A Morphological Analysis of the Design Space of Input Devices

STUART K. CARD, JOCK D. MACKINLAY, and GEORGE G. ROBERTSON
Xerox Palo Alto Research Center

The market now contains a bewildering variety of input devices for communication from humans to computers. This paper discusses a means to systematize these devices through morphological design space analysis, in which different input device designs are taken as points in a parametrically described design space. The design space is characterized by finding methods to generate and test design points. In a previous paper, we discussed a method for generating the space of input device designs using primitive and compositional movement operators. This allowed us to propose a taxonomy of input devices. In this paper, we summarize the generation method and explore the use of device footprint and Fitts's law as a test. We then use calculations to reason about the design space. Calculations are used to show why the mouse is a more effective device than the headmouse and where in the design space there is likely to be a more effective device than the mouse.

Categories and Subject Descriptors: H.1.2 [**Models and Principles**]: User/Machine Systems—*human factors* J.6 [**Computer Applications**]: Computer-Aided Engineering—*computer-aided design*

General Terms: Design, Human Factors

Additional Key Words and Phrases: Design knowledge systematization, design rationale, design space, input devices, morphological analysis, semantics

1. INTRODUCTION

Human–machine interface technology has been developed to the point where it is appropriate to systematize existing research results and craft into a body of engineering and design knowledge. A case in point is the design of input devices. A bewildering variety of such devices now exists on the market, including typewriter keyboards, mice, headmice, pens and tablets, dialboxes, Polhemus sensors, gloves, and body suits. How can we make sense of this variety? How can we identify promising opportunities for new design? As part of their development, most engineering disciplines organize the designs that have emerged in terms of abstractions that give insight into the design space (e.g., [32]). This insight allows individual designs to be grouped into families, aids in the teaching of cumulated knowledge, suggests new designs, and may form the basis of toolkits for composing individual designs. In this

Authors' current address: Xerox Palo Alto Research Center, 3333 Coyote Hill Road, Palo Alto, CA 94304.

paper we continue the development of a set of abstractions that provide one method for bringing order to knowledge about input devices.

Previous work on systematizing human–machine input devices has provided three lines of development: toolkits, taxonomies, and performance studies. We argue that a fourth line of development, a morphological design space analysis, can be used to integrate the results of this previous work.

Toolkits. User interface toolkits or user interface management systems help with a wide range of problems, including the construction, run-time execution, and post-run-time analysis of a user interface [33]. They may help systematize input device knowledge by providing a library of prebuilt input device modules [25, 27], architecture and specification techniques for combining these modules [2, 34], or postprocessing analysis tools [24]. Sometimes, as in [2], they even provide architectural models of input device interactions. But the device models implicit in user interface toolkits sketch only a limited picture of the design space of input devices and their properties. Even for the construction of interfaces, they present interface designers with many design alternatives but do little to help with the design decisions themselves. In order to achieve a systematic framework for input devices, toolkits need to be supported by technical abstractions about the user, the devices themselves, and the task they are used to perform.

Taxonomies. Two recent attempts have been made to provide taxonomies of the design space of input devices. Foley, Wallace, and Chan [15] focused on computer graphics subtasks. They classified input devices under the graphics subtasks they were capable of performing (e.g., the tablet and the light pen are capable of character recognition). They also reviewed experimental evaluations of input devices. Buxton and Baecker [4, 7] proposed a taxonomy of input devices classified according to the physical properties and the number of spatial dimensions they sense. The limitation of the Foley–Wallace–Chan scheme is that the categories, while reasonable, are somewhat ad hoc, and there is no attempt at defining a notion of completeness for the design space. The limitation of the Buxton and Baecker scheme is that it only includes continuous devices. In addition to the two taxonomies, Bleser and Sibert have designed a tool for selecting interaction techniques based on their characteristics [6]. Rather than abstracting the design space, the goal of their tool is to include most of the factors, including the physical packaging of an input device, in a design tool. The importance of attempts to make taxonomies of input devices is that they make progress in helping us to understand the design space—not only the devices that exist and their relationships, but also potential devices that might be invented.

Performance studies. Studies have been made of human performance with pointing devices. English and Engelbart [12] studied several devices and found the mouse the fastest device. Card, English, and Burr [9] confirmed these empirical results and discovered that pointing speed with the mouse is governed by Fitts's law [14] with a bandwidth similar to that of the hand. Subsequent studies have empirically compared speed and preference of

various devices and confirmed and improved the use of Fitts's law [1, 13, 15, 17, 19, 20, 36]. Unfortunately, these studies have not always agreed, largely because some studies have not attempted to disentangle task, subject, and human performance variables. The performance studies have, however, established abstractions, including Fitts's relation [20], which, in turn, expresses the linkage between device performance and fundamental performance parameters of the human.

Morphological design space analysis. In this paper, we begin to relate this previous work by employing another technique, morphological design space analysis, in which we seek to comprehend different input device designs as points in a parametrically described design space. The goal is to find abstractions both for generating the design space and for testing the designs contained therein. In order to represent the designs as points in this design space, some parametric representation is determined that can represent the central idea of particular designs. The essence of this technique has previously been used by a small number of researchers/designers in the analytical study of other design spaces. Bertin [5] used a similar method in his study of the semiology of graphics and applied it to the graphical design of diagrams, networks, and maps. Zwicky [37] used a similar method to generate the design space of jet engines. In both cases, the result was insight into the properties of the design space as well as into the production of novel designs. Mackinlay [21, 22] applied the technique to human–computer interfaces. He formalized Bertin's analysis and used it to generate graphic designs automatically for the presentation of statistical data. Human psychophysical data were used to select from among those designs that accurately expressed the data, those that were the most effective in communicating it. Several researchers have since built on Mackinlay's analysis, either to extend it to new domains or to add more analysis of the user's tasks and goals (e.g., [11, 30]).

Mackinlay, Card, and Robertson [23] applied the morphological approach to the analysis of input devices. They built on results of the taxonomies to propose a parametric representation to be used in generating points in the input device design space. This paper continues that analysis, exploring how results from input device performance studies can be integrated on the test side of the generate-and-test paradigm to give additional insight into the design space. First, we summarize the representation of the design space, and then we discuss how figures of merit might be associated with it.

2. GENERATING THE DESIGN SPACE

Let us reflect, for a moment, on the role of input devices. An input device is part of the means used to engage in dialogue with a machine. Unlike human–human conversation, the dialogue is between fundamentally dissimilar agents, in terms of both perception and processing. Furthermore, it takes place under conditions (e.g., the persistence of displays) that are different from the evanescent, sequential, oral conversation that is often taken as the model for communication. Instead of words, the user may move the mouse

and press buttons, and the machine may show highlighted animated diagrams.

The design of human–machine dialogues is, at least in part, the design of artificial languages for this communication. Mackinlay [21, 22], in work on the automatic generation of displays, suggested that each display could be thought of as a sentence in a formal language and that such sentences could be analyzed as to their ability to transmit an intended semantic meaning from the machine to the user. In the case of input devices, we can analyze the manipulations of an input device as sentences in a formal language of communication. In particular, we can try to analyze the sentences that are possible with an input device in terms of combinatoric operators. What are the primitive moves? What are the composition operators for combining primitive moves into complex sentences of moves? The combination of primitive moves and composition operators gives us a parametric abstract representation that represents the space of sentences it is possible to transmit. In a sense, this generates the design space of input devices. Input devices are those devices that allow some portion of these potential sentences actually to be realized and communicated from human to machine. Of course, the human has a communicative intention. This communicative intention must be encoded into the simple sentences of input device movements (the coding may be variable, depending on the situation of the moment) and then decoded by the application. In our analysis, we make the idealized assumption that functions of the application program express the semantics of the interaction. A radio, for example, may have a function *increment-volume-to* (*loudness*). A possible human intention may be expressible in terms of invoking this function. The job of an input device, such as a volume control knob, is to allow the user to make some combination of moves such that the moves can be interpreted as actually invoking the desired function with the desired parameters.

Conceptually, the most general case of human–machine interaction is the case of a human interacting with an embedded computer (e.g., the autopilot on an airplane). Such an interaction can be modeled as the interaction in an artificial language among at least three agents [8, 31]:

(1) a human,
(2) a user dialogue machine, and
(3) an application.

We can trace the semantics of an input device by tracing the mappings from human action through mappings inherent in the device and finally into changes in the parameters of the application.

There are two key ideas in modeling the language of input device interaction:

(1) a *primitive movement vocabulary*, and
(2) a set of *composition operators*.

The movement vocabulary gives the elementary sentences that can be expressed in the artificial language. The composition operators give methods of combining this vocabulary into a combinatorically richer set.

2.1 Primitive Movement Vocabulary

We begin with the observation inspired by Baecker and Buxton [4] that

> *basically, an input device is a transducer from the physical properties of the world into logical parameters of an application.*

Formally, we represent the input device as a six-tuple,

$$\langle \mathbf{M}, \mathbf{In}, \mathbf{S}, \mathbf{R}, \mathbf{Out}, \mathbf{W} \rangle,$$

where

—**M** is a manipulation operator,
—**In** is the input domain,
—**S** is the current state of the device,
—**R** is a resolution function mapping from the input domain set to the output domain set,
—**Out** is the output domain set, and
—**W** is a general-purpose set of device properties that describe additional aspects of how a device works (perhaps using production systems).

Generally, we assume somewhat idealized devices without stickiness, significant lags, noise, linearity problems, etc. But noise and linearity could be modeled through the **R** input-to-output mapping, and lag could be modeled through the **W** work properties mechanism.

Table I lists the various manipulation operators, **M**, possible for an input device. They are an extension of the physical properties suggested by Baecker and Buxton [4]. They represent all combinations of linear and rotary, absolute and relative, and position and force. Although other input devices are possible (based, say, on speech or heat), virtually all input devices use some combination of the properties listed in Table I.

Figure 1 illustrates the description of a simple set of radio controls, using our primitive movement vocabulary. The volume knob is rotated about the Z axis (conventionally assumed to be normal to the panel surface). It is a continuous device that maps using the identity operator from an input domain set of 0–270 degrees into the same set. The selector knob, on the other hand, maps from the set consisting of 0–90 degrees into the ordered sequence $\langle 0, 45, 90 \rangle$ degrees. Finally, the station knob is a dial that moves any number of turns to the right or to the left. It is presumed to connect to a slider that moves back and forth between 0 and 5 inches. The station knob is a relative device. It keeps turning after the slider is against one side and no

Table I. Physical Properties Used by Input Devices

	Linear	Rotary
Position		
Absolute	Position **P**	Rotation **R**
Relative	Movement **dP**	Delta rotation **dR**
Force		
Absolute	Force **F**	Torque **T**
Relative	Delta force **dF**	Delta torque **dT**

longer moves. But, if the knob direction is reversed, then the slider reverses immediately. The volume knob, the selection switch, and the slider each go through another mapping into the parameters of an application.

2.2 Composition Operators

The example in Figure 1 also illustrates the notion of a composition operator. The output domain set of the station knob is mapped into the input domain set of the slider. This sort of composition operator is called a connection. There are three composition operators:

(1) merge composition,
(2) layout composition, and
(3) connect composition.

These operations are illustrated in Figure 2 for the mouse. *Merge composition* is the combination of two devices such that the resulting input domain set is the cross product of the input domains of the two devices. A mouse can be thought of as the merge composition of two orthogonal one-dimensional sliders. *Layout composition* is the collocation of two devices on different places of a common panel or space. The three buttons of a mouse and the *XY* sensors are all four layout-composed together to form the mouse. *Connect composition* occurs when the output domain of one device is mapped onto the input domain of another device. For the mouse, the output is connected to the input for the screen cursor. The screen cursor, of course, is not actually a physical device. This illustrates another point about the modeling scheme, namely, that devices do not have to be physical devices, but can also be virtual devices implemented in software, such as the cursor.

The modeling scheme uses a formalized notation, which we have not reproduced here, to keep track of mappings of movements of the input devices (signals, really) through various transformations into application-defined meanings. See [23] for more details.

2.3 The Design Space for Input Devices

The design space for input devices is basically the set of possible combinations of the composition operators with the primitive vocabulary. We graph a simplified partial visualization of this space in Figure 3. This is our equivalent to Foley, Wallace, and Chan's [15] and Buxton's [7] classifications. A

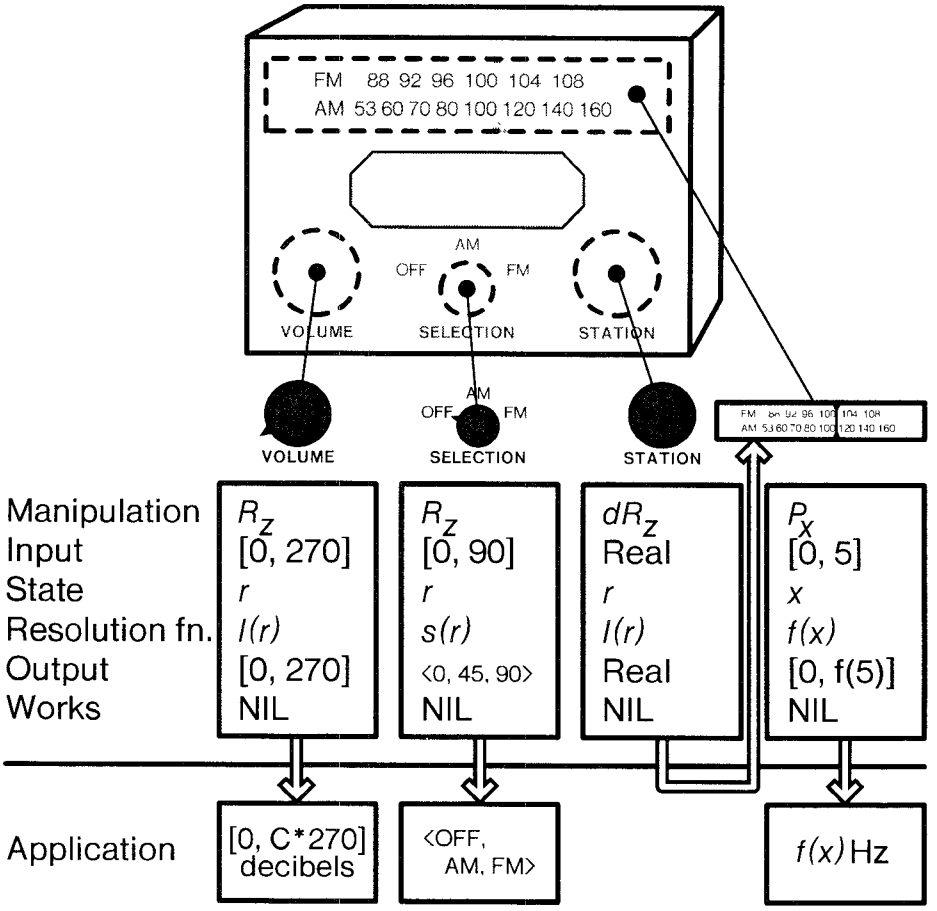

Fig. 1. Analysis of a simple radio. Two rotational devices are connected directly to the application. The third rotational device is connected to a positional device, which is then connected to the application.

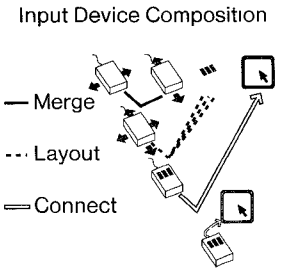

Fig. 2. Composition operators used in describing the mouse

device is represented in the figure as a set of circles connected together. Each circle represents a transducer in the device, plotted according to the canonical physical property it transduces. Each line indicates a composition operator: connection composition (double-line arrow), layout composition (dotted

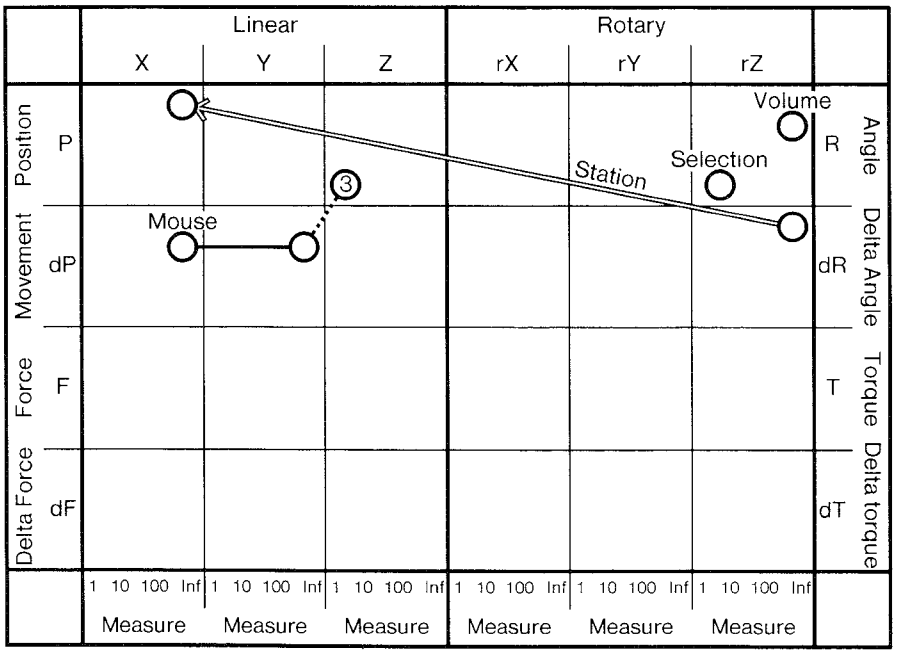

Fig. 3. Input device taxonomy. The diagram describes an input device taxonomy that is based on the analysis presented in this paper. Circles are used to indicate that a device senses one of the physical properties shown on the vertical axis along one of the linear or rotary dimensions shown on the horizontal axis. For example, the circle representing the radio volume control indicates a device that senses an angle around the Z axis The position in a column indicates the number of values that are sensed (i.e., the measure of the domain set). For example, the circle representing the selection control represents a discrete device. Lines are used to connect the circles of composite devices. A black line represents a merge composition (such as the X and Y components of the mouse). The dashed line represents a layout composition (such as the three buttons on a mouse, represented by a circle with a 3 in it to indicate identical devices).

line), or merge composition (black line). It is important to note that each group of linked circles in Figure 3, which collectively represent a device, is only a single point in a very large design space and that variants in devices are possible (e.g., in the input or output domains or the mappings), which are below the level of detail visualized in the figure. These variants are, however, describable in the more formal notation of [23].

In Figure 3 we have plotted the devices of our radio example and the mouse to illustrate their use. The radio volume knob is in the cell for sensors of angles relative to the Z axis. It is located on the right side of the cell, showing that it is continuous. The selection knob is similar but is located nearer the left side, showing that it takes just a few values. The station knob is located in the cell for relative angle and is connected to a slider for the tuning mechanism. A mouse is depicted in the figure as a circle on X movement, a circle on Y movement, and a circle containing the number 3 on Z positioning. This indicates that the mouse is a layout composition of four

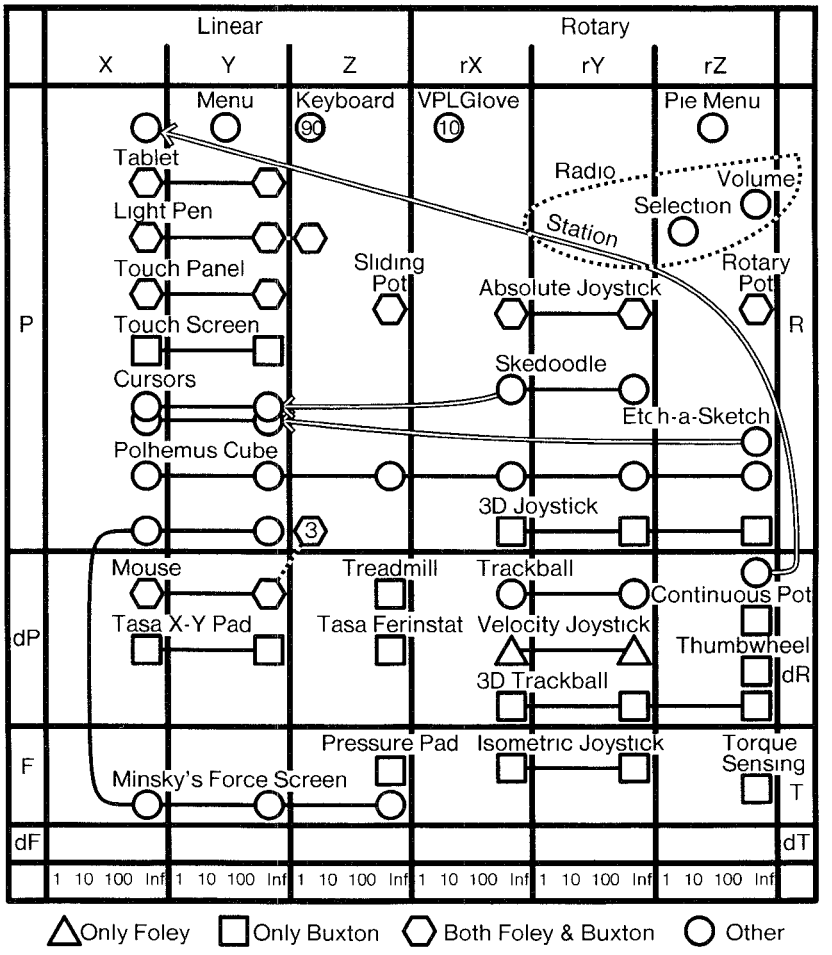

Fig. 4. A broad range of input devices plotted on the taxonomy. Devices previously classified by Foley, Wallace, and Chan [15] and by Baecker and Buxton [4, 7] are indicated by triangles, squares, and hexagons. Hexagons indicate devices included in both previous taxonomies. Other devices, indicated by circles, include the radio devices described previously and some unusual devices to demonstrate the generality of the taxonomy.

devices: one device that is itself the merge composition of two elementary devices sensing change in X and Y, and three other devices that are simple buttons. The placement of the X and Y circles to the right of the column indicates nearly continuous resolution. The location of the button circles to the left indicates controls with only two states.

To demonstrate the coverage of the taxonomy, we have reclassified the devices listed by Foley, Wallace, and Chan [15] and by Buxton and Baecker [4, 7] (see Figure 4). With the exception of voice, we have been able to position all of the devices considered so far. Furthermore, it is possible to generate potential new devices by placing circles in various cells of the

diagram. Of course, many of these devices might be undesirable, but the point is that Figure 3 is a sufficiently rich depiction of the design space for input devices that it can be used both to classify nearly all existing devices and to generate ideas for new devices not yet invented. In particular, we have used our model of the input device design space to help design novel egocentric motion devices for virtual 3-D environments [23, 29].

3. TESTING POINTS IN THE DESIGN SPACE

Up to this point, we have described how to model the space of input device designs, including methods to help generate the space. We have shown that we can distinguish systematically among devices used in other taxonomies. But we also need to be able to test points in the design space in order to characterize regions of it. We need to be able to utilize results from performance studies of input devices. Following Mackinlay [21, 22], the mappings implied by specific input device designs can be evaluated according to two basic criteria: (1) *expressiveness* (the input conveys exactly and only the intended meaning) and (2) *effectiveness* (the input conveys the intended meaning with felicity).

In some design spaces, expressiveness is a major concern. For example, in the design of visual displays it may be relatively easy to generate a display that conveys an intended meaning, but difficult to prevent the display from conveying some unintended meaning as well (e.g., an arbitrarily ordered set of bars on the bar chart visually conveys an ordering relationship among the bars that is not intended). In the design of input devices, an expressiveness problem arises when the number of elements in the **Out** set does not match the number of elements in the **In** set to which it is connected. If the projection of the **Out** set includes elements that are not in the **In** set, the user can specify illegal values; and if the **In** set includes values that are not in the projection, the user cannot specify legal values. Perhaps the user wishes to convey the meaning of the system "Select pixel $x = 105$, $y = 32$" with a device that has a resolution of $\frac{1}{4}$ in (as for some touch panels). The user will not be able to express the request exactly, and there will be some loss of expressiveness, serious or not depending on the situation.

More interesting for input devices is effectiveness, and this is the aspect on which we will dwell. Assuming that it is possible to express the user's intention in a sentence in the repertoire of the input device, there is still the question of how well this can be done in terms of speed, errors, or other figures of merit for an input device (in a particular task for a particular user). Possible figures of merit include the following:

—*Desk footprint*. The amount of area consumed by the device on the desk.

—*Pointing speed* (really, *device bandwidth*). How quickly the device can be used to select a target. In addition to just the simple time difference, devices that are slower than the user's unaided hand put stress on the user, because the slower pointing time may reflect more difficult guidance by the user (e.g., error correction) and may make the user have to think about

operating the device consciously rather than unconsciously using the tool to manipulate the environment.

—*Pointing precision.* How small a target can conveniently be selected with the device.

—*Errors.* There are a number of interrelated metrics for errors: percentage of times a target is missed, the end-point distance from the target, and various statistical measures of distributions of these, such as the root mean squared error of a drawn line compared to a model line.

—*Time to learn.* The amount of practice required before a user is proficient with a device.

—*Time to grasp the device.* How long it takes to engage the device if the hands are doing something else.

—*User preference.* How well users like the device and whether they prefer it to other devices.

—*Cost.* The retail cost.

These figures of merit include human performance measures such as speed and errors, as well as pragmatic concerns such as desktop footprint, panel space, or cost. To illustrate how we can annotate the design space of input devices (and, in particular, the representation of it given in Figure 4), we discuss two of these figures of merit: footprint and bandwidth. It is footprint we choose because it is concrete and straightforward and, therefore, makes an easy example. It also illustrates how nonperformance pragmatic concerns are of importance.

3.1 Footprint

An input device requires a certain amount of space on a desk: its footprint. Desk space is a small, finite resource; hence, a small footprint is better than a large one. The actual footprint of some devices such as the mouse depends on the sequence of actions in an application. But, to estimate the space required, we can use an extreme task.

As an example, compare the relative desk footprint of different input devices for pull-down menus on the original Macintosh (12″ screen) and on the SuperMac (19″ screen). The mouse must be able to move from any place on the screen to the menu bar, which is at the extreme top of the screen. The footprint is, therefore, an image of virtually the entire screen in the movement of the device. Table II estimates the footprint for various devices. Three of the devices (light pen, on-screen touch pad, and rotary potentiometers) require no additional footprint (assuming that no footprint is needed to store the lightpen and that the potentiometers mount on the case). Two of the devices (trackball and joystick) have small footprints that are independent of screen size. Two of the devices (mouse and tablet) have footprints that vary dramatically as the screen size increases from 12″ to 19″. The $C{:}D$ ratio for the mouse and tablet in Table II is the "control–display ratio." The traditional control-display ratio for a mouse is 1:2; that is, for each inch of control (mouse) movement, the display (cursor) moves two inches.

Table II. Footprint Estimates for Various Input Devices

Device	Footprint (in^2) 12″ screen	19″ screen
Light pen	0	0
On-screen touch pad	0	0
Rotary pots	0	0
Headmouse	0	0
Trackball (2″ × 2″)	4	4
Joystick (2″ × 2″)	4	4
Mouse (C : D = 1 : 2)	4	43
Tablet (C : D = 1 : 1)	69	173

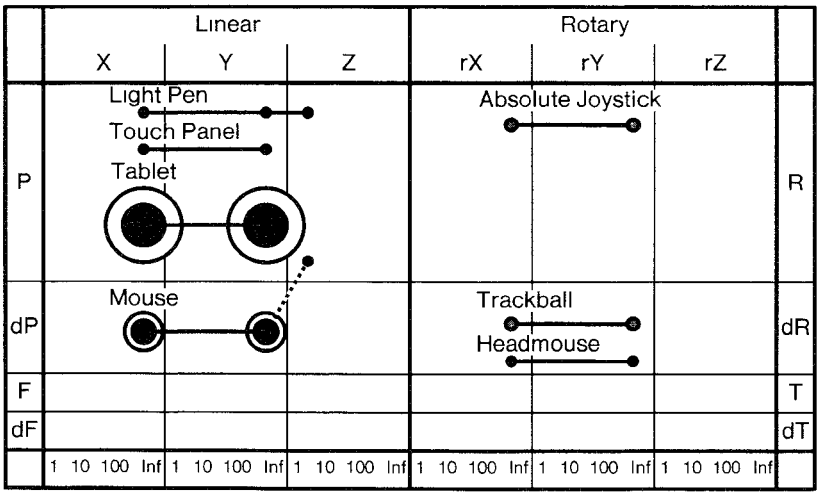

Fig. 5. Footprint of input devices for Macintosh pull-down menus. The filled circles describe the device footprint for the 12″ screen, and the open circles describe the device footprint for the 19″ screen.

In Figure 5 we have annotated our depiction of the input device design space, by using the area of the circle representing each sensor to indicate the footprint required. The filled circles are for a 12″ screen, and the open circles are for a 19″ screen. Black dots represent no additional footprint. Several facts about the design space of potential devices are evident from the diagram: (1) The tablet and the mouse are very expensive in footprint relative to other devices. (2) Going from 12″ to 19″ displays causes a major increase in footprint size for the mouse and tablet (unless the control-display ratio is changed). In fact, this dramatic increase in footprint for the mouse as a function of screen size is behind two innovations in mouse design: One of these, powermice (the control-display ratio changes as a function of velocity),

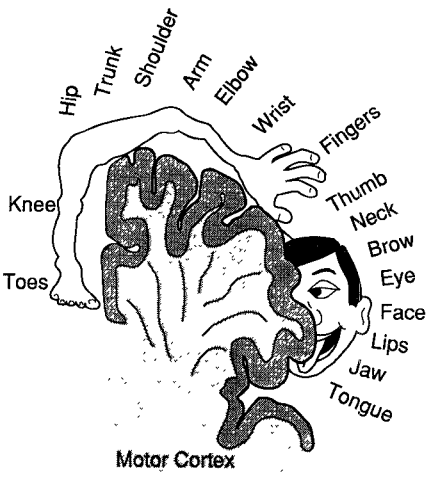

Fig. 6. Motor regions of the cerebral cortex, after [26].

sacrifices pointing speed to get a smaller footprint [16]. The other, higher-resolution mice, uses an improved sensor to get a smaller footprint without sacrificing speed.

3.2 Bandwidth

Now let us turn to another figure of merit, bandwidth. It is usually desirable for an input device to be as fast to use as possible, but speed of use is actually a joint product of all three elements in our model: (1) the human, (2) the application, and (3) the device. For the moment, we restrict ourselves to tasks that involve pointing at a target with a continuous device.

The speed and error performance of a particular device may depend on a number of subtleties such as the relationship between the geometry of the device and the geometry of the hand or coefficients of friction. But we can give a rough characterization of input device design effectiveness in terms of the following:

(1) human—the bandwidth of the human muscle group to which the input device transducer is attached;
(2) application—the precision requirements of the task to be done with the device; and
(3) device—the effective bandwidth of the input device.

Bandwidth of the human muscle group. Some groups of muscles can be controlled more finely than other groups of muscles. Figure 6 shows that more of the motor cortex is devoted to some groups of muscles (such as the fingers) than to others (such as the neck). Undoubtedly, the determinants of muscle performance are more complex than just simple cortical area; still, roughly speaking, those groups of muscles having a large area devoted to them are heuristically promising places to connect with input device

transducers if we desire high performance. Of course, there are several factors to be taken into account (e.g., promising muscle groups may already be occupied with other tasks or be physically or socially inconvenient, or the task may only require crude control), but the point is that the muscle group that is connected to an input device may impose inherent bandwidth limitations on that device.

Figure 7 shows data from experiments by Langolf [18] and by Radwin, Vanderheiden, and Lin [28]. Subjects in Langolf's experiment performed a peg insertion task under the microscope for the curves marked finger and wrist, and the Fitts's dotting task [14] for the curve marked arm. Langolf observed the approximate muscle groups being used and made the observation that different muscle groups gave rise to different Fitts's law slopes. Subjects in Radwin, Vanderheiden, and Lin's experiment used a headmouse to select targets on a CRT. The figure plots the movement times observed in both experiments as a function of Fitts's Index of Difficulty,

$$MovementTime = K + (I_M \cdot I_D), \tag{1}$$

where K is a constant that depends on how the target is selected and how the trial is begun, I_M is the reciprocal of the bandwidth of the device, and I_D is the Fitts's Index of Difficulty,

$$I_D = \log_2 \frac{2D}{S}.$$

D is the distance to the target, and S is the width of the target. Bandwidth is measured in bits/s. The convenient units for I_M are ms/bit. Figure 7 gives a rough index for the bandwidth of different parts of the body.

Precision requirements of tasks. The user will deploy the input device in subtasks of the application. We can characterize an application by listing a selection of these tasks and by analyzing input device performance relative to this set of tasks. Table III gives some values from such an analysis for simple text editing. The target size (column (1) in the table) ranges from 5.5 cm for a typical paragraph to 0.069 cm for a period. Associated with each task, we have shown a calculation of Fitts's Index of Difficulty, I_D, in column (2). A technical difficulty is that eq. (1) has been shown to be inaccurate for pointing tasks with I_D below about 2 bits (very easy tasks) [35]. We can overcome this problem by using Welford's more accurate method for computing I_D:

$$I_D = \log_2 \left(\frac{D}{S} + .5 \right).$$

I_D ranges from 1.18 bits for pointing to a paragraph to 7.14 bits for pointing to a period. Since we do not have specific data on target distance, the calculation assumes a typical distance of half way from the center of a 19″ screen to the side edge, that is, $A = 9.7$ cm. In a more refined analysis, actual empirical data could be used or different assumptions could be made.

Device bandwidth. Now consider the effectiveness of an input device for a given application. The input device is connected to a certain set of human

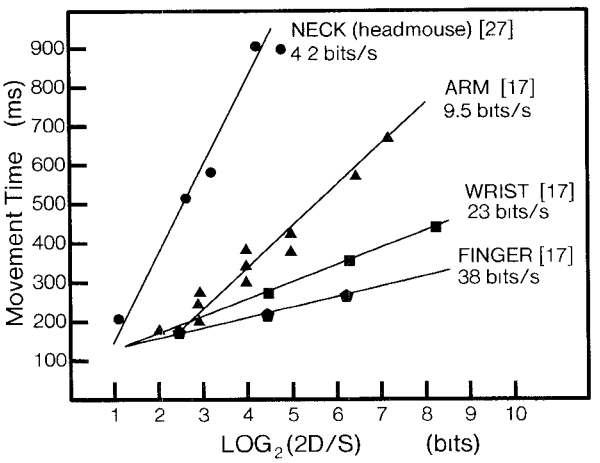

Fig. 7. Summary of Fitts's dotting experiments [18, 28].

Table III. Fitts's Law Slopes for Various Devices

			Movement time		
Target	(1) Size S[a] (cm)	(2) I_D (bits)	(3) Mouse (ms)	(4) Headmouse (ms)	(5) Fingers (ms)
Paragraph[b]	5.5	1.18	113	280	30
Word[c]	2.3	2.24	220	540	56
Character[d]	0.41	4.59	440	1100	115
Period[e]	0.069	7.14	690	1710	179

[a]Based on 74 pixel/in. = 29.1 pixel/cm, as displayed on a Macintosh screen [3, p. 214].
[b]Estimated size based on 10 lines × 16 pixel/line = 160 pixels/29.1 pixel cm^{-1} = 5.5 cm.
[c]Based on 5.5 characters/word = 60 pixels.
[d]Based on character n = 12 pixels. Times Roman 12-point font on a Macintosh. (Examined and counted in MacDraw II.)
[e]Based on a 2-pixel-wide dot.

muscles. This determines an upper bound on the bandwidth of the device (as indexed by the Fitts's law slope). Of course, the device itself can degrade the achievable performance. Currently, empirical testing is required to check whether this is the case. But the performance of many devices is more or less set by the muscle groups with which the device is designed to connect. For the moment, we assume that this is so in order to compute the inherent limitations imposed on a device by the muscle group to which it is connected.

We would like to use Figure 7 to characterize a device according to what tasks are easy for it to perform. But, first, three technical details must be addressed. The lines in Figure 7 have intercepts near, but not exactly at, the origin. This reflects small details of the experimental conditions such as how the target is selected and how a trial is started. These vary slightly from line to line and make comparisons among the muscle groups more difficult.

Therefore, for the purpose of our calculation, we set each of the intercepts exactly to zero. Second, since Langolf's measurements [18] were made using a serial task and since we are interested in discrete tasks, we have to adjust for the 2 ~ 3 bits/s higher value in continuous tasks [20]. We have, therefore, added 2 bits/s Langolf's measurements for the arm, wrist, and fingers. Third, we wish to adjust the slopes to use Welford's formulation of the Fitts's Index of Difficulty. Mackenzie's reanalysis of Fitts's data ([19, Table 2]) suggests that this would reduce the measured bandwidth by 0.3 ~ 0.4 bit/s. However, since this is relatively small in comparison to the adjustment already made for a discrete task, no additional adjustment seems warranted. The lines in Figure 8 now reflect our best estimate of bandwidth differences among the muscles. These normalizations are compatible with direct measurements on the mouse [9], which have been added to Figure 8.[1]

We now return to the question of which tasks are easy and which are hard to perform with a device. Of course, tasks with a high Fitts's Index of Difficulty are harder and those with a low Index of Difficulty are easier, but just as we can talk about days that are hot and days that are cold and reference these terms to approximate regions on the Fahrenheit temperature scale, we can find approximate regions on the Fitts's Index of Difficulty scale. We start by estimating the size thresholds where pointing tasks become easy and hard. The mouse is currently the dominant pointing device, so it makes a convenient comparative reference point. Subjectively, pointing to a word is a relatively easy target with a mouse. But pointing to targets smaller than an average word (of 5.5 characters = 2.3 cm) begins to be less than easy. So we take a word as the *hardest easy target*. This choice is supported by the fact that many mouse-based text editors have special features for selecting a word (e.g., double clicking anyplace within the word) partially so that the user can avoid the harder, more precise selections when possible. Pointing to a word at our assumed distance of D = 9.7 cm and I_M = 96 ms/bit for the mouse from Figure 8 takes[2]

$$
\begin{aligned}
MovementTime[\,hardest\ easy\ task\,] &= MovementTime[\,mouse,\ word\,] \\
&= I_M \cdot I_D \\
&= I_M \cdot \log_2 \frac{D}{S} + .5, \\
&= 96\ \text{ms/bit} \cdot \log_2 \left(\frac{9.7}{2.3} + .5 \right) \\
&= 96\ \text{ms/bit} \cdot 2.24\ \text{bit} \\
&= 220\ \text{ms}.
\end{aligned}
$$

[1] We have used data from our earlier experiments on the mouse, since these simulate a discrete text pointing situation. Recent experiments [19] get a lower bandwidth for the mouse, but use a continuous task and a mouse with variable gain.

[2] Most of the calculations in this paper are rounded to two significant digits (some tabulated intermediate values, such as I_D, are kept at three significant digits to lessen round-off errors).

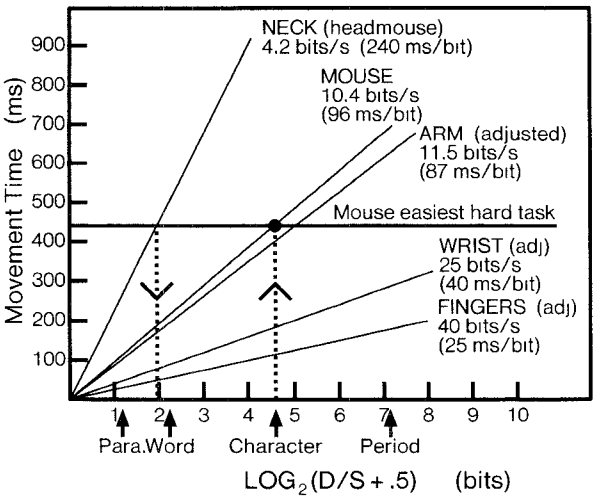

Fig. 8. Simplification of Figure 7 suitable for calculations.

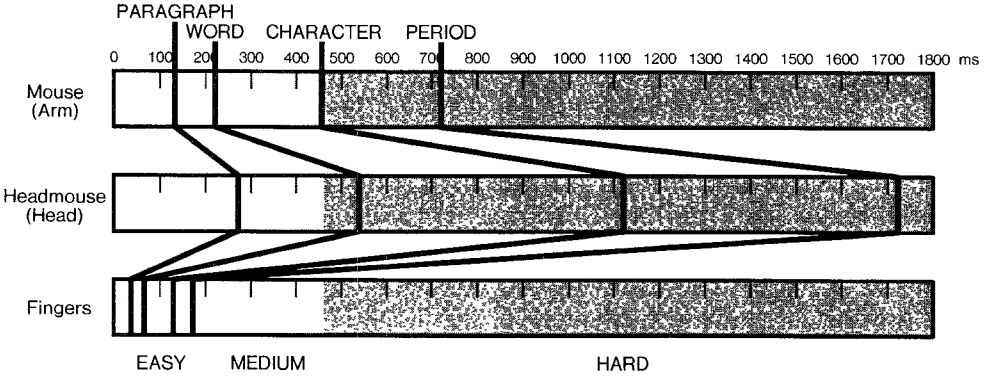

Fig. 9. Calculated effect of muscle group on task difficulty. The figure shows the time for the tasks in Table III for a perfect device connected to different muscle groups.

Therefore, we classify tasks that require less than 220 ms as *easy*. This relationship is shown in Figure 9.

Subjectively, pointing to a character is relatively hard with a mouse. But pointing to targets larger than a character (of typical width 12 pt = .41 cm on a Macintosh) begins to feel easy. So we take a character as the *easiest hard target*. Pointing to a character at our assumed distance takes 440 ms. We therefore classify tasks that require more than 440 ms as *hard*. The pointing tasks of Table III are classified in Figure 9a into easy, medium, or hard using this scheme.

We can use our classification to define the *precision* of a device. For the mouse, we define its precision to be the Fitts's Index of Difficulty I_D of the

easiest hard task. For the mouse, this is the I_D of the character, or 4.59 bits. This line of reasoning can be extended to other devices, using the mouse as our standard.

> *We characterize the* precision *of a device as the I_D that requires the same amount of time as the* easiest hard task *of the mouse.*

On the basis of our assumption that the typical distance is $D = 9.7$ cm, we can reduce this computation to

$$\begin{aligned} \textit{Device precision} &= \textit{Time}[\textit{ hardest easy task for mouse}]/I_M \\ &= \frac{440 \text{ ms}}{I_M \text{ bits}}. \end{aligned} \tag{2}$$

Of course, it is to be understood that differences in particular devices, systems, tasks, and users mean that these computations are approximate to a degree. Nonetheless, the comparisons we shall make are not very sensitive to the exact values selected for the parameters. Notice that, although we have used the mouse to set our definition of easy and hard tasks, the definition is not dependent on the mouse. The mouse is simply used as a convenient and familiar way to determine the approximate time boundaries of these categories.

The computation implied by this definition is depicted graphically by the dashed line in Figure 8. A line extending vertically from the I_D for the easiest hard task for the mouse intersects the mouse's Fitts's law slope. A line extending horizontally from this intersection point intersects the Fitts's law line from another muscle group or device. A line extending vertically down from this second intersection point determines the I_D for the target that could be pointed to in the same time as the easiest hard task for the mouse. This is taken as the precision of the device. A similar graphical computation can be used with this figure to determine comparisons between devices, such as: What sort of target could have been pointed to by a mouse in the time it takes a headmouse to point to a word?

3.3 Example 1: Effect of Muscle Groups on Input Devices

Now let us calculate the consequences of which muscle groups are used in building an input device. Compare the mouse (which uses a lot of arm and shoulder movement) to a headmouse, a plug-compatible replacement for the mouse based on neck movements. A headmouse has three ultrasonic sensors worn on the head like earphones, a transmitter mounted on the display, and plugs into the mouse port. Moving the head changes the XY coordinates of the mouse cursor in the appropriate manner.

We compute a comparison of the two devices based on the application tasks in Table III. Column (3) in Table III shows the time required to point to each task target with the mouse based on the approximation in Figure 8. For example, a word has an I_D of about 2.24 bits and requires about 220 ms to point to, as previously calculated.

Table IV. Characteristics of Input Devices

Property	Device (1) Mouse	(2) Headmouse	(3) Fingers
Bandwidth[a]	10.4 bits/s	4.2 bits/s	40.0 bits/s
I_M[b]	96 ms/bit	240 ms/bit	25 ms/bit
Device precision[c]	4.6 bits	1.8 bits	17.6 bits
Mouse-relative precision[d]	100%	39%	380%

[a]Data from Figure 8.
[b]I_M = 1/bandwidth.
[c]Device precision = $[I_{M(\text{mouse})}/I_{M(\text{device})}]\log_2(9.7\text{ cm}/0.41\text{ cm} + .5)$.
[d]Mouse-relative precision = device-precision/mouse-precision.

Column (4) shows the amount of time the headmouse requires to point to the same target, according to Figure 8. For a headmouse, pointing to a word requires

$$\begin{aligned} MovementTime[\,headmouse, word\,] &= 240\text{ ms/bit} \cdot 2.24\text{ bits} \\ &= 540\text{ ms}. \end{aligned}$$

Figure 9b depicts graphically the amount of time required to point to each of the targets using the mouse as compared to the headmouse. The spectrum of pointing times is shifted to the right. Easy tasks become hard or moderate. The figure makes clear the penalty incurred by using the neck muscles with the headmouse instead of the arm muscles with the mouse: A user can point to a character with the mouse in about the same time it takes a user with the headmouse to point to a word.

Another effect of muscle group used can be seen by comparing the computed precision of the mouse and headmouse. Using eq. (2), the precision of the headmouse is 440 ms/(240 ms/bit) = 2.0 bits, as compared to the 4.6 bits we computed for the mouse earlier. This means that the mouse is more precise. To be concrete, it means that the headmouse could only point to a target with an $I_D = 1.8/4.6 = 39$ percent as great in the same amount of time.

It is interesting to calculate the performance that might be achievable if we were able to couple the high-performance fingers (whose bandwidth we have estimated at 40 bits/s in Figure 8) with the transducer of an appropriate input device. This computation is carried through in column (5) of Table III. Our characterization for the mouse, headmouse, and ultimate finger-operated devices is summarized in Table IV. The computed device precision for the fingers is 17.6 bits, almost four times as much as for the mouse. If such a device could be built, Figure 9c shows that all of the tasks, even the high-precision task of pointing to a single period, might be made into easy tasks. Whether this performance is practically achievable is unknown, but the calculation shows a region of promise in the design space. This is, of

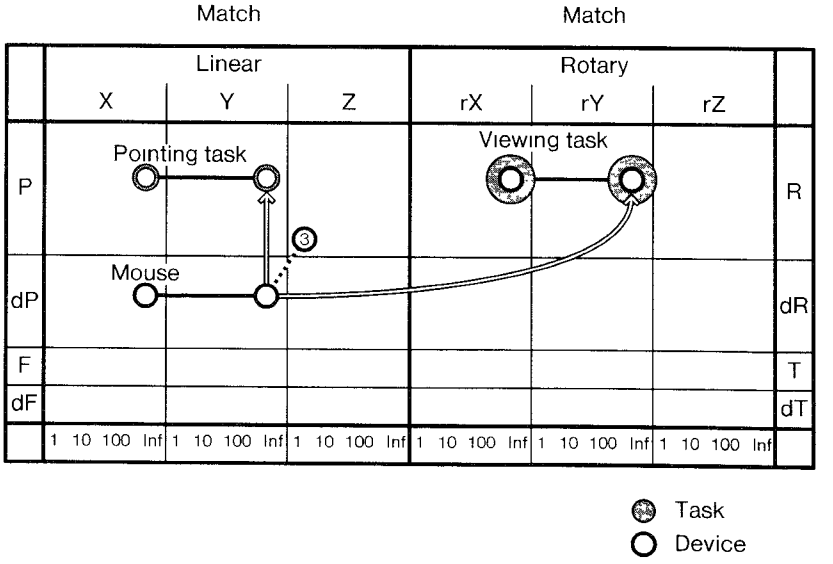

Fig. 10 Analysis of a virtual head movement device based on the mouse. White circles indicate device sensors, and gray circles indicate tasks. Size reflects the precision of the device, as defined in the text, or the precision required by the task A smaller circle indicates higher precision, with the mouse and pointing corresponding to character precision and viewing to paragraph precision. When the white device circle is smaller than the gray task circle, the device is adequate for the task. According to the figure, the mouse can be used both for viewing and for pointing.

course, precisely the sort of speculation we wish to enable by systematizing the design space of input devices.

3.4 Example 2: Display Selection in a 3-D Information Environment using Mouse and Headmouse

We now apply the preceding analysis of device bandwidth to one of our designs for virtual input devices in a virtual workspace. In this design, the user is given a simulated head in a virtual 3-D environment. Moving the mouse forward and back rotates the virtual head up and down; moving the mouse left and right rotates the head to the left or to the right. The screen also contains a circle cursor fixed to its center that can be used to point into the virtual environment. The user points to an object by moving it to the cursor and pressing a mouse button. Thus, the user can accomplish two basic tasks with this arrangement: viewing the virtual world and pointing at objects.

Figure 10 shows the set of connected devices and tasks implied by this description. The mouse is connected to a 2-D rotational task for viewing the virtual world and a 2-D positional task for moving an object to the cursor. Devices and their projection onto tasks are shown with filled circles. The size of the circles approximate the precision of the devices and the difficulty of the

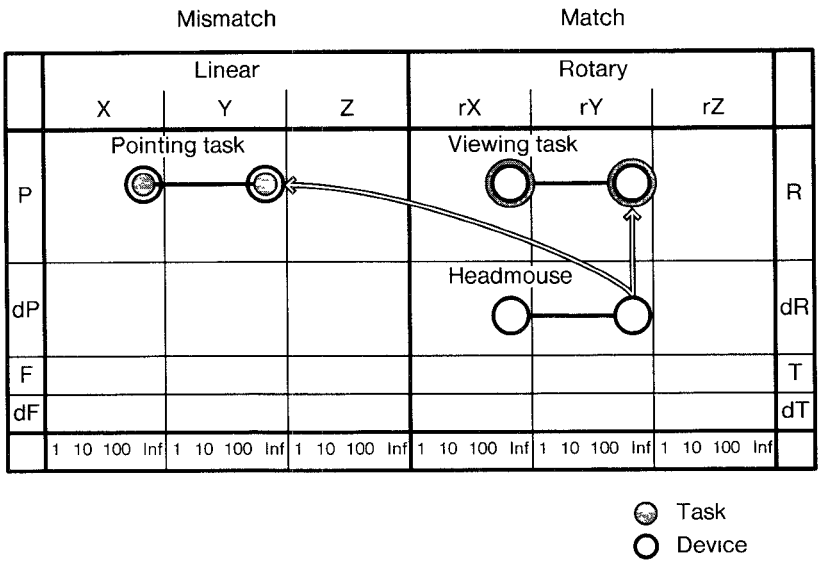

Fig. 11. Analysis of a virtual head movement device based on the headmouse. According to the figure, the headmouse, with precision slightly larger than a word, is precise enough for viewing but not for pointing. Therefore, these two tasks need to be broken apart in the design if the headmouse is to be used.

tasks. For purposes of the calculation, we have assumed that moving the virtual head to look around in the virtual 3-D world is roughly equivalent in difficulty to pointing to a paragraph, and that pointing to objects is roughly equivalent to pointing to a character, which is also the easiest hard task for the mouse. Since the task circles contain the filled circles projected from the mouse, it is clear that the mouse is precise enough for both tasks.

The headmouse seems like an obvious device for this application, but in fact, our analysis shows that it is appropriate for only half of the tasks. When we make the equivalent diagram for the headmouse in Figure 11, we see that the headmouse is matched to the viewing task, but it is not precise enough for the pointing task. If we want to use the headmouse, we should separate out the pointing task and put it on some other, more precise device. Incidentally, a similar analysis for editing would suggest that the headmouse is not very good for text editing because the transducer has been connected to a muscle with too little bandwidth for the precision of editing subtasks, such as pointing to a character.

4. CONCLUSION

In this paper we have illustrated a way of systematizing knowledge about input devices. We have provided a method for helping to generate points in the design space. We have shown how designs can be critiqued in terms of expressiveness and effectiveness, and have used two effectiveness metrics,

footprint and bandwidth, to illustrate how regions of the space can be systematically analyzed.

The virtue of the present analysis is that it allows one to generate and calculate the consequences of interesting regions of the design space. This allows one to concentrate prototyping and engineering efforts in areas where analysis shows promising possibilities. For example, calculation shows that the headmouse is not likely to be successful for text editing for the fundamental reason that, despite the fact that the headmouse has an impressive transducer, this transducer is attached to a muscle group that does not have enough bandwidth to place the tasks of interest in the easy region. This analysis could be done in a short time on the back of an envelope and could make a strong prediction about whether an expensive research, development, and marketing effort is justified. One member of the lab refused to believe these calculations, claiming that the difference in pointing time between a headmouse and a mouse was purely a matter of users having had more practice with the mouse (this argument is obviously faulty because learning mainly affects the intercept of Fitts's law, not the slope). To prove his point, he was determined to spend eight hours the next day using the headmouse for text editing until he became as skilled with it as with a mouse. Not only did the differences continue to hold (as predicted), but he was also unable to move his neck for three days, much to the bemusement of his colleagues. We have purposely used a highly approximative style of calculation to illustrate the utility of rapid, simple analysis based on simple assumptions for gaining insight into the design space. More refined analyses could be pursued, perhaps in connection with an empirical study to test and advance these conclusions. But, even at the present level of approximation, the analyses have force, as our colleague discovered.

The present analysis also suggests a promising direction for developing a device to beat the mouse by using the bandwidth of the fingers. In fact, this analysis has been used by one of the authors to design novel input devices.

The design of human–machine interfaces, it has been argued, can be at least partially viewed as the design of artificial languages for communicating between human and machine. This paper has analyzed the basic semantics of one component of such artificial languages: input devices. Mackinlay [21, 22], as noted, has made a similar analysis of graphical presentations of data—communication in the other direction, from machine to human. Both studies have tried to work out a systematic description of the semantics of the messages to be communicated between human and machine. There are, of course, many additional complexities to human–machine communication that have not been dealt with (e.g., feedback, turn-taking, or animation), but the techniques used in these studies seem likely to be useful for future systematic treatments of other areas. In particular, it allows us to accumulate theory, empirical results, and design in a coherent framework.

Carroll [10] has called for a paradigm in human–computer interaction termed *usability-innervated invention* in which the analysis of human performance participates directly in the creation of computer-based artifacts, either through the explicit attempt to have new artifacts embody psychological

claims ("psychology as the mother of invention") or through the analysis of the tasks the artifacts aid. We think morphological analyses of design spaces can play a role in such a paradigm and that the analyses can be used to integrate the results from several disciplines. For example, often we expect the operators for the generation of the design space to be computer-science oriented and the development and application of theories (or empirical analyses) for testing the space to be psychological. But Bertin's operators for generating the design space were largely semiotic and perceptual. In any case, the emphasis is on discovering the structure of the design space and its consequences.

REFERENCES

1. ALBERT, A. The effect of graphic input devices on performance in a cursor positioning task. In *Proceedings of the Human Factors Society-26th Annual Meeting* (Seattle, Wash., Oct. 25-28, 1982). Human Factors Society, Santa Monica, Calif., 1982, pp. 54-58.
2. ANSON, E. The device model of interaction. *Comput. Graph. 16*, 3 (July 1982), 107-114. (Also, *SIGGRAPH '82 Proceedings.*)
3. APPLE COMPUTER. *Technical Introduction to the Macintosh Family.* Addison-Wesley, Reading, Mass., 1987.
4. BAECKER, R. M., AND BUXTON, W., EDS. *Readings in Human-Computer Interaction: A Multidisciplinary Approach.* Kaufmann, Los Altos, Calif., 1987, pp. 357-365.
5. BERTIN, J. *Semiology of Graphics.* University of Wisconsin Press, Madison, Wis., 1983. (Translation by W. J. Berg of 1973 edition of Semiologie graphique.)
6. BLESER, T. W., AND SIBERT, J. Toto: A tool for selecting interaction techniques. In *Proceedings of User Interface Software and Technology* (Snowbird, Utah, Oct. 3-5, 1990). ACM, New York, 1990, pp. 135-142.
7. BUXTON, W. Lexical and pragmatic considerations of input structures. *Comput. Graph. 17*, 1 (Jan. 1983), 31-37.
8. CARD, S. K. Human factors and artificial intelligence, In *Intelligent Interfaces: Theory, Research and Design*, P. A. Hancock and M. H. Chignell, Eds. Elsevier Science Publishers B.V., North-Holland, Amsterdam, 1989, pp. 270-284.
9. CARD, S. K., ENGLISH, W. K., AND BURR, B. J. Evaluation of mouse, rate-controlled isometric joystick, step keys, and text keys for text selection on a CRT. *Ergonomics 21*, 8 (Aug. 1978), 601-613.
10. CARROLL, J. M. Evaluation, description and invention: Paradigms for human-computer interaction. In *Advances in Computers*, vol. 29, M. C. Yovits, Ed. Academic Press, San Diego, Calif., 1989, 47-77.
11. CASNER, S. M. A task-analytic approach to the automated design of graphic presentations. *ACM Trans. Graph. 10*, 2 (Apr. 1991), 111-151.
12. ENGLISH, W. K., ENGELBART, D. C., AND BERMAN, M. L. Display-selection techniques for text manipulation. *IEEE Trans. Hum. Factors Electron. HFE-8*, 1 (March 1967), 5-15.
13. EPPS, B., SNYDER, H., AND MUTOL, W. Comparison of six cursor devices on a target acquisition task. In *Proceedings of the Society for Information Display* (San Diego, Calif., May 6-8, 1986). Society for Information Display, 1986, pp. 302-305.
14. FITTS, P. M. The information capacity of the human motor system in controlling amplitude of movement. *J. Exper. Psychol. 47*, 6 (June 1954), 381-391.
15. FOLEY, J. D., WALLACE, V. L., AND CHAN, P. The human factors of computer graphics interaction techniques. *IEEE Comput. Graph. Appl. 4*, 11 (Nov. 1984), 13-48.
16. JELLINEK, H., AND CARD, S. K. (1990). Powermice and user performance. In *CHI'90 Conference Proceedings* (Seattle, Wash., Apr. 1-5, 1990). ACM Press, New York, 1990, pp. 213-220.

17. KARAT, J., MCDONALD, J., AND ANDERSON, M. A comparison of selection techniques: Touch panel, mouse and keyboard. In *Human-Computer Interaction—INTERACT 84*, B. Shackel, Ed. Elsevier North-Holland, Amsterdam, 1985, pp. 189-193.
18. LANGOLF, G. D. Human motor performance in precise microscopic work. Ph.D. dissertation, Dept. of Indsutrial Engineering, Univ. of Michigan, Ann Arbor, Mich., 1973. (Also published by the MTM Association, Fairlawn, N J., 1973.)
19. MACKENZIE, I. S. Fitts' law as a performance model in human-computer interaction. Ph.D. thesis, Dept. of Education, Univ. of Toronto, Ontario, Canada, 1991.
20. MACKENZIE, I. S. Fitts' law as a research and design tool in human-computer interaction. *Hum -Comput. Interaction.* In press.
21. MACKINLAY, J. Automatic design of graphical presentations. Ph.D. dissertation Computer Science Dept., Stanford Univ., Calif., 1986. (Also Tech. Rep. Stan-CS-86-1038.)
22. MACKINLAY, J. Automating the design of graphical presentations of relational information. *ACM Trans. Graph. 5*, 2 (Apr. 1986), 110-141.
23. MACKINLAY, J. D., CARD, S. K., AND ROBERTSON, G. G. A semantic analysis of the design space of input devices. *Hum.-Comput. Interaction 5*, 2-3 (1990), 145-190.
24. OLSEN, D. R., AND HALVERSON, B. W. Interface usage measurements in a user interface management system. In *Proceedings of the ACM SIGGRAPH Symposium on User Interface Software* (Banff, Alberta, Canada, Oct. 17-19, 1988). ACM, New York, 1988, pp. 102-108.
25. OLSEN, D. R., ET AL. ACM SIGGRAPH workshop on software tools for user interface management. *Comput. Graph. 21*, 2 (April 1987), 71-147.
26. PENFIELD, W., AND RASMUSSEN, T. *The Cerebral Cortex of Man: A Clinical Study of Localization of Function.* Macmillan, New York, 1990.
27. PFAFF, G. E.. *User Interface Management Systems.* Springer-Verlag, New York, 1985.
28. RADWIN, R. G., VANDERHEIDEN, G. C., AND LIN, M. A method for evaluating head-controlled computer input devices using Fitts' law. *Hum. Factors 32*, 4 (Aug. 1990), 423-438.
29. ROBERTSON, G. G. , CARD, S. K., AND MACKINLAY, J. The cognitive coprocessor architecture for interactive user interfaces. In *Proceedings of the ACM-SIGGRAPH Symposium on User Interface Software and Technology* (Williamsburg, Va., Nov. 13-15, 1989). ACM, New York, 1989, pp. 10-18.
30. ROTH, S. F., MATTIS, J., AND MESNARD, X. Graphics and natural language as components of automatic explanation. In *Intelligent User Interfaces*, J. Sullivan and S. Tyler, Eds. ACM Press, New York, pp. 207-239.
31. SHERIDAN, T. B. Supervisory control of remote manipulators, vehicles and dynamic processes: Experiments in command and display aiding. *Adv. Man-Machine Syst. Res. 1* (1984), 49-137.
32 SIEWIOREK, D., BELL, G., AND NEWELL, A. *Computer Structures.* McGraw-Hill, New York, 1981.
33. TANNER, P. P., AND BUXTON, W. A. S. Some issues in future UIMS development. In *User Interface Management Systems*, G. E. Pfaff, Ed. Springer-Verlag, New York, 1985, pp. 67-79.
34. VAN DEN BOS, J Abstract interaction tools: A language for user interface management systems. *ACM Trans. Program. Lang. Syst. 10*, 2 (July 1988), 215-247.
35. WELFORD, A. T. *Fundamentals of Skill.* Methuen, London, 1968.
36 WHITEFIELD, D., BALL, R., AND BIRD, J. Some comparisons of on-display and off-display touch input devices for interaction with computer generated displays. *Ergonomics 26*, 11 (1983), 1033-1053.
37. ZWICKY, F. The morphological approach to discovery, invention, research, and construction. In *New Methods of Thought and Procedure*, F. Zwicky and A. G. Wilson, Eds. Springer-Verlag, New York, 1967, 273-297.