PART IV
Designing and Developing 3D User Interfaces

Thus far, we have focused on the low-level components of 3D UIs—input/output devices and interaction techniques. Through the guidelines presented in each chapter, we have shown you how to choose devices and techniques for your application that match its requirements and that will result in high levels of usability.

But how do you put all of these components together? What do complete 3D UIs look like? How do you verify that your system is easy to use and efficient? The answers to these questions are the focus of Part IV.

We recommend a *usability engineering* process (Gabbard et al. 1999) when constructing 3D UIs. This type of process begins with *requirements gathering*—an analysis of the existing situation, the problems users are having, the tasks users need to perform, and the characteristics of the users themselves. Next, you develop the *design* of the system and its UI, and build one or more *prototypes* that represent the system. Finally, you *evaluate* the prototype to find usability problems and to assess the quality of your design. In addition, usability engineering uses an *iterative* process, with multiple design-prototype-evaluate cycles.

In this part of the book, we address parts of this process that are *unique to 3D UIs*. (For a good overall introduction to usability engineering, we recommend books by Rosson and Carroll, 2001, or Hix and Hartson,

1993.) Chapter 10 deals with the *design* phase. It presents general design approaches and specific UI strategies that have been proven to work well in 3D UIs—these approaches and strategies can serve as the foundation for a 3D UI design. In Chapter 11, we look at the *evaluation* of 3D UIs, surveying the distinctive characteristics of 3D UI evaluation and various approaches to assessing usability.

We do not explicitly cover the requirements analysis phase, since 3D UI requirements analysis is very similar to generic requirements analysis processes. We also do not discuss the details of 3D UI prototyping or implementation. The current state of development tools for 3D UIs is extremely dynamic and uncertain—there are hosts of 3D modeling tools, programming languages, integrated development environments, toolkits, and libraries for 3D applications. We have chosen to keep our discussion on a high level and focus on 3D UI design, since any specific development information we might present could quickly become obsolete.

CHAPTER 10

Strategies in Designing and Developing 3D Interfaces

This chapter discusses some general strategies and principles for designing 3D UIs. Unlike the design of 3D interaction techniques (Part III), which is motivated by the requirements of particular tasks, the design strategies that we discuss in this chapter are high level and can be used in a wide variety of 3D tasks and applications. Some of these strategies are designed to match 3D UIs to the basic properties of human psychology and physiology; others are based on common sense, rules of thumb, or cultural metaphors.

10.1. Introduction

The previous chapters have focused on the basic building blocks common to many 3D UIs: input devices, displays, and interaction techniques for performing basic interaction tasks, such as manipulation and navigation. Although these techniques can be very useful in designing a variety of 3D interfaces, their simple mechanical combination does not necessarily guarantee an intuitive, easy-to-use, and enjoyable interactive experience.

On a microlevel, the devil is in the details, meaning that the effectiveness and usability of a 3D UI depends on the minute implementation de-

tails of interaction techniques, such as the careful choice of parameters and the match between the properties of interaction techniques and input/output devices.

On a macrolevel, there are many high-level, general design strategies driven not by the requirements of an interaction task but derived from more general principles, such as the strengths and limitations of human psychology and physiology, common sense, rules of thumb, cultural metaphors, and so on. For example, the basic principles for designing two-handed interaction techniques were developed independently of any interaction task. Rather, they were motivated by the simple observation that people naturally use two hands in real-world activities, so using two hands in a 3D UI might improve usability or performance. Similarly, many of the principles that we discuss in this section are general enough to be applicable to a wide range of interaction tasks.

The strategies and principles discussed in this chapter can be roughly divided into two large groups: *designing for humans* (i.e., strategies to match the design of interaction techniques and applications to human strengths, limitations, and individual differences), and *inventing 3D interaction techniques* (i.e., designing techniques based on commonsense approaches, creative exploration of 3D UI design, rules of thumb, etc.).

10.1.1. Designing for Humans

Many of the interface design principles from the human factors and general UI literature (Shneiderman 1998) can be applied to the design of 3D UIs. Indeed, reduction of short-term memory load, consistency of interface syntax and semantics, feedback, error prevention, and aesthetic appeal are as important in 3D interaction as in any other human–machine interface. Expanding to the third dimension, however, brings new challenges, such as designing for user comfort, that require different design strategies that are usually not explored in traditional UI literature. In addition, different strategies can be applied to UI design for different user groups, such as children, disabled individuals, or novices.

10.1.2. Inventing 3D User Interfaces

There are numerous application domains for 3D UIs, and even though the interaction techniques discussed earlier in this book can provide a starting point for UI design, it is not realistic to propose that they can cover all possible applications. Therefore, it's often necessary to invent

new 3D interaction techniques and design new interaction experiences. While human-factors-based principles can offer valuable insight into how 3D interfaces should be designed, they may not necessarily help designers to invent new and enjoyable interaction techniques. In this chapter therefore, we also survey some of the informal, rule-of-thumb approaches that have often been used in creating new 3D UIs; they can trigger a designer's imagination and provide a starting point for creating new and compelling 3D interaction experiences.

10.1.4. Chapter Roadmap

The chapter begins with a discussion of strategies and principles to match the design of 3D UIs with the human characteristics, including issues of feedback, constraints, two-handed interaction, user populations, and user comfort (section 10.2). Section 10.3 is focused on strategies for inventing 3D UIs, such as those that are based on replicating the real world, adapting techniques from 2D interaction, and using magic and aesthetics in 3D UI design. A final section of guidelines (section 10.4) summarizes some important ideas discussed in this chapter as well as some practical tips and techniques for designing interactive 3D applications.

10.2. Designing for Humans

All UIs must be designed in accordance with the most basic characteristics of human physiology and psychology, matching interface design to these characteristics. These *human factors* principles of interface design can be found throughout the literature, which can be very informative, though somewhat overwhelming. In this section, we do not attempt to discuss all of the human factors issues that might be relevant to 3D user interaction. Instead, we focus on some of the most basic topics that apply directly to 3D UIs. The recommended reading list at the end of the chapter provides some references for those who might want more information.

10.2.1. Feedback in 3D User Interfaces

Providing effective feedback is crucial in the design of any interface, whether it is a 3D VR system, a desktop 2D GUI, or a simple knob on a stereo system. Feedback refers to any information conveyed to the user that helps the user understand the state of the system, the results of an

operation, or the status of a task. This information may come from the system, from the environment, or from the user's own body.

It has long been understood that our ability to self-regulate body movements, such as manipulating objects or walking through the environment, is mediated by feedback-control mechanisms (Wiener 1948). In human–machine interaction, the user controls his movements by integrating various kinds of feedback provided by external sources (e.g., the UI) and self-produced by the human body (e.g., kinesthetic feedback). When interacting with a 3D UI, the user's physical input, such as hand movements, is captured by devices and translated into visual, auditory, and haptic feedback displayed to the user via display devices. At the same time, feedback is generated by the user's own body; this includes kinesthetic and proprioceptive feedback, which allow the user to "feel" the position and motion of his limbs and body.

Therefore, the goal of a 3D UI designer is to create an interactive system that provides sufficient levels of feedback to the user and ensures compliances (agreement) between different levels and types of feedback.

Multiple Dimensions in Feedback

Several different dimensions of feedback can be sensed by the user and provided by the interface. We consider two basic classifications of the feedback dimensions: sense-based and system-based.

The *sensory dimensions* of feedback include visual, auditory, tactile, and olfactory feedback from sources external to the user's body and proprioceptive and kinesthetic feedback generated by the user's body in response to limb and body movements. The 3D UI designer has direct control of the external visual feedback provided to the user. Given the appropriate devices the 3D UI can provide feedback to most other sensory feedback channels, except for the kinesthetic and proprioceptive senses. Providing compliant feedback to multiple sensory channels, such as combining haptic and visual feedback, improves user performance and satisfaction in 3D UIs.

Some types of sensory feedback are still difficult to provide effectively in 3D UIs. For example, force and tactile feedback devices are still bulky and difficult to use; consequently, VE applications are often criticized for the absence of haptic feedback. The idea of *sensory feedback substitution* has often been used in designing VE systems—visual or audio cues can be provided to compensate for missing haptic feedback. See Chapter 3 for further discussion of sensory feedback.

From a *system* point of view, feedback can be split into three categories: reactive, instrumental, and operational feedback (Smith and Smith 1987). *Reactive feedback* comes from operating interfaces and combines all the self-generated visual, auditory, haptic, and proprioceptive information that results from the user's actions. *Instrumental feedback* is the information concerning the actions and motions of the interface controls and tools sensed by the human user. Finally, *operational feedback* is feedback that the user receives from the system as the results of his or her actions. For example, when a user manipulates a 3D virtual object with a 6-DOF device, the user gets reactive feedback from moving the device with her arm (in the form of kinesthetic feedback), instrumental feedback from observing the movement of the 6-DOF device and feeling its shape, and operational feedback from observing the motion of the virtual object.

The boundaries of the categories in this classification are a bit fuzzy, but it is still important in analyzing the different techniques for providing feedback to the user and particularly in discussing the main principle in designing feedback—*compliance.*

Feedback Compliance in 3D UI

The key principle in designing effective feedback for interactive systems is the principle of compliance between different dimensions of the feedback provided to the user. It suggests that for efficient interaction, the 3D UI should *maintain spatial and temporal correspondence between multiple feedback dimensions* that the user receives. For the sensory dimensions, for example, if the visual feedback conflicts with the kinesthetic or proprioceptive feedback generated by the body, then user performance rapidly degrades (Smith and Smith 1987). This would happen, for example, if virtual objects moved in the opposite direction from the hand movement, a condition often called *feedback displacement.*

Because 3D UIs may tend to involve and engage users more than other UIs, a lack of compliance between the sensory dimensions may result not only in decreased performance, but also in much more dramatic effects such as headaches, blurred vision, dizziness, disorientation, or even severe vomiting (*cybersickness*). The most basic explanation for cybersickness is a conflict between the feedback from the visual sense and other intrinsic feedback systems, such as vestibular and perceptual (LaViola 2000a).

A number of specific compliances have been discussed in the human factors and 3D UI literature, and we discuss them in this section.

Spatial compliances. Spatial feedback compliances include directional compliance, also called stimulus-response (S-R) compatibility (Fitts and Jones 1953); nulling compliance; and others. *Directional compliance* suggests that a virtual object should move in the same direction as the manipulated input device. For example, when the user rotates a virtual object using a 6-DOF input device, both the device and virtual object should rotate in the same direction; that is, they should rotate around the same axis of rotation (Poupyrev, Weghorst et al. 2000). Directional compliance preserves the correspondence between the motions of virtual objects and the observed or felt motion of the physical input device. This allows the user to effectively anticipate the motion of the virtual objects in response to input and therefore plan and execute the desired trajectory of motion. Ensuring directional feedback compliance is important, although it has been shown that humans are extremely adaptable and can compensate for disparities between stimulus and response. However, a significant breakdown in directional feedback compliance might result in a decrease in user performance and even in simulation sickness such as in case of viewpoint control.

Another category of spatial compliances is *nulling compliance.* Nulling compliance means that when the user returns the device into the zero initial position or orientation, the virtual objects also return to the corresponding initial position or orientation (Buxton 1986). The importance of preserving nulling compliance depends on the application. It has been shown, for example, that if the device is attached to the user's body, the nulling compliance might be important, as it allows the user to use "muscle memory" to remember the initial, neutral position of the device and object.

On the system level it is also desirable that instrumental and operational feedbacks are spatially compliant: for example, a virtual hand should be aligned as closely as possible with the user's real hand.

Temporal compliances and latency. The most typical example of temporal *in*compliance is latency—the temporal delay between user input and sensory feedback generated by the system in response to it. The investigation of latency was particularly relevant in the early days of VEs, when it was a critical problem due to slower computers and computer graphics rendering hardware. A large number of user studies have investigated the negative effects of latency on performance (e.g., Ellis et al. 1999; Allison et al. 2001).

From the sensory-motor point of view, the reason that latency affects user performance is the incompliance between internal feedback (e.g., proprioceptive and kinesthetic feedback) and external sensory feedback received by the user (e.g., visual feedback from the system). This incompliance can significantly decrease user performance. For example, studies of flight simulators have shown that viewpoint control lags as short as 50 ms have a significant impact on user performance, while latencies that are much longer can lead to oscillations and loss of control (Wickens 1986).

Not only absolute latency but also its variability may affect user performance. Indeed, in most complex computer graphics applications, latency is not constant. Experimental studies have found that at 20 frames per second (fps) update rate, latency fluctuations as large as 40% do not significantly affect user performance on a manipulation task. At 17 fps, their effect becomes noticeable, and fluctuations become a significant problem at 10 fps, which is normally considered a lower bound for acceptable frame rate in 3D computer graphics applications (Watson et al. 1997).

With the rapid improvements in computer graphics hardware and software, the problem of latency may become less critical. On the other hand, as rendering performance increases, the requirements for visual realism and environment complexity also increase, so latency may remain an issue for some time to come.

The simplest technique for dealing with latency is to increase the update rate by reducing the environment's complexity, using more sophisticated culling algorithms, or rendering the scene progressively (simple rendering during interaction and high-quality rendering during breaks in interaction; (Airey et al. 1990). However, note that simply increasing the update rate does not eliminate all the sources of latency. For example, tracking devices have an inherent latency that is not tied to the update rate. Predictive filtering of the user input stream (e.g., tracking data) has also been investigated and has resulted in some success in reducing latency (Liang and Green 1991; Azuma and Bishop 1994).

Feedback Substitution

In designing and developing 3D UIs, it is often difficult to allow for all possible types of feedback. In particular, haptic feedback often requires devices that are expensive and difficult to operate. In the absence of haptic feedback, a feedback substitution principle has been often used. Instead of

Figure 10.1 *Feedback substitution: a visual cue substitutes for haptic feedback when a user touches a virtual object. (Reprinted with permission of HIT Lab, University of Washington)*

haptic feedback, additional audio or visual cues can be provided. For example, in a selection task, the action of touching the virtual object can be indicated with a visual highlight (e.g., drawing a frame around the object or changing its color (Figure 10.1). An enhancement of this technique, predictive highlighting (Butterworth et al. 1992), shows the user the 3D object that would likely be selected if the user continued the operation. This was found particularly useful in model editing mode, where the most probable vertex would highlight when the 3D cursor approached it.

Passive Haptic Feedback

Another method of providing simple haptic feedback is to match the shape and appearance of a virtual object with the shape and appearance of a physical object so that the user can both see and feel the object. This approach is called *passive haptic feedback,* or *props.* Passive feedback is a type of instrumental feedback: it provides users with a tactile sense of the virtual tools they are using. Although this approach is significantly less flexible than "real" force-feedback devices, it has been quite successful for real-world 3D UIs (see also Chapter 3 for relevant discussion).

One of the first uses of props in a 3D UI was a two-handed interface for interactive visualization of 3D neurological data (Hinckley et al. 1994). A 6-DOF magnetic tracker was embedded into in a doll's head (Figure 10.2), and by manipulating the toy head, the user was able to quickly and reliably relate the orientation of the input device to the orientation of

Figure 10.2 *Two-handed interface with passive haptic feedback provided by the doll head and cutting plane tool held by the user. (Hinckley et al. 1994, © 1994 IEEE)*

the volumetric brain data on the screen. This resulted in efficient and enjoyable interaction, because from the user perspective, the interaction was analogous to holding a miniature "real" head in one hand. The tactile properties of the prop allowed the user to know its orientation without looking at the device, so the focus of attention could be kept entirely on the task. While this interface was nonimmersive, in immersive VEs, passive props can be even more effective. By spatially registering tracked physical objects with virtual objects of the same shape, the designer can provide the user with inexpensive yet very realistic haptic feedback. Hoffman refers to this technique as *tactile augmentation* (Hoffman et al. 1998).

Using passive physical props is an extremely useful design technique for 3D UIs. Props provide inexpensive physical and tactile feedback, significantly increasing the sense of presence and ease of interaction in immersive and nonimmersive environments (Hoffman et al. 1998). They establish a common perceptual frame of reference between the device and the virtual objects, which in addition to ease of use may make it easier to learn the 3D UI (Hinckley 1994). The introduction of tactile augmentation also allows designers to explicitly control the realism of VEs, which can be useful in such applications as the treatment of phobias (Carlin et al. 1997).

There are also several disadvantages of using passive haptic feedback. First is the issue of scalability: using multiple physical props requires multiple trackers, which might be expensive and difficult to

implement. Second, experimental studies have not yet shown any quantitative improvement in user performance when using props (Hinckley et al. 1997; Ware and Rose 1999). However, the studies did show that users preferred physical props.

10.2.2. Constraints

In 3D UIs, constraints are usually very narrowly understood as relations between 3D virtual objects in a scene. The theory of constraints, however, is more general and does not necessarily consider the type of objects involved in constraint specification. Constraints are generally defined as a *relation* between variables that must be satisfied (Marriott and Stuckey 1998). Examples of such relations could be that a line should stay horizontal, that values in spreadsheet cells are always related through a formula, or that pressure in a closed volume should stay below a specified critical level. The objective of constraint theory is the development of algorithms that can find the values of variables satisfying the specified constraints.

For interactive computer graphics, constraints can be defined as relations that define some sort of *geometrical coherence* of the virtual scene during the user's interaction with it. While the implementation of constraints in interactive 3D computer graphics can use the theory and algorithms developed in constraint theory, from the user interaction point of view, the way constraints are *applied* is more important than the details of their implementation. The main reason that constraints are used in 3D UIs is that they can significantly simplify interaction while improving accuracy and user efficiency. Several types of constraints can be used in 3D UIs.

Physically realistic constraints are an often used application of constraints. One of the best examples of such constraints is collision detection and avoidance. When collision detection is enabled, the user's freedom of movement is constrained by the boundaries of the virtual objects—the user's virtual viewpoint or virtual hand is not allowed to pass through them. Another typical constraint is gravity—objects fall to the virtual ground when the user releases them. Physical constraints should be used with caution: in some applications, such as training or games, satisfying physical constraints is important because it is a critical part of the experience. However, in other applications, introducing such constraints may make interaction more difficult and frustrating. For example, in modeling applications, it might be convenient to leave objects "hanging in the air" when the user releases them (Smith 1987). The flexibility of

3D UIs allows us to selectively choose which physical properties of the physical environment are implemented in the virtual world, and the choice should be based on the requirements of the application.

Constraints also are used to *reduce the number of DOF of the user input* so as to make interaction simpler. For example, a virtual object can be constrained to move only on the surface of a virtual plane, which makes positioning it easier, since the user has to control only 2 DOF instead of 3 DOF. Such constrained manipulation has often been used in desktop 3D UIs to allow effective manipulation of 3D models using a mouse (e.g., Bier 1990). Another example is constraining travel to the ground or terrain of the virtual scene. This allows a user to effectively manipulate the viewpoint using only 2D input devices (e.g., Igarashi et al. 1998), and it helps users with 3D input devices to maintain spatial orientation.

Dynamic alignment tools, such as snap grids, guiding lines, and guiding surfaces, are a more complex way to reduce the required DOF. In this approach, the position and orientation of objects are automatically modified to align them with a particular guiding object, which can be as simple as a grid in space or as complex as a 3D surface (e.g., Bier 1986; 1990). With this type of constraint, objects "snap" to alignment with these guides. For example, objects can snap to an equally spaced 3D grid in space in order to make the alignment of several objects easier, or an object can automatically rotate so that it lies exactly on the guiding surface when the user brings it near to the guiding surface.

Intelligent constraints take into account the *semantics* of objects and attempts to constrain their interaction in order to enforce meaningful relations. For example, a virtual lamp can be constrained to stand only on horizontal surfaces such as tables, while a picture frame only "hangs" on vertical walls (Bukowski and Séquin 1995).

The disadvantage of all constraints is that they reduce user control over the interaction, which might not be appropriate for all applications. In many applications, therefore, it is important to allow the user to easily turn constraints on and off. However, when the user requirements are clearly understood and interaction flow carefully designed, constraints can be a very effective design tool for 3D UIs.

10.2.3. Two-Handed Control

Using both hands for 3D interaction allows users to transfer their everyday manipulation experiences and skills to interaction with 3D computer-generated environments. Furthermore, two-handed interaction can

significantly increase user performance on certain tasks both in real and virtual interaction. For example, it has been shown that a writing task using both hands resulted in 20% more efficient performance than when only one hand was used (Guiard 1987).

Therefore, two-handed or *bimanual* input has been an active topic of investigation in UIs since the early 1980s (Buxton and Myers 1986). The benefits of two-handed input have been demonstrated in various tasks and applications, including both 2D interfaces (Bier et al. 1993) and 3D interfaces (Sachs et al. 1991; Hinckley et al. 1994; Mapes and Moshell 1995; Zeleznik et al. 1997). In this section, we briefly overview some of the important principles that have been proposed for designing bimanual 3D UIs.

Guiard's Framework of Bimanual Manipulation

The theoretical foundation behind most of the current research on bimanual 3D interaction was proposed by Guiard, who studied the underlying mechanisms controlling the distribution of work between the dominant and nondominant hands of humans (Guiard 1987). He observed that some tasks are inherently *unimanual,* such as throwing darts. Other tasks are *bimanual symmetric,* where each hand performs identical actions either *synchronously,* such as pulling a rope or weightlifting, or *asynchronously,* such as milking a cow or typing on the keyboard. A third class of bimanual actions is *bimanual asymmetric* tasks (sometimes called *cooperative* manipulation) in which the actions of both hands are different but closely coordinated to accomplish the same task (Figure 10.3). A familiar

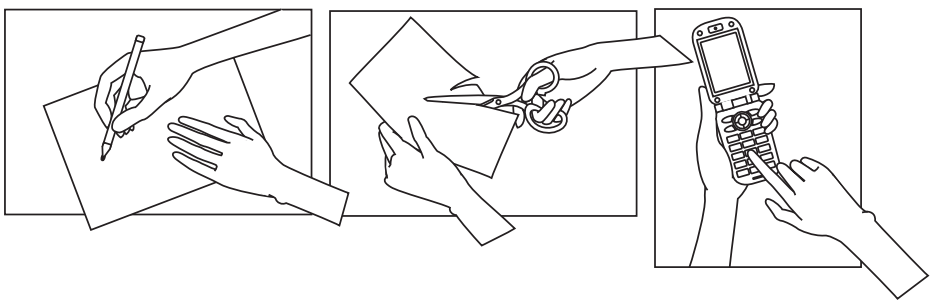

Figure 10.3 *Examples of the asymmetric separation of work between hands in bimanual manipulation: the nondominant hand defines a spatial framework for the preferred hand. (Illustrations by Keiko Nakao)*

example of such an asymmetric bimanual task is writing: the nondominant hand controls the orientation of the page for more convenient and efficient writing by the dominant hand.

Guiard proposed three principles that characterize the roles of the hands in tasks involving an asymmetric division of labor (Guiard 1987):

1. The nondominant hand dynamically adjusts the spatial frame of reference for the actions of the dominant hand.
2. The dominant hand produces fine-grained precision movements, while the nondominant hand performs gross manipulation.
3. The manipulation is initiated by the nondominant hand.

We should note, however, that the separation of labor between hands is a fluid, dynamic process, and in complex tasks, hands rapidly switch between symmetric and asymmetric manipulation modes.

Guiard's principles provide an important theoretical framework for investigating and designing two-handed 3D UIs. In the rest of this section, we discuss some examples of such interfaces, classifying them according to the asymmetric and symmetric division of labor.

Asymmetric Bimanual 3D Interaction Techniques

The bimanual 3D interface for neurological visualization that we've already seen in section 10.2.1 (Hinckley et al. 1994) was one of the earliest attempts to develop bimanual interaction techniques based on Guiard's principles. In Hinckley's interface, the nondominant hand controlled the orientation of the 3D volumetric neurosurgical data (using a doll's head prop), while the dominant hand controlled a virtual cutting plane that "sliced" the data and presented slices on the screen for analysis (Figure 10.2). According to the principles of asymmetric separation of labor, the nondominant hand controlled the gross position and orientation of a 3D virtual workspace, and the dominant hand performed fine operations in that workspace.

The basic method presented above has been explored in a number of 3D interaction systems and for a variety of tasks. A two-handed interface for a workbench display (Cutler et al. 1997), for example, provides tools for two-handed *object manipulation and rotation* using free space, 6-DOF input devices. The nondominant hand controls the position of a virtual object and the orientation of its rotation axis, while the dominant hand controls rotation around the axis. Zeleznik and colleagues (1997) propose

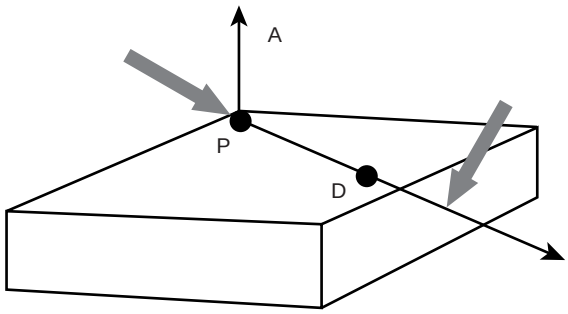

Figure 10.4 *Bimanual positioning and rotation of 3D objects: objects are constrained to move only on the plane. (Zeleznik et al. 1997, © 1997 ACM; reprinted by permission)*

a similar 3D position and rotation technique using two mice as input devices (Figure 10.4). With the nondominant hand, the user positions a point on a 2D ground plane, creating a vertical axis (left cursor in Figure 10.4), while the dominant hand allows the user to rotate an object around that axis (right cursor in Figure 10.4). Scaling and zooming were implemented in a similar way.

Virtual menus have often been implemented using bimanual manipulation in which the virtual menu is held in the nondominant hand while the dominant hand is used to select an item in the menu (Mapes and Moshell 1995; Cutler et al. 1997). Virtual writing techniques (Poupyrev, Tomokazu et al. 1998) use the nondominant hand to hold a tracked tablet input device, while the user writes on the tablet with the dominant hand. Similar two-handed, props-based interfaces are quite common (Coquillart and Wesche 1999; Schmalstieg et al. 1999). The Voodoo Dolls interaction technique that we discussed in Chapter 5 is yet another example of using asymmetric bimanual control for object manipulation (Pierce et al. 1999).

A somewhat different two-handed interface has been implemented for desktop 3D viewpoint control (Balakrishnan and Kurtenbach 1999). The nondominant hand controls the position of the camera, while the dominant hand performs application-specific tasks, such as selection, docking, and 3D painting.

A number of user studies have shown that carefully designed asymmetric bimanual interfaces have a significant performance advantage and are strongly preferred by users (Hinckley et al. 1997). A study by Balakrishan and Kurtenbach (1999), for example, found that in a selection task, user performance was 20% faster than a one-handed interface when

the user controlled the viewpoint using the nondominant hand and used the dominant hand to perform another task.

Symmetric Bimanual 3D Interaction Techniques

Symmetric two-handed manipulation has received somewhat less attention. A typical task that has been implemented using symmetric bimanual manipulation is *scaling*, where the user can scale objects by picking up two sides of the object and moving the hands apart or together simultaneously (Zeleznik et al. 1997). A bimanual *manipulation* technique implemented on a workbench display (Cutler et al. 1997) allowed the user to rotate the virtual scene with a "steering-wheel" gesture. Both of these techniques use synchronous interaction.

Asynchronous, symmetric, two-handed manipulation can also be implemented for interaction with 3D environments. For example, the Polyshop system implemented a rope-pulling gesture for 3D travel (see Chapter 6, section 6.3.6): the user pulled himself through the environment by pulling on an invisible rope with both hands (Mapes and Moshell 1995).

10.2.4. Designing for Different User Groups

An important part of 3D UI design is to determine the characteristics of the target user population. These user characteristics can have a significant effect on the design of a usable 3D UI. Here, we briefly overview some of the key user characteristics and their influence on interface design.

Age

The methods of both information presentation and interaction depend on the age of the user. Children often require a different interface design than adults because they are physically smaller, they have a shorter attention span, and the mental model they form about the interaction differs from that of adults. Older users may need text presented in a larger font size, or they may not be able to make movements quickly. More research is needed to determine the effects of age on the usability and performance of 3D UIs.

Prior Experience with 3D UIs

Targeting a user population that is already proficient with 3D input and output devices allows us to design more complex 3D UIs. Also,

observations indicate that children's experience with console games or desktop 3D computer games can be positively correlated with performance in 3D UIs. On the other hand, 3D UIs for novice users need to be simplified and highly learnable.

Physical Characteristics

Simple things like a user's height can affect the usability of a 3D UI. For example, a basic implementation of the Go-Go technique (see Chapter 5) has a fixed threshold at which the nonlinear arm extension begins. Users with shorter arm lengths, therefore, will not be able to reach nearly as far into the environment as other users. In this case, an adaptive threshold based on arm length would be appropriate. The user's handedness (i.e., the dominant hand) is another example. Many input devices are designed only for right-handers. Asymmetric, bimanual interfaces must be adaptable for both left- and right-handed users.

Perceptual, Cognitive, and Motor Abilities

People's color recognition and stereo vision abilities are different, and this of course affects the choice of a display system. A specific cognitive characteristic affecting 3D UI design is the user's spatial ability (ability to think and plan actions in a 3D space). If the user population is known to have lower than average spatial abilities, a simplified interface for 3D travel and manipulation may be in order (e.g., the use of additional constraints). People with cognitive or motor disabilities may require a simplified interface as well as the use of special-purpose devices.

10.2.5. Designing for User Comfort

In the design of 2D UIs, a great deal of effort is spent in designing a system that is understandable, intuitive, and well organized, but very rarely do designers have to consider the physical actions users will be performing (although issues such as repetitive stress injuries have made these actions more prominent). In 3D UIs, however, users are often interacting while standing, while wearing or holding heavy or bulky devices, and while moving the whole body. Thus, issues of user comfort and safety are extremely relevant. We offer several guidelines for designing comfortable 3D UIs, with a particular emphasis on *public installations* as an example of 3D UIs for real-world use.

> Move wires and cables out of the way; reduce weight of the equipment.

HMDs, trackers, and other input/output devices are typically connected to the host computer or to an interface box. One of the most common complaints of VE users is that wires and cables get in the way during system use. If the wires are on or near the floor, users can trip over them or get them wrapped around their legs when they turn. Hanging wires can get in the way during arm movements. All of these situations are annoying, can interrupt the user's interaction, and can reduce the sense of presence. Especially for public systems, it is important to find a cable-management solution, such as hanging cables from the ceiling, to minimize these problems. Care should be taken to leave enough length in the cables to allow users free physical movement.

Hanging wires from the ceiling can also reduce the weight of the equipment worn by the user. Weight is an important problem, especially in immersive VE installations; many 3D input and display devices are heavy. HMDs are particularly notorious for their weight—even short periods of use can result in user fatigue. In mobile AR systems, the user must also carry the computer and other peripherals (such as a GPS receiver), so total weight may be even more important. Every effort should be made to reduce the overall weight of the equipment worn by the user.

> Provide physical barriers to keep the user and the equipment safe.

With HMD-based systems, the user cannot see the physical world and thus is prone to walk into walls, chairs, tables, or other physical objects. In surround-screen systems, the screens seem to disappear when the virtual world is projected on them, so users tend to walk into the screens, which may damage them. The most common approach to addressing these issues is to establish a "safe zone" around the user by making a physical barrier, such as a railing, that the user can hold on to. The safe zone should be large enough to allow sufficient physical movement, but small enough to keep the user away from any potentially dangerous areas.

> Limit interaction in free space; provide a device resting place.

The interface design should limit free-space interaction in which the user hand is not physically supported. For example, the image-plane interaction technique (Chapter 5, section 5.4.2) requires the user to hold her hand in free space to select an object, which over a long period of time is very tiresome. Hence, interaction sequences should be designed so that free-space interaction occurs in short chunks of time with resting periods in between. The user should be allowed to rest her hands or arms without breaking the flow of interaction. Also, consider providing a resting place for a device if it is not attached to the user—this could be a stand, a hook, a tool belt, or some other method for putting a device away when it is not needed.

Design public systems to be sanitary.

When a large number of users wear, hold, or touch the same devices, hygiene and health issues are important (Stanney et al. 1998). The designer might consider a routine schedule for cleaning the devices. Another approach is to use removable parts that can be disposed of after a single use, such as thin glove liners for glove-based input devices or HMD liners—light plastic disposable caps that users put on before using the HMD.

Design for relatively short sessions.

Cybersickness and fatigue can be a significant problem when using immersive VE systems. Even though graphics, optics, and tracking have greatly improved, many users may still feel some symptoms of sickness or fatigue after 30 to 45 minutes of use. In public VR installations, the time that the user spends in the system can be explicitly limited (Davies and Harrison 1996). In other applications, the interface designer can mitigate these problems by allowing the work to be done in short blocks of time.

10.3. Inventing 3D Interfaces

This section surveys some of the informal approaches that have often been used in creating 3D UIs and interaction techniques. These approaches lie in a continuum between strict *imitations of reality* (naturalism

or isomorphism) and *magic* (nonisomorphism, things that can't be found in the real world). The approaches presented here should be taken as illustrative examples to help designers and developers to ask the "right questions" that will lead to the development of compelling 3D UIs. Of course, the process of creating something new is difficult to formalize and explain; that is perhaps the most magical, or artistic, part of designing 3D UIs.

10.3.1. Borrowing from the Real World

The most basic, tried-and-true approach is to attempt to simulate, or adapt from, the real, physical world. In some cases the goal is to replicate the real world as closely as possible; in other cases, only elements of the real world are brought into the VE and creatively adapted to the needs of 3D interaction. In this section, we discuss some of the basic approaches and techniques that have been reported in the literature.

Simulating Reality

Simulating reality is key in all simulation applications, such as flight simulators, medical training, treatment of phobias, some entertainment applications, and human factors evaluations of human-controlled mechanisms such as vehicles (e.g., Loftin and Kenney 1995; Carlin et al. 1997; Burdea et al. 1998; Cruz-Neira and Lutz 1999).

The advantage of using this approach is that the user already knows how to use the interface from everyday experience, so the time spent learning how to use the system can be kept to a minimum. Furthermore, the interface often can be implemented based either on the designer's intuition and common sense or on the clearly specified technical design requirements of the application. Advanced and special-purpose 3D interaction techniques, such as some of those presented earlier in this book, might not be necessary in such applications. Interaction with virtual space in this type of interface might be frustrating and difficult, but if the real-world interaction is also frustrating and difficult, then this is actually a feature!

Designers may, however, need to compromise on how realistic the simulation needs to be. Due to the limitations of current technology, the simulations that we can create are either far from real-life experiences or prohibitively expensive (e.g., professional flight simulators). The realism of simulation that the developer should aim for thus depends heavily on

Figure 10.5 *Immersive entertainment environment simulating a dinosaur habitat focuses on the fluidity of interaction, but not on visual realism. (Reprinted with permission of HIT Lab, University of Washington)*

the requirements of the application. The virtual interaction should attempt to reproduce real-world interaction in those details that are essential for the application.

For example, in a VE for entertainment, the goal might be to provide visitors with a first-person immersive experience in an environment that cannot be normally experienced in the real world (Figure 10.5). Exact visual realism might be less important than responsiveness and ease of learning for this application, and thus interaction can be limited to a very simple flying technique that allows the user to fully experience the environment. As another example, in a medical training simulator designed to teach medical students palpation skills in diagnosing prostate cancer, a realistic visual simulation is not required at all, so only a primitive 3D visual model is needed. A realistic and precise simulation of haptic sensation, however, is a key issue, since learning the "feel" of this technique is the main goal of the system (Burdea et al. 1998). On the other hand, in a system developed for treatment of spider phobia, a simple toy spider was used to provide the patients with a passive haptic sensation of the virtual spider that they observed in the VE (Carlin et al. 1997). It was found that such passive tactile feedback combined with the graphical representation of spider was realistic enough to trigger a strong reaction from the patient.

These examples demonstrate that the importance of realism is dependent on the application; 3D UIs should deliver only as much realism as needed for the application. The effect of realism, or level of detail, on user

performance, learning, and training transfer is an important topic of research but is outside the scope of this book. For further reading on this subject see Luebke and colleagues (2002).

Adapting from the Real World

Instead of attempting to replicate the real world, 3D UIs can also adapt artifacts, ideas, and philosophy from the real world.

The use of *real-world metaphors*—adapting everyday or special-purpose tools as metaphors for 3D interaction—has been a very effective technique for the design of 3D widgets and interaction techniques. For example, a virtual vehicle metaphor has been one of the most often used metaphors for 3D navigation. A virtual flashlight has been used to set viewpoint or lighting directions, and shadows have been used not only to add realism to the rendering, but also to provide simple interaction techniques—the user could manipulate objects in a 3D environment by dragging their shadows (Figure 10.6; Herndon et al. 1992).

As these examples show, the metaphor is only a starting point, and interaction techniques based on the metaphors should be carefully designed to match the requirements of the applications and limitations of the technology used. For example, the ECG monitor widget used in the VR emergency room project (Kaufman et al. 1997) had to be larger and less detailed than a real ECG monitor because of the lower resolution of

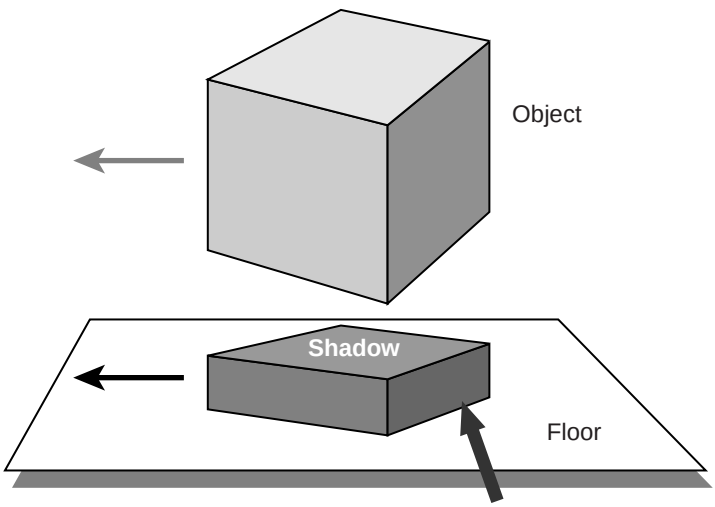

Figure 10.6 *Shadows can be used for constrained 3D manipulation (Herndon et al. 1992, © 1992 ACM; reprinted by permission)*

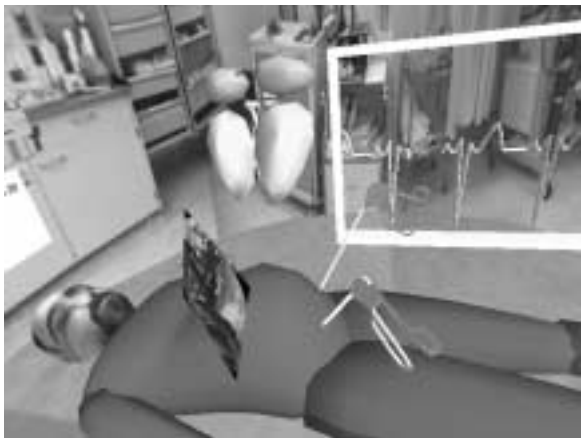

Figure 10.7 *ECG monitor widget designed for VR experience. (Reprinted with permission of HIT Lab, University of Washington)*

VE displays. At the same time, it was made transparent so that it did not occlude large parts of the VE (Figure 10.7).

Not only single tools, objects, and their features, but also entire domains of human activity can inspire and guide the design of 3D UIs. For example, *architecture* and *movies* have been an important source of inspiration. The objective is not simply to replicate, but instead to creatively adapt the basic principles and ideas in designing 3D UIs and virtual spaces. For example, games have been heavily influenced by the movie culture, exploring transition and storytelling techniques fundamental to film. It has also been observed that both architecture and virtual worlds need certain principles to order and organize shapes and forms in space (Alexander et al. 1977); the design principles from architecture therefore can be transferred to the design of 3D UIs.

Campbell (1996), for example, attempted to design VR spaces using basic architectural principles (Figure 10.8), suggesting that this would allow users to rely on their familiarity with real-world architectural spaces in helping them to quickly understand and navigate through 3D virtual spaces. At the same time, designers of virtual spaces are not limited by the fundamental physical restrictions that are encountered in traditional architecture, giving them new creative freedom in designing 3D interactive spaces. See the discussion of architectural wayfinding cues in Chapter 7 for more on this topic.

Another common technique for adapting elements from the real world to the design of a VE is to borrow natural physical gestures that we

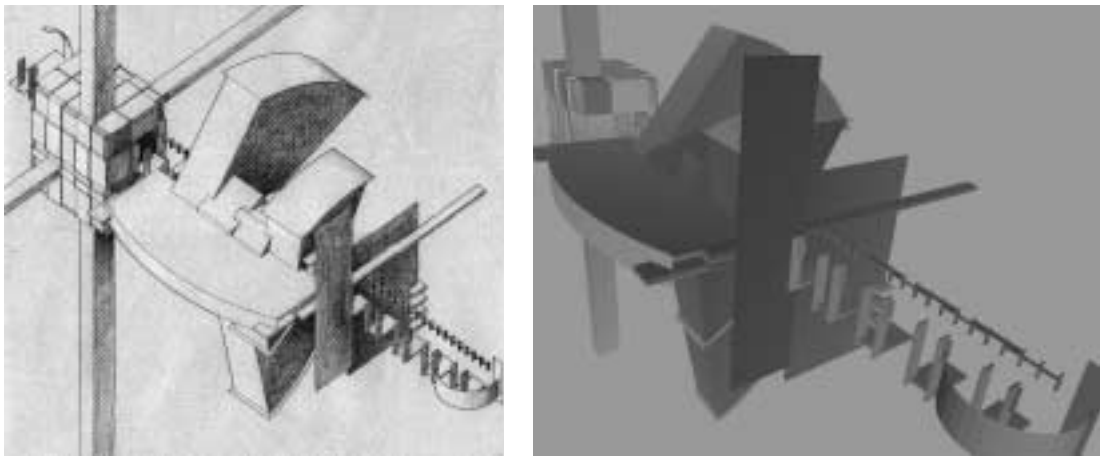

Figure 10.8 *Virtual architecture design from architectural sketch (left) to VE (right). (Campbell 1996; reprinted by permission of the author)*

use in real life. For example, in the Osmose interactive VE, the user navigated by using breathing and balance control, a technique inspired by the scuba diving technique of buoyancy control (Davies and Harrison 1996). The user was able to float upward by breathing in, to fall by breathing out, and to change direction by altering the body's center of balance. The intention was to create an illusion of floating rather than flying or driving in the environment.

Numerous real-world artifacts and concepts can be used in designing 3D UIs, providing an unlimited source of ideas for the creativity of designers and developers. Because users are already familiar with real-world artifacts, it is easy for them to understand the purpose and method of using 3D interaction techniques based on them. Metaphors, however, are never complete, and an important goal is to creatively adapt and change the real-world artifacts and interactions. It is also difficult to find real-world analogies and metaphors for abstract operations. For example, in the dVise VR authoring system, an "egg" widget was used to create new objects—a metaphor that is not very easy to grasp without prior explanation. In such cases, a symbolic representation might be much more appropriate.

10.3.2. Adapting from 2D User Interfaces

Adapting interaction techniques from traditional 2D UIs has been another common 3D UI design technique. Two-dimensional interaction

techniques have many attractive properties. First, 2D UIs and interaction have been thoroughly studied, and interaction paradigms in 2D interfaces are well established, which makes it relatively easy for 3D interface designers to find an appropriate interaction technique. Second, users of 3D UIs are already fluent with 2D interaction, so learning can be kept to a minimum. Third, interaction in two dimensions is significantly easier than interaction in three dimensions—the user has to manipulate only 2 DOF rather than 6 DOF. Consequently, 2D interaction may allow users to perform some tasks, such as selection and manipulation, with a higher degree of precision. Finally, some common tasks that the user may need to perform do not scale well into three dimensions. For example, writing and sketching is significantly easier to perform in 2D than in 3D. These considerations have prompted researchers to design 3D UIs that attempt to take advantage of both 2D and 3D interaction techniques, trying to combine these two input styles in a seamless and intuitive manner (see Chapter 8 for the use of 2D interaction for system control).

Literal Approach: Overlaying a 2D GUI on a 3D World

With little or no modification, 2D UI elements can be embedded directly into a 3D environment. Certainly, for desktop-based 3D applications, this is a natural technique—the 3D scene is rendered in a window, and traditional 2D GUI elements (menus, sliders, etc.) can be attached to this window outside of the 3D environment. A similar approach has been used quite often in immersive VEs, particularly for system control tasks and when interacting with inherently 2D information. Figure 10.9 presents an example of one such system, which simply provides familiar cascading 2D menus overlaid on the 3D world (Bolter et al. 1995).

The shortcoming with this approach is that the GUI interaction elements are introduced as a separate layer on top of the 3D world, so it introduces an additional mode of interaction: the user has to switch into the menu mode and then switch back to 3D UI mode. The approach also does not scale well to other 2D tasks.

2D GUI as an Element of the 3D Environment

An alternative way to place a 2D GUI into a 3D environment is to render the interface as a first-class object in the 3D world. For example, menus and other interface elements can be rendered on some planar surface within the VR, either as 3D buttons and menus arranged on a plane or as

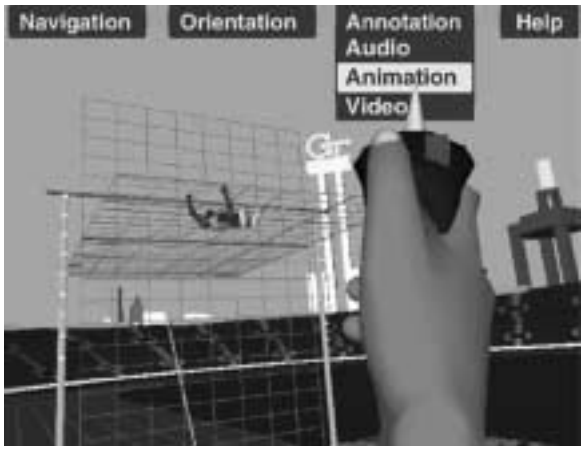

Figure 10.9 *Overlaid 2D interface elements in a 3D VE. (Bolter et al. 1995, © 1995 IEEE).*

a dynamic 2D texture attached to a polygon (Angus and Sowizral 1996). The user can interact with these 2D UI elements in the same way we interact with them in desktop environments—by touching and dragging them on a 2D virtual surface using virtual hand or ray-casting interaction techniques (e.g., Mine 1995b). The difficulty with this approach is that there is no haptic feedback, so interacting with the 2D interface might be difficult and frustrating. To overcome this problem, a physical prop—such as a wooden clipboard—can be tracked and registered with the 2D interface so that it appears on top of the clipboard. The user, holding the physical clipboard in one hand, can touch and interact with 2D interface elements using a pen held in the other hand, which is also tracked. This design technique is sometimes called the *pen-and-tablet* technique.

One of the first systems that used this approach was implemented by Angus and Sowizral (1996); it provided the immersed user (who was inspecting a virtual model of an aircraft) with a hyperlinked document that included 2D plans, drawings, and other information (Figure 10.10). A similar technique was developed for the semi-immersive workbench environment by (Coquillart and Wesche 1999), where instead of using a wooden pad, 2D data was spatially registered with a tracked, transparent plastic pad. When the user looked through the pad, he perceived the illusion that the 2D information appeared on the pad (see also transparent prop discussion below).

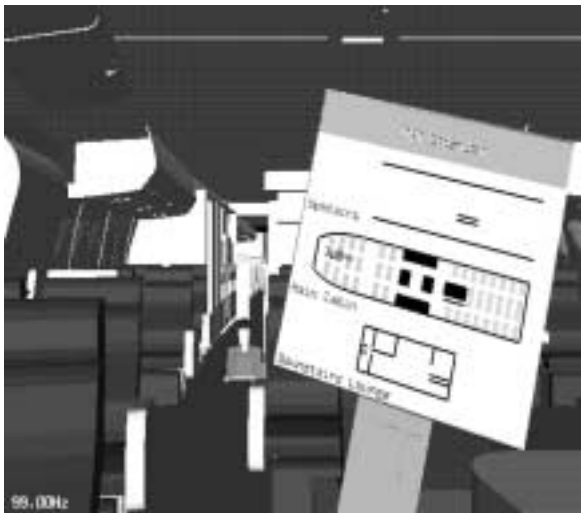

Figure 10.10 *Embedding an interactive 2D interface into a 3D VE. (Angus and Sowizral 1996, © 1996 IEEE)*

The pen and tablet technique can be extended by using a touch-sensitive tablet instead of a passive prop. With this active prop, the user can perform significantly more advanced interaction than simply pressing 2D buttons. Virtual Notepad, for example, allows users not only to view 2D information while immersed in a VE, but also to annotate it with 2D drawings (Figure 10.11; Poupyrev, Tomokazu et al. 1998).

Figure 10.11 *The Virtual Notepad. (Poupyrev, Tomokazu et al. 1998, © 1998 IEEE)*

You have probably noticed that the pen and tablet idea has been introduced several times in the book already when we talked about different tasks. That's because it's a rather generic approach in designing 3D UIs that also incorporates many of the principles and strategies we have discussed before, including

- two-handed, asymmetric interaction
- physical props (passive haptic feedback)
- 2D interaction, reducing the DOF of input
- a surface constraint to aid input
- body-referenced interaction

2D Interaction with 3D Objects

It has been often noted that 3D interaction is difficult: the coordinated control of 6 DOF requires significantly more effort than manipulation of only 2 DOF. Reducing the number of degrees of control is especially crucial when high-precision interaction is needed, such as when creating 3D models or performing virtual surgery. Using constraints is one technique to reduce the number of controlled DOF and simplify interaction. A particular instance of this technique that has been successful is based on constraining input with physical 2D surfaces and using gesture-based interaction.

Schmalstieg and his coauthors (1999), for example, developed an effective 2D gestural interface for 3D interaction by using tracked, transparent, passive props with a workbench display (Figure 10.12). The user looks at the 3D environment through the transparent prop (e.g., a simple, transparent Plexiglas plate) and interacts with objects by drawing 2D gestures on the prop. The transparent plate acts as a physical constraint for the user input, making drawing relatively easy.

The system determines the objects that the user interacts with by casting a ray from the user's viewpoint through the pen and transparent pad. For example, **Error! Reference source not found.** demonstrates how a group of 3D objects can be selected by drawing a lasso around them on the prop.

Transparent props allow the development of very generic techniques that can be used to interact with any 3D objects in the scene. Furthermore, when the position of objects can be constrained to lie on a virtual ground plane, the user can draw 2D gestures directly on the surface of the

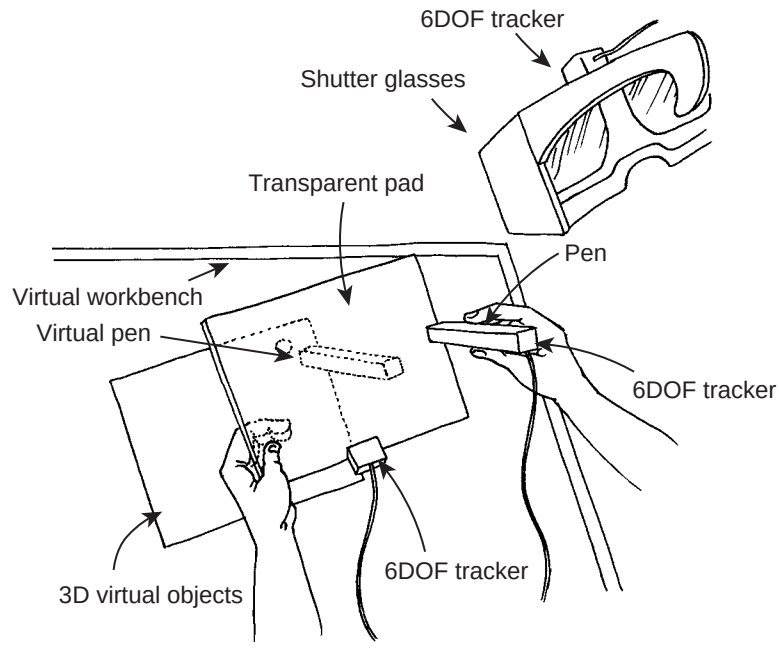

Figure 10.12 *Using transparent props for 2D interaction in a VE.*

display, as long as the pen-tracking technology is provided. For example, in Ergodesk, the user interacts in 2D by drawing directly on the display surface (Figure 10.13), sketching 3D models using a three-button stylus (Forsberg et al. 1997). The system is based on the SKETCH modeling system (Zeleznik et al. 1996) that interprets lines as operations and parameters for 3D modeling commands. Creating a cube, for example, requires the user to draw three gesture lines, one for each of the principal axes, meeting at a single point.

The sketching interface is simple to learn and allows the user to easily create 3D objects by sketching them on a 2D physical surface. This is significantly easier then manipulating or sculpting 3D objects directly using all 6-DOF of input. Furthermore, the user can use another hand for performing traditional 3D input tasks, such as navigating or manipulating 3D objects.

10.3.3. Magic and Aesthetics

It has long been argued that the real power of 3D UIs lies not in just simulating or adapting real-world features, but in creating a "better" reality by

Figure 10.13 *Sketching 3D objects directly on the display surface in Ergodesk. (Forsberg et al. 1997, © 1997 IEEE)*

utilizing magical interaction techniques (e.g., Smith 1987; Stoakley et al. 1995; Shneiderman 2003). One advantage of magical interaction techniques is that they allow users to overcome many human limitations that are so prominent in the real world: limitations of our cognitive, perceptual, physical, and motor capabilities. The second advantage of this approach is that it reduces the effect of technological limitations by compensating for them with enhanced capabilities of the UI. In fact, most of the interaction techniques discussed in Part III of this book are magic techniques to some degree. For example, Go-Go (Chapter 5, Section 5.4.3) and flying techniques (Chapter 6, section 6.3.3) enhance our motor capabilities by allowing us to reach further and travel in ways we can't in the real world; the world-in-miniature technique (Chapter 5, section 5.4.4) extends our perceptual capabilities by allowing us to see the entire 3D environment at once; and many system control techniques enhance our cognitive capabilities by visually presenting available choices rather then forcing us to remember them.

There are many approaches that can help to develop new magical techniques. Considering human limitations and looking for solutions that help to overcome them is one possible approach. Cultural clichés and metaphors, such as a flying carpet, can also suggest interesting possibilities for designing magical 3D UIs. For example, the Voodoo Dolls technique (Chapter 5) is based on a magical metaphor that suggests that

the user can affect remote objects by interacting with their miniature, toy-like representations. Because such metaphors are rooted in the popular culture, users may be able to grasp interaction concepts almost immediately. The relationships between techniques and cultural metaphors, however, can also be a source of difficulties. For example, in order to understand the metaphor of flying carpet techniques—a very popular metaphor in VR (Butterworth et al. 1992; Pausch et al. 1996)—the user needs to know what a magic carpet is and what it can do. While some metaphors are quite universal, others might be significantly more obscure. For example, the Virtual Tricorder metaphor (Wloka and Greenfield 1995) is based on imaginary devices from the *Star Trek* television series, which may not be well known by users outside of countries where this show is popular. It is not easy to find effective and compelling metaphors for magical interaction techniques; however, the right metaphor can lead to a very enjoyable and effective 3D UI.

The discussion of realism and magic in designing 3D UIs also directly relates to the *aesthetics* of the 3D environment. The traditional focus of interactive 3D computer graphics, strongly influenced by the film and simulation industries, was to strive for photorealistic rendering—attempting to *explicitly* reproduce physical reality. While this approach is important in specific applications, such as simulation and training, photorealism may be neither necessary nor effective in many other 3D applications. In fact, modern computer graphics are still not powerful enough to reproduce reality so that it is indistinguishable from the real world. Furthermore, humans are amazingly skilled at distinguishing real from fake, particularly when it comes to humans: that is why faces rendered with computer graphics often look artificial and therefore unpleasant.

In many applications, however, photorealism may not be entirely necessary: sketchy, cartoon-like rendering can be effective and compelling in communicating the state of the interaction, while at the same time being enjoyable and effective. One example of such an application is a 3D learning environment for children (Johnson et al. 1998).

Nonphotorealistic cartoon rendering can be particularly effective in drawing humans, because they suggest similarities using a few strong visual cues while omitting many less relevant visual features. We can observe this effect in political cartoons: even when drawings are grossly distorted, they usually have striking similarities with the subjects of the cartoons, making them immediately recognizable. Simple, cartoonlike rendering of virtual humans is also effective from a system point of view,

Figure10.14 *Cartoon drawing used in 3D online communities. (Reprinted here with permission of There, Inc.)*

since simpler 3D models result in faster rendering. Thus, more computational power can be dedicated to other aspects of human simulation, such as realistic motion. This allows designers to create effective and enjoyable interfaces for such applications as online 3D communities and chat environments (Figure 10.14).

Another advantage of nonphotorealistic aesthetics in 3D interfaces is the ability to create a mood or atmosphere in the 3D environment as well as to suggest properties of the environment rather then render them explicitly. For example, a rough pencil-style rendering of an architectural model can inform the client that the project is still unfinished (Klein et al. 2000). The possibility to create mood and atmosphere is key in entertainment applications as well as in media art installations. One of the systems that pioneered this broader sense of aesthetics in VR was Osmose, a VE designed by Char Davies (Davies and Harrison 1996). The aesthetics of Osmose was neither abstract nor photorealistic, but somewhere in between. Implemented using multilayer transparent imagery and heavy

Figure 10.15 *Vertical tree. Digital frame captured in real-time through head-mounted display during live performance of immersive virtual environment OSMOSE (1995). (Char Davies)*

use of particle animation, Osmose provided the user with the impression of being surrounded by the VE, creating a strong sense of being there (i.e., sense of presence; Figure 10.15).

The aesthetic aspect of 3D UI design is an interesting and very important part of compelling, immersive, and easy-to-use interfaces. As the rendering power of computer graphics increases, and therefore the range of tools available to artists and designers broadens, the importance of aesthetics in 3D interaction will continue to grow in the future.

10.4. Design Guidelines

We conclude the chapter with a number of specific design guidelines, some of which summarize our previous discussion and others that offer helpful rules of thumb for designing and developing 3D UIs.

> Ensure temporal and spatial compliance between feedback dimensions.

It is crucial to ensure the spatial and temporal correspondence between reactive, instrumental, and operational feedback dimensions. In particular, interfaces must be designed to ensure directional compliance between a user's input and the feedback she receives from the system. Reduce latency and use multisensory feedback when appropriate, such as auditory and haptic feedback in combination with visuals.

> Use constraints.

Using constraints can greatly increase the speed and accuracy of interaction in many interaction tasks that rely on continuous motor control, such as object manipulation and navigation. However, also consider including functionality that allows the user to switch constraints off when more freedom of input is needed.

> Consider using props and passive feedback, particularly in highly specialized tasks.

Using props and passive haptic feedback is an easy and inexpensive way to design a more enjoyable and effective interaction. The drawback of props is that they are highly specialized—the physical shape of props cannot change—therefore, they are particularly effective in very specialized applications.

> Use Guiard's principles in designing two-handed interfaces.

Guiard's framework has proven to be a very useful tool for designing effective two-handed interfaces. Use these principles to determine the functions that should be assigned to each hand.

> Consider real-world tools and practices as a source of inspiration for 3D UI design.

Adapting everyday or special-purpose tools as metaphors has been a very effective method for designing 3D interaction techniques. Because users are often already familiar with real-world artifacts, it is easy for them to understand the purpose and method of using 3D interaction techniques based on them.

> Consider designing 3D techniques using principles from 2D interaction.

Two-dimensional interaction techniques have many attractive properties: they have been thoroughly studied and are well established, most users are already fluent with 2D interaction, and interaction in two dimensions is significantly easier than interaction in a 3D environment.

> Use and invent magical techniques.

The real power of 3D UIs is in creating a "better" reality through the use of magical interaction techniques. Magical interaction allows us to overcome the limitations of our cognitive, perceptual, physical, and motor capabilities. It is not easy to find effective and compelling metaphors for magical interaction techniques, but if one is found, it can lead to a very enjoyable and effective 3D UI.

> Consider alternatives to photorealistic aesthetics.

In many applications, photorealism may not be entirely necessary: sketchy, cartoonlike rendering can be effective and compelling in communicating the state of the interaction while at the same time being enjoyable. Nonphotorealistic cartoon rendering suggests similarities using

a few strong visual cues, which results in simple and more effective rendering. Another advantage of nonphotorealistic aesthetics in 3D interfaces is the ability to create a mood or atmosphere as well as to suggest properties of the environment rather then render them explicitly.

Recommended Reading

For an excellent review of the basics in feedback-control mechanisms and a discussion of early psychological and human factors experiments in this area, consider the following chapter:

> Smith, T., and K. Smith (1987). Feedback-Control Mechanisms of Human Behavior. *Handbook of Human Factors.* G. Salvendy (Ed.), John Wiley & Sons, 251–293.

For an in-depth discussion of some of the aspects of realism and level of detail and their effect on user performance, see the following book:

> Luebke, D., M. Reddy, J. Cohen, A. Varshney, B. Watson, and R. Huebner (2002). *Level of Detail for 3D Graphics,* Morgan Kaufmann.

Two classic texts on architectural design and philosophy can inspire you to develop new interaction techniques and approaches for planning and developing VEs:

> Lynch, K. (1960). *The Image of the City,* MIT Press.
>
> Alexander, C., S. Ishikawa, and M. Silverstein (1977). *A Pattern Language: Towns, Buildings, Construction,* Oxford University Press.