CHAPTER 10

Interacting with Visualizations

A good visualization is not just a static picture or a 3D virtual environment that we can walk through and inspect like a museum full of statues. A good visualization is something that allows us to drill down and find more data about anything that seems important. Ben Shneiderman has coined what he calls a "mantra" to guide visual information-seeking behavior and the interfaces that support it: "Overview first, zoom and filter, then details on demand," (Shneiderman, 1998). But in reality we are just as likely to see an interesting detail, zoom out to get an overview, find some related information in a lateral segue, and then zoom in again to get the details of the original object of interest. The important point is that a good computer-based visualization is an interface that can support all of these activities. Ideally, every data object on a screen will be active and not just a blob of color on the screen. It will be capable of displaying more information as needed, disappearing when not needed, and accepting user commands to help with the thinking processes.

Interactive visualization is a process made up of a number of interlocking feedback loops that fall into three broad classes. At the lowest level is the *data manipulation loop*, through which objects are selected and moved using the basic skills of eye–hand coordination. Delays of even a fraction of a second in this interaction cycle can seriously disrupt the performance of higher-level tasks. At an intermediate level is an *exploration and navigation loop*, through which an analyst finds his or her way in a large visual data space. As people explore a new town, they build a cognitive spatial model using key landmarks and paths between them; something similar occurs when they explore data spaces.

But exploration can be generalized to more abstract searching operations. Kirsh and Maglio (1994) define a class of epistemic actions as activities whereby someone hopes to better understand or perceive a problem. At the highest level is a *problem-solving loop* through which the analyst forms hypotheses about the data and refines them through an augmented visualization process. The process may be repeated through multiple visualization cycles as new data is added, the problem is reformulated, possible solutions are identified, and the visualization is revised or

replaced. Sometimes the visualization may act as a critical externalization of the problem, forming a crucial extension of the cognitive process.

This chapter deals with two of the three loops: low-level interaction and exploration. General problem solving is discussed in Chapter 11.

Data Selection and Manipulation Loop

There are a number of well established "laws" that describe the simple, low-level control loops needed in tasks such as the visual control of hand position or the selection of an object on the screen.

Choice Reaction Time

Given an optimal state of readiness, with a finger poised over a button, a person can react to a simple visual signal in about 130 msec (Kohlberg, 1971). If the signals are very infrequent, the time can be considerably longer. Warrick et al. (1964) found reaction times as long as 700 msec under conditions such that there could be as much as two days between signals. The participants were engaged in routine typing, so they were at least positioned appropriately to respond. If people are not positioned at workstations, their responses will naturally take longer.

Sometimes, before someone can react to a signal, he or she must make a choice. A simple choice reaction-time task might involve pressing one button if a red light goes on and another if a green light goes on. This kind of task has been studied extensively. It has been discovered that reaction times can be modeled by a simple rule called the *Hick–Hyman law* for choice reaction time (Hyman, 1953).

According to this law,

$$\text{Reaction time} = a + b \log_2(C) \tag{10.1}$$

where C is the number of choices and a and b are empirically determined constants. The expression $\log_2(C)$ represents the amount of information processed by the human operator, expressed in bits of information.

Many factors have been found to affect choice reaction time—the distinctness of the signal, the amount of visual noise, stimulus–response compatibility, and so on—but under optimal conditions, the response time per bit of information processed is about 160 msec plus the time to set up the response. Thus, if there are eight choices (3 bits of information), the response time will typically be on the order of the simple reaction time plus approximately 480 msec. Another impor-

tant factor is the degree of accuracy required—people respond faster if they are allowed to make mistakes occasionally, and this effect is called a *speed–accuracy tradeoff*. For a useful overview of factors involved in determining reaction time, see Card et al. (1983).

2D Positioning and Selection

In highly interactive visualization applications, it is useful to have graphical objects function not only as program output—a way of representing data—but also as program input, a way of finding out more about data.

Selection using a mouse or some similar input device (such as a joystick or trackball) is one of the most common interactive operations in the modern graphical user interface, and it has been extensively studied. A simple mathematical model provides a useful estimation of the time taken to select a target that has a particular position and size:

$$\text{Selection time} = a + b \log_2(D/W + 1.0) \qquad \textbf{(10.2)}$$

where D is the distance to the center of the target, W is the width of the target, and a and b are constants determined empirically. These are different for different devices.

This formula is known as Fitts' law, after Paul Fitts (1954). The term $\log_2 (D/W + 1.0)$ is known as the *index of difficulty* (ID). The value $1/b$ is called the *index of performance* (IP) and is given in units of bits per second. There are a number of variations in the index-of-difficulty expression, but the one given here is the most robust (MacKenzie, 1992). Typical IP values for measured performance made with the fingertip, the wrist, and the forearm are all in the vicinity of 4 bits per second (Balakrishnan and MacKenzie, 1997). To put this into perspective, consider moving a cursor 16 cm across a screen to a small (0.5 cm) target. The index of difficulty will be about 5 bits. The selection will take more than a second longer than selecting a target that is already under the cursor.

Fitts' law can be thought of as describing an iterative process of eye–hand coordination, as illustrated in Figure 10.1. The human starts by judging the distance to the target and initiates the hand movement. On successive iterations, a corrective adjustment is made to the hand movement based on visual feedback showing the cursor position. The number of iterations around the control loop increases both as the distance to the target gets larger and as the size of the target gets smaller. The logarithmic nature of the relationship derives from the fact that on each iteration, the task difficulty is reduced in proportion to the remaining distance.

In many of the more complex data visualization systems, as well as in experimental data visualization systems using 3D virtual-reality (VR) technologies, there is a significant lag between a hand movement and the visual feedback provided on the display (Liang et al., 1991; Ware and Balakrishnan, 1994).

Fitts' law, modified to include lag, looks like this:

$$\text{Mean time} = a + b\,(\text{Human Time} + \text{MachineLag})\log_2(D/W + 1.0) \qquad (10.3)$$

According to this equation, the effects of lag increase as the target gets smaller. Because of this, a fraction-of-a-second lag can result in a subject's taking several seconds longer to perform a simple selection task. This may not seem like much, but in a VR environment intended to make everything seem easy and natural, lag can make the simplest task difficult.

Fitts' law is part of ISO standard 9214-9, which sets out protocols for evaluating user performance and comfort when using pointing devices with visual display terminals. It is invaluable as a tool for evaluating potential new input devices.

Hover Queries

The most common kind of selection action with a computer is done by dragging a cursor over an object and clicking the mouse button. The *hover query* dispenses with the mouse click. Extra information is revealed about an object when the mouse cursor passes over it. Usually it is implemented with a delay; for example, the function of an icon is shown by a brief text message after

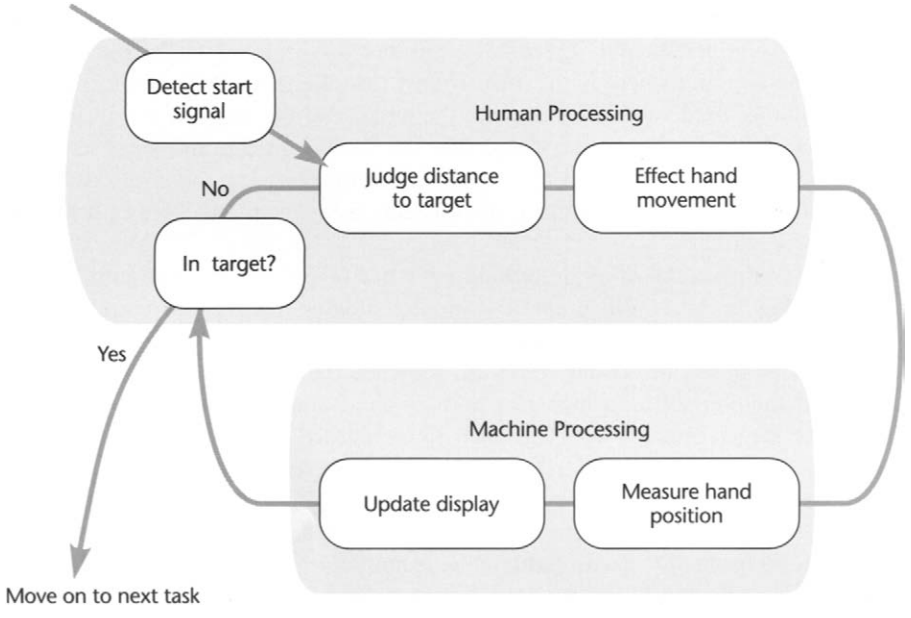

Figure 10.1 The visually guided reaching control loop. The human processor makes adjustments based on visual feedback provided by the computer.

hovering for a second or two. However, a hover query can function without a delay, making it very fast. The enables the mouse cursor to be dragged over a set of data objects, rapidly revealing the data contents and perhaps allowing an interactive query rate of several per second in special circumstances.

Path Tracing

Fitts' law deals with single, discrete actions, such as reaching for an object. Other tasks, such as tracing a curve or steering a car, involve continuous ongoing control. In such tasks, we are continually making a series of corrections based on visual feedback about the results of our recent actions. Accot and Zhai (1997) used Fitts' law to derive a prediction about continuous steering tasks. Their derivation revealed that the speed at which tracing could be done should be a simple function of the width of the path:

$$v = W/\tau \tag{10.4}$$

where v is the velocity, W is the path width, and τ is a constant that depends on the motor control system of the person doing the tracing. In a series of experiments, the researchers found an almost perfect linear relationship between speed of path-following and the path width, confirming their theory. The actual values of τ lay between .05 and .11 sec, depending on the specific task. To make this more concrete, consider the problem of tracing a pencil along a 2 mm–wide path. Their results suggest that this will be done at a rate of between 1.8 and 4 cm/sec.

Two-Handed Interaction

In most computer interfaces, users select and move graphical objects around the screen with a mouse held in one hand, leaving the other unoccupied. But in interacting with the everyday world, we frequently use both our hands. This leads us to the question of how we might make the computer interface better by taking advantage of both hands (Buxton and Myers, 1986).

The most important principle that has been discovered relating to the way tasks should be allocated to the two hands is Guiard's *kinematic chain theory* (Guiard, 1987). According to this theory, the left hand and the right hand form a kinematic chain, with the left hand providing a frame of reference for movements with the right, in right-handed individuals. For example, if we sculpt a small object out of modeling clay, we are likely to hold it in the left hand and do the detailed shaping with the right. The left hand reorients the piece and provides the best view, whereas the right pokes and prods within that frame of reference.

A number of interface designers have incorporated this principle into demonstrably superior interfaces for various tasks (Bier et al., 1993, Kabbash et al., 1994). For example, in an innovative computer-based drawing package, Kurtenbach et al. (1997) showed how templates, such as the French curve, could be moved rapidly over a drawing with the left hand while an artist used his or her right hand to paint around the shape.

Another way that using the left hand can be beneficial is in positioning tools for easy access. In interactive drawing packages, users spend a lot of time moving between the drawing and various menus, positioned off to the side of the screen. The *toolglass* and *magic lens* approach, developed by Bier et al. (1993), got around this problem by allowing users to use the left hand to position tool palettes and the right hand to do normal drawing operations. This allowed for very quick changes in color or brush characteristics. As an additional design refinement, they also made some of the tools transparent (hence toolglasses).

In an application more relevant to information visualization, Stone et al. (1994) developed the magic lens idea as a set of interactive information filters implemented as transparent windows that the user can move over an information visualization with the left hand. The magic lens could be programmed to be a kind of data X-ray, revealing normally invisible aspects of the data. For example, a magic lens view of a map might show some or all of the regions with high rainfall, or alternatively, the geology underneath. In the magic lens design, the right hand can be used in a conventional way, to control a cursor that can then be used to click within the magic lens, to make selections or position objects.

Learning

Over time, people become more skilled at any task, barring fatigue, sickness, or injury. A simple expression known as the *power law of practice* describes the way task performance speeds up over time.

$$\log(T_n) = C - \alpha \log(n) \tag{10.5}$$

where $C = \log(T_1)$ is based on the time to perform the task on the first trial and, T_n is the time required to perform the nth trial, and α is a constant that represents the steepness of the learning curve.

One of the ways in which skilled performance is obtained is through the *chunking* of small subtasks into programmed motor procedures. The beginning typist must make a conscious effort to hit the letters *t, h,* and *e* when typing the word *the,* but the brains of experienced typists can execute preprogrammed bursts of motor commands so that the entire word can be typed with a single mental command to the motor cortex. Skill learning is characterized by more and more of the task becoming automated and encapsulated. To encourage skill automation, the computer system should provide rapid and clear feedback of the consequences of user actions (Hammond, 1987).

Control Compatibility

Some control movements are easier to learn than others, and this depends heavily on prior experience. If you move a computer mouse to the right, causing an object on the screen to move to

the right, this positioning method will be easy to learn. A skill is being applied that was gained very early in life and has been refined ever since. But if the system interface has been created such that a mouse movement to the right causes a graphical object to move to the left, this will be incompatible with everyday experience and positioning the object will be difficult. In the behaviorist tradition of psychology, this factor is generally called *stimulus–response (S–R) compatibility.* In modern cognitive psychology, the effects of S–R compatibility are readily understood in terms of skill learning and skill transfer.

In general, it will be easier to execute tasks in computer interfaces if the interfaces are designed in such a way that they take advantage of previously learned ways of doing things. Nevertheless, some inconsistencies are easily tolerated, whereas others are not. For example, many user interfaces amplify the effect of a mouse movement so that a small hand movement results in a large cursor movement. Psychologists have conducted extensive experiments that involve changing the relationship between eye and hand. If a prism is used to laterally displace what is seen relative to what is felt, people can adapt in minutes or even seconds (Welch and Cohen, 1991). This is like using a mouse that is laterally displaced from the screen cursor being controlled.

On the other hand, if people are asked to view the world inverted with a mirror, it can take weeks of adaptation for them to learn to operate in an upside-down world (Harris, 1965). Snyder and Pronko (1952) had subjects wear inverting prisms continuously for a month. At the end of this period, reaching behaviors seemed error-free, but the world still seemed upside-down. This suggests that if we want to achieve good eye–hand coordination in an interface, we do not need to worry too much about matching hand translation with virtual object translation, but we should worry about matching the axis or rotation.

Some imaginative interfaces designed for virtual reality involve extreme mismatches between the position of the virtual hand and the proprioceptive feedback from the user's body. In the Go-Go Gadget technique (named after the cartoon character, Inspector Gadget), the user's virtual hand is stretched out far beyond his or her actual hand position to allow for manipulation of objects at a distance (Poupyrev et al., 1996).

Studies by Ramachandran (1999) provide interesting evidence that even under extreme distortions people may come to act as if a virtual hand is their own, particularly if touch is stimulated. In one of Ramachandran's experiments, he hid a subject's hand behind a barrier and showed the subject a grotesque rubber Halloween hand. Next, he stroked and patted the subject's actual hand and the Halloween hand in exact synchrony. Remarkably, in a very short time, the subject came to perceive that the Halloween hand was his or her own. The strength of this identification was demonstrated when the researcher hit the Halloween hand with a hammer. The subjects showed a strong spike in galvanic skin response (GSR), indicating a physical sense of shock. No shock was registered without the stroking. The important point from the perspective of virtual reality interfaces is that even though the fake hand and the subjects' real hand were in quite different places, a strong sense of identification occurred.

Consistency with real-world actions is only one factor in skill learning. There are also the simple physical affordances of the task itself. It is easier for us to make certain body movements

than others. Very often we can make computer-mediated tasks easier to perform than their real-world counterparts. When designing a house, we do not need to construct it virtually with bricks and concrete. The magic of computers is that a single button click can often accomplish as much as a prolonged series of actions in the real world. For this reason, it would be naïve to conclude that computer interfaces should evolve toward VR simulations of real-world tasks or even enhanced Go-Go Gadget–style interactions.

Vigilance

A basic element of many interaction cycles is the detection of a target. Although several aspects of this have already been discussed in Chapter 5, a common and important problem remains to be covered—the detection of infrequently appearing targets.

The invention of radar during World War II created a need for radar operators to monitor radar screens for long hours, searching for visual signals representing incoming enemy aircraft. Out of this came a need to understand how people can maintain *vigilance* while performing monotonous tasks. This kind of task is common to airport baggage X-ray operators, industrial quality-control inspectors, and the operators of large power grids. Vigilance tasks commonly involve visual targets, although they can be auditory. There is extensive literature concerning vigilance (see Davies and Parasuraman, 1980, for a detailed review). Here is an overview of some of the more general findings, adapted from Wickens (1992):

1. Vigilance performance falls substantially over the first hour.
2. Fatigue has a large negative influence on vigilance.
3. To perform a difficult vigilance task effectively requires a high level of sustained attention, using significant cognitive resources. This means that dual tasking is not an option during an important vigilance task. It is not possible for operators to perform some useful task in their "spare time" while simultaneously monitoring for some difficult-to-perceive signal.
4. Irrelevant signals reduce performance. The more irrelevant visual information is presented to a person performing a vigilance task, the harder it becomes.

Overall, people perform poorly on vigilance tasks, but there are a number of techniques that can improve performance. One method is to provide reminders at frequent intervals about what the targets will look like. This is especially important if there are many different kinds of targets. Another is to take advantage of the visual system's sensitivity to motion. A difficult target for a radar operator might be a slowly moving ship embedded in a great many irrelevant noise signals. Scanlan (1975) showed that if a number of radar images are stored up and rapidly replayed, the image of the moving ship can easily be differentiated from the visual noise. Generally, anything that can transfer the visual signal into the optimal spatial or temporal range of the visual system should help detection. If the signal can be made perceptually different or distinct from irrelevant information, this will also help. The various factors that make color, motion, and texture distinct can all be applied. These are discussed in Chapters 4 and 5.

Exploration and Navigation Loop

View navigation is important in visualization when the data is mapped into an extended and detailed visual space. The problem is complex, encompassing theories of pathfinding and map use, cognitive spatial metaphors, and issues related to direct manipulation and visual feedback.

Figure 10.2 sketches the basic navigation control loop. On the human side is a cognitive logical and spatial model whereby the user understands the data space and his or her progress through it. If the data space is maintained for an extended period, parts of its spatial model may become encoded in long-term memory. On the computer side, the visualization may be updated and refined from data mapped into the spatial model.

Here, we start with the problem of 3D locomotion; next, we consider the problem of pathfinding, and finally move on to the more abstract problem of maintaining focus and context in abstract data spaces.

Locomotion and Viewpoint Control

Some data visualization environments show information in such a way that it looks like a 3D landscape, not just a flat map. This is done with remote sensing data from other planets, or with maps of the ocean floor or other data related to the terrestrial environment. The data landscape idea has also been applied to abstract data spaces such as the World Wide Web (see Figure 10.3 for an example). The idea is that we should find it easy to navigate through data presented in this way because we can harness our real-world spatial interpretation and navigation skills.

James Gibson (1986) offers an environmental perspective on the problem of perceiving for navigation:

> *A path affords pedestrian locomotion from one place to another, between the terrain features that prevent locomotion. The preventers of locomotion consist of obstacles, barriers, water margins, and brinks (the edges of cliffs). A path must afford footing; it must be relatively free of rigid foot-sized obstacles.*

Gibson goes on to describe the characteristics of obstacles, margins, brinks, steps, and slopes. According to Gibson, locomotion is largely about perceiving and using the affordances offered for navigation by the environment. (See Chapter 1 for a discussion of affordances). His perspective can be used in a quite straightforward way in designing virtual environments, much as we might design a public museum or a theme park. The designer creates barriers and paths in order to encourage visits to certain locations and discourage others.

We can also understand navigation in terms of the depth cues presented in Chapter 8. All the perspective cues are important in providing a sense of scale and distance, although the stereoscopic cue is important only for close-up navigation in situations such as walking through a crowd. When we are navigating at higher speed, in an automobile or a plane, stereoscopic depth is irrelevant, because the important parts of the landscape are beyond the range of stereoscopic

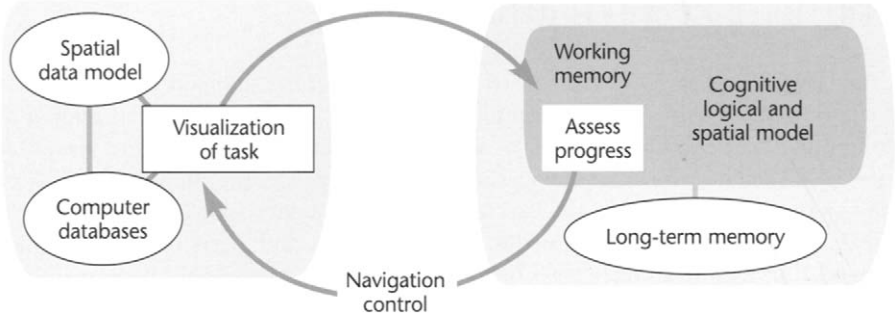

Figure 10.2 The navigation control loop.

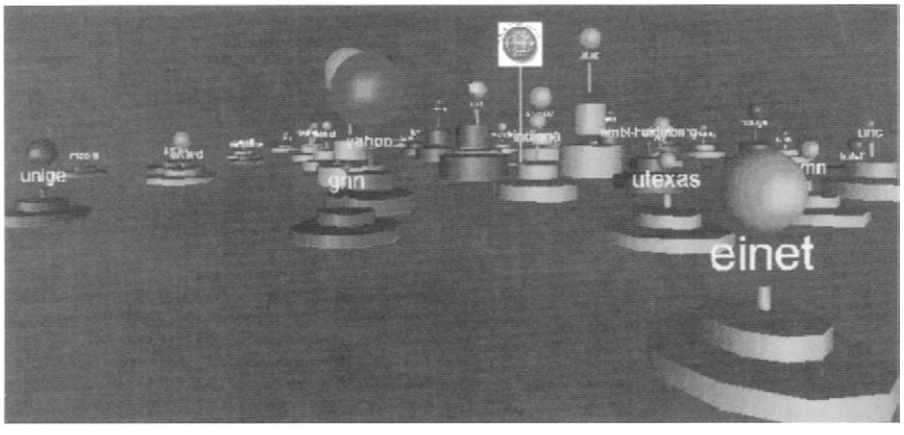

Figure 10.3 Web sites arranged as a data landscape (T. Bray, 1996).

discrimination. Under these conditions, structure-from-motion cues and information based on perceived objects of known size are critical.

It is usually assumed that smooth-motion flow of images across the retina is necessary for judgment of the direction of self-motion within the environment. But Vishton and Cutting (1995) investigated this problem using VR technology, with subjects moving through a forestlike virtual environment, and concluded that relative displacement of identifiable objects over time was the key, not smooth motion. Their subjects could do almost as well with a low frame rate, with images presented only 1.67 times per second, but performance declined markedly when updates were less than 1 per second. The lesson for the design of virtual navigation aids is that these environments should be sparsely populated with discrete but separately identifiable objects—there must be enough landmarks that several are always visible at any instant, and frame rates ideally should be

at least 2 per second. However, it should also be recognized that although judgments of heading are not impaired by low frame rates, other problems will result. Low frame rates cause lag in visual feedback and, as discussed previously, this can introduce serious performance problems.

Spatial Navigation Metaphors

Interaction metaphors are cognitive models for interaction that can profoundly influence the design of interfaces to data spaces. Here are two sets of instructions for different viewpoint control interfaces:

1. "Imagine that the model environment shown on the screen is like a real model mounted on a special turntable that you can grasp, rotate with your hand, move sideways, or pull towards you."

2. "Imagine that you are flying a helicopter and its controls enable you to move up and down, forward and back, left and right."

With the first interface metaphor, if the user wishes to look at the right-hand side of some part of the scene, she must rotate the scene to the left to get the correct view. With the second interface metaphor, the user must fly her vehicle forward, around to the right, while turning in toward the target. Although the underlying geometry is the same, the user interface and the user's conception of the task are very different in the two cases.

Navigation metaphors have two fundamentally different kinds of constraints on their usefulness. The first of these constraints is essentially cognitive. The metaphor provides the user with a model that enables the prediction of system behavior given different kinds of input actions. A good metaphor is one that is apt, matches the system well, and is also easy to understand. The second constraint is more of a physical limitation. A particular metaphor will naturally make some actions physically easy to carry out, and others difficult to carry out, because of its implementation. For example, a walking metaphor limits the viewpoint to a few feet above ground level and the speed to a few meters per second. Both kinds of constraints are related to Gibson's concept of affordances—a particular interface affords certain kinds of movement and not others, but it must also be *perceived* to embody those affordances.

Note that, as discussed in Chapter 1, we are going beyond Gibson's view of affordances here. Gibsonian affordances are properties of the *physical* environment. In computer interfaces, the physical environment constitutes only a small part of the problem, because most interaction is mediated through the computer and Gibson's concept as he framed it does not strictly apply. We must extend the notion of affordances to apply to both the physical characteristics of the user interface and the representation of the data. A more useful definition of an interface with the right affordances is one that makes the possibility for action plain to the user and gives feedback that is easy to interpret.

Four main classes of metaphors have been employed in the problem of controlling the viewpoint in virtual 3D spaces. Figure 10.4 provides an illustration and summary. Each metaphor has a different set of affordances.

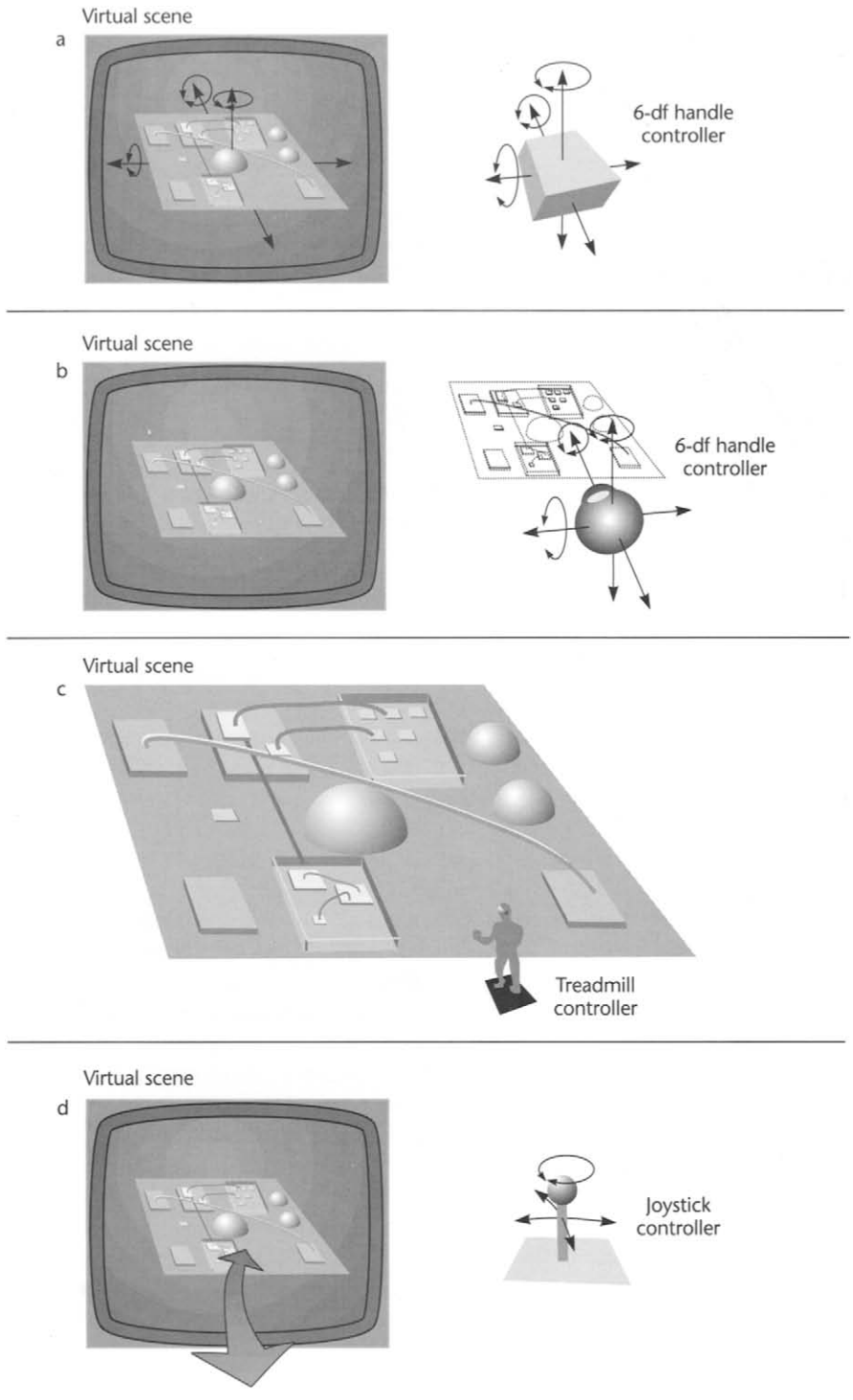

Figure 10.4 Four different navigation metaphors: (a) World-in-hand. (b) Eyeball-in-hand. (c) Walking. (d) Flying.

1. World-in-hand. The user metaphorically grabs some part of the 3D environment and moves it (Houde, 1992; Ware and Osborne, 1990). Moving the viewpoint closer to some point in the environment actually involves pulling the environment closer to the user. Rotating the environment similarly involves twisting the world about some point as if it were held in the user's hand. A variation on this metaphor has the object mounted on a virtual turntable or gimbal. The world-in-hand model would seem to be optimal for viewing discrete, relatively compact data objects, such as virtual vases or telephones. It does not provide affordances for navigating long distances over extended terrains.

2. Eyeball-in-hand. In the eyeball-in-hand metaphor, the user imagines that she is directly manipulating her viewpoint, much as she might control a camera by pointing it and positioning it with respect to an imaginary landscape. The resulting view is represented on the computer screen. This is one of the least effective methods for controlling the viewpoint. Badler et al. (1986) observed that "consciously calculated activity" was involved in setting a viewpoint. Ware and Osborne (1990) found that although some viewpoints were easy to achieve, others led to considerable confusion. They also noted that with this technique, physical affordances are limited by the positions in which the user can physically place her hand. Certain views from far above or below cannot be achieved or are blocked by the physical objects in the room.

3. Walking. One way of allowing inhabitants of a virtual environment to navigate is simply to let them walk. Unfortunately, even though a large extended virtual environment can be created, the user will soon run into the real walls of the room in which the equipment is housed. Most VR systems require a handler to prevent the inhabitant of the virtual world from tripping over the real furniture. A number of researchers have experimented with devices like exercise treadmills so that people can walk without actually moving. Typically, something like a pair of handlebars is used to steer. In an alternative approach, Slater et al. (1995) created a system that captures the characteristic up-and-down head motion that occurs when people walk in place. When this is detected, the system moves the virtual viewpoint forward in the direction of head orientation. This gets around the problem of bumping into walls, and may be useful for navigating in environments such as virtual museums. However, the affordances are still restrictive.

4. Flying. Modern digital terrain visualization packages commonly have fly-through interfaces that enable users to smoothly create an animated sequence of views of the environment. Some of these are more literal, having aircraftlike controls. Others use the flight metaphor only as a starting point. No attempt is made to model actual flight dynamics; rather, the goal is to make it easy for the user to get around in 3D space in a relatively unconstrained way. For example, we (Ware and Osborne 1990) developed a flying interface that used simple hand motions to control velocity. Unlike real aircraft, this interface makes it as easy to move up, down, or backward as it is to move forward. They reported that subjects with actual flying experience had the most difficulty; because of

their expectations about flight dynamics, pilots did unnecessary things such as banking on turns and were uncomfortable with stopping or moving backward. Subjects without flying experience were able to pick up the interface more rapidly. Despite its lack of realism, this was rated as the most flexible and useful interface when compared to others based on the world-in-hand and eyeball-in-hand metaphors.

The optimal navigation method depends on the exact nature of the task. A virtual walking interface may be the best way to give a visitor a sense of presence in an architectural space; something loosely based on the flying metaphor may be a more useful way of navigating through spatially extended data landscapes. The affordances of the virtual data space, the real physical space, and the input device all interact with the mental model of the task that the user has constructed.

Wayfinding, Cognitive, and Real Maps

In addition to the problem of moving through an environment in real time, there is the metalevel problem of how people build up an understanding of larger environments over time. This problem is usually called *wayfinding*. It encompasses both the way in which people build mental models of extended spatial environments and the way they use physical maps as aids to navigation.

Unfortunately, this area of research is plagued with a diversity of terminology. Throughout the following discussion, bear in mind that there are two clusters of concepts, and the differences between these clusters relate to the dual coding theory discussed in Chapter 9.

One cluster includes the related concepts of declarative knowledge, procedural knowledge, topological knowledge, and categorical representations. These concepts are fundamentally logical and nonspatial.

The other cluster includes the related concepts of spatial cognitive maps and coordinate representations. These are fundamentally spatial.

Seigel and White (1975) proposed that there are three stages in the formation of wayfinding knowledge. First, information about key landmarks is learned; initially there is no spatial understanding of the relationships between them. This is sometimes called *declarative knowledge*. We might learn to identify a post office, a church, and the hospital in a small town.

Second, *procedural knowledge* about routes from one location to another is developed. Landmarks function as decision points. Verbal instructions often consist of procedural statements related to landmarks, such as "Turn left at the church, go three blocks, and turn right by the gas station." This kind of information also contains *topological knowledge,* because it includes connecting links between locations. Topological knowledge has no explicit representation of the spatial position of one landmark relative to another.

Third, a *cognitive spatial map* is formed. This is a representation of space that is two-dimensional and includes quantitative information about the distances between the different locations of interest. With a cognitive spatial map, it is possible to estimate the distance between any two points, even though we have not traveled directly between them, and to make statements such as "The university is about one kilometer northwest of the train station."

In Seigel and White's initial theory and in much of the subsequent work, there has been a presumption that spatial knowledge developed strictly in the order of these three stages: declarative knowledge, procedural knowledge, and cognitive spatial maps. Recent evidence from a study by Colle and Reid (1998) contradicts this. They conducted an experimental study using a virtual building consisting of a number of rooms connected by corridors. The rooms contained various objects. In a memory task following the exploration of the building, subjects were found to be very poor at indicating the relative positions of objects located in different rooms, but they were good at indicating the relative positions of objects within the same room. This suggests that cognitive spatial maps form easily and rapidly in environments where the viewer can see everything at once is the case for objects within a single room. It is more likely that the paths from room to room were captured as procedural knowledge. The practical application of this is that overviews should be provided wherever possible in extended spatial information spaces.

The results of Colle and Reid's study fit well with a somewhat different theory of spatial knowledge proposed by Kosslyn (1987). He suggested that there are only two kinds of knowledge, not necessarily acquired in a particular order. He called them *categorical* and *coordinate* representations. For Kosslyn, categorical information is a combination of both declarative knowledge and topological knowledge, such as the identities of the landmarks and the paths between them. Coordinate representation is like the cognitive spatial map proposed by Seigel. A spatial coordinate representation would be expected to arise from the visual imagery obtained with an overview. Conversely, if knowledge were constructed from a sequence of turns along corridors when the subject was moving from room to room, the natural format would be categorical.

Landmarks provide the links between categorical and spatial coordinate representations. They are important both for cognitive spatial maps and for topological knowledge about routes. Vinson (1999) created a generalized classification of landmarks based on Lynch's classification (1960) of the "elements" of cognitive spatial maps. Figure 10.5 summarizes Vinson's design guidelines for the different classes of landmarks. This broad concept includes paths between locations, edges of geographical regions, districts, nodes such as public squares, and the conventional ideal of a point landmark such as a statue.

Vinson also created a set of design guidelines for landmarks in virtual environments. The following rules are derived from them:

- There should be enough landmarks that a small number are visible at all times.
- Each landmark should be visually distinct from the others.
- Landmarks should be visible and recognizable at all navigable scales.
- Landmarks should be placed on major paths and at intersections of paths.

Creating recognizable landmarks in 3D environments can be difficult because of multiple viewpoints. Darken et al. (1998) reported that Navy pilots typically fail to recognize landmark terrain features on a return path, even if these were identified correctly on the outgoing leg of a low-flying exercise. This suggests that terrain features are not encoded in memory as fully three-

Lynch's Types	Examples	Functions
Paths	Street, canal, transit line	Channel for navigator movement
Edges	Fence, riverbank	Indicates district limits
Districts	Neighborhood	Reference region
Nodes	Town square, public building	Focal point for travel
Landmarks	Statue	Reference point into which we cannot enter

Figure 10.5 The functions of different kinds of landmarks in a virtual environment. *Adapted from Vinson (1999).*

dimensional structures, but rather are remembered in some viewpoint-dependent fashion. (See Chapter 7 for a discussion of viewpoint-dependent object memory.)

An interesting way to assist users in the encoding of landmarks for navigation in 3D environments was developed by Elvins et al. (1997). They presented subjects with small 3D subparts of a virtual cityscape that they called *worldlets*, as illustrated in Figure 10.6. The worldlets provided 3D views of key landmarks, presented in such a way that observers could rotate them to obtain a variety of views. Subsequently, when they were tested in a navigation task, subjects who had been shown the worldlets performed significantly better than subjects who had been given pictures of the landmarks, or subjects who had simply been given verbal instructions.

Cognitive maps can also be acquired directly from an actual map much more rapidly than by traversing the terrain. Thorndyke and Hayes-Roth (1982) compared people's ability to judge distances between locations in a large building. Half of them had studied a map for half an hour or so, whereas the other half never saw a map but had worked in the building for many months. The results showed that for estimating the straight-line Euclidean distance between two points, a brief experience with a map was equivalent to working in the building for about a year. However, for estimating the distance along the hallways, the people with experience in the building did the best.

To understand map-reading skills, Darken and Banker (1998) turned to orienteering, a sport that requires athletes to run from point to point over rugged and often difficult terrain with the aid of a map. Experienced orienteers are skilled map readers. One cognitive phenomenon the researchers observed was related to an initial scaling error rapidly remedied; they observed that "initial confusion caused by a scaling error is followed by a 'snapping' phenomenon where the

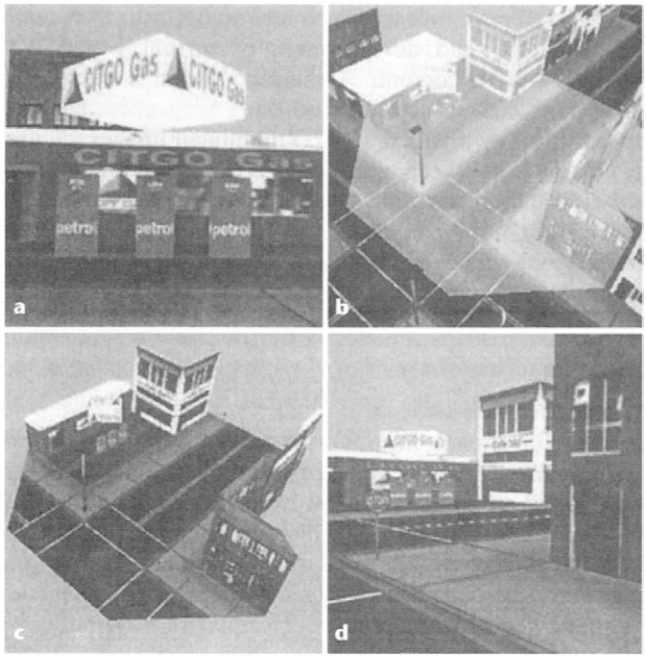

Figure 10.6 Elvins et al. (1998) conceived the idea of worldlets as navigation aids. Each worldlet is a 3D representation of a landmark in a spatial landscape. (a) A straight-on view of the landmark. (b) The region extracted to create the worldlet. (c) The worldlet from above. (d) The worldlet from street level. Worldlets can be rotated to facilitate later recognition from an arbitrary viewpoint.

world that is seen is instantaneously snapped into congruence with the mental representation" (Darken et al., 1998). This suggests that wherever possible, aids should be given to identify matching points on both an overview map and a focus map.

Frames of Reference

The ability to generate and use something cognitively analogous to a map can be thought of as applying another perspective or *frame of reference* to the world. A map is like a view from above. Cognitive frames of reference are often classified into *egocentric* and *exocentric*. According to this classification, a map is just one of many exocentric views—views that originate outside of the user.

The egocentric frame of reference is, roughly speaking, our subjective view of the world. It is anchored to the head or torso, not the direction of gaze (Bremmer et al., 2001). Our sense of what is ahead, left, and right does not change as we rapidly move our eyes around the scene, but it does change with body and head orientation.

As we explore the world, we change our egocentric viewpoint primarily around two axes of rotation. We turn our bodies mostly around a vertical axis (pan) to change heading, and swivel our heads on the neck (also pan) about a similar vertical axis for more rapid adjustments in view direction. We also tilt our heads forward and back, but generally not to the side (roll). Thus, human angle of view control normally has only two degrees of freedom. These heading (pan) and tilt rotational degrees of freedom are illustrated in Figure 10.7.

A consequence of the fact that we are most familiar with only two of the three degrees of freedom of viewpoint rotation is that when displaying maps, either real or in a virtual environment, we are most comfortable with only two degrees of freedom of rotation. Figure 10.8 illus-

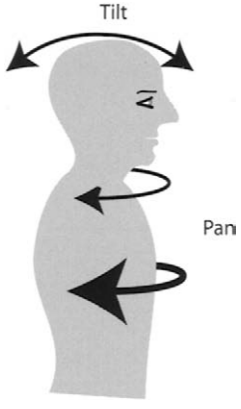

Figure 10.7 Primary rotation axes of egocentric coordinates.

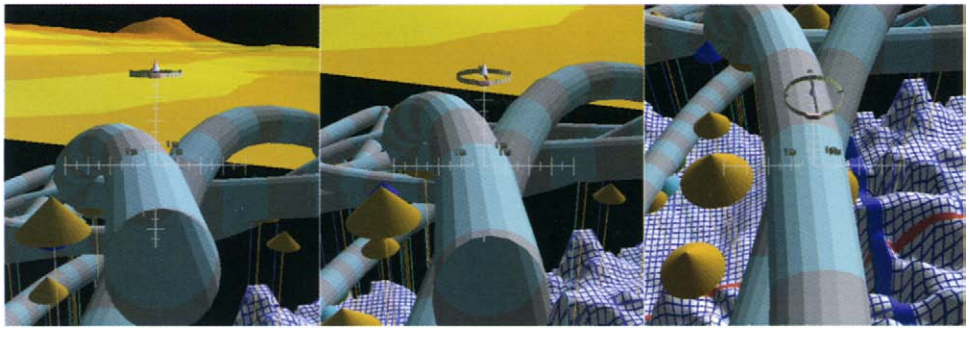

Figure 10.8 View-control widgets for examining geographic data. Note that the rotational degrees of freedom match the rotational degrees of freedom of egocentric coordinates. The three views show different amounts of tilt. The handle on the top widgets can be dragged left and right around the ring to change the view heading.

trates an interface for rotating geographical information spaces constructed to have the same two degrees of freedom (Ware et al., 2001). The widgets allow rotation around the center point (equivalent to turning the body) and tilt from horizontal up into the plane of the screen (equivalent to forward-and-back head tilt), but they do not allow rotation about the line of site through the center of the screen (equivalent to the rarely used sideways head tilt).

Because we tend to move our bodies forward, and only rarely sideways, a simple interface to simulate human navigation can be constructed with only three degrees of freedom, two for rotations (heading and tilt) and one to control forward motion in the direction of heading. If a fourth degree of freedom is added, it may be most useful to allow for something analogous to head turning. This allows for sideways glances while traveling forward.

The term *exocentric* simply means external. In 3D computer graphics, exocentric frames of reference are used for applications such as monitoring avatars in video games, controlling virtual cameras in cinematography, and monitoring the activities of remote or autonomous vehicles. Obviously, there is an infinity of exocentric views. The following is a list of some of the more important and useful ones.

Another person's view For some tasks, it can be useful to take the egocentric view of someone else who is already present in our field of view. Depending on the angular disparity in the relative directions of gaze, this can be confusing, especially when the other person is facing us. In the ClearBoard system (Ishii and Kobayashi, 1992), a remote collaborator appeared to be writing on the other side of a pane of glass. By digitally reversing the image, a common left–right frame of reference was maintained.

Over-the-shoulder view A view from just behind and to the side of the head of an individual. This view is commonly used in cinematography.

God's-eye view Following a vehicle or avatar from above and behind. Figure 10.9(a) illustrates. This view is very common in video games. Because it provides a wider field of

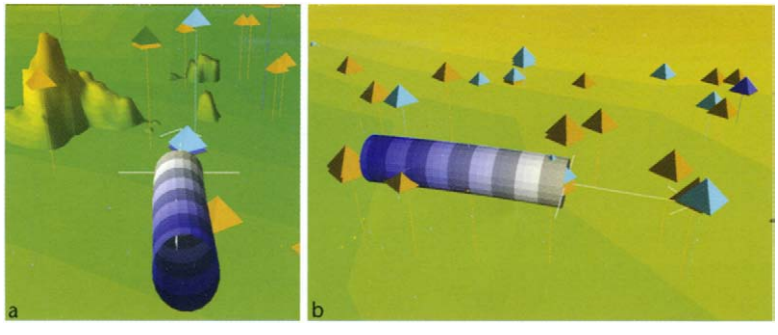

Figure 10.9 (a) God's-eye view of a moving vehicle shown by the tube object in the foreground. (b) Wingman's view of the same vehicle.

view, it can be better for steering a remote vehicle than the more obvious choice, an egocentric view from the vehicle itself (Wang and Milgram, 2001).

Wingman's view Following a vehicle or avatar while looking at it from the side. Figure 10.9(b) illustrates. Exocentric views that follow a moving object, such as the God's-eye or wingman's views, are sometimes called *tethered* (Wang and Milgram, 2001).

Map view A top-down view.

Whether an egocentric or an exocentric frame of reference is likely to be most useful depends on the task (McCormick et al., 1998). Some tasks, such as steering a virtual vehicle, are better done with an egocentric view or an exocentric God's-eye tethered view. Other tasks involving global spatial awareness, such as estimating the distance between a set of objects, can be performed better with an exocentric map view.

If we have multiple views simultaneously, then the links between views can be made visually explicit (Ware and Lewis, 1995; Plumlee and Ware, 2003). Figure 10.10 illustrates

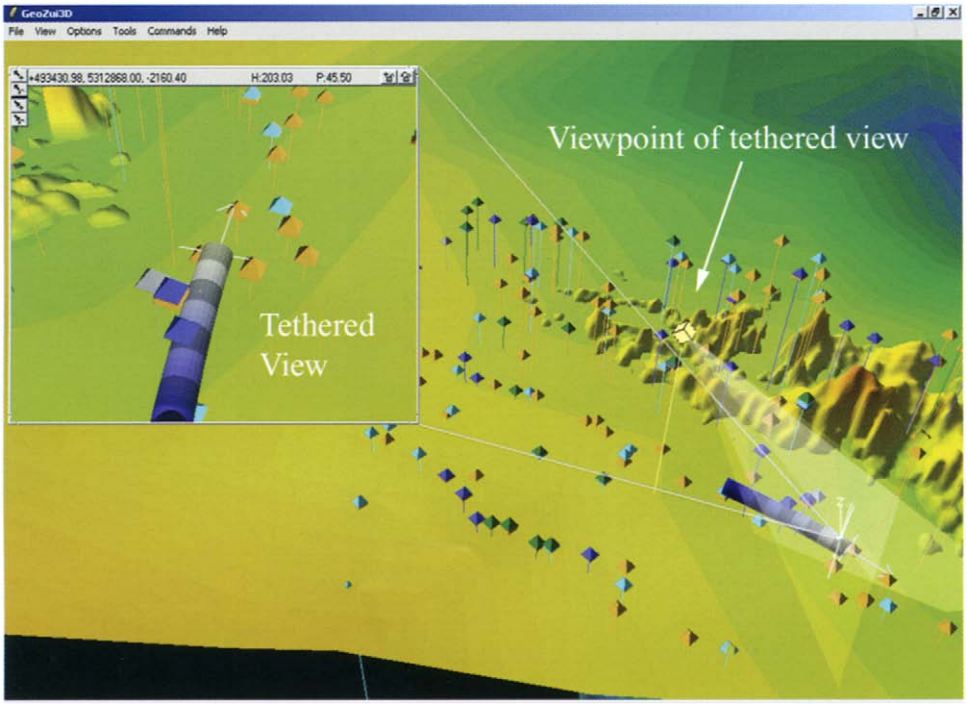

Figure 10.10 The subwindow in the upper left corner provides a tethered view. The overview contains a number of graphical devices to make the tethering explicit (Plumlee and Ware, 2003).

graphical methods for showing where the tethered view is with respect to a larger overview. These include a viewpoint proxy, a transparent pyramid showing the direction and angle of the tethered view, and lines that visually link the secondary window with its source.

None of the exocentric views has been studied as much as the map view.

Map Orientation

How should a map be displayed? Two alternatives have been extensively studied: the track-up display, shown in Figure 10.11(b), and the north-up display, shown in Figure 10.11(a). A *track-up map* is oriented so that the straight-ahead direction, from the point of view of the navigator, is the up direction on the map. The second alternative is to display the map so that north is always up, at the top of the map.

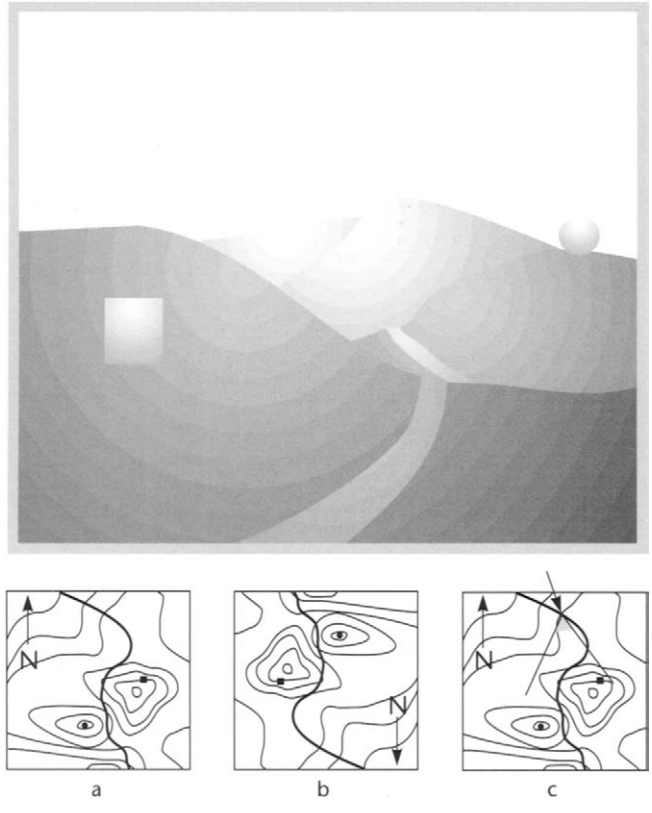

Figure 10.11 (a) North-up map. (b) Track-up map. (c) North-up map with user view explicitly displayed.

One way of considering the map orientation problem is in terms of control compatibility. Imagine yourself in a car, driving south from Berlin, Germany, to Rome, Italy. With a north-up map, a right turn becomes a left direction on the map. Many people find this confusing and reorient the map, even though this means that the place names are upside-down. Experimental studies of map use confirm this, showing that fewer errors are made when subjects use a track-up map (Eley, 1988). However, the north-up map does have advantages. Expert navigators often prefer this orientation because it provides a common frame of reference for communicating with someone else.

It is possible to enhance a north-up map and make it almost as effective as a track-up map, even for novices. Aretz (1991) provided a north-up map for helicopter navigators, but with the addition of a clear indicator of the forward field of view of the navigator. This significantly enhanced the ability of the user to orient himself or herself. Figure 10.11(c) illustrates this kind of enhanced map.

Supporting Visualizations with Maps

The research suggests a number of ways that visualizations can be enhanced with maps:

- Overview maps should be provided when an information space is large. Given how hard it is to build up a mental map by exploring an environment, an overview can substantially reduce the cognitive load.
- User location and direction of view within the map should be indicated. A common way of doing this is with a *You are here* arrow.
- Imagery of key landmarks should be provided. A landmark image on a map should be constructed from a viewpoint that will occur when the wayfinder encounters the actual landmark.

It should be borne in mind that procedural instructions can be more useful than a map when the task itself requires navigating from landmark to landmark. In this case, the cognitive representation of the task is likely to be topological. If the problem is to guide a user from node to node through a virtual information space, providing a sequence of instructions may be more appropriate than providing a map. A verbal or written set of procedural instructions can also be enhanced with landmark imagery.

Focus, Context, and Scale

We have been dealing with the problem of how people navigate through 3D data spaces, under the assumption that the methods used should reflect the way we navigate in the real world. The various navigation metaphors are all based on this assumption. However, there are a number of successful spatial navigation techniques that do not use an explicit interaction metaphor, but do involve visual spatial maps. These techniques make it easy to move rapidly between views at different scales; because of this, they are said to solve the *focus–context* problem. If we think of the

problem of wayfinding as one of discovering specific objects or locations in a larger landscape, the focus–context problem is simply a generalization of this, the problem of finding detail in a larger context. The focus–context problem is not always spatial; there are also structural and temporal variations.

Spatial scale Spatial-scale problems are common to all mapping applications. For example, a marine biologist might wish to understand the spatial behavior of individual codfish within a particular school off the Grand Banks of Newfoundland. This information is understood in the context of the shape of the continental shelf and the boundary between cold Arctic water and the warm waters of the Gulf Stream.

Structural scale Complex systems can have structural components at many levels. A prime example is computer software. This has structure within a single line of code, structure within a subroutine or procedure (perhaps 50 lines of code), structure at the object level for object-oriented code (perhaps 1000 lines of code), structure at the packet level, and structure at the system level. Suppose that we wish to visualize the structure of a large program, such as a digital telephone switch (comprising as many as 20 million lines of code); we may wish to understand its structure through as many as six levels of detail.

Temporal scale Many data visualization problems involve understanding the timing of events at very different scales. For example, in understanding data communications, it can be useful to know the overall traffic patterns in a network as they vary over the course of a day. It can also be useful to follow the path of an individual packet of information through a switch over the course of a few microseconds.

It is worth noting that the focus–context problem has already been solved by the human visual system. The brain continuously integrates detailed information from successive fixations of the fovea with the less detailed information that is available at the periphery. This is combined with data coming from the prior sequence of fixations. For each new fixation, the brain must somehow match key objects in the previous view with those same objects moved to new locations. Differing levels of detail are supported in normal perception because objects are seen at much lower resolution at the periphery of vision than in the fovea. Because we have no difficulty in recognizing objects at different distances, this also means that scale-invariance operations are supported in normal perception. The best solutions to the problem of providing focus and context in a display are likely to take advantage of these perceptual capabilities.

Although the spatial scale of a map, the structural levels of detail of a computer program, and the temporal scale in communications monitoring are very different application domains, they can all be represented by means of spatial layouts of data and they belong to a class of related visualization problems. The same interactive techniques can usually be applied. In the following sections, we consider the perceptual properties of four different visualization techniques to solve the focus–context problem: distortion, rapid zooming, elision, and multiple windows.

Distortion Techniques

A number of techniques have been developed that spatially distort a data representation, giving more room to designated points of interest and decreasing the space given to regions away from those points. What is of specific interest is spatially expanded at the expense of what is not, thus providing both focus and context. Figure 10.12 illustrates one such method, called *intelligent zooming* (Bartram et al., 1994). Parts of the graph are dynamically repositioned and resized based on selected points of interest, and selected nodes are expanded to show their contents. Some techniques have been designed to work with a single focus, such as the hyperbolic tree browser (Lamping et al., 1995), shown in Figure 10.13. Others allow multiple foci to be simultaneously expanded, for example, the table lens (Rao and Card, 1994) illustrated in Figure 10.14. Many

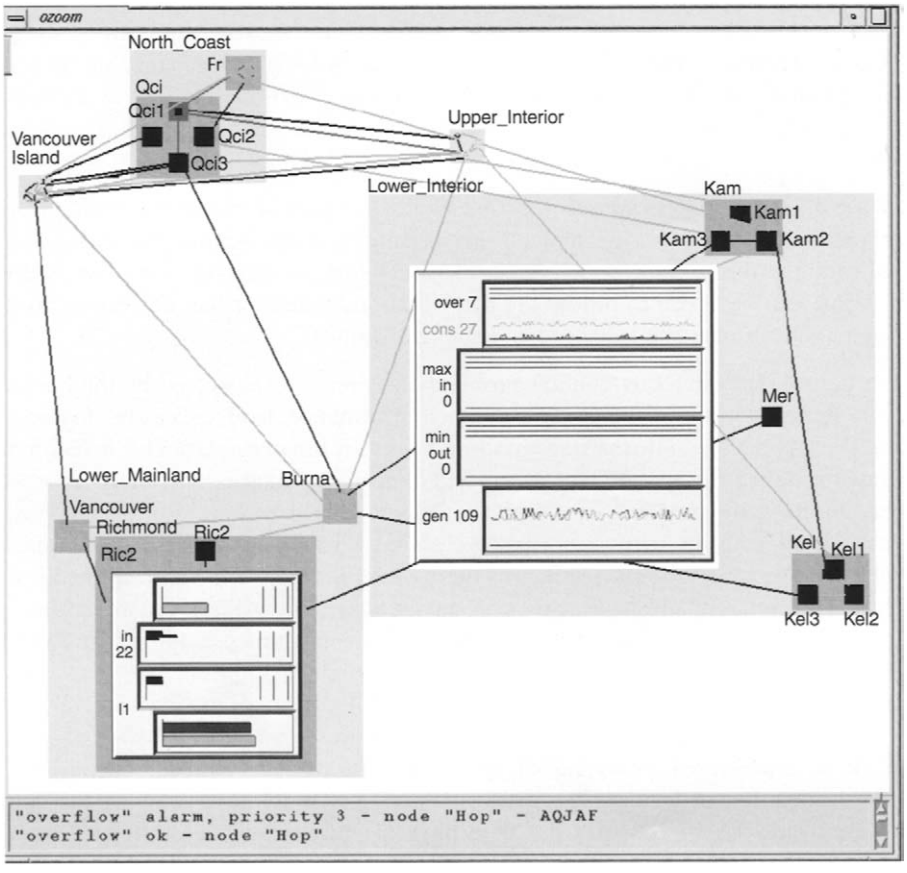

Figure 10.12 A view of the intelligent zoom system developed by Bartram et al. (1998).

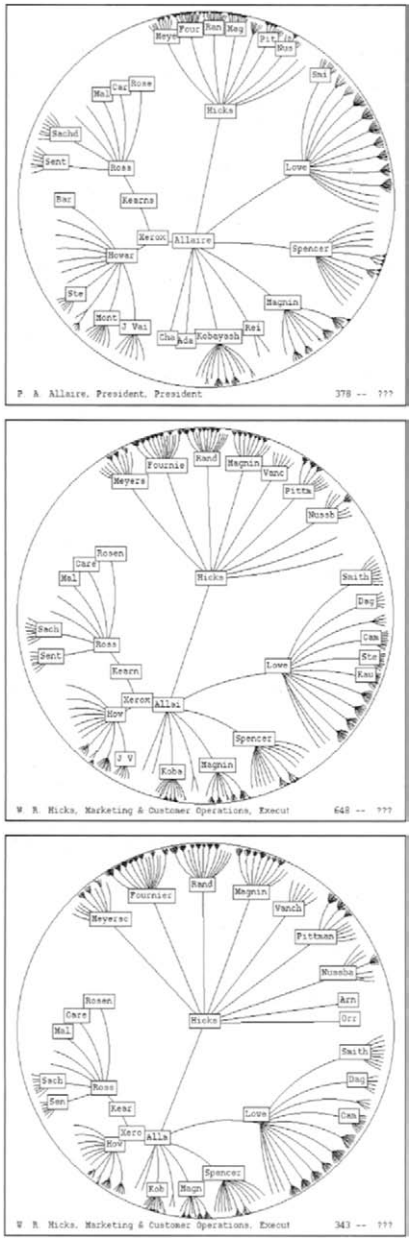

Figure 10.13 Hyperbolic tree browser from Lamping et al. (1995). The focus can be changed by dragging a node from the periphery into the center.

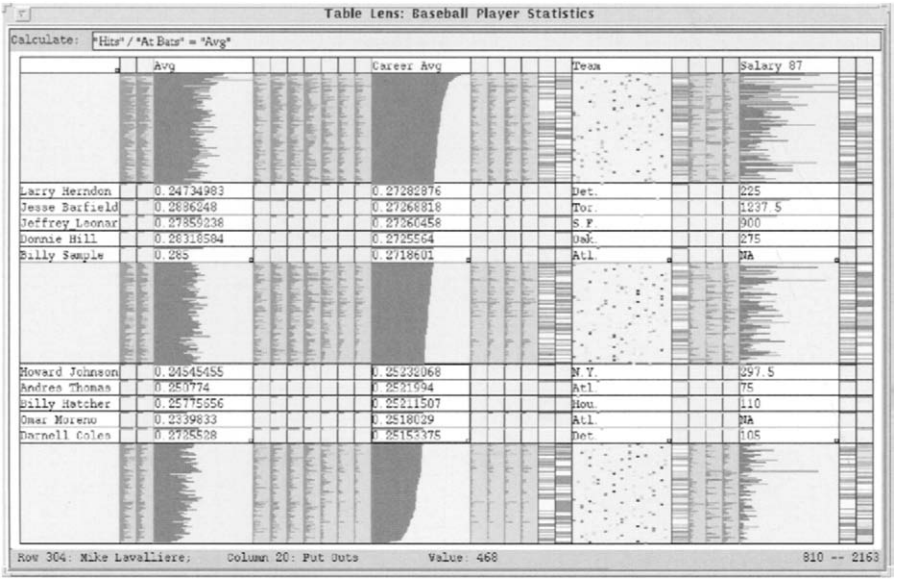

Figure 10.14 Table lens from Rao and Card (1994). Multiple row- and column-wise centers of focus can be created.

of these methods use simple algebraic functions to distort space based on the distance from each focus.

An obvious perceptual issue related to the use of distorting focus–context methods is whether the distortion makes it difficult to identify important parts of the structure. This problem can be especially acute when actual geographical maps are expanded. For example, Figure 10.15 from Sarkar and Brown (1994) shows a distorted view of a map of major cities in North America, together with communications paths between them. The focus is on St. Louis, with the graph expanded at that point, whereas all other regions are reduced in size. The result achieves the goal of making the information about St. Louis and neighboring cities clearer, at the expense of an extreme distortion of the shape of the continent. Compromises are possible; Bartram et al. (1994) do not distort the focal information locally presented in the graph nodes, but they do distort the overall graph layout (see Figure 10.12).

Rapid Zooming Techniques

In rapid zooming techniques, a large information landscape is provided, although only a part of it is visible in the viewing window at any instant. The user is given the ability to zoom rapidly into and out of points of interest, which means that although focus and context are not simultaneously available, the user can move rapidly and smoothly from focus to context and back. If

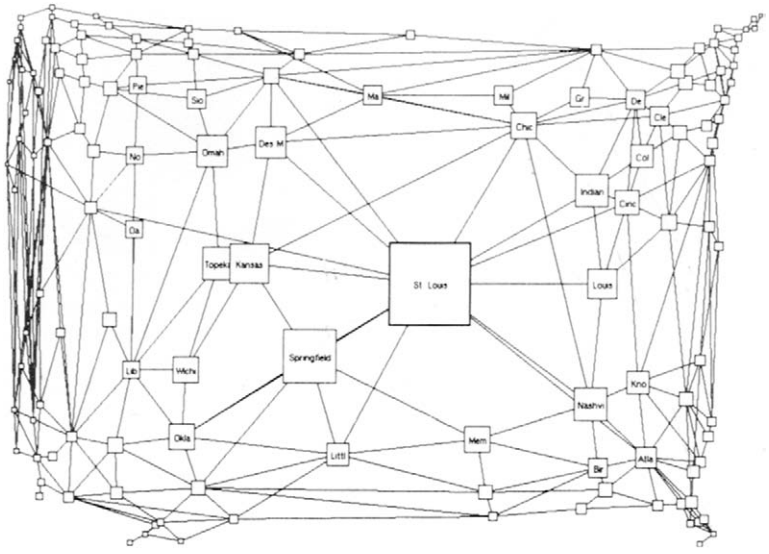

Figure 10.15 Fish-eye view of links between major American cities. The focus is on St. Louis (Sarkar and Brown 1994).

rapid smooth scaling is used, the viewer can perceptually integrate the information over time. The Pad and Pad++ syastems (Bederson and Hollan, 1994) are based on this principle. They provide a large planar data landscape, with an interface using a simple point-and-click technique to move rapidly in and out. Care has been taken to make the animation smooth and continuous.

Mackinlay et al. (1990) invented a rapid-navigation technique for 3D scenes that they called *point of interest navigation.* This method moves the user's viewpoint rapidly, but smoothly, to a point of interest that has been selected on the surface of some object. At the same time, the view direction is smoothly adjusted to be perpendicular to the surface. A variant of this is to base the navigation on an object. Parker et al. (1998) developed a similar technique that is object- rather than surface-based; clicking on an object scales the entire 3D "world" about the center of that object while simultaneously bringing it to the center of the workspace. This is illustrated in Figure 10.16.

In all these systems, the key perceptual issues are the rapidity and ease with which the view can be changed from a focal one to an overview and back. Less than a second of transition time is probably a good rule of thumb, but the animation must be smooth to maintain the identity of objects in their contexts. To maintain a sense of location, landmark features should be designed to be recognized consistently, despite large changes in scale.

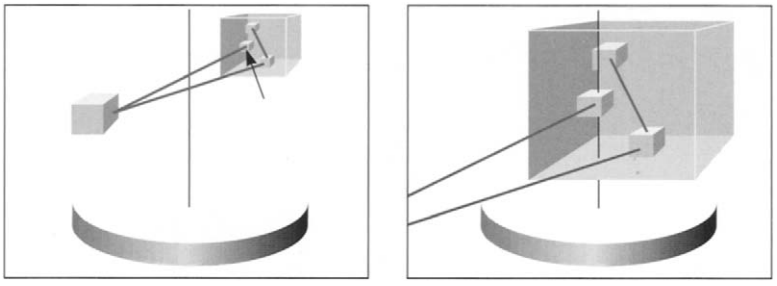

Figure 10.16 In the NV3D systems (Parker et al., 1998), clicking and holding down the mouse causes the environment to be smoothly scaled as the selected point is moved to the center of the 3D workspace.

Elision Techniques

In visual *elision,* parts of a structure are hidden until they are needed. Typically, this is achieved by collapsing a large graphical structure into a single graphical object. This is an essential component of the Bartram et al. (1994) system, illustrated in Figure 10.12, and of the NV3D system (Parker et al., 1998; see Figure 8.25). In these systems, when a node is opened, it expands to reveal its contents.

The elision idea can be applied to text as well as to graphics. In the *generalized fish-eye* technique for viewing text data (Furnas, 1986), less and less detail is shown as the distance from the focus of interest increases. For example, in viewing code, the full text is shown at the focus; farther away, only the subroutine headers are made visible, and the code internal to the subroutine is elided.

Elision in visualization is analogous to the cognitive process of chunking, discussed earlier, whereby small concepts, facts, and procedures are cognitively grouped into larger "chunks." Replacing a cluster of objects that represents a cluster of related concepts with a single object is very like chunking. This similarity may be the reason that visual elision is so effective.

Multiple Windows

It is common, especially in mapping systems, to have one window that shows an overview and several others that show expanded details. The major perceptual problem with the multiple-window technique is that detailed information in one window is disconnected from the overview (context information) shown in another. A solution is to use lines to connect the boundaries of the zoom window to the source image in the larger view. Figure 10.17 illustrates a zooming window interface for an experimental calendar application. Multiple windows show day, month, and year views in separate windows (Card et al., 1994). The different windows are connected by lines that integrate the focus information in one table within the context provided by another. Figure 10.18 shows the same technique used in a 3D zooming user interface. The great advan-

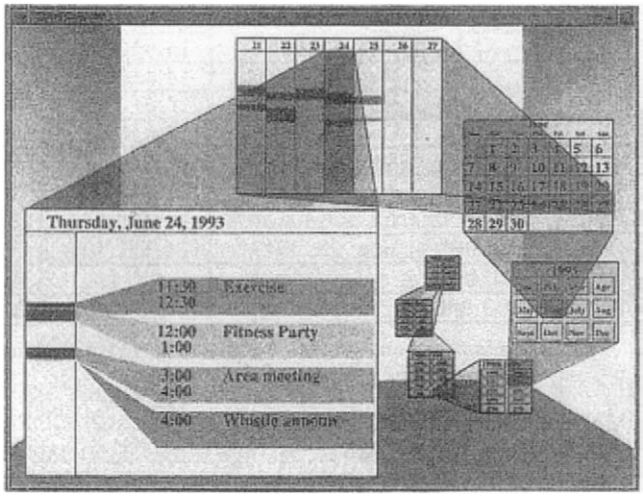

Figure 10.17 The spiral calendar (Card et al., 1994). The problem with multiple-window interfaces is that information becomes visually fragmented. In this application, information in one window is linked to its context within another by a connecting transparent overlay.

tage of the multiple-window technique over the others listed previously is that it is both nondistorting and able to show focus and context simultaneously.

Rapid Interaction with Data

In a data exploration interface, it is important that the mapping between the data and its visual representation be fluid and dynamic. Certain kinds of interactive techniques promote an experience of being in direct contact with the data. Rutkowski (1982) calls it the principle of *transparency;* when transparency is achieved, "the user is able to apply intellect directly to the task; the tool itself seems to disappear." There is nothing physically direct about using a mouse to drag a slider on the screen, but if the temporal feedback is rapid and compatible, the user can obtain the illusion of direct control. A key psychological variable in achieving this sense of control is the responsiveness of the computer system. If, for example, a mouse is used to select an object or to rotate a cloud of data points in 3D space, as a rule of thumb visual feedback should be provided within 1/10 second for people to feel that they are in direct control of the data (Shneiderman, 1987).

Interactive Data Display

Often data is transformed before being displayed. Interactive data mapping is the process of adjusting the function that maps the data variables to the display variables. A nonlinear mapping

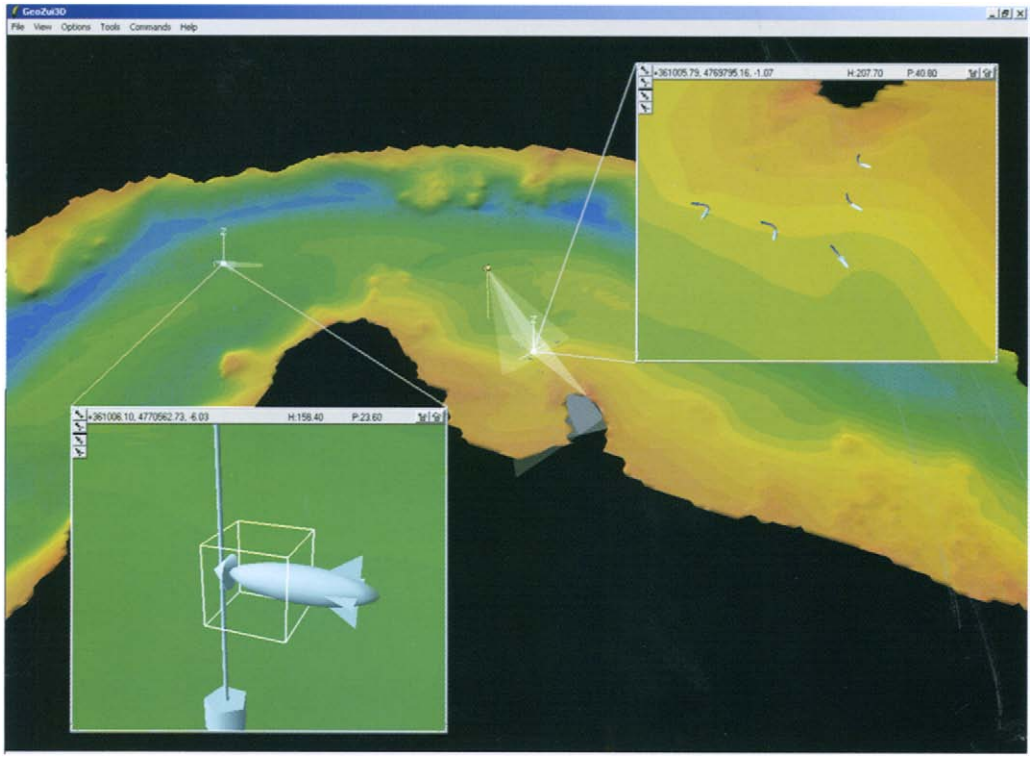

Figure 10.18 The GeoZui3D system allows for subwindows to be linked in a variety of ways (Plumlee and Ware, 2003). In this illustration the focus of one subwindow is linked to an undersea vehicle docking. A second subwindow provides an overview of a group of undersea vehicles. The faint translucent triangles in the overview show the position and direction of the subwindow views.

between the data and its visual representation can bring the data into a range where patterns are most easily made visible. Figure 10.19 illustrates this concept. Often the interaction consists of imposing some transforming function on the data. Logarithmic, square root, and other functions are commonly applied (Chambers et al., 1983). When the display variable is color, techniques such as histogram equalization and interactive color mapping can be chosen (see Chapter 4). For large and complex data sets, it is sometimes useful to limit the range of data values that are visible and mapped to the display variable; this can be done with sliders.

Ahlberg et al. (1992) call this kind of interface *dynamic queries* and have incorporated it into a number of interactive multivariate scatter-plot applications. By adjusting data range sliders, subsets of the data can be isolated and visualized. An example is given in Figure 10.20, showing a dynamic query interface to the Film Finder application (Ahlberg et al., 1992). Dragging the

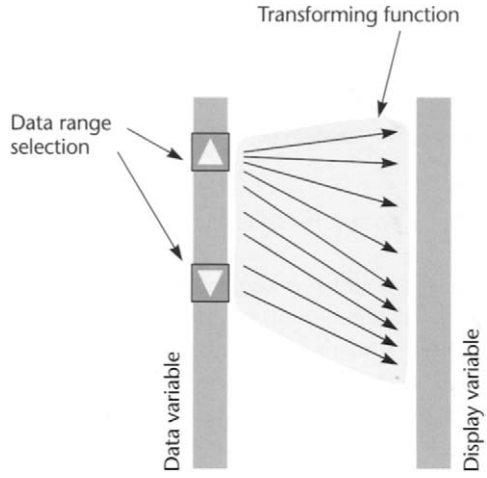

Figure 10.19 In a visualization system, it is often useful to change interactively the function that maps data values to a display variable.

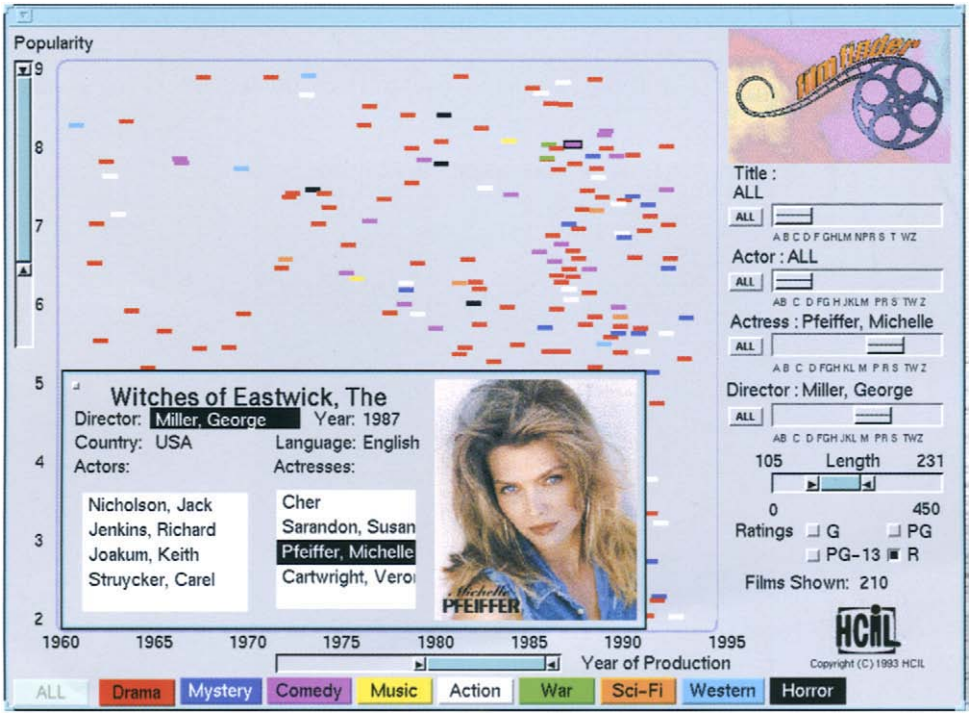

Figure 10.20 The Film Finder application of Ahlberg and Shneiderman (1987) used dynamic query sliders to allow rapid interactive updating of the set of data points mapped from a database to the scatter-plot display in the main window. *Courtesy of Matthew Ward.*

Year of Production slider at the bottom causes the display to update rapidly the set of films shown as points in the main window.

Another interactive technique is called *brushing* (Becker and Cleveland, 1987). This enables subsets of the data elements to be highlighted interactively in a complex representation. Often data objects, or different attributes of them, simultaneously appear in more than one display window, or different attributes can be distributed spatially within a single window. In brushing, a group of elements selected through one visual representation becomes highlighted in all the displays in which it appears. This enables visual linking of components of heterogeneous complex objects. For example, data elements represented in a scatter plot, a sorted list, and a 3D map can all be visually linked when simultaneously highlighted.

Brushing works particularly well with a graphical display technique called *parallel coordinates* (Inselberg and Dimsdale, 1990). Figure 10.21 shows an example in which a set of automobile statistics are displayed: miles per gallon, number of cylinders, horsepower, weight, and so on. A vertical line (parallel coordinate axis) is used for each of these variables. Each automobile is represented by a vertical height on each of the parallel coordinates, and the entire automobile is represented by a compound line running across the graph, connecting all its points. But because the pattern of lines is so dense, it is impossible to trace any individual line visually and

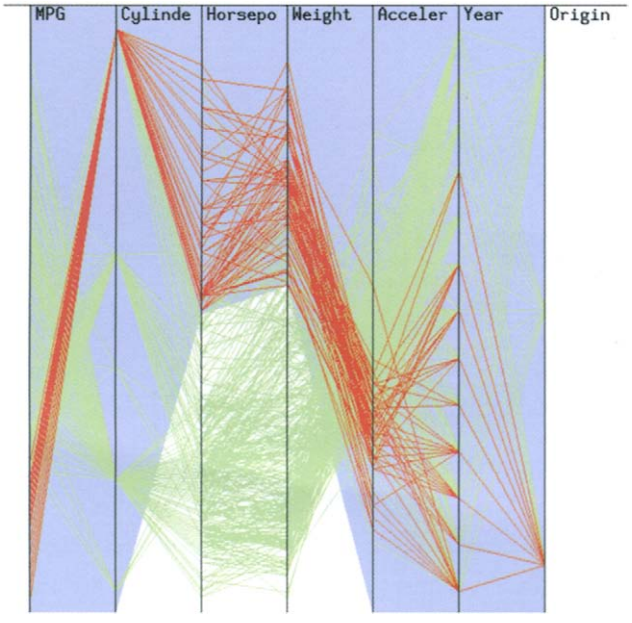

Figure 10.21 In a parallel-coordinates plot, each data dimension is represented by a vertical line. This example illustrates brushing. The user can interactively select a set of objects by dragging the cursor across them. *From: XmdvTool (http://davis.wpi.edu/~xmdv). Courtesy of Matthew Ward (1990).*

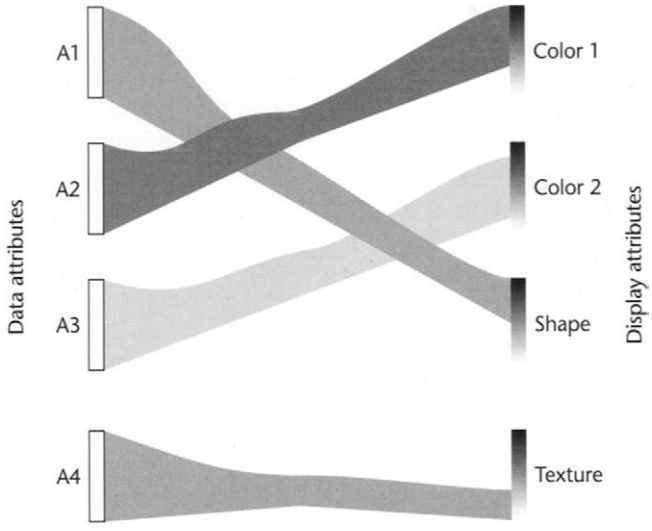

Figure 10.22 In some interactive visualization systems, it is possible to change the mapping between data attributes and the visual representation.

thereby understand the characteristics of a particular automobile. With brushing, a user can select a single point on one of the variables, which has the result of highlighting the line connecting all the values for that automobile. This produces a kind of visual profile. Alternatively, it is possible to select a range on one of the variables, as illustrated in Figure 10.21, and all the lines associated with that range become highlighted. Once this is done, it is easy to understand the characteristics of a set of automobiles (those with low mileage, in this case) across all the variables.

As discussed in Chapter 5, it is possible to map different data attributes to a wide variety of visual variables: position, color, texture, motion, and so on. Each different mapping makes some relationships more distinct and others less distinct. Therefore, allowing a knowledgeable user to change the mapping interactively can be an advantage. (See Figure 10.22.) Of course, such mapping changes are in direct conflict with the important principle of consistency in user interface design. In most cases, only the sophisticated visualization designer should change display mappings.

Conclusion

This chapter has been about the how to make the graphic interface as fluid and transparent as possible. Doing so involves supporting eye–hand coordination, using well-chosen interaction metaphors, and providing rapid and consistent feedback. Of course, transparency comes from

practice. A violin has an extraordinarily difficult user interface, and to reach virtuosity may take thousands of hours, but once virtuosity is achieved, the instrument will have become a transparent medium of expression. This highlights a thorny problem in the development of novel interfaces. It is very easy for the designer to become focused on the problem of making an interface that can be used rapidly by the novice, but it is much more difficult to research designs for the expert. It is almost impossible to carry out experiments on expert use of radical new interfaces for the simple reason that no one will ever spend enough time on a research prototype to become truly skilled.

Having said that, efforts to refine the user interface are extremely important. One of the goals of cognitive systems design is to tighten the loop between human and computer, making it easier for the human to obtain important information from the computer via the display. Simply shortening the amount of time it takes to select some piece of information may seem like a small thing, but information in human visual and verbal working memories is very limited; even a few seconds of delay or an increase in the cognitive load, due to the difficulty of the interface, can drastically reduce the rate of information uptake by the user. When a user must stop thinking about the task at hand and switch attention to the computer interface itself, the effect can be devastating to the thought process. The result can be the loss of all or most of the cognitive context that has been set up to solve the real task. After such an interruption, the train of thought must be reconstructed, and research on the effect of interruptions tells us that this can drastically reduce cognitive productivity (Field and Spence, 1994; Cutrell et al., 2000).