A SURVEY OF INTERACTION DEVICES AND TECHNIQUES FOR 3D MODELING

In this chapter, a presentation is given of the current state of research into computer human interaction in a number of areas relating to 3D modeling. The presentation starts with a brief overview of the problems of current modeling applications. In support of the conclusions of the previous chapter, it is established that an important restricting factor for conceptual modeling is that the designer has to work with devices registering two degrees of freedom for creating 3D form. In addition, the designer can only use one hand effectively for interacting with models in applications. Therefore, the focus of the rest of this chapter is to introduce the results of previous research in these areas.

First, classifications of interaction devices are used to find the requirements for a device that is better suited for conceptual modeling than the devices currently used. Classifications found in the literature indicate that a device capable of sensing 6-DoF will allow a designer to position objects and tools in 3D directly. Then, the literature on the use of interaction devices in applications is discussed to find that there are still a lot of unanswered issues regarding the use of devices sensing 6-DoF in applications. Finally, the use of two-handed interaction is established based on the results of previous research. It is found that to support the concurrent activity of two-hands, the interaction designer should consider the capabilities of the hands. Not only do the hands differ from each other but they also play a different role in the combined activity.

2.1 Restrictions when interacting with 3D modeling applications

In the previous chapter, shortcomings of the current CAD tools were identified when used for conceptual modeling. CAD tools were defined as the combination of a computer system and the application. One of the problems identified with the current CAD tools was

they do not allow a designer to employ all the skills that traditional tools such as clay modeling tools do. This problem was attributed to the interaction devices of the CAD tools. Most CAD tools are operated with a mouse and a keyboard that constitute a narrow bandwidth channel through which the skill of the designer has to pass. On the one end is the designer with motor capabilities and skill. On the other end is the application with the model and the tools the designer can use to manipulate the model. The traditional tools allow a designer to position and orient objects and tools in 3D directly with two hands. In contrast, the mouse is the only device that CAD tools provide for direct manipulation and it can be moved in two dimensions only. Several solutions can be found in CAD tools that aim to address the discrepancy between manipulating in 3D with interaction devices that register movements in two dimensions. In this section, a brief summary is presented.

2.1.1 Insufficient degrees of freedom

The 2D interaction devices present two kinds of mapping problems. The first is that only movements in two dimensions at a time can be made. The movements of an interaction device are mapped to the position of a 2D cursor on the screen. To specify the position of an object however, three dimensions are involved. Therefore, the user needs to position the object in successive gestures for which two solutions can be identified in CAD applications.
The first solution is to provide the user with three orthogonal views.

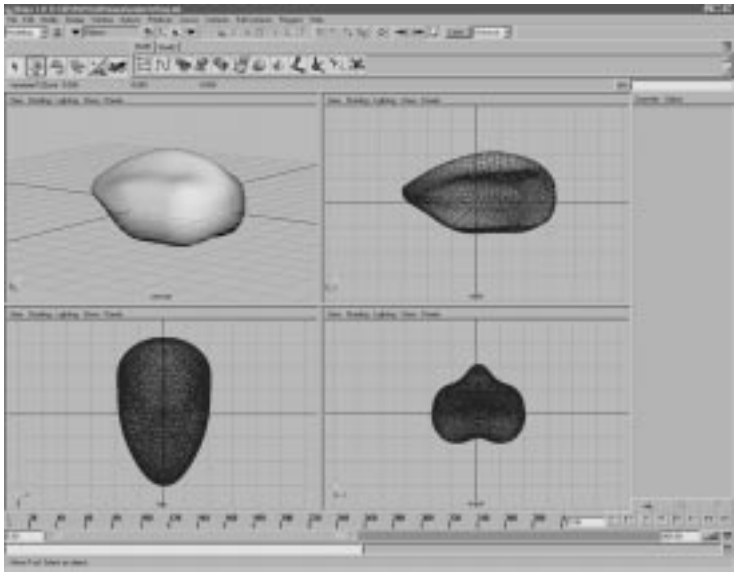
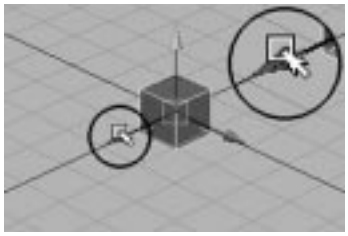


Figure 2-1. Typical screen layout in modeling applications with four views on the same object.

Figure 2-1 presents a typical screen layout of CAD applications with four views on the same model. Commonly, the three orthogonal views provide an isometric view on the model while one view provides a perspective view. In the three orthogonal views, the user can move objects parallel to the screen. Successive two-dimensional movements of the cursor in the orthogonal views enable the user to position objects in three dimensions.

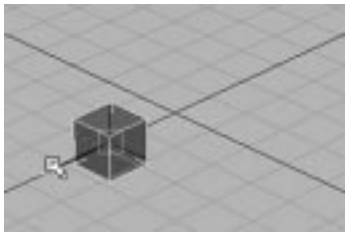
To enable the user to move objects not parallel to the screen, some applications provide user interface elements known as three-dimensional widgets (Brookshire Conner, Snibbe, Hemdon, Robbins, Zeleznik & van Dam, 1992). A three-dimensional widget is the combination of a 3D object and associated behavior. They can be used for moving objects and for modeling operations such as twisting, bending and tapering. Figure 2-2 presents three arrow-shaped widgets added to an object. Using the widgets, the user can move the object along the direction indicated by the arrow shape of the widget. Although widgets allow the user to move an object in the scene directly, objects can still not be moved in more than two dimensions at a time. With the arrow-shaped widgets in the figure, positioning objects in 3D still involves successive movements using different widgets.

Figure 2-2. Using widgets to position a cube in a modeling application.



The mouse is moved such that the cursor appears in front of the conical part of the widget.

The button of the mouse is pressed and the other two widgets disappear.



The mouse is moved and the cube moves in the direction indicated by the shape of the widget.

The second mapping problem is that orientations in 3D can not be specified directly because 2D interaction devices specify the position of the cursor. Several solutions have been presented for that problem in the past with different levels of success (Chen, Mountford & Sellen, 1988). However, they all require the user to convert the intended rotation into movements in 2D. In addition to the type conversion problem, these solutions suffer from the same problem encountered for moving objects in 3D. Often, the user is required to subdivide the intended rotation into sequential rotations, for instance about successive axes.

Both a shortage of degrees of freedom and a conversion of degrees of freedom introduce a barrier between the designer and the design, especially for conceptual modeling. The designer should be allowed to focus on the task of creating form and the tools should be low

profile so as not to hinder the train of thought. However, when compared to the traditional tools, current CAD tools force the designer to think of operating the tool and to divide intended modeling actions into unnatural small chunks.

2.1.2 The number of degrees of freedom necessary

The aim of the subsequent sections is to select interaction devices and interaction techniques that match the work methods of traditional modeling techniques more closely. It is hoped that this leads to a system that allows the designer to concentrate on the act of designing. When modeling with clay for example, the designer can move both hands freely to position objects and tools directly in 3D. To mimic this situation with a CAD system, the system can be equipped with two interaction devices that capture movements in space. This suggests that each device should be capable of sensing positions and orientations in space. Therefore, each device senses 6-DoF. In addition, the software should respond to movements of both hands. With one device in each hand, the designer can control 12-DoF at any time.

When observing designers working with traditional tools, one could argue that not all these DoF are needed all of the time and that constraint operation is common. For instance, models are sometimes put on the table surface so that movements are planar for precise positioning. In this case, using a mouse seems more appropriate than using two 6-DoF devices because no superfluous DoF need to be controlled. However, even with constraint operation, the two hands often move freely in 3D. Consider for instance the situation when a model is clamped to the table and a file is used to modify its shape. This seems to be a constraint situation but actually, both hands are used to carefully position the file relative to the model. Therefore, 6-DoF interaction with two hands will be considered and constraint interaction will be considered a special case.

Constraint operation can be supported in different ways.
For example, when a designer wants to rotate an object about one axis only, a spatial interaction device can be used but inadvertent movements of the user could displace the object in addition to the desired rotation. When using 6-DoF interaction devices, the displacements can be avoided by discarding the superfluous position

and orientation readings in software. However, an alternative, and potentially preferable, approach is to use an interaction device that registers the desired rotation only (the Turntable could be used for this purpose). The advantage is that the design of the specialized device can be targeted to the task it is to control.
Another example is the drawing on the surface of 3D objects. A spatial interaction device can be used for this although not all the DoF of the device are needed to specify locations on the surface of the object. When a graphics tablet is used for the same task, it registers the location of the pen with the needed DoF. An additional advantage is that the shape of the pen is optimized for the drawing task.

2.2 Evaluating pointing devices

In this section, a search is conducted for the functional requirements of an interaction device that is better suited for 3D modeling than the mouse. The scope of the search is limited to pointing devices, a set of interaction devices of which the mouse is a member. Since there are so many pointing devices, a systematic approach is presented here. First, interaction devices in general and, more specifically, pointing devices are introduced. Then, ways of discriminating and classifying pointing devices are presented. This is followed by the formulation of the functional requirements.

Traditionally, devices used to communicate with a computer system are known as input devices. In the words of Card et al. (inspired by Baecker & Buxton, 1987):

"basically, an input device is a transducer from the physical properties of the world into logical parameters of an application"
(Card, Mackinlay & Robertson, 1991; page 103)

Using the term input device implies a one-way communication between the user and the computer system and emphasizes the device as an object, not as a tool for a user to accomplish a task. The term interaction device was proposed instead to accentuate the use of the devices (Baber, 1997). This was motivated by the desire that the design and use of these devices would get as much attention as the other aspects of human computer interaction.
To reflect the change of philosophy and to stress the task the user

needs to complete, we can formulate the following alternative observation:

basically, an interaction device is an input device that a person uses to perform a task in an application

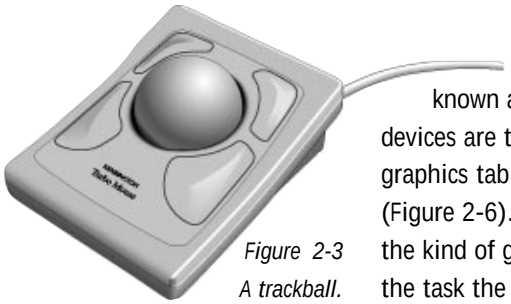


Figure 2-3 A trackball.

The mouse belongs to a group of interaction devices known as pointing devices. Some other familiar pointing devices are the trackball (Figure 2-3), the joystick (Figure 2-4), the graphics tablet (Figure 2-5) and the touch screen (Figure 2-6). The name of this group refers to both the kind of gestures made with the devices and the task the devices most commonly operate. The design of a pointing device determines what kind of movements can be captured. The mouse for instance captures planar movements and the joystick senses rotations about two axes. Since many applications are of the WIMP type (and therefore apply a cursor), pointing devices are often used to control the location of the cursor. Hence, the name pointing device: the device controls the position of the cursor which, in turn, is used to point at items on the computer screen.

Figure 2-4. A joystick

Figure 2-5. A graphics tablet with stylus and puck.

Figure 2-6. A touch screen.

As can be seen in Figure 2-7, the device itself is only a part of the whole communication between the user and a WIMP application. The figure shows how the mouse controls the position of the cursor. The mouse transmits the displacement data, measured by sensors on the ball of the mouse, to the driver running on the computer. The driver updates the position of the cursor with the incoming displacement data. Often, the driver applies a transfer function (for example, a gain) to the incoming data. The user interface management system (UIMS) forwards the position of the cursor to the current application.

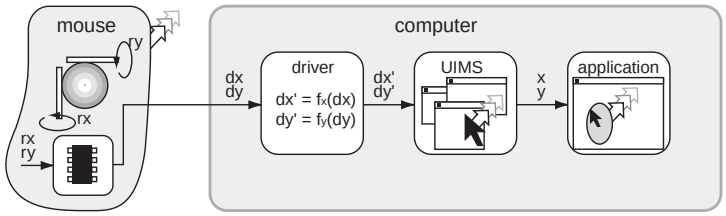


Figure 2-7. Using a mouse to communicate with a WIMP application.

Although pointing devices are probably the most common and widespread of all interaction devices, they are not the only means for a user to communicate with an application. Speech recognition technology for instance, allows a user to communicate with an application by giving spoken (procedural and contextual) commands. Another example is glove-based input, in which the user can operate an application by gesturing. However, the focus in this chapter will be on pointing devices.

2.2.1 Discriminating factors of pointing devices

To gain insight into the characteristics of pointing devices, the task that the devices are supposed to control is disregarded for the moment. Instead, the properties of the devices themselves are considered first, starting with a presentation of ergonomical and operational factors. Operational factors of pointing devices are used in the classifications that are described later.

Ergonomic factors

Pointing devices can be discriminated by the limb that is used to control the device. Most pointing devices are operated with the hand but other limbs such as the finger, the eyes, the head and the foot are also used. Based on the human motor skills, different levels of pointing performance are reached for device operation with different limbs.

The time to complete a pointing task is often predicted with the help of Fitts' law (Fitts, 1954). In Fitts' work, a mathematical relationship was established between the time to reach a target, the distance to the target and the target width. Based on Shannon's theorem from information theory, Fitts established Equation 2-1 that is now known as Fitts' law. The equation can be used to predict how much pointing time increases when the target is moved away or reduced in size. Several modifications to the original law have been proposed since but they all relate the movement time to the distance between start point and target and the target width.

$$MT = a + b \log_2(2A/W) \text{ (sec)} \quad (2\text{-}1)$$

MT movement time
a,b empirically determined constants
A the distance from start point to target
W the width of the target

The logarithmic term is dependent on the parameters and determines the difficulty of the task. It is known as the "index of difficulty", ID (Equation 2-2) and measured in bits. The "index of performance" IP is defined as the index of difficulty divided by the movement time (Equation 2-3) and is therefore measured in bits per second. The slope parameter b is related to the motor system processing-rate. Note that when the value for the constant a in Fitts' law is zero, the index of performance reduces to 1/b.

$$ID = \log_2(2A/W) \text{ (bits)} \quad (2\text{-}2)$$

$$IP = ID/MT \text{ (bits/sec)} \quad (2\text{-}3)$$

It has been established that Fitts' law holds for pointing tasks with a number of limbs such as the arms, wrists, fingers, head, eyes and feet. Fitts speculated that the different limbs would show different indices of performance. Subsequent research by Langolf established that the fingers, hand and arm did indeed show decreasing values for the processing rate (1/b) and therefore have different indices of performance (Langolf, Chaffin & Foulke, 1976).

The difference in performance between body parts has been observed in other research too. In studies by Gibbs (1962) and Hammerton and Tickner (1966), the performance of the hand, forearm and thumb was compared during target acquisition tasks. In both studies, it was found that the performance of the hand was superior to that of the thumb. The hypothesis in the neurophysiological studies is that fine muscle groups (the fingers for instance) allow for better performance. Based on this research and Fitts' studies, Zhai predicted a performance difference between a device used with the hand only and a device held between the fingers (Zhai, Milgram & Buxton, 1996). It was found that the device held between the fingers performed faster than the other device in a 6-DoF docking task.

In addition to the limb used, the grip[1] used to hold a device is important for the effectiveness of the device. The grip is chosen by the user but is determined in part by the device itself. For example, the shape and weight of the device can afford the device to be held with a particular grip. The grip on a device determines whether device use is better for stabilizing objects or for precise manipulation. On the highest level, the power grip and the precision grip are distinguished (MacKenzie & Iberall, 1994). With the power grip (Figure 2-8), the emphasis is on security and stability while the precision grip (Figure 2-9) emphasizes dexterity and sensitivity.

[1] Instead of grip, the term of grasp is often used. Grasp is known to relate closer to dynamical aspects of the posture of a hand. However, for simplicity, only the term grip will be used here.

The relation with the limb used is clear. With a power grip, the fingers are not actively involved in specifying the position and orientation of a device. With a precision grip however, the fingers are actively involved and smaller muscle groups

Figure 2-8. The 3DZoneMaster device is held with a power grip.

Figure 2-9. The stylus of a graphics tablet is held with a precision grip.

are used resulting in higher accuracy. An example of a device that affords a power grip is the 3DZoneMaster, as can be seen in Figure2-8. The stylus of the graphics tablet in Figure 2-9 is an example of a device that is designed for a precision grip. The consequence is that the devices are fit for different tasks. The stylus is apt for precise tasks such as drawing while the 3DZoneMaster is better used for controlling computer games.

Operational factors

In addition to the limb control and grip, another factor plays an important role in the operation of interaction devices. The physical property sensed by the device determines the way in which the user's movements are translated in application responses. Properties commonly sensed by interaction devices are presented in Figure 2-10. Some devices such as the mouse are sensitive to their position and need to be moved to cause the application to respond. Other devices such as the trackball are sensitive to their orientation and need to be rotated. The two inertial types of devices are sensitive to the force and/or torque a user applies to them.

	Linear	Angular
Position	Move	Rotate
Force	Apply Force	Apply Torque

Figure 2-10. Properties sensed by interaction devices.

Another operational factor is the mode of operation (also known as mode of sensing) of a device; a scale defined by the outer limits

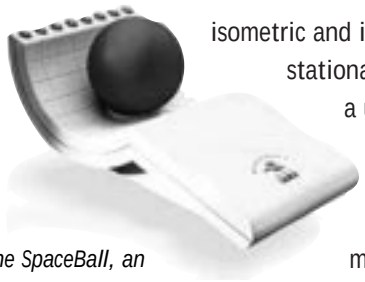


Figure 2-11. The SpaceBall, an isometric device.

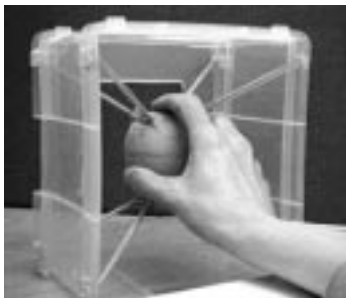


Figure 2-12. The EGG, an elastic sensing device.

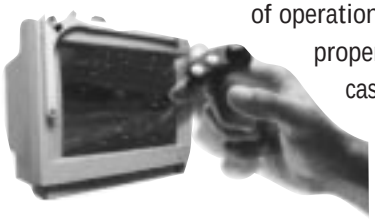


Figure 2-13. The FreeD, an isotonic device.

isometric and isotonic. In general, an isometric device is a stationary device and offers resistance to the movements a user applies to them, causing muscle fiber contractions when operated. Figure 2-11 presents the SpaceBall, an example of an isometric device. The user can apply force to the stationary ball to move and orient objects in three dimensions. On the other end of the scale, isotonic devices offer no significant resistance and these devices can easily be moved without the need for significant muscle fiber contractions. Examples of isotonic devices are the mouse and the FreeD device in Figure 2-13 (the FreeD is formerly known as Owl or RingMouse). Isotonic and isometric devices are the extremes of the scale and in the region in between the two, devices with elastic sensing can be found. Elastic sensing devices vary in the resistance they offer against the forces applied to them. Figure 2-12 presents an example of an elastic sensing device (the EGG device) that was used by Zhai in a comparison of elastic and isometric devices (Zhai, 1995). The mode of operation is strongly related to the property sensed by the device. In most cases, isotonic devices sense displacement and/or rotation while isometric devices sense force and/or torque.

A further factor divides devices in two categories too: absolute devices and relative devices. This factor is also related closely to the property sensed by the device as with the mode of operation. Absolute devices establish the property sensed with respect to a fixed origin. A graphics tablet for example, establishes the position of a stylus on the surface of the tablet. A relative device such as the mouse registers displacements only and does not measure the position of the mouse on the surface of the desk directly. The result is that the mouse can be lifted and lowered in another location (an activity known as clutching) where the mouse continues sensing displacements. The effect is that the user can continue working in another location. With the stylus of a graphics tablet, the same operation would lead to new position readings of the stylus as soon as it is lowered in proximity to the tablet surface.
Note that absolute devices can be made to behave as relative devices but that the reverse is impossible. The position readings of

the stylus of the graphics tablet for instance, can be converted to displacements in software with the result that the tablet behaves as a relative device. The displacements measured by the mouse can be converted to a position and in fact, this is what happens when the mouse is used to control a cursor. However, the lifting and lowering the mouse still does not lead to a change of the position calculated which means that the mouse is still not an absolute device. In addition, accumulating measurement errors can prevent a relative device from behaving as an absolute device.

The last common discriminating operational factor is whether the device is direct or indirect. This factor is often used to distinguish devices but does not relate to the property sensed only. Instead, it describes the relationship between the property sensed and the property controlled. In the situation of pointing on a computer screen, the property controlled is the desired position on the screen. In this situation, a device is direct if a position can be indicated on the screen directly and therefore the device needs to be able to specify positions. Examples of direct devices are the light pen and the touch screen. Indirect devices are physically located away from the screen and need an intermediary for indicating a position to the system. The cursor is a well-known intermediary and allows indirect devices such as the mouse and the trackball to be used for controlling applications.

2.2.2 Classifications of pointing devices

Given the enormous amount of devices currently available, classifications[2] have been developed to gain insight in the discriminating factors of devices and to order the design space of interaction devices. Two approaches can be distinguished that will be reviewed briefly. The first approach distinguishes devices by the types of tasks they are capable of performing. The other approach categorizes devices by operational aspects such as those discussed above. After a presentation of both approaches, the value of the classification is established by evaluating interaction devices for controlling 3D modeling tasks.

[2] In literature on pointing devices, classifications are often referred to as taxonomies. In this work, the term classification is used except when referencing established classifications.

Logical devices

The concept of logical devices was conceived to relieve interface designers and application programmers from reckoning with all the aspects of input devices. Instead of reading each input device directly, the interface designer makes the application refer to a

"logical input device" (Wallace, 1976) for input, which is mapped to an available input device with characteristics most corresponding to the input requested.

In the description of the Simple Graphics Package (SGP), Foley and van Dam (1982) identified five logical interaction devices. The ISO GKS (Graphical Kernel System) and GKS-3D standards (ISO 7942 & ISO 8805) discriminate the six logical input devices presented in Table 2-1.

Device	Purpose
Locator	Specifying coordinates
Stroke	Specifying a sequence of coordinates
Valuator	Specifying real numbers
Choice	Specifying a selection from a set of alternatives
Pick	Identifying a displayed object
String	Specifying a string of characters

Table 2-1
Logical input devicesdefined by GKS

The term logical input device already suggests that there is not necessarily a one-to-one mapping between logical devices and physical devices. Indeed, several logical devices can be operated with one physical device and one logical device can represent several physical devices. In most systems for example, the logical devices Locator, Pick and Stroke can all be operated with the mouse.
The major advantage of using the concept of logical devices is that the application developer needs not worry about the implementation details of each specific input device. At the same time, this is also the major disadvantage of the concept; it neutralizes the differences between the operational aspects of the devices and therefore assumes that devices are interchangeable as long as they can be mapped to the same logical device. Ignoring differences between physical devices can have serious impact on the user experience as can be seen when a drawing task is considered. From the logical interaction device standpoint the mouse and a stylus on a graphics tablet can both be used as "stroke" devices and used for drawing tasks. Nonetheless, to a user drawing with a stylus is a totally different experience than drawing with a mouse.

Operational characteristics

Alternatives to the logical devices approach are classifications in which the operational aspects of the interactions are taken into account (Buxton, 1983; Baecker & Buxton, 1986). The attributes

used in their taxonomy are the number of dimensions sensed, the property sensed (position, motion, pressure) and whether the device senses mechanically or by touch.

Later, a morphological design approach was used by Card to develop an extension to Buxton's taxonomy (Mackinlay, Card & Robertson, 1990; Card, Mackinlay & Robertson, 1991). Like Buxton's taxonomy, the taxonomy by Card et al. shows the property sensed by the device (position or force) and how it is sensed (absolute or relative). In addition, rotary and linear degrees of freedom are discriminated and the measurement resolution of a device is represented. A device is modeled out of different transducers that are combined with the composition operators "merge", "layout", and "connect".

Figure 2-14 presents the taxonomy proposed by Card et al. with the representation of a mouse with three buttons. The columns represent the degrees of freedom a device senses. The three left columns denote if linear degrees of freedom are sensed along the X-, Y- or Z-axes. The other columns, labeled rX, rY and rZ, represent sensed rotations about the same axes respectively. The rows in the figure represent the properties sensed by devices. Each transducer in the device is represented with a circle in the figure. The position of a circle in the cell of the taxonomy indicates the number of values a device senses. The left two circles in the representation of the mouse indicate that it has two transducers that measure

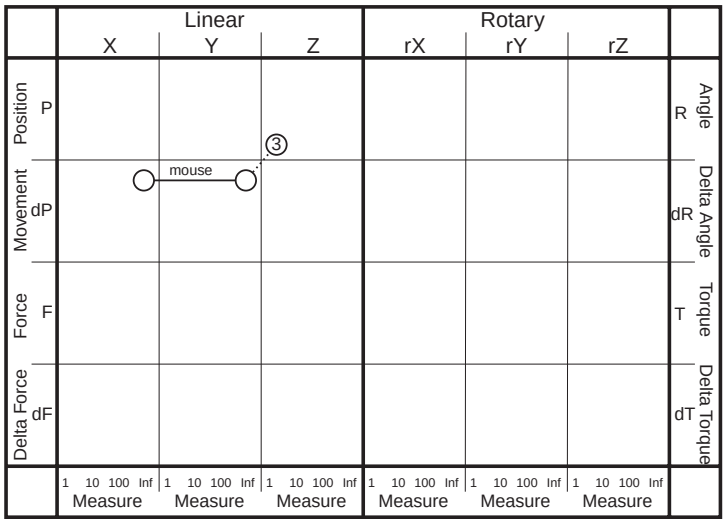


Figure 2-14. Input device taxonomy by Card et al. (1991) with a mouse represented.

movements along the X- and Y-axis respectively. The fact that the two circles are located to the right in their cell indicates that they measure displacement continuously. The right circle represents the three buttons on the mouse. The number three in the circle indicates that the three buttons are identical. The location of the circle in the cell indicates that the buttons register only one state, on or off.

The lines connecting the dots are the visual representation of the connection operators. A straight line is drawn between two transducers with a merge operator. With a merge operator, two transducers of a device represent a device of higher dimensionality. With the mouse, the transducers sensing movements along the X- and Y-axes are merged to form a device that senses movements in the plane spanned by the X- and Y-axis. A dashed line means that a layout operator connects the transducers. The layout connector indicates that two transducers are located on the same device without being merged.

Figure 2-15. Representation of a broad range of input devices in taxonomy of Card et al. (1991).

Figure 2-16. Pictures of the devices represented in the taxonomy of Figure 2-15.

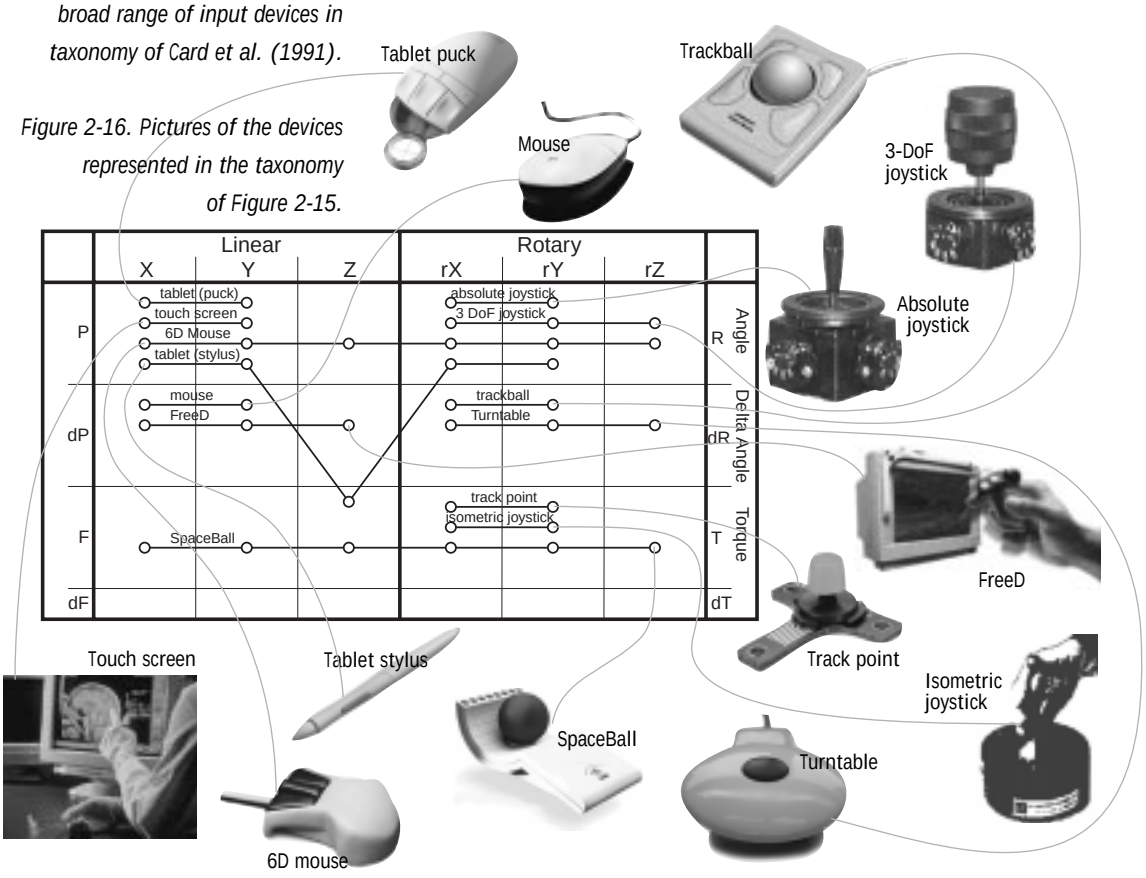


Figure 2-15 presents several similar devices and some devices with different representations (the devices are shown in Figure 2-16). For the sake of clarity, buttons have been omitted from the figure, leaving only the continuous aspects of devices. In addition, the figure only presents the merge operator and the measurement resolution is absent. The tablet puck and the touch screen both share the same representation in the taxonomy because they both sense the same two linear degrees of freedom. Although the trackball also senses two degrees of freedom, it shows up in a different area of the taxonomy because it senses rotary instead of linear degrees of freedom. The two 6-DoF devices are an example of two devices that differ in the mode of operation only. Both devices measure all six degrees of freedom but are located in different areas of the taxonomy. The reason being that the 6D Mouse is an isotonic device measuring position and orientation, while the SpaceBall is isometric and sensitive to force and torque applied to it.

The last classification presented here classifies 6-DoF devices by their modes of operation. In addition, the classification presents another discriminating factor; the mapping of properties sensed and properties controlled. This mapping is known as the control order of the device. In the situation of cursor positioning, a zero-order control device like the mouse maps the position of a device to the position of the cursor. A first-order device such as a track point controls the velocity of the cursor. Zero- and first-order control devices are also known as position and rate control devices respectively. In principle, higher order devices are also possible but not commonly used for operation of computer systems.

Figure 2-17 shows the framework that was introduced by Zhai for studying 6-DoF devices (Zhai, 1995). It presents three axes of which the mode of operation (Sensing mode) and control order (Mapping) have been discussed above. The integration mode axis represents the degree of control integration. The 6-DoF are measured either with one device (integrated) or with separate devices (separated). An example of a completely separated configuration is a set of six dials that are used to control each degree of freedom separately. Zhai subsequently used the framework for an experiment in which only integrated devices were regarded. Using a 6-DoF matching task, the performance of the following four extreme devices were compared: isotonic sensing with position control, isometric sensing

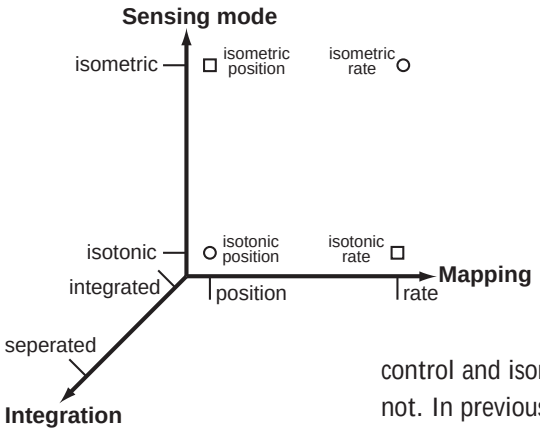


Figure 2-17. Zhai's framework for classifying 6-DoF devices by sensing mode, mapping and integration (Zhai, 1995).

with position control, isometric sensing with rate control and isotonic sensing with rate control (the circles and squares in Figure 2-17). It was found that isotonic position control and isometric rate control (the circles in the figure) lead to higher performance than the other two possibilities (the rectangles). This was in line with previous research but the absence of a performance difference between isotonic position control and isometric rate control after 20 minutes of practice was not. In previous research, position control is often favored over rate control for performance in target acquisition tasks (Poulton, 1974; Kim, Tendick, Ellis & Stark, 1987). Zhai argued that his results might have differed from those in the previous research because he used 6-DoF devices instead of the 2- or 3-DoF devices used in the earlier studies.

The value of classifications, choosing a device for 3D modeling

In general, classifications are useful for understanding the relations between interaction devices, especially those classifications that distinguish devices by their operational characteristics. Another useful aspect is that they can indicate areas where no interaction device exists yet. This means that a device can be invented that is different from previous devices. That is not to say that the invented device is unavoidably useful and usable. That depends on its ability to enable the user to control the application. Next, the value of the classifications is evaluated by a case study in which the requirements of a device for 3D modeling applications are established.

The functional requirements for an interaction device are not represented in the classifications presented above and therefore, the usefulness and usability of a device can not be established from the classifications directly. However, when functional requirements for a device have been formulated, the classifications can be used to identify which devices correspond to the requested functional requirements. Thus, the first step is to find the functional requirements. Consequently, one needs to evaluate the correspondence between device capabilities and application needs. One aspect of this correspondence is known as the expressiveness of a device (Card et al., 1991). Card's interpretation of expressiveness refers to the number of values a device senses for a given property. For instance, a mismatch between a device and an application task

would be to ask the user to select an object on the screen with a device with insufficient accuracy.

Another aspect of the correspondence between devices and applications describes the relation between the number of properties sensed and the number of properties controlled. This will be referred to as the fit between the degrees of freedom provided by the device and the degrees of freedom expected by the application. An optimal fit provides a one-to-one relation between the properties sensed and the properties controlled. In an under-controlled situation, the device senses fewer properties than need to be specified in an application. Therefore, the user needs to specify sets of application properties sequentially with the same device. In an over-controlled situation, all application properties can be set at once but the device senses more properties than needed. Both situations are not optimal because they either force the user to divide intended actions in subsequent movements or confuse the user because the outcome of device actions is ambiguous.

Next, the needs for a conceptual modeling application are considered and the concept of fit is used to identify the class of devices optimally suited for the conceptual modeling. Regarding computer supported conceptual modeling; the designer needs to be able to manipulate a three-dimensional design. It was stated above that this would be supported by enabling the designer to position and orient objects or tools directly in 3D in uninterrupted movements. A device with an optimal fit would therefore allow the designer to specify a position and orientation in 3D that is mapped directly to the controlled object or tool in the application. Such a device thus needs to sense 6-DoF at once. Card's taxonomy in Figure 2-15 presents two examples of devices that measure the 6-DoF required for specifying a position and an orientation, the SpaceBall and the 6D Mouse.

Thus far, the concept of fit helped to find the desired degrees of freedom for a pointing device for conceptual modeling. However, Figure 2-15 indicates four possible types of 6-DoF devices, not counting mixed mode devices such as the stylus of a graphics tablet that senses position and force. One of those is a hypothetical device because there are no relative devices sensing force and torque so

that three types of devices are left. These devices sense movement, position or force. To choose between those devices, Zhai's framework (Figure 2-17) can be used because the device needs to be used to specify the position and orientation of objects and tools in 3D. Regretfully, the evaluations of isotonic or isometric control in literature are ambiguous and a final decision for isotonic or isometric control can not be made. However, instead of choosing, combining the results of the work of Zhai et al. and Poulton suggests that a 6-DoF isotonic device with position control can be used for modeling tasks.

Omissions in the classifications

The case study above shows that classifications can be useful in deciding what type of interaction devices is to be considered for a given application. However, classifications can not be used for that purpose directly. Only when the functional requirements of a device are known, classifications can be searched for appropriate devices. In addition, the fit between the needs of an application and the characteristics of the devices is not present. Therefore, the choice for an interaction device still relies on experience and intuition of the interface designer, especially for applications that go beyond the WIMP metaphor.

The classifications above focus on the technological aspects of the devices and not on the user operating them or on the task that they control. One of the most apparent omissions of the classifications is that they ignore the ergonomic aspects of interaction devices. It was already stated that the limb used to hold a device and the grip with which the device is held is important for precision and comfort. Therefore, the shape and weight of the device play an important role in the effectiveness of the device because these factors afford the grip on the device. Form factors will be encountered again in the next chapter when the design of the Frogs interaction devices is discussed. There, the shape of 6-DoF devices and the relationship to their use are addressed.

2.2.3 Effectiveness of pointing devices

To determine the effectiveness of pointing devices, a number of factors have been identified in previous research. Table 2-2 presents a number of factors that have been used to establish the usability of devices. In the majority of studies, only the first two factors, pointing time and accuracy have been considered. It has been

established that Fitts' law (Equation 2-1) holds for pointing devices in rapid pointing tasks. In this context, Fitts' law establishes a relation between the pointing time and the accuracy of the device. Therefore, the index of performance (Equation 2-3) can be established for a pointing device and can be used to compare devices. In fact, this has been the subject of many comparative studies, mostly on 2-DoF devices. However, the results between studies have not been consistent since the conditions in the studies were different.

Factor	Description
Pointing time	The time needed to acquire objects on the screen
Accuracy	The accuracy with which objects can be identified on the screen
Footprint	The amount of desk-space occupied when using the device
Acquisition time	The time needed to grasp the device
Cost	The amount of money paid for the device
User preference	The usability of the device as indicated by the user

Table 2-2
Factors that determine the effectiveness of a pointing device

The footprint of a device is the amount of space needed to operate the device. Using a trackball for example requires less desk-space than operating a mouse. With 6-DoF devices, an isotonic device like the 6D Mouse takes more space to operate than an isometric desktop device like the SpaceBall. The acquisition time is the time needed before a user can start operating the device. When the hands of a user are on the keyboard for example, time is needed before the mouse can be used to select an item from a menu. The significance of this factor depends on the application. Some applications require the user to switch between the keyboard and the mouse a lot, while others are operated with the mouse only and then acquisition time is much less influential.

The user preference factor is really a grab-bag of factors that determine the perceived effectiveness by users of the device. The other factors can be determined objectively but, the usability of the device is determined from indications by users. The perceived effectiveness is mostly established in comparative studies where users are asked to rank different devices. The ranking often depends on the target user group and the amount of fatigue, discomfort and satisfaction they experienced while using the device.

2.3 Applying pointing devices to operate applications

In the previous discussion on pointing devices, the relationship between the pointing device and the application has only been minimally addressed. Just the properties of the application controlled by interaction devices have been discussed. In this section, the void is addressed by presenting two approaches for applying interaction devices in applications. The difference in the approaches is comparable to the differences between the classifications of interaction devices. First, an example of a bottom-up approach is presented. Here, the focus is on the use of the devices, as with the classifications that consider the operational aspects of devices. Second, in a top-down approach, application tasks are categorized by searching for generic interactions first and then the connection between devices and interaction techniques are considered. This approach resembles the concept of logical interaction devices.

2.3.1 Modeling pointing device use

The first abstraction of device use in applications is the three-state model that closely relates to the use of the device itself (Buxton, 1990). The model relates the state of an application to the state of the device. Figure 2-18 for example, presents the state transition diagram for the use of a mouse. Pressing or releasing the button of the mouse transfers the state of the system between states one and two. If the button of the mouse is up, the system is in tracking state until the mouse button is pressed. Then, the dragging state is entered. Most operating systems notify applications about the state of a device by passing them events. Applications can process the events by updating their internal state model and can activate specific functionality for different state transitions. In the case of moving an icon on the desktop for instance, the button down event can be used to find which icon from the desktop is to be moved. Dragging events can be used to provide the user feedback about the current location of the icon. The button up event can be used to finalize the drag operation by parking the icon at the new location. The three-state model of the mouse contains only two states. Figure 2-19 presents the three-state model of the stylus used in combination with a graphics tablet. This model has one additional state that is called out of range. This state represents the fact that with a graphics tablet, the stylus can be lifted off the tablet making position readings impossible. While it is also possible to lift the

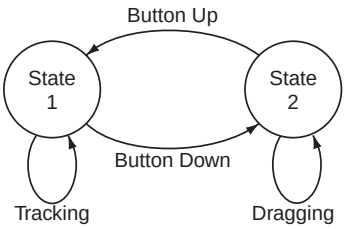


Figure 2-18. The three-state model for the mouse.

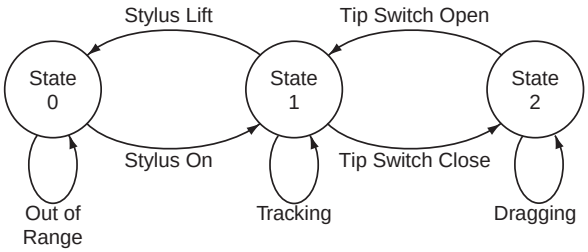


Figure 2-19. The three-state model for the stylus used with a graphics tablet.

mouse off the table, the system has no means of detecting this. In contrast, the graphics tablet is aware whether the stylus is out of range and therefore the extra state is represented in the figure.[3]

With the aid of the three-state model, two types of application tasks are distinguished; the point/select task and the dragging task. For devices with a button such as the mouse and the stylus, these tasks can be represented in terms of the movements in states 1 and 2 and changes between those states. Pointing is undertaken by moving the device in state 1 and selecting corresponds to transition from state 1 to state 2 and back to state 1. Dragging is an activity undertaken by moving the device in state 2 and requires a user to move the device while holding a button down.

Note that the two examples presented are 2-DoF devices and in fact, references to this model often consider 2-DoF devices only. Nevertheless, the three-state model is applicable for devices with higher degrees of freedom also. The three-state model of the 6D mouse for instance, is identical to that of the stylus if only one button is considered.

[3] Note that there is no indication of what happens when the stylus is moved out of range when in state 2 or when the stylus comes into range with the tip switch closed (transitions $0 \rightarrow 2$ and $2 \rightarrow 0$). Different behaviors have been found with different tablets, but the extra transitions have been omitted from the figure since no extra states are introduced.

2.3.2 Primitive interactions

Another approach to modeling the use of interaction devices in applications is to start from the viewpoint of the application. This requires an identification of elementary interactions forming the building blocks for the dialogue between the user and the computer graphics system. Foley, Wallace and Chan (1984) took this approach and identified six primitive interactions[4] that closely relate to the concept of logical devices discussed above. From the user's point of view, primitive interactions are the vocabulary of the conversation with the system. Based on observations of computer graphics systems, they identified the six interactions in Table 2-3 that are almost identical to the logical devices in Table 2-1.

[4] Foley et al. use the term primitive interaction task (1984). However, since the term task is used in this work for something a user wants to accomplish with a computer system, the term primitive interaction is used here.

Task	Description
Select	The user makes a selection from a set of alternatives
Position	The user specifies a position on the screen
Orient	The user specifies an orientation in 2D or 3D space
Path	The user generates a series of positions or orientations over time
Quantify	The user specifies a value to quantify a measure.
Text	The user inputs a text string

Table 2-3 Primitive interactions proposed by Foley et al. (1984)

None of the primitive interactions allows the user to manipulate objects directly. Four additional controlling interactions[5] are identified by Foley et al. for manipulating objects instead of specifying a choice, a position, an orientation or other data types such as numbers and text as the primitive interactions do. The controlling interactions are given in Table 2-4.

Task	Description
Stretch	The user grabs a feature of an object and moves it, leaving the other features in place
Sketch	The user creates an object using an input device like a brush or a pen
Manipulate	The user alters the position, orientation or scale of an object
Shape	The user alters the shape of a smooth curved line or surface

Table 2-4 Controlling interactions proposed by Foley et al. (1984)

For each of the primitive and controlling interactions, a comprehensive set of interaction techniques is presented. In addition, several interaction devices are considered for each interaction technique and each combination is evaluated using several ergonomic measures. The ergonomic measures include factors such as cognitive, perceptual and motor load that are established subjectively.

The concept is useful for getting a picture of the building blocks of a user interface for computer graphics applications. Further, the combined evaluation of interaction technique and interaction device is insightful. However, in the evaluation of interaction devices, only keyboards and pointing devices with at most 2-DoF are considered. Furthermore, Foley et al. took the liberty of ignoring the physical design of the interaction devices. Instead, they assumed that the devices they selected were optimally designed for their intended use, although they admit that the design of the interaction device can seriously influence the ergonomic quality of the interface.

[5] Foley et al. use the term controlling task (1984).

Regretfully, the evaluations are subjective and both the sets of interaction techniques and interaction devices are not complete. The problem of the concept is that it can not be used for conceptualizing and evaluating novel interaction techniques. Instead, it represents the state of interfaces for computer graphics at the time of writing. For instance, there is no reference to 6-DoF devices and interaction techniques for these devices can be very different from those used in combination with 2-DoF devices. In the light of the use of 6-DoF devices, the question raises whether the six primitive tasks should be used unaltered. With 2-DoF devices, the separation of position and orient tasks is logical because the position and orientation of objects can never be controlled simultaneously. With 6-DoF devices however, position and orientation can be specified at once and the original separate position and orientation tasks can be regarded as special cases of the combined task of specifying a stance.

2.4 Two-handed operation

In the previous chapter, two-handed operation was introduced as a potential means to improve the interface between the designer and the CAD system. However, when creating bimanual interfaces, one should be aware that people do not use their hands in the same way. This has implications for applications where interaction devices are used with both hands but not at the same time. In addition, when people work with two hands, the roles the hands play in the combined activity is different. This is important for applications that need to support two interaction devices operated with two hands at the same time.

Results of previous research are presented regarding the difference between the hands and concurrent two-handed activity. Then, examples of two-handed systems illustrate the potential of two-handed operation for computer systems. Finally, the benefits of two-handed operation are summarized.

2.4.1 Difference between the hands

People have a choice when they need to perform a task that involves only one hand. Usually, the same hand is chosen and this hand is referred to as the dominant hand or the preferred hand. For the majority of the population, the dominant hand is the right hand, although it is known that for creative people the percentage

of left-handers is higher than average. To determine whether someone is right- or left-handed is somewhat ambiguous because some people use both their left and right hand for single-handed tasks. For instance, some people hold a pen in their left hand for writing and use a mouse with their right hand. Therefore, the preferred hand is often established with the aid of handedness tests in which the subjects are questioned to determine their dominant hand in a number of single-handed activities (Oldfield, 1971).

In previous research, it has been established that there is a difference between the capabilities of the hands. For instance, for rapid aimed movements such as used in the Fitts' law (Equation 2-1) experiments, it has been shown that people often produce shorter movement times and higher accuracy with their dominant hand than with their non-dominant hand. Kabbash et al. list three theories that attempt to explain this difference (Kabbash, MacKenzie & Buxton, 1993). The first theory is that the transfer function between intended movements and actual movements of the non-dominant hand responds more "noisily" than that of the dominant hand. Therefore, the movements of the non-dominant hand show larger variability resulting in more errors and longer movement times. The second theory is that people differ in the way they use sensory feedback control for directing movements of each hand. The third theory of Todor and Doane expands on the second theory (Todor & Doane, 1971). It is based on psychomotor models of movement and models for the control of the hands by different hemispheres of the brain.

Kabbash et al. are interested in this theory because it predicts that in a Fitts task, the left and right hand show different performance patterns when the size of the target is varied. Specifically, it predicts that when the target size increases while keeping the index of difficulty constant, movement times with the non-dominant hand decrease while those with the dominant hand increase. In an experiment, they used three pointing devices to test movement times and errors of both hands. They found that for large target areas, the results for the non-dominant hand were the same as those of the dominant hand. For the smaller targets however, the use of the dominant hand leads to shorter movement times. They concluded that for rough pointing or motion, using the non-dominant hand is as appropriate as using the dominant hand.

2.4.2 Working with two hands together

The previous section focussed on single-handed activity with either the dominant or the non-dominant hand. Guiard (1987) has pointed out that the concept of a single-handed task is sometimes unclear. In most of the handedness tests, a lot of activities are considered that appear to be single-handed but are in fact two-handed (at least to some degree). The activity of writing, for instance, appears to be restricted to the dominant hand that holds the pen. However, it has been shown that when a person is not able to position the paper with the other hand, performance degrades. Therefore, a lot of everyday activity is bimanual even if it seems that only one hand is involved. When working with computers however, the use of the non-dominant hand is very limited[6]. Often, the non-dominant hand is used for nothing more than pressing keys on the keyboard to change the response or mode of an application to activity of the dominant hand. In developing computer interfaces that do support two-handed activity, it is advisable to reflect the way people use their hands or else we may end up with interfaces that are difficult to operate. To this purpose, Guiard's Kinematic Chain (KC) model is introduced that describes the role of the hands in concurrent two-handed activity (Guiard, 1987). Different types of two-handed interaction in computer interfaces are then evaluated for adherence to the everyday use of the two hands.

Although the KC model is not the only model for two-handed operation, it has been attracting the interest of computer interface researchers. With this model, researchers have been able to understand why some two-handed interfaces have been successful while others were disappointing. The model establishes that the non-dominant hand is not just a poor approximation of the dominant hand. Instead, it plays its own important role in two-handed tasks. Regarding the role of the hands in cooperative two-handed activities, Guiard presents the following three high-order principles (considering right-handedness only):

[6] Typing on a keyboard is an example of two-handed activity commonly supported in computer interfaces. However, two-handed use of pointing devices is virtually non-existing in commercial applications.

Right-to-left reference The motion of the right hand typically finds its spatial references in the results of motion of the left hand. Thus, the left hand sets the stage for the right hand. In the case of sewing, the left hand not only stabilizes the cloth but also positions the cloth such that the right hand can easily do the sewing.

Asymmetric scales The left and the right hand are involved in different motions. The left and right hand's motions are labeled macrometric and micrometric respectively. When compared to the motions of the left hand, the right hand's motions tend to be of higher precision with higher temporal frequency and smaller spatial amplitude. With sewing, the movements of the cloth in the left hand are slower and larger than the fast and precise moments of the needle in the right hand.

Left-hand precedence The contribution of the left hand to the combined activity starts earlier than that of the right hand. In sewing for instance, it makes no sense to start sewing with the right hand when the cloth has not been positioned with the left hand.

The KC model captures the way in which people work with two hands in everyday life by regarding the hands as abstract motors. To model the cooperative activity of the hands in bimanual tasks, the KC model distinguishes three types of assemblages of the motors. In an orthogonal assemblage, the two hands are involved in independent activities. In a parallel assemblage, both hands are involved in the same task and the contributions of both hands are symmetrical. The largest class of bimanual activities is characterized by a serial assembly in which both hands are involved in the same task too. The difference with a parallel assemblage is that in a serial assemblage, the contributions of the hands are dependent. Here, the input of the motor of the right hand is dependent on the output of the motor of the left hand. To establish how to apply two-handed operation in computer interfaces, first three possibilities for adding two-handed support are presented. Then, the value of these possibilities is evaluated using the KC model.

In a discussion of supporting two-handed activity in a graphical toolkit, Chatty distinguishes three types of two-handed operation with increasing degrees of cooperative activity (Chatty, 1994). The first type is independent two-handed operation in which the user has a device for each hand but can use the devices only sequentially. In parallel two-handed operation, the user can use both hands to control two tasks concurrently but only separate tasks. The last type two-handed operation is combined activity in which the user can use both hands in a combined task.

A closer look at independent two-handed operation reveals that it does not support concurrent two-handed activity. The only difference with single-handed operation is that two devices can be used instead of one device. However, it serves a purpose in the sense that independent operation is needed as a basis for the other two types of two-handed operation. With parallel two-handed operation, the user of a computer system can use two devices at the same time. With this kind of interaction, two types of human bimanual movement of the KC model can be realized; bimanual tasks with orthogonal and parallel assemblages. Both orthogonal and parallel assemblies constitute only a small part of the bimanual moves of people. Only when combined two-handed operation is offered in a computer interface, the largest class of human bimanual movements can be supported. Those are movements with a serial assemblage.

2.4.3 Examples of two-handed operation in computer interfaces

In an early study, Buxton and Myers established the appropriateness of two-handed interaction for computer interfaces by presenting the results of two experiments (Buxton & Myers, 1986). In the first experiment, subjects were asked to position and scale a graphical object. The scale of the object was set with a slider in one hand while the position of the object was controlled with a puck held in the other hand. The results of this experiment showed that the subjects were perfectly able to control both the scale and position of the object at the same time. In the second experiment, one- and two-handed operation was compared in a combined selection and navigation task. Subjects that were able to navigate with one hand while selecting with the other were faster than the subjects that had to navigate and select with one hand sequentially.

The Toolglass interface technique developed by Bier et al. is another example of a two-handed interface for 2D graphical applications (Bier, Stone, Pier, Buxton & DeRose, 1993). It features a semi-transparent menu (the Toolglass) that is moved with the non-dominant hand. The Toolglass is moved over the work area of an application and it can give the user an alternative view of the work like the local scaling lens in Figure 2-20. In addition, it allows the user to manipulate objects. Manipulation with the Toolglass is accomplished by moving it over an object and clicking on it with the dominant hand. For example, coloring an object encompasses

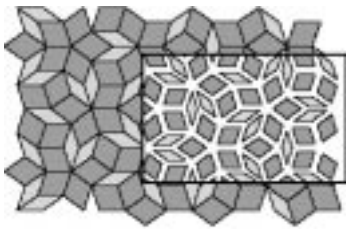
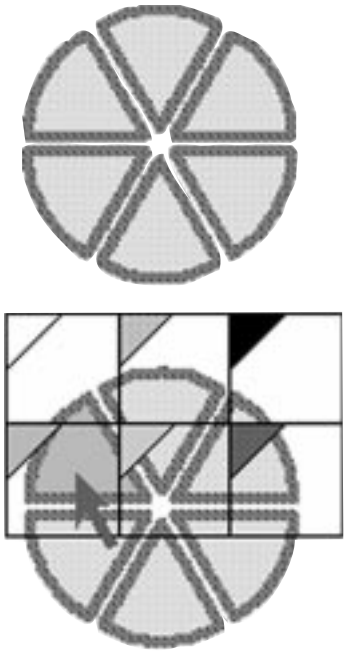


Figure 2-20. The local scaling lens that shrinks each object under the lens around its center.

Figure 2-21. Changing the color of the contents of a wedge with the Toolglass. The wedges shown left have not been colored. One of the wedges shown right is being filled by bringing the desired color chip of the rectangular Toolglass over the wedge with the non-dominant hand. The arrow-shaped cursor is then brought over the color chip with the dominant hand and clicking changes the color of the wedge.

bringing the desired color chip on the Toolglass over the object and clicking on the color chip (Figure 2-21). The Toolglass interface has been used in a number of studies on two-handed interaction.



In one of those subsequent studies, the Toolglass technique was used in a comparison of four interfaces for the same line drawing task (Kabbash, Buxton & Sellen, 1994). Kabbash et al. compared one single-handed interface and three two-handed interfaces to identify what type of two-handed interfaces lead to the best performance. From the three two-handed interfaces, one was regarded not to reflect the everyday use of the two hands (the serial assembly in the KC model). The other two-handed interfaces did reflect the serial assembly; these were called the asymmetric dependent type of interfaces because left and right hand were involved in different activities with the dominant hand depending on actions of the non-dominant hand. They concluded that two-handed operation could be superior to one-handed operation if the division of tasks between the hands is of the asymmetric dependent type. In an ill-designed interface with independent task assignment, two hands could even be worse than one hand.

Two-handed interaction has not only been applied to 2D graphical applications but to 3D systems as well. An example of such a system is the SKETCH system that is operated with 2-DoF interaction devices (Zeleznik, Forsberg & Strauss, 1997). The system allows users to perform a number of CAD operations with two hands. Objects can be moved, rotated and scaled, the viewpoint and other camera parameters can be manipulated and several editing operations are supported. For most of the operations, the two devices control the position of two cursors that together provide for 4-DoF instead of the 2-DoF that are provided with one device only. Figure 2-22 shows

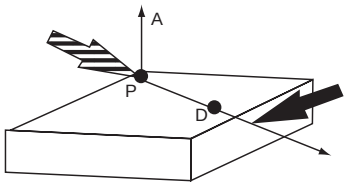


Figure 2-22. Rotating an object with two cursors in SKETCH.

how objects can be rotated with two hands in SKETCH. The striped cursor operated with the non-dominant hand defines the center of rotation (point P in the figure). The object is moved when the non-dominant hand cursor is moved. The black cursor is controlled with the dominant hand and is used to specify the rotation angle about the axis of rotation through point P, parallel to vector A.

Finally, some systems combine two-handed operation with 3D interaction devices. Examples of those systems have been presented in the previous chapter. These systems have demonstrated that people can transfer the skill from everyday use of the two hands to 3D environments with little effort (Shaw & Green, 1994; Hinckley, Pausch & Proffitt, 1997). For example, Sachs et al. reported that users of the 3-Draw system could benefit from their ability to know where their hands are relative to each other (Sachs, Roberts & Stoops, 1991). This was reinforced by the observations of Hinckley et al. that subjects could use their two-handed neurological props system immediately (Hinckley, Pausch, Goble & Kassell, 1994a).

2.4.4 Benefits of two-handed operation

Summarizing the research on computer interfaces, the potential benefits of two-handed operation are recapitulated. It should be noted that all the benefits appear only for interfaces that take the characteristics of the hands in bimanual activities into account. The danger in designing two-handed interfaces is that the results can impose such a high cognitive load on users that they are too demanding to operate. This can happen when user needs to operate an interface where left and right hand tasks are independent. However, the results of previous research suggest that it is possible to create two-handed interfaces that are intuitive to use and therefore produce good results. The common theme in these successful interfaces is that they support the common everyday use of the two hands, as observed by Guiard.

Reducing task completion time

Compared to one-handed operation, two-handed operation can lead to faster task completion because of two reasons. The first apparent reason for higher performance is that the use of the second hand allows a user to control more functionality at the same time. For example whilst, in a one-handed interface, position and size of an object have to be specified sequentially, a two-handed interface could allow for the concurrent specification of both attributes.

The second somewhat less obvious reason for increased speed is that each hand can remain in its individual "home position" (Kabbash, Buxton & Sellen, 1994). In a two-handed 2D interface for instance, the left hand can be used to manipulate items on the left side of the screen, while the right hand covers the right side. If the same interface is to be operated with one hand, the user needs to move between both sides of the screen.

Reducing the workload per hand

By separating a task over two hands, the workload can be split over the hands. This relates to the second reason for reduced task completion time. Consider for example the task of assembling two objects. With one-handed interaction, one object needs to be brought to the other object, while with two-handed operation each hand could move one object and both objects could meet at in the middle between the original positions of the objects. This strategy results in each hand moving only half of the distance.

Employing the kinesthetic sense

When working with two hands, people can sense where one hand is relative to the other (kinesthetic sense). It has been shown that the kinesthetic sense in two-handed interfaces can closely correspond to the visual feedback of a computer system (Balakrishan & Hinckley, 1999). In an experiment, the performance of four interfaces was compared using a colored line drawing task with the Toolglass described above. By varying the levels of visual feedback and the strength of the coupling between the hands, it was established that in reduced visual feedback conditions, the kinesthetic sense becomes a significant factor.

Understanding 3D spatial relationships

In the context of 3D interaction, users can experience an increased spatial awareness by employing their kinesthetic sense. This is especially useful when 3D interaction is combined with a 2D display that provides limited feedback about the depth of the objects presented. Consider again the placement of one object upon another object. If each object is moved with one hand, the user can easily deduce the relative depth of the objects by moving the objects although the 2D display provides little depth cues. Hinckley et al. have shown that the relationship between the hands is strong (Hinckley, Pausch & Proffitt, 1997). In a memory test, subjects were asked to remember the position of their dominant hand without visual feedback. It was found that when people were able to position their dominant hand relative to the other, results were

more accurate than when they had no support from their non-dominant hand.

Corresponding to natural skills

Two-handed interaction does fit the way people use their hands in everyday tasks. Therefore, operating two-handed computer interfaces is potentially less demanding than one-handed interfaces. One reason is that with two-handed interaction, users do not have to split a task up into unnaturally small chunks. When manipulating a 3D model in a one-handed situation for example, users need to switch constantly between orienting the model and doing the actual manipulating. With two-handed interfaces however, both actions can be integrated seamlessly. For modeling, this could mean that working with the computer resembles working with the traditional tools.

Increasing the expressiveness of interfaces

The above-mentioned benefit of two-handed interaction is one not easily established numerically like task completion time or accuracy. Two-handed interaction has added value over one-handed interaction besides increasing speed, like the difference between working with 2-DoF interaction devices and working directly in 3D. For example, the user can develop more task solution strategies with the greater degree of control that two-handed operation provides. It has been reported that when bimanual operation has cognitive as well as manual benefits (Hinckley, Pausch & Proffitt, 1997; Leganchuk, Zhai & Buxton, 1998). This suggests that when working with a system with two hands, people are able to find more efficient task solution strategies. For a designer this could mean that the design space is better explored.

2.5 Conclusion

This chapter has established the potential of 3D interaction devices and two-handed interaction for conceptual modeling. It was found that current CAD tools predominantly use 2-DoF interaction devices. These interaction devices and their associated interaction techniques pose a barrier between the designer and the model being created. Based on classifications of interaction devices, a lower barrier was predicted when the designer was provided with 6-DoF interaction devices that allow working in 3D directly. Since 6-DoF interaction devices are used in experimental systems only, most evaluative research on interaction devices concentrates on 2-DoF

devices used with omnipresent WIMP applications. In addition, devices are often tested out of the context of a real task, even for 2-DoF devices. Often, only pointing speed and accuracy are measured. It was concluded that the effectiveness of a device is determined to a large extend by the task it will be used for.

Besides the survey of devices, two-handed interaction was evaluated as a means to find a better correspondence between the skills of a designer and the functionality of a CAD tool. It was found that two-handed interaction does have the potential to provide the better fit but creating support for two hands is not straightforward.
In designing a two-handed interface, the interaction designer should be aware of the role of both hands in combined two-handed activity. The results of several researches have been combined to find that the Kinematic Chain model helps to understand what is involved in designing a successful two-handed interface.

To add 3D interaction with two hands to CAD tools, new interaction techniques are needed because the current CAD tools are not easily modified to include it. The next chapter presents the development of those interaction techniques and describes the design of a 6-DoF device that was used to evaluate these techniques. In the description of the device, a summary will be made of the form aspects of 6-DoF devices that are often ignored in literature on interaction devices. In the later chapters, the value of the interaction devices and techniques is evaluated in experiments. The evaluations have in common that a task is used that resembles a common design task.