Chapter 1:

AN INTRODUCTION TO HUMAN INPUT TO COMPUTERS

The human body has 200 joints. 56 of them are in the hands!
[Add Reference]

Introduction

In recent years there has been much talk about the "look and feel" of user interfaces. I often speculate that if typography had to reflect effort invested in input compared to output, this would be more properly written:

Look Feel

"Hands on" computing is a myth, and is more accurately stated as "finger on" (note the singular) except when typing, and in virtually no case do personal computers recognize that our fingers are capable of anything but binary on/off action.

A professional violinist will spend more on a *bow* than computing professionals spend on a workstation. This is simply in order to be able to capture the full subtlety of gestural nuance that they are capable of producing. Meanwhile I still can't buy a personal computer that lets me do what a 15¢ pencil can, namely draw a line whose thickness or strength varies depending on how hard I push, or what angle I hold it.

This is all by way of saying that input to computer systems is still in its infancy. It is a neglected area that is central to improving the quality of human computer interaction. This book is an attempt to help bring about a change in this situation, and stimulate a new wave of activity in research, design and practice.

The type of interaction that we deal with is known as *haptic*. The term comes from a Greek word having to do with contact.

Haptic interaction with computers has primarily to do with input. While we have physical contact with transducers such as mice, their design does not afford us to feel the edges of things that we are pointing at, for example. While rare, there are, however, some haptic output devices. One example would be a device capable of producing output in braille. Another class of device that delivers haptic output is what is known as *force feedback*. These are devices which can both sense users action and which can be controlled by the computer. At the CERN Accelerator in Geneva, for example, a knob has been implemented whose feel can be changed by the computer (Beck and Stumpe, 1973). A number of other force feedback devices are described, albeit briefly, by Minsky, Brooks, Behensky, Milliken, Russo and Druin (1989). Three examples which we will discuss more fully in a later chapter are Cadoz, Luciani and Florens (1984), Iwata (1990) and Brooks, Ouh-Young, Batter & Kilpatrick (1990).

While not necessarily controlled by a computer, *every* haptic input device can be considered to provide some output by way of the tactile or kinesthetic feedback that it provides. In practice, the quality and appropriateness of this "feel" is often extremely important in determining a device's effectiveness and acceptance in a particular context.

The bulk of the discussion will concern manual input, since that is the most common usage, and the richest source from which we can draw examples. However, it is important to remember that the concepts that arise in the use of the hands generally apply to control through other parts of the body. For example, foot pedals provide an alternative approach (Pearson and Weiser, 1988; Sellen, Kurtenbach and Buxton, 1990). Taking into account users with physical disabilities, tongue-activated joysticks, provide another important example.

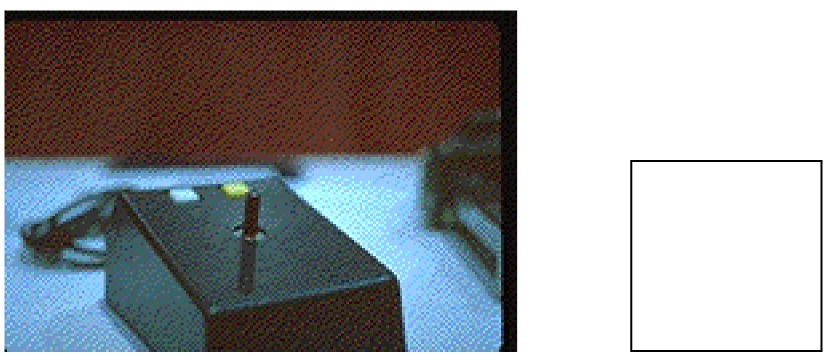


Figure 1: Two Isometric Joysticks

The Choice of Technology Makes a Difference

Each input device has its own strengths and weaknesses, just as each application has its own unique demands. With the wide range of input devices available, one of the problems that confront the designer is to obtain a match among application, input technology and user. Part of the problem has to do with recognizing the relevant dimensions along which the application's demands should be characterized. Another is knowing how each technology being considered performs along those dimensions. Finally, a key consideration is the end user. These are topics addressed below and in Buxton (1986a).

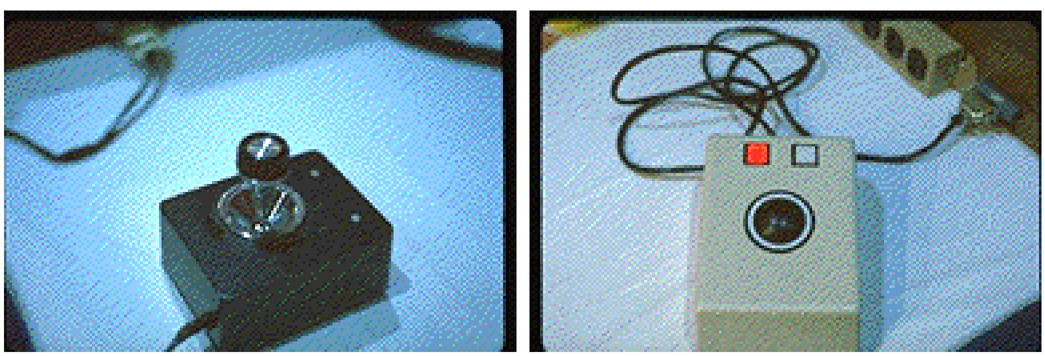
Example 1: The Isometric Joystick

An *isometric joystick* is a joystick whose handle does not move when it is pushed. Rather, its shaft senses how hard you are pushing it, and in what direction. It is, therefore, a pressure-sensitive device. Two isometric joysticks are shown in Figure 1. They are both made by the same manufacturer. They cost about the same, and are electronically identical. In fact, they are plug compatible. How they differ is in their size, the muscle groups that they consequently employ, and the amount of force required to get a given output.

> *Remember, people generally discuss joysticks vs mice or trackballs. Here we are not only comparing joysticks against joysticks, we are comparing one isometric joystick to another. When should one be used rather than the other? The answer obviously depends on the context. What can be said is that their differences may often be more significant than their similarities. In the absence of one of the pair, it may be better to utilize a completely different type of transducer (such as a mouse) than to use the other isometric joystick.*

Example 2: Joystick vs. Trackball

Let's take an example in which subtle idiosyncratic differences have a strong effect on the appropriateness of the device for a particular transaction. In this example we will look at two different devices. One is the joystick shown in Figure 2(a).



(a) (b)

Figure 2: A 3-D Joystick (a) and a 3-D Trackball (b).

In many ways, it is very similar to the isometric joysticks seen in the previous example. It is made by the same manufacturer, and it is plug-compatible with respect to the X/Y values that it transmits. However, this new joystick moves when it is pushed, and (as a result of spring action) returns to the center position when released. In addition, it has a third dimension of control accessible by manipulating the self-returning spring-loaded rotary pot mounted on the top of the shaft.

Rather than contrasting this to the joysticks of the previous example (which would, in fact, be a useful exercise), let us compare it to the 3-D trackball shown in Figure 2(b). (A 3-D trackball is a trackball constructed so as to enable us to sense clockwise and counter-clockwise "twisting" of the ball as well as the amount that it has been "rolled" in the horizontal and vertical directions.)

This trackball is plug compatible with the 3-D joystick, costs about the same, has the same "footprint" (consumes the same amount of desk space), and utilizes the same major muscle groups. It has a great deal in common with the 3-D joystick of Figure 2(a).

In many ways the trackball has more in common with the joystick in Figure 2(a) than do the joysticks shown in Figure 1!

If you are starting to wonder about the appropriateness of always characterizing input devices by names such as "joystick" or "mouse", then the point of this section is getting across. It is starting to seem that we should lump devices together according to some "dimension of maximum significance", rather than by some (perhaps irrelevant) similarity in their mechanical construction (such as being a mouse or joystick). The prime issue arising from this recognition is the problem of determining which dimension is of maximum significance in a given context. Another is the weakness of our current vocabulary to express such dimensions. Despite their similarities, these two devices differ in a very subtle, but significant, way. Namely, it is much easier to simultaneously control all three dimensions when using the joystick than when using the trackball. In some applications this will make no difference. But for the moment, we care about instances where it does. We will look at two scenarios.

Scenario 1: CAD

We are working on a graphics program for doing VLSI layout. The chip on which we are working is quite complex. The only way that the entire mask can be viewed at one time is at a very small scale. To examine a specific area in detail, therefore, we must "pan" over it, and "zoom in". With the joystick, we can pan over the surface of the circuit by adjusting the stick position. Panning direction is determined by the direction in which the spring-loaded stick is off-center, and speed is determined by its distance off-center. Zooming is controlled by twisting the shaft of the joystick.

Which way should one twist to zoom in and which way to zoom out? Why?

With the trackball, we exercise control by rolling the ball in the direction and at the speed that we want to pan. Panning is easier with trackball than the spring-loaded joystick. This is because of the strong correlation (or compatibility) between stimulus (direction, speed and amount of roll) and response (direction, speed and amount of panning) in this example. With the spring-loaded joystick, there was a position-to-motion mapping rather than the motion-to-motion mapping seen with the trackball. Such cross-modality mappings require learning and impede achieving optimal human performance. However, if our application demands that we be able to zoom and pan simultaneously, then we have to reconsider our evaluation. With the joystick, it is easy to zoom in and out of regions of interest while panning. One need only twist the shaft-mounted pot while moving the stick. However, with the trackball, it is nearly impossible to twist the ball at the same time that it is being rolled. The 3D trackball is, in fact, better described as a 2+ device.

Scenario 2: Process Control

We are using the computer to control an oil refinery. The pipes and valves of a complex part of the system are shown graphically on the CRT, along with critical status information. My job is to monitor the status information and when conditions dictate, modify the system by adjusting the settings of specific valves. I do this by means of *direct manipulation*. That is, valves are adjusted by adjusting their graphical representation on the screen. Using the joystick, this is accomplished by pointing at the desired valve, then twisting the pot mounted on the stick. However, it is difficult to twist the joystick-pot without also causing some change in the X and Y values. This causes problems, since graphics pots may be in close proximity on the display. Using the trackball, however, the problem does not occur. In order to twist the trackball, it can be (and is best) gripped so that the finger tips rest against the bezel of the housing. The finger tips thus prevent any rolling of the ball. Hence, twisting is orthogonal to motion in X and Y. The trackball is the better transducer in this example *precisely because of its idiosyncratic 2+1 property*.

Thus, we have seen how the very properties that gave the joystick the advantage in the first scenario were a liability in the second. Conversely, with the trackball, we have seen how the liability became an advantage. What is to be learned here is that if such cases exist between these two devices, then it is most likely that comparable (but different) cases exist among all devices. What we are most lacking is some reasonable methodology for exploiting such characteristics via an appropriate matching of device idiosyncrasies with structures of the dialogue.

Strength vs Generality

In the previous example we saw how the idiosyncratic properties of an input device could have a strong affect on its appropriateness for a specific task. It would be nice if the world was simple, and we could consequently figure out what a system was for, find the optimal device for the task to be performed on it, and be done. But such is seldom the case. Computer systems are more often used by a number of people for a number of tasks, each with their own demands and characteristics. One approach to dealing with the resulting diversity of demands is to supply a number of input devices, one optimized for each type of transaction. However, the benefits of the approach would generally break down as the number of devices increased. Usually, a more realistic solution is to attempt to get as much generality as possible from a smaller number of devices. Devices, then, are chosen for their range of applicability. This is, for example, a major attraction of graphics tablets. They can emulate the behavior of a mouse. But unlike the mouse, they can also be used for tracing artwork to digitize it into the machine.

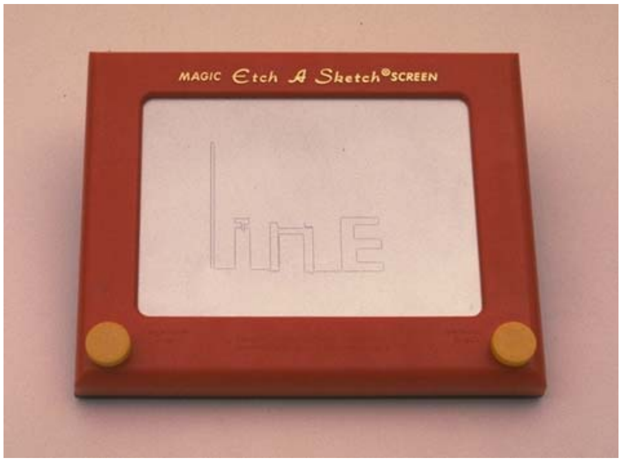
Having raised the issue, we will continue to discuss devices in such a way as to focus on their idiosyncratic properties. Why? Because by doing so, we will hopefully identify the type of properties that one might try to emulate, should emulation be required.

Appropriate Gestures can Simplify Syntax

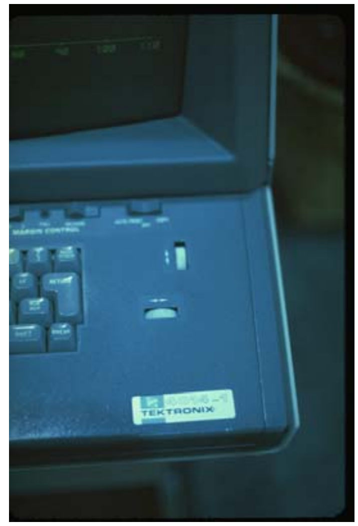
It is often useful to consider the user interface of a system as being made up of a number of horizontal layers. Most commonly, syntax is considered separately from semantics, and lexical issues independent from syntax. Much of this way of analysis is an outgrowth of the theories practiced in the design and parsing of artificial languages, such as in the design of compilers for computer languages. Thinking of the world in this way has many benefits, not the least of which is helping to avoid "apples-and-bananas" type comparisons. There is a problem, however, in that it makes it too easy to fall into the belief that each of these layers is independent. A major objective of this section is to point out how false an assumption this is. In particular, we will illustrate how decisions at the lowest level, the choice of input devices, can have a pronounced effect on the complexity of the system and on the user's mental model of it.

Two Children's Toys

The *Etch-a-Sketch* (shown in Figure 3(a)) is a children's drawing toy that has had a remarkably long life in the marketplace. One draws by manipulating the controls so as to cause a stylus on the back of the drawing surface to trace out the desired image. There are only two controls: both are rotary pots. One controls left-right motion of the stylus and the other controls its up-down motion. For the purpose of the examples that follow, any similarities between the controls of the Etch-a-Sketch and those of the classic *Tektronix 4014 Graphics Terminal* (shown in Figure 3(b)) are purely intentional.



(a) Etch-a-Sketch
A popular children's toy which uses two 1 dimensional controls for drawing. The left knob controls horizontal movement of the drawing "stylus" and the right knob vertical movement.



(b) Tektronix 4014 Graphics Terminal
For those of us of a certain generation, this is the device on which we did some of our first computer graphics in the mid '70s. If you need a push to bridge the gap between the Etch-a-Sketch toy and "serious" computer design, note the similar use of two 1 dimensional rotary potentiometers for graphics input

Figure 3: Computers as Toys / Toys as Computers

The *Skedoodle* (shown in Figure 4) is another toy based on very similar principles. In computerese, we could even say that the two toys are semantically identical. They draw using a similar stylus mechanism and even have the same "erase" operator (turn the toy upside down and shake it). However, there is one big difference. Whereas the Etch-a-Sketch has a separate control for each of the two dimensions of control, the Skedoodle has integrated both dimensions into a single transducer: a joystick.

Figure 4: The *Skedoodle*

Like the Etch-s-Sketch, the Skedoodle is a children's drawing toy. Other than the oval (rather than rectangular "screen", the main difference between the two is how one draws. The computer terminal on the right is to the Skedoodle what the Tektronix 4014 is to the Etch-a-Sketch.

Since both toys are inexpensive and widely available, they offer an excellent opportunity to conduct some field research. Find a friend and demonstrate each of the two toys. Then ask him or her to select the toy felt to be the best for drawing. What all this is leading to is a drawing competition between you and your friend. However, this is a competition that you will always win. The catch is that since your friend got to choose toys, you get to choose what is drawn. If your friend chose the Skedoodle (as do the majority of people), then make the required drawing be of a horizontally-aligned rectangle, as in Figure 5a. If they chose the Etch-a-Sketch, then have the task be to write your first name, as in Figure 5b. This test has two benefits. First, if you make the competition a bet, you can win back the money that you spent on the toys (an unusual opportunity in research). Secondly, you can do so while raising the world's enlightenment about the sensitivity of the quality of input devices to the task to which they are applied.

(a) Geometric Figure (b) Cursive Script

Figure 5: Two Drawing Tasks

What is true with these two toys (as illustrated by the example) is equally true for any and all computer input devices: they all shine for some tasks and are woefully inadequate for others.

> *If you understand the importance of the points being made here, you are hereby requested to go out and apply this test on every person that you know who is prone to making unilateral and dogmatic statements of the variety "mice (tablets, joysticks, trackballs, ...) are best".*
>
> *A good working premise is to assume that every device is best for something and worst for something else. The trick is understanding what those "somethings" are, and finding the right match among device, task, context and user.*

We can build upon what we have seen thus far. What if we asked how we can make the Skedoodle do well at the same class of drawings as the Etch-a-Sketch? An approximation to a solution actually comes with the toy in the form of a set of templates that fit over the joystick (Figure 6).

If we have a general-purpose input device (analogous to the joystick of the Skedoodle), then we can provide tools to fit on top of it to customize it for a specific application. (An example would be the use of "sticky" grids in graphics layout programs.) However, this additional level *generally comes at the expense of increased cost in the complexity of the control structure*. If we don't need the generality, then we can often avoid this complexity by choosing a transducer whose operational characteristics implicitly channel user behavior in the desired way (in a way analogous to how the Etch-a-Sketch controls place constraints on what can be easily drawn).

Figure 6: Skedoodle with Templates

Summary

In this introduction we have introduced the main themes that pervade this book. We have seen that there is an intimate, but poorly understood, relationship among the user's motor-sensory capabilities and intent, the technique used to achieve that intent, and the transducer.

In what follows, we explore this relationship more deeply, and attempt to establish some theory and models that can be applied by designers in making decisions about input.