Performance-oriented Interaction Techniques

Doug A. Bowman
Department of Computer Science
Virginia Polytechnic Institute & State University

1 Introduction & Background

Interaction in a three-dimensional virtual environment can be extremely complex. Users must often control six degrees of freedom (DOFs) simultaneously, move in three dimensions, and give a wide array of commands to the system. To make matters worse, the standard and familiar input devices such as mice and keyboards are usually not present, especially in immersive VEs.

Meanwhile, VE applications are themselves becoming increasingly complicated. Once a technology only for interactively simple systems such as architectural walkthrough (Brooks, 1992) or phobia treatment (Hodges et al, 1995), VEs are now proposed for use in domains such as manufacturing, design, medicine, and education. All of these domains will require a much more active user, and therefore a more complex user interface (UI).

One of the main concerns with the advent of these complex applications is performance, defined broadly. In this chapter, we will consider two aspects of performance: task performance and technique performance with human effects. Task performance refers to the quality of task completion, such as time for completion or accuracy. It is usually measured quantitatively. Technique performance refers to the qualitative experience of the user during interaction, including ease of use, ease of learning, and user comfort. This is often called usability.

Therefore, in this chapter we will consider the design of interaction techniques that maximize performance, and the use of such techniques in interactively complex VE applications. This is an extremely important topic. Since VEs support human tasks, it is essential that VE developers show concern for human performance issues when selecting interaction techniques and metaphors for their systems. Until recently, however, most VE interaction design was done in an ad hoc fashion, because little was known about the performance characteristics of VE interaction techniques. This chapter will present a wide range of techniques, along with design guidelines based on performance evaluations from new research.

1.1 Methodology

Many of the results in this chapter stem from the use of a particular methodology (Bowman & Hodges, 1999) for the design, evaluation, and application of interaction techniques, with the goal of optimizing performance. In order to understand the context in which the guidelines presented here were developed, we will briefly discuss the parts of this methodology relating to design.

Principled, systematic design and evaluation frameworks (e.g. Price, Baecker, & Small, 1993) give formalism and structure to research on interaction, rather than relying solely on experience and intuition. Formal frameworks provide us not only with a greater understanding of the advantages and disadvantages of current techniques, but also with better opportunities to create robust and well-performing new techniques, based on the knowledge gained through evaluation. Therefore, we follow several important design and evaluation concepts, elucidated in the following sections.

1.1.1 Initial evaluation

The first step towards formalizing the design of interaction techniques is to gain an intuitive understanding of the tasks and current techniques available for the tasks. This is accomplished through experience using ITs and through observation and evaluation of groups of users. Often in this phase we perform informal user studies or usability tests, asking users what they think of a particular technique, or observing them trying to complete a given task with the technique. These initial evaluation experiences are drawn upon heavily for the process of taxonomization and categorization (section 1.1.2). It is helpful, therefore, to gain as much experience of this type as possible so that good decisions can be made in the next phase of formalization.

1.1.2 Taxonomization and categorization

Our next step in creating a formal framework for design and evaluation is to establish a *taxonomy* of interaction techniques for each of the interaction tasks described above (figure 1). Such taxonomies partition the tasks into separable subtasks, each of which represents a decision that must be made by the designer of a technique. Some of these subtasks are related directly to the task itself, while others may only be important as extensions of the metaphor on which the technique is based. In this sense, a taxonomy is the product of a careful task analysis. Once the task has been broken up to a sufficiently fine-grained level, the taxonomy is completed by listing possible methods (technique components) for accomplishing each of the lowest-level subtasks. An interaction technique is made up of one technique component from each of the lowest-level subtasks, such as the shaded components in figure 1.

Ideally, the taxonomies we establish for the universal tasks need to be correct, complete, and general. Any IT that can be conceived for the task should fit within the taxonomy. Thus, the subtasks will necessarily be abstract. The taxonomy will also list several possible technique components for each of the subtasks, but do not claim to list each conceivable component. For example, in an object coloring task, a taxonomy might list touching the virtual object, giving a voice command, or choosing an item in a menu as choices for the color application subtask. However, this does not preclude a technique which applies the color by some other means, such as pointing at the object.

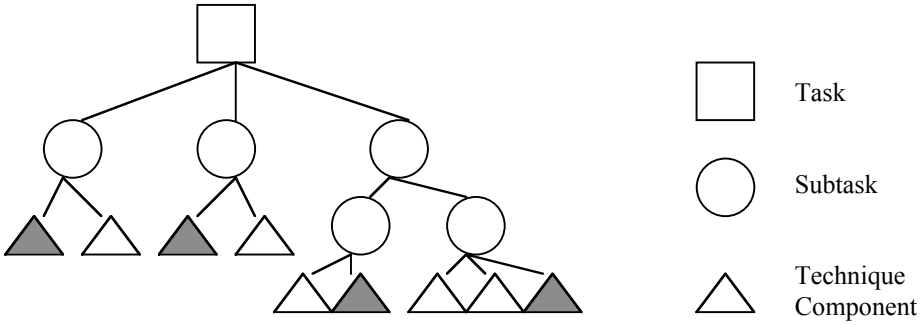

Figure 1. General taxonomy format

One way to verify the generality of the taxonomies we create is through the process of *categorization* – defining existing ITs within the framework of the taxonomy. If existing techniques for the task fit well into a taxonomy, we can be more sure of its correctness and completeness. Categorization also serves as an aid to evaluation of techniques. Fitting techniques into a taxonomy makes explicit their fundamental differences, and we can determine the effect of choices in a more fine-grained manner.

1.1.3 Guided design

Taxonomization and categorization are good ways to understand the low-level makeup of ITs, and to formalize the differences between them, but once they are in place, they can also be used in the design process. We can think of a taxonomy not only as a characterization, but also as a design space. In other words, a taxonomy informs or guides the design of new ITs for the task, rather than relying on a sudden burst of insight.

Since a taxonomy breaks the task down into separable subtasks, we can consider a wide range of designs quite quickly, simply by trying different combinations of technique components for each of the subtasks. There is no guarantee that a given combination will make sense as a complete interaction technique, but the systematic nature of the taxonomy makes it easy to generate designs and to reject inappropriate combinations.

Categorization may also lead to new design ideas. Placing existing techniques into a design space allows us to see the "holes" that are left behind – combinations of components that have not yet been attempted. One or more of the holes may contain a novel, useful technique for the task at hand. This process can be extremely useful when the number of subtasks is small enough and the choices for each of the subtasks are clear enough to allow a graphical representation of the design space, as this makes the untried designs quite obvious (Card, Mackinlay, & Robertson, 1990).

1.2 Universal interaction tasks

What user tasks do we need to support in immersive VEs? At first glance, there appear to be an extremely large number of possible user tasks – too many, in fact, to think about scientific design and evaluation for all of them. However, as Foley (1979) has argued for 2D interaction, there is also a set of "universal" tasks (simple tasks that are present in most applications, and which can be combined to form more complex tasks) for 3D interfaces. These universal tasks include *navigation*, *selection*, and *manipulation*.

Navigation refers to the task of moving the viewpoint within the three-dimensional space, and includes both a cognitive component (wayfinding), and a motor component (travel, also called viewpoint motion control). Selection refers to the specification of one or more objects from a set. Finally, manipulation refers to the modification of various object attributes (including position and orientation, and possibly scale, shape, color, texture, or other properties). Selection may be used on its own to specify an object to which a command will be applied (e.g. "delete the selected object"), or it might denote the beginning of a manipulation task.

These simple tasks are the building blocks from which more complex interactions arise. For example, the user of a surgery simulator might have the task of making an incision. This task might involve approaching the operating table (navigation), picking up a virtual scalpel (selection), and moving the scalpel slowly along the desired incision line (manipulation). One class of complex tasks, *system control*, involves the user giving commands to the system. For example, this might be

accomplished by bringing a virtual menu into view (manipulation) and then choosing a menu item (selection). However, system control is so ubiquitous in VE applications that the design of system control techniques can be considered separately.

This chapter will be targeted at developers and researchers who produce complete VE applications. It will provide background information, a large set of potential techniques for the universal interaction tasks, and guidelines to help in the choice of an existing technique or the design of a new technique for a particular system. Use of these guidelines should lead to more usable, useful, efficient, and effective VEs.

2 Naturalism vs. magic

A common misconception about virtual environments is that, in order to be effective, they should work exactly the way the real world works, or at least as close as is practically possible (interaction with a VE application should be "natural"). In fact, the very term "virtual reality" promotes such a view – that virtual reality should be the same as "real reality." In fact, this is not always the case. It may be very useful to create VEs which operate quite differently from the physical world.

In the previous chapter, techniques for natural user movement were discussed. There are several advantages to such techniques. Natural mappings are present, so that users can easily perform the task based on principles they are familiar with from daily life. Also, this simulation of the physical world may create a greater sense of immersion or presence in the virtual world. Finally, realism may enhance the user experience.

However, there is an alternative to the naturalistic approach, which we'll call "magic" (Smith, 1987). In this approach, the user is given new abilities, and non-natural methods for performing tasks are used. Examples include allowing the user to change his scale (grow or shrink), providing the user with an extremely long arm to manipulate faraway objects, or letting the user fly like a bird. Magic techniques are less natural, and thus may require more explanation or instruction, but they can also be more flexible and efficient if designed for specific interaction tasks.

Clearly, there are some applications that need naturalism. The most common example is training, in which users are trained in a virtual environment for tasks that will be carried out in a physical environment. Such applications have the *requirement* of natural interaction. On the other hand, applications such as immersive architectural design do not require complete naturalism – the user only

has the goal of completing certain tasks, and the performance requirements of the system do not include naturalism.

Therefore, in this chapter we will present, for the most part, techniques involving some magic or non-realism, in the interest of optimizing performance. Such techniques may enhance the user's physical, perceptual, or cognitive abilities, and take advantage of the fact that the VE can operate in any fashion. No possible techniques are excluded from consideration as long as they exhibit desirable performance characteristics (task performance and usability).

3 General guidelines

When attempting to develop guidelines that will produce high-performance VE interaction, we can look both at generic guidelines that inform interaction design at a high level, and specific guidelines for the common tasks described in the introduction. The next two sections will cover these areas. Guidelines are not intended to be exhaustive, but are limited to those which are especially relevant to enhancing performance, and those which have been verified through formal evaluation. A large number of VE-specific usability guidelines can be found in (Gabbard & Hix, 1998).

3.1 Existing HCI Guidelines

The first thing to remember when developing interaction for virtual environments is that interaction is not new! The field of human-computer interaction (HCI) has its roots in many areas, including perceptual and cognitive psychology, graphic design, and computer science, and has a long history of design and evaluation of two-dimensional computer interfaces. Through this process, a large number of general-purpose guidelines have been developed which have wide applicability to interactive systems, and not just the standard desktop computer applications with which everyone is familiar.

Therefore, we can take advantage of this existing knowledge and experience in our interaction design for VEs. If we do not meet these most basic requirements, then our system is sure to be unusable. Furthermore, the application of these principles to VEs may lead to VE-specific guidelines as well. These guidelines are well-known, if not always widely practiced, so we will not go over them in detail here.

Practice user-centered design and follow well-known general principles from HCI research.

Two important sources for such general guidelines are Donald Norman's *The Design of Everyday Things* (Norman, 1990) and Jakob Nielsen's usability heuristics (Nielsen & Molich, 1992). These

guidelines focus on high-level and abstract concepts such as making information visible (how to use the system, what the state of the system is, etc.), providing affordances and constraints, using precise and unambiguous language in labeling, designing for both novice and expert users, and designing for prevention of and recovery from errors.

Following such guidelines should lead to a more understandable, efficient, and usable system. However, because of their abstract nature, applying these principles is not always straightforward. Nevertheless, they must be considered as the first step towards a well-performing system.

3.2 Choice of devices

Another basic question one must ask when designing a VE system regards the choice of input and output devices to be used. Currently, little empirical data exists about relative interaction performance, especially for VE display devices. There are, however, a few general guidelines we can posit here.

Three common VE display devices, as described in chapter 3, are head-mounted displays (HMDs), semi-surrounding projected stereo displays, such as the CAVE™, and desktop stereo displays, such as the Responsive Workbench. These display types have very different characteristics, and interaction with these displays is likely to be extremely different as well.

Use HMDs or CAVE-like displays when immersion within a space is a performance requirement. Use workbench displays when viewing a single object or set of objects from an exocentric point of view.

These two guidelines get to the heart of the difference between the display types. HMDs and CAVEs encourage an egocentric, inside-out point of view, and are therefore appropriate for first-person tasks such as walkthroughs or first-person gaming. Workbench displays support an outside-in point of view and therefore work well for third-person tasks such as manipulating a single object or arranging military forces on a 3D terrain.

In CAVE-like displays, design the system to minimize the amount of manual rotation needed.

Most projected VEs do not completely surround the user. For example, the CAVE™ places graphics on four surfaces of a six-sided cube (floor and three walls). Thus, the ceiling and back wall are missing. This means that for the user to view what is directly behind or above her, she must rotate the environment indirectly, using some input device. In an application in which immersion is important, this manual rotation will likely break the illusion of presence within the space to some degree. One way to alleviate this problem is to adopt a vehicle metaphor for navigation, so that the user is always facing

the front wall and using the side walls for peripheral vision. With a steering wheel for choosing vehicle direction, manual rotation seems much more natural.

Input devices and their differences have been studied more extensively than display differences. Common VE input devices include 6 DOF trackers, continuous posture-recognition gloves, discrete event gloves, pen-like devices, simple button devices, and special-purpose devices such as the Spaceball™ or force-feedback joysticks.

Use an input device with the appropriate number of degrees of freedom for the task.

Many inherently simple tasks become more complex if an improper choice of input device is made. For example, toggling a switch is inherently a one degree of freedom task (the switch is on or off). Using an interaction technique which requires the user to place a tracked hand within a virtual button (a three DOF task) makes it overly complex. A simple discrete event device, such as a pinch glove, makes the task simpler. Of course, one must tradeoff the reduced degrees of freedom with the arbitrary nature of the various positions the user must learn when using a pinch glove or other such device. Also, such devices can generate only a finite number of events. In general, we should strive to reduce unnecessary DOFs when it is practical (Hinckley, Pausch, Goble, & Kassell, 1994).

Use physical props to constrain and disambiguate complex spatial tasks.

This guideline is related to the previous discussion about degrees of freedom. Physical props can help to reduce the number of DOFs that the user must control. For example, the pen & tablet interaction metaphor (Bowman, Wineman, Hodges, & Allison, 1998) uses a physical tablet (2D surface) and a tracked pen. A 2D interface is virtually displayed on the surface of the tablet, for tasks such as button presses, menu selection, and 2D drag & drop (figure 2). The physical props allow the user to do these tasks precisely, because the tablet surface guides and constrains the interaction to two dimensions.

Figure 2. Physical (left) and virtual (right) views of a pen & tablet system

Physical props can also make complex spatial visualization easier. For example, in the Netra system for neurosurgical planning (Goble, Hinckley, Pausch, Snell, & Kassell, 1995), it was found that surgeons had difficulty rotating the displayed brain data to the correct orientation when a simple tracker was used to control rotation. However, when the tracker was embedded within a doll's head, the task became much easier, because the prop gave orientation cues to the user.

Use absolute devices for positioning tasks and relative devices for tasks to control the rate of movement.

This guideline is well-known in desktop computing, but not always followed in VEs. Absolute positioning devices such as trackers will work best when their position is mapped to the position of a virtual object. Relative devices such as joysticks excel when their movement from the center point is mapped to the rate of change (velocity) of an object, usually the viewpoint. Interaction techniques which use absolute devices for velocity control or relative devices for position control will perform less efficiently and easily.

3.3 Interacting in three-dimensional space

By its very nature, 3D (also called spatial) interaction is qualitatively and quantitatively different than standard 2D interaction. As we saw in the previous section, the choice of 3D input devices can be quite important, but there are also other general principles related to the way 3D interaction is implemented in software.

Take advantage of the user's proprioceptive sense for precise and natural 3D interaction.

Proprioception is a person's sense of the location of the parts of his body, no matter how the body is positioned. For example, a driver can easily change gears without looking, because of his knowledge of

his body and hand position relative to the gearshift. Mine, Brooks, and Sequin (1997) discuss how we can take advantage of this sense in VEs by providing body-centered interactions. One possibility is to give the user a virtual tool belt on which various tools (e.g. pointer, cutting plane, spray paint, etc.) can be hung. Because the user knows where the various tools are located on his body, he can interact and choose tools much more efficiently and easily, without looking away from his work.

Use well-known 2D interaction metaphors if appropriate and practical.

It seems to be an unspoken rule among VE application developers that interaction techniques should be new and unique to VEs. This is as much a myth as the concept discussed earlier that all interaction should mimic the real world. In fact, there are many 2D interaction metaphors which can be used directly in or adapted for use in VEs. Pull-down or pop-up menus, 2D buttons, and 2D drag & drop manipulation have all been implemented in VEs with success (Bowman et al, 1998). Again, the issue often is related to reducing the number of DOFs the user must control. When 2D interaction metaphors are used, the provision of a 2D surface for interaction (such as the pen & tablet metaphor discussed above) can increase precision and efficiency.

Allow two-handed interaction if appropriate.

Most VE interfaces "tie one hand behind the user's back," allowing only input from a single hand. This severely limits the flexibility and expressiveness of input. By using two hands in a natural manner, the user can specify arbitrary spatial relationships, not just absolute positions in space. However, it should not be assumed that both hands will be used in parallel to increase efficiency. Rather, the most effective two-handed interfaces are those in which the non-dominant hand provides a frame of reference in which the dominant hand can do precise work (Hinckley, Pausch, Profitt, Patten, & Kassell, 1997).

Provide redundant interaction techniques for a single task.

One of the biggest problems facing evaluators of VE interaction is that the individual differences in user performance seem to be quite large relative to 2D interfaces. Some users seem to comprehend complex techniques easily and intuitively, while others may never become fully comfortable. Work on discovering the human characteristics that cause these differences is ongoing, but one way to mitigate this problem is to provide multiple interaction techniques for the same task. For example, one user may think of navigation as specifying a location within a space, and therefore would benefit from the use of a technique where the new location is indicated by pointing to that location on a map. Another user

may think of navigation as executing a continuous path through the environment, and would benefit from a continuous steering technique. In general, “optimal” interaction techniques may not exist, even if the user population is well known, so it may be appropriate to provide two or more techniques each of which have unique benefits. Of course, the addition of techniques also increases the complexity of the system, and so this must be done with care and only when there is a clear benefit.

4 Techniques for common VE tasks

4.1 Travel

Travel, also called viewpoint motion control, is the most ubiquitous and common VE interaction task – simply the movement of the user within the environment. Travel and wayfinding (the cognitive process of determining one’s location within a space and how to move to a desired location – see chapter 28) make up the task of navigation. Chapter 15 presented locomotion interface devices, which are physical devices supporting travel tasks, so in this section we will focus on passive movement, in which the user remains physically stationary while moving through the space.

There are three primary tasks for which travel is used within a VE. *Exploration* is travel which has no specific target, but which is used to build knowledge of the environment or browse the space. *Search* tasks have a specific target, whose location may be completely unknown (naïve search) or previously seen (primed search). Finally, *maneuvering* tasks refer to short, precise movements with the goal of positioning the viewpoint for another type of task, such as object manipulation. Each of these three types of tasks may require different travel techniques to be most effective, depending on the application.

4.1.1 Technique classifications

Because travel is so universal, a multitude of techniques have been proposed (see (Mine, 1995) for a survey of early techniques). Many techniques have similar characteristics, so it will be useful to present classifications of techniques rather than discussing each technique separately.

A simple taxonomy of passive movement techniques was described in (Bowman, Koller, & Hodges, 1997), and is reproduced in figure 3. This taxonomy partitions the task into three subtasks: direction or target selection, velocity and/or acceleration selection, and conditions of input (specifying the beginning and end of movement).

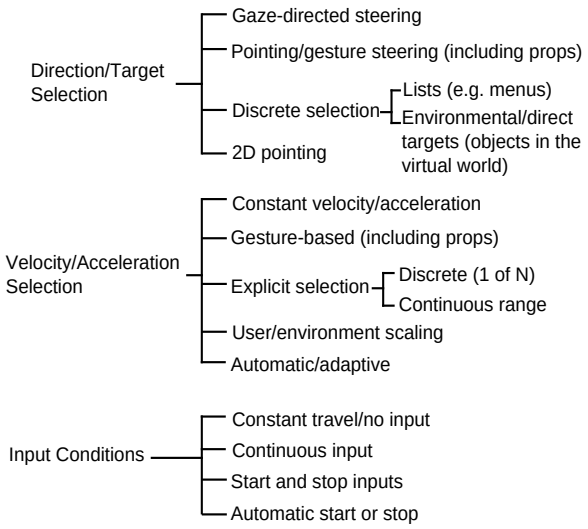

Figure 3. Taxonomy of passive, first-person travel techniques

Most techniques differ only in the direction or target specification subtask, and several common techniques are listed in the taxonomy. Gaze-directed steering uses the orientation of the head for steering, while pointing gets this information from the user's hand. The orientation of other body parts, such as the torso or foot, could also be used. Physical devices, such as a steering wheel, provide another way to specify direction. Other techniques specify only a target of motion, by choosing from a list, entering coordinates, pointing to a position on a map, or pointing at the target object in the environment.

The velocity/acceleration selection subtask has been studied much less, but several techniques have been proposed. Most systems generally default to a constant velocity. Gesture-based techniques use hand or body motions to indicate velocity or acceleration (e.g. speed depends on the distance of the hand from the body). Again, physical props such as accelerator and brake pedals can be used. The velocity or acceleration could be chosen discretely from a menu. Finally, velocity and acceleration may be automatically controlled by the system in a context-sensitive fashion (e.g. depending on the distance from the target or the amount of time the user has been moving).

The conditions of input may seem trivial, but this simple subtask can have an effect on performance. Generally, the user simply gives a single input, such as a button press, to begin moving, and another to stop moving. There may also be situations where it is appropriate for the system to automatically begin and/or end the motion, or where the user should be moving continuously.

Another way to classify travel techniques relates to the amount of control that the user has over viewpoint motion. *Steering* techniques give the user full control of at least the direction of motion.

These include continuous steering, such as gaze-directed or pointing techniques, and discrete steering, such as a ship steering technique in which verbal commands are interpreted to give the user control of the ship's rudders and engines (William Wells, personal communication, 1999). On the other end of the control spectrum, *target-based* techniques only allow the user to specify the goal of the motion, while the system determines how the viewpoint will move from the current location to the target. This requires the use of a selection technique (see section 4.2). *Route-planning* techniques represent a middle ground. Here the user specifies a path between the current location and the goal, and then the system executes that path. This might be implemented by drawing a path on a map of the environment, or placing markers using some manipulation technique and having the system interpolate a spline between these control points.

Finally, there is a class of techniques which do not fit well into the above classifications that use manual manipulation to specify viewpoint motion. Ware and Osborne (1990) identified the "camera in hand" metaphor, where the user's hand motion above a map or model of the space specifies the viewpoint from which the scene will be rendered, and the "scene in hand" metaphor, in which the environment itself is attached to the user's hand position. Both of these techniques are exocentric in nature, but manual viewpoint manipulation can also be done from a first-person perspective. Any direct object manipulation technique (see section 4.3) can be modified so that the user's hand movements affect the viewpoint instead of the selected object. The selected object remains fixed in the environment, and the user moves around that object using hand motions. Such a technique might be extremely useful for maneuvering tasks where the user is constantly switching between travel and manipulation.

4.1.2 Travel performance requirements

What types of performance requirements do VE applications have for travel techniques? Depending on the type of travel task, and the nature of the application, there are many possible performance needs. Some representative requirements categories are presented here.

1. *Speed* (task completion time)
2. *Accuracy* (proximity to the desired target)
3. *Spatial Awareness* (the user's knowledge of his position and orientation within the environment during and after travel)
4. *Ease of Learning* (the ability of a novice user to use the technique)

5. *Ease of Use* (the complexity or cognitive load of the technique from the user's point of view)
6. *Information Gathering* (the user's ability to actively obtain information from the environment during travel)
7. *Presence* (the user's sense of immersion or "being within" the environment due to travel)
8. *User Comfort* (lack of strain, simulator sickness, dizziness, or nausea)

4.1.3 Guidelines for designing travel techniques

Make simple travel tasks simple by using target-based techniques.

If the goal of travel is simply to move to a new location, such as moving to the location of another task, target-based techniques provide the simplest metaphor for the user to accomplish this task. In many cases, the exact path of travel itself is not important; only the end goal is important. In such situations, target-based techniques make intuitive sense, and leave the user's cognitive and motor resources free to perform other tasks. The use of target-based techniques assumes that the desired goal locations are known in advance or will always coincide with objects in the environment, so if this is not true, such techniques will not be appropriate.

Always use physical head motion for viewpoint orientation if possible.

Almost all VE systems use head tracking to render the scene from the user's point of view. Since viewpoint motion control involves both viewpoint position and orientation, it makes sense to use the head tracking information for setting viewpoint orientation. However, in certain applications, especially those in which the user is seated, it might be tempting to specify viewpoint orientation indirectly, for example by using a joystick. It has been shown (Chance, Gaunet, Beall, & Loomis, 1998) that such indirect orientation control, which does not take advantage of proprioception, has a damaging effect on the spatial orientation of the user. Therefore, when it is important for the user to understand the spatial structure of the environment, physical head motion should always be used.

Avoid the use of teleportation; instead, provide smooth transitional motion between locations.

Teleportation, or "jumping," refers to a target-based travel technique in which velocity is infinite – that is, the user is moved immediately from the starting position to the target. Such a technique seems very attractive from the perspective of efficiency. However, evaluation (Bowman et al, 1997) has shown that disorientation results from teleportation techniques. Interestingly, all techniques that used

continuous smooth motion between the starting position and the target caused little disorientation, even when the velocity was relatively high.

If steering techniques are used, train users in strategies to acquire survey knowledge. Use target-based or route-planning techniques if spatial orientation is required but training is not possible.

Spatial orientation (the user's spatial knowledge of the environment and her position and orientation within it) is critical in many large-scale VEs, such as those designed to train users about a real world location. The choice of interaction techniques can affect spatial orientation. In particular, evaluation (Bowman, Davis, Hodges, & Badre, 1999) has shown that the highest performance on spatial orientation can be obtained with the use of steering techniques where the user has the highest degree of control, but only if sophisticated strategies are used (e.g. flying above the environment to obtain a survey view, moving in structured patterns). If such strategies are not used, steering techniques may actually perform worse, because users are concentrating on controlling motion rather than viewing the environment. Techniques where the user has less control over motion, such as target-based and route-planning techniques, provide moderate levels of spatial orientation due to the low cognitive load they place on the user during travel – the user can take note of spatial features during travel because the system is controlling motion.

Use non-head-coupled techniques for efficiency in relative motion tasks. If relative motion is not important, use gaze-directed steering to reduce cognitive load.

Relative motion is a common VE task in which the user wishes to position the viewpoint at a location in space relative to some object. For example, an architect wishes to view a structure from the proposed location of the entrance gate, which is a certain distance from the front door – movement must be relative to the door, and not to any specific object. A comparison of steering techniques showed that a pointing technique performed much more efficiently on this task than gaze-directed steering, because pointing allows the user to look at the object of interest while moving, while gaze-directed steering forces the user to look in the direction of motion. Gaze-directed steering performs especially badly when motion needs to be away from the object of interest. Thus, techniques that are not coupled to head motion support relative motion tasks. On the other hand, non-head-coupled techniques are slightly more cognitively complex than gaze-directed steering, so it may still be useful if relative motion is not an important task.

Consider integrated travel and manipulation techniques if the main goal of viewpoint motion is to maneuver for object manipulation.

Manual viewpoint manipulation techniques use object manipulation metaphors to specify viewpoint position. Such techniques have been shown experimentally to perform poorly on general travel tasks such as exploration and search. However, such techniques may prove quite useful if the main goal of travel is to maneuver the viewpoint during object manipulation. Manual viewpoint manipulation allows the use of the same technique for both travel and object manipulation tasks, which may be intermixed quite frequently in applications requiring complex manipulation.

Provide wayfinding and prediction aids to help the user decide where to move, and integrate those aids with the travel technique.

The design of interaction techniques for travel assumes that the user knows where to go and how to get there, but this is not generally the case. Wayfinding aids (Darken, 1996) may be needed, especially in large-scale VEs where the user is expected to build survey knowledge of the space. Such aids include maps, signposts, compass markings, and paths. During travel, the user may need to know whether the current path will take them to the desired location. Predictor displays (Wickens, Haskell, & Harte, 1989) have long been used in aircraft simulation to show the pilot the result of the current heading, pitch, and speed. Such displays might also be useful in other VEs which involve high speed three-dimensional motion.

4.2 Selection

Selection is simply the task of specifying an object or set of objects for some action. Most often, selection precedes object manipulation (section 3.3), or specifies the object of some command (section 3.4), such as “delete the selected object.” In interactively complex VEs, selection tasks occur quite often, and therefore efficiency and ease of use are important performance requirements for this task.

4.2.1 Technique classifications

The most obvious VE selection technique is again the one that mimics real-world interaction – simply touching the desired object with a virtual hand. Within this general metaphor, there are several possible implementations, however. The virtual hand could simply be a rigid object controlled by a single 6 DOF tracker, or it could be controlled by a glove which recognizes a multitude of different hand postures for more precision. Another issue relates to the precision of selection. Can the user only

select at the object level, or can specific points on an object's surface be selected? Finally, there is the issue of selection feedback. Most systems present simple graphical (e.g. highlight the object) or audio feedback to indicate touching, but haptic feedback (chapter 5) more closely simulates real-world touching and allows the user to select objects without looking at them.

No matter its implementation, this simple virtual hand metaphor suffers from a serious problem. The user can only select objects in the VE which are actually within arm's reach. In many large-scale VEs, especially in the application domains of design or prototyping, the user will wish to select remote objects – those outside of the local area surrounding the user. Therefore, a number of "magic" techniques have been developed for object selection. These fall into two main categories: arm-extension and ray-casting.

Arm-extension techniques still use a virtual hand to select objects via touching, but the virtual hand has a much greater range than the user's physical hand. The simplest example is a technique which linearly maps the physical hand movements onto the virtual hand movements, so that for each unit the physical hand moves away from the body, the virtual hand moves away N units. The Go-Go technique (Poupyrev, Billinghurst, Weghorst, & Ichikawa, 1996) takes a more thoughtful approach. It defines a radius around the user within which the physical hand is mapped directly to the virtual hand. Outside that radius, a non-linear mapping is applied to allow the virtual hand to reach quite far into the environment, although still only a finite distance (figure 4). Other techniques allow infinite arm extension, such as an indirect technique which uses two buttons to extend and retract the virtual hand. Such techniques are generally less natural and induce more cognitive load.

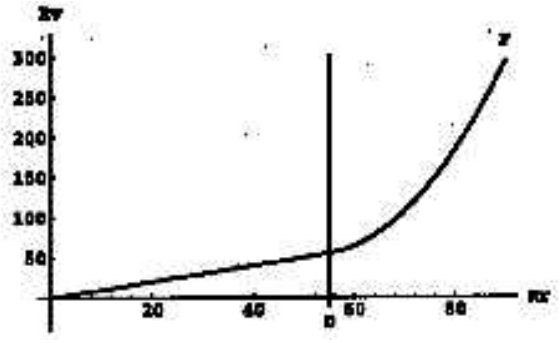

Figure 4. Mapping function for Go-Go technique

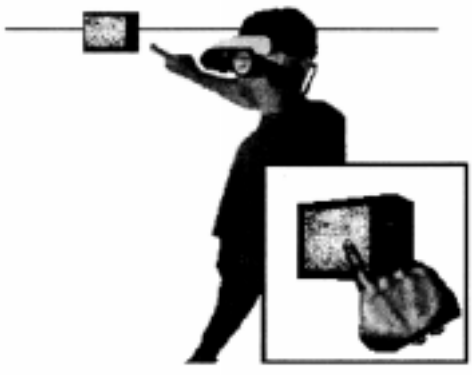

Figure 5. Occlusion selection

Ray-casting techniques move away from the object touching metaphor and instead adopt a pointing metaphor. A ray emanates from the user, with the user controlling its orientation only, and the first

object the ray intersects may be selected. The ray may be linear, or it may be cone-shaped so that small objects are more easily selected at a distance. The most common implementation of ray-casting is to attach the ray to the user's virtual hand, so that simple wrist movements allow pointing in any direction (Mine, 1995). Another class of techniques use gaze direction for ray-casting, so that an object can be selected by placing it in the center of one's field of view. Finally, some techniques use a combination of gaze direction and hand position for selection (Pierce et al, 1997), with the ray emanating from the eyepoint and passing through the virtual hand position (figure 5). This is often called occlusion, or framing, selection.

Bowman and Hodges (1999) defined a taxonomy of selection techniques, which is presented at the top of figure 6. Note that besides the subtask we've been discussing (indication of object), there are also feedback and indication to select subtasks. The latter refers to the event used to signal selection to the system, such as a button press, gesture, or voice command.

Finally, we note that all of the techniques we've presented are designed for single-object selection only. Selection of multiple objects simultaneously has not been the subject of much research, but techniques from 2D interfaces may work reasonably well in three dimensions. These include sequential selection with a modifier button pressed and rubberbanding or "lassoing." Techniques such as rubberbanding which must be extended to specify a 3D volume will present interesting usability challenges.

4.2.2 Selection performance requirements

The possible performance requirements for selection techniques are a subset of the ones presented for travel, plus a requirement indicating which objects can be selected:

1. *Speed* (task completion time)
2. *Accuracy* (the lack of selection errors)
3. *Ease of Learning* (the ability of a novice user to use the technique)
4. *Ease of Use* (the complexity or cognitive load of the technique from the user's point of view)
5. *User Comfort* (lack of strain, simulator sickness, dizziness, or nausea)
6. *Expressiveness* (the number and distance of objects that can be selected)

4.2.3 Guidelines for designing selection techniques

Use a natural technique if all selection is within arm's reach.

The simple virtual hand metaphor works well in systems where all of the interaction with objects is local. This usually includes VE applications implemented on a workbench display, where most of the objects lie on or above the surface of the table.

Use ray-casting techniques if speed of remote selection is a requirement.

Evaluation has shown (Bowman & Hodges, 1999) that ray-casting techniques perform more efficiently than arm-extension techniques over a wide range of possible object distances, sizes, and densities. This is due to the fact that ray-casting selection is essentially 2D (in the most common implementation, the user simply changes the pitch and yaw of the wrist).

Ensure that the chosen selection technique integrates well with the manipulation technique to be used.

Selection is most often used to begin object manipulation, and so there must be a seamless transition between the selection and manipulation techniques to be used in an application. Arm-extension techniques generally provide this transition, because the selected object is also manipulated directly with the virtual arm, and so the same technique is used throughout the interaction. As discussed below, however, it is possible to integrate ray-casting techniques with efficient manipulation techniques.

Consider multi-modal input for combined selection and command tasks.

When selection is used in combination with system control tasks, it may be more efficient and natural to use multi-modal interaction. For example, one may point at an object and then give the voice command "delete."

If possible, design the environment to maximize the perceived size of objects.

Selection errors are affected by both the size and distance of objects, using either ray-casting or arm-extension techniques (Bowman & Hodges, 1999). These two characteristics can be combined in the single attribute of visual angle, or the perceived size of the object in the image. Unless the application requires precise replication of a real-world environment, manipulating the perceived size of objects will allow more efficient selection.

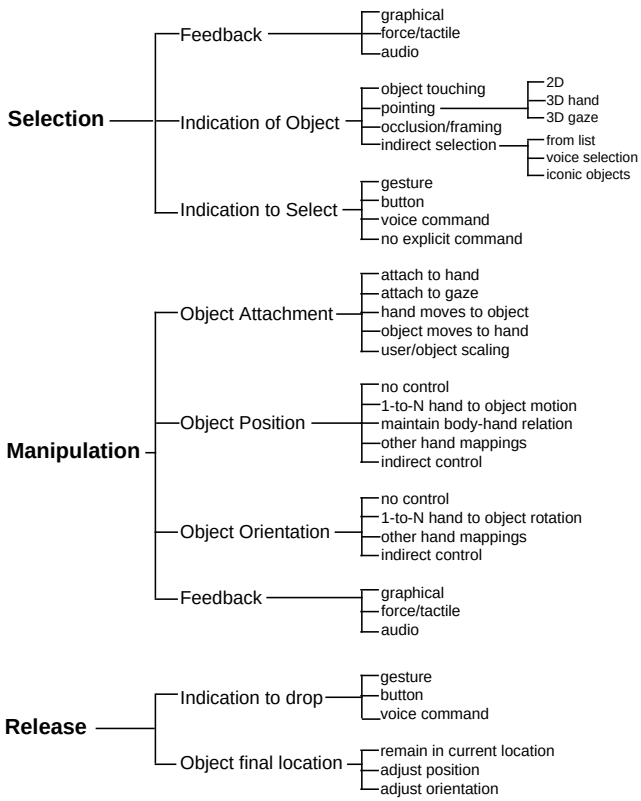

Figure 6. Taxonomy of single-object selection & manipulation techniques

4.3 Manipulation

As we've noted, manipulation goes hand in hand with selection. Manipulation refers broadly to modification of the attributes of the selected object. Attributes may include position, orientation, scale, shape, color, or texture. For the most part, research has only considered the manipulation of the position and orientation of rigid objects, although some special-purpose applications include object deformation or scaling. Object manipulation tasks have importance in such applications as design, prototyping, simulation, and entertainment, all of which may require environments that can be modified by the user.

4.3.1 Technique classifications

Again, the most common object manipulation technique is a natural one, in which the selected object is rigidly attached to the virtual hand and moves along with it until some signal is given to release the object. This technique is simple and intuitive, but certain object orientations may require the user to twist the arm or wrist to uncomfortable positions, and it does not use the inherent dexterity of the user's fingers. Recent research, then, has focused on more precise and dextrous object manipulation using fingertip control (Kitamura, Yee, & Kishino, 1998). This can be simulated to a degree using a rigid virtual hand if a clutching mechanism is provided. Researchers have also proposed two-handed

object manipulation techniques, which often use the non-dominant hand as a reference (e.g. a pivot point for rotation), and the dominant hand for the fine, precise manipulation.

As with selection, the natural manipulation techniques suffer from the limitations of reach. Also, manipulating large objects within arm's reach may occlude the user's view. Therefore, techniques for remote manipulation are also important, and several categories of such techniques have been proposed.

The arm-extension and ray-casting selection techniques can also be used for object manipulation. Arm-extension metaphors simply attach the object to the virtual hand and allow the user to control it using the same physical-to-virtual hand mapping (Poupyrev et al, 1996). Ray-casting techniques may attach the object to the ray itself, which allows intutive, but limited and imprecise manipulation (Bowman & Hodges, 1997). Because ray-casting is so efficient as a selection mechanism, several researchers have attempted to increase the utility of ray-casting for manipulation. One idea is to select using ray-casting and then move the virtual hand to the selected object for direct manipulation (Bowman & Hodges, 1997). Another set of techniques scales the user or the environment so that the virtual hand, which was originally far from the selected object, is actually touching this object, so that it can be manipulated directly (Pierce et al, 1997).

Another metaphor, called the World-in-Miniature (Stoakley, Conway, & Pausch, 1995), solves the remote object manipulation problem by giving the user a small hand-held copy of the environment. Direct manipulation of the WIM objects causes the larger environment objects to move as well. This technique may be implemented in 3D, or in 2D using the pen & tablet metaphor.

All of the techniques in the preceding section can be implemented in a general way – that is, allow objects to be positioned and oriented anywhere in the space – but most applications will benefit from the addition of special manipulation aids and constraints. One way to do this is through the use of snapping or gridlines (Mine, 1995), such that objects can only end in a discrete number of positions or orientations, or objects line up with other objects. Another technique creates a physical simulation including gravity and impenetrable objects, so that manipulation results in a realistic configuration of objects. Physical simulation may be compute-intensive, however. Finally, objects themselves can be given intelligence (Bukowski & Sequin, 1995), so that coffee cups rest on their bottom surface and paintings hang on walls, for example.

The taxonomy presented by Bowman and Hodges (1999) also addresses object manipulation, and the release of objects after manipulation (figure 6). The size of the manipulation taxonomy indicates

that the design space is quite large. New techniques can be created using the process of guided design by combining components for each of the lowest-level subtasks.

4.3.2 Manipulation performance requirements

Manipulation techniques may have several requirements for performance as well,:

1. *Speed* (task completion time)
2. *Accuracy* (proximity to the desired object position and orientation)
3. *Ease of Learning* (the ability of a novice user to use the technique)
4. *Ease of Use* (the complexity or cognitive load of the technique from the user's point of view)
5. *User Comfort* (lack of strain, simulator sickness, dizziness, or nausea)
6. *Expressiveness* (the ability to position and orient the object at any desired location in the environment)

4.3.3 Guidelines for designing manipulation techniques

Reduce the number of degrees of freedom to be manipulated if the application allows it.

Provide general or application-specific constraints or manipulation aids.

These two guidelines address the same issue: reducing the complexity of interaction from the user's point of view. This can be done by considering the characteristics of the application (e.g. in an interior design task, the furniture should remain on the floor), by off-loading complexity to the computer (using constraints or physical simulation), or by providing widgets to allow the manipulation of one or several related DOFs (Mine, 1997). This also relates to the guideline concerning the DOFs of the input device to be used.

Allow direct manipulation with the virtual hand instead of using a tool.

Manipulation techniques which allow the direct positioning and orienting of virtual objects with the user's hand have been shown empirically (Bowman & Hodges, 1999) to perform more efficiently and to provide greater user satisfaction than techniques using a tool (such as a virtual light ray).

Avoid excessive scaling of the user or environment.

Techniques which scale the user or the world to allow direct manipulation have some desirable characteristics. The user's perception of the scene does not change at the moment of selection, and small physical movements can allow large virtual movements (Pierce et al, 1997). However,

experimental data shows a correlation between the use of such techniques and discomfort (dizziness and nausea) in users (Bowman & Hodges, 1999).

Use indirect depth manipulation for increased efficiency and accuracy.

Indirect control of object depth, using joystick buttons for example, is completely unnatural, and requires some training to be used well. However, once this technique is learned, it provides more accurate object placement, especially if the target is far from the user (Bowman & Hodges, 1999). This increased accuracy leads to more efficient performance as well. Moreover, these techniques do not exhibit the arm strain that can result from the use of more natural arm-extension techniques.

4.4 System control

Many of the other interactions found in VE applications fall under the heading of system control. This category includes commands, mode changes, and other modifications of system state. Often, system control tasks are composites of the other universal tasks. For example, choosing a menu item is a selection task, while dragging an object to a trash can for deletion is a manipulation task.

There has been little empirical evaluation of VE system control techniques, and no formal classification that the author is aware of. Therefore, in this section we will focus on several categories of techniques including menus, voice commands, tools, and gestures.

4.4.1 Virtual menus

Menus are the most common form of system control found in VEs, and many of the virtual menu systems that have been developed are simple adaptations of menus from 2D desktop systems.

The most simple menu system is a series of labeled buttons that appears in the virtual environment. These may be at a specific location in the environment, or they may be attached to the user for greater availability from any location. Slightly more complex are pull-down menus, which appear only as a label, and whose items are revealed when the label is selected (Jacoby & Ellis, 1992). Pop-up menus have also been implemented, so that the menu appears at the location of the user's hand for easy access. Other implementations use menus on a virtual surface, such as in the pen & tablet metaphor or on the surface of a workbench display. Mine (1997) developed a rotary menu system in which items are chosen by rotating the wrist. This takes advantage of the fact that menu selection is essentially a one-dimensional task, and so menu selection is done by changing only one DOF. Figure 7 shows three example virtual menu systems.

Many virtual menu systems have faced a set of common problems. One is that the resolution of text, especially in HMDs, is low, and so menus and labels must contain fewer items and take up more of the display space. Also, input using trackers is imprecise, so menu items must be large and few submenus can be used. For a command-intensive application such as immersive design, these problems force designers to think of creative ways to issue commands.

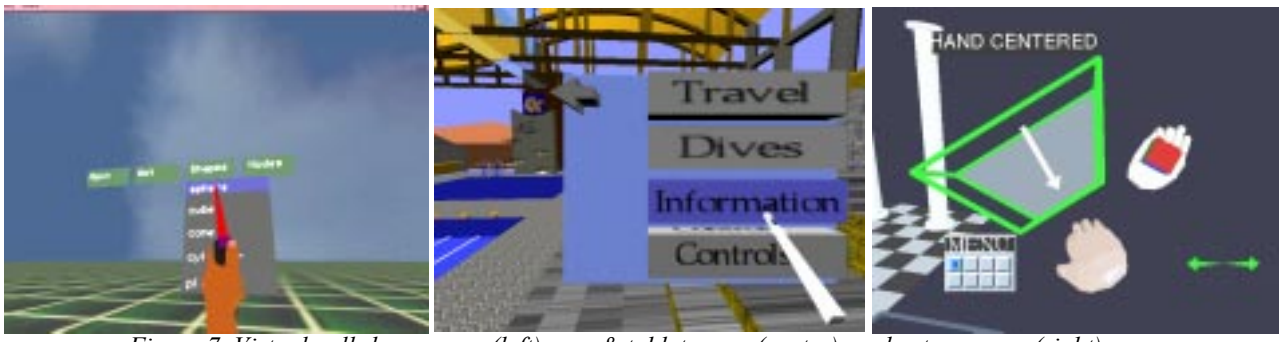

Figure 7. Virtual pull-down menu (left), pen & tablet menu (center), and rotary menu (right).

4.4.2 Voice commands

The use of voice as a command input is very popular in VEs. Voice has many advantages, including the simple input device (a microphone), the freedom to use the hands for other operations, and the flexibility of voice input to specify complex commands. Voice also has disadvantages, including limited recognition capability, forcing the user to remember arbitrary command utterances, and the distraction to other users in the same room.

Voice has most often been used to implement simple commands such as “save,” “delete,” or “quit,” but it has also been used in more complex menu hierarchies. Darken (1994) combined voice input with visual menus so that recall of command names was eliminated. Currently, much voice research is focused on multi-modal interaction, where voice and gesture or other input modalities are combined to form a richer and more precise interaction.

4.4.3 Gesture commands

Many of the earliest VE systems used gloves as input, and glove gestures to indicate commands. Gesture recognition is covered in detail in chapter 10. Advantages of using gestures include the flexibility and number of degrees of freedom of the human hand, the lack of a need for traditional input devices, and the ability to use two hands. However, gestures suffer from many of the same problems as voice, including forced user recall and poor recognition rates. For these reasons, most production VE systems use menu or button systems, or only primitive and limited gesture commands.

4.4.4 Virtual tools

Since having a large command space proves problematic for VE applications, developers have looked for ways to reduce the number of commands needed. One way to do this is through the use of virtual tools. Tools are common in many desktop applications as a more direct way to indicate an action to the system. For example, instead of selecting an area of the screen and choosing an "erase" command, the user can simply select an eraser tool and directly erase the desired areas.

There have been similar efforts in virtual environments. One example is Mine's ISAAC system (Mine, 1997), which makes use of a wide variety of tools to perform interactive modifications to a geometric scene. The Virtual Tricorder (Wloka & Greenfield, 1995) is a multi-purpose tool that changes appearance and function depending on the system mode and state.

4.4.5 System control performance requirements

The potential performance requirements for system control tasks are similar to the other universal interaction tasks, since they are the building blocks used to create a system control technique, but there are some additions:

1. *Speed* (task completion time)
2. *Accuracy* (the probability that a command will be recognized)
3. *Ease of Learning* (the ability of a novice user to use the technique)
4. *Ease of Use* (the complexity or cognitive load of the technique from the user's point of view)
5. *User Comfort* (lack of strain, simulator sickness, dizziness, or nausea)
6. *Expressiveness* (the ability to access all of the commands in the system)
7. *Unobtrusiveness* (the degree to which the system control components obscure the environment)
8. *Affordance* (the ability of the user to know where commands are located and what they do)

4.4.6 Guidelines for designing system control techniques

Unlike the other three universal interaction tasks, there has been very little evaluation of VE system control techniques. Therefore, most of the guidelines below are drawn from experience and intuition. Much more research is needed in this area.

Reduce the necessary number of commands in the application.

In many cases, system control in VEs is awkward and distracts from the actual task. Developers need to look closely at what commands are absolutely necessary, and what actions can be done using direct manipulation or virtual tools instead.

When using virtual menus, avoid submenus and make selection at most a 2D operation.

Input to VEs (usually based on 6 DOF trackers) can be quite imprecise. Requiring users to navigate more than one level of menus can lead to frustration and inefficiency. The same holds for menu systems that require 3D input. The use of a physical surface such as the pen & tablet metaphor can alleviate some of these problems, as can a 1D solution such as rotary menus.

Indirect menu selection may be more efficient over prolonged periods of use.

In a usability study of two hand-held virtual menu systems (Bowman, Hodges, & Bolter, 1998), it was found that menu item selection for novice users was more efficient when the items were touched directly with a stylus. However, as users became more expert with the menu system, a technique using joystick buttons to navigate the menu performed better, because users could "click ahead" of the system and because no large hand movements are needed.

Voice and gesture-based commands should include some method of reminding the user of the proper utterance or gesture.

The lack of affordances is a major problem for most gesture and voice command systems, unless the number of commands is very small. Therefore, small visual reminders of the allowed words or gestures may greatly enhance performance.

Integrate system control with other interaction tasks.

If the user thinks of the command subsystem as a completely separate component, the level of presence in the VE may decrease and the level of distraction may increase with the use of the system control technique. As much as is possible, system control should seem a part of the environment in which the user is working. For example, pull-down menus attached to the user's view seem unnatural and detached from the environment, while a hand-held menu seems part of the environment. Also, if the devices and widgets used for system control can also be used for travel, selection, or manipulation, the use of these techniques will not be as distracting to the user.

5 Evaluation & Application

This chapter has focused on the design of VE interaction techniques with high performance, but design is not the end of the story. *Evaluation* and *application* are also necessary components to the entire process of determining the appropriate interaction techniques for a VE application.

Following the guidelines and principles in this chapter should lead to interaction techniques that are well-suited for the performance requirements of VE applications. However, because of the youth of this research area, one cannot guarantee performance levels no matter how many guidelines are followed. For this reason, evaluation and assessment of VE interaction is essential. Chapter 39 covers the topic of usability evaluation in detail. Application designers need to test their systems with members of the intended user population to ensure that required performance levels are met.

For those who are designing new VE interaction techniques and performing basic research, the aspect of application must be kept in mind. Future research should focus on developing techniques that meet the performance requirements of current and proposed real-world VE applications. For examples of such applications, see chapter 47.

6 Conclusions

This chapter has been intended as a practical guide both for the researcher and developer of interactive VE applications. It is essential to remember that most of the guidelines and principles presented here are the results of formal evaluation, and that further guidelines should also be the product of careful testing.

We have also emphasized the concept of performance, with a broad definition. In this area, we stress that user comfort, satisfaction, ease of use, and other subjective parameters are in some cases more important than more traditional performance variables such as speed and accuracy. Application developers must carefully consider the performance requirements of their systems before choosing interaction techniques.

Another consideration is the required naturalism of the interface. Certain applications, as we have seen, need interaction that closely matches interaction in the physical environment. However, other applications with different requirements may benefit from the use of magic techniques which extend the user's real-world capabilities.

Finally, the issue of standards should be addressed. There has been a great deal of discussion about the possibility of a standard interface or set of interaction techniques for VEs, much like the desktop metaphor for personal computers. Time may prove the author wrong, but the current situation appears to suggest that it will not be fruitful to pursue a standardized interface for virtual environment interaction. Most of the types of VE applications that are currently in use or that have been proposed are highly specialized and domain-specific, and they exhibit a wide range of interaction performance requirements. There is currently no killer application for VEs which promises to put a head-mounted display in every home. Furthermore, evaluation has shown time and time again that the optimal interaction technique is non-absolute, but is instead application- and task-dependent. For these reasons, the author advocates the development of VE interaction techniques that are optimized for particular tasks and applied to particular systems.

Acknowledgements

The author would like to thank the following people for their help, support, and discussion regarding the issues in this chapter: Larry Hodges, Donald Allison, Drew Kessler, Rob Kooper, Donald Johnson, Albert Badre, Elizabeth Davis, Ivan Poupyrev, Joseph LaViola, Ernst Kruijff, Mark Mine, Matthew Conway, Jeffrey Pierce, Andrew Forsberg, Ken Hinckley, the other members of the 3DUI mailing list, and Dawn Bowman.

References

1. Bowman, D., Davis, E., Hodges, L., & Badre, A. (1999). Maintaining Spatial Orientation During Travel in an Immersive Virtual Environment. Submitted to *Presence: Teleoperators and Virtual Environments.*
2. Bowman, D., Koller, D., & Hodges, L. (1997). Travel in Immersive Virtual Environments: An Evaluation of Viewpoint Motion Control Techniques. In *Proceedings of Virtual Reality Annual International Symposium*, 45-52.
3. Bowman, D. & Hodges, L. (1997). An Evaluation of Techniques for Grabbing and Manipulating Remote Objects in Immersive Virtual Environments. In *Proceedings of the ACM Symposium on Interactive 3D Graphics*, 35-38.

4. Bowman, D. & Hodges, L. (1999). Formalizing the Design, Evaluation, and Application of Interaction Techniques for Immersive Virtual Environments. To appear in *Journal of Visual Languages and Computing*.
5. Bowman, D., Hodges, L., & Bolter, J. (1998). The Virtual Venue: User-Computer Interaction in Information-Rich Virtual Environments. *Presence: Teleoperators and Virtual Environments*, *7*(5), 478-493.
6. Bowman, D., Wineman, J., Hodges, L., & Allison, D. (1998). Designing Animal Habitats Within an Immersive VE. *IEEE Computer Graphics & Applications*, *18*(5), September/October, 9-13.
7. Brooks, F. et al. (1992). Final Technical Report: Walkthrough Project. Report to National Science Foundation.
8. Bukowski, R. & Sequin, C. (1995). Object Associations: A Simple and Practical Approach to Virtual 3D Manipulation. In *Proceedings of the ACM Symposium on Interactive 3D Graphics*, 131-138.
9. Card, S., Mackinlay, J., & Robertson, G. (1990). The Design Space of Input Devices. In *Proceedings of CHI*, 117-124.
10. Chance, S., Gaunet, F., Beall, A., & Loomis, J. (1998). Locomotion Mode Affects the Updating of Objects Encountered During Travel. *Presence: Teleoperators and Virtual Environments*, *7*(2), 168-178.
11. Darken, R. (1994). Hands-off Interaction with Menus in Virtual Spaces. In *Proceedings of SPIE, Stereoscopic Displays and Virtual Reality Systems*, *2177*, 365-371.
12. Darken, R. (1996). Wayfinding Behaviors and Strategies in Large Virtual Worlds. In *Proceedings of CHI*, ??-??.
13. Foley, J. (1979). A Standard Computer Graphics Subroutine Package. *Computers and Structures*, *10*, 141-147.
14. Gabbard, J., & Hix, D. (1998). Usability Engineering for Virtual Environments through a Taxonomy of Usability Characteristics. Submitted to *Presence: Teleoperators and Virtual Environments*.
15. Goble, J., Hinckley, K., Pausch, R., Snell, J., & Kassell, N. (1995). Two-Handed Spatial Interface Tools for Neurosurgical Planning. *IEEE Computer*, July, 20-26.

16. Hinckley, K., Pausch, R., Goble, J., & Kassell, N. (1994). Design Hints for Spatial Input. In *Proceedings of the ACM Symposium on User Interface Software and Technology*, 213-222.
17. Hinckley, K., Pausch, R., Profitt, D., Patten, J., & Kassell, N. (1997). Cooperative Bimanual Action. In *Proceedings of CHI*, 27-34.
18. Hodges, L., Rothbaum, B., Kooper, R., Opdyke, D., Meyer, T., North, M., de Graff, J., & Williford, J. (1995). Virtual Environments for Treating the Fear of Heights. *IEEE Computer*, *28*(7), 27-34.
19. Jacoby, R. & Ellis, S. (1992). Using Virtual Menus in a Virtual Environment. In *Proceedings of SPIE, Visual Data Interpretation*, *1668*, 39-48.
20. Kitamura, Y., Yee, A., & Kishino, F. (1998). A Sophisticated Manipulation Aid in a Virtual Environment using Dynamic Constraints among Object Faces. *Presence: Teleoperators and Virtual Environments*, *7*(5), 460-477.
21. Mine, M. (1995). Virtual Environment Interaction Techniques. UNC Chapel Hill Computer Science Technical Report TR95-018.
22. Mine, M. (1997). ISAAC: A Meta-CAD System for Virtual Environments. *Computer-Aided Design*, *29*(8), 547-553.
23. Mine, M., Brooks, F., & Sequin, C. (1997). Moving Objects in Space: Exploiting Proprioception in Virtual-Environment Interaction. In *Proceedings of SIGGRAPH*, in *Computer Graphics*, 19-26.
24. Nielsen, J. & Molich, R. (1992). Heuristic Evaluation of User Interfaces. In *Proceedings of CHI*, 249-256.
25. Norman, D. (1990). *The Design of Everyday Things*. Doubleday, New York.
26. Pierce, J., Forsberg, A., Conway, M., Hong, S., Zeleznik, R., & Mine, M. (1997). Image Plane Interaction Techniques in 3D Immersive Environments. In *Proceedings of the ACM Symposium on Interactive 3D Graphics*, 39-44.
27. Poupyrev, I., Billinghurst, M., Weghorst, S., & Ichikawa, T. (1996). The Go-Go Interaction Technique: Non-linear Mapping for Direct Manipulation in VR. In *Proceedings of the ACM Symposium on User Interface Software and Technology*, 79-80.
28. Price, B., Baecker, R., & Small. I. (1993). A Principled Taxonomy of Software Visualization. *Journal of Visual Languages and Computing*.

29. Smith, R. (1987). Experiences with the Alternate Reality Kit: An Example of the Tension between Literalism and Magic. In *Proceedings of ACM CHI+GI*, 61-67.
30. Stoakley, R., Conway, M., & Pausch, R. (1995). Virtual Reality on a WIM: Interactive Worlds in Miniature. In *Proceedings of CHI*, 265-272.
31. Ware, C. & Osborne, S. (1990). Exploration and Virtual Camera Control in Virtual Three Dimensional Environments. In *Proceedings of the ACM Symposium on Interactive 3D Graphics*, in *Computer Graphics*, *24*(2), 175-183.
32. Wickens, C., Haskell, I., & Harte, K. (1989). Ergonomic Perspective of Flight Path Displays. *IEEE Control Systems Magazine*.
33. Wloka, M. & Greenfield, E. (1995). The Virtual Tricorder: A Unified Interface for Virtual Reality. In *Proceedings of the ACM Symposium on User Interface Software and Technology*, 39-40.