtual world generally consisted of creating instances of SGI OpenGL Performer objects and arranging them in a tree like structure called a scene graph. Motivated by the needs of artists in the Electronic Visualization Laboratory, Dave Pape developed a system called XP that allowed users to define the arrangement of typical Performer objects such as switches, transforms and geometry objects via a simple text file. The initial system was sub-classed from the base OpenGL Performer classes, but eventually Pape created a separate set of classes that mirrored the Performer scene graph.

The original XP system was renamed to Ygdrasil when it was completely rewritten to include two important features (105). The original system allowed users to develop their own XP objects. Yet, the whole underlying system had to be recompiled after each change to a user defined object. The system was rebuilt around a dynamic shared library system that loads user objects when they are requested in a scene file. This aspect of the system was modeled after the Bamboo VR development system (106). The second significant change involved the addition of transparent networking to each node. This addition allows a user to network two virtual worlds together without any additional programming. This seamless integration of networking was modeled after the Avacado VR development system (107). Objects created at one site are responsible for sending any updates of their state to other client sites via a central network server. Ygdrasil allows any number of sites to contribute content to the distributed scene graph that comprises the shared virtual world.

There are a number of similarities between Ygdrasil and the Virtual Reality Modeling Language (VRML). Both systems allow users to create a scene graph in script, are interpreted at runtime and allow users to create their own user modules. It is worth noting that VRML fell out of favor at about the same time that systems like Ygdrasil were being developed. This may have something to do with the difference in how they approached networked virtual worlds. The standard developed by the VRML working group, called Living Worlds, handed off the management of networking to 3rd parties (108). This forced content developers to add networking for individual state variables explicitly within their scene files.

The early Ygdrasil system was a significant success within the small community that used it. Prior to its development, artists interested in developing highly interactive virtual reality worlds were forced to either solicit the help of experts in the field or develop strong technical skills themselves. In 2001, the Electronic Visualization Laboratory was able to put together a single very large tele-collaborative virtual world with pieces contributed by 11 different electronic artists. This single virtual world was then presented in the CAVE system at the Ars Electronica Museum in Linz and connected to as many as 7 different locations around the world at one time (109). This single demonstration showed both that tele-collaborative virtual reality could be accomplished over ordinary internet connections and that the Ygdrasil system allows artists to develop virtual reality applications without the technical assistance required in the past.

7.1.3.2 User Community Needs

Despite the Ars Electronica show and other successes, the original Ygdrasil system still had shortcomings. Although basic environments could be created completely from scene files, the coarse granularity of the basic components did not allow users to create significant new functionality out of their parts. The Ygdrasil system allows behaviors to be programmed by setting sending messages between objects, also known as nodes for their location within the scene graph. For example, a user trigger node might be configured to send a message to a light node when a user enters a specific area in the virtual world (i.e. myLight.on). Messages sent between nodes often need access to the internal state variables of a node. For example, the internal state variables of a light node consist of the light state, on or off, the light direction and the ambient, diffuse and specular colors associated with it. In contrast to VRML which allows the developer to access every state variable associated with nodes, the original Ygdrasil system allowed most state variables to be changed but only made a limited set of state variables available when constructing a message. Moreover, the scripting language provides built in means

of performing even the most basic computations on those variables when constructing messages. Taken together, these limitations in the original system forced all but the most basic users to create custom nodes that must be written in C++. Ygdrasil users need a system that allows them to access all internal node state variables, create intermediate results from calculations on those variables and generate messages from those results.

Unlike VRML and some other virtual reality development systems, Ygdrasil users have no graphical interface to aid the construction of their scenes. A graphical interface can aid the placement of models and interaction areas. Without such an interface, the location and size of content must be entered manually into the text scene file and then evaluated using a desktop simulator. This trial-and-error process of changing the scene file, saving out the text file and running a simulation is time consuming and inefficient. Furthermore, a graphical editor can be a valuable tool in providing scaffolding to both novice and advanced users. Scene graph creation requires identifying the correct name of the desired node and knowing the correct syntax of the messages that initialize its internal state. A graphical interface can allow the developer to choose from a list of available nodes and the messages they accept. Both VRML and Ygdrasil use a scripting language that is prone to syntax errors when created by hand. Scene file syntax errors can either cause the simulation to crash immediately or, worse yet, produce results that do not become apparent until the whole scene has been explored. Ygdrasil users need a graphical interface that allows them to focus their efforts on creating virtual worlds instead of chasing down node documentation and syntax errors.

As a computer science student at EVL and as a teacher of Ygdrasil for over four years, the author has seen the above problems firsthand. The Electronic Visualization Laboratory has been a unique environment where electronic artists and computer scientists work together to further the goals of one another. While electronic artists are often versed in the creation of 3D modeling content, they often find themselves in need of high-end dynamic rendering techniques. This forces the artist to think in ways that have been fundamental to the success of computer scientists. Yet, the need to either learn a procedural and logical methodology or to rely on the benevolence of computer scientists is often at odds with the fundamental creative process. Virtual reality artists need tool that abstract the implementation of high end rendering techniques such as clipping planes, video texturing, stencil buffers, particle systems and rendering to textures. A development system that supports their needs will allow them to utilize these techniques directly within their scenes without significant technical expertise.

7.2 System Design

The Ygdrasil graphical interface application was motivated by both a concern for the existing community of Ygdrasil users and an attempt to adhere faithfully to the Withindows framework. The existing Ygdrasil user community stands to benefit from a tool that offers an improvement to existing workflow without imposing dramatic changes to that workflow. Users should be able to load scenes developed previously with a text editor and have no trouble editing scenes created completely within the graphical interface. Users are also likely to benefit from a design that allows complete backwards compatibility with previously developed virtual reality worlds. Because virtual reality worlds are effectively 3D applications in themselves, the decision was made to strengthen the underlying scene description language in order to build the application on top of it. This first class object approach allows the majority of the development effort to benefit the user development of future Ygdrasil applications. With regards to implementing the Withindows framework, two issues are most important. The underlying execution environment that will allow desktop selection to transition directly into the more general immersive image-plane selection must be considered. And, in order to keep the resulting application usable, a clear separation between global functionality and viewpoint management must be maintained.

7.2.1 Execution Environment

The Withindows framework effectively calls for desktop operating systems to provide a more flexible notion of window management. When operating systems such as Windows and XWindows allow windows to be located in three dimensional space instead of screen space, then a transition to immersive applications will become significantly easier. Because 3D window management and a generalized image-plane model are not supported by contemporary operating systems, another approach must be taken. The approach used initially by Angus was to map the XWindows bitmap to a texture and display it within a 3D environment (47). This approach is made easier today by the Virtual Network Computing (VNC) toolset developed at AT&T¹². Using VNC, an XWindow can be rendered off screen to a buffer and subsequently rendered on a texture within the environment (110). The VNC protocol also has provisions for transmitting keyboard and mouse events to the XWindow. Although VMC is a useful tool for bringing desktop applications into immersive settings, it does not provide any means of providing the remaining features of a Withindows application.

¹² <http://www.cl.cam.ac.uk/research/dtg/attarchive/vnc/xvnc.html>

The chapter introducing the framework described a hypothetical Withindows application based on the Alias Maya application. Although such an application could be built, adding any significant coordination between the user, the position of the application window and the view presented within it would require some access to the inner workings of the Maya application. And, even if viewpoints could be coordinated between viewing window and the surrounding environment, adding the ability to interactively manipulate content and access context sensitive menus within the surrounding environment would pose a significant challenge. Another approach to the problem involves constructing the application completely within the 3D graphical environment. The main application window can then be presented in a desktop window as long as some mapping between mouse position and the image-plane selection into the application is maintained. A 3D application can appear as a 2D application if is presented in parallel perspective and it is dynamically resized in response to changes in the size of the desktop application window. This approach has the advantage that all of the programming remains under the control of the application developer. A drawback to this approach is that many of the widgets and system services the application requires will have to be constructed entirely within a 3D environment.

One way to leverage the development effort of 2D widgets in a 3D environment is to make them available to future developers of Ygdrasil applications. This is the same approach that was pioneered by the original Smalltalk development environment (111). The Smalltalk language was the first object oriented programming language and it allowed users to create objects that could be reused without concern for how they were implemented. Smalltalk came with its own graphical Integrated Development Environment (IDE) that was built out of first class Smalltalk objects. This programming environment was groundbreaking because it unified the development of Smalltalk with the development of the IDE. When improvements to the underlying language were developed, the IDE inherited them immediately. An improved execution engine resulted in a faster IDE. Improved algorithm implementations improved not only applications but also the IDE itself. Changes in object interfaces become immediately recognized by the IDE because it is intimately related to the underlying language. By building the Ygdrasil graphical interface out of Ygdrasil itself the development effort to produce 3D application windows will help future Ygdrasil users create their own 3D applications. The effort to create 2D widgets and file system services will also help future users of the language create their own user interfaces. And, any effort to improve the debugging and execution model of the language required to create a complex development application is likely to improve upon the ability of future users to debug and execute their own virtual reality applications.

7.2.2 **Global Functionality and Viewpoint Management**

The general design of an application using the Withindows framework has already been described in a previous chapter. Each application window that manages a viewpoint will have a set of icons and menu options for that purpose. This includes the following common functions; zoom, rotate, pan, view-all, select-focus, view-top, view-right, view-front (Figure 35). Beyond the management of viewpoint there are a number of global functions that should not be restricted to viewing windows. The primary global functionality revolves around the ability to select individual Ygdrasil nodes and adjust their internal state variables. The typical method of invoking context sensitive menus in desktop applications is through the use of the right-click. For example, the user should be able to select a light node and adjust its on-off state either within a viewing window or in the area surrounding them. Like in many other 3D modeling applications, objects that are purely graphical in nature should be directly selectable. However, not all elements in a typical scene graph have visual representations. Many applications create a debug outline for scene elements such as lights and cameras so that they can be selected within the scene. In Ygdrasil and other scene graph description languages elements such as switches, transforms and geometry nodes are likely to be located at the same location within the 3D description of the world. For this reason, it is useful to provide a hierarchical browser to allow the user to access and organize nodes within the scene graph (Figure 36). Although it could conceivably be confined to a small portion of the viewing window, the limited resolution within immersive environments suggests that an icon to toggle the viewpoint between hierarchy view and scene view is appropriate. In the case of a hierarchy viewer, the zoom, pan, view-all and select-focus functions are no less as applicable to a 2D representation. There is also reason to believe that a combination of the select-focus, view-all and the ability to rotate the hierarchy may prove useful tool for selecting distant nodes.

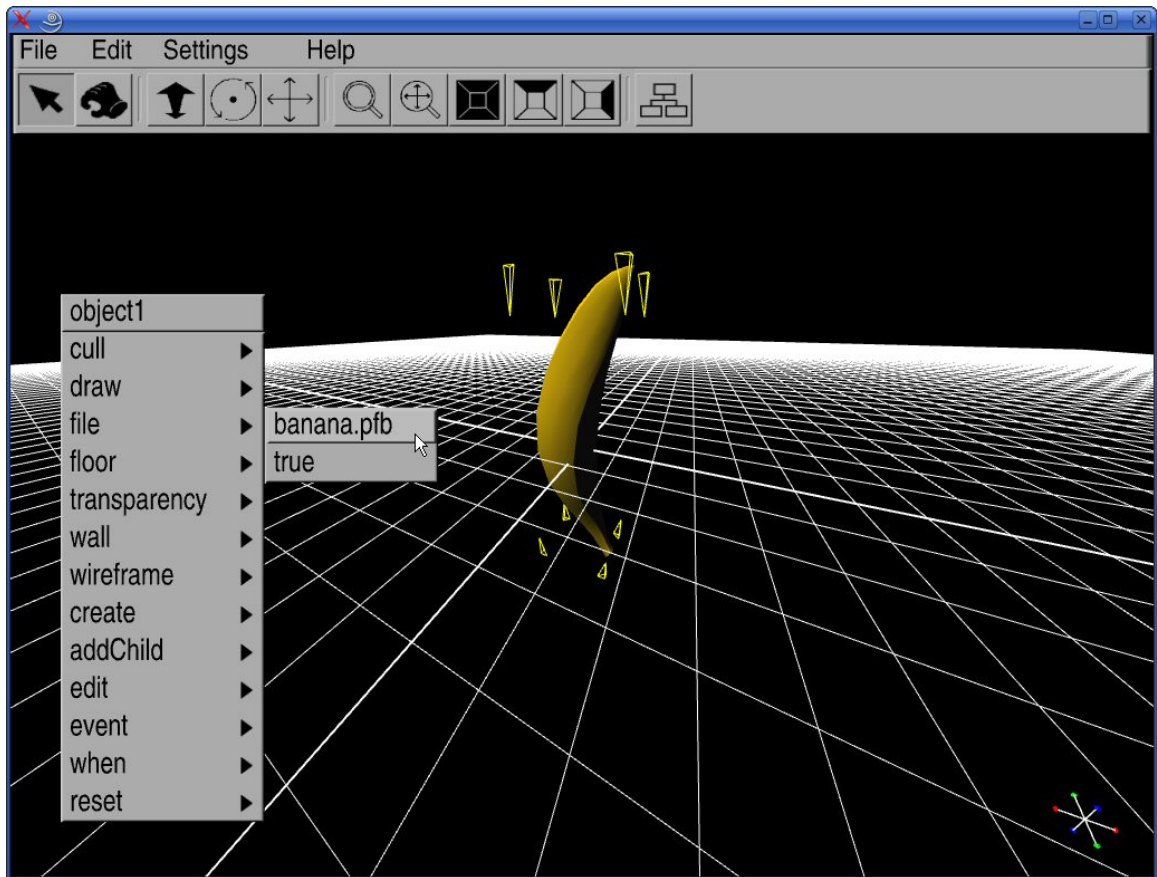


Figure 35. Alternate viewing window presented in a desktop environment. Once selected, right clicking brings up a context sensitive menu of object properties.

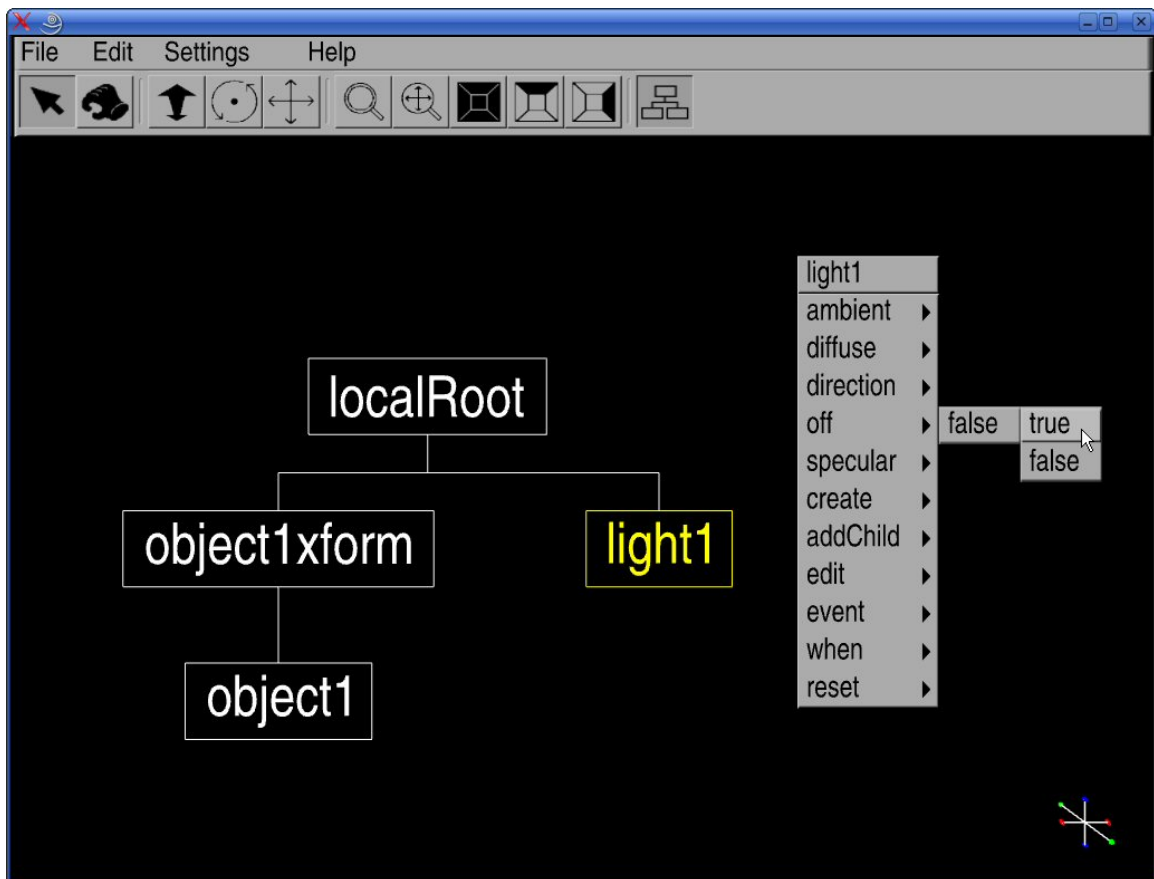


Figure 36. A button on the viewing window allows the view to toggle between the alternate viewpoint and a node hierarchy browser.

In addition to context sensitive interaction with Ygdrasil nodes, there will also be a number of global functions specific to developing scenes. The ability to load existing scenes, save out the current scene, prune the scene graph and filter the debug outlines and debug reporting of various node types are global functions that must also be accessed by the user. The Ygdrasil environment has been designed around an automatically generated `ygWorld` node that encapsulates these functions. This design choice allows developers to access these functions by addressing messages to the ubiquitous world object (i.e. `world.quit`). In a manner similar to how desktop operating systems handle context sensitivity, the context sensitive menu of the `ygWorld` node can be accessed when no other Ygdrasil node is currently selected. Typically objects are de-selected by clicking on the background, and this process is a natural one for effectively selecting the node associated with the entire virtual world.

The typical method of invoking context sensitive menus in desktop applications is through the use of the right-click. Whether an individual node or the world node has been selected, a menu will appear at the virtual cursor location. When this menu is invoked over a viewing window, the menu will be presented on the plane of the viewing window. As a result, desktop use of the interface will give a consistent presentation. When a context sensitive menu is invoked with the cursor over the surrounding scene, the menu will appear directly at the virtual cursor location and oriented towards the user (Figure 37). By encapsulating all global functionality within context sensitive menus, the viewing windows can remain dedicated to viewpoint management while the global functionality can be accessed either within a viewing window or floating in the surrounding scene. Since the viewing windows will be a critical part of the application, additional drop down menus along the top of the viewing window can provide easy access to global functions and help give the viewing window the look and feel of a typical desktop application.

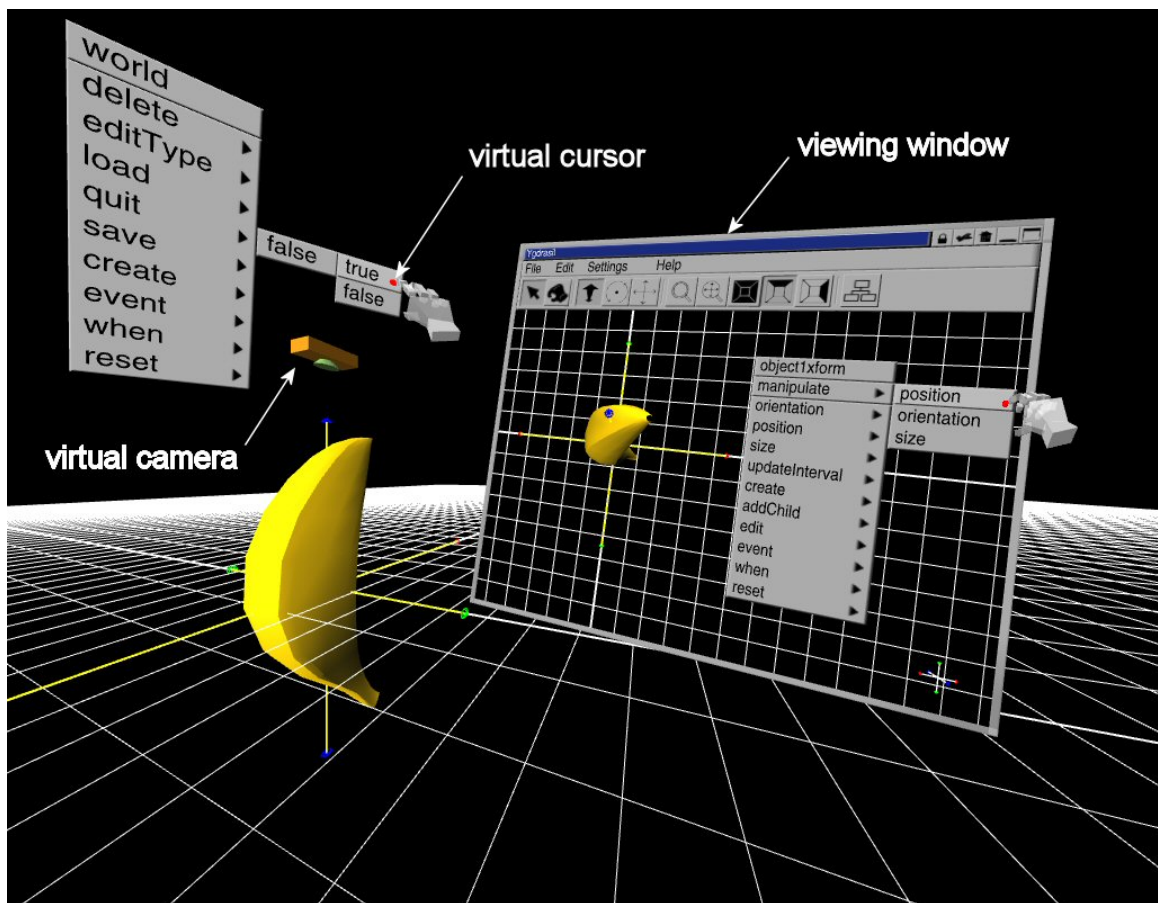


Figure 37. Context sensitive menus can be accessed by right clicking both within and outside of the viewing window. Global functions are accessed through window drop down menus or by selecting the background before right clicking.

7.3 **System Development**

The majority of the system development effort for the Ygdrasil graphical interface centered on the development interface elements that were built directly out of Ygdrasil nodes. This somewhat tedious work would not have been possible if several improvements had not been made to the underlying language. These improvements gave the underlying system the robustness and flexibility to implement a complex application consisting solely of Ygdrasil nodes. The most significant improvements to the system involved a more sophisticated macro implementation, a more robust concurrency model and a unified networking and messaging data structure. The Ygdrasil system utilizes the CAVElib VR library to manage view frustums and provide a desktop simulator. Modifications to the simulation system were necessary in order to facilitate presenting the 3D application as a desktop application.

7.3.1 **Simulator Improvements**

The CAVElib is perhaps the most well known software in the field of virtual reality development. Developed at the Electronic Visualization Laboratory, the CAVE library was the first library to provide a complete solution for the management of real-time tracking and stereo display requirements of virtual reality. The main execution loop of the Ygdrasil development system is built on top of CAVElib for OpenGL Performer library routines. The CAVElib simulator is a simulation tool built into the library that allows the user to simulate the tracking input conditions of a typical VR user in order to test applications on the desktop. Between 1992 and 2005 the majority of CAVElib applications ran on SGI computers. Today, CAVElib, the OpenGL Performer library and the Ygdrasil environment all run on Linux or Windows platforms. The original simulator interface was written in the mid 1990's, and as a result, has a somewhat dated interaction model. The user typically controls their interaction with the virtual environment by a combination of user viewpoint, tracked hand inputs and controller button and joystick input. The original design used a combination of the spacebar and mouse to mimic the inputs of the WANDA joystick. The individual controls of the viewpoint were mapped to the arrow keys, and the control of the hand input was controlled by a combination of keyboard and mouse input. Beyond the rudimentary control of the user input, a number of keyboard assigned functions allow the user control over various aspects of the display. These controls included moving the viewpoint outside the tracking volume in an orbital viewpath and drawing a visual marker to indicate the tracked hand location among others.

Although this interface was successful in helping users debug their VR applications on the desktop, the hardwired keyboard inputs prevented anyone from creating a usable desktop application from it. Furthermore, the prior viewpoint and hand interaction was based on the Cartesian coordinate system and, as such, made even the most rudimentary grabbing of objects notoriously difficult to accomplish. A new enhanced interface was added alongside the traditional interface and activated via keyword option in the CAVE configuration file that establishes tracking and display configuration. This new interface uses the mouse to control viewpoint directly as is common on PC computer games. In addition, the hand tracking control was changed to a body-centric semi-spherical system that moves the hand in a plane in front of the user viewpoint. A keyboard option changes the function of the mouse to change depth from the user and horizontal position. This enhanced interface easily allows a user to move the hand in line with a location of interest and then move the hand outward to reach it. The transition between viewpoint and hand control is now based on a mode that can be toggled with a key press. The mode can also be initialized to either mouse-look or mouse-wand within the configuration file. Options to assign each of the keyboard controls within the configuration files were also added to the simulator code.

The results of these enhancements are that users have a more natural method managing virtual travel through scenes. A combination of WANDA joystick functions mapped to the arrow keys and wand motion in the image-plane allows users to travel with more flexibility than before. And, simulator control keys can be reprogrammed to the peripheral keys on the keyboard in order to allow an application to use desktop keystrokes. However, the most important result of these changes is the ability to map the hand motion directly to the position of the desktop mouse by initializing the mode to mouse-wand. Because the position of the hand maps to the desktop mouse, image-plane selection on interfaces within the simulation environment becomes desktop interaction when the viewpoint remains fixed and the interface fills the simulator window completely.

7.3.2 Ygdrasil Architecture Improvements

The most important change made to the Ygdrasil system is architectural and will appear invisible to most users. The original system used a combination of messages passed between users and network keys passed between clients to maintain the internal state of nodes. Unfortunately, this system required that messages passed to a node be translated into internal state variables which were in turn translated into network keys for distribution to other client copies of the node. For example, the ygGeometry node has several boolean state variables that correspond to the floor, wall and draw state of the geometry. A floor geometry will support terrain following, a wall

will prevent the user from passing through and the draw state determines whether the geometry is visible. The prior implementation accepted three separate boolean messages, one each for floor, wall and draw, which were then loaded into an integer by way of a bitwise AND operation. This single integer was then sent over the network to the server and distributed to client nodes. Because the system included no systematic way of interpreting node messages, it was the responsibility of the developer to parse each message, place its content in an internal variable, declare the data type of network keys and load the network key with the appropriate data. Not only was this system difficult to program and maintain on its own, but it also made it created a burden for any attempts to save the current state in human readable form and to document node messages and their arguments.

The underlying architecture was redesigned around a message registration system and aggregate network keys. Registration of messages involves establishing a direct correspondence between message arguments and first class data types such as floats, integers and strings. This allows the system to easily report the argument signature of each message. Nodes previously had to be individually programmed to give access to their internal state but can now make each state variable available at runtime. This functionality becomes a necessity in order for a graphical interface to query a node and present the possible messages and arguments to the user. Message registration also allows the system to easily reconstruct the internal state of the node in the form of intra-node messages. This allows the Ygdrasil system to easily reproduce a human readable version of the scene file that will result in the exact state of the system. The further implication of message registration is that messages can be automatically parsed because the argument signature is known ahead of time. Once parsed into internal state variables, network keys are constructed and delivered to the network that aggregate the state variables comprising a single message. The former system only defined network keys by data type; a message that contained a float and a boolean would require two network keys to deliver the content in the message. The new architecture creates a single network key that aggregates all message arguments into a single data structure that can then be automatically parsed on the client node because the argument signature of the message is known. This unification of message, internal state variable and networking key significantly reduces the effort of developing networked nodes while potentially reducing the number of separate network messages that must be delivered.

The second important change to the underlying Ygdrasil system involves the significant tightening of the concurrency model. The prior system only processed intra-node messages at the end of each frame in order to

limit the amount of time message processing could consume. This created the possibility that a sequence of messages, one initiated by another, could play out over several frames. This architecture makes it difficult to predict exactly what the state of the system will be at any single application frame. This concurrency model allowed messages to be delayed by an increment of time was often adequate for virtual reality applications. The new concurrency model evaluates all messages immediately so that other nodes that rely on the changes can proceed without concern for when they will occur. Although modern processors have alleviated some of the concern for minimizing the time consumed during message processing, this method does not preclude managing the frame rate. An accumulated frame time can be maintained and the scene graph evaluation halted when a limit has been reached. The next evaluation of the scene graph can then be started at the same location once the frame has been rendered.

The last improvement to the system architecture has a significant impact on the ability to form program units out of re-usable pieces of code. A ygScene node was introduced that allows users to create a prototype scene and instance it an unlimited number of times within scenes. The original Ygdrasil system did not automatically increment the names of nodes and, as a result, required that any duplication of code be assigned different names in order to avoid conflicts with intra-node messaging. The system was changed to automatically increment node names when a previous node of the same name was found. A node was then created to not only load a scene file and increment node names, but to also increment all node name references within that scene. The result is that a single scene file of unlimited internal complexity can be re-used without concern that the internal messages between nodes will be compromised. This functionality is also provided by the VRML 97 standard in the form of the PROTO command and provides a similar functionality to developers using that language. As with the VRML macro facility, messages can be directed at the individual parts of each instantiated scene by addressing them by their original name through the ygScene node that loaded them.

7.3.3 Ygdrasil Node Improvements

The remainder of the improvements to the Ygdrasil system consisted of new or altered nodes that make increased functionality available to the user. Several nodes were created to access file system content, query the available node types and to access node messages and their arguments. Each of these nodes produces first class data types such as strings, floats and integers that must be stored in intermediate variables or used in calculations. In response to the limitations of the existing scripting language, a set of first class string, float and

integer nodes were developed to aid this process. In contrast to VRML, which uses a separate scripting language to handle intermediary calculations, the decision was made to incorporate calculations directly into the script in the form of scene graph nodes. Calculations are performed on these nodes by means of recursion on multinary trees. All of the calculation nodes are derived from an elementary value node that stores a dynamic array of floating point values. Nodes that perform floating point calculations such as add, multiply and minimum are arranged in the scene graph and derive their own values from calculations on their immediate children nodes. A set of boolean nodes for functions such as and, or, greaterThan and isEqual were also derived from the base value node. In this manner, arbitrarily complex floating point and logical calculations can be performed within a scene graph.

Another node type that was derived from the base value node is the timer node. This node is of greater significance to the users of the development system than to the development of the graphical interface application. Many of the changes mentioned above such as access to internal state variables and intermediate calculations directly contribute to the ability of users to create more complex interactions directly within scene files. The original collection of nodes available to users often had their concept of dynamic timing built into them. A node designed to change a material property would require a message to establish the duration of the change and could only be used to change the single property it was designed for. This strategy led to the creation of nodes for every eccentric purpose with names like `turner`, `fadeOut`, `fadeIn`, `toAndFrom`, etc. This coarse design granularity limited the power of the scripting language and encouraged users to write their own special purpose nodes. In order to improve the expressive power of applications built from existing components, a separate timer node was created that allows a single node to dynamically control an arbitrary set of parameters. This change, and others aimed at separating nodes into their constituent parts, increased the ability of users to construct highly dynamic scenes without needing to create their own custom nodes.

While the aforementioned improvements help developers accomplish relatively low-level tasks directly within the scripting environment, the access to high-end rendering techniques is more often a serious source of frustration for virtual reality artists. The technical expertise required to implement techniques like video texturing, stencil buffering and rendering to texture force artists to either learn the skills of a computer scientist or solicit help from one. In order to facilitate access to these techniques from within the scripting language a strategy was undertaken that seeks to segment the functionality of Ygdrasil and tools such as Photoshop and Maya. These

tools are often familiar to artists are already an excellent environment for the fine manipulation of content. The strategy seeks to avoid detailed manipulation of content primitives from within Ygdrasil in favor of more high level manipulation of content. This strategy is best described in terms of concrete examples. The morpher node accepts two or more models that have been formatted appropriately within Maya and allows the dynamic morphing between vertex and texture states with a single keyframe value. The applyTexture node applies a sub-image onto an existing image at a designated location. This node allows the application of burn marks and transparent holes onto any texture in the environment by specifying only the relative location on the larger texture at which the sub-image should be applied. Another part of the strategy is the observation that advanced rendering techniques can often be broken up in terms of source content and destination content (Figure 38). For example, the application of a clipping plane relies on a source position in the world and a destination section of the scene graph to which it applies. A rendering to texture relies on a source section of the scene graph that is to be rendered and a destination texture node that is to be loaded with the result. The application of a stencil buffer relies on a source section of the scene graph that is to be rendered subject to the buffer and a destination section of the scene graph that is to be rendered to form the buffer. This segmentation of source and destination allows a number of high-end rendering techniques to be applied directly within the scene graph in a consistent manner. By allowing artists to avoid the technical details of high level programming languages and graphics libraries, this improved Ygdrasil development environment helps artists focus their attention on the creative process and leverage the technical skills they have already acquired with other tools.

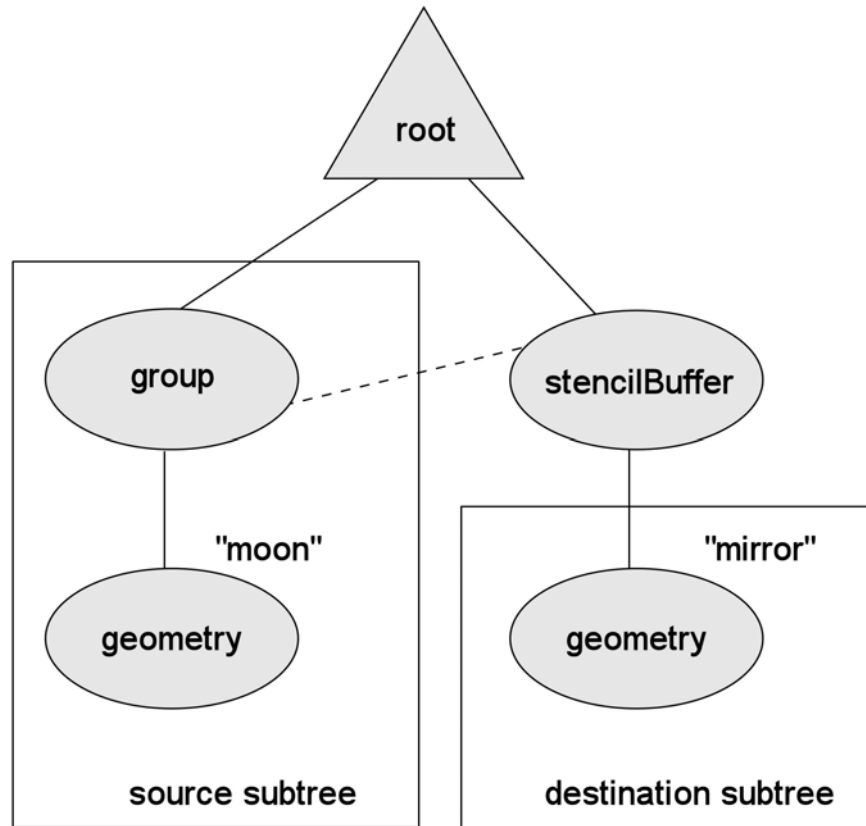


Figure 38. Advanced rendering techniques can often be broken down into the application of a source subtree of the scene graph (the rendering of a moon for example) onto a destination subtree (a mirror defining the stencil buffer mask).

7.4 System Evaluation

There are two groups of users that stand to benefit from the development of a Withindows based graphical interface for Ygdrasil. The first group consists of novice users that have little or no experience with virtual reality development or the Ygdrasil system. The second group that will benefit is the established group of experienced VR developers already using Ygdrasil or similar development systems. The graphical interface was evaluated during the course of one semester in a classroom environment with novice users. This evaluation was invaluable for discovering usability issues related to basic functionality and for evaluating the overall utility of the application in a teaching environment. The interface was also evaluated in the context of developing more advanced virtual environments with two groups of experienced artists and the author. This process allowed the application to be evaluated in several different workflow environments. The results of the work with more

complicated environments found some usability issues associated with the implementation but, overall, validated the expected utility of the application.

7.4.1 Classroom Evaluation

Instruction with the Ygdrasil graphical interface in a classroom environment is an excellent opportunity to evaluate and improve the software over a long duration. The highly structured environment and the long student commitment make it possible to thoroughly evaluate the usability of the software in a desktop environment. An introduction to virtual reality programming class has been taught in the art department at the University of Illinois at Chicago since 1995. Between 2001 and 2006 the author instructed the class and used the Ygdrasil system exclusively for laboratory exercises and student projects. The newly developed graphical interface was used by the final group of students in the spring of 2006. Using the new system throughout the semester allowed the basic functionality to be rigorously debugged and improvements to be made that facilitate use in a teaching environment. The graphical interface significantly reduced the number of programming errors and allowed a more streamlined and visually oriented workflow. The system was well received by the students but could not be evaluated in an immersive environment due to software and hardware problems.

Approval was granted by the university Investigational Research Board to evaluate the students in the context of meeting the class requirements. Five students took the course and each signed a consent form at the beginning of the semester and agreed to fill out a questionnaire at the end of the class. Of the five students, four were novices and one was repeating the class for additional credit and exposure to VR development. The class laboratories were redesigned in anticipation of having the graphical interface available for the duration of the semester. Each student completed 12 laboratories using the graphical interface on the desktop exclusively. The laboratories covered the usual range of topics from loading and placing models to building complex dynamic behaviors. During the course of the lab work, a number of system bugs were discovered. The system errors ranged in severity from basic menu selection and data input to system crashes. These bugs were usually corrected in time for the following class and did not prevent students from completing their work.

A coherent laboratory series that exercised the important aspects of development with the system was developed and refined. This allowed usability problems to be identified and addressed over the course of several months. Students were able to work in an environment that produced almost no syntax errors due to the

structured nature of the interface. Students were also able to significantly reduce the amount of time they spent on trial-and-error positioning of models and interaction areas. When questioned, each of the five students indicated that they felt the software was usable and that they enjoyed their experience with it. The single advanced student indicated that the software made a dramatic improvement in her ability to complete the laboratories. This student was able to use the system to make some improvements to a project developed during the previous semester. However, she also found the software in need of improvements to handle this large and complex project. This subject is covered in more detail in the next section.

Because the application consisted of nothing more than a viewing window presented over the 3D simulator, students could accidentally toggle the mode to mouse-view and find themselves looking at the application window from an oblique angle. This caused some confusion until students were introduced to the process of testing their scenes. The application included a menu function to minimize the viewing window to immediately engage the surrounding scene in a first person manner. This WYSIWIG access reduced the amount of time students spent saving and restarting simulations in order to debug their applications. Ideally the students would have been exposed to the interface in an immersive setting. Unfortunately, the limitations of the legacy SGI hardware running the EVL CAVE system could not handle the increased processing requirements of the system.

The functionality provided by the graphical interface is not unique. Other graphical virtual world development systems such as CosmoWorlds, WorldUp and Division provide the ability to program in a structured environment and immediately exercise the results on the desktop. The main result of the classroom evaluation was the validation of the application for desktop use. One interesting result was the impression that students spent more time learning the interface than they did learning the underlying concepts. In contrast to prior classes, the evaluation class was instructed exclusively in the graphical environment and explanations about the resulting script were only addressed when they appeared appropriate. Despite the syntax errors and trial-and-error placement techniques of the prior teaching environments, a lack of graphical interface may have helped focus students more on the scripting language. It is at this point that the ability of the underlying language to produce human readable scripting becomes an advantage. Because the saved scripts have not changed their format, future instruction can easily include a more balanced mix of scripting and graphical interface development. The discussion in computer science about the merits of graphical and non-graphical programming environments is a

lively one. While the graphical interface is useful for object placement and smaller projects, programming more complex environments benefits from a greater understanding of the underlying language.

7.4.2 Expert Evaluation

A virtual world development environment must satisfy a more rigorous set of criteria for expert users. Expert users are much more likely to be focused on complex interactivity than novice users. These complex behaviors consist of messages passed between nodes, and often a single press of the controller button can initiate a cascade of a dozen messages. For this reason, the programming of behaviors is a critical component of any successful development system and this puts a burden on the graphical interface to represent the overall structure of the behaviors to the user. However, because the development effort was focused on realizing the benefits of the Withindows framework, supporting object behavior programming was not the strength of the graphical interface.

The Ygdrasil graphical interface was used in with three projects that involved highly dynamic content. The first project was a collaboration between two experienced VR artists and the author that used the new interface throughout the duration of the project. The interface was also used to continue development of another highly interactive project with two different VR artists that had been started before the interface was available. The final project was the development of the testbed application used to conduct the user study presented in the prior chapter. In each of these projects, the immersive application was a valuable tool for the same reasons that motivated the Withindows framework. Object positioning tasks were aided by alternate viewpoints, immersive access to the full desktop function set proved useful and image-plane selection on objects in the surrounding environment and within viewing windows proved superior for object selection and widget interaction. The evaluation was not formal, the author was involved in all three projects, and a number of implementation issues were discovered. However, the system did demonstrate the basic principles of the Withindows framework, that image-plane selection and alternate viewing windows facilitate the single-authoring of productive transitional applications.

7.4.2.1 Desktop Workflow

The first expert use of the system came with the development of a project related to Swedish folklore. The Norde project was initiated by VR artists Bino & Cool from The Royal Institute of Technology in Stockholm

Sweden. These electronic artists have extensive experience building VR applications and previously created a large tele-collaborative world using Ygdrasil in 2003. The graphical interface was used on the desktop to sketch out the initial layout of the new environment along with interaction areas. The author then added the more complex behaviors to the scene with a text editor. The interactions were tested in the graphical editor and most significant changes were made directly in the text file and reloaded into the editor. However, one benefit that the new system brought was the ability to easily debug behaviors by accessing objects within the hierarchy viewer. Some aspects of the interaction are part of a cascade of events that occur after or without user interaction. The desktop interface made it possible to highlight the node responsible for a behavior, simulate the initiation of that behavior and watch the result dynamically from an arbitrary viewpoint. This was a significant advantage over the prior system because lack of runtime access to objects meant that virtual worlds had to be evaluated in a very serial manner. Small corrections to behaviors were made within the editor and then the result was saved out to the text file. The simulator was also used on the desktop to create a combined view of the viewing window and the surrounding environment. The unique nature of image-plane selection made it easy to interact with the viewing window even when it was presented much smaller and at oblique angles to the viewpoint. This combined view allowed the simultaneous user triggering of interaction areas and the viewing of the result within the viewing window. Again, it was then possible to make small changes to the scene file via direct selection of objects or through the hierarchy browser.

7.4.2.2 Immersive Workflow

Immersive interaction was provided by a single wall passive stereo system known as a C-Wall. This system uses a wired Ascension Flock of Birds tracker and a 10 foot wide by 8 foot high rear projection screen. Although the viewing window could not be placed directly under hand with this type of setup, the large size of the display did allow simultaneous interaction with the viewing window and the surrounding environment. Again, the alternate viewpoint was useful for fine tuning the position of the interaction areas because both first person interaction and object placement could happen almost simultaneously. This same task methodology was used with the third project, the study testbed. This project required that the user seated in a fixed location easily see and interact with each of the eight possible window locations at any one time. Four boxes were sized, positioned and transparency adjusted in the environment through the viewing window while the author was seated in the designated user position. These two immersive manipulation scenarios both highlight the utility of maintaining a first person perspective while adjusting the position and size of objects and interaction zones with respect to it.

The second expert project that utilized the graphical interface was a highly interactive project with a novel virtual travel system. The Way Of Making Bubbles (WOMB) project used head position relative the center of the CAVE to translate the user and a swiping motion of the hand to rotate the world and hence control the direction of travel. This project, in collaboration with Constantin and Marilu Boytscheff of the University of Konstanz, Germany, was started before the Ygdrasil graphical interface was developed and had proceeded into a fine tuning stage. The scene graph of the application was extremely broad and it quickly became obvious that this made it very difficult to navigate the tree for nodes of interest. Zooming out to view the entire graph reduced the size of each node to degree that made their respective text impossible to read. This problem motivated two solutions. The first solution stems from the observation that during testing and debugging the user frequently needs to return to the same viewpoints. User defined bookmarks can allow the user to save a viewpoint and then recall it easily. As is done frequently with video game save states, these bookmarks can be annotated with a snapshot of the viewpoint. In the case of the Ygdrasil graphical interface, both viewpoints of the secondary world and viewpoints into the node hierarchy are logical uses of this type of bookmarking system. Although not implemented into the Ygdrasil interface, this system was incorporated into a prior project that also used an alternate viewing window for selection, teleportation and context sensitive information (112).

One problem with the use of bookmarks is that the user must actively engage in the process of creating them during the immersive development process. A second solution relies on the acknowledgement that interactive development and tuning activities often rely on a relatively small subset of nodes. A combination of depth filtering and special purpose nodes can help reduce the effective complexity of the visible scene graph. A special purpose node inserted into the scene graph can indicate that a sub-graph of the tree should not be displayed. Some highly dynamic scene graphs, such as particle systems, actively create and delete nodes within the scene graph. Segmenting parts of the scene graph from view prevents the frequent reorganizing of the node hierarchy representation. A depth filter was also added to the hierarchy viewer to allow the user to cull nodes beyond a certain depth in the representation. Exceptionally broad parts of the graph were then moved to a lower location in the scene graph and culled during runtime if necessary. Frequently, nodes that need adjustment cannot easily be moved to an arbitrary location within the scene graph. As a result, a technique similar to the model-view-controller graphical interface design was used to place surrogate nodes at the top of the graph that would in turn adjust nodes deeper in the scene graph. For example, a value node was created to adjust the width,

height and depth of an interaction area. This value node was then assigned the behavior of adjusting the interaction zone when its values were updated. This procedure has the effect of creating a customized user interface to the scene graph nodes at the top of the scene graph. When combined with depth filtering, the node hierarchy allowed a custom interface to be organized and accessed during immersive application tuning sessions.

7.4.2.3 Viewpoint Management

The Ygdrasil interface was developed using the picture window strategy for alternate viewing windows. This means that changes to the user viewpoint allow the user to see a different part of the secondary scene through the viewing window. This technique contributed to both advantages and disadvantages that on the whole suggest that a viewing window strategy would find overall greater usability. The picture window strategy was implemented with a stencil buffer to allow a stereo rendered view of the surrounding scene from another viewpoint. In practice, users kept the viewing window relatively small in order to keep an overview of the surrounding scene. As a result, users were able to view and interact with a greater portion of the secondary scene by moving their viewpoint to bring objects into view. However, this ability often proved a detriment because, in contrast to a desktop system, users often take advantage of the freedom to move change their own viewpoint. The system was implemented with surround-fixed windows. This meant that floating windows, although oriented towards the user when positioned, stayed in a fixed position relative to the physical space around the user. Users would often adjust a viewpoint within the viewing window and then turn to look at content in the surrounding scene. Upon returning to work on the viewing window, the user would loose track of the viewpoint they had been using and have to go to extra effort to re-asertain that viewpoint. This problem ended up forcing users to keep their head in a fixed viewpoint while simultaneously manipulating objects in a viewing window and appreciating their affect in the surrounding environment.

Another problem with using a picture window strategy is that the effect of user viewpoint changes is greater on objects at a distance. Image-plane selection is a combination of hand position and viewpoint position. As an object moves farther away from the user the effect of head motion on the image-plane target increases. Because a viewing window strategy is insensitive to head motion, the object depth is effectively the surface of the viewing window and thus reduces the effect of head motion when using image-plane selection. The one advantage of a picture window strategy is the ability to view additional content by changing the viewpoint alone. However, there are already three different strategies available to the user to adjust the viewpoint. The unlock

function, that disconnects secondary scene from the viewing window, allows the window to be moved to bring objects into view in an analogous manner to viewpoint changes. And, the basic 2½D rotation and panning techniques along with the 6 DOF combined technique allow the user to adjust viewpoint without moving the position of the viewing window. In the context of several intuitive strategies for managing the viewpoint, it appears that a viewing window strategy is more appropriate choice in a practical Withindows framework. A viewing window strategy also eliminates any concerns about collaborative viewpoint awareness. The strategy can easily be implemented using a viewpoint rendering to texture. This method ensures that each other participant sees the same viewpoint as the primary user and obviates the need for a specific strategy to handle that case.

7.4.2.4 Using 6 DOF Input

Six degree of freedom input plays a role in object manipulation, window management and viewpoint management in Withindows applications. As predicted, image-plane grab allowed objects to be manipulated in depth at a scale appropriate to their distance. It proved even more difficult than usual to estimate the depth of an object when the viewpoint was confined to a framed window. As a result, some scaffolding was added to support the task in the form of a drop line from the object to the horizontal surface. This type of scaffolding is commonly used in 3D modeling applications to aid the perception of depth when using desktop 2½D techniques. The Ygdrasil graphical interface supplied a highlighted planar grid that could be oriented perpendicular to any one of the three orthogonal axes. Typically users keep the axis aligned with the horizontal plane and rely on a tick-marked drop line from the object onto the horizontal grid during constrained manipulations that include height adjustments. This drop line was also useful in determining object depth during image-plane grab manipulations. As with any depth operation that is scaled relative to the user, movements from a distance often required dropping the object at a closer distance to re-establish the control-display relationship between user and the object. This was especially true when attempting to drag-and-drop an object between the surrounding and secondary scenes. A drag-and-drop action from a distant viewing window position onto a surface next to the user often caused confusion when the object would appear beyond an opaque surface. The same was true when dragging objects at a distance in the surrounding scene into a location where a different scaling relationship was more appropriate. Once the object was brought into view in the new location, dropping the object and grabbing it again established a more appropriate control-display relationship. The 2½D drag-and-drop technique proved more appropriate for drag-and-drop tasks in simple situations because the object effectively snaps to the horizontal surface when it is the constrained surface. However, this technique also caused confusion when the

original object height was significantly higher or lower than the desired location. For the same reason, drag-and-drop actions using other 2½D axis constraints can also cause considerable confusion. For these reasons, the image-plane grab appeared best suited for more arbitrary drag-and-and drop tasks.

Viewpoint management tasks with the 6 DOF input proved a useful shortcut. As with typical desktop interfaces, the Ygdrasil graphical interface uses two cursor modes. In the selection mode, button clicks over the viewing window allow objects to be selected and manipulated. A button on the viewing window allows the user to switch to a viewpoint mode where click-and-drag actions initiate either rotate, pan or zoom actions. The 6 DOF viewpoint management was initiated by holding down the lock/unlock button and presented a shortcut because it could be initiated regardless of the current mode. The initial design only called for the 6 DOF device to control 3 degrees of freedom; azimuth, zenith and zoom. The ability to simultaneously control pan was incorporated into the initial design. Although conceptually possible to do, the addition of the pan made the control very difficult to use, and it was later removed. It also became clear that the task was very sensitive to unsteadiness in the input device. A filter was added between the hand input and the resulting viewpoint rotation and zoom. The linear averaging filter helped to stabilize the input and did not appear to introduce any usability problem. The 6 DOF viewpoint management offers a convenient modeless shortcut and the 6 DOF image-plane grab offers a robust method of moving objects between viewing windows and surrounding environment. Overall, the 6 DOF input tasks appear to be of significant value during immersive tasks.

8. CONTRIBUTIONS

The goal of any scholarly research is to contribute to an ongoing dialectic. This thesis makes its contributions to the general field of 3D user interfaces and to the more specific areas of virtual reality and augmented reality. The most important theoretical contribution of this work is Withindows, a unified framework for the development of user interfaces for desktop, immersive and transitional configurations in between. This framework is based on two important methodological contributions; the use of image-plane selection in stereo environments via a virtual cursor in the dominant eye and extensions to through-the-lens techniques that resolve prior inefficiencies.

The Withindows framework relies on a number of ideological contributions to the ongoing dialectic surrounding virtual and augmented reality. Foremost among these contributions are recognition of the importance of supporting transitional configurations between desktop and immersion, the acknowledgement of the need to provide alternate viewpoints for many virtual and augmented reality tasks and the utility of moving prior physically intensive 3D tasks to a less fatiguing working position. This thesis also makes empirical and applied contributions to the field through a user study and a working prototype. A comparison of image-plane and ray-casting use on traditional interfaces found significant support for the thesis that image-plane selection can be used on interfaces presented under the hand with little fatigue. And, a Withindows-based proof of concept application targeted at the needs of an existing user group was developed and evaluated with novice and expert users. This chapter seeks to document the contributions of this thesis and serve as a brief summary of the document as a whole.

8.1 **Ideological Contributions**

The thesis presented here makes a number of ideological contributions, all of which are summarized in the chapter on guidelines for productive 3D user interfaces. A primary ideological contribution is the concept that contemporary trends of a geo-referenced internet, the ubiquitous internet and ubiquitous displays are pushing computation off of the desktop and into 3D space. A likely merging of the computational power of the desktop with context sensitivity in the physical world suggests that, instead of forming a completely unique domain, 3D user interfaces represent a superset of 2D interfaces. It follows from this idea that 3D user interfaces need a holistic

framework focused on creating applications that operate along the continuum between desktop and full immersion.

A second important ideological contribution is based on the observation that much of the research into primary virtual reality tasks has been unwilling to abandon immersion as a goal or to overcome corporealism, prejudices towards physically intuitive methods. A re-conceptualization of virtual and augmented reality is offered that seeks to separate the goals of simulation from those of efforts to increase the computational power of the individual, described here as augmentation. From this observation follows the significant contribution that both virtual and augmented reality user interfaces should make alternate viewpoints beyond those of first person perspective available when applicable. It is also argued that the limitations of first person perspective make traditional 2D user interface elements a natural fit for application in 3D space.

8.2 Theoretical Contributions

The main theoretical contribution of this thesis is the Withindows framework. Withindows is a unified framework for developing applications that operate along the continuum between desktop and immersive use. Of significant contribution are extensions to the two principle components of the Withindows framework, image-plane selection and Through-the-Lens techniques. It is shown how image-plane selection can bring consistency to interactions in stereoscopic environments by combining a virtual cursor in the dominant eye and a reinforcing cursor in the non-dominant eye. Immersive image-plane selection is formulated as a generalization of the traditional desktop interface with fixed viewpoint, monoscopic display and a virtual cursor restricted to the visual plane. This theoretical basis unifies desktop and immersive interaction and allows the principle components of stereoscopy, head-tracking and 6 DOF inputs to be added or removed as necessary. Several theories are put forward in support of making image-plane selection the canonical interaction method across platforms. It is argued that image-plane selection is more robust to interactions on affine distorted surfaces when compared to ray-casting. And, it is argued that image-plane selection is more accurate at a distance due to a previously undocumented problem described here as the alignment problem.

The analogy between desktop and immersion is carried forward into another significant contribution as it is shown how the addition of image-plane selection to through-the-lens techniques solves prior scaling problems with object manipulation techniques. The claim is made that the extensions described here to TTL techniques

solve prior inefficiencies with object selection, viewpoint management, virtual travel and global search. Another significant contribution is the theory that image-plane selection on through-the-lens windows allows the primary immersive tasks of object selection, object manipulation, virtual travel and global search to proceed with a minimum of fatigue. The theory is also put forward that image-plane selection works naturally with clutching techniques, flat surfaces and gaze based techniques to potentially reduce fatigue in more general cases.

While it is argued that image-plane selection techniques are effectively 2 DOF, the Withindows framework accommodates higher DOF input devices through extensions to viewpoint management and object manipulation techniques that do not supersede those default methods. Other benefits attributed to the framework include application to Universal Design principles and single authoring development strategies. The Withindows framework is also based, in part, on the theory that a single canonical interaction method offering exposure to immersive applications on the desktop will increase their usability and ensuing adoption.

8.3 Empirical and Applied Contributions

This research makes empirical contributions to the field of 3D user interfaces through the results of a user study. The user study found that users reported similar levels of user fatigue when using both image-plane selection and ray-casting on flat interfaces presented below the hand. The study also found similar levels of ease of use under the same conditions for one of the two study tasks, a drag-and-drop task. Despite some eye strain that was likely caused by poor experimental setup, these results show that image-plane selection using a dominant eye cursor presents no more significant usability problems than ray-casting when used under hand. The implications of these results are that one significant part of the Withindows framework, image-plane selection on traditional interface elements presented underhand, does not present significant usability problems. The further implication of the results is that a strategy of moving primary immersive tasks of significant duration to windows underhand is a valid strategy for reducing fatigue levels in 3D user interfaces. Although not statistically indicated, the study results also appear to reinforce prior work showing that image-plane selection is faster than ray-casting on spatially homogeneous targets.

The applied contributions of this research are the development of a working proof of concept application and the evaluation of some usability aspects with actual users. The virtual world builder developed by this research effort demonstrated that a single-authoring approach can produce an application using the Withindows

framework that transitions from desktop use to immersive use while retaining its full functionality. The working application is also a significant contribution because it directly addresses the needs of an established group of users in a domain where immersive use offers concrete benefits. Classroom evaluations of the software indicated that it was an improvement over an existing script-based authoring tool and, as a result, valuable in an educational setting. Expert evaluation with the application found alternate viewpoints to be valuable for object manipulations with respect to user viewpoint. Use of the application with experts also helped resolve questions about alternate viewpoint rendering and management implementation issues.

CITED LITERATURE

1. Bush, V.: As We May Think. *The Atlantic Monthly*, July, 1945.
2. Sutherland, I. E.: Sketchpad - A Man-Machine Graphical Communication System. *Proceedings of the AFIPS Spring Joint Computer Conference*, p. 329-346, 1963.
3. Engelbart, D.C.: Augmenting Human Intellect: A Conceptual Framework, *Stanford Research Institute*, Summary Report AFOSR-3233, Menlo Park, CA, October, 1962.
4. Sutherland, I. E.: A head-mounted three dimensional display. *Proceedings of the AFIPS Fall Joint Computer Conference*, volume 33, pages 757-764, 1968.
5. Fisher, S., McGreevy, M., Humphries, J. and Robinett, W.: Virtual Environment Display System, *ACM 1986 Workshop on 3D Interactive Graphics*, Chapel Hill, North Carolina, October 23-24, 1986.
6. Cruz-Neira, C., Sandin, D. J., DeFanti, T. A.: Surround-screen projection-based virtual reality: the design and implementation of the CAVE. *Proceedings of the 20th annual conference on Computer graphics and interactive techniques*, p.135-142, September, 1993.
7. Bowman, D. A., Wineman, J., Hodges, L. F. and Allison, D.: Designing Animal Habitats Within an Immersive VE. *IEEE Computer Graphics and Applications*, v.18 n.5, p.9-13, Sept 1998.
8. Schmalstieg, D., Encarnação, M. and Szalavári, Z.: Using transparent props for interaction with the virtual table. *Proceedings of the 1999 symposium on Interactive 3D graphics*, p.147-153, Atlanta, Georgia, April 26-29, 1999
9. Lindeman, R. W., Sibert, J. L. and Hahn, J. K.: Hand-Held Windows: Towards Effective 2D Interaction in Immersive Virtual Environments. *Proceedings of IEEE Virtual Reality*, p.205-212, 1999.
10. Darken, R., Cockayne, W. and Carmein, D.: The omni-directional treadmill: a locomotion device for virtual worlds. *Proceedings of the 10th annual ACM symposium on User interface software and technology*, p.213-221, October 14-17, Banff, Alberta, Canada, 1997.
11. Cournia, N., Smith, J. D. and Duchowski, A. T.: Gaze- vs. hand-based pointing in virtual environments. *CHI '03 extended abstracts on Human factors in computing systems*, April 05-10, Ft. Lauderdale, Florida, 2003.
12. Mapes, D. and Moshell, J.: A Two-Handed Interface for Object Manipulation in Virtual Environments. *Presence: Teleoperators and Virtual Environments*, 4(4), p. 403-416, 1995.
13. Bowman, D. A., Koller, D. and Hodges, L. F.: Travel in Immersive Virtual Environments: An Evaluation of Viewpoint Motion Control Techniques, *Proceedings of the 1997 Virtual Reality Annual International Symposium (VRAIS '97)*, p.45, March 01-05, 1997.
14. 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*, p.265-272, Denver, Colorado, May 07-11, 1995.
15. Darken, R. P. and Sibert, J. L.: A toolset for navigation in virtual environments. *Proceedings of the 6th annual ACM symposium on User interface software and technology*, p.157-165, Atlanta, Georgia, December, 1993.
16. Bowman, D. A., Davis, E. T., Hodges, L. F. and Badre, A. N.: Maintaining spatial orientation during travel in an immersive virtual environment. *Presence: Teleoperators and Virtual Environments*, 8(6), p.618-631, 1997.

17. Darken, R. P. and Sibert, J. L.: Wayfinding strategies and behaviors in large virtual worlds. *Proceedings of the SIGCHI conference on Human factors in computing systems: common ground*, p.142-149, Vancouver, British Columbia, Canada, April 13-18, 1996.
18. Koh, G., Wiegand, T. E., Garnett, R. L., Durlach, N. I., and Shinn-Cunningham, B.: Use of Virtual Environments for Acquiring Configurational Knowledge about Specific Real-World Spaces: I. Preliminary Experiment. *Presence: Teleoperators and Virtual Environments*, 8(6), p.632-656, December, 1999.
19. Zambaka, C., Lok, B., Babu, S., Xiao, D., Ulinski, A., and Hodges, L. F.: Effects of Travel Technique on Cognition in Virtual Environments. *Proceedings of the IEEE Virtual Reality 2004*, Washington, DC, 149, March 27 - 31, 2004.
20. Bolt, R. A.: "Put-that-there": Voice and gesture at the graphics interface. *Proceedings of the 7th annual conference on Computer graphics and interactive techniques*, p.262-270, Seattle, Washington, July 14-18, 1980.
21. Poupyrev, I., Weghorst, S., Billinghamurst, M. and Ichikawa, T.: Egocentric object manipulation in virtual environments: empirical evaluation of interaction techniques. *Computer Graphics Forum, EUROGRAPHICS '98*, 17(3), p.41-52, 1998.
22. 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*, p.39-43., Providence, Rhode Island, April 27-30, 1997.
23. Leigh, J., Johnson, A.: CALVIN: an Immersimedia Design Environment Utilizing Heterogeneous Perspectives. *Proceedings of IEEE International Conference on Multimedia Computing and Systems '96*, p.20-23, Hiroshima, Japan, June 17 - 21, 1996.
24. Pierce, J. S. and Pausch, R.: Navigation with Place Representations and Visible Landmarks. *Proceedings of the IEEE Virtual Reality*, p.173, Washington, DC, March 27 - 31, 2004.
25. Viegas, J., Conway, M. J., Williams, G., and Pausch, R.: 3D magic lenses. *Proceedings of the 9th Annual ACM Symposium on User interface Software and Technology*, p.51-58, Seattle, Washington, November 06 - 08, 1996.
26. Guven, S., Feiner, S.: Visualizing and navigating complex situated hypermedia in augmented and virtual reality. *IEEE/ACM International Symposium on Mixed and Augmented Reality*, p.155-158, Santa Barbara, California, October, 2006.
27. Fukatsu, S., Kitamura, Y., Masaki, T., and Kishino, F.: Intuitive control of "bird's eye" overview images for navigation in an enormous virtual environment. *Proceedings of the ACM Symposium on Virtual Reality Software and Technology*, p.67-76, Taipei, Taiwan, November 02 - 05, 1998.
28. Koller, D. R., Mine, M. R., and Hudson, S. E.: Head-tracked orbital viewing: an interaction technique for immersive virtual environments. *Proceedings of the 9th Annual ACM Symposium on User interface Software and Technology*, p.81-82, Seattle, Washington, November 06 - 08, 1996.
29. Poupyrev, I., Billinghamurst, M., Weghorst, S., and Ichikawa, T.: The Go-Go Interaction Technique: Nonlinear Mapping for Direct Manipulation in VR. *Proceedings of the 9th Annual ACM Symposium on User interface Software and Technology*, Seattle, Washington, p. 79-80, 1996.
30. Bowman, D. A. and Hodges, L. F.: An evaluation of techniques for grabbing and manipulating remote objects in immersive virtual environments. *Proceedings of the 1997 symposium on Interactive 3D graphics*, p.35-38, Providence, Rhode Island, April 27-30, 1997.
31. Mine, M. R., P. Brooks, F. P., Carlo, H. and Sequin, C. H.: Moving objects in space: exploiting proprioception in virtual-environment interaction. *Proceedings of the 24th annual conference on Computer graphics and interactive techniques*, p.19-26, Los Angeles, California, August, 1997.

32. Pierce, J. S. and Pausch, R.: Comparing Voodoo Dolls and HOMER: Exploring the Importance of Feedback in Virtual Environments. *Proceedings of the SIGCHI conference on Human factors in computing systems: Changing our world, changing ourselves*, Minneapolis, Minnesota, April 20-25, 2002.
33. Norman, D.: *The Psychology of Everyday Things*. New York, Basic Books, 1988
34. Ammoura, A., Zaïane, O. R., and Ji, Y.: Immersed Visual Data Mining: Walking the Walk. *Proceedings of the 18th British National Conference on Databases: Advances in Databases*, p.202-218, July 09 - 11, 2001.
35. Bowman D., Poupyrev I., LaViola J. and Kruijff E.: *3D User Interfaces: Theory and Practice*. Addison-Wesley/Pearson, p.417, Boston, 2005.
36. Bowman, D. and Wingrave, C.: Design and Evaluation of Menu Systems for Immersive Virtual Environments. *Proceedings of IEEE Virtual Reality*, p.149-156, 2001.
37. Billingham, M., Bowskill, J., Dyer, N., and Morphet, J.: An Evaluation of Wearable Information Spaces. *Proceedings of the Virtual Reality Annual international Symposium*, p. 20, March 14 - 18, 1998.
38. Coninx, K., Van Reeth, F. and Flerackers, E.: A hybrid 2D/3D user interface for Immersive Object Modeling. *Proceedings of Computer Graphics International '97*, p.47-55, Hasselt-Diepenbeek, Belgium, 1997.
39. Stanney, K.: Realizing the full potential of virtual reality: human factors issues that could stand in the way. *Proceedings of the Virtual Reality Annual International Symposium*, p.28, March 11-15, 1995.
40. Raffle, H. S., Parkes, A. J., and Ishii, H.: Topobo: a constructive assembly system with kinetic memory. *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, p. 647-654, Vienna, Austria, April 24 - 29, 2004.
41. Wloka, M. M. and Greenfield, E.: The virtual tricorder: a uniform interface for virtual reality. *Proceedings of the 8th Annual ACM Symposium on User interface and Software Technology*, p.39-40, Pittsburgh, Pennsylvania, November 15 - 17, 1995.
42. Ward, K., Galoppo, N., and Lin, M.: Interactive Virtual Hair Salon. *Presence: Teleoperators and Virtual Environments*, p.237-251, 16(3), June, 2007.
43. Allison, D., Wills, B., Hodges, L. F., and Wineman, J.: Gorillas in the Bits. *Proceedings of the 1997 Virtual Reality Annual international Symposium*, p.69, March 01 - 05, 1997.
44. Butterworth, J., Davidson, A., Hench, S., and Olano, M. T.: 3DM: a three dimensional modeler using a head-mounted display. *Proceedings of the 1992 Symposium on interactive 3D Graphics*, p.135-138, Cambridge, Massachusetts, 1992.
45. Bowman, D. A., Hodges, L. F., Allison, D., and Wineman, J.: The Educational Value of an Information-Rich Virtual Environment. *Presence: Teleoperators and Virtual Environments*. p.317-331, 8(3), June, 1999.
46. Feiner, S., MacIntyre, B., Haupt, M. and Solomon, E.: Windows on the world: 2D windows for 3D augmented reality, *Proceedings of the 6th annual ACM symposium on User interface software and technology*, p.145-155, Atlanta, Georgia, December, 1993.
47. Angus, I. and Sowizral, H.: Embedding the 2D Interaction Metaphor in a Real 3D Virtual Environment. *Proceedings of SPIE (Stereoscopic Displays and Virtual Reality Systems)*, p.282-293, Vol. 2409, Bellingham, Washington, 1995.
48. Bowman, D. A., Hodges, L. F., and Bolter, J.: The Virtual Venue: User-Computer Interaction in Information-Rich Virtual Environments. *Presence: Teleoper. Virtual Environments*.p. 478-493, 7(5), October, 1998.
49. Raskar, R., Beardsley, P., van Baar, J., Wang, Y., Dietz, P., Lee, J., Leigh, D., and Willwacher, T.: RFID lamps: interacting with a self-describing world via photosensing wireless tags and projectors. *ACM SIGGRAPH 2004 Papers*, p. 406-415, Los Angeles, California, August 08 - 12, 2004.

50. Borriello, G. and Want, R.: Embedded computation meets the World Wide Web. *Communications of the ACM* 43, p.59-66, no. 5, May, 2000.
51. Hallberg, J. Nilsson, M. Synnes, K.: Positioning with Bluetooth. *10th International Conference on Telecommunications*, p.954-958, vol. 2, February 23 – March 1, 2003.
52. Royer, E. M. and Toh, C. K.: A review of current routing protocols for ad-hoc mobile networks. *IEEE Personal Communications*, p.46-55, 6(2), 1999.
53. Raskar, R., van Baar, J., Beardsley, P., Willwacher, T., Rao, S., and Forlines, C.: iLamps: geometrically aware and self-configuring projectors. *ACM SIGGRAPH 2003 Papers*, p. 809-818, San Diego, California, July 27 - 31, 2003.
54. Raskar R., Bimber, O. and Inami, M.: Spatial Augmented Reality. *ACM SIGGRAPH 2007 courses*, San Diego, California, August 5-9, 2007.
55. Tidwell, M., Johnston, R.S., Melville, D. and Furness, T.A.: The Virtual Retinal Display - A Retinal Scanning Imaging System. *Proceedings of Virtual Reality World '95*, p. 325-333, 1995.
56. Peterka, T., Kooima, R., Girado, J., Ge, J., Sandin, D., and DeFanti, T.. Evolution of the Varrier Autostereoscopic VR Display. *IS&T/SPIE Electronic Imaging 2007 Conference Proceedings*, 2007.
57. Jeong, B., Jagodic, R., Renambot, L., Singh, R., Johnson, A. and Leigh, J.: Scalable Graphics Architecture for High-Resolution Displays. *Proceedings of IEEE Information Visualization Workshop*, Minneapolis, Minnesota, October 23-25, 2005.
58. Chen, J. J. and Adams, C.: Short-range wireless technologies with mobile payments systems. *Proceedings of the 6th international Conference on Electronic Commerce*, p. 649-656, Delft, The Netherlands, October 25 - 27, 2004.
59. Piekarski, W. and Thomas, B. H.: The Tinmith system: demonstrating new techniques for mobile augmented reality modelling. *Proceedings of the Third Australasian Conference on User interfaces - Volume 7*, p. 61-70, Melbourne, Victoria, Australia, 2002.
60. Lee, S., Kim, I., Ahn, S. C., Ko, H., Lim, M., and Kim, H.: Real time 3D avatar for interactive mixed reality. *Proceedings of the 2004 ACM SIGGRAPH international Conference on Virtual Reality Continuum and Its Applications in industry*, p.75-80, Singapore, June 16 - 18, 2004.
61. Debevec, P. E., Taylor, C. J., and Malik, J.: Modeling and rendering architecture from photographs: a hybrid geometry- and image-based approach. *Proceedings of the 23rd Annual Conference on Computer Graphics and interactive Techniques*, p. 11-20, 1996.
62. Kameda, Y., Takemasa, T., Ohta, Y.: Outdoor see-through vision utilizing surveillance cameras. *Third IEEE and ACM International Symposium on Mixed and Augmented Reality*, p.151-160, Arlington, Virginia, November, 2004.
63. Thiebaut, M.: Virtual director: Steering scientific visualization with virtual camera choreography. Masters thesis, *Electronic Visualization Laboratory*, University of Illinois at Chicago, 1997.
64. Bolter, J., Hodges, L. F., Meyer, T., and Nichols, A.: Integrating Perceptual and Symbolic Information in VR. *IEEE Computer Graphics Applications*, p. 8-11, 15(4), July, 1995.
65. Brady, R., Pixton, J., Baxter, G., Moran, R., Potter, C. 5., Canagher, B. & Belmont, A.: Crumbs: A virtual environment tracking tool for biological imaging, *Proceedings of IEEE Biomedical Visualization Symposium*, Atlanta, Georgia, October 30, 1995.

66. Bell, B., Feiner, S., and Höllerer, T.: View management for virtual and augmented reality. *Proceedings of the 14th Annual ACM Symposium on User interface Software and Technology*, p.101-110, Orlando, Florida, November 11 - 14, 2001.
67. Stenicke, F., Ropinski, T., Bruder, G. and Hinrichs, K., Interscopic user interface concepts for fish tank virtual reality systems, *Proceedings of IEEE Virtual Reality*, p. 27-34, Charlotte, North Carolina, March 10-14, 2007.
68. Ware, C. and Lowther, K., Selection using a one-eyed cursor in a fish tank VR environment, *ACM Transactions on Computer-Human Interaction (TOCHI)*, 4(4), p.309-322, December, 1997.
69. Gnanayutham, P., Bloor, C., and Cockton, G.: Discrete acceleration and personalised tiling as brain?body interface paradigms for neurorehabilitation. *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, p. 261-270, Portland, Oregon, USA, April 02 - 07, 2005.
70. Bowman D., Poupyrev I., LaViola J. and Kruijff E.: *3D User Interfaces: Theory and Practice*. Addison-Wesley/Pearson, p.151, Boston, 2005.
71. Wingrave, C. and Bowman, D.: Baseline Factors for Raycasting Selection. *Proceedings of HCI International*, 2005.
72. Poupyrev, I., Weghorst, S., and Fels, S.; Non-isomorphic 3D rotational techniques. *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, p. 540-547, The Hague, The Netherlands, April 01 - 06, 2000.
73. MacKenzie, I. S., Kauppinen, T., and Silfverberg, M.: Accuracy measures for evaluating computer pointing devices. *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, p. 9-16, Seattle, Washington, 2001.
74. Wingrave, C., Tintner, R., Walker, B., Bowman, D. and Hodges, L.: Exploring Individual Differences in Raybased Selection: Strategies and Traits. *IEEE Virtual Reality*, Bonn, Germany, March 12-16, 2005.
75. Bowman, D., Johnson, D., and Hodges, L.: Testbed Evaluation of Virtual Environment Interaction Techniques. *Presence: Teleoperators and Virtual Environments*, 10(1), p.75-95, 2001.
76. Myers, B. A., Bhatnagar, R., Nichols, J., Peck, C. H., Kong, D., Miller, R., and Long, A. C.: Interacting at a distance: measuring the performance of laser pointers and other devices. *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems: Changing Our World, Changing Ourselves*, p.33-40, Minneapolis, Minnesota, April 20 - 25, 2002.
77. Tan, D. S., Gergle, D., Scupelli, P., and Pausch, R.: Physically large displays improve performance on spatial tasks. *ACM Transactions Computer-Human Interaction*, p. 71-99, 13(1), March, 2006.
78. Lee, S., Seo, J., Kim, G. J. and Park C.: Evaluation of pointing techniques for ray casting selection in virtual environments, *Third International Conference on Virtual Reality and Its Application in Industry*, p.38-44, April, 2003.
79. Vogel, D. and Balakrishnan, R.: Distant freehand pointing and clicking on very large, high resolution displays, *Proceedings of the 18th annual ACM symposium on User interface software and technology*, Seattle, Washington, October, 2005.
80. Baudisch, P., Cutrell, E., Hinckley, K., and Gruen, R.: Mouse ether: accelerating the acquisition of targets across multi-monitor displays. *CHI '04 Extended Abstracts on Human Factors in Computing Systems*, p.1379-1382, Vienna, Austria, April 24 - 29, 2004.
81. Nacenta, M. A., Sallam, S., Champoux, B., Subramanian, S., and Gutwin, C.: Perspective cursor: perspective-based interaction for multi-display environments. *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, p.289-298, Montréal, Québec, Canada, April 22 - 27, 2006.

82. Baudisch, P. and Rosenholtz, R.: Halo: a technique for visualizing off-screen objects. *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, p. 481-488, Ft. Lauderdale, Florida, April 05 - 10, 2003.
83. Volda, S., Podlaseck, M., Kjeldsen, R., and Pinhanez, C.: A study on the manipulation of 2D objects in a projector/camera-based augmented reality environment. *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, p. 611-620, Portland, Oregon, April 02 - 07, 2005.
84. Lee, G. A., Billinghamurst, M., and Kim, G. J.: Occlusion based interaction methods for tangible augmented reality environments. *Proceedings of the 2004 ACM SIGGRAPH international Conference on Virtual Reality Continuum and Its Applications in industry*, p. 419-426, Singapore, June 16 - 18, 2004.
85. Larimer, D. and Bowman, D. VEWL: A Framework for Building a Windowing Interface in a Virtual Environment. *Proceedings of INTERACT: IFIP TC13 International Conference on Human-Computer Interaction*, p.809-812, 2003.
86. Stoev, S., Schmalstieg, D. and Wolfgang Straßer, W.: Two-Handed Through-the-Lens-Techniques for Navigation in Virtual Environments, *Proceedings of the Eurographics Workshop on Virtual Environments*, 2001.
87. Stoev, S. L. and Schmalstieg, D.: Application and taxonomy of through-the-lens techniques. *Proceedings of the ACM Symposium on Virtual Reality Software and Technology*, p.57-64, Hong Kong, China, November 11 - 13, 2002.
88. Phillips, C. B. and Badler, N. I.: JACK: a toolkit for manipulating articulated figures. *Proceedings of the 1st Annual ACM SIGGRAPH Symposium on User interface Software*, p. 221-229, Alberta, Canada, October 17 - 19, 1988.
89. Zeleznik, R. C., Forsberg, A. S. and Schulze, J. P.: Look-That-There: Exploiting Gaze in Virtual Reality Interactions, *Technical Report CS-05-04*, Brown University, Department of Computer Science, March, 2005.
90. Cornsweet, T. N., *Visual Perception*, Academic Press, 1970.
91. Holm, R., Stauder, E., Wagner, R., Priglinger, M. and Volkert, J.: A Combined Immersive and Desktop Authoring Tool for Virtual Environments, *Proceedings of the IEEE Virtual Reality Conference*, p.93, March, 2002.
92. Snowdon, D. and Jää-Aro, K.: A subjective Virtual Environment for collaborative information visualization, *Virtual Reality Universe '97*, California, USA, April, 1997.
93. Park, K. S., Kapoor, A., and Leigh, J.: Lessons learned from employing multiple perspectives in a collaborative virtual environment for visualizing scientific data. *Proceedings of the Third international Conference on Collaborative Virtual Environments*, p. 73-82, San Francisco, California, 2000.
94. Simon, R., Wegscheider, F., and Tolar, K.: Tool-supported single authoring for device independence and multimodality. *Proceedings of the 7th international Conference on Human Computer interaction with Mobile Devices & Services*, p. 91-98, Salzburg, Austria, September 19 - 22, 2005.
95. Rotard, M., Knödler, S., and Ertl, T.: A tactile web browser for the visually disabled. *Proceedings of the Sixteenth ACM Conference on Hypertext and Hypermedia*, p. 15-22, Salzburg, Austria, September 06 - 09, 2005.
96. Mulder, J. D.: Menu selection in desktop virtual reality. *Central European Multimedia and Virtual Reality Conference*, Prague, Czech Republic, June 08-10, 2005.
97. MacKenzie, I. S. and Jusoh, S.: An Evaluation of Two Input Devices for Remote Pointing. *Proceedings of the 8th IFIP international Conference on Engineering For Human-Computer interaction*, p. 235-250, May 11 - 13, 2001.

98. Forsberg, A., Herndon, K., and Zeleznik, R. 1996. Aperture based selection for immersive virtual environments. *Proceedings of the 9th Annual ACM Symposium on User interface Software and Technology*, p. 95-96, Seattle, Washington, November 06 - 08, 1996.
99. Whisenand, T. G. and Emurian, H. H.: Some effects of angle of approach on icon selection, *Conference companion on Human factors in computing systems*, p.298-299, Denver, Colorado, May 07-11, 1995.
100. Balakrishnan, R. and MacKenzie, I. S.: Performance differences in the fingers, wrist, and forearm in computer input control. *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, p. 303-310, Atlanta, Georgia, March 22 - 27, 1997.
101. Livingston, M. A. and State, A.: Magnetic Tracker Calibration for Improved Augmented Reality Registration, *PRESENCE: Teleoperators and Virtual Environments*, 6(5), October, 1997.
102. Leigh, J., Johnson, A. E., Vasilakis, C. A., and DeFanti, T. A.: Multi-perspective collaborative design in persistent networked virtual environments. *Proceedings of the Virtual Reality Annual international Symposium*, p.253, March 30 - April 03, 1996.
103. Stenius, M. Collaborative object modeling in virtual environments, Masters Thesis, KTH, School of Computer Science and Engineering, Royal Institute of Technology, Stockholm, Sweden, 1996.
104. Liang, J. and Green, M.: JDCAD: A Highly Interactive 3D Modeling System, *3rd International Conference on CAD and Computer Graphics*, p.217-222, Beijing, China, August, 1993.
105. Pape, D., Anstey, J., Dolinsky, M., and Dambik, E. J.: Ygdrasil: a framework for composing shared virtual worlds. *Future Generation Computing Systems*, p.1041-1049, 19(6), August, 2003.
106. Watsen, K. and Zyda, M.: Bamboo - A Portable System for Dynamically Extensible, Real-Time, Networked, Virtual Environments. *Proceedings of the Virtual Reality Annual international Symposium*, p.252, March 14 - 18, 1998.
107. Tramberend, H.: Avocado: A Distributed Virtual Reality Framework. *Proceedings of the IEEE Virtual Reality*, p.14, March 13 - 17, 1999.
108. Hawkes, R. and Wray, M.: LivingSpace" A Living Worlds Implementation using an Even-based Architecture. *HP Labs*, Technical Report HPL-98-181, 1998.
109. Pape, D. and Sandin, D.: Alive on the Grid. *Proceedings of 6th World Multiconference on Systemics, Cybernetics and Informatics*, p.485-490, Vol. XII, 2002.
110. Bues, M., Blach, R., and Haselberger, F.: Sensing surfaces: bringing the desktop into virtual environments. *Proceedings of the Workshop on Virtual Environments*, p. 303-304, Zurich, Switzerland, May 22 - 23, 2003.
111. Goldberg, A. and Robson, D.: *Smalltalk-80: the Language and its Implementation*. Addison-Wesley Longman, Reading, Massachusetts, 1983.
112. Fischnaller, F., Hill, A.: CITYCLUSTER "From the Renaissance to the Megabyte Networking Age": A Virtual Reality and High-Speed Networking Project, *Presence: Teleoperators and Virtual Environments*, p.1-19, February, 2005.

APPENDIX

The research reported in this document was conducted with the approval of the Office for the Protection of Research Subjects. Two applications for exemption were approved by the Institutional Review Board are described below.

Protocol Number: 2006-0079

Claim of exemption as defined in U.S. Department of Health and Human Services Regulations for the Protection of Human Subjects [(45 CFR 46.101(b))]

Title: A Comparative Study of Image-Plane and Virtual-Hand 3D Interaction Techniques

Principle Investigator: Alex S. Hill

Granted: February 13, 2006

Category:

(2) Research involving the use of educational tests (cognitive, diagnostic, aptitude, achievement), survey procedures, interview procedures or observation of public behavior, unless: (i) information obtained is recorded in such a manner that human subjects can be identified, directly or through identifiers linked to the subjects; and (ii) any disclosure of the human subjects' responses outside the research could reasonably place the subjects at risk of criminal or civil liability or be damaging to the subjects' financial standing, employability, or reputation.

Protocol Number: 2006-0012

Claim of exemption as defined in U.S. Department of Health and Human Services Regulations for the Protection of Human Subjects [(45 CFR 46.101(b))]

Title: Developing an Effective User Interface for Both Immersive and Desktop Visualization Environments

Principle Investigator: Alex S. Hill

Granted: January 27, 2006

Category:

(2) Research involving the use of educational tests (cognitive, diagnostic, aptitude, achievement), survey procedures, interview procedures or observation of public behavior, unless: (i) information obtained is recorded in such a manner that human subjects can be identified, directly or through identifiers linked to the subjects; and (ii) any disclosure of the human subjects' responses outside the research could reasonably place the subjects at risk of criminal or civil liability or be damaging to the subjects' financial standing, employability, or reputation.

VITA

NAME: Alex S. Hill

EDUCATION: B.S., Mathematics, Trinity University, San Antonio, Texas, 1988
M.S., Mechanical Engineering, University of Texas at Austin, Austin, Texas, 1992
Ph. D., Computer Science, University of Illinois at Chicago, Chicago, Illinois, 2007

TEACHING: School of Art and Design, University of Illinois at Chicago, Advanced Computer Art-Design, Associate Adjunct Professor, 2001-2006

HONORS: Graduate Fellows in K-12 Education, 2001
Phi Kappa Phi Honor Society, 1992
Tau Beta Phi Engineering Honor Society, 1992

PROFESIONAL MEMBERSHIP: Association for Computing Machinery
Institute of Electrical and Electronics Engineers

PUBLICATIONS: Franz Fischnaller, Alex Hill: CITYCLUSTER "From the Renaissance to the Megabyte Networking Age": A Virtual Reality and High-Speed Networking Project, *Presence: Teleoperators & Virtual Environments*, February 1, 2005, Vol. 14, No. 1, Pages 1-19

Alex Hill, Daria Tsoupikova, Development of Rutopia 2 VR Artwork Using New Ygdrasil Features, *Proceedings of the 3rd International Conference on Computer Graphics Theory and Applications*, Springer, March 2007, Barcelona, Spain