TWO-HANDED 3D INTERACTION TECHNIQUES FOR HETEROGENEOUS INTERACTION DEVICES

This chapter presents the development of interaction devices and techniques in the current project. The previous chapter has shown the potential of 3D interaction and two-handed operation for computer supported conceptual modeling. Here, the bimanual interaction techniques are described that have been implemented in a conceptual modeling application called ID8Model. The program can be used with interaction devices of different nature, thus allowing for evaluations of the appropriateness of devices for specific interaction techniques.

The interaction issues addressed in this chapter regard concepts developed for manipulating models. The presentation starts with the integration of 3D interaction devices with the display of a CAD system. Then, selection of objects is addressed and to this purpose, the 3D cursor is introduced as the 3D counterpart of the familiar 2D cursor. Subsequently, the clutch mechanism is described. It allows the user to work in a comfortable posture while operating a system with 3D interaction. Examples of design oriented tools in the ID8Model application illustrate how the interaction techniques developed can be used for design oriented tasks with a number of different interaction devices.

At the end of the chapter, the development of the Frog interaction device is presented. With the device, the 3D interaction techniques in ID8Model have been prototyped. In the Frog design, several ergonomical issues have been addressed, resulting in a 6-DoF device that can be held between the fingers for precise manipulation tasks.

3.1 Introduction

When developing an interface for a desktop computer application an interaction designer is supported by numerous design guidelines (Apple Computer Inc., 1993; Redmond-Pyle & Moore, 1995) and ample tools are available to guide the designer in designing a computer interface. That is, as long as the designer is looking to

hoofdstuk 3 25-08-1999 13:54 Pagina 86

develop an application using with a standard desktop user-interface, of the WIMP (Windows, Icons, Menus, Pointing Device) type. These interfaces are popular and useful for a range of applications but the underlying assumption is that they will be employed on a standard desktop-computer set-up (with a mouse and keyboard).

In the previous chapter, it was established that the familiar WIMP interfaces and 2-DoF interaction devices like the mouse create a barrier between the designer and the design in computer modeling. To lower the barrier, another form of interaction was proposed where the designer can manipulate objects directly in 3D. The term spatial input has been proposed to distinguish interfaces that employ this form of free-space interaction from a variety of interfaces labeled “3D interfaces” (Hinckley, Pausch, Goble & Kassell, 1994a). We will adopt this term but in the light of the discussion of the previous chapter, will refer to spatial interaction instead of spatial input to stress the importance of the task a user needs to accomplish. Additionally, devices suited for spatial interaction will be referred to as spatial interaction devices. The isotonic 6-DoF devices, introduced in the previous chapter, are spatial interaction devices because they allow a user to interact with an application directly in 3D.
When spatial interaction devices are applied, WIMP interfaces can no longer be used and novel interaction techniques need to be developed. In addition, support for concurrent bimanual use of interaction devices is not found in interaction guidelines or in interface toolkits either. In this chapter, these issues will be discussed and spatial interaction techniques are presented. The interaction techniques described in this chapter have been developed to explore the potential of bimanual 3D interaction.

Although the focus in this work is on the interaction techniques, the display method of the system needs to be considered. When interaction takes place on a computer system with a single static viewpoint, the effects of movement parallax and stereo vision are not available to the user. This deprives the user of two important factors in the perception of depth. However, there are depth cues when the viewpoint is static. There is occlusion, the effect that objects close to the viewer obscure objects further away. The lighting conditions in a scene can also help to understand where objects are located. Depending on their position and orientation,

relative to light sources, objects will be shaded differently. In addition, the effect of perspective foreshortening can help to identify the depth of objects because close objects will be displayed larger than distant objects.

Next, a summary of the main features of ID8Model is given. These features were formulated as demands for the application at the start of the project. ID8Model was not intended as a fully functional conceptual modeling application but rather as a test-bed for interaction devices, as a tool to prototype interaction techniques and as a tool for experimentation. Although the functionality of ID8Model was not supposed to be complete, the tools implemented should reflect the way designers work with traditional tools as much as possible. Every tool should be operable with either one or two hands to allow comparing of one- and two-handed operation. In addition, the tools should be operable with every pointing device present on the system so those devices can be compared. Regarding the functionality of the tools, it was decided to create a number of tools, at least one in each of the categories listed below.

Functionality:

Shape creation Initially, a tool should be created to enable the designer to create primitive shapes (block, sphere, cylinder, cone, torus, etc.). Later, interfaces could be created for more elaborate shape creation procedures.

Assembly The designer should be able to arrange parts of an object to evaluate different configurations. Assembly with ID8Model should resemble assembly in the real world, using two hands in 3D (when two spatial interaction devices are available).

Shape editing Editing the surface of an object allows a designer to modify existing shapes. Local and global editing should both be supported. Moving individual vertices belonging to the surface of an object is considered a form of local editing. Global editing operations such as bending and twisting affect the shape of an object globally.

Input:

Heterogeneous interaction devices ID8Model should be able to support every pointing device, from spatial interaction devices to more universal 2D devices such as the mouse and the graphics tablet. The application should take advantage of all the capabilities of a device. For example, when a graphics tablet is connected,

hoofdstuk 3 25-08-1999 13:54 Pagina 88

	the application should process the pressure and tilt values of the stylus where appropriate. Therefore, the interaction techniques need to be designed for use with different kinds of interaction devices.
Simultaneous use of multiple interaction devices	All the devices present on the system should be useable simultaneously. Often, several devices can be used to operate an application but only one if them can be used at a time.
Bimanual interaction	Support of the simultaneous use of devices should be designed such that bimanual operation is facilitated. In addition, the interaction techniques should be designed for bimanual as well as single-handed use. First, because this allows the comparison of one- and two-handed operation and second, because in an actual CAD tool, interaction techniques should allow a designer to choose for single-handed operation of the system without significant repercussions.
Space and time multiplexing	Both the dedication of a device to a specific task in the application (space multiplexing) and the use of the same device for different tasks at different points in time (time multiplexing) should be supported. These are two ways of assigning interaction devices to functionality in an application. By supporting both approaches, the advantages and disadvantages can be established.
Output:	
Display on a 2D screen	Development of the ID8Model system takes place on a system with a standard display (no support for movement parallax and stereo vision). The other depth cues, mentioned above, should all be implemented
Enhanced display	Although development takes place on a system with a 2D screen, it is anticipated that 3D display techniques will be used later. Therefore, the application should be prepared for 3D display techniques.

3.2 Display and manipulation spaces

Working with a CAD system, the designer manipulates the computer model with interaction devices and looks at the computer screen to observe the results of his actions. The space in which the devices are moved will be denominated manipulation space. Feedback of the results of the designer's actions is presented on the computer screen in what will be called the display space. The model under manipulation is represented in the computer in the modeling space.

The modeling space is 3D because it represents 3D models (using a coordinate system known as the world coordinate system). In most CAD systems, both the manipulation and the display spaces are 2D. The mouse for instance is moved only parallel to the surface of the desk and 2D display is presented on a computer screen. With spatial interaction however, the manipulation space is 3D and in systems with 3D display capabilities, the display space is 3D. In this section, it is presented how the 3D manipulation space should relate to the display space. In other words: where should spatial interaction with a CAD tool take place? This question shall be addressed in two steps because two conversions are involved. The conversion of movements in manipulation space to modifications to the model in modeling space and the conversion of those modifications to the display space. For example, when objects are moved, movements of interaction devices are transformed to movements of objects in the modeling space. The moving objects are then transformed to display space.

The discussion starts with a summary of the display generation process in computer graphics systems. This process transforms the modeling space into the display space. The emphasis will be on the method used to project the 3D scene on the 2D screen. This leads to the description of the display space of a projection method that provides perspective foreshortening.
Then, the discussion continues with the manipulation space in which spatial interaction devices are used. Without precautions, the same activities with different devices may lead to different results in modeling space and, subsequently, in display space. Therefore, a general approach is needed so that the devices can be used the same way. The introduction of a single coordinate system, to which all the devices report their positions and orientations, facilitates the uniform approach. Integration of manipulation and display spaces is accomplished with the single coordinate system.

3.2.1 The display space

In CAD tools, the purpose of the display generation process is to inform the user about the current shape of the model and the state of the system. This permits the user to review the results of modeling activities accurately. An obstacle is that, in most systems, the interaction with a 3D scene has to be reviewed on a 2D projection of the scene. Above, it was mentioned that perspective

hoofdstuk 3 25-08-1999 13:54 Pagina 90

foreshortening helps to establish depth of objects on a 2D screen. Perspective foreshortening is accomplished with the aid of a perspective projection that projects computerized objects from the 3D to the 2D screen. The use of perspective projection has implications for the display space and this will be illustrated next.

Perspective projection is not the only means to project object data from the 3D to the 2D domain. Perspective projection is one of a class of projection methods, formally known as planar geometric projections but referred to as projections from this point on. In most CAD applications, two kinds of projections can be found: parallel and perspective projections. Perspective projection is the most realistic view because it creates perspective foreshortening. Objects are rendered larger when they are closer to the viewer and vice versa, so that an observer can establish their relative depth. Applying the same texture to objects facilitates the establishment of the dimensions of objects. Figure 3-1 illustrates the effects of perspective foreshortening on three textured boxes of identical dimensions.

Figure 3-1. Perspective projection of three boxes of equal size with different distances to the viewer.

Figure 3-2. Parallel projection of three boxes of equal size with different distances to the viewer.

Parallel projection does not create perspective foreshortening and therefore, it is difficult for users to estimate the distance of objects. This is illustrated in Figure 3-2 where the same three boxes of Figure 3-1 have been projected with parallel projection. The figure shows that it is impossible to retrieve the depth of the boxes because they all have the same size. This can be desirable in situations where it is important that the dimensions of objects can be obtained independent of their depth. However, when a user manipulates objects, it is important that the depth of the objects is immediately clear and therefore perspective projection is advised.

What follows is a brief summary of the display generation process. Only those elements are presented that are needed to discuss the integration of display and manipulation spaces.

The display generation process is in computer graphics literature often referred to as the 3D viewing process, although it has nothing to do with viewing by an observer. Rather, it refers to the process of generating an image on the screen from a data model representing

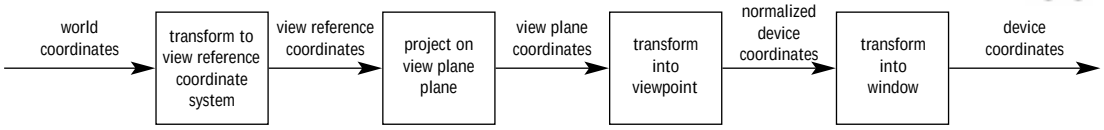


Figure 3-3. Model of the viewing process.

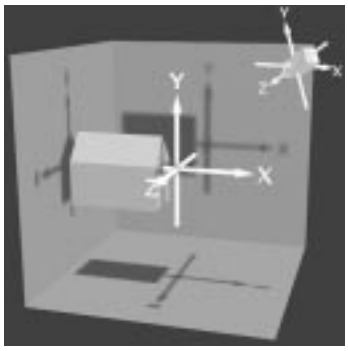


Figure 3-4. The world and view reference coordinate systems.

Figure 3-5. The view plane used for perspective transformation.

3D objects. It has been described extensively in computer graphic textbooks. The viewing process is applied on the scene, which is the constitution of the model and the interface elements. All entities in the scene are defined in the world coordinate system and are rendered on the display in four steps, as illustrated in Figure 3-3[1].

In the first step of the viewing process, objects in world coordinates are transformed with the view reference transformation. The transformation is illustrated in Figure 3-4. The house is defined in the world coordinate system, shown centered in the scene. The origin of the view reference coordinate system, or view reference point (VRP), coincides with the camera in the upper right corner of the figure. The view reference transformation transforms the coordinates of the house in the world coordinates to view reference coordinates. Conceptually, one could think of the VRP as the position of the camera through which the observer is looking at the scene.[2]

The central element in the perspective projection is the view plane, as illustrated in Figure 3-5. The view plane is specified in world coordinates by the view reference point (VRP) and the view direction. The view direction is the direction from the viewpoint (COP) to the view reference point. The view plane is positioned in the view reference point and the normal of the viewing plane (VPN) coincides with the viewing direction. The scene is projected into a window on the view plane with its associated view plane coordinate system (the u and v axes in the figure). The view up vector (VUP) determines the direction of the v axis of the view plane and forms a left-handed coordinate system together with the u axis and the view plane normal. The view plane coordinate system is used to define a window inside the view plane, specified by minimum and maximum u and v values. The scene is projected in the window, with the aid of

[1] Unfortunately, not all textbooks present the same terminology. In the following discussions on display and manipulation spaces, the terminology and the models of the well-accepted standard work by Foley and van Dam (1982) is used.

[2] Note that the world coordinate system forms a right-handed system and the view reference coordinate system is left-handed. A left-handed system is often used and has the consequence that when the x- and the y-axis are aligned with the screen, the z-axis points into the screen. Thus giving the natural interpretation that increasing values for z indicate points further away from the viewer.

hoofdstuk 3 25-08-1999 13:54 Pagina 92

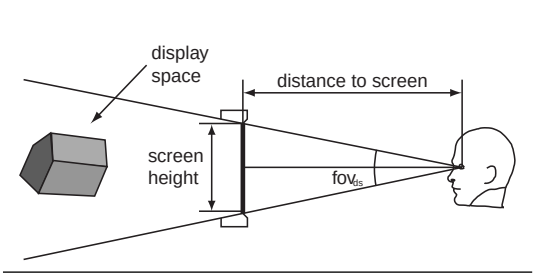


Figure 3-6. Display space of an observer.

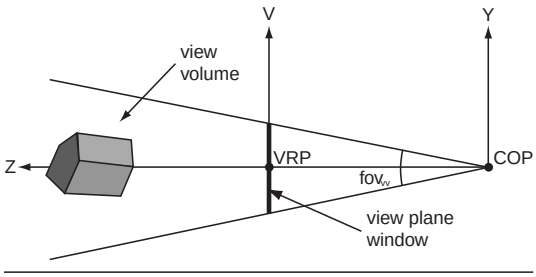


Figure 3-7. View volume of the perspective projection.

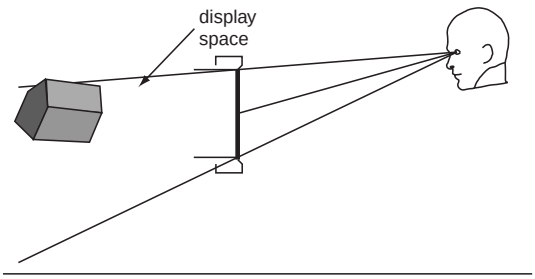
the center of projection (COP). Next, it will be shown how the parameters of the perspective projection can be chosen such that an observer in front of the monitor perceives an image with the correct perspective. For simplicity, it is assumed that the observer looks with one eye that is located on a line through the center and perpendicular to the screen as shown in Figure 3-6. In addition, it is assumed that the height of the displayed image is equal to the height of the screen. With these assumptions, the display space is a pyramid with opening angle fov_{ds} (the field of view of the display space).

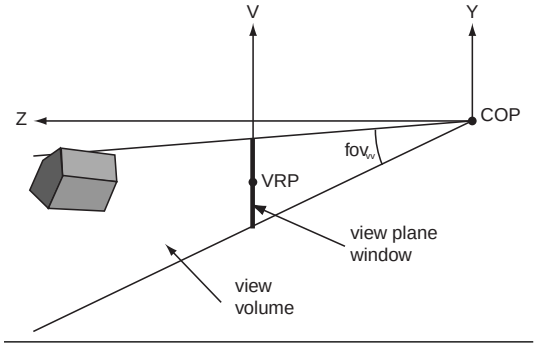
Figure 3-7 shows how a perspective projection can be set up for the display space in Figure 3-6. The view volume is the part of the scene that is visible to the observer. In Figure 3-7, the view volume is a semi-infinite pyramid with opening angle fov_{VV} (the field of view of the view volume). The opening angles of the display space and the view volume should be equal to present the observer with a perspective corresponding to the current position. Pasman (1997) has studied the effects of inaccuracies on perspective matching.

Figure 3-8. Display space of an observer not located in front of the center of the screen.

The perspective projection just presented is known as an on-axis projection because the projection of the COP on the view plane coincides with the VRP (the COP is on the normal of the view plane).

Figure 3-9. View volume of an off-axis perspective projection.





In off-axis projection, this is not the case. It is relevant in systems that apply movement parallax as a depth cue. In these systems, the position of the observer's head is measured to generate images with the correct perspective for the current viewing position. The observer's head position is coupled to the display, hence the term head-coupled display. More specifically, off-axis projection is used in systems that use the Fish Tank VR projection method (Ware, Arthur & Booth, 1993), such as the CAVE (Cruz-Neira, Sandin & DeFanti, 1993) and Cubby (Djajadiningrat, 1998). Figure 3-8 shows the display space for a user not located in front of the center of the screen. The corresponding view volume is shown in Figure 3-9.

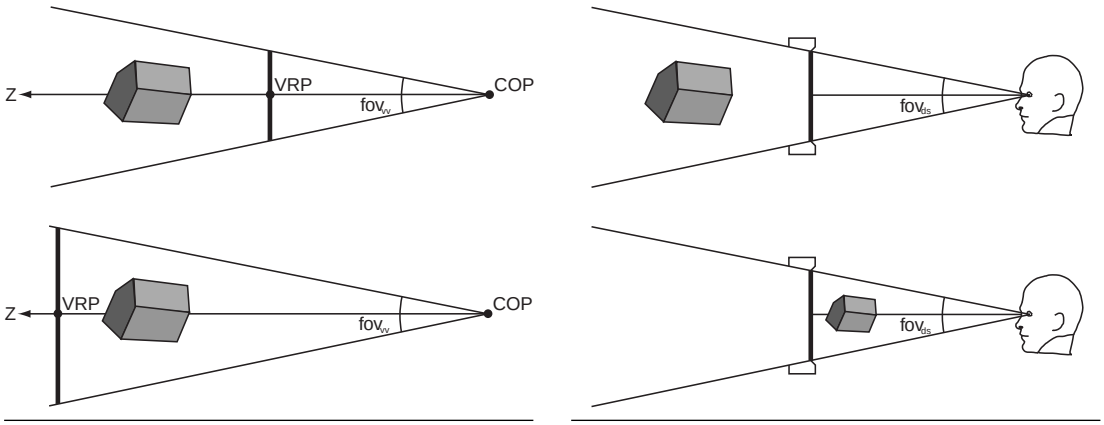


Figure 3-10. Two locations and dimensions of the view plane in the same scene on the left and their representation in display space on the right.

In systems with head-coupled or stereo display, the location of the view plane relative to the objects does determine if the scene appears to be located in front of, or behind the screen. Thereby creating the illusion of a true 3D display space. This is illustrated in Figure 3-10 that presents two situations with view planes of different dimensions and different locations of the view reference point. In the display space, the scenes are different because of their distance to the observer. This is not the case in systems without 3D display capabilities. Then, the observer can not tell these situations apart because the images on the screen are equal and the display space is 2D.

The result of the perspective projection is a 2D image, specified in the coordinate system of the view plane. The last two steps in the viewing process are a series of 2D transformations (scaling, translation and/or rotation) applied to the image. The transformations determine how and where the image will be displayed on the screen.

hoofdstuk 3 25-08-1999 13:54 Pagina 94

3.2.2 Integrating display and manipulation spaces

Above, it was mentioned that a uniform approach is needed to use spatial interaction devices for interaction with CAD applications. The reason is that most interaction devices report their position and/or orientation relative to their individual coordinate systems. If no action is undertaken, manipulation with the devices will lead to different results in display space. However, a user expects similar responses when different interaction devices are used. To this purpose, a single coordinate system is defined, to which positions and/or orientations of all the devices can be transformed. This will ease the implementation of a uniform response.
Recently, some attempts have already been made by commercial developers to offer a uniform approach in API's (Application Programmer's Interfaces). Game oriented API's, such as DirectX by Microsoft and Apple's Game Sprockets, offer the developer support for different interaction devices. However, these API's are intended for game-related interaction devices such as joysticks. Since spatial interaction is not often used in games, there is virtually no support for this kind of interaction. On the other hand, the WorldToolKit API by Sense8 was created for real-time visual simulation and is often used for both immersive and desktop VR systems. It does offer extensive support for spatial input but it too, lacks a common coordinate system. The QuickDraw 3D API by Apple supports spatial input without a common coordinate system too. The API has been used in this project because it allows connecting multiple devices to different functions in the software (Djajadiningrat & Gribnau, 1998; Gribnau & Djajadiningrat, 1998).
The GKS-3D standard (ISO 8805, 1988) prescribes that interaction devices should report positions relative to the device coordinate system associated with the screen. Although the choice the coordinate system is arbitrary, the GKS choice seems convenient since it links to the display space with which we seek integration as argued below.

If the spatial interaction devices report their positions and orientations in a common coordinate system, the next step is to determine how to convert these positions and orientations from the manipulation to the display space. This is accomplished by transforming the positions and orientations into the world coordinate system that is used internally by the system to represent all the items in the scene. Therefore, it should be decided how to transform

the position and orientation of the interaction devices from their common device coordinate system into the world coordinate system. The implications this transformation has for the user shall be clarified with the concepts of unified and non-unified systems.

Figure 3-11, shows an example of a situation where the display and the manipulation have been superimposed. This situation has been referred to as a unified system (Djajadiningrat, 1998), as opposed to non-unified systems with separate display and manipulation spaces. The 2D equivalent of a unified system would be a computer equipped with a light pen, where the user manipulates with the light pen directly on the display. A unified situation can be achieved technically, by transforming the coordinates of the interaction device with the inverse of the sequence of transforms from the 3D viewing process (excluding the perspective projection). Figure 3-11 demonstrates that it can appear to a user as if the interaction device is located in the display space. In a system with a single static viewpoint, the display space is not 3D but 2D. Unified in this context means that the projections of the positions of the interaction device in the real and the virtual world on the screen match. This situation is obviously disturbed as soon as the position of the head of the observer is not in accordance with the perspective generated.

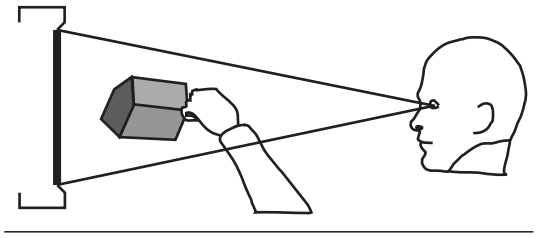


Figure 3-11. A unified display and manipulation space

Djajadiningrat argues that unified systems can only be realized with head-coupled display. He lists a number of advantages and disadvantages of unified systems that will be repeated here, but modified and extended to fit the context of systems with a static viewpoint. The advantages of unified systems are listed first.

- The visual presence of the hand in the display space can be beneficial in that it allows the user to observe his manipulating hand.
- The one-to-one relation between the display and manipulation space makes it easy for the user to predict how to move the interaction device in the real world to arrive at a position in the virtual world.

Following are disadvantages of unified systems.

- The visual presence of the hand can be a disadvantage when it obscures the part of the scene that is of interest to the user. During manipulation tasks, the location of the controlled object is usually the area of the scene that the user is interested in.

hoofdstuk 3 25-08-1999 13:54 Pagina 96

- When the hand is visually present in the display space, lags of the system are more noticeable because both the hand and the controlled object are visible.
- The fatigue problem. The manipulation space is limited to the display space that is located in front of the monitor. Manipulation is only possible by bringing one's hands inside the manipulation space. Thus, the user is forced to keep his hands in the air while working with the system.
- The manipulation space is small because it is limited to the dimensions of the display space. It is therefore much smaller than the physical limits of the set-up, such as the position of the monitor and the table surface. It is usually smaller than the space that is reachable to the user also.
- The small manipulation space has another disadvantage. When a monitor with a CRT is used for display, the manipulation space is close to the monitor. This presents a problem working with electromagnetic trackers because they are susceptible to the electromagnetic field radiated by the monitor. This results in a jittery appearance of the controlled objects that is annoying when high precision is required. There are some remedies to minimize the influence of the monitor to a small area in front of the screen but stray fields are an inherent problem with these trackers (Nixon, McCallum, Fright & Price, 1998).
- With a fixed manipulation space, the precision of the manipulation is fixed also. In some cases, it might be advantageous to a user to work with large movements on a small part of the scene and thus achieving high precision. In other cases, the user might want to work with low precision and cover a large area of the scene with small gestures. In a unified system with fixed precision, this is impossible. In other words, a unified system has a fixed Control/Display ratio (C/D ratio). The term C/D ratio (or its the reciprocal value, the D/C gain) descends from 2D user interfaces and describes the ratio of interaction device displacement and the displacement of the controlled object on the screen.

Evaluating the benefits and disadvantages of the unified approach, unification does not seem workable for a system with a single static viewpoint. Then, how should the manipulation space be linked to the display space? We shall address this question by considering the elements of the transformation matrix between manipulation and display space separately.

First, there is evidence that a rotation between the manipulation and the display space is not desirable. Tendick, Jennings, Tharp and

Starck (1993) show that when the angle between display and manipulation space exceeds 45º, performance in an endoscopy task deteriorates. This has been confirmed by our own experimentation with a number of rotation angles. Combined with the fact that there is no apparent benefit or need for a rotation when a user is working with a desktop system it was decided to disregard rotation.

Second, there is probably not one optimal offset between display and manipulation spaces suited for all tasks and all users all of the time. For example, it might be beneficial to have different offsets for reasons of comfort or to avoid occlusion of parts of the display space by the hands of the user. Instead of searching for the optimal manipulation space, interactive control is proposed. The user determines where the manipulation space is located. Section 3.4 presents clutch mechanisms that give the user interactive control over the offset between manipulation and display space. It has been shown by Groen and Werkhoven (1998) that a displacement of display and manipulation space (of 10 cm) was not of significant influence to manipulation performance (speed/accuracy). The experiment was conducted in an immersive VR environment but the results suggest that the display and manipulation spaces can be separated in a desktop VR environment too, without sacrificing performance.

Third, concerning scaling, there is probably also not one optimal scaling factor between the display and the manipulation space[3]. Again, interactive control over the C/D ratio presents the possibility to set the optimal C/D ratio for the current task. We found that changing the C/D ratio is not needed often and is a far less important issue than control over the offset between display and manipulation space. The next question is, whether the C/D ratio should be equal for all directions. In 2D user interfaces, the C/D ratio is one uniform scaling factor for both the horizontal and the vertical dimensions. In our experience, there is no reason to change this in 3D user interfaces, especially since modeling tasks typically take place in a confined area. We found that one uniform C/D ratio for all three axes is enough to control the precision required for the current task. In immersive VR however, there is sometimes need for a non-uniform C/D ratio when there is need for seamless direct manipulation of both nearby and distant objects. To this purpose, the Go-Go interaction technique was developed by Poupyrev and

[3] Only positive scaling factors are considered. Negative scaling values result in mirroring the manipulation space which seems not very functional for intuitive manipulation.

hoofdstuk 3 25-08-1999 13:54 Pagina 98

Billinghurst (1996) which uses the metaphor of interactively growing the user's arm when it is extended beyond a predetermined range.

Finally, shearing the manipulation space will not be considered because it does not seem to serve a purpose for facile manipulation[4]. Therefore, in ID8Model, interactive control over both the offset and the scaling of the manipulation space is implemented.

To conclude the discussion on the integration of display and manipulation space, it is considered whether the manipulation spaces of both hands in bimanual operation should be identical. Above, it was stated that the same integration of display and manipulation space should be used for all devices. However, using two devices with two hands at the same time is different from using them sequentially with one hand. There seems to be no need for different C/D ratios or shearing factors for left- and the right-handed devices. However, an offset between the manipulation spaces might be necessary. This is motivated by the fact that interaction devices can collide when the user wants to use them close to each other.

3.3 Interacting with 3D objects

As shown in the previous chapter, selection, or target acquisition, is one of the basic interaction tasks in a user interface. It is needed whenever a user needs to manipulate different virtual objects at different points in time. Selection will indicate the system which object should be manipulated. This is true for both 2D and for spatial user interfaces. In a WIMP application for instance, an icon can be moved only after it has been selected by bringing the cursor over it and pressing the mouse button. To select an item from a 3D scene, most WIMP applications use a similar approach. The user moves the cursor over the projection of the item and presses the mouse button. Usually, the front object is selected. That makes sense because items further away from the viewer are occluded at the selection point. When spatial interaction devices are used, there is need to reconsider the selection mechanism because they are used in a 3D manipulation space. However, the selection technique implemented in ID8Model shows that the selection technique if

[4] The inverse of a perspective projection results in shearing the manipulation space also.

WIMP applications can be combined with spatial interaction elegantly.
The discussion of 3D selection techniques starts with the introduction of the 3D cursor, the central element in a spatial interface and the representation of the interaction device in the scene. Then, two established selection techniques for spatial input are presented and evaluated to find that there is need for another selection technique. Cursor-based ray casting is presented as a selection technique that combines the advantages of the other two selection techniques. Finally, we investigate selection for two-handed operation.

3.3.1 The 3D cursor

As stated above, the 3D cursor is the representation of the interaction device in the modeling space. The question could be raised whether the cursor could represent the hand of the user like in many immersive VR systems. In VR, the user interacts with the virtual world with a glove as interaction device. The representation of the glove in the virtual world is often a simplified representation of the hand of the user. With handheld interaction devices, the 3D cursor as representation of the interaction device is a more suitable solution, since the user might hold the device in many different grips.

Absolute and relative devices

Akin to WIMP interfaces is that a cursor is a means to use relative as well as absolute interaction devices. In the previous chapter, relative devices were distinguished from absolute devices by the way the coordinates are sensed. With respect to position measurement, absolute devices sense positions and relative devices sense the displacement a user applies to them. The mouse is a well-known example of a relative device and in WIMP interfaces, a cursor is used to employ the mouse for specifying positions on the screen to applications. Instead of directly specifying the location on the screen, the user moves the mouse and changes the location of the cursor that specifies the location. This introduces an extra step in the process of specifying locations on the screen, not needed with absolute devices such as a drawing tablet. With absolute interaction devices, the cursor “only” serves to help the user relate the manipulation and the display spaces.

A 3D cursor helps to realize the same functionality in 3D user interfaces. Relative and absolute interaction devices both control a

hoofdstuk 3 25-08-1999 13:54 Pagina 100

3D cursor that is used to specify a position and orientation in the display space. With respect to the location of the 3D cursor, relative devices will displace the cursor when activated while absolute devices specify the location of the cursor directly.

The shape of the cursor

The next objective of the 3D cursor is to convey the state of the interaction device in the display space to the user. The shape of the cursor should therefore be optimal for conveying this information. The question is what parameters determine the optimal shape of the cursor for that purpose.

In many WIMP interfaces, the shape of the cursor is used for more than representing the position of the interaction device alone. As shown in the previous chapter, the same interaction device is frequently used for several tasks at different points in time. This is realized by introducing modes into the application. Often, WIMP interfaces present feedback about the current mode of the application with the shape of the cursor. For instance, when the user starts a drawing task, the cursor shape changes to a pencil shape or when text editing is activated, the cursor changes to a beam shape (Figure 3-14). That same functionality is useful for a 3D cursor in a modeling application and the shape considerations covered next hold for every cursor shape, regardless of the mode.

The first parameter considered is the size of the cursor and correlates directly with the notability of the cursor. On one hand, the cursor should be large enough to be noted easily. On the other hand, it should not be so large that it occludes much of the area of the display space, obscuring parts of the display of interest to the user. Therefore, in considering the size of the cursor one should be aware about the tradeoff between notability and hindrance. The tradeoff is unavoidable but there are possibilities to increase the notability of the cursor without increasing its size. For example, the notability of the cursor improves by choosing a color for the cursor that contrasts with the background. Other techniques can diminish the occlusion problem without decreasing the cursor size. The “Silk Screen cursor” for example is semi-transparent to minimize the occlusion problem (Zhai, Buxton & Milgram, 1994).

Once the user notes the cursor, it should communicate the current position of the interaction device in the display space. Just like with

Figure 3-12. The arrow cursor.

Figure 3-13. The hand cursor.

Figure 3-14. The beam cursor.

the WIMP interface 2D cursors in Figure 3-12, Figure 3-13 and Figure 3-14, there should be one position on the cursor that represents *the* position of the interaction device in the display space. This location is known as the "hotspot" of the cursor. In selecting objects, the hotspot the hotspot is the location that is used to identify the target. In the figures, the hotspots are the crosses. The user should be able to deduct the location of the hotspot from the shape of the 3D cursor. Hence, the shape of the 3D cursor should facilitate in revealing the location of its hotspot.

If orientation data of a device is used, for orienting items in the scene for instance, the 3D cursor could provide feedback by representing the orientation of the interaction device. In WIMP interfaces, the 2D cursor only serves to identify the location of an interaction device in the display space. Nonetheless, some interaction devices, such as drawing tablets, sense orientation but their orientation is never reflected via the 2D cursor.

Next, examples of 3D cursor shapes are presented in Figure 3-15 to Figure 3-19 and their appropriateness for representing the state of the interaction device is considered using the issues above. The cursors all convey the position of the interaction device but differ in the other issues.

Figure 3-15.Wireframe cursor.

The most common 3D cursors are wireframe cursors such as skitters and jacks (Bier, 1986) and the tetrahedron cursor used by Zhai, Migram and Buxton (1996). These cursors present both the position and the orientation of the interaction device to the user and the location of the hotspot is clear. They do not obscure much of the display space but their main disadvantage is that they can be hard to find. Figure 3-15 resents a wireframe cursor in ID8Model. Three extra lines accompany the wireframe cursor. These are "assist lines" between the hotspot of the cursor and the boundaries of the workspace. They help identify the location of the cursor.

hoofdstuk 3 25-08-1999 13:54 Pagina 102

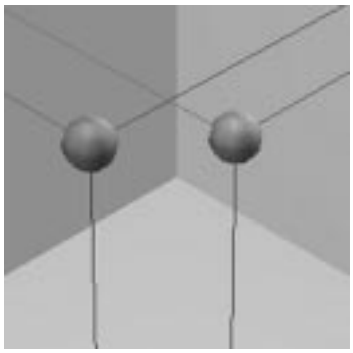


Figure 3-16.Sphere cursor.

The sphere cursor presents the position of the interaction device but it can not reflect its orientation to the user. Unless transparency is used, the sphere cursor may obscure a large part of the display space. In addition, the location of the hotspot is located in the center of the sphere, not directly visible to the user. A semi-transparent sphere cursor has been used in SmartScene (Multigen Inc), in combination with a wireframe cursor (Figure 1-20).

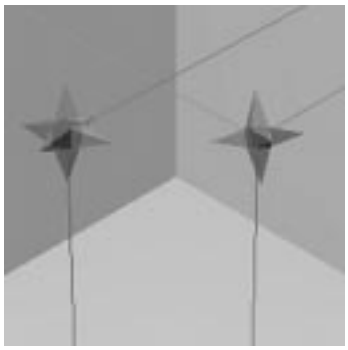


Figure 3-17.Star cursor.

The star cursor was designed in this project to remedy the low notability of the wireframe cursors. The hotspot is a box in the center of the star shape with a different color than the rest of the cursor. The star cursor does not occupy as much display space as the sphere cursor with the same radius. The use of semi-transparency reduces the space needed even further. The star shape communicates the orientation of the interaction device but it does not have a preferred direction.

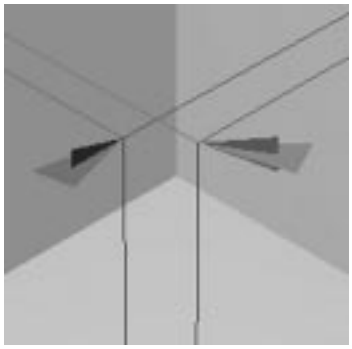


Figure 3-18.Dart cursor.

The dart cursor shape was conceived in this project too. It has much in common with the star shape but it is less symmetrical. It does have a preferred direction to represent the orientation of interaction devices better. Furthermore, the location of the hotspot is situated in the top of the two triangles, resembling the arrow cursor of WIMP interfaces in Figure 3-12.

In unified systems, the possibility exists to make the 3D cursor an extension of the physical interaction device. This has several benefits; the main one is that it overcomes the occlusion problem in unified systems. Without the cursor extension, the device and the

hand of the user could obscure the part of the display space of interest to the user. An example of an extension cursor the cursor of Cubby (Djajadiningrat, 1998), shown in Figure 3-19. The tip of the pen-shaped device is the cursor that is drawn as an extension the device itself.

Figure 3-19. Extension cursor.

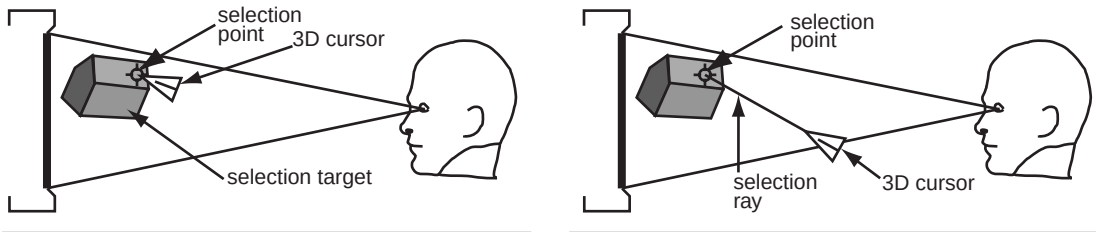
3.3.2 Overview of 3D selection techniques

The 3D cursor has been introduced as the representation of the interaction device in the display space. Next, it is considered how the 3D cursor can be used to acquire targets for subsequent manipulation. First, two selection techniques are presented that are commonly applied in 3D interfaces. It is found that both techniques have advantages and disadvantages and therefore, yet another selection technique is introduced in the next section.

The first selection technique is direct positioning. This is probably the most obvious way to specify an item in the 3D scene. The position of the 3D cursor is used to specify the item, candidate for selection, as shown in Figure 3-20. Common is the use of a threshold; an item is selected when the distance to the cursor is smaller than a threshold value. When there are more candidates for selection, the closest one is selected. The strong point of direct positioning is the simplicity of the concept. The 3D cursor is moved to the position where the selection should take place. Another advantage is that no rotational degrees of freedom are needed in selecting objects, therefore allowing operation with interaction devices that sense position only. The most important disadvantage of direct positioning is its limited range. If the position of the 3D

Figure 3-20. In direct positioning, the cursor is moved towards the object.

Figure 3-21. In spotlight selection, the cursor is used to steer a ray originating at the cursor.



hoofdstuk 3 25-08-1999 13:54 Pagina 104

cursor and the desired target position are separated by a large distance, the interaction device will need to be moved over a large distance before a selection can occur.

The second selection technique, shown in Figure 3-21, is spotlight selection[5] (Liang & Green, 1991). Instead of specifying a selection point in the 3D scene directly, the position and orientation of the interaction device is used to shoot a ray into the scene. The origin of the ray is situated at the hotspot of the 3D cursor. The intersection of the ray and the object specify the selection point. By using a ray, the user needs to specify more degrees of freedom than necessary for direct positioning but the selection of distant objects is facilitated because it is not necessary to move the 3D cursor to the item. The use of the rotational degrees of freedom of the interaction device allows the user to remain in a comfortable posture while directing the ray. Unlike direct positioning, points selectable are constrained to lie on the surface of the objects in the scene. Often, this is exactly what is desired but it might be necessary to specify points within objects or on objects behind the first visible object. In these cases, spotlight selection can be augmented by a mechanism that allows the user to cycle through all intersections of the ray and the objects. Disadvantages of spotlight selection are the cumbersome selection of distant or very small objects and the need for interaction devices to provide rotational degrees of freedom.

Instead of casting a ray, sometimes a translucent cone is used to specify a region of interest. This addresses the problem with selecting distant or very small objects. The distance of each item intersecting the volume of the cone to the center of the cone is calculated and the closest item is selected. A disadvantage of the use of a cone is that it can be hard to predict for a user which of the set of items intersecting the cone will be selected. Also, there is probably not an optimal cone size for all circumstances. The first issue was addressed by Liang and Green (1993) by giving the user feed-forward of the current selection candidate by highlighting it.

[5] Spotlight selection is sometimes known as ray casting. Here, the term ray casting is reserved for rays originating at the center of projection.

3.3.3 Cursor-based ray casting

In cursor-based ray casting, the origin of the ray is defined by the center of projection of the 3D viewing process. Figure 3-22 illustrates cursor-based ray casting. The ray in the figure is only

present for illustrative purposes. Preliminary testing revealed that the position of the cursor alone presented ample feedback for the selection process. Cursor-based ray casting resembles selection with a 2D cursor in WIMP interfaces. In many WIMP applications, an object is selected by moving the 2D cursor in front of the target and pressing the selection button. Cursor-based ray casting employs the same concept; the 3D cursor is moved in front of the target. It is therefore easy to predict whether an object will be selected.

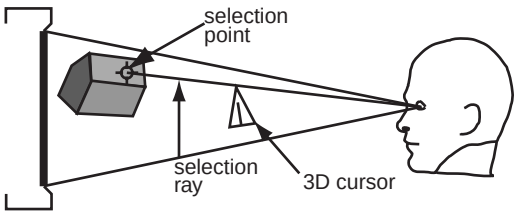


Figure 3-22. In cursor-based ray casting, the cursor is used to steer a ray originating at the viewpoint.

Cursor-based ray casting resembles the technique used in the THRED system (Shaw & Green, 1994) and addresses disadvantages of direct positioning and spotlight selection. Cursor-based ray casting is based on spotlight selection and inherits the advantages associated with it. It has the additional advantage of being well integrated with the presence of the 3D cursor. Because only the position of the cursor is used, it has the advantage that it can be used with interaction devices that do not measure rotational degrees of freedom.

Often, it is convenient for a user to move multiple objects in one gesture. Therefore, a selection mechanism should be present that allows a user to select more than one object. In WIMP interfaces, this is accomplished with either modified selection (the user repeatedly clicks on objects to select) or rubber banding. With rubber banding, a user presses the mouse button and moves the mouse at which point a selection rectangle appears. The rectangle indicates which objects will be selected as soon as the mouse button is released. The same technique can be used with the 3D cursor and is easily integrated with cursor-based ray casting.

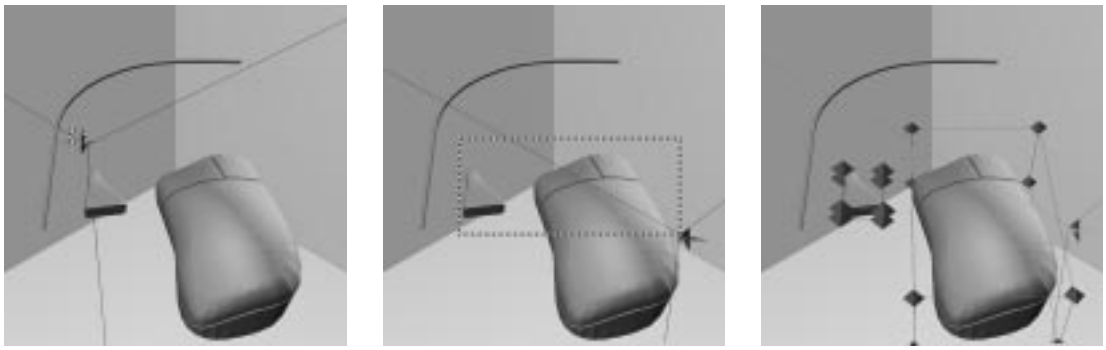
Figure 3-23 shows how rubber band selection is accomplished with a 3D cursor in ID8Model. The figures show a scene with different parts of a model of a mouse as observed by a user looking at the computer screen. In the left picture, the user has positioned the dart cursor in the upper left corner such that none of the mouse parts will be selected when the selection button is activated. After pressing the selection button, a selection rectangle appears on the screen. The middle picture shows the selection rectangle that results from moving the cursor to the lower right corner. Upon release of

hoofdstuk 3 25-08-1999 13:54 Pagina 106

the button, the parts with projections intersecting the rectangle are selected. The selected objects are recognized by the surrounding wireframe box. (CD-ROM, 3-1)

The selection procedure resembles rubber banding in WIMP interfaces because it too provides a selection rectangle. Technically however, the procedure is different because although the user specifies a rectangle, a pyramidal volume is used to select objects. Like single object selection, only two degrees of freedom are needed to specify the corners of the selection rectangle.

Figure 3-23. Rubber band selection of different parts of a model of a mouse with a 3D cursor (for the sake of clarity, the width of the rubber band rectangle edges has been exaggerated).



3.4 Displacing the manipulation space

It was stated earlier that the user should have control over the displacement between the manipulation and the display spaces. Traditionally, displacing the manipulation space is known by the somewhat awkward term recalibration. A good recalibration mechanism is an important part of an interface with spatial interaction because it allows users to remain in a comfortable posture while operating the system. If the working area is large, users might be forced to reach over large distances. This is especially important with spatial interaction devices because it can be very tiring to keep one's hands up in the air for long periods on end.

Recalibration alters the relationship between the display and manipulation spaces and can have a negative effect. In the absence of a recalibration mechanism, a constant spatial relationship exists between an interaction device and the virtual object controlled, such as the 3D cursor. This means that users can employ their "motor memory" when interacting with the virtual world. For instance, when a virtual object is left at the center of the display

space, the user can pick it up later from that same position. The interaction devices need to be brought to the same location they were at, the moment the object was released. After recalibration however, the interaction device can be moved to another position and therefore, the manipulation space moved with respect to the display space. However, it seems inevitable to include a recalibration mechanism into a system with spatial interaction devices, due to the aforementioned reason of comfort.

Next, an overview is given of recalibration mechanisms and their appropriateness in a desktop VR environment. Then, recalibration for two-handed operation is reconsidered because when two devices are used simultaneously, the question arises whether two separate recalibration mechanisms for each hand are desirable. The recalibration mechanisms presented have in common that they only displace the manipulation space. As discussed in section 3.2, it is also possible to rotate and/or scale the manipulation space with respect to the display space. It was concluded that there was no apparent need for this and therefore the discussion of recalibration will only deal with displacement.

3.4.1 Overview of recalibration techniques

Three types of recalibration mechanisms are commonly distinguished: command-based recalibration, “clutching” and continuous recalibration (Hinkley, Pausch, Goble & Kassell, 1994a).

In command-based recalibration, the user activates a “centering” or “homing” command. In systems with a 3D cursor, like JDCAD (Liang & Green, 1991), such a command typically moves the cursor to the center of the scene. With this kind of recalibration, a user can establish a comfortable manipulation space by bringing the hand(s) to the center of the desired space and activating the home command.

The second kind of recalibration is clutching (or ratcheting) which is often applied in spatial interfaces (Chung, 1992; Galyean & Hughes, 1991; Ware, 1990). Clutching was already encountered in the previous chapter when the difference between absolute and relative devices was discussed. As a mouse is a relative device, it can be lifted and lowered in another location without affecting the cursor. The result is that the manipulation space has been shifted relative

hoofdstuk 3 25-08-1999 13:54 Pagina 108

to the display space. In most spatial interfaces, a clutch button is used to temporarily disconnect the interaction device from the entity controlled, the 3D cursor or an object in the scene for instance. As soon as the clutch button is released, the connection is reestablished. This allows the user to perform movements in a series of grab and release cycles.

Lastly, continuous recalibration makes recalibration invisible to the user. This is not possible in all circumstances but in immersive VR, it is applicable and universal. In immersive VR, the manipulation space of both hands follows the user's body. To change the manipulation space, the user simply moves to another position. The display space can be moved by moving one's head. Therefore, both the manipulation and display spaces can be moved. Generally however, the relationship between the user's hand in the actual world and the representation of the hand in the virtual world is not dynamically changed.
After the implementation of the first two recalibration mechanisms in ID8Model, the tentative conclusion was made that command-based recalibration and clutching are both useful in a desktop VR system but under different circumstances. Command-based recalibration seemed most appropriate to establish the initial working area, usually at the start of a session. Clutching appeared more suitable during the session because it is easier to predict the result of a clutch action and it offered the user more control. Continuous recalibration was not evaluated because it does not apply in a non-immersive VR setting.
Another finding was that the amount of expected use of the clutch mechanism is dependent on the settings of the C/D ratio. The greater the C/D ratio, the more the clutch mechanism needs to be used. To reduce the amount of clutching to a minimum, one is tempted to select a low C/D value but that also degrades the achievable precision. Thus, a tradeoff is encountered between achievable precision and the amount of clutching needed.

3.4.2 Clutch techniques for two-handed operation

The clutch techniques presented thus far did not consider two-handed operation. Without clutching, the manipulation spaces for both devices have a fixed relationship. If a clutch mechanism is provided for both devices separately, then the two manipulation spaces of the devices can be separated. It was shown that clutching

changes the location of the manipulation space, thereby interfering with the relationship between the real world and the virtual world. The result of separate clutching is that the offset between the manipulation spaces is changed, thereby interfering with the user's kinesthetic sense of where the hands are relative to each other. For instance, when the manipulation spaces for both interaction devices are identical, the user can bring two devices together and can assume that their 3D cursors will come together too. As soon as clutching has changed the offset between the manipulation spaces, this relation is disturbed and the interaction devices will have to be moved to different relative positions to bring the cursors together.

In the previous chapter, it was shown that people's awareness of where their hands are relative to each other is strong. This suggests that the spatial relation between the two hands should be kept intact. This can be accomplished by using the same manipulation space for both hands, like the combined clutch in the THRED system, described by Shaw and Green (1994). On the other hand, it was already mentioned that it could be necessary to allow for an offset between the manipulation spaces because the devices can collide when interacting with objects that are close together.

It is difficult to decide what the best strategy is for two-handed clutching because it is not clear what is more important. Is a fixed relation between the interaction devices important or the ability to create an offset to facilitate working with devices in the same area? Therefore, all three approaches for two-handed operation were supported in ID8Model, no clutch for both hands, combined clutch for both hands and a separate clutch for each hand. In Chapter 5, an experiment is described in which the three clutch mechanisms are compared.

3.5 Creating two-handed interfaces

In this section it is presented how the cursor can be used to create bimanual spatial interfaces. First, it is discussed how the 3D cursor permits the use of spatial interaction devices as well as non-spatial devices. Then it is shown how multiple devices can be used at the same time without conflicts. Finally, examples of three modeling tools in the ID8Model application show how two-handed operation of different devices can result in intuitive and efficient interaction.

hoofdstuk 3 25-08-1999 13:54 Pagina 110

3.5.1 Supporting non-spatial interaction devices

Above, it was stated that the ID8Model program should support spatial as well as non-spatial interaction devices. This was motivated by the belief that spatial interaction is not necessarily the optimal form of interaction all of the time. Not all of the modeling activities require control with the 6-DoF that spatial devices provide. Instead of ignoring the excessive degrees of freedom in software, a device sensing lesser degrees of freedom can be used with the potential benefit that the device can be more optimized for the task. The Turntable for instance, was designed for orienting a model only. Next, it will be shown how the 3D cursor can be used to visually represent the state of non-spatial devices also. As an example, devices supported by the operating system of the computer will be used. These are devices like the mouse or the graphics tablet, connected to the 2D cursor of the operating system.

Pointing devices used for 2D cursor control are often 2-DoF devices. The mouse for instance, senses displacement in two dimensions. However, the graphics tablet senses the orientation of a stylus on its surface and this data is often ignored in applications. In ID8Model, the orientation readings are used to orient the 3D cursor. The significance of this becomes clear in the description of the assembly tool below. The support offered for pointing devices is a combination of conversion of the coordinates of the 2D cursor to the position of the 3D cursor and support for the extra degrees of freedom that the devices sense.

The mapping of the 2D cursor coordinates to the position of the 3D cursor resembles the link between the manipulation and display spaces of the spatial devices. The difference is that the cursor is moving in a 2D plane, whereas the 3D cursor is free to move in the

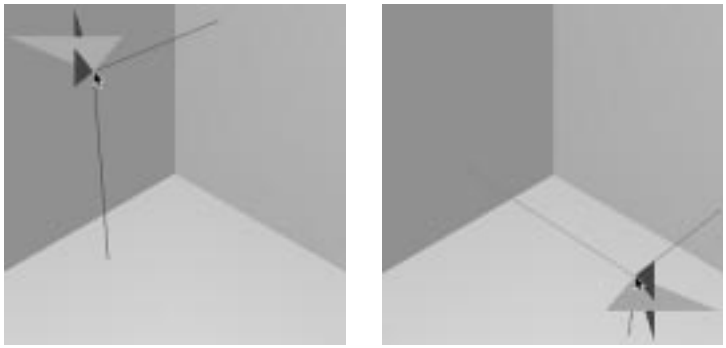


Figure 3-24. The position of the 3D cursor follows the position of the 2D cursor because the hotspots of the 2D and the 3D cursor coincide (for the sake of clarity, the size of the 3D cursor has been exaggerated).

scene. It was decided to mimic the interaction techniques of standard WIMP style modeling applications. This was accomplished by making the projection of the hotspot of the 3D cursor coincide with the hotspot of the 2D cursor. Figure 3-24 shows the 2D cursor in two different positions and the associated positions of the dart cursor that follows the 2D cursor (CD-ROM 3-2). The lines originating from the hotspot of the dart cursor are the assist lines introduced above. Making the hotspots of the 2D and 3D cursor coincide has the benefit that a close fit exists with the WIMP style interaction techniques needed to interact with other applications and with the operating system.

3.5.2 Supporting multiple interaction devices

With multiple interaction devices, a single 3D cursor can not effectively represent the state of all the devices at the same time. Additionally, when multiple devices are connected to a single 3D cursor, conflicts can occur because the devices will compete in setting the position and orientation of the cursor. Therefore, all devices have a unique cursor in ID8Model. This prevents device conflicts and helps the user to identify the state of each individual device. The prerequisite is that the position, orientation and button states of different devices can be uniquely identified. The introduction of device identifiers breaks with the tradition in current operating systems that cannot distinguish different devices. Multiple cursors solve the problem of conflicts for cursor control but it shifts the device conflict problem down. When two cursors can both be used to move objects for instance, conflicts may arise when they move the same object.

To avoid conflicts between devices, ID8Model divides devices in two categories: devices for the dominant hand and devices for the non-dominant hand. With this division, the application can assume that only two devices are active simultaneously since a user can only hold and manipulate with two devices[6]. Therefore, the software needs to avoid conflicts between the two active devices only. An additional benefit of the division between dominant and non-dominant hand devices is that it allows the creation of asymmetric interfaces with different roles for each hand. To support this, the application needs to know which hand is used to hold a device. Currently, ID8Model inquires the user for identification of the hand used to hold a device.

[6] The implication of allowing only one active left- and right-handed device is that ID8Model is a single user system.

hoofdstuk 3 25-08-1999 13:54 Pagina 112

Next, some of the tools in the ID8Model application are presented to demonstrate how the 3D cursor is employed in two-handed modeling tasks. The examples show how two devices can be active simultaneously and how different interaction devices can be used with the same tool.

3.5.3 Assembling with two hands

The assemble tool is used to move and orient objects. With the tool, models can be assembled out of different objects. In the following description, the use of the dominant hand is described first. It is shown how the 3D cursor is used to manipulate objects with different devices. Then, two-handed assembly is considered and two different interfaces for bimanual assembly are presented.

The left column of Figure 3-25 shows how objects can be moved and oriented with a mouse. In the first picture, the mouse has been used to move the 2D cursor in front of the model. The 3D cursor is located under the 2D cursor on the surface of the model. In the next picture, the button on the mouse has been pressed to select the object so that it can be moved. The third picture shows how subsequent movements with the mouse moved the model up, parallel to the screen (CD-ROM 3-3). The spheres and pyramids surrounding the model are widgets (Brookshire Conner, Snibbe, Hemdon, Robbins, Zeleznik & van Dam, 1992) that can be used to rotate the model. The widgets appear when the object is selected. The fourth picture shows the object after it is rotated with one of the ring shaped widgets (CD-ROM 3-4). With ring widgets, objects can be rotated around an axis through their center. The pyramid widgets allow rotations about an arbitrary rotation axis, with the virtual sphere algorithm (Chen, Mountford, & Sellen, 1988) (CD-ROM 3-5). The last picture in the column shows that moving the object towards the viewer can be accomplished by pressing a key on the keyboard while dragging the object (CD-ROM 3-3).

When the stylus (Figure 2-5) of a graphics tablet is used to move objects, the orientation of an object can be specified additionally. In the middle column of Figure 3-25, it is shown that the 3D cursor represents both the position and the orientation of the stylus and that the state of the cursor is used to update the model. The first picture shows the 3D cursor on the surface of the model under the 2D cursor. This resembles the situation with the mouse. After

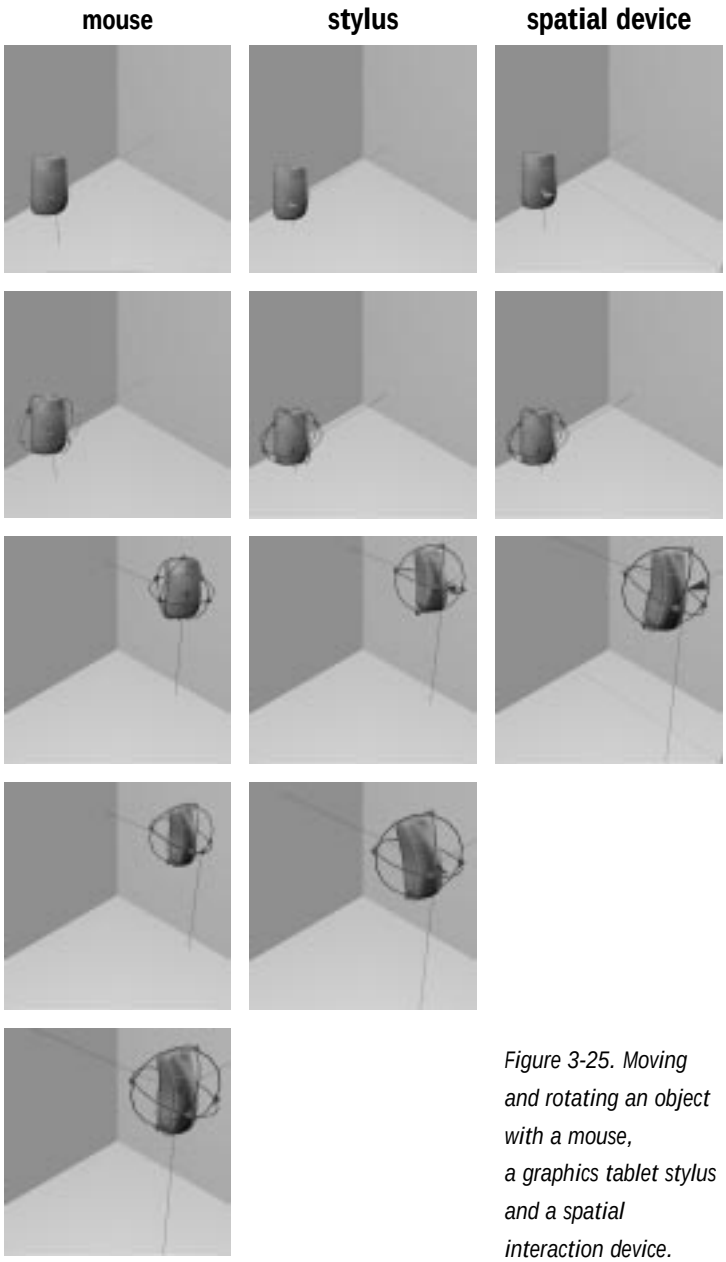


Figure 3-25. Moving and rotating an object with a mouse, a graphics tablet stylus and a spatial interaction device.

selecting the object (second picture), the model moves as well as rotates by moving the stylus, as can be seen in the third picture. Rotation takes place about the hotspot of the 3D cursor that maintains on the surface of the model (CD-ROM 3-6). This has no technical reason but it reflects the way real objects rotate when held at a point on the surface. The last picture shows that with a stylus, a key on the keyboard must be used to move the object closer to the viewer.

When using a spatial interaction device, the user can fully specify the position and orientation of an object. Unlike with a stylus, spatial interaction devices can specify the position of objects without the need to use the keyboard or to change view. This is shown in the right column of Figure 3-25. The first picture shows the object before it is selected with a spatial device. The 3D cursor is located in front of the object. As soon as the selection button on the device is activated, the object moves towards the cursor (second picture) and the object can be moved as well as rotated. The last picture shows that movements of the device, reflected by the cursor, change both the position and orientation of the object (CD-ROM 3-7).

hoofdstuk 3 25-08-1999 13:54 Pagina 114

In the development of ID8Model, two interfaces for two-handed assembling have been explored. The interfaces differ in design philosophy. In one interface, the asymmetry of the use of both hands is reflected in the interface itself. The other interface is symmetric but allows a user to use the hands asymmetrically. Later (in Chapter 5), the results are presented of an experiment in which both interfaces were compared with an assembly task.

Dependent selection

With dependent selection, the asymmetry of the use of both hands in assembling is reflected in the interface. The asymmetric interface was motivated by the belief that selection is more difficult with the non-dominant hand than it is with the dominant hand. Therefore, with dependent selection, a user needs only to select objects with the dominant hand as described above. Objects moved with the non-dominant hand follow from the selection made with the dominant hand. More precise, all the objects in the scene are dragged with the non-dominant hand except those moved with the dominant hand. In this case, a cursor for non-dominant hand devices is not needed since the non-dominant hand is not used for selecting. Figure 3-26 shows that when the dominant hand is not moving objects, the non-dominant hand controls the position and orientation of all the objects in the scene (CD-ROM 3-8). In this situation, the non-dominant hand aids the selection of a target by bringing the object close to the cursor of the dominant hand device. In Figure 3-27, the dominant hand has selected an object and then the non-dominant hand can move the

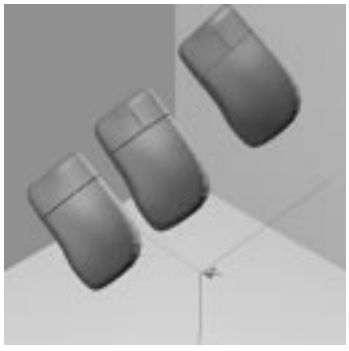
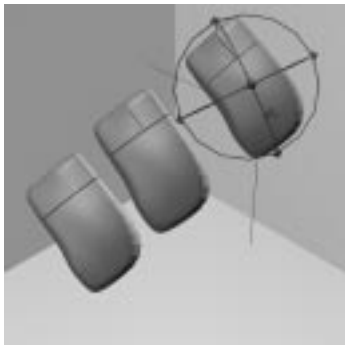


Figure 3-26. Moving objects with the non-dominant hand while none is moved with the dominant hand.



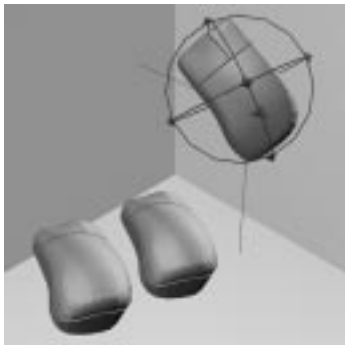


Figure 3-27. Moving objects with the non-dominant hand that are not moved with the dominant hand.

rest of the objects in the scene (CD-ROM 3-9). Then, the non-dominant hand can aid in assembling objects by positioning the target object such that placement of an object with the dominant hand is facilitated.

Independent selection

Independent selection presents a symmetric interface to a user. Both hands can be used to select objects with a 3D cursor. Figure 3-28 shows how both hands can be used to select two objects and how they are brought together for assembly (CD-ROM 3-10). Although the interface is symmetric, a user can use it asymmetrically. For example, one hand can be used to orient an object roughly while the other hand places another object on top of it with precision.

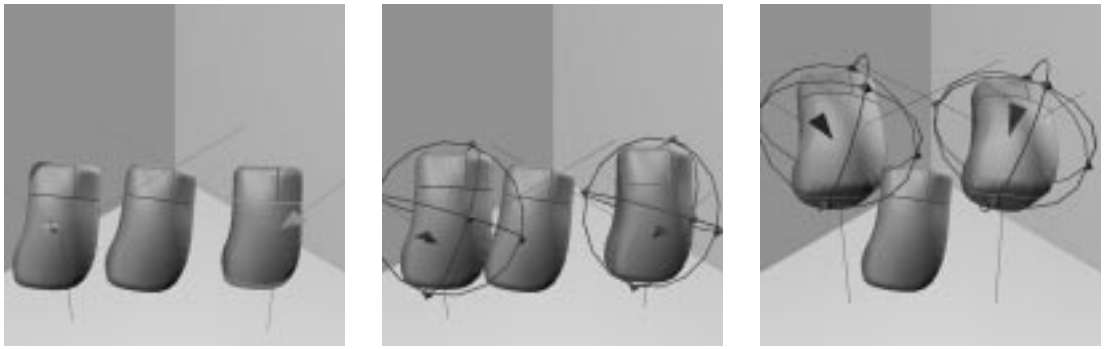


Figure 3-28. Selecting and moving two objects at the same time with independent selection.

3.5.4 Drawing with two-hands

The draw tool in ID8Model can be used to draw on the surface of objects. With this tool, devices held in the dominant hand are used for the actual drawing while non-dominant hand devices are used for holding the objects being drawn. With the non-dominant hand, objects to be held are selected and moved using the 3D cursor as with the assemble tool.

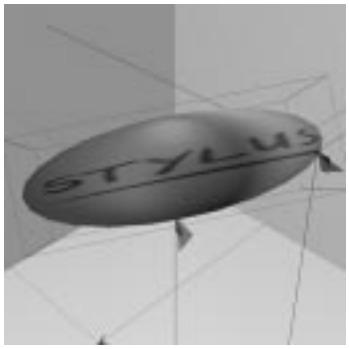


Figure 3-29. The draw tool used with a stylus in the right hand and a spatial interaction device in the left hand. The 3D cursor used to draw touches the surface of the model.

The tool shows how different interaction devices can be used for the same purpose but that their use imposes a different experience. Figure 3-29 for instance, shows drawing with a stylus held in the dominant hand. The pressure applied on

hoofdstuk 3 25-08-1999 13:54 Pagina 116

the stylus determines the width or opacity of the stroke drawn on the surface of the object (CD-ROM 3-11). Figure 3-30 shows how a spatial interaction device is used to draw on the surface.
With a spatial interaction device, the distance to the surface can be used to control either width or the opacity of the line stroke drawn on the object (CD-ROM 3-12).

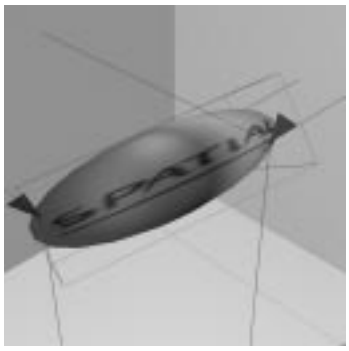


Figure 3-30. The draw tool used with spatial interaction devices in left and right hands. The 3D cursor used to draw does not touch the surface of the model.

Impressions of users working with the tool suggested that the use of a stylus for drawing is more convenient than drawing with a spatial device.

3.5.5 Sculpting with two hands

The sculpt tool allows the designer to change the shape of an object by pressing another object against its surface. Figure 3-31 presents two pictures of the sculpt tool being used (CD-ROM 3-13). Devices held with the dominant hand are used to move the sphere shape. As soon as the sphere touches the surface of an object with the select button pressed, the surface of the object is displaced. Devices for the non-dominant hand can be used to move and orient objects like with the draw tool. In the figures, the dart cursor for the non-dominant hand device is used to hold the object. This allows a user to hold an object with one hand while sculpting with the other hand. The size of the sphere can be changed to permit a user to sculpt locally or globally. In theory, arbitrary shapes can be used to sculpt objects but only the sphere has been used so far.

Figure 3-31. The state of the model before activation of the sculpt tool. The left hand is used to hold the model with the 3D cursor. The right hand controls the position of the sphere.

Although some aspects of the tool need additional work, it demonstrates how complex geometric modeling operations can be operated through an intuitive and effective interface. With the aid of spatial interaction, sculpting on a computer can be as natural as sculpting with clay. It could even surpass some of the limitations of physical materials

Figure 3-32. The state of the model after activation of the sculpt tool. The right hand has moved the sphere into the surface, thereby displacing vertices.

and tools. For example, the tool allows bringing the sphere cursor inside the model and pushing its surface outwards.

3.6 Designing the Frog, a spatial interaction device

In the previous chapter, it was established that a spatial interaction device would be appropriate for manipulating objects in the modeling space. It was also found that ergonomical issues of interaction devices are often overlooked in literature on pointing devices. However, factors like the shape and the weight of the device were shown to have a significant impact on the usability of the device. These factors were encountered in the development of the Frog spatial interaction device that was needed for prototyping the above-described interaction techniques. Next, examples of spatial interaction devices are presented, followed by the design of the Frog.

3.6.1 Examples of spatial interaction devices

Before developing the Frog, the availability of spatial interaction devices was established. It was concluded that there were little references to spatial pointing devices in literature and only few commercial examples were found. Some examples of spatial devices can be seen in Figure 3-33 to Figure 3-38 while Figure 3-19 presented another pen-shaped spatial device earlier. The devices have in common that they all use the same sensing technology. The electromagnetic sensing technology they use is reliable, accurate and fast but has a major disadvantage; a wire is needed to connect the sensor to a processing unit. The result is that although the sensors are light and small, they do not move and revolve freely. However, the other aspects of the sensing technology proved beneficial for creating spatial interaction devices.

Figure 3-33. 6D Mouse.

Figure 3-34. 3BALL.

Figure 3-35. The Bat.

The devices in Figure 3-33 and Figure 3-34 are commercial devices, provided by manufacturers of electromagnetic sensors. The other devices have been created for research projects and are not available commercially. The 6D Mouse is held in the hand with a power grip. All the other devices are held between the fingers with

hoofdstuk 3 25-08-1999 13:54 Pagina 118

Figure 3-36. Hinckley's props.

Figure 3-37. Hinckley's ball.

Figure 3-38. Zhai's ball.

a precision grip. Compared to power grip, the precision grip allows for precise manipulation. In the previous chapter, it was shown that besides higher precision, the use of smaller muscle groups like the muscles of the fingers leads to higher performance in pointing tasks.

The devices in the figures differ in the presence of buttons. Some devices, like the neurological props devices in Figure 3-36 and the ball in Figure 3-37, do not provide buttons at all. These devices offer the user the most freedom to decide how to hold the device. Whenever a button is used on a device, the number of possible grips diminishes, especially when buttons need to be used while moving the device.

6.3.2 Design of the Frog

The Frog device presented here was designed because the search for spatial interaction devices established that very few commercial devices were available and none of them exactly fitted our needs. It was decided that a device needed to be built, based on the previous devices. The first requirement for the device was that it should be used with a precision grip. In addition, the device should have two buttons so that the selection and clutch mechanisms can be activated from the device without the need to acquire a separate device such as the keyboard. Because two-handed operation needed to be supported, a device was needed for either hand. It was decided to design a symmetrical device that could be used with both hands. Other shape requirements where that a fixed grip should be provided while allowing for manipulation with the fingers only and the shape should afford the intended grip. Weight should be reduced to the minimum because with spatial interaction, the device is often held up in the air.

Figure 3-39, Figure 3-40 and Figure 3-41 show the Frog from different viewpoints (CD-ROM, Frog model). The two buttons found a

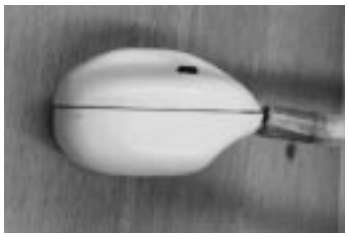


Figure 3-39. Side view of the Frog.

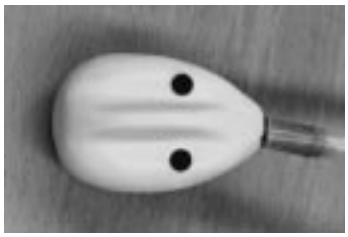


Figure 3-40. Top view of the Frog.

Figure 3-41. Frog held between the fingers.

Figure 3-42. Inside of the Frog.

place on top of the device and they are operated with the index and middle fingers. The thumb is used to support the device so that the fingers and the thumb together provide the desired precision grip. The ridge between the fingers and the dent for the thumb help to provide a stable grip and they hint the user about the intended grip. The Frog is symmetrical so that it can be used with either the left or the right hand. At the same time, the shape of the device is such that the user can relate the orientations of the device and a cursor easily. It is just large enough to enclose both the sensor and the two buttons.

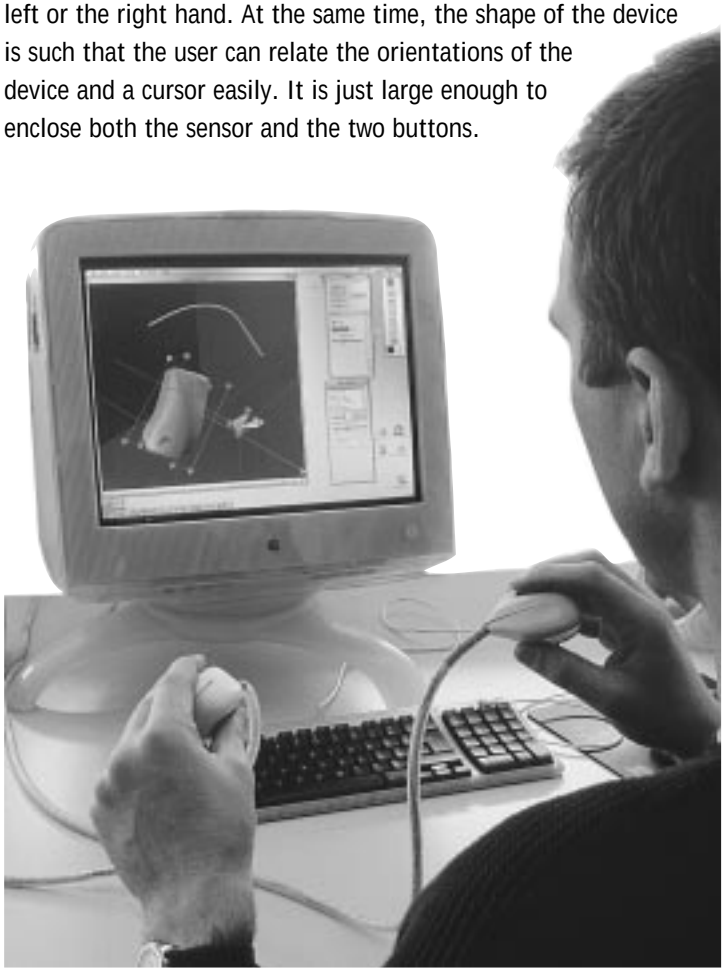


Figure 3-43. Frogs in use with the ID8Model program.

hoofdstuk 3 25-08-1999 13:54 Pagina 120

3.7 Conclusion

In this chapter, it has been established that spatial interaction changes interaction with CAD systems completely. The interfaces of current CAD programs need to be changed to enable operation with spatial interaction devices. Therefore, spatial interaction could not be achieved by substituting the traditional interaction devices, such as the mouse and the graphics tablet, for spatial devices, like the Frog, only. Instead, an integral approach was taken with attention for both devices and software. The creation of the novel interaction techniques presented in this chapter was worthwhile since preliminary evaluations showed that spatial interaction can lower the barrier between the designer and the modeling task at hand. This is especially beneficial for modeling in the conceptual phase of the design because it allows a designer shift attention from the CAD system to the design. This permits the designer to explore more of the design space in a shorter time-span.

The interaction techniques presented in this chapter allow operation with spatial as well as non-spatial interaction devices because of two reasons. First, it allowed the comparison of different devices for the same tasks. Second, by evaluating different devices, it was found that spatial interaction is sometimes not the preferred form of interaction in a 3D modeling system. When drawing on objects for example, a graphics tablet with a stylus is favored, probably because it resembles drawing with paper and pen. Therefore, the designer can utilize the drawing skills acquired with traditional tools. When drawing with a spatial device, the designer is required to make gestures in 3D, like with an airbrush.

Novel interaction techniques were needed for bimanual operation of CAD systems also. Most current CAD systems are not equipped to handle input from more than one interaction device concurrently, let alone that they are suited for bimanual use. The bimanual interaction techniques presented above were developed with the concurrent everyday use of both hands in mind to prevent the creation of a two-handed interface to difficult to operate to be useful. In the previous chapter, it was shown that the hands play different roles in bimanual activities. During the design of bimanual interfaces, it was found that the principles of bimanual activity presented in the work of Guiard (1987) are helpful in this regard. However, Guiard's work is not a guide for two-handed interface

design. Rather, it can help limit the design space of possible two-handed interfaces by rejecting those interface designs that do not correspond to people's everyday use of their hands. Often, several interface designs can be created that all adhere to this criterion. For example, two different interfaces were presented for assembling objects. Both interfaces were designed to support two-handed operation corresponding to the everyday use of the two hands. To establish whether the anticipated use of two hands realizes in each of these interfaces, user testing is needed.

In the next two chapters, user tests are described in which several aspects of two-handed operation were explored. The next chapter establishes whether two-handed operation has benefits in an assembly task. The assembly tool with dependent selection is used in a comparison of one- and two-handed operation. The questions are: is two-handed operation faster than one-handed operation, can two-handed operation help divide the workload over the two hands and how big is the contribution of each of the hands in the combined activity?
In Chapter 5, open questions are addressed regarding the optimal interface for assembly and the influence of the clutch mechanism. Again using the assembly tool, assembly with dependent selection is compared to assembly with independent selection. In addition, the clutch mechanisms presented above were compared.