

Exploiting Proprioception in Virtual-Environment Interaction

by

Mark Raymond Mine

A dissertation submitted to the faculty of the University of North Carolina at Chapel Hill in partial fulfillment of the requirements for the degree of Doctor of Philosophy in the Department of Computer Science.

Chapel Hill

1997

Approved by:

Dr. Frederick P. Brooks Jr., Advisor

Dr. Henry Fuchs

Dr. Gary Bishop, Reader

Dr. Anselmo Lastra

Dr. John Tector, Reader.

Dr. Carlo H. Sequin

© 1997
Mark Raymond Mine
ALL RIGHTS RESERVED

ABSTRACT

Mark Raymond Mine
Exploiting Proprioception in Virtual-Environment Interaction
(Under the direction of Frederick P. Brooks, Jr.)

Manipulation in immersive virtual environments is difficult partly because users must do without the haptic contact with real objects they rely on in the real world to orient themselves and the objects they are manipulating. To compensate for this lack, I propose exploiting the one real object every user has in a virtual environment, his body. I present a unified framework for virtual-environment interaction based on *proprioception*, a person's sense of the position and orientation of his body and limbs. I describe three forms of body-relative interaction:

- Direct manipulation—ways to use body sense to help control manipulation
- Physical mnemonics—ways to store/recall information relative to the body
- Gestural actions—ways to use body-relative actions to issue commands

Automatic scaling is a way to bring objects instantly within reach so that users can manipulate them using proprioceptive cues. Several novel virtual interaction techniques based upon automatic scaling and our proposed framework of proprioception allow a user to interact with a virtual world intuitively, efficiently, precisely, and lazily.

Two formal user studies evaluate key aspects of body-relative interaction. The virtual docking study compares the manipulation of objects co-located with one's hand and the manipulation of objects at a distance. The widget interaction experiment explores the differences between interacting with a widget held in one's hand and interacting with a widget floating in space.

Lessons learned from the integration of body-relative techniques into a real-world system, the Chapel Hill Immersive Modeling Program (CHIMP), are presented and discussed.

ACKNOWLEDGMENTS

Thanks to

Frederick P. Brooks Jr., Gary Bishop, Henry Fuchs, Anselmo Lastra, John Tector, and Carlo H. Sequin for serving on my doctoral dissertation committee;

Frederick P. Brooks Jr., my advisor, for his insights, inspiration, and for making it all so clear;

Gary Bishop for many fruitful years of collaboration and for not minding too much that my dissertation didn't have wires and accelerometers coming out of it;

Henry Fuchs for the inspiration of his boundless energy, enthusiasm, and love of knowledge;

Anselmo Lastra for his kindness, advice, and for keeping Pixel-Planes 5 alive long enough for me to graduate;

Carlo Sequin for asking the hard questions, and for helping me to keep it simple;

John Tector for the many wonderful conversations about architecture and design;

Warren Robinett for leading me to the University of North Carolina;

Rick Zobel for paving the way for my investigations into immersive design;

Robert Zeleznik for his invaluable contributions to this work;

Linda Houseman, Dave Harrison, Todd Gaul, and Peggy Wetzel for all of their help during the years;

Hans Weber and Greg Welch for being such good friends and for the meetings of the IHBI at the TOT¹;

Erik Erikson for Vinimini, G2, Speed Racer, and for keeping it fun;

Eliza Graves for the laughter and the smiles;

My parents for all they have done for me through the years;

Dylan for the incredible joy he has brought to my life;

Baby X for the many wonderful years to come;

and most importantly,

Sandra for her unwavering love, support, faith, and devotion, and for, more than anyone else, making it all possible.

Financial support for this work came from the following agencies: Defense Advanced Research Projects Agency, Link Foundation, Lockheed Missiles and Space, Inc. (indirect DARPA)

¹Institute for Half-Baked Ideas at the Top of the Hill

TABLE OF CONTENTS

	Page
LIST OF TABLES	x
LIST OF FIGURES	xi
LIST OF ABBREVIATIONS	xiii
 Chapter	
I. Introduction	1
1.1 The Research	1
1.2 The Challenge	1
1.3 The Attack	2
1.4 A Complication	3
1.5 A Proposal	5
1.6 Overview	6
II. Related Work	8
2.1 3-DoF and 6-DoF Object Manipulation Using 2D Input	8
2.2 Object Manipulation Using Higher-Dimensional Input	11
2.3 Two-handed Interaction	14
2.3.1 Example Techniques	14
2.3.2 Theoretical and Experimental Results	19
2.4 Manipulating Objects Using Gesture and Voice	22
2.5 Systems for Interactive Design	24
2.5.1 Working Through-the-window	25
2.5.2 Working Immersed	30
III. Body-Relative Interaction Techniques	33
3.1 Working Within Arm's Reach	33
3.2 Sample Interaction Techniques	36
3.2.1 Direct Manipulation	36
3.2.1.1 Scaled-World Grab for Manipulation	36
3.2.1.2 Scaled-World Grab for Locomotion	38
3.2.2 Physical Mnemonics	38
3.2.2.1 Pull-Down Menus	38
3.2.2.2 Hand-Held Widgets	39
3.2.2.3 FOV-Relative Mode Switching	41
3.3.3 Gestural Actions	41
3.3.3.1 Head-Butt Zoom	41

	3.3.3.2	Look-at Menus	43
	3.3.3.3	Two-Handed Flying	43
	3.3.3.4	Over-the-Shoulder Deletion	44
IV.		User Study 1—Virtual Object Docking	46
	4.1	Introduction	46
	4.2	Hypotheses	47
	4.3	The Experiment	47
	4.3.1	Subjects	47
	4.3.2	Experimental Platform	47
	4.3.3	The Task	48
	4.3.4	Experimental Conditions	49
	4.3.5	Experimental Procedure	50
	4.4	Results	51
	4.5	Questionnaire Results	54
	4.6	Discussion	56
	4.7	Conclusion	56
V.		User Study 2—Proprioception and Virtual Widget Interaction	58
	5.1	Introduction	58
	5.2	Hypotheses	59
	5.3	The Experiment	59
	5.3.1	Subjects	59
	5.3.2	Experimental Platform	60
	5.3.3	The Task	61
	5.3.4	Experimental Procedure	62
	5.4	Results	63
	5.5	Questionnaire Results	64
	5.6	Discussion	65
	5.7	Conclusions	66
VI.		CHIMP—The Chapel Hill Immersive Modeling Program	68
	6.1	CHIMP Overview	68
	6.2	Managing Modes	73
	6.2.1	Rotary Tool Chooser	74
	6.2.2	Two-Dimensional Control Panels	75
	6.2.3	Look-at Menus	77
	6.2.4	Pull-Down Menus	77
	6.3	Object Selection	78

6.4	Object Manipulation	81
6.5	Object Generation	82
6.6	Constrained Object Manipulation	83
6.6.1	Co-Located Widgets	83
6.6.2	Hand-Held Widgets	85
6.7	Numeric Input in a Virtual World	86
VII.	Final Words	90
7.1	Conclusions	90
7.2	Contributions	91
7.3	Future Work	93
	Localized Haptic Feedback	93
A.	A Review of the State-of-the-Art of Computer-Aided Modeling	95
A.1	Introduction	95
A.2	Modeling Techniques and Paradigms	96
A.2.1	Input for a Three-Dimensional Task	96
A.2.1.1	Numeric Input	96
A.2.1.2	Relative Input	96
A.2.1.3	2D Interactive Input	97
A.2.2	Output of a Three-Dimensional Space	98
A.2.2.1	Format of the Modeling View	99
A.2.2.2	Three-Dimensional Visualization: Separate or Integrated	100
A.2.2.3	Three-Dimensional Visualization: Static or Interactive	101
A.2.2.4	Complications of Two-Dimensional Output	101
A.3	Modeling System Capability Comparison	102
A.4	Modeler Reviews Overview	105
A.5	Archicad	107
A.5.1	Overview	107
A.5.2	Model Creation	108
A.5.3	Model Modification	108
A.5.4	Model Interaction/Visualization	109
A.5.5	Manual	109
A.5.6	Comments/Impressions	110
A.6	AutoCAD	111
A.6.1	Overview	111
A.6.2	Model Creation	112

	A.6.3	Model Modification	113
	A.6.4	Model Interaction/Visualization	114
	A.6.5	Manual	114
	A.6.6	Comments/Impressions	114
A.7		DesignWorkshop	115
	A.7.1	Overview	115
	A.7.2	Model Creation	116
	A.7.3	Model Modification	117
	A.7.4	Model Interaction/Visualization	117
	A.7.5	Manual	118
	A.7.6	Comments/Impressions	118
A.8		Designer's Workbench	119
	A.8.1	Overview	119
	A.8.2	Model Creation	120
	A.8.3	Model Modification	121
	A.8.4	Model Interaction/Visualization	122
	A.8.5	Manual	122
	A.8.6	Comments/Impressions	122
A.9		Form-Z	123
	A.9.1	Overview	123
	A.9.2	Model Creation	124
	A.9.3	Model Modification	125
	A.9.4	Model Interaction/Visualization	125
	A.9.5	Manual	126
	A.9.6	Comments/Impressions	126
A.10		IGRIP	128
	A.10.1	Overview	128
	A.10.2	Model Creation	130
	A.10.3	Model Modification	130
	A.10.4	Model Interaction/Visualization	131
	A.10.5	Manual	131
	A.10.6	Comments/Impressions	132
A.11		Minicad+4	133
	A.11.1	Overview	133
	A.11.2	Model Creation	134
	A.11.3	Model Modification	135

A.11.4	Model Interaction/Visualization	136
A.11.5	Manual	136
A.11.6	Comments/Impressions	136
A.12	MultiGen	138
A.12.1	Overview	138
A.12.2	Model Creation	139
A.12.3	Model Modification	140
A.12.4	Model Interaction/Visualization	140
A.12.5	Manual	141
A.12.6	Comments/Impressions	141
A.13	Sculpt 3D	143
A.13.1	Overview	143
A.13.2	Model Creation	144
A.13.3	Model Modification	145
A.13.4	Model Interaction/Visualization	145
A.13.5	Manual	146
A.13.6	Comments/Impressions	146
A.14	Upfront	148
A.14.1	Overview	148
A.14.2	Model Creation	149
A.14.3	Model Modification	150
A.14.4	Model Interaction/Visualization	150
A.14.5	Manual	151
A.14.6	Comments/Impressions	151
A.15	WalkThrough	153
A.15.1	Overview	153
A.15.2	Model Creation	154
A.15.3	Model Modification	154
A.15.4	Model Interaction/Visualization	155
A.15.5	Manual	155
A.15.6	Comments/Impressions	156
B.	References	157

LIST OF TABLES

Table 1.1:	Successful virtual-world application domains.	3
Table 2.1:	Interactive design systems input/output comparison.	25
Table 4.1:	Mean time of trial completion by experimental condition.	52
Table 4.2:	Mean questionnaire results by technique.	54
Table 4.3:	F statistic and significance by questionnaire category.	54
Table 4.4:	Co-located vs. fixed-offset, F statistic and significance by questionnaire category.	55
Table 4.5:	Co-located vs. variable-offset, F statistic and significance by questionnaire category.	55
Table 4.6:	Fixed-offset vs. variable-offset, F statistic and significance by questionnaire category.	55
Table 5.1:	Mean positional accuracy by experimental condition.	63
Table 5.2:	Mean questionnaire results by technique.	65
Table 5.3:	F statistic and significance by questionnaire category.	65
Table 6.1:	CHIMP system overview.	70
Table 6.2:	CHIMP's hand-held widgets.	71
Table 6.3:	CHIMP's control panels.	72
Table A.1:	Modeling packages reviewed.	95
Table A.2:	Modeling system capability comparison.	103
Table A.3:	Modeling system paradigms.	106

LIST OF FIGURES

Figure 2.1:	Nielson's triad cursor.	8
Figure 2.2:	Constrained geometric transformation using widgets.	10
Figure 2.3:	Using object associations.	11
Figure 2.4:	The Rockin' Mouse.	12
Figure 2.5:	Zhai et al.'s framework for the study of multi-degree-of-freedom manipulation schemes.	13
Figure 2.6:	Layers in a toolglass system.	14
Figure 2.7:	Using toolglasses, two-hands, and transparency in T3.	15
Figure 2.8:	Marking Menus interaction.	16
Figure 2.9:	Using two hands and props in Netra.	17
Figure 2.10:	Object manipulation and spline editing using Fitzmaurice et al's graspable user interface.	18
Figure 2.11:	The Responsive Workbench.	19
Figure 2.12:	Guiard's handwriting experiment.	20
Figure 2.13:	Buxton and Meyer's two handed input experiment.	21
Figure 2.14:	Kabbash et al's two-hand connect the dots experiment.	22
Figure 2.15:	VIDEODESK two-handed interaction.	23
Figure 2.16:	Using a gesture to move a group in GEdit.	23
Figure 2.17:	Schmandt's stereoscopic display.	26
Figure 2.18:	University of Alberta's JDCAD system.	27
Figure 2.19:	Using T junctions to infer object placement in SKETCH.	29
Figure 2.20:	UNC's nanoWorkbench.	30
Figure 2.21:	University of Virginia's World-In-Miniature.	31
Figure 3.1:	Automatic scaling of the world when the user grabs and releases an object.	34
Figure 3.2:	Vectors used in determining automatic scaling factor.	35
Figure 3.3:	Using a pull-down menu.	39
Figure 3.4:	Using a hand-held widget.	40
Figure 3.5:	Selecting a region for closer inspection.	42
Figure 3.6:	Look-at menu.	43
Figure 3.7:	Two-handed flying.	44
Figure 3.8:	Over-the-shoulder deletion.	45
Figure 4.1:	Experimental conditions for the docking test.	49
Figure 4.3:	Mean docking times by technique.	53

Figure 5.1 :	Widget test objects.	60
Figure 5.2 :	Widget test experimental conditions	62
Figure 5.3 :	Mean positional accuracies by technique.....	64
Figure 6.1 :	Using the CHIMP system.....	69
Figure 6.2 :	CHIMP's primary and secondary input devices.	69
Figure 6.3 :	Rotary tool chooser.....	74
Figure 6.4 :	Interacting with a control panel using a laser beam.	76
Figure 6.5 :	Interacting with a control panel using occlusion selection.....	76
Figure 6.6 :	CHIMP's look-at menus.....	77
Figure 6.7 :	Occlusion selection, first person point of view.	79
Figure 6.8 :	Occlusion selection, third person point of view.	79
Figure 6.9 :	Spotlight selection, third person point of view.	80
Figure 6.10 :	Spotlight selection, first person point of view.	80
Figure 6.11 :	First and second generation constrained manipulation widgets.	84
Figure 6.13 :	Constrained manipulation mode selection based upon hand separation.	86
Figure 6.14 :	Numeric input using the Arithma Addiator.	87
Figure 6.15 :	Linear interactive numbers.	87
Figure 6.16 :	Rotary interactive numbers.	88
Figure A.1 :	Orthogonal-view system.	100
Figure A.2a :	Perspective projection ambiguity.	101
Figure A.2b :	Three orthogonal views of the object in Figure A.2a.	101
Figure A.3 :	Archicad interface.	107
Figure A.4 :	AutoCAD interface.	111
Figure A.5 :	DesignWorkshop interface.	115
Figure A.6 :	Designer's Workbench interface.....	119
Figure A.7 :	Form-Z interface.	123
Figure A.8 :	IGRIP interface.....	128
Figure A.9 :	Minicad+ interface.	133
Figure A.10 :	2D vs. 3D objects.	134
Figure A.11 :	MultiGen interface.	138
Figure A.12 :	Sculpt 3D interface.	143
Figure A.13 :	Upfront interface.....	148
Figure A.14 :	WalkThrough interface.	153

LIST OF ABBREVIATIONS

1D	one-dimensional
2D	two-dimensional
3D	three-dimensional
ANOVA	analysis of variances
CAD	computer-aided design
CHIMP	Chapel Hill Immersive Modeling Program
DoF	degree of freedom
GUI	graphical user interface
HMD	head-mounted display
K	kilo
MANOVA	multivariate analysis of variances
UI	user interface
UNC	University of North Carolina
VE	virtual environment
VR	virtual reality
WIM	world in miniature

Chapter 1

Introduction

1.1 The Research

The goal of my research is a better understanding of what it means to work in a virtual world. The focus is the characterization of the benefits and limitations of this new medium. The hope is that improved understanding will lead to more effective virtual-environment interaction techniques; those that minimize user energy and make it possible to perform real-world work in a virtual world.

To motivate my research I have chosen the driving problem of three-dimensional (3D) modeling for architectural design, for several reasons. First, to evaluate the benefits and limitations of working in a virtual world fairly, it is important to concentrate on real-world tasks and not toy problems; architectural models are complex models that are difficult to build. Second, if we are to realize any benefits from working in a virtual world, it is important to focus on tasks that will truly profit from being in an immersive environment; the shapes and spaces inside architectural models, more so than mechanical models, are just as important as their external form.

1.2 The Challenge

The architectural design of three-dimensional spaces is inherently a difficult task. Even given the ultimate design system, in which thoughts magically become material, an architect would still encounter many difficulties in solving a typical design problem with its myriad of trade-offs and constraints.

In the real world, these inherent difficulties are compounded by *incidental difficulties*, problems which are the result of the chosen medium of expression and not inherent in the design problem itself. The duration and flexibility of the design cycle is highly sensitive to the amount of time required to represent and modify designs in the

chosen medium. This is clearly true of sketches, of formal drawings, and of scale model—all media used for the expression of architectural designs.

The choice of the computer as a design medium has greatly simplified and sped up many aspects of the architectural design process. Just ease of copying and erasing is one big plus. Many of the gains, however, are restricted to the transformation of existing design data or aspects (structural, mechanical, electrical) such as redrawing a single design from many views, or managing large databases of materials and parts. The specification of original data is still a time-consuming and difficult task (a half a man year for a 30K-element model [Brooks, 1994]).

It is my belief that many of the shortcomings of the computer as medium for the design of three-dimensional spaces are the result of the limitations of existing two-dimensional (2D) interfaces. Two-dimensional displays inherently inject ambiguity into the interpretation of displayed information [Gregory, 1973] . The use of two-dimensional input devices, such as the mouse or the data tablet, precludes the direct specification of three-dimensional positions, orientations and extents. Designers are forced to look and work through small windows onto their virtual world. They tend to limit themselves to views along the principal axes plus a few other classical view directions. Indeed, despite many years of research and development, few computer-aided design programs approach the flexibility of the cocktail napkin or the architect's "trash"² as a design tool.

1.3 The Attack

The underlying thesis motivating this research is that architects can design three-dimensional spaces more easily in an immersive environment than they can modeling through-the-window using conventional workstation inputs and displays. I believe this to be true for several reasons.

In an immersive environment one can directly perceive and manipulate three-dimensional objects instead of interacting with abstract interface elements. Users can harness interaction skills learned in the real world. This helps to make the computer interface really transparent and allows users to work more directly with the objects of design.

²Tracing paper used by architects which can be placed on top of existing drawings to try out new design ideas quickly without having to redraw the entire design.

In a virtual world one turns his head to look at something. Contrast this with the frustration of setting one's viewpoint in a 3D through-the-window application. Dynamic viewpoint change, whether immersive or through the window, gives better space perception.

Finally, using a head-mounted display, one becomes immersed within the virtual space within which one intuitively changes viewpoint. Not only does this make it easier to understand the shapes and spaces being created, it means that controls and information can now be distributed about the user instead of being packed into a small window.

1.4 A Complication

Working in a virtual world is not without its own set of incidental difficulties. Indeed, though promising results have been demonstrated in several key application domains (Table 1), the number of successful virtual environment applications still remains small, with even fewer applications having gone beyond the research laboratory. Why?

Table 1.1: Successful virtual-world application domains.

Domain	Example Applications
"Being There", experience for the sake of experience	Phobia treatment: [Rothbaum, et al., 1995] Aesthetics: [Davies and Harrison, 1996] Entertainment: [Pausch, et al., 1996]
Training and practice of different skills	Surgery: [Hunter, et al., 1993] Military : [Macedonia, et al., 1994] Maintenance: x[Wilson, et al., 1995] Wayfinding: x[Witmer, et al., 1995]x
Visualization of unrealized or unseeable objects	Architecture: [Brooks, 1986] Fluid Flow: [Bryson and Levit, 1992] Nano-surfaces: [Taylor, et al., 1993]
Design	3D models: [Butterworth, et al., 1992] Cityscapes: [Mapes and Moshell, 1995]

Besides the well known technological limitations such as system latency and display resolution, several less obvious factors complicate the task of virtual object manipulation and hamper the development of real-world virtual environment applications.

Many of these successes fall within the realm of spatial visualization. The applications exploit the intuitive view specification (via head tracking) offered by VR systems but make little use of direct virtual-object manipulation. Why is it difficult to do much more than look around in a virtual world?

1) *The precise manipulation of virtual objects is hard.* Although immersion, head-tracked view specification, and six DoF hand tracking facilitate the coarse manipulation of virtual objects, the precise manipulation of virtual objects is complicated by:

- *Lack of haptic feedback:* Humans depend on haptic feedback and physical constraints for precise interaction in the real world; the lack of physical work-surfaces to align against and rest on limits precision and exacerbates fatigue. Though there is considerable ongoing research in the area of active haptic feedback [Durlach and Mavor, 1995], general-purpose haptic feedback devices that do not restrict the mobility of the user are not yet practical or available.
- *Limited input information:* Most virtual-environment systems accept position and orientation (pose) data on the user's head and (if lucky) two hands. One also typically has a button or glove to provide signal/event information. This suffices for specifying simple 6-DoF motion and placement. In the real world, we do this and much more:
 - a) Object modification, usually with tools.
 - b) Directing the cooperation of helping hands, by spoken commands ("Put that there").
 - c) Measuring.
 - d) Annotating objects with text.

In contrast, today in most VR systems:

- a) Tool selection is difficult.
 - b) Voice command technology is marginally effective.
 - c) Measuring tools are rarely available.
 - d) Alphanumeric input is difficult.
- *Limited precision:* The lack of haptic and acoustic feedback, inaccurate tracking systems, and the whole-hand input typical of current VR systems restrict users to the coarse manipulation of virtual objects. Fine-grained manipulations are extremely difficult using this "boxing-glove" style interface. Shumin Zhai of the University of Toronto, for example, has demonstrated that users' task completion times were slower in a 3D docking task when using a 3D input device which excluded the use of the fingers (vs. a similar device that utilized the fingers) [Zhai, et al., 1996].

2) *Virtual environments lack a unifying framework for interaction*, such as the desktop metaphor used in conventional through-the-window computer applications.

Without haptics neither real-world nor desktop computer interaction metaphors are adequate in a virtual environment. Knowledge on how to manipulate objects or controls can no longer be "stored in the world" [Norman, 1988] , with the physical constraints of the devices giving the user clues as to their use (e.g. a dial can only be rotated about its axis).

The desktop metaphor further breaks down when the user is inside the user interface. Interface controls and displays must move with the user as he moves through the environment and be made easy to locate and reach. The differences between working in a conventional computer environment and working immersed are analogous to the differences between a craftsman at a workbench and one moving about a worksite wearing a toolbelt. His toolbelt had better be large and filled with powerful tools.

1.5 A Proposal

Thesis statement:

By providing a real-world frame of reference in which to operate and a more direct and precise sense of control, proprioception helps to compensate for the lack of haptic feedback in virtual-environment interaction.

Without touch, a user can no longer feel his surroundings to tell where he is nor use the felt collision of a manipulandum (an object being manipulated) with stationary objects to refine spatial perception. It is imperative, therefore, to take advantage of one thing every user can still feel in the virtual world, his body.

A person's sense of the position and orientation of his body and its several parts is called *proprioception* [Boff, et al., 1986] . I propose that proprioception can be used to develop a unified set of interaction techniques that allow a user to interact with a virtual world intuitively, efficiently, precisely, and lazily.

In a series of user observations, I have found that body-relative interaction techniques (exploiting proprioceptive feedback) are more effective than techniques relying solely on visual information. Such body-relative interaction techniques provide:

- a physical real-world frame of reference in which to operate
- a more direct and precise sense of control
- "eyes off" interaction (the user doesn't have to constantly watch what he's doing)

A user can take advantage of proprioception during body-relative interaction in at least three ways:

- *Direct manipulation*: If a virtual object is located directly at the user's hand position, the user has a good sense of the position of the object (even with eyes closed) due to proprioception, and thus a greater sense of control. It is easier to place an object precisely by hand than when it is attached to the end of a fishing rod. Manipulation schemes that provide conflicting stimulus/response cues or use non-linear mappings between hand motion and object motion are more difficult for a user to understand and control [Britton, et al., 1978] .
- *Physical mnemonics*: Since a user can no longer feel the world around him, it can be difficult to find, select, and use virtual controls in world space, especially if the user is free to walk about the environment. Users can store virtual objects, in particular menus and widgets [Conner, et al., 1992] , relative to his body. If controls are fixed relative to the user's body, he can use proprioception to find the controls, as one finds his pen in his pocket, or his pliers in his tool belt. If controls are attached to the user's body, they move with him as he moves through the environment and are always within reach. Finally, controls can be stored out of view (behind the user's back for example), reducing visual clutter, yet remaining easily accessible (like an arrow from a quiver).
- *Gestural actions*: Just as a user's body sense can be used to facilitate the recall of objects, it can be used to facilitate the recall of actions, such as gestures used to invoke commands or to communicate information.

1.6 Overview

This dissertation presents the results of my investigations of proprioception and body-relative interaction as a framework for virtual environment interaction. It is organized as follows:

Chapter two: *Related Work* presents relevant work in the areas of interactive design, object manipulation, two-handed interaction and haptic feedback.

Chapter three: *Body-relative Interaction* describes several novel body-relative interaction techniques based on the framework of proprioception introduced in section 1.5. Automatic scaling is presented as a means of instantly bringing objects in reach so that users can manipulate them using proprioceptive cues.

Chapter four: *Virtual Object Docking* presents the results of a user study investigating the benefits of direct manipulation by comparing the manipulation of a virtual object attached to the user's hand versus one at a fixed offset.

Chapter five: *Proprioception and Virtual Widget Interaction* presents the results of a user study exploring the benefits of body-relative interaction by comparing interaction with widgets floating in space with those attached to the user's hand.

Chapter six: *The Chapel Hill Immersive Modeling Program* demonstrates and analyzes the effectiveness of the integration of body-relative interaction techniques in a real-world system.

Chapter seven: *Final Words* presents some final thoughts and discussion of future work such as the use of localized haptic feedback and fingertip control for greater precision and control in situations where proprioception information alone will not suffice.

Appendix A: *A Review of the State of the Art of Computer-Aided Modeling* presents the results of a review of the interaction techniques used and functions included in several commercial through-the-window computer-aided modeling packages. I performed this review at the start of my dissertation research to obtain a better understanding of the current state-of-the-art of computer-aided modeling.

Chapter 6

CHIMP The Chapel Hill Immersive Modeling Program

This chapter presents a description of CHIMP, the Chapel Hill Immersive Modeling Program. CHIMP is an immersive system intended for the preliminary stages of architectural design. The main goal during the development of the CHIMP system has been the creation of an integrated set of virtual-environment interaction techniques that exploit the benefits of working immersed while compensating for its limitations. I present an overview of the CHIMP system and discuss several relevant aspects of the CHIMP design.

6.1 CHIMP Overview

In the CHIMP system users create and manipulate models while immersed within a virtual world, as opposed to the customary through-the window computer-aided design (CAD) environment. CHIMP includes both one and two-handed interaction techniques, minimizes unnecessary user interaction, and takes advantage of the head-tracking and immersion provided by the virtual environment system.

Figure 6.1 shows the author using the CHIMP system. The hardware configuration includes an immersive head-mounted display, a magnetic tracking system with three 6 degree-of-freedom (DoF) sensors (one for the head and one for each hand), two separate input devices (one for each hand) and a high speed graphics computer (Pixel-Planes 5, see [Fuchs, et al., 1989]).

Figure 6.2 shows the two input devices in detail. The user holds the primary input device in his dominant hand and the secondary input device in his non-dominant hand. The primary input device is a modified handle of a commercial joystick with five buttons (only

three are used in the CHIMP system). The secondary input device is a custom built device with two buttons.



Figure 6.1: Using the CHIMP system.



Figure 6.2: CHIMP's primary (shown on the right) and secondary (shown on the left) input devices.

Table 6.1 presents an overview of the CHIMP's features.

Table 6.1: CHIMP system overview.

Object Generation	Rectangular solid Cylinder Cone Sphere 2D/3D path Library object
Derivative Object Generation	Surface of revolution
6 DoF Manipulation	Scaled-world grab Extender grab
Selection	Occlusion selection Spotlight selection 2D multiselect 3D multiselect
Object Transformation	1D/2D/3D translate 1D/3D rotate 3D scale Duplicate Set center of rotation/scaling Align centers/extents
Surface Attribute Control	Color sample Color set
Viewpoint Manipulation	Pan (scaled-world grab) Two-handed fly Orbital mode Head-butt zoom Dynamic Scaling
Snaps	Grid Ground plane Bounding box vertex Bounding box edge Midpoint bounding box edge Bounding box face Center bounding box face Bounding box normal Bounding box orientation
Object control	Numeric input (interactive numbers)

There are four primary forms of interaction in CHIMP:

- object selection/manipulation
- object generation
- object transformation
- viewpoint manipulation

Object selection/manipulation is CHIMP's default mode. Users select and manipulate objects with their dominant hand. The default form of manipulation is unconstrained 6 DoF manipulation.

Object generation is performed using the CHIMP's five basic modeling objects: rectangular solid, cylinder, cone, sphere, 2D/3D path. The user adds new modeling objects to the environment using his dominant hand.

Object transformation is performed using CHIMP's hand-held widgets. There are six hand-held widgets, some of which perform multiple functions (Table 6.2). The user holds CHIMP's widgets in his non-dominant hand and interacts with them using his dominant hand.

Table 6.2: CHIMP's hand-held widgets.

Widget	Function
Multiselect	Multiple object selection using a 2D or 3D box
Constrained Manipulation	1D/2D/3D translation 1D/3D rotation 3D scaling
Color cube	Color set/sample
Viewpoint Control	Two-handed flying Orbital mode Head-butt zoom Dynamic scaling
Center of action	Set the center of rotation/scaling for the selected object
Alignment	Align selected object centers/extents

CHIMP provides several forms of viewpoint manipulation. At any time the user can pan the world using his non-dominant hand. With CHIMP's hand-held viewpoint-control widget the user can fly through the scene (two-handed flying), quickly look at an object from all sides (orbital mode), look at an object at different scales (head-butt zoom), and look at the scene at different scales (dynamic scaling).

The user selects the current modeling object or hand-held widget using two pull-down, look-at menus. When the user has finished using a modeling object or hand-held widget he gets rid of it using over-the-shoulder deletion. This returns him to CHIMP's default object selection/manipulation mode.

The user controls various aspects of the CHIMP environment using four pull-down control panels (Table 6.3).

Table 6.3: CHIMP's control panels.

Control Panel	Function
Main	• Quit • Cut, copy, paste • Duplicate,delete • Select/deselect all • Group/ungroup • Undo/redo • Reset user/object position/scale • Multiselect fence mode • New object extents mode • Set object color
Snap	• Position • Center • Vertex • Midline • Edge • Face • Grid • Ground plane • Orientation • Bounding box normal • Bounding box orientation • Grid orientation • Snap distance
Grid	• Translation grid spacing (X,Y,Z) • Orientation grid spacing
Numeric Control	• Object position (X,Y,Z) • Object orientation (R,P,Y) • Object scale (X,Y,Z)

I have used the CHIMP system as a testbed for the evaluation of virtual-environment interaction techniques. I consider CHIMP a suitable testbed since computer-aided modeling for the preliminary stages of architectural design is a real-world problem that depends on the complex six degree-of-freedom manipulation of objects and shapes and intuitive viewpoint specification.

The design of CHIMP is the end product of several years of research into virtual environment interaction techniques. The following sections describe some of the more interesting aspects of CHIMP's design and present some of the lessons learned during the development of CHIMP about working in a virtual world.

One of the first tasks I undertook as part of my dissertation research was a review of the state of the art in computer-aided modeling. The results of my review are presented in Appendix A.

Section 6.2 discusses the challenge of mode control in complex virtual-environment applications such as CHIMP.

Sections 6.3 - 6.5 discuss issues concerning object selection, manipulation, and generation.

Section 6.6 gives a brief history of the development of CHIMP's constrained motion widgets, one of the primary techniques used for the controlled transformation of CHIMP objects.

Finally, Section 6.7 discusses numeric input in a virtual world.

6.2 Managing Modes

Most complex applications such as CHIMP rely on a large assortment of tools and modes. Developing effective techniques for managing these modes is one of the most difficult challenges facing developers of virtual-environment applications. Several factors contribute to this situation.

First of all, most immersive virtual-environment systems are limited in terms of the forms and rates of input information that are available. Wearing a non-transparent head-mounted display and standing free in the VR environment makes it nearly impossible to use conventional keyboards, mice, or slider-button boxes effectively. This makes it difficult to use traditional techniques for mode control and interaction such as hot keys, command strings, numeric input, and interactive dials. Available inputs such as gestures and the buttons on hand-held input devices must be used differently for many different forms of interaction and quickly become overloaded. In informal user trials I have observed that this results in user confusion; users often forget which button or gesture invokes a specific function.

Secondly, conventional two-dimensional graphical user interface (GUI) elements (such as toolbars, menus, and dialog boxes) used for mode control and interaction in through-the-window environments are difficult to use in a virtual world. Whereas it is possible to create three-dimensional equivalents of many of these GUI elements, their lack of haptic feedback, their higher dimensionality, and the limitations in display space and display resolution all complicate and slow user interaction.

Finally, users are still unfamiliar with virtual environments and the new virtual-environment-specific interaction techniques. The years of experience users have using conventional through-the-window computer applications typically do not apply to working in a virtual world and in some cases actually confound the use of VR systems.

The remainder of this section describes several of the techniques I have experimented with for selecting tools and controlling modes in virtual environment applications. I include discussions of the merits and limitations of each technique. Though each technique has its advantages (such as simplicity, familiarity, or ease of use), none is totally satisfactory. The most common problem is lack of scalability—the number of modes or tools that can be reasonably selected using the technique is limited. Another problem is ease of use. Though virtual menus and control panels are familiar and scalable, for example, they are often difficult and frustrating to use in a virtual world.

6.2.1 Rotary Tool Chooser

One of the first techniques I developed for switching between high level tools (such as fly, grab, and select) I called the Rotary Tool Chooser (RTC). The rotary tool chooser was a one-dimensional menu based upon J.D. Liang's ring menu [Liang 1994]. A one-dimensional menu is one in which a user has to vary only one degree of freedom to move between menu selections [Mine 1995].

To invoke the RTC the user presses a button on his input device. This results in the available tools being displayed in an arc about the user's hand (Figure 6.3). To select a tool the user simply rotates his hand, like turning a dial, until the desired tool falls within the selection box. To confirm his selection the user releases the input button.

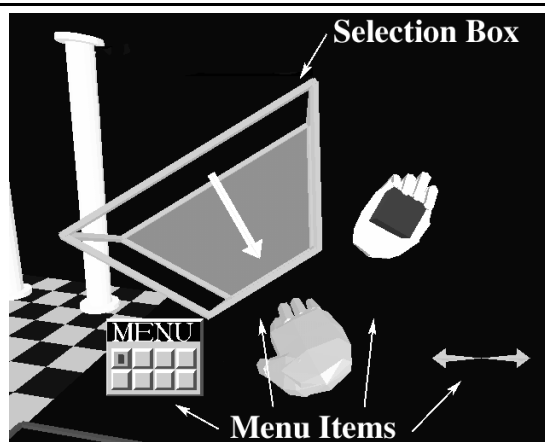


Figure 6.3: Rotary tool chooser.

Instead of being centered about the user's hand, the arc of items can appear in front of the user's face, locked in head space. I have found this to improve interaction with the RTC since the user no longer has to bring his hand into view in order to see the chooser.

The key advantage of the Rotary Tool Chooser is its simplicity. Since the user only has to control one parameter, the twist about one axis (all other changes in user hand position are ignored), the rotary tool chooser is quick and easy to use.

The main problem I had with the RTC is that the number of tools which can be selected in this fashion is limited. The resolution of the HMD limits the number of items which can be displayed clearly in an arc. Similarly the resolution and noise characteristics of the tracking device, combined with the instability of the user's hand, limit the user's ability to discriminate between choices. I have found that users can reasonably select from among six item arranged in a semi-circle given current HMDs and tracking systems.

6.2.2 Two-Dimensional Control Panels

I have also experimented with conventional toolbars and control panels as ways of selecting tools and modes in a virtual environment. An earlier version of CHIMP, for example, included a toolbar, a row of buttons on a virtual control panel which was used to select the current modeling tool.

The advantage of using control panels is that they are familiar to users. The disadvantage is that the invocation of a control panel to change modes breaks the flow of work in an application. This is particularly true in a virtual environment, since control panels are difficult to locate and they are more difficult to use due to the lack of 2D constraints and haptic feedback.

One way to compensate for the lack of haptic feedback in a virtual environment is to limit the number of degrees of freedom that the user has to specify in order to interact with the menu. To use a 2D menu, for example, a user should only have to specify two dimensions. Forcing the user to position his hand in three-dimensions in order to press a virtual button slows down menu interaction and increases user frustration.

In my experience two effective techniques for interacting with 2D control panels are laser beams and occlusion selection. With laser beams the cursor location (i.e. the point of interaction on the control panel) is based upon the intersection of a laser beam attached to the user's hand and the control panel (Figure 6.4). With occlusion selection the cursor location tracks the point on the control panel which visually lies behind some point on the user's virtual hand, such as his fingertip (Figure 6.5). Laser beams minimize required user hand motion but amplify tracking system noise due to lever arm effects. Occlusion selection techniques are more precise but are more fatiguing due to the need to hold one's

hand up over the control panel and the larger hand motions required to reach all points on the control panel.

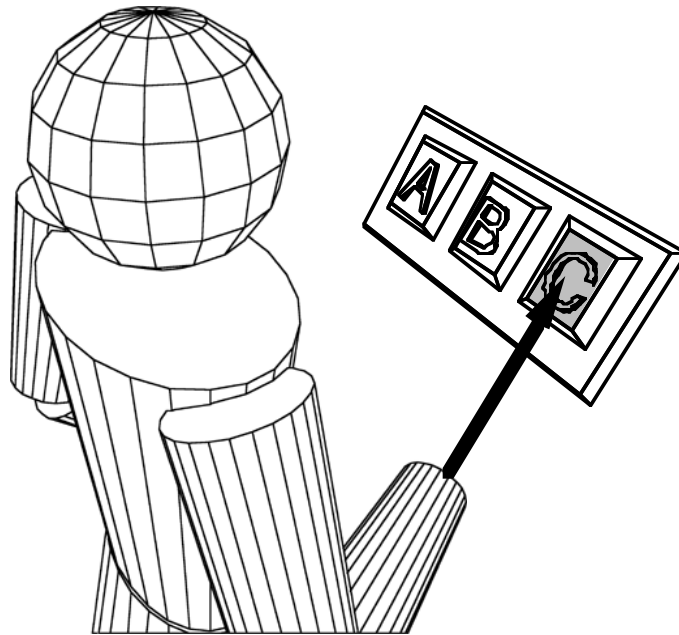


Figure 6.4: Interacting with a control panel using a laser beam.

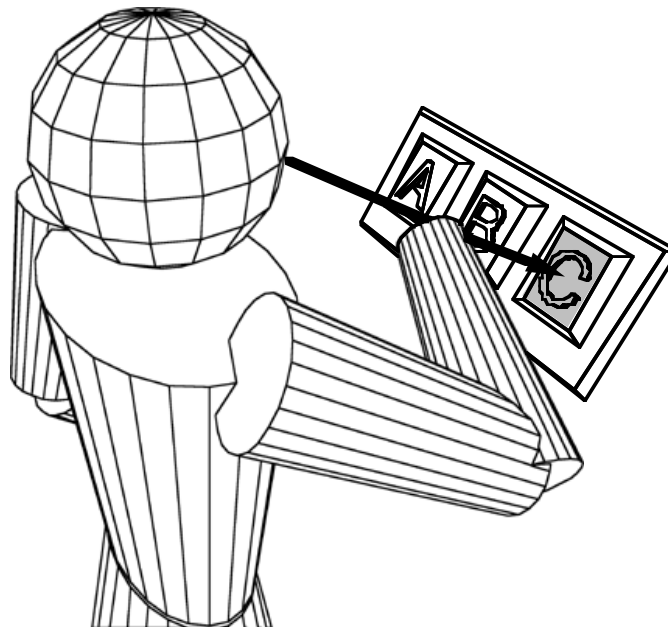


Figure 6.5: Interacting with a control panel using occlusion selection.

6.2.3 Look-at Menus

More recently I have found that interaction and mode selection in a virtual world can be enhanced if one takes advantage of additional user output channels. The look-at menus technique presented in Chapter 3 allows users to select among a limited set of options using just head motion. As described in that chapter, the currently selected object in a menu depends upon the current orientation of the users head. By moving his head the user can move from item to item within a menu.

CHIMP includes two look-at menus which are used to select the current hand-held widget and the current modeling object (Figure 6.6). The user selects items from these menus using his head motion.

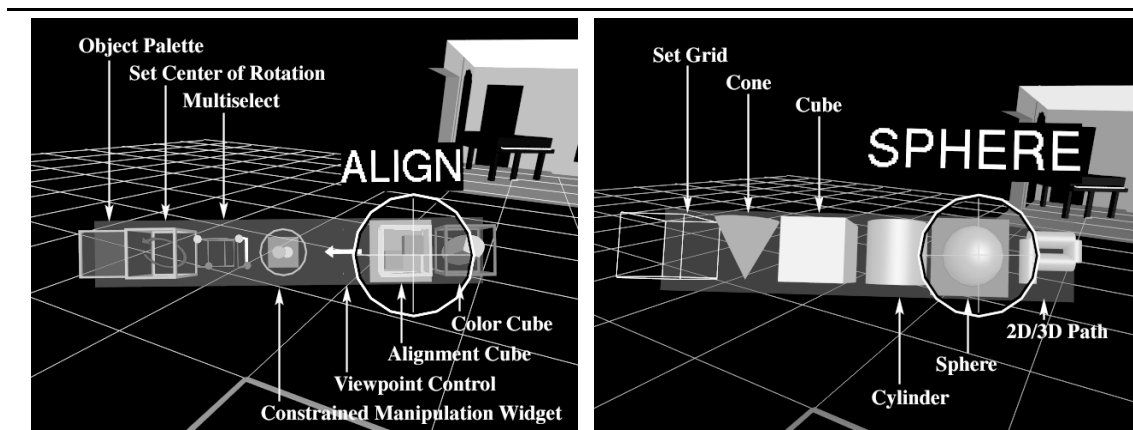


Figure 6.6: CHIMP's look-at menus used to select current hand-held widget (left) and modeling object (right). The circular crosshairs cursor moves with the user's head.

6.2.4 Pull-Down Menus

As described in Chapter 3, menus and control panels are easier to find if they are stored relative to the user's body. Currently, CHIMP's four control panels and two look-at menus are stored just above the user's field of view as a set of pull-down menus. This makes them easy to access and eliminates problems of obscuration of the scene by the menus.

Since the menus are pull-down menus that move with the user as he moves through the environment they are easy to locate and access. Whenever the user needs a menu he simply reaches up and pulls it into view. He does not have to search through the environment to find it. Once he has finished using a menu, he releases it, at which point it floats back up to its body-relative hiding place.

Though pull-down menus are easy to acquire, the number of items that can be reasonably be stored above the user's field of view is limited. In informal user trials I have found that one can reasonably store three menus above one's field of view (one to the left, one directly above, and one to the right of the current view direction). To access many more menus than that requires some supplemental form of menu selection. In the CHIMP system, for example, the four control panels (Table 6.3) are condensed into a single pull-down item. When the user reaches up and pulls down a control panel, he pulls down a single panel (the last one he used). A set of buttons found at the bottom of each panel is used to access the other three panels.

As an alternative to using a row of buttons I have considered using some more scalable form of menu selection such as Kurtenbach's Marking Menus [Kurtenbach and Buxton, 1993] . Instead of pressing a button to switch between menus the user might instead make the appropriate marking gesture (Chapter 2). A two level marking menu with eight items at each level could be used to select up to 64 control panels quickly and easily.

6.3 Object Selection

CHIMP provides two forms of object selection: occlusion selection and spotlight selection. CHIMP automatically switches between these two forms of selection based upon the current position of the user's hand, his head pose, and his field of view. The user can select objects using occlusion selection whenever his hand is in view, spotlight selection is used at all other times. CHIMP also includes several widgets for selecting/deselecting multiple objects.

Occlusion selection is an image plane interaction technique [Pierce, et al., 1997] , selection is based upon the relationship of projected images on the 2D screen in the user's head-mounted display. Attached to the user's dominant hand is a crosshairs cursor. To select an object the user moves the projection of the crosshair cursor over the projected image of the object to be selected and presses the trigger button (Figure 6.7). In three-dimensions this equates to the selection of objects that are intersected by the vector from the user's head through his hand (Figure 6.8).

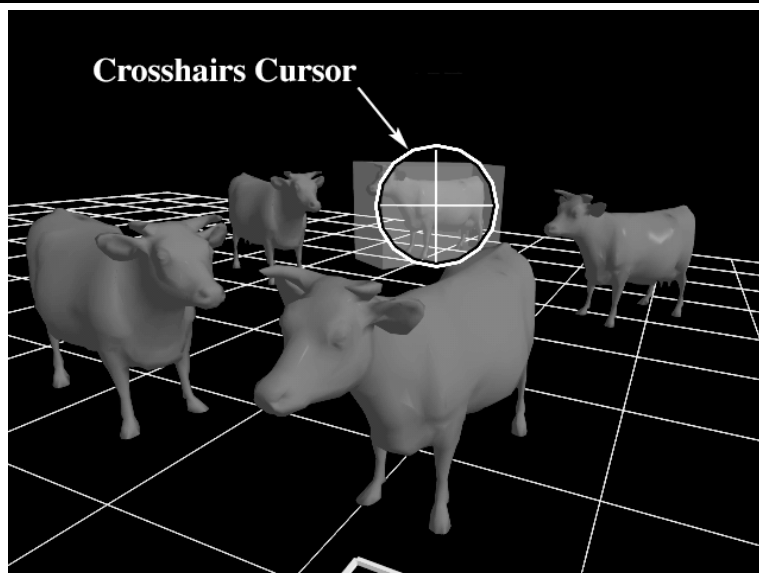


Figure 6.7: Occlusion selection, first person point of view.

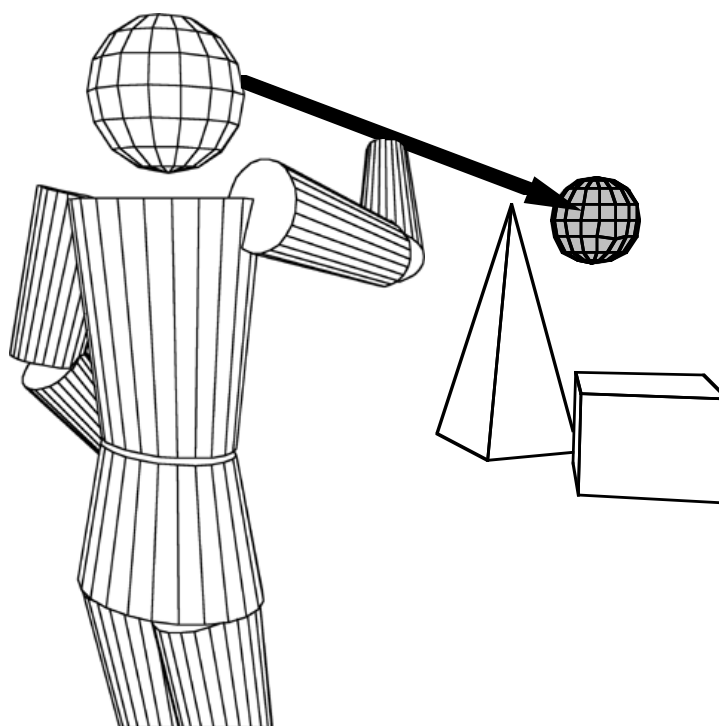


Figure 6.8: Occlusion selection, third person point of view.

In spotlight mode, the user selects an object by pointing a spotlight attached to his dominant hand and pressing the trigger button (Figure 6.9). CHIMP uses a partially transparent cone to represent the spotlight (Figure 6.10). I follow Zhai who showed with his innovative Silk Cursor technique [Zhai, et al. 1994] that transparency can help in the

selection of objects. To signal when objects can be selected, both the spotlight and the indicated object's bounding box change color.

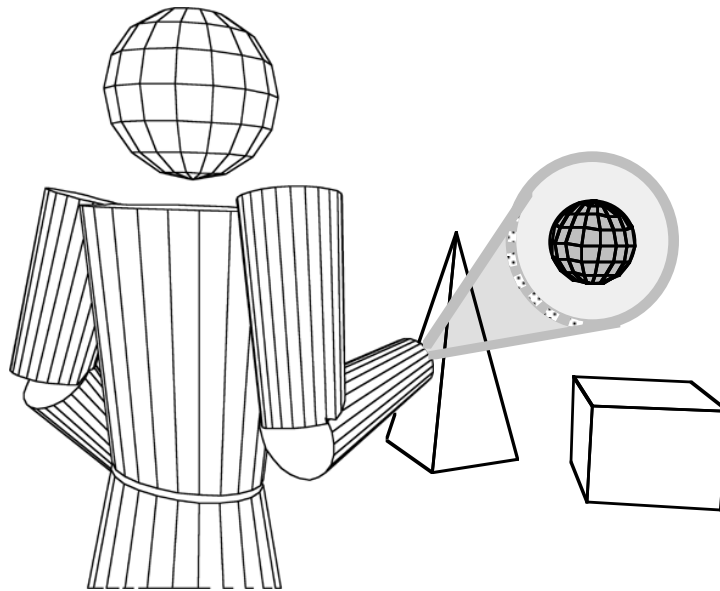


Figure 6.9: Spotlight selection, third person point of view.

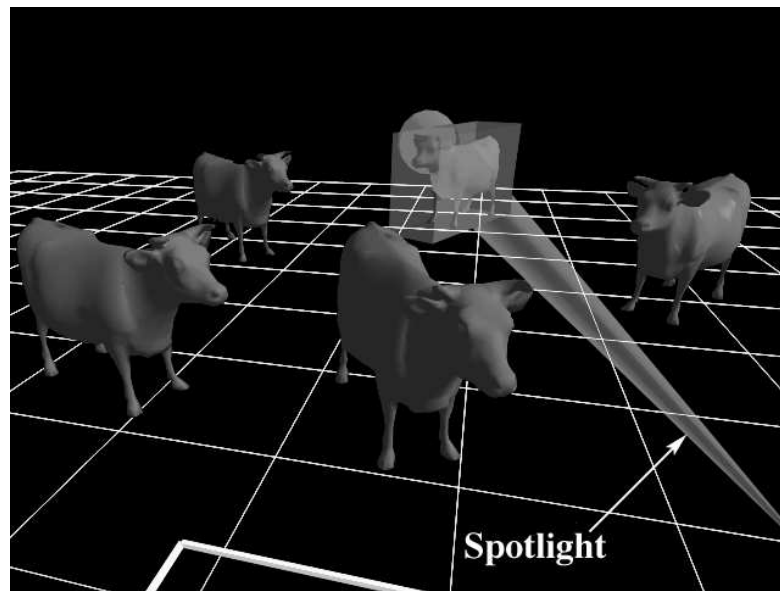


Figure 6.10: Spotlight selection, first person point of view.

CHIMP uses spotlight selection, selection of objects which fall within a cone, instead of laser beam selection, the selection of objects whose bounding box is intersected by a beam, because spotlight selection facilitates selection of small and distant objects that are hard to hit with a laser beam [Liang and Green, 1994] .

I include occlusion selection in CHIMP because it is an effective technique for selecting among multiple closely grouped objects, and it is similar to conventional mouse-based techniques used in through-the-window systems (and thus easily understood). Occlusion selection, however, requires the user to keep his arm upraised for extended periods of time, since his hand must be visible for occlusion selection to work. Spotlight selection avoids this potentially tiring situation by allowing users to select objects with their hands resting comfortably at their side. Selecting between several closely grouped objects, however, is somewhat more difficult with spotlight selection than with occlusion selection. The two selection techniques, therefore, nicely complement each other; the more accurate occlusion selection can be used when the user's hand is in view, the less fatiguing spotlight selection when his hand is out of view.

6.4 Object Manipulation

CHIMP offers two primary forms of direct manipulation, extender grab and scaled-world grab, both of which were described in Chapter 3. The user manipulates objects with extender grab if, at the start of the grab, his hand is outside of his current field-of-view, otherwise he uses scaled-world grab. This corresponds directly to the relationship between spotlight selection and occlusion selection; objects selected with a spotlight are manipulated using extender grab, those selected with occlusion selection are manipulated using scaled-world grab. The switching between the two forms of manipulation is automatic and intuitive to the user.

The reasons for including two forms of manipulation in CHIMP are similar to the reasons for including two forms of object selection. Scaled-world grab, like occlusion selection, works best when the user's hand is in view. Extender grab and spotlight selection work well when the user's hand is out of view. In addition, the pairings of manipulation and selection techniques are logical. Scaled-world grab, the manipulation of objects co-located with one's hand, works well with occlusion selection, the selection of objects occluded by one's hand. Similarly, extender grab, the manipulation of objects at a distance, works well with spotlight selection, the selection of objects at a distance.

Initially I used two different techniques for remote object manipulation in CHIMP, hand-centered and object-centered grab. In hand-centered grab, objects pivot about the center of the user's hand, like objects at the end of a long stick. In object-centered grab objects pivot about their own center. This allows the user to rotate an object about its center while being physically removed from the object (standing back to get a better perspective, for example).

I switched to scaled-world grab and extender grab because they minimize unnecessary user interactions. Users can make large-scale changes to the environment more quickly and easily than they can with hand-centered and object-centered grab. Though large-scale changes are possible using hand-centered grab, particularly when the lever arm between hand and object is big, these changes are primarily restricted to directions perpendicular to the lever arm. The range of motion in object-centered grab is limited because changes in the position and orientation of the user's hand are only mapped 1:1 onto the position and orientation of the selected objects. In both cases arbitrary, large-scale changes in position and orientation of an object require the user to repeatedly grab, move, release, and then re-grab the object.

Contrast this with scaled-world grab and extender grab where the resulting motion of the object is automatically scaled based upon the distance of the object from the user at the start of the grab. This means that the user can bring to most distant object to his side in a single operation. It also means that the range and precision of manipulation correspond to the visual scale at which the user is viewing the scene. When looking at a tree close up he can move it within its local environment. If he pulls back so that he can see the entire forest he can easily move the tree from one end of the forest to the other. In addition, for scaled-world grab, the user has a better sense of the position and orientation of the manipulandum since it is co-located with his hand (Chapter 4).

6.5 Object Generation

Though two-handed techniques are good for the creation of approximate geometric objects, I have found that, for more exact specification of objects, one-handed techniques are easier to control, less tiring, and more precise.

The difficulty I encountered with many two-handed techniques is that the user has to control too many degrees of freedom simultaneously. To create an object with two hands the user typically makes an "I want it this big" gesture to simultaneously size all three dimensions of the object. Though excellent for the rough specification of an object's position, orientation, and size, this form of interaction is less suitable for the precise construction of objects relative to other objects in the scene. The user, for example, may want to create a new object that is on top of object A, as wide as object B, and as tall as object C. This is difficult to do if the user is simultaneously controlling all three-dimensions of the object's size as well as its position and orientation using two hands.

I have opted, therefore, for a three-step process of object construction. Users first specify the position and orientation of the origin of the new object. Next they specify the

size and shape of the object's profile. Finally they set the object's height. Though a greater number of steps, I personally find that the amount of information that must be dealt with at each step is more manageable.

6.6 Constrained Object Manipulation

In real-world manipulation, people depend intuitively on constraints - we move objects along the floor, for example. CHIMP provides several widgets which can be used for constrained object manipulation. Selected objects can be:

- translated along a line (1D) or in a plane (2D)
- translated (3D) without rotation
- rotated (1D) about any vector
- rotated (3D) without translation
- uniformly scaled

The evolution of CHIMP's constrained manipulation widgets has taught several important lessons about interaction in a virtual world.

6.6.1 Co-Located Widgets

Initially I experimented with widgets that were co-located with the objects they control, a direct adaptation to the virtual environment of the 3D widgets work performed at Brown University [Conner, et al., 1992] . I found that there are several important differences between using a widget through the window and using one in the immersive environment.

Typically, users select widgets in a through-the-window environment by placing a 2D cursor on top of the projection of the widget on the 2D screen. They interact with the widget by moving the 2D cursor, the resulting behavior of the associated object depending upon the programming of the widget. A 2D translation widget, for example, might restrict resulting movement of an object to a plane parallel to the plane of the screen. Alternately, the widget might restrict object movement to a plane aligned with one of the scene's principal planes. In either case, the resulting object movement depends upon the change in the projected location of the 2D cursor within the selected plane.

Analogous image-plane interaction techniques can be used in an immersive environment. A key difference in the immersive environment, however, is that the image plane used for interaction is tied to user head pose, which changes continually during interaction (users can not keep their head still for extended periods of time). This must be

accounted for in the specification of widget behavior. For the screen-aligned 2D translation widget described above, for example, a snapshot of the user's head pose must be used to select the orientation and position of the screen-aligned plane used to constrain object motion. If not, the plane would change continuously during interaction as the user moved his head.

More typically than image-plane techniques, users interact with widgets in virtual environments using physical interaction or laser pointing. In the case of physical interaction, users interact with widgets by reaching out and grabbing them. Since the user can leverage off of real world interaction skills this form of widget interaction is natural and intuitive. With laser pointing users select and interact with widgets by pointing at them using hand-held laser beams. The advantage of laser beams is that they allow users to interact with widgets at a distance. Note that some form of intersection must be calculated between the laser beam and the widget to define the point of interaction.

Figure 6.11a shows the design of the original constrained-manipulation widget used in the CHIMP system. Figure 6.11b shows a second generation of the widget. Both widgets were co-located with the object they control. The user interacted with both using laser beams. In both, the design of the widgets was driven by the poor resolution of the head-mounted displays in use. Affordances had to be highly exaggerated in order to be visible within the head-mounted display.

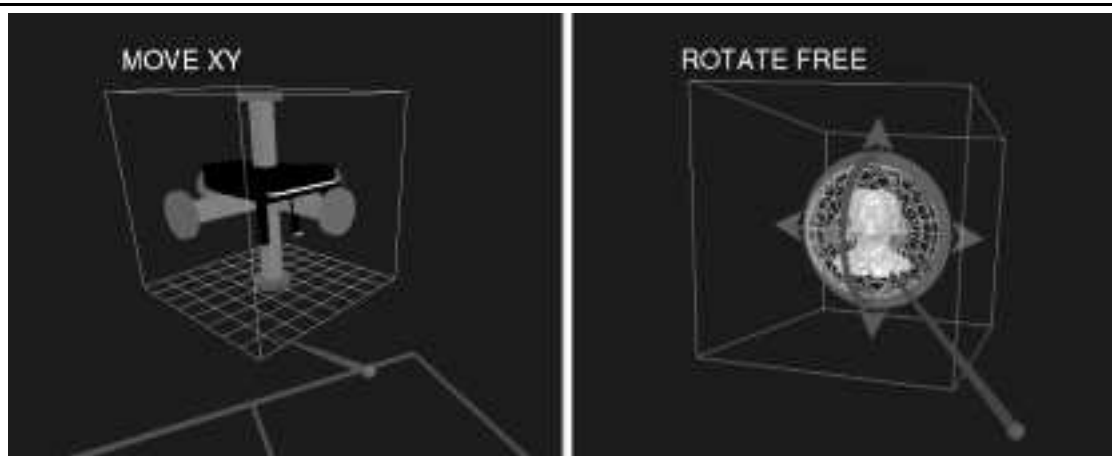


Figure 6.11: A) First and B) second generation constrained manipulation widgets.

Changes between the first and second generation widgets were primarily intended to improve the widget's affordances and minimize obscuration. Translation handles, for example, were changed from three-dimensional bars to simple arrows to make the direction of motion afforded by the handle much clearer. Rotational handles were changed from

three-dimensional disks (which were intended to suggest wheels) to two-dimensional rings. This was intended to minimize obscuration of the object by the handle when the handle was viewed from a high angle of incidence.

In my experience, there are several problems with using co-located widgets in a virtual environment. When using physical interaction, the user must be able to reach a widget to use it. If it is out of reach, the user is forced to switch to a locomotion mode to move to the widget. This slows down interaction and breaks the flow of work. When using laser beams, the user will find small and/or distant widgets difficult to select and use. Finally, in both forms of interaction, the lack of haptic feedback in a virtual environment makes widget interaction more difficult since the user must depend heavily on visual information to select and use a widget.

6.6.2 Hand-Held Widgets

To overcome some of the limitations of widgets co-located with the objects they control I switched to hand-held widgets (Chapter 3).

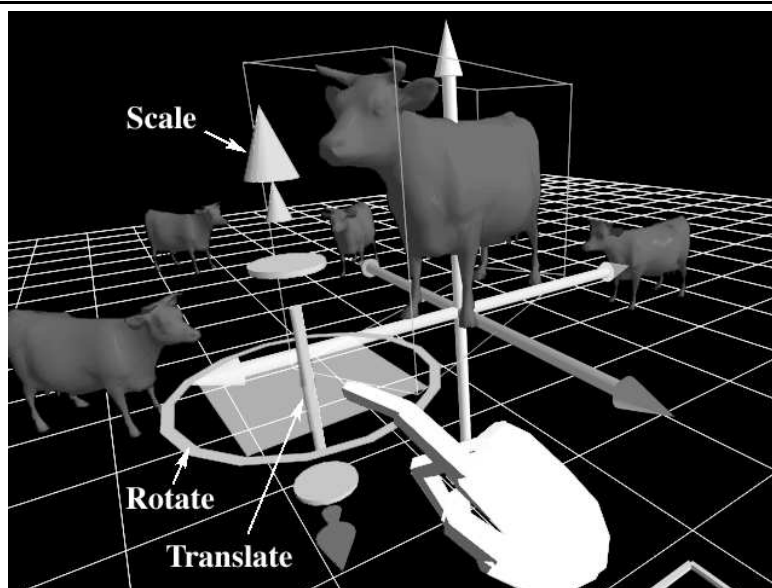


Figure 6.12: CHIMP's hand-held constrained manipulation widgets.

Figure 6.12 shows the hand-held constrained manipulation widgets currently used in the CHIMP system. The user selects the current mode of transformation using relative hand position; grab the ring to rotate the object, grab the arrows to scale it, grab the rod to move it along a vector. The axis of rotation or translation is based upon the orientation of the hand-held widget which the user controls with his non-dominant hand. He can choose to have the axis of interaction snapped to the principal axis closest to the current orientation

of the widget or to have it precisely aligned with the current widget orientation. The user can adjust the scale of the resulting transformation by pressing a modifier key and changing his hand separation; for fine-grained motion he brings his hands together, for large scale transformations he pulls them apart.

I tried and rejected several alternate forms of hand-held widget interaction before settling on the current design. I tried both relative hand orientation (hands pointing parallel, anti-parallel, and perpendicular) and hand separation (Figure 6.13) for selecting the current manipulation mode (translate, rotate, or scale). The problem with these forms of mode selection is that they lacked good visual feedback to help the user make his selection, and they could only be used to select among a small set of choices. Similarly I tried using arm extension for controlling the scale of transformation. Though this was easy to use, it was considerably more tiring than hand separation.

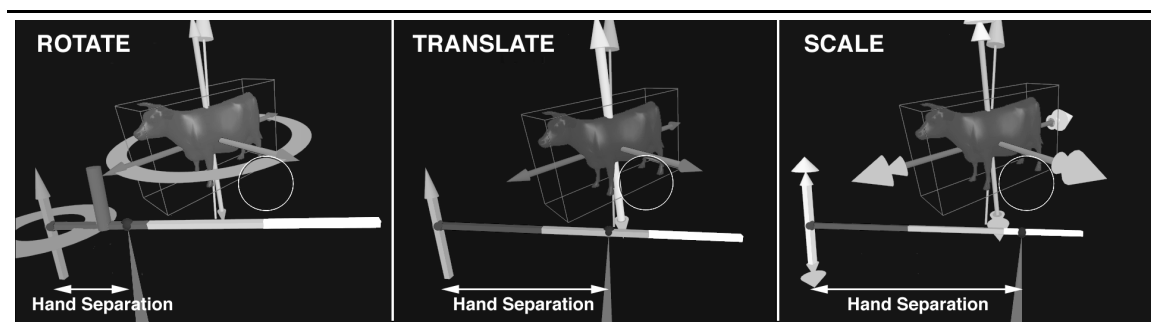


Figure 6.13: Constrained manipulation mode selection based upon hand separation.

I have found that hand-held widgets are easier and more efficient to use. Widgets stored in body-relative locations are easier to locate and access. Widgets held in the hand do not obscure the object they control. They are easier to see since the user can easily bring them up for a closer look. Finally, since the user is interacting relative to his own hand, he can take advantage of proprioception when using the widget.

6.7 Numeric Input in a Virtual World

In the CHIMP system *interactive numbers* are used to provide a means for specifying numeric values from within the virtual environment.



Figure 6.14: Numeric input using the Arithma Addiator.

The original interactive numbers were inspired by the old mechanical calculating devices such as the Arithma Addiator (Figure 6.14) in which numbers are input by moving physical sliders up and down using a pointed stylus. In place of the pointed stylus is a laser beam emanating from the user's hand. The user can point the beam at any digit in a number and press and hold on the trigger button. This will invoke a linear pop-up list that includes the digits 0-9 (figure 6.15). By moving his hand up or down the user can then slide the desired new digit into place.

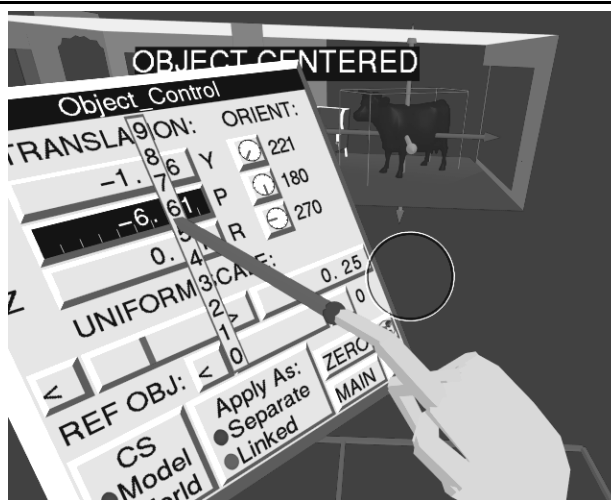


Figure 6.15: Linear interactive numbers.

In more recent versions I have replaced the linear pop-up list with a pie menu which surrounds the currently selected digit (Figure 6.16). To select a number the users strokes in the direction of the desired digit in the pie menu (see [MacKenzie, et al., 1994] for an analysis of similar techniques used for numeric input in pen-based computers). The advantage of this scheme is that users quickly learn the stroke direction required for a

particular digit and as a result doesn't have to pay as close attention when inputting a number. Another advantage is that the pie menu can be treated as an interactive dial. If the user moves the cursor in a circle about the selected digit, the appropriate carry will be propagated when the cursor crosses the zero point on the dial. This means the user can use the interactive number to crank up (or down) the entire value, the selected digit controlling the rate of change of the selected parameter.

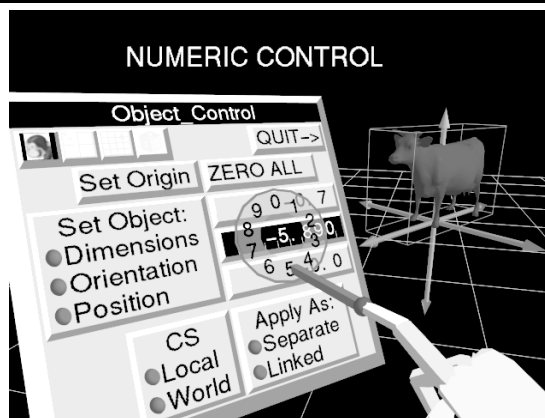


Figure 6.16: Rotary interactive numbers.

If the user selects an empty space to the left of the number's current most-significant digit he can change that space to any digit (1-9) and zeros will fill in all the intermediate spaces. This allows him to quickly enter large values. If the user points at an existing number, presses a modifier key, and then moves his hand left or right he copies the currently selected digit left or right, clearing any numbers already there.

The user can also grab the decimal point and slide it left or right (increasing or decreasing the number of digits to the right of the decimal point), and he can change the sign of the current value at any time.

The advantages of the interactive numbers scheme is that it allows one to double up input and output in the same region of the control panel instead of having a numerical display and a separate input widget (such as a slider or a virtual keypad). Moreover, in principle, interactive numbers are unbounded, though in practice they are limited by the number of digits that can be displayed. Furthermore, the technique doesn't require the user to learn how to use an unfamiliar input device such as a chord keyboard or have to struggle with a virtual keypad that suffers from the lack of haptic feedback (touch typing without the touch). Interactive numbers are, however, susceptible to noise in tracking data.

Techniques such as interactive numbers may be unnecessary if one has means for reliable voice input. However there may be cases in which it is not possible or desirable to use voice input (loud environmental conditions, difficulty talking all day, or simple user preference). In such cases a manual technique for numeric input is required.

Chapter 7

Final Words

7.1 Conclusions

One of the fundamental advantages of working in a virtual world is that natural forms of interaction such as reaching out to grab and manipulate an object, leaning forward to get a closer view, and turning one's head to look at something, can be powerfully enhanced in ways not possible in the real world. Simple and intuitive actions become powerful and effective virtual-environment interaction techniques that minimize user work such as scaled-world grab, head-butt zoom, and orbital mode.

The virtual world, however, is not without incidental difficulties. In a typical virtual world a person's vision is poor, and the sense of touch is limited at best. He communicates without voice and the written word, and interacts using whole-hand gestures but not his fingers. He lacks physical surfaces on which he can ground himself to steady his actions and on which he can rest to limit fatigue. These incidental difficulties complicate interaction in a virtual world and limit the effectiveness of virtual-world applications.

The goal of my research has been a better understanding of the strengths and weaknesses of working in a virtual world. My interests lay in the development of intuitive and effective virtual-environment interaction techniques that enhance natural forms of interaction, exploit the benefits of working in a virtual world, and compensate for its limitations.

In this dissertation, I proposed that one use proprioception to compensate for the lack of haptic feedback in a virtual world. I presented three ways that a user can exploit proprioception during virtual-environment interaction: by manipulating objects within his natural working volume (direct manipulation), by storing objects at fixed locations relative

to his body (physical mnemonics), and by using his body sense to help recall actions used to invoke commands or to communicate information (gestural actions).

To bring instantly within reach objects which lie outside of the user's reach, I devised automatic scaling. Automatic scaling minimizes user work and enables direct manipulation within a user's natural working volume.

I developed several novel virtual-environment interaction techniques based upon my framework of body-relative interaction. Both formal user studies and informal user trials have shown these techniques to be effective forms of interaction in a virtual world.

The virtual docking study confirmed the intuition that it is more effective to manipulate an object that is co-located with one's hand (as is the case in scaled-world grab) than it is to manipulate an object held at some offset. Subjects were able to align hand-held docking cubes with target cubes significantly faster than they were able to align docking cubes held at a fixed and variable offsets. The results of the experiment, however, did not measure the effects of automatic scaling on the user's ability to manipulate objects.

The virtual widget interaction study demonstrated that subjects could take advantage of proprioception when interacting with virtual widgets. In the absence of visual feedback, they were able to return to a point on a widget held in their hand significantly more accurately than they could to a similar point on a widget floating in space. This suggests that proprioceptive cues can be used to augment visual feedback when interacting with hand-held widgets.

Observations during informal user trials of the CHIMP system support and complement the results of the formal user studies. Users have found body-relative interaction to be intuitive and easy to use. They quickly adapt to techniques such scaled-world grab, head-butt zoom, and over-the-shoulder deletion. Automatic scaling of the world for both manipulation and locomotion is a powerful technique which enables users to quickly and easily accomplish their goals. Though it has no real-world counterpart, it feels natural and intuitive. Users manipulate distant objects with ease and are often surprised to learn that scaling has taken place.

7.2 Contributions

The overall contribution of my work is an improved understanding of what it means to work in a virtual world. The byproducts of this understanding are natural, intuitive, and effective virtual-environment interaction techniques that minimize user work and thus maximize user efficiency.

I consider the following specific items to be the significant contributions of this research:

- *A Framework for Virtual Environment Interaction.* The development of a framework for virtual environment interaction based upon proprioception.
- *Automatic Scaling.* The invention of automatic scaling as a technique for bringing remote objects instantly within reach.
- *Body-relative Interaction Techniques.* The creation of several novel virtual-environment interaction techniques based upon the framework of proprioception for virtual-environment interaction.
 - Direct Manipulation:
 - Scaled-world grab for manipulation
 - Scaled-world grab for locomotion
 - Physical Mnemonics
 - Pull-down menus
 - Hand-held widgets
 - Gestural Actions
 - Head-butt zoom
 - Two-handed flying
 - Over-the-shoulder deletion
- *The virtual docking user study.* A formal user study which compared the manipulation of virtual objects co-located with one's hand and the manipulation of virtual objects held at an offset.
- *The virtual widget interaction study.* A formal user study which contrasted interaction with widgets held in the hand and interaction with widgets floating in space.
- *CHIMP.* Integration of numerous body-relative interaction techniques into a real-world system, the Chapel Hill Immersive Modeling Program.
- *Interactive Numbers.* The development of Interactive Numbers, a manual technique for numeric input in a virtual world.
- *Modeling System Review.* A detailed review of the current state-of-the-art in computer-aided modeling.

7.3 Future Work

Localized Haptic Feedback

Though proprioception greatly enhances virtual-environment interaction, precise manipulation is still harder in virtual spaces than in real space. Several factors complicate fine-grained manipulation.

First, the lack of physical work surfaces and haptic feedback makes the controlled manipulation of virtual objects much more difficult. Users typically manipulate virtual objects by holding their arms out without support. In the real world, a person generally grounds the arm at the forearm, or elbow, or heel of hand to steady hand motions and to reduce fatigue when performing precise manipulation.

Second, humans depend upon naturally occurring physical constraints to help determine the motion of objects they are manipulating (sliding a chair along a floor, for example). Whereas it is possible to implement virtual equivalents of physical constraints [Bukowski and Sequin, 1995], it is more difficult for the user to take advantage of these constraints without haptic feedback. He can only see that the chair is on the floor, he can't feel the contact, hear it, or sense the vibration as the chair slides.

Third, users in a virtual world must typically do without the fingertip control they rely on for the fine-grained manipulation of objects in the real world. Though instrumented gloves show promise for the fine-grained manipulation of objects [Kijima and Hirose, 1996], their use has been primarily limited to the recognition of a few commands associated with hand posture [Kessler, et al., 1995].

I am interested in exploring the use of hand-held tablets as a means of giving users a real surface on which they can work using haptic constraints (following the lead of [Sachs, et al., 1991] and [Stoakley, et al., 1995]). The tablet can be used as a two-dimensional drawing surface (to define detailed two-dimensional shapes) or it can be used as the input space for a two-dimensional menu (allowing users to interact precisely with widgets and controls).

If the user interacts with the tablet using a hand-held stylus, he can take advantage of the user's fingertip control precision. In addition the friction between tablet and stylus and the grounding of the stylus against the tablet give the user better control.

To provide a larger work surface the tablet can be docked in a larger fixed physical surface such as a lectern or a drafting table which can also provide grounding and support during object manipulations (see related work in [Mapes and Moshell, 1995]).