THE DESIGN SPACE OF INPUT DEVICES

Stuart K. Card, Jock D. Mackinlay, and George G. Robertson

Xerox Palo Alto Research Center
3333 Coyote Hill Road
Palo Alto, CA 94304
415-494-4362, card.pa@xerox.com

ABSTRACT

A bewildering variety of devices for communication from humans to computers now exists on the market. In order to make sense of this variety, and to aid in the design of new input devices, we propose a framework for describing and analyzing input devices. Following Mackinlay's semantic analysis of the design space for graphical presentations, our goal is to provide tools for the generation and test of input device designs. The descriptive tools we have created allow us to describe the semantics of a device and measure its *expressiveness*. Using these tools, we have built a taxonomy of input devices that goes beyond earlier taxonomies of Buxton & Baecker and Foley, Wallace, & Chan. In this paper, we build on these descriptive tools, and proceed to the use of human performance theories and data for evaluation of the *effectiveness* of points in this design space. We focus on two figures of merit, footprint and bandwidth, to illustrate this evaluation. The result is the systematic integration of methods for both generating and testing the design space of input devices.

KEYWORDS: Input devices, semantics, design knowledge systematization.

INTRODUCTION

Human-machine interface technology has 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 exist on the market, including typewriter keyboards, mice, headmice, pen and tablets, dialboxes, Polhemus cubes, gloves, and body suits. Given an abundance of designs, most engineering disciplines proceed to organize these designs by developing abstractions, theories, or other organizing principles in order to get a view of the design space behind these and potentially other designs (e.g., [21]). Previous work on human-machine input devices has provided three lines of development in this area: toolkits, taxonomies, and performance studies.

Toolkits. User interface toolkits or user interface management systems help with a wide range of problems including the construction, runtime execution, and post-runtime analysis of a user interface [22]. They may help systematize input device knowledge by providing a library of pre-built input device modules [18, 16], architecture and specification techniques for combining these modules [2, 23], or post-processing analysis tools [17]. Sometimes, as in Anson [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 in performing.

Taxonomies. Two recent attempts have been made to provide abstractions that systematize the design space of input devices. Foley, Wallace, and Chan [10] 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, 3] have 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, and 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.

Performance studies. Several studies have been made of the performance of different pointing devices. English & Englebart [7] studied several devices and found the mouse the fastest device. Card, English, & Burr [5] confirmed these empirical results and discovered that pointing speed with the mouse is governed by Fitts's Law [9] with a bandwidth similar to that of the hand. Subsequent studies have empirically compared speed and preference of various devices ([10, 1, 8, 11, 25]. Unfortunately these have not always agreed and most studies have not attempted to disentangle task, subject, and human performance variables.

ANALYTIC DESIGN FRAMEWORK

In order to understand how the above results could be accommodated in a single systematic framework, it is useful to consider the role of input devices in human-machine communication. An input device is part of the means used to engage in dialogue with a computer or other machine. The dialogue is not, of course, in natural language, but is conducted in ways peculiarly suited to interaction between human and machine. Unlike human-human conversation, the dialogue is between fundamentally dissimilar agents–both in terms of perception and processing. Furthermore it takes place under conditions (e.g., the persistance 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. Instead of words, 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 [13, 14], in work on the automatic generation of displays, suggested that each display could be thought of as a sentence in such a language and that such sentences could be analyzed as to their ability to transmit an intended semantic meaning from the machine to the user. This analysis has direct consequences for the design of machines. Semantic theories provide the means by which the design space can be generated. Human performance studies provide the means by which design points in the space can be tested. We can use this basic approach as a means for systematizing knowledge about human interface technology, including the integration of theoretical, human performance, and artifact design efforts. In particular, we can use this approach to integrate knowledge gained from toolkit, taxonomy, and human performance literature.

In an earlier paper [15], we addressed the semantic analysis of input devices and used this to generate the design space of input devices. In this paper, we build on the results from the earlier paper and proceed to the use of human performance theories and data for the evaluation of points in this design space. The result is the systematic integration of methods for both generating and testing the design space.

GENERATING THE DESIGN SPACE

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 [6]:

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.

Primitive Movement Vocabulary

We begin with the observation inspired by Baecker and Buxton [3] that:

> *Basically, an input device is a transducer from the physical properties of the world into logical values 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,

	Linear	Rotary
Position		
Absolute	Position P	Rotation R
Relative	Movement dP	DeltaRotation dR
Force		
Absolute	Force F	Torque T
Relative	DeltaForce dF	DeltaTorque dT

Figure 1. Physical properties sensed by input devices.

- **R** is a resolution function that maps 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).

Figure 1 lists the various manipulation operators possible for an input device. They are an extension of the physical properties suggested by Baecker and Buxton [3]. They represent all combinations of linear and rotatary, absolute and relative, position and force. Although other input devices are possible (based, say on speech or heat), virtually all input devices use some combination of these properties.

Figure 2 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 and maps using the identify operator from an input domain set of 0 to 270 degrees into the same set. The selector knob, on the other hand, maps from the set consisting of 0 to 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 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.

Composition Operators

The example in Figure 2 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:

- Merge composition
- Layout composition
- Connect composition

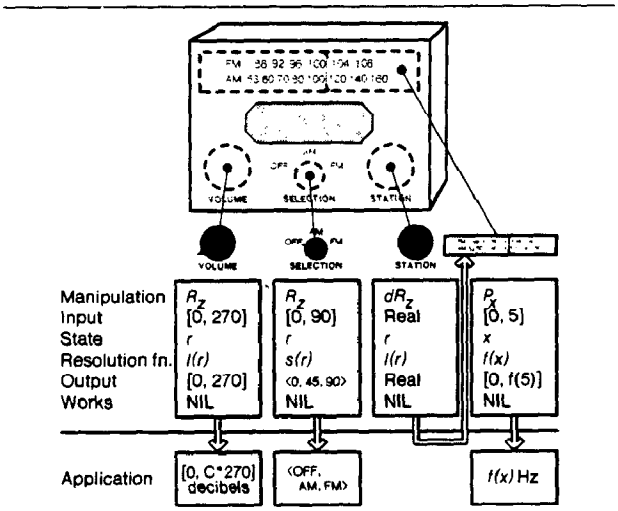

Figure 2. An analysis of a simple radio that illustrates various input devices. Two rotational devices are connected directly to the application. The third rotational device is connected to a positional device, which is then connected to the application.

Merge composition is the combining of two devices such that the resulting input domain set is the cross product of the input domains of the two devices. A mouse, for example, 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. For example, the three buttons of a mouse and the XY sensor 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 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 visualization of this space in Figure 3. This is our equivalent to Foley, Wallace, & Chan's [10] and Buxton's [3] classification. A 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 (dotten line) or merge composition (black line).

We have plotted the devices of our radio example and the mouse on this diagram to illustrate its use. The radio volume knob is in the cell in Figure 3 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 se-

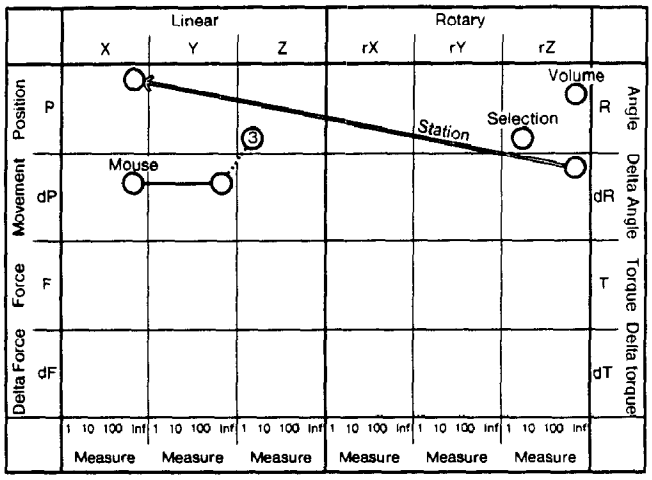

Figure 3. This diagram describes an input device taxonomy that is based on the analysis presented in this paper. Circles are used to indicate that a device senses one of the physical properties shown on the vertical axis along one of the linear or rotary dimensions shown on the horizontal axis. For example, the volume circle indicates a device that senses angle around the z axis. The position in a column indicates the number of values that are sensed (that is, the measure of the domain set). For example, the selection circle represents a discrete device. Lines are used to connect the circles of composite devices. The double line represents a connection composition, such as for the radio station composite device. 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 buttons on a mouse, which are represented by a circle with a 3 in it (since they are identical devices).

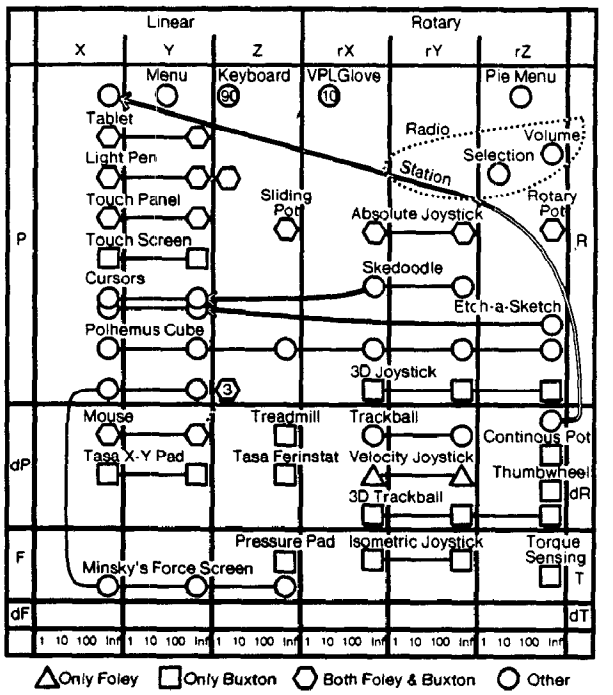

Figure4. A broad range of input devices plotted on the taxonomy. Devices previously classified by Foley [8] and Buxton [3,2] are indicated by triangles, squares, and hexagons. Hexagons indicate devices included in both previous taxonomies. Other devices, indicated by circles, include the radio devices described previously and some unusual devices to demonstrate the generality of the the taxonomy.

lection knob is similar, but it 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 says that the mouse is a layout composition of four devices: one device which 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 indicate 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, & Chen and Buxton (see Figure 4). With the exception of voice, we have been able to position on the diagram all of the devices considered so far. Furthermore, it is possible by placing circles in various cells of the diagram to generate potential new devices. Of course many of these devices might not be good ones, 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 ones not yet invented. In particular, we haved used our model of the input device design space to help design novel egocentric motion devices for virtual 3D environments [15, 19].

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 now turn to testing it. Following Mackinlay [13, 14], 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).

Expressiveness

There are several sorts of expressiveness problems that can arise with input devices. One sort of 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.

For example, if the user wishes to convey the meaning to the system "Select point x=105, y=32", but the device he is using has a resolution of 1/4 inch (as for some touch panels), then he will not be able to express his requests exactly and there will be some loss of expressiveness–serious or not depending on the situation.

Effectiveness

More interesting for input devices is effectiveness. Several figures of merit are possible:

- Pointing speed (really device bandwidth)
- Pointing precision
- Errors
- Time to learn
- Time to grasp the device
- User preference
- Desk footprint
- 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. Let us illustrate two of these: footprint and bandwidth.

Footprint

An input device requires a certain amount of space on a desk. Since a workspace, such as a desk, is a small, finite resource, smaller footprints are usually better. The actual amount of space required to operate a device depends on the sequence of actions in an application. But for design purposes, we can index this by taking an extreme application parameter manipulation, then mapping this backward through the input device to determine what input device manipulation would be required to make this change in the application parameter.

As an example, let us compare the relative desk footprint of different input devices for pull-down menus on the original Macintosh (12 inch screen) and the SuperMac (19 inch screen). The mouse must be able to move from any point of 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. For various devices, we can estimate this, as below.

Device	*Footprint (square inches)*	
	19" Screen	*12" Screen*
Mouse (C:D = 1:2)	43	4.1
Tablet (C:D = 1:1)	173	69
Light Pen	0	0
Touch Pad	0	0
Trackball (2"x2")	4	4
Rotary Pots	0	0
Joystick (2"x2")	4	4

Figure 5 plots the sensors for these devices, circle size proportional to the area of the footprint. Several facts about the design space of potential devices are evident

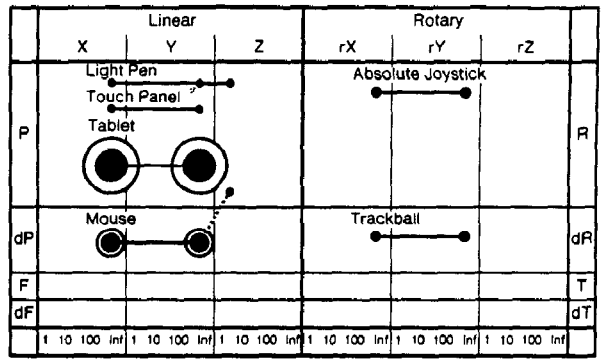

Figure 5. Footprint of input devices for Macintosh pull-down menus. Filled in circles the describe the device footprints for the 12 inch screen. The white circles describe the device footprints for the 19 inch screen.

from the diagram: (1) The tablet and the mouse are very expensive in footprint relative to other devices. (2) Going from 12 inch to 19 inch displays causes a major increase in footprint size for the mouse and tablet (but has no effect on the other devices). This increase in footprint size might profitably cause designers to consider whether the original pull-down menu design is the most appropriate for the larger screen.

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 it is not quite accurate to characterize input devices in terms of speed of use. The time to do something is actually a joint product of all these elements in our model: the human, the device, and the application. For the moment, we restrict ourselves to tasks which involve pointing to 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 human and device bandwidth and application precision requirements:

1. [Human] The bandwidth of the human muscle group to which the input device transducer is attached.
2. [Device] The resulting bandwidth of the input device.
3. [Application] The precision requirements of the task to be done with the device.

Bandwidth of the human muscles. Figure 6 shows data from an experiment by Langolf [12]. Subjects in the experiment performed the Fitts dotting task [9] under the microscope. Movements of different muscles caused

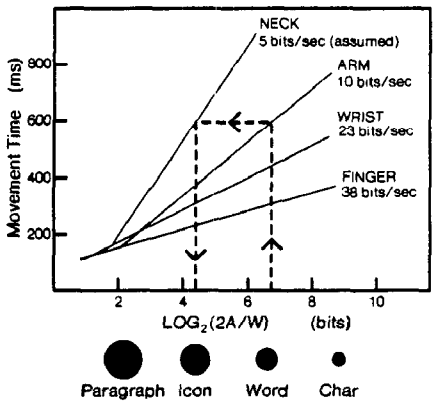

Figure 6. Movement time as a function of Fitt's Index of Difficulty from Langolf's study of movements under a microscope (Langolf, 1973, p.145). The older form of Fitt's Law (Fits, 1954) is used (A = Distance, W = Width of Target). Data was of three subjects for 125-200 movements.

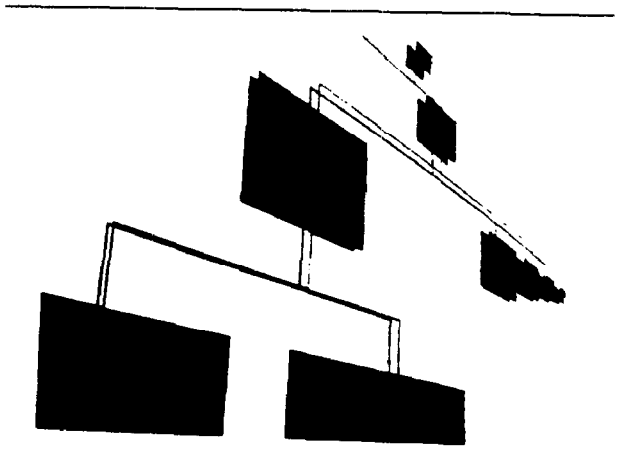

Figure 7. An application of simulated head movement for viewing an organization chart.

different Fitts's Law slopes suggesting different muscle bandwidths. We can use Figure 6 as a rough index for the bandwidth of different parts of the body.

Bandwidth of Input Devices. An approximate measure of the difficulty of many input activities can be estimated by selecting one, or a few, extreme subtasks as design benchmarks (the way a heating engineer would take a $-20°$F winter day, a $100°$F summer day, and a $70°$F typical day as design benchmarks to run calculations on). For example, in text editing, we might select the subtask of pointing to a character (because it is probably the hardest pointing task for editing) as our design benchmark.

We can index the difficulty of a task by using Fitts's Index of Difficulty [9], which, in Welford's [24] reformulation is given as:

$$I_D = \log_2(D/S + .5) \text{ bits},$$

where D is the distance and S is the size of the target, that is, the precision with which pointing has to be done. The time required to do one of these tasks is given by:

$$MovementTime = Const + (1/B) * (I_D) \text{ sec},$$

where B is the bandwidth of the device. The quantity B is often given for Fitts's Law in terms of its reciprocal.

Values of the Index of Difficulty for some common graphics subtasks are located on the abscissa of Figure 6. We now go one step farther. Using the bandwidth of the arm (which is the main muscle group involved in moving the mouse) as a reference, we can compare device bandwidth and task precision requirements. We take an application task such as pointing to a character on a display. This task has an associated I_D index of difficulty number as indicated in Figure 6 on the abscissa. Moving vertically upward, we intersect the line indicating the time required by an arm to point to a target of this difficulty. We can then move horizontally to find the Fitts's Law line associated with the muscles connected to the transducer of another input device. From this intersection, we can move back down to the abscissa to find the Index of Difficulty value that this device could point to in the same time that the arm could point to a character. As a result, we say that an input device will be matched with the precision requirements of a pointing task, if the pointing device (as indicated by the relative size of the circles) is as good as the pointing hand. This calculation has been used to set the size of circles that characterize both device bandwidths and application precision requirements in the following example.

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

Suppose the user is located in a virtual 3D environment (Figure 7). 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 contains a fixed circle cursor on its center. Moving the mouse thus moves the world, keeping the cursor fixed. This simulates moving the user's direction of gaze around in this virtual world. Pressing a button on the mouse selects whatever object is inside the circle.

An interesting alternative to the mouse for this application is the "headmouse" device. A headmouse has a pair of 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 in the appropriate way.

It seems like an obvious device for this application, but analysis shows that it is only appropriate for half of the task. Figure 8 shows the set of connected devices implied by the above description of the use of the mouse. The mouse is connected to a virtual head and then to

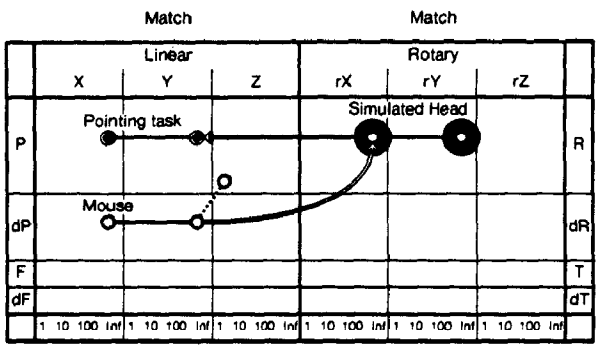

Figure 8. Bits per second analysis of mouse for viewing and pointing.

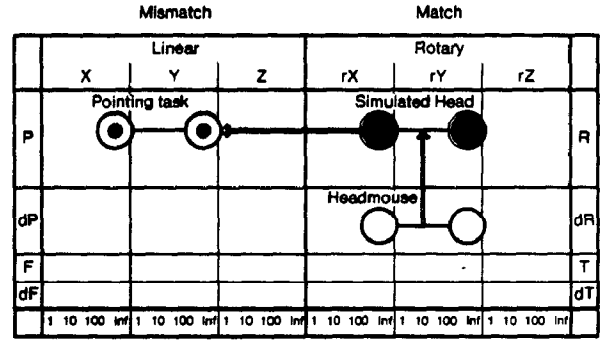

Figure 9. Bits per second analysis of headmouse for viewing and pointing.

a virtual cursor on a surface in the virtual 3D world. In the figure, the circles of all these virtual devices are drawn to indicate the precision with which pointing can be done–a smaller circle indicates tighter precision computed according to Figure 6. Filled circles represent the precision requirements of the application. In this task there are essentially two application precision requirements: loose precision for controlling the virtual head looking around and tight precision for pointing at a position on the whiteboard. A white circle and a filled circle of the same size indicate a fit. A filled circle larger than a white circle indicates that the device has more precision than needed. But a filled circle smaller than a white circle indicates that the task requires more pecision than is being provided.

In Figure 8, we see that the mouse provides more than enough precision for moving the head and is a match for pointing to the whiteboard. But when we make the equivalent diagram for the headmouse in Figure 9, we see that the headmouse is matched to the task of looking around, 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. Incidently, a similar analysis for editing would suggest that the headmouse is not a very good idea 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.

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 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 [13, 14], 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.

It is interesting to note that somewhat similar methods have been used in other areas of engineering, for example, to design jet engines [26]. A technique similar to ours (called "morphological search") was used to define several design spaces and, it is claimed, led to a number of surprising and novel inventions.

References

[1] Albert, A., (1982). The effect of graphic input devices on performance in a cursor positioning task. *Proceedings of the Human Factors Society-26th Annual Meeting*, pp. 54-58.

[2] Anson, E. (1982). The device model of interaction. *Computer Graphics, 16*(3), 107-114. Also, SIGGRAPH '82 Proceedings.

[3] Baecker, R. M., & Buxton, W. (Eds.), (1987). *Readings in human-computer interaction: A multidisciplinary approach.* Los Altos, CA: Morgan Kaufmann, 357-365.

[4] Buxton, W. (1983). Lexical and pragmatic considerations of input structures. *Computer Graphics 17*(1), 31-37.

[5] Card, S.K., English, W.K., & Burr, B.J. (1978). Evaluation of mouse, rate-controlled isometric joystick, step keys, and text keys for text selection on a CRT. *Ergonomics 21,* 601-613.

[6] Card, S.K. (1989). Human factors and artificial intelligence. In P.A. Hanccock & M.H. Chignell (Eds.), *Intelligent interfaces: theory, research and design.* Elsevier Science Publishers B.V. (North-Holland).

[7] English, W. K., Engelbart, D. C., & Berman, M. L. (1967). Display-selection techniques for text manipulation. *IEEE Transactions on Human Factors in Electronics HFE-8,* 5-15.

[8] Epps, B., Snyder, H., & Mutol, W. (1986). Comparison of six cursor devices on a target acquisition task. *Proceedings of the Society for Information Display,* pp. 302-5.

[9] Fitts, P. M. (1954). The information capacity of the human motor system in controlling amplitude of movement. *Journal of Experimental Psychology 47,* 381-391.

[10] Foley, J. D., Wallace, V. L., & Chan, P. (1984). The human factors of computer graphics interaction techniques. *IEEE Computer Graphics & Applications 4*(11), 13-48.

[11] Karat, J., McDonald, J., & Anderson, M. (1985). A comparison of selection techniques: touch panel, mouse and keyboard. In B. Shacket (Ed.), *Human-Computer Interaction - INTERACT 84,* pp. 189-193.

[12] Langolf, G. D. (1973). Human motor performance in precise microscopic work. PdD dissertation. University of Michigan. Also published by the MTM Association, Fairlawn, New Jersey, 1973.

[13] Mackinlay, J. (1986a). Automatic design of graphical presentations. PhD dissertation. Computer Science Dept., Stanford University. Also Tech. Rep. Stan-CS-86-1038.

[14] Mackinlay, J. (1986b) Automating the design of graphical presentations of relational information. *ACM Transactions on Graphics,5*(2, April), 110-141.

[15] Mackinlay, J.D., Card, S.K., & Robertson, G.G. (in press). A semantic analysis of the design space of input devices. To appear in *Human-Computer Interaction,* Lawrence Erlbaum.

[16] Olsen, D. R., et al. (1987). ACM SIGGRAPH workshop on software tools for user interface management. *Computer Graphics 21*(2), 71-147.

[17] Olsen, D. R., & Halversen, B. W. (1988). Interface usage measurements in a user interface management system. *Proceedings of the ACM SIGGRAPH Symposium on User Interface Software.* Banff, Alberta, Canada, October, 19988, 102-108. New York: ACM

[18] Pfaff, G. E. (1985). *User interface management systems.* New York: Springer-Verlag.

[19] Robertson, G. G., Card, S. K. & Mackinlay,J. (1989). The cognitive coprocessor architecture for interactive user interfaces. *Proceedings of ACM Symposium on User Interface Software & Technology.* Williamsburg, VA, in press. New York:ACM.

[20] Sheridan, T.B. (1984). Supervisory control of remote manipulators, vehicles and dynamic processes: experiments in command and display aiding. *Advances in Man-Machine System Research, 1,* 49-137, JAI Press.

[21] Siewiorek, D., Bell, G., & Newell, A. (1981). *Computer structures.* New York: McGraw-Hill.

[22] Tanner, P. P. & Buxton, W. A. S. (1985). Some issues in future UIMS development. In G. E. Pfaff (ed.), *User interface management systems,* (pp. 67-79), New York: Springer-Verlag.

[23] van den Bos, J. (1988). Abstract iteraction tools: a language for user interface management systems. *ACM Transactions on Programming Languages and Systems, 10*(2), 215-247.

[24] Welford, A. T. (1968). *Fundamentals of skill.* London: Methuen.

[25] Whitefield, D., Ball, R., & Bird, J. (1983). Some comparisons of on-display and off-display touch input devices for interaction with computer generated displays. *Ergonomics, 26*(11), pp. 1033-1053.

[26] Zwicky, F. (1967) The morphological approach to discovery, invention, research, and construction. In F. Zwicky & A. G. Wilson (Eds.), *New methods of thought and procedure.* Springer-Verlag (New York).