Working in a Virtual World: Interaction Techniques Used in the Chapel Hill Immersive Modeling Program

Mark R. Mine

Department of Computer Science
University of North Carolina
Chapel Hill, NC 27599-3175

mine@cs.unc.edu

Abstract

This paper presents a description of the interaction techniques used in the Chapel Hill Immersive Modeling Program (CHIMP). CHIMP is intended for the preliminary stages of architectural design. It is an immersive system; users work directly within a virtual world. The main goal has been to develop interaction techniques that exploit the benefits of working *immersed* while compensating for its limitations. Interaction techniques described and discussed in this paper include:

- Action at a distance
- Look-at menus
- Remote controls (hand-held widgets)
- Constrained object manipulation using two-hands
- Two-handed control panel interaction
- Worlds in miniature
- Interactive numbers

Keywords:

Virtual reality, Virtual environments, Computer-aided modeling, Geometric modeling, User interface design, Two-handed interaction, Two-handed interfaces, Interactive computer graphics.

1. Introduction

1.1. CHIMP Overview

The UNC-Chapel Hill Immersive Modeling Program (or CHIMP for short) is a virtual environment application for the preliminary phases of architectural design. 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 uses 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]). 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.

1.2. Design Philosophy

The underlying thesis of the CHIMP system is that one can design three-dimensional (3D) spaces more easily in an immersive environment than one can modeling through-the-window (using conventional workstation inputs and displays), because the immersive environment avoids some of the incidental difficulties of a two-dimensional (2D) interface. Two-dimensional displays inherently inject ambiguity into the interpretation of displayed information [Gregory 1973]. Two-dimensional input devices, such as the mouse or the data tablet, preclude the direct specification of three-dimensional positions, orientations, and extents.

Working in a virtual world, on the other hand, allows:

- Immersion within the virtual space
- Direct perception and six degree-of-freedom (DOF) manipulation of three-dimensional virtual objects
- Intuitive view specification via head-tracking

The virtual environment, however, is not without its own set of incidental difficulties. View specification via headtracking, though one of the most intuitive form of view specification (turning one's head to look at something), requires cumbersome

and expensive hardware (position trackers and high-speed interactive graphics systems). Immersion using head-mounted displays greatly complicates a person's interaction with the real world (talking with co-workers, getting a sip of coffee).

If real-world work is to be accomplished in a virtual world, we must develop interaction techniques that take into account the strengths and weaknesses of the virtual environment. It is insufficient to simply import through-the-window interaction techniques or to try to recreate real-world interaction in a virtual world.

The differences between working through-the-window and working immersed are analogous to the differences between a craftsman working in front of a workbench and one moving about a work site with a tool belt attached to his waist. One must be careful about the distribution of controls and information about a user, since these controls and information can be difficult to locate and reach (like a misplaced hammer).

Similarly, in a virtual environment one must compensate for the lack of haptic feedback which people depend upon for the controlled manipulation of objects in the real world (due to a lack of fine motor control). Humans, for example, depend upon additional constraints, such as a ruler, to move an object in a straight line.

Interaction techniques that take into account the unique nature of this new medium can powerfully enhance natural forms of real-world interaction in ways not possible through a window. The interaction techniques presented in this paper have been found to be effective forms of interaction for use within a virtual world.

2. Related Work

Several other systems have moved beyond the limitations of two-dimensional input and output in the interactive design of virtual objects. One can categorize these systems according to the type of input devices used (number and degrees-of freedom of each), and the type of outputs used:

Input:

- Single 3DOF input (3D position only)
- Single 6DOF input (3D position and orientation)
- Two 6DOF inputs (one for each hand)

Output:

- Conventional workstation display
- Static stereo display (workstation monitor with stereo output)
- Head-tracked kinetic display (non-stereo display with head tracking)
- Head-tracked stereo display (Fish Tank VR)
- Immersive head-mounted display

Figure 1 shows a comparison of the inputs and outputs used in the systems discussed below.

Christopher Schmandt of the Architecture Machine Group at the Massachusetts Institute of Technology (MIT) used a video display combined with PLZT shutter glasses and a half-silvered mir-

OUTPUT	INPUT		
	3DOF	6DOF	2 X 6DOF
Conventional Display			Sachs/MIT/3-Draw Shaw/Alberta/THRED
Stereo Display		Schmandt/MIT	
Head Tracking		Liang/Alberta/JDCAD	
Head-tracked Stereo		Deering/Sun/HoloSketch	Gobetti/CRS4
Immersive Head-mounted Display	Clark/Utah	Butterworth/UNC/3DM Bowman/Ga.Tech/CDS	Stoakley/U.Va./WIM Mapes/IST/Polyshop Mine/UNC/CHIMP

Figure 1: Interactive Design Systems Input/Output Comparison. Systems are specified by Author/Institution/System-name (if applicable).

ror to create a stereo image that was registered with a physical workspace [Schmandt 1983]. Users could reach in and directly interact with virtual objects in the workspace using a 6DOF *magic wand*. The wand used an early Polhemus tracking system and included a button for user input. Though users were enthusiastic about the ability to work directly in a three-dimensional space, they encountered problems such as: magnetic interference in the tracking system; errors in depth judgment; and difficulty in creating planar and rectilinear objects without the proper constraints, both physical (such as arm rests) and/or virtual (snap to grid).

The 3-Draw system developed by Sachs et. al. at MIT used two hands, each tracked by a 6DOF sensor, to interact with an image shown on a conventional (non-stereo) display [Sachs, et al. 1991]. Sachs developed 3-Draw for the design of complex free-form shapes. In the 3-Draw system the user held a tracked palette in one hand that acted as a movable reference frame in modeling space. By using a stylus held in the other hand, one could draw two-dimensional curves on the palette which resulted in curves in three-dimensional space. Sachs reported that users found the interface natural and quick, and that the simultaneous use of two hands provided kinesthetic feedback that enabled users to feel as though they were holding the objects displayed on the screen.

JDCAD, an interactive 3D modeling system designed and built by Jiandong Liang at the University of Alberta, used a single 6 DOF input device (the bat) combined with a kinetic head-tracked (non-stereo) display [Liang and Green 1994]. JDCAD included: object selection using the spotlight metaphor; innovative menuing systems (the daisy and the ring menus); and object creation, manipulation, and viewing techniques customized for the 6DOF input. Liang showed promising results when comparing JDCAD with conventional through-the-window modeling systems (that use 2D input devices). Development of JDCAD system (now called JDCAD+) continues at the University of Alberta with the addition of new animation editing and scene composition functions. These functions allow non-programmers to construct interesting virtual environments without the aid of a programmer [Halliday and Green 1996].

Chris Shaw's THRED system (Two-handed Refining Editor), developed at the University of Alberta, is a two-handed (6DOF each) computer-aided design system for sketching free-form polygonal surfaces such as terrains and other natural objects [Shaw and Green 1994]. Surfaces are hierarchically-refined polygonal surfaces based on quadrilaterals. Models are displayed on a conventional workstation monitor. In the THRED system each hand has a distinct role, the less dominant hand setting context (such as the axis of interaction) and the dominant hand being responsible for actions such as picking and manipulation.

Michael Deering at Sun Microsystems Computer Corporation has created the HoloSketch system, a VR based sketching system that extends the 2D sketch-draw paradigm to 3D. A head-tracked stereo system that uses a 6DOF wand, the HoloSketch system supports several types of 3D drawing primitives such as rectangular solids, spheres, cylinders, cones, free-form tubes and many more. Users control HoloSketch using a 3D multi-level fade up circular menu which, when invoked, fades up centered around and slightly behind the current location of the wand. The fade-up menu is used to select the current drawing primitive or to perform one-shot actions such as *cut* or *paste*. Deering reports that trials with non-computer scientists (the intended users) have shown that significant productivity gains are possible over conventional 2D interface technology.

Gobbetti and Balaguer at the Center for Advanced Studies, Research and Development in Sardinia, created an integrated environment for the rapid prototyping of 3D virtual worlds [Gobbetti and Balaguer 1995]. Their system is built on top of the VB2 system, a graphics architecture based on objects and constraints developed by the authors at the Swiss Federal Institute of Technology, Lausanne [Gobbetti and Balaguer 1993]. The animation system has a fully 3D user-interface that uses two hands for input (using a mouse and a Spaceball), and a head-tracked stereo display (using LCD shutter glasses) for output. Their system uses multi-way constraints to simplify the development of 3D widgets and the specification of compound widgets and interaction techniques.

Back in the mid 70's at the University of Utah, Jim Clark built one of the earliest interactive design systems that used an immersive head-mounted display. Clark developed his pioneering system for use in the interactive design of free-form surfaces in three-dimensions [Clark 1976]. Clark's system used a mechanically tracked head-mounted display (designed by Ivan Sutherland) and a 3-D wand that computed positions by measuring the

length of three monofilament lines attached to the ceiling. Primarily limited by the state of available technology (in particular the tracking technology and graphics system), the system addressed many of the issues that face developers of interactive design systems today.

3DM (Three-dimensional modeler) is the interactive design system created at the University of North Carolina in the early 90's [Butterworth, et al. 1992]. Using a stereo head-mounted display and a single 6 DOF bat, 3DM allows users to interactively create geometric objects from within a virtual environment. 3DM includes several grid and snap functions; it lacks, however, many of the other aids and constraints that we since have found necessary for accomplishing precise work.

The University of Virginia's Worlds-in-Miniature (WIM) system (discussed in more detail below and in [Stoakley, et al. 1995]) is an innovative system that uses a hand-held miniature representation of the virtual environment (the WIM) for object manipulation and viewer navigation. Using two 6DOF inputs (one attached to a physical clipboard held in one hand and one attached to a 6DOF bat held in the other hand) users manipulate objects in the virtual world by manipulating objects in the virtual hand-held miniature (which is attached to the clipboard). Moving the copy of an object in the WIM results in the corresponding movement of the full-sized original. Moving a representation of the user in the WIM effects a change in viewpoint. The authors report that users quickly adapted to the WIM interface finding it intuitive and easy to use.

The Conceptual Design Space system is an immersive head-mounted display system for interactive architectural design that was developed at Georgia Tech's Graphics, Visualization and Usability Center [Bowman and Hodges 1995]. Using a single 6DOF input, users interact with objects and widgets using a ray-casting metaphor. The developers of the CDS system have adapted many of their interface elements and tools directly from 2D interfaces. To help create a highly usable immersive architectural design tool, the developers of CDS have worked closely with actual architects and members of the College of Architecture at Georgia Tech.

In the Polyshop system, developed at University of Central Florida's Institute for Simulation and Training, two hands are used to scale, rotate, and translate objects within the virtual world (see [Mapes and Moshell 1995]). Users select specific modes of operations using two ChordGloves, which provide a discrete form of gesture recognition. ChordGloves have separate electrical contacts at the end of each finger and on each palm. Different gestures consist of different chord patterns of electrical contacts. Dan Mapes continues this work at Multigen Inc. with the development of SmartScene, a virtual-world scene-building application.

3. CHIMP Interaction Techniques

3.1. Conventions

Several of the interaction techniques used in CHIMP require the use of both hands. These tasks can be labeled as *asymmetric bimanual tasks* [Guiard 1987], two-handed activities with each hand performing a different role. Though the assignment of roles to hands depends solely on the preference of the user, I will describe actions as being associated with either the right or the left hand to simplify discussions. The user, however, can easily switch hands since there is no lateral bias in the input devices used or the assignment of tasks to hands.

The CHIMP system uses two separate bats, one for each hand. A bat is a hand-held input device which contains a tracking sensor (used to detect the position and orientation of the user's hands) and several buttons. The main button on each input device (used for most actions) is called the *trigger*. Each input device has a coordinate system associated with it, and within the coordinate system a main pointing axis.

In the discussions that follow, statements such as "rotate about the main axis of the user's left hand" are meant to imply a rotation about the main axis of the input device held in the user's left hand.

3.2. Action At A Distance

CHIMP uses Action-at-a-distance (AAAD) for the remote selection of and interaction with objects within the virtual world. In AAAD mode, users select objects by pointing a spotlight (attached to their right hand) and pressing the trigger button (see figure 2). CHIMP uses a partially transparent cone to represent the spotlight. We follow Zhai who showed with his innovative Silk Cursor technique [Zhai, et al. 1994] that transparency can help in the selection of objects. The spotlight also changes color when pointing at an object to signal when objects can be selected.

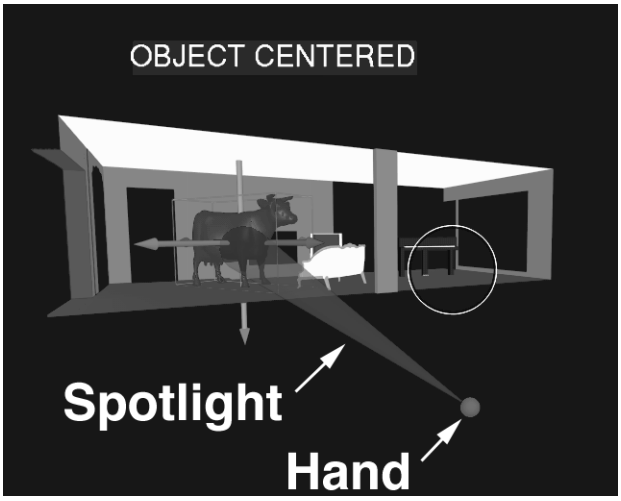

Figure 2: At-a-distance selection using CHIMP's spotlight

CHIMP uses spotlight selection (selection of objects which fall within a cone) instead of laser beam selection (selection of objects whose bounding box is intersected by a beam), because spotlight selection facilitates selection of small and distant objects that can be hard to hit with a laser beam.

CHIMP provides two basic forms of at-a-distance interaction for selected objects: hand-centered and object-centered. In both forms of interaction, changes in the position and orientation of the user's hand are mapped onto the position and orientation of the selected objects. The only difference between the two is the center of rotation used in applying changes in orientation to the object.

In hand-centered manipulation, objects move about the center of the user's hand, like objects at the end of a long stick. When the lever arm between the hand and the object is large, hand-centered manipulation allows large scale object motions without the user having to navigate through the environment (to reach the object or its target destination). Hand-centered interaction, however, is particularly sensitive to noise in the tracking data and to instability in the user's hand position.

In object-centered manipulation, objects rotate and move relative to their own center. Object-centered interaction enables one to make a localized change while standing back to get a global view.

3.3. Look-At Menus

I developed the look-at menu technique to help avoid some of the problems with mode selection (such as hand-centered or object-centered mode) that I had previously encountered in virtual environment applications. Mode selection has typically been dependent upon the mapping of the various modes onto buttons of the input device, since many immersive VR applications make do without the benefits of a keyboard. In a virtual environment, physical keyboards are difficult to use since one is moving about and blind to the outside world, and virtual keyboards are awkward due to the lack of haptic feedback. The mapping of modes to buttons complicates interaction with the application (since it is difficult to remember which button does what), and limits the number of modes that can easily be selected using the input device. The look-at menus interaction technique is an attempt to provide a scalable technique for mode/tool selection in the virtual world. CHIMP uses look-at menus, for example, for:

- changing the current center-of-interaction (hand or object)
- selecting the current interaction tool (e.g., manipulate or fly)
- invoking the various control panels throughout the environment

Look-at menus can be attached to any object in the environment including the user (one at each of his hands for example). Look-at menus consist of a pop-up menu (which is initially hidden) and a hot point (typically represented by a small red sphere).

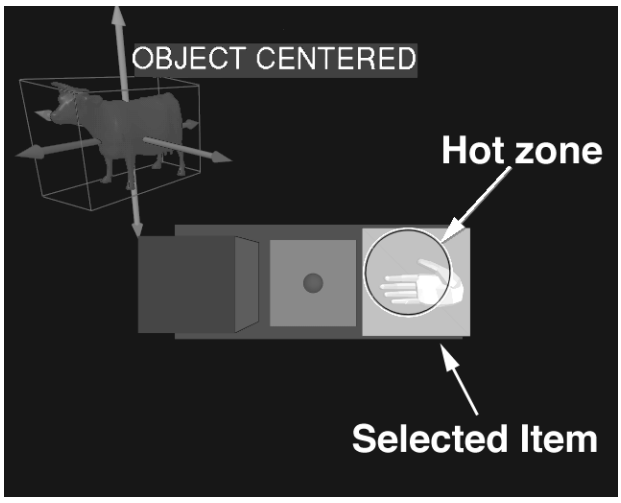

Figure 3: Look-at menu

The pop-up menu consists of a collection of icons, one for each menu choice (see figure 3). At the center of the user's field of view (and thus tied to the user's current direction of view) is the look-at menu hot zone. The hot zone is represented by a wireframe circle that looks like the focus region in a single-lens reflex camera. When a menu's hot point falls within the look-at menu hot zone, that menu becomes active and the user can interact with it in several different ways. To signal to the user that a hot-point has fallen within the hot-zone CHIMP replaces the wireframe ring by a thicker colored ring.

If the user presses and holds the look-at menu button (on his input device) when a menu is active the corresponding pop-up menu become visible, centered above the hot spot. The active item in the menu depends on the current gaze direction of the user, simulated in the absence of eye tracking by the current view direction of the user's head. To select a new item the user moves his head to look at the desired item. When the user releases the button on the input device the currently selected item is activated. If the user is looking beyond the extents of the pop-up when he releases the button no menu item is selected. Menus can be used to select modes and/or tools and callbacks can be associated with each menu item.

If, instead of holding down the pop-up menu button, the user quickly presses and releases it, the user can select the default menu item without having to wait for the associated pop-up menu to become visible. This allows the user to quickly select a default item without having to look at a menu and cognitively process the displayed information (similar to the style of interaction found in the innovative Marking Menus techniques, see [Kurtenbach and Buxton 1993]). The default item is the one in the pop-up that is centered over the hot-point. If desired, the default action can be a null action, forcing the user to wait for the menu to become visible before making a valid selection.

Since menus can be attached to objects in the virtual world, menus are associated with the objects they control rather than placed at some arbitrary position in screen space. This helps reduce the cognitive distance between menu and controlled object. In addition, placing menus in the environment takes advantage of the ability to distribute information throughout the virtual space rather than concentrating the information in a limited fixed space (such as a menu bar).

Another advantage of the look-at menu scheme is that it makes use of an additional input channel (i.e., head orientation) and does not tie up the user's hands for menu interaction. This hands-free operation makes it possible to interact with a menu without interrupting the current operation. The user can change between hand-centered and object-centered manipulation, for example, without releasing the currently held object.

On the other hand, the distribution of menus throughout the environment can make it potentially more difficult to find a desired menu. In addition, since the menus are not visible until activated, the user can not know what a menu contains until he activates it.

3.4. Two-handed Interaction

3.4.1. Overview and Related Work

Recently, many researchers have begun exploring the benefits of using two-handed interaction techniques in human-computer interfaces. The work of Bier, et. al. on the Toolglass and Magic Lenses interface, for example, demonstrated that the use of two hands exploits user's everyday skills and can reduce steps, cursor motion and errors during interaction ([Bier, et al. 1993]). In the Toolglass and Magic Lens system, one hand positions a movable see-through interface that is pointed through (to underlying objects) using a cursor controlled by the other hand.

Goble, Hinckley, et. al. at the University of Virginia ([Goble, et al. 1995]) have demonstrated the advantages of using two hands in conjunction with *props*, (real-world hand held tools) in their *Netra* system, an interactive tool for neurosurgical planning. In the Netra system neurosurgeons control the current viewpoint and cross-section plane used to display medical imaging data on a conventional workstation by manipulating real-world props held in their two hands. A small doll head held in one hand controls the viewpoint and scale of the displayed information. A small plate held in the other hand controls the current cross-sectioning plane. The use of physical props takes advantage of a user's highly developed ability to manipulate real-world objects and provides visual and kinesthetic feedback that reminds the user of the prop's use.

Several of the systems discussed in the introduction section above (3-Draw, U.Va's WIM, Polyshop, Gobetti's Animation system, and THRED) also make use of two hands for object

manipulation and environment interaction. In all systems, researchers report that users quickly adapt to the two-handed mode of interaction, finding it intuitive, easy to use, and in many cases more effective than one-handed interaction.

A theoretical basis for the design of effective two-handed interaction techniques can be found in the work of Yves Guiard [Guiard 1987]. Guiard (of the Centre National de la Recherche Scientifique in Marseille France) demonstrates that the actions of a person's right hand are typically performed relative to a coordinate frame defined by his left hand. In addition the right hand works at a finer spatial-temporal scale than the left hand and the actions of the right hand typically start after those of the left hand . Effective two-handed interaction techniques must take these results into consideration, with the left hand setting the context in which the right hand can interact.

In the CHIMP system, two-handed interaction is used for:

- Selecting the particular sub-mode of constrained object manipulation
- Performing constrained object manipulation
- Selecting the current flying sub-mode
- Setting the current orbit distance used in orbital-mode flying
- Interacting with the various CHIMP palettes

The user enters a two-handed mode of interaction when his hands are closer together than a predetermined limit (currently 0.5 meters), otherwise he is in AAAD mode using the CHIMP spotlight (a one-handed form of interaction since selection and interaction depends solely on the position and orientation of the user's right hand).

The following sections describe some of the two-handed interaction techniques used in CHIMP.

3.4.2. Remote Controls

Originally CHIMP used widgets that were co-located with the objects they were controlling (analogous to the style of interaction found in the seminal work on 3D widgets at Brown University, see [Conner, et al. 1992]). In the immersive environment it proved difficult to select the individual widgets (either by moving one's hand to the corresponding widget or by pointing at them using a beam). As an alternative I have been experimenting with *remote controls*, hand-held widgets used to remotely interact with objects in the environment.

In a remote control, widgets are attached to the user's left hand instead of being co-located with the object they control. The user interacts with the widgets using his right hand. Though this increases the cognitive distance between object and control it facilitates interaction with the various widgets. I believe this is because proprioceptive information makes it easier to interact with a widget attached to one's hand than to interact with a widget floating in space (whose position can only be deduced visually). In addition, because the user can use proprioceptive information, he doesn't have to look at his hand (and the widget) to interact with it. This helps to reduce visual clutter, since expert users can move the widgets out of view. Users are no longer distracted by the visual representation of the widgets and can instead focus on the object of interaction.

One example of a hand-held widget in the CHIMP system is the *mode bar*. The mode bar is a widget used for selecting the current constrained manipulation submode (translate, rotate, or scale) or the current flying submode (pointing mode or orbital

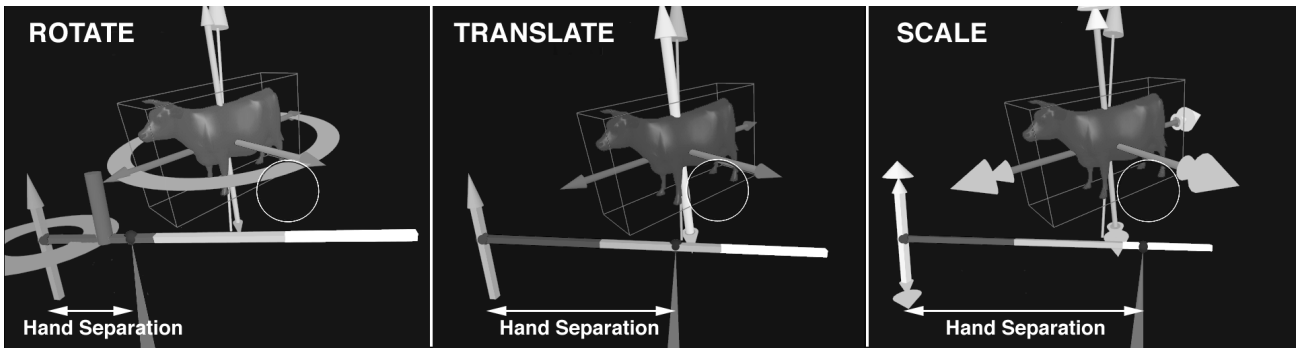

Figure 4: Two-handed mode selection using the CHIMP mode bar. Each mode is selected based upon a different hand-separation.

mode). The mode bar is simply a thin bar attached to the left hand, that is divided into color-coded regions, each region corresponding to a particular sub-mode (see figure 4). To select a submode the user simply moves his right hand to the corresponding colored region and presses the trigger button.

The mode bar always lies along the vector between the user's two hands. As a result sub-mode selection is independent of the hand's current position and orientation and depends solely upon the user's current hand separation. For example, for flying mode selection, if the user's hands are closer together than 0.25 meters he is in a pointing mode of flying. If they are between 0.25 meters and 0.5 meters, he is in an orbital mode of flying. If his hands are more than 0.5 meters apart, he exits two-handed interaction and resumes the default one-handed AAAD interaction.

Though expert users can depend primarily upon proprioceptive information to interact with the mode bar, the visual feedback provided by the graphical representation is extremely helpful for novice users. The user can look at the mode bar and see precisely how far apart he must hold his hands to enter a particular sub-mode.

3.4.3. Constrained Object Manipulation

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

- moved along a line or in a plane
- rotated about any vector
- rotated about the object's center
- uniformly scaled about the object's center

The user selects a constrained interaction mode using the mode bar, described above. The user specifies the resulting transformation using his two hands as follows:

The user first selects the axis of interaction based upon the orientation of his right hand. The user can choose between having the axis of interaction snapped to the principal axis closest to the current hand orientation or having the axis of interaction precisely aligned with the current hand orientation.

Once the user has selected an axis of interaction, he initiates the constrained manipulation by pulling and holding the trigger button with his right hand. The magnitude of the transformation depends upon the relative motion of his two hands. For translation the user specifies the motion of the selected objects by moving his right hand along the selected axis, like sliding a bead along a wire. For rotation the user twists his right hand about its axis, like turning a screwdriver. For scale the user determines the size of the objects by the separation of his two hands. If he increases his hand separation the objects scale up, if he decreases it the objects scale down.

To provide a large dynamic range of possible translations CHIMP multiplies the resulting translation of the selected objects by a scaling factor that depends on the distance of the user's hand from his body. To perform fine-grain motions he holds his hand close to his body. For large-scale object translations he extends his arm fully.

Though complicated to describe, the two-handed interaction style is easy to learn and easy to use. Users have found it easier to interact with CHIMP's various modes of constrained manipulation using the two-handed interaction techniques than they did with the previous one-handed style of interaction.

3.4.4. Two-handed Interaction and Flying

The CHIMP mode bar is also used for selecting the flying mode. Users can choose between two flying modes: pointing mode and orbital mode.

In pointing-mode flying, a user flies in whatever direction he is pointing with his right hand.

Orbital mode is a unique form of viewpoint specification first pioneered by Jim Chung at UNC (see [Chung 1994]). In orbital mode selected objects are constrained to stay in front of the user, regardless of which direction he is looking. The side of the object that is visible to the user depends on his current view direction: look up and he sees the object's bottom, look down and he sees its top (see figure 5). This nicely matches the behavior we have observed in users of interactive design systems (such as architects) who like to zoom around a model to view it from all sides, and to zoom above to get a bird's eye view.

Orbital mode is a powerful technique for quickly and easily setting one's viewpoint relative to a selected object. It makes effective use of an additional input channel (head orientation). Though non-obvious, it turns out to be a highly intuitive form of view specification. Note that one can make a distinction between orbital *viewing* and

orbital *flying*. In orbital viewing (shown in Figure 5), a selected object moves about the user's head. In orbital flying, it is the user that moves through the environment, as he views an object from all sides.

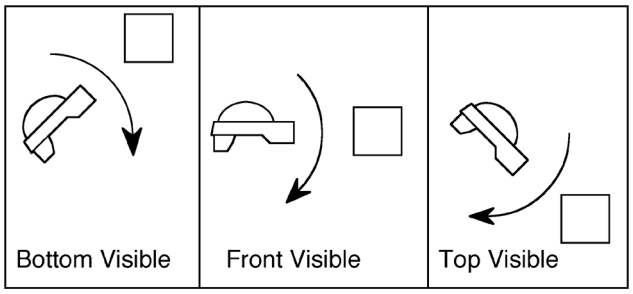

Figure 5: Orbital mode viewing

For either mode, flying is controlled using the trigger: press to start, release to stop. In pointing-mode the user's speed of motion depends on the extension of his right arm; held closely he flies slowly, extended fully he flies at top speed. In orbital mode, the orbit distance (the distance from the user to the center of the object or selected objects) depends upon the separation of his two hands. When he holds his hands close together he orbits closely, if he spreads his hands apart his orbit distance increases. Tangential velocity is controlled by how quickly he turns his head. Radial velocity depends on the rate of change of his hand separation.

I have also experimented with a two-handed flying mode in which the direction of motion is based upon the current vector between the user's two hands, and the speed of motion depends upon the user's hand separation. Though I thought that this would be an effective form of flying control, in practice it seems difficult to use. Possibly this is because it is easier to understand which direction a single hand is pointing (as in pointing-mode flying) than it is to understand which direction the vector between two hands is pointing. In addition, though hand separation is an effective speed control by itself, it is less effective when combined with relative hand position for direction control. It is difficult for a user to keep his speed constant as he moves his hands around to change his direction of motion.

3.4.5. Control Panel Interaction

Two hands are also used in the CHIMP system for all control panel interaction. CHIMP control panels are the virtual environment equivalent of standard graphical user interface dialog boxes, however, instead of being locked into screen space, the control panels float in 3D space with an associated position, orientation, and scale.

When a control panel becomes active it attaches to the user's left hand. To interact with the panel he uses a laser beam emanating from his right hand. Using a laser beam reduces the control panel interaction task from three dimensions to two. This helps speed up menu/widget interaction. Widget selection techniques in menu systems that require the user to co-locate a hot point (such as the tip of his finger) with a widget in three-dimensional space are difficult to use and suffer from the lack of haptic feedback we depend upon when pushing a button in the real world.

Preliminary experience also indicates that users find it easier to interact with panels attached to their hand than they did with panels floating freely in virtual space. One disadvantage of the hand-based scheme, however, is that if a panel is invoked as the result of a user action, the user may not know that a new panel has become active since he may be holding his hand out of view. This is unlike the conventional GUI case where a new dialog box appears prominently in the center of the screen and blocks all other operations until dealt with.

The CHIMP system includes several control panels: the general environment control panel, an object control panel and a tear-off object palette.

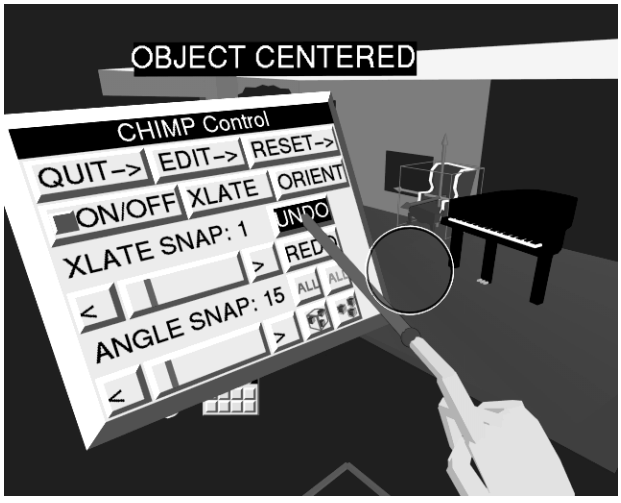

Figure 6: Environment control panel

The environment control panel (figure 6) provides access to functions such as:

- high level editing functions (such as cut, copy, and paste)
- functions to reset object scale, user position and environment state (to return it to the state found at program startup)
- undo/redo functions.

The object control panel (figure 7) is used to display and input information about the currently selected objects. When invoked, the object control panel displays the current position, orientation, and scale of the selected objects. If more than one object is selected, a slider can be used to scroll through the information of each of the selected objects. Changing a value on the object control panel changes the state of the object in the virtual world. Thus, using the object control panel, the user can precisely set the position, orientation, and scale of a selected object. Numeric input is accomplished using the interactive-numbers technique discussed below.

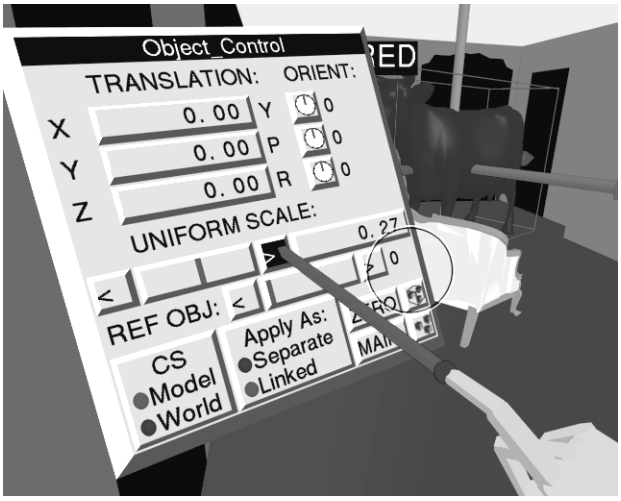

Figure 7: Object control Panel

The tear-off object palette (figure 8) contains miniature representations of all the objects currently loaded into the environment. A user can add a new object to the environment by grabbing one of the miniature copies on the object palette and tearing off a new copy (by moving his hand away from the palette). The new copy can then be placed anywhere in the environment. The object palette can contain both primitive objects (such as cones, cylinders and spheres) and more complex library objects (such as tables, pianos and chairs).

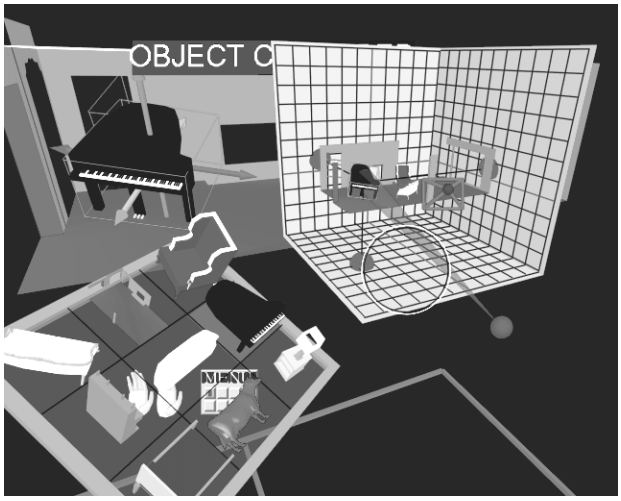

Figure 8: Tear-off object palette

In the CHIMP system, multiple control panels can be attached to the user's left hand at one time. To help avoid visual clutter only one panel is visible at a time, the visible panel depending upon the orientation of the user's left hand. As the user rotates the sensor about its main axis different panels pop into view. For example the tear-off object palette is attached to the back of the environment control panel. If when looking at the environment control panel the user flips the panel over (by rotating the hand-held sensor 180 degrees) the control panel will disappear and the object palette will pop into view. The advantage of this system is that the user can quickly access multiple panels without visual clutter becoming unmanageable due to multiple panels floating in space. The disadvantage is a lack of affordances, since a user may not know that another panel is on the back of the current one unless he turns his hand around (albeit a very easy operation).

3.5. Worlds In Miniature

Instead of extending one's reach into the environment it is possible to bring the environment into reach. This is accomplished in the CHIMP system using an implementation of the Worlds-in-Miniature (WIM) metaphor. In a WIM system (see figure 9) a three-dimensional miniature representation of the current environment floats in front of the user, easily within arms reach. The user can reach into the WIM and grab and manipulate objects just as he can grab and manipulate objects in the immersive world surrounding him. Any change he makes in the miniature representation results in the corresponding change in the immersive

world, and vice versa. See [Stoakley, et al. 1995] for some interesting results of similar research being performed at the University of Virginia.

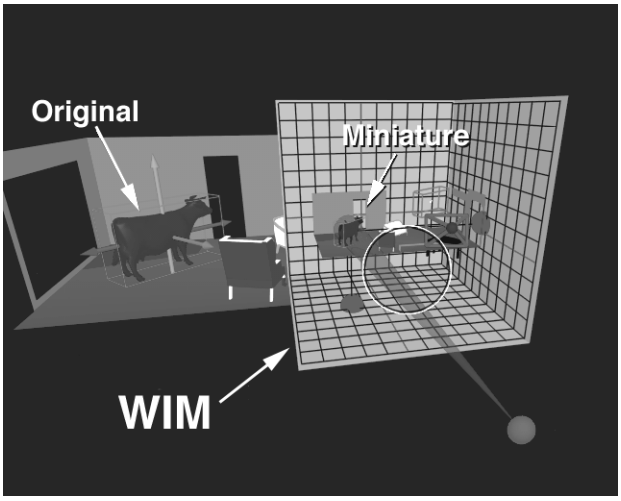

Figure 9: Worlds-in-miniature

The advantage of using a WIM system is that it allows one to perform large scale manipulations more easily (such as moving a chair across a room without having to actually move through the environment to reach it). In addition it helps provide global context while immersed in one's current location. Researchers at the University of Virginia are also exploring the use of WIMs for navigation in the virtual world (see [Pausch, et al. 1995]).

Though highly effective for the gross manipulation of objects, the WIM interface does not solve the precise manipulation problem (and in fact aggravates it, since one is restricted to large scale motions of objects due to the small scale of the WIM). The controlled manipulation of objects in a WIM (and in virtual environments in general) depends on the implementation of additional constraints such as real-time collision detection (see [Gottschalk, et al. 1996]) or innovative techniques such as Bukowski's object associations (see [Bukowski and Sequin 1995]).

There are several potential enhancements to CHIMP's current implementation of the WIM metaphor.

Since the amount of information displayed in a WIM can quickly become overwhelming, it is important to be able to display an arbitrary subset of the environment. If the WIM can only display the entire environment (as is currently the case in the CHIMP system), it makes it difficult to distinguish, select, and interact with objects in the WIM.

It would also be useful if these arbitrary subsets can be displayed at different scales since we often observe users alternating between global and local views when working in interactive design systems. This would enable users to interact with multiple views of the same space, each at a different location and scale.

Finally, if multiple WIMs can be displayed simultaneously, one can also move objects from WIM to WIM, allowing for large scale interaction with better local resolution and control. A user, for example, could move a chair between two WIMs, each showing a different room in a house in detail, instead of having to move a chair across a single WIM displaying the entire house at a lower resolution.

Given these capabilities, the WIM technique becomes in effect a three-dimensional windowing system, with each WIM presenting a different view of the virtual space.

In the University of Virginia system, the WIM is attached to a tracked clipboard held in the user's left hand. By moving the clipboard the user changes the position and orientation of the WIM, allowing him to view it from different directions. The user also has the option of detaching the WIM from the clipboard, leaving it floating in space. This is required to help minimize user fatigue (resulting from holding the clipboard up in the air) and to allow the user to pass his head through the WIM (by leaning into the WIM floating in space) which would not be possible if it was still attached to the clipboard.

In the CHIMP system the WIM metaphor has been combined with orbital mode viewing (described above) to provide an orbital WIM. Normally the WIM is left floating in space. In this state the user is free to interact with the WIM, move his head inside the WIM for a close up view, or to stand back to get a global view. If the user wishes to view the WIM from another direction he *grabs* the WIM (by pointing at one of its three orthogonal grid planes and pulling on the trigger) which starts orbital-mode viewing. He is then free to view the WIM from any direction. By moving his hand in and out from his body the user changes the WIM's orbital distance.

One advantage of an orbital WIM is that it maintains the correspondence between WIM and world

orientation. This helps to avoid the additional cognitive step of registering the orientation of a hand-held miniature with the immersive world. Further research is required, however, to determine which form of WIM viewing is more effective (a hand-held WIM or an orbital WIM).

3.6. Interactive Numbers

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

Figure 10: Numeric input using the Arithma Addiator

Working with interactive numbers is similar to using one of the old mechanical calculating devices (such as the Arithma Addiator, for example, see Figure 10) 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 pop-up list that includes the digits 0-9 (figure 11). By moving his hand up or down the user can then slide the desired new digit into place. Alternately, if the user moves his hand left or right he can copy the currently selected digit left or right.

If he 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 in large values. Alternately, if he copies that space to the right he clears out any numbers he encounters.

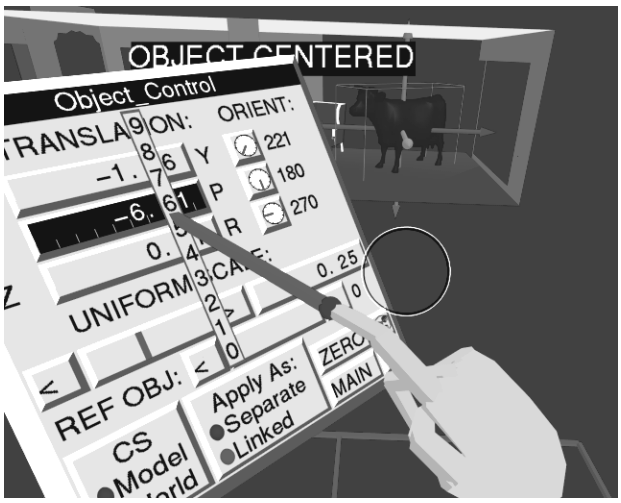

Figure 11: Interactive Numbers

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 interactive numbers 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 theory, interactive numbers are unbounded, one can enter in any magnitude number (though in effect 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 and require that user hand motion be scaled or filtered to smooth out interaction.

Techniques such as interactive numbers may be unnecessary given a means for reliable voice input. However there may be cases in which it is not possible or desirable to use voice input (due to environmental conditions, because users may be unable to talk all day long without irritating their throat, or simply because users prefer not to). In such cases interactive numbers or some similar technique may be required.

4. Conclusions and Future Work

In writing this paper I had four main goals:

- To describe the interaction techniques used in the CHIMP system
- To demonstrate some of the advantages of working in a virtual world
- To illustrate some of the limitations of virtual-environment interaction
- To suggest some potential means of overcoming those limitations

I have found the interaction techniques used in the CHIMP system to be effective techniques for working in a virtual world. The use of two-hands, intuitive view specification via head tracking, look-at menus, hand-held widgets and remote object manipulation all take advantage of the benefits offered by the virtual environment.

Future work on the CHIMP system will include:

- Development of new tools and techniques for object creation
- Exploration of interactive techniques for the specification of object-relative manipulations including:
  - Feature-based manipulations (e.g., parallel to an existing face or perpendicular to an edge).
  - Object alignment and distribution functions
- Incorporation of real-time collision detection to aid in relative-object positioning
- Continued exploration of two-handed interaction techniques
- Continued refinement of virtual-environment interaction techniques such as look-at menus and interactive numbers

The ultimate future research goal is the exploration of my original thesis: that it is more effective to create three-dimensional shapes from within the virtual world than it is through-the window. This will involve the evaluation of the CHIMP system by architects and designers who will compare its effectiveness and ease of use with existing through-the-window design products.

5. References

E. A. Bier, M. C. Stone, K. Pier, W. Buxton and T. D. DeRose (1993). "Toolglass and magic lenses: the see-through interface." *Computer Graphics (Proceedings. SIGGRAPH '93)* : 73-80.

D. Bowman and L. F. Hodges (1995). *Immersive Design Tools for Virtual Environments*. SIGGRAPH '95 Technical Sketch, Los Angeles, CA, ACM SIGGRAPH.

R. W. Bukowski and C. H. Sequin (1995). "Object Associations: A Simple and Practical Approach to Virtual 3D Manipulation." *Proceedings of the 1995 ACM Symposium on Interactive 3D Graphics* : 131-138.

J. Butterworth, A. Davidson, S. Hench and T. Olano (1992). "3DM: A Three Dimensional Modeler Using a Head-Mounted Display." *Computer Graphics (Proceedings 1992 Symposium on Interactive 3D Graphics)* **25**(2): 135-138.

J. Chung (1994). *Intuitive Navigation in the Targeting of Radiation Therapy Treatment Beams.* University of North Carolina, Ph.D. Thesis

J. H. Clark (1976). "Designing Surfaces in 3-D." *Communications of the ACM* **19**(8): 454-460.

D. B. Conner, S. S. Snibbe, K. P. Herndon, D. C. Robbins, R. C. Zeleznik and A. vanDam (1992). "Three-dimensional widgets." *Computer Graphics (1992 Symposium on Interactive 3D Graphics)* **25**(2): 183-188.

H. Fuchs, J. Poulton, J. Eyles, T. Greer, J. Goldfeather, D. Ellsworth, S. Molnar, G. Turk, B. Tebbs and L. Israel (1989). "Pixel-Planes 5: A heterogeneous multiprocessor graphics system using processor-enhanced memories." *Computer Graphics (Proceedings SIGGRAPH '89)* **23**(3): 79-88.

E. Gobbetti and J.-F. Balaguer (1993). "VB2-an architecture for interaction in synthetic worlds." *Proceedings of UIST 93: The Sixth Annual Symposium on User Interface Software and Technology* : p. viii+267, 167-78.

E. Gobbetti and J.-F. Balaguer (1995). "An integrated environment to visually construct 3D animations." *Computer Graphics (Proceedings SIGGRAPH 95)* : p. 518, 395-8.

J. C. Goble, K. Hinckley, R. Pausch, J. W. Snell and N. F. Kassell (1995). "Two-handed spatial interface tools for neurosurgical planning." *Computer* **28,**(7): p. 20-6.

S. Gottschalk, D. Manocha and L. Ming (1996). "OBB-Tree: A Hierarchical Structure for Rapid Interference Detection." *To appear in: Computer Graphics (Proceedings SIGGRAPH '96)* :

R. L. Gregory (1973). *Eye and Brain*. London, World University Library.

Y. Guiard (1987). "Asymmetric Division of Labor in Human Skilled Bimanual Action: The Kinematic Chain as a Model." *The Journal of Motor Behavior* **19**(4): 486-517.

S. Halliday and M. Green (1996). "A Geometric Modeling and Animation System for Virtual Reality." *Communications of the ACM* **39**(5): 46-53.

G. Kurtenbach and W. Buxton (1993). "The limits of expert performance using hierarchic marking menus." *Proceedings of INTERCHI '93: Human Factors in Computing Systems* : p. xviii+547, 482-7.

J. Liang and M. Green (1994). "JDCAD: a highly interactive 3D modeling system." *Computers & Graphics* **18**(4): 499-506.

D. P. Mapes and J. M. Moshell (1995). "A Two-Handed Interface for Object Manipulation in Virtual Environments." *Presence* **4**(4): 403-416.

R. Pausch, T. Burnette, D. Brockway and M. E. Weiblen (1995). "Navigation and locomotion in virtual worlds via flight into hand-held miniatures." *Computer Graphics (Proceedings SIGGRAPH 95)* : p. 518, 399-400.

E. Sachs, A. Roberts and D. Stoops (1991). "3-Draw: a tool for designing 3D shapes." *IEEE Computer Graphics and Applications* **11**(6): 18-26.

C. Schmandt (1983). "Spatial input/display correspondence in a stereoscopic computer graphic work station." *Computer Graphics* **17,**(3): p. 253-61.

C. Shaw and M. Green (1994). "Two-handed polygonal surface design." *Proceedings of UIST `94: The Seventh Annual Symposium on User Interface Software and Technology* : p. 226, 205-12.

R. Stoakley, M. J. Conway and R. Pausch (1995). "Virtual Reality on a WIM: Interactive Worlds in Miniature." *Proceedings of CHI '95: the 1995 Conference on Human Factors in Computing Systems* : 265-272.