LECTURE 2

EXAMPLES OF A MODEL-BASED HUMAN FACTORS

In the last lecture, we presented a model, the Model Human Processor, intended to summarize the most important limitations and capablities of humans. We shall now proceed to more extensive examples in the human-computer interaction domain.

Pointing Devices

Let us begin by analyzing the pointing devices to be used with a display. Our first consideration is how long it takes the hand itself to move to some target.

> PROBLEM. How long does it take to move the hand to a target S cm wide whose center is D cm distant? (See Fig. 10.)

SOLUTION. According to the Model Human Processor, the hand will make a series of micromovements to the target. Each micromovement will attempt to hit the target, but there will be some error each time. The time to reach the target will therefore be

(1)

$$T = nT_{step}$$

where T is the total time, T_{step} is the time for each micromovement, and n is the number of steps required.

Let X_i = be the distance remaining to the target after micromovment$_i$. Assume a constant error ε for each micromovement. Then,

$$X_0 = D$$

$$X_1 = \varepsilon X_0 = \varepsilon D$$

$$X_2 = \varepsilon X_1 = \varepsilon(\varepsilon D) = \varepsilon^2 D$$

The hand stops when it is within the target, or

$$X_n = \varepsilon^n D = \frac{S}{2}$$

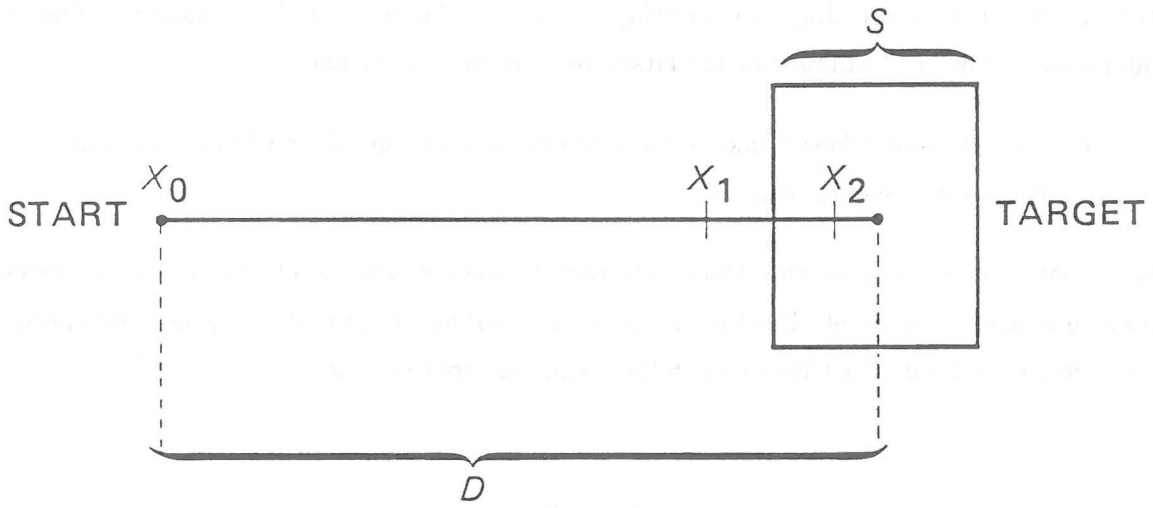

Fig. 10.

Solving for n gives:

$$n = -\frac{\log_2\left[\frac{2D}{S}\right]}{\log_2 \varepsilon}$$

We use this last term to substitute for n in Eq. (1) to obtain,

(2)

$$T = K \log_2\left[\frac{2D}{S}\right],$$

$$\text{where } K = \frac{-T_{step}}{\log_2 \varepsilon}$$

Eq. (2) is known as Fitts's Law. We can estimate the constants from first principles:

$$\varepsilon = \text{absolute discrimination} = 1/7$$

$$T_{step} = \tau_P + \tau_C + \tau_M = 240 \text{ msec}$$

to obtain a value for K of

$$K = -240/-2.8 = 86 \text{ msec/bit} \quad \text{[from model]}.$$

This is similar to the values obtained empirically in the literature:

(3)

$$K = 100\,[50\sim120]\,\text{msec/bit} \quad \text{[from literature]}.$$

We now plot data (Fig. 11) from an experiment run to measure the time to point to targets of different distances and sizes with the "mouse" pointing device (a small box that rolls on a table to move a cursor). The pointing time for the mouse is plotted as a function of $\log_2(D/S + .5)$, a slight variant of the Fitts's Law expression in Eq. (2) that adds a small correction factor inside the log (Wellford, 1968). It can be seen that, indeed, the mouse pointing data is well fit by Fitts's Law. In this particular experiment (Card, English, and Burr, 1978), the mouse was compared to other devices: a joystick, step keys, entity keys. The mouse was best. The normal thing is just to make this comparison in line with the method of comparative experiments in human factors illustrated in the first lecture. But such comparisons are dangerous. If we do not know *why* the results come

MOUSE POSITIONING DATA

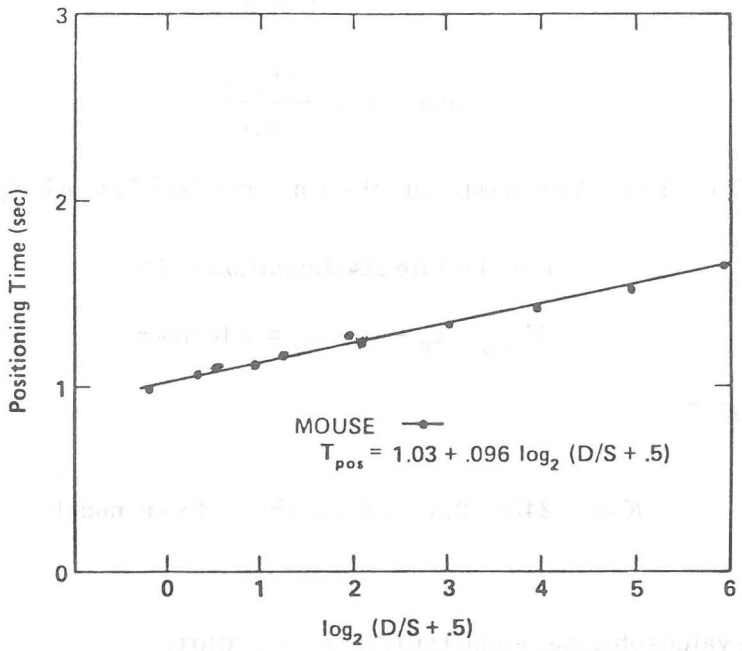

Fig. 11

out the way they do when we attempt to apply them in a new design, the results may not generalize.

Because we have a model and not just a comparative experiment, we can go farther. We know that the reason positioning time varies with size and distance is that positioning time for the mouse follows Fitts's Law. We now take note that the coefficient for Fitts's Law in Fig. 11 is about the same as the .1 sec/bit coefficient that derives from numerous tasks in the literature (Eq. 3) as well as our derivation from the Model Human Processor for hand pointing. This means that the limitation on pointing speed is in the human eye-hand system, not in the mouse. This in turn means that we are unlikely to make another device that will be faster (at least with the same muscles). Thus, using a model-based human factors we not only have more confidence in our results, we can also gain insight that helps us understand whether there are opportunities of improving the engineering of our subject device.

Maximum Mouse Velocity

> PROBLEM. A manufacturer wishes to build a system with a mouse. It would be both convenient and inexpensive if the system need cope with cursor speeds no greater than 50 cm/sec. Would that be enough?

SOLUTION. To solve this problem, we compute the average mouse velocity for the cycle on which it will be the maximum. That should be the first cycle, because the time is constant for each step and the largest distance is traveled on the first step:

$$V_{max} = \frac{X_0 - X_1}{\tau_P + \tau_C + \tau_M}$$

$$= \frac{D - \varepsilon D}{\tau_P + \tau_C + \tau_M}$$

$$= \left[\frac{1 - \varepsilon}{\tau_P + \tau_C + \tau_M} \right] D$$

When we substitute for the constants we get,

$$V_{max} = 4.9D \text{ cm/sec.}$$

That is, to find the maximum velocity of the cursor on the screen, take the maximum dustance moved (in cm) and multiply it by 4.9 to get the velocity in cm/sec. The longest distance to move the cursor is diagonally from one corner to another. The screen being considered had a diagonal of 35 cm. Hence, the fastest velocity expected would be,

$$V_{max} = (4.9)(35)$$

$$= 171 \text{ cm/sec}.$$

This number is more than a factor of three greater than the 50 cm/sec considered. The validity of this calculation has been checked and is in agreement with empirical results (Card, Moran, and Newell, 1983).

Text Editing

Now let us consider the analysis of a different sort of system: computer-based text editors. In this case, we begin with an application of the Rationality Principle from the Model Human Processor.

> PROBLEM. How long does it take an expert to modify text with a computer starting from a manuscript with marked modifications?

SOLUTION. The user will produce a goal-oriented method. We can analyze the user's behavior in terms of Goals, Operators, Methods available for meeting the goals, and Selection rules for deciding among alternative methods for the same goal. Such an analysis we call a GOMS analysis after the components in it. For example, a single editing task on the line-oriented editor POET looks like this:

```
GOAL: EDIT-MANUSCRIPT
    GOAL: EDIT-UNIT-TASK                repeat until no more unit tasks
        GOAL: ACQUIRE-UNIT-TASK         if task not remembered
            GET-NEXT-PAGE               if at end of manuscript
            GET-NEXT-TASK
        GOAL: EXECUTE-UNIT-TASK         if an edit task was found
            [select USE-QS-METHOD USE-LF-METHOD]
            GOAL: MODIFY-TEXT
```

THE KEYSTROKE MODEL

$$T_{total} = \text{Sum of } T_{unit\ task}$$

$$T_{unit\ task} = T_{acquisition} + T_{execute}$$

OPERATORS

K[text]	KEY-IN text	T_K = 0.20 sec
P[object]	POINT-TO object	T_P = 1.10 sec
H[device]	HANDS-TO device	T_H = 0.40 sec
M	MENTAL prepare	T_M = 1.35 sec
R(delay)	WAIT FOR delay	

RULES FOR LOCATION OF M OPERATOR

M only before decision point which cannot be chunked

1. First letter of a command name: M K[r]
2. First terminator of a variable argument string:

   K[word] M K[ESC], M K[ESC ESC] (1st terminator only)

3. Pointing to a command: M P[QUIT] K[#1]

Fig. 12

[select USE-S-COMMAND USE-M-COMMAND]

VERIFY-EDIT

Roughly, this notation says that that the user breaks up the task of editing the manuscript into subgoals of editing a unit task (more or less a single modification). To accomplish each of these he first acquires the task by turning to the marked-up manuscript then executes the task acquired. To execute the task, he has a selection among different methods. Finally, he verifies that the edit is correct. Models of this sort can be drawn in outline, as above, or in extended detail. They can be used to predict sequences of user actions and times for editing particular manuscripts. (See Card, Moran, and Newell, 1983). When tested against real data they capture a reasonable first-order approximation of editing behavior.

In the limit, as the user becomes more expert, we can give a more simplified description consisting of the basic motor operations plus some small amont of mental operation. Because this model has the fewest number of operators when given at the keystroke level of operation (other levels of aggregation are possible), it is called the Keystroke-Level Model. The operators of the Keystroke-Level Model are summarized in Fig. 12. Notice that we still need an operator for mental activity *M* because even with practice and expertise, not all of the perceptual and cognitive activity is overlapped by motor operations. The rules in Fig. 12 predicting the location of *M* operations can be reduced to the proposition that whatever operations can exist together in memory as a chunk will have their cognitive operations overlapped by motor operations, but extra time (coded as *M* operators) will be required between chunks.

As an example, we give an analysis using the Keystroke-Level Model for an a display-oriented text-editor that uses the mouse.

> PROBLEM. How long does it take to correct a mistyped word typed *n* words previously.

SOLUTION. In this particular editor, the correction is accomplished by typing control-W to erase words back from the end until the defective word is reached, retyping it, then retyping the text erased. We can list each of the steps involved and code them in terms of the operators of the Keystroke-Level Model as follows:

Method W:

1.	Press and hold CONTROL key	M K[CONTROL]
2.	Invoke Backword *n* times	*n*(.25M K[W])

3.	Type new word	5.5K[*word*]
4.	Retype destroyed text	5.5(n – 1)K

$$T_{execute} = (1 + .25n)t_M + (1 + 6.5n)t_K$$
$$= 1.6 + 2.16\ n \text{ sec}$$

This simplified model is a reasonably good predictor of performance time for experts (Fig. 13) and given that it is known what the sequence of operations will be. But when one is designing a system, that is just what one is designing.

This editor also comes with another method for accomplishing the same goal: The user stops typing, reaches for the mouse and points to the bad word, replaces it, then resumes typing. We can use the Keystroke-level Model to analyze this method giving us a basis on which to compre the two methods:

Method R:

1. TERMINATE TYPE IN	MK[ESC]
2. REPLACE BAD WORD	
2.1 Home hand onto mouse	H[MOUSE]
2.2 Point to target word	P[*word*]
2.3 Select word	K[MOUSEBUTTON2]
2.4 Home hand onto keyboard	H[KEYBOARD]
2.5 Invoke Replace command	MK[r]
2.6 Type new word	4.5K[*word*]
2.7 Terminate Replace	MK[ESC]
3.RESUME TYPING	
3.1 Home hand onto mouse	H[MOUSE]
3.2 Point to last input word	P[*word*]
3.3 Select it	K[MOUSEBUTTON2]
4. RE-ENTER TYPE-IN MODE	
4.1 Home hand onto keyboard	H[KEYBOARD]
4.2 Invoke Insert command	MK[i]

KEYSTROKE MODEL DATA

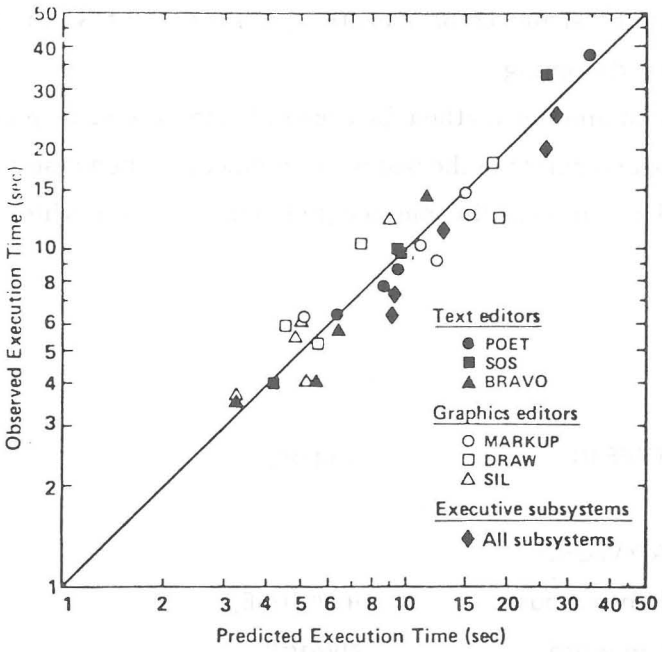

Fig. 13

$$T_{execute} = 4t_M + 10.5t_K + 4t_H + 2t_P$$
$$= 1.6 + 2.16\ n \text{ sec}$$

Because we have a model of the interaction we can make a parametric comparison (solid lines, Fig. 14). We see that there is a region of the parameter n in which each method is superior. Of course, we have ignored the relative frequencies with which users are likely to make each correct words for each value of n. Adding that to our analysis would be straightforward, but would complicate the presentation.

Now suppose we wanted to build a new command that would allow the user to move a cursor back to the word to be altered and then skip forward again without destroying the good text. In this new method, the user will type CONTROL-S to move the cursor back to the word to be changed and CONTROL-R to resume. The question is, can we tell before implementing anything about the usefulness of this command? Again we do an analysis:

Method S:

1.	Setup Backskip command	M K[CONTROL]
2.	Execute Backskip $n-1$ times	$(n-1)$(.25M K[s])
3.	Call Backward command	M K[w]
4.	Type new word	4.5K[*word*]
5.	Call Resume command	M K[CONTROL r]

$$T_{execute} = (3 + (n-1))\ t_M + (n + 7.5)\ t_K$$
$$= 5.8\ t_M + .62n\ t_K$$

From Fig. 14(dotted line), we can see that there is a range in which this new method, Method S, is superior to the other two methods. Of course, in the design of the whole system we still have to worry whether the addition of a new command makes the system harder to learn or more prone to errors, but at least on grounds of time alone we can say the results are encouraging. Because we have a model to manipulate, we are in a position to explore consequences of designs analytically and to understand benchmark experiments.

We could carry the analysis into a number of other areas (see Card, Moran, and Newell, 1983 for some of these), but these examples should suffice to give the general idea. Our point was to illustrate how a model-based approach could help us to make progress on the human side of computer science. We first used a simplified engineering model of the human to summarize basic

COMPARISON OF METHODS W, R AND S

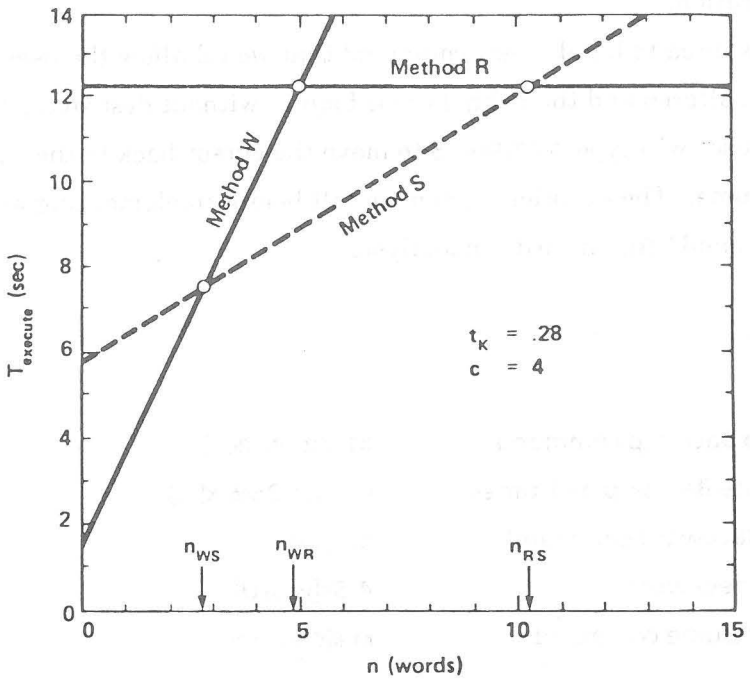

Fig. 14

human capabilities. We then pushed into other models, the Fitts's Law model of the mouse and the Keystroke-Level Model of expert system interaction that allowed us to pursue calculations and understanding at a level above that of the basic Model Human Processor model. There are issues that are beyond the beginings of this framework but that we are hopeful can be brought within it with more research: the use of visual displays, for example, and the behavior of novice and casual users. But our theme has been that if ergonomic/human-factors knowledge is model or theory-based, it is easier to be put to analytic use. This then is our vision for how the human part of computer science might be developed. In the final lecture, we shall consider explicitly the question of how one might fit such knowledge as it develops into the computer science curriculum.

DISCUSSION

In relation to the speaker's claim that the mouse is a device of "optimal efficiency", **Professor Randell** suggested that using a finger to point at the screen might be more efficient. **Dr. Card** explained that experiments have indeed indicated that pointing a finger at the screen is more efficient than sliding a mouse around the table. However, this result was not due to any differences between the finger and the mouse as such; it was due to the fact that the finger and the mouse were used in different ways in the experiments. In order to re-position an item on the screen, using the finger, firstly the finger was 'pointed' at the item, then taken off the screen, then finally pointed at the new position. On the other hand, the mouse was 'slid' from one place to another. 'Sliding' the mouse around the table and 'sliding' the finger around the screen, are both less efficient than either 'pointing' the finger to different places on the screen or using the mouse in a similar manner - i.e. 'pointing/banging' it around the table (an activity which tends to make the mouse's wheels drop off). Pointing is more efficient because the finger/mouse travels through the air, and the act of bringing it into contact with the screen/desk forces it to stop. On the other hand, when a mouse/finger is sliding across a table/screen, it is more difficult to make it stop - it must slow down before it stops.

Dr. Card mentioned some drawbacks of using ones finger to point at the screen:

1. Fingers are greasy.

2. Fingers are fat. This makes it difficult to point at a small item such as a 'full-stop' - i.e. fingers have a "low resolution".

3. You can't see the thing you're pointing at because it's covered by your finger.

4. Your arm soon gets tired when you're holding it upright to point at the screen.

Someone suggested that a horizontal screen, like a desk-top, would overcome the fourth drawback. Dr. Card claimed that it might be difficult to design such a screen. The CRT would protrude beneath the desk, which would make seating arrangements less than comfortable.

Certain members of the (predominantly male) audience expressed a degree of anxiety concerning the possible harmful side effects of radiation....

Dr. Card was asked whether any form of "amplification" is used; where a small movement of the mouse causes a greater movement of the cursor on the screen. The speaker replied that amplification is usually a factor of 2. He explained why amplification is an important issue; for example, an amplification factor of only 1, coupled with a large screen, would mean that the user has to move the mouse over long distances (probably knocking his/her

coffee over in the process).

Again in relation to Dr. Card's experiments indicating that the mouse is an "optimal device", the speaker was asked whether or not he included in his measurements the time it takes to make a selection, by doing one or more clicks of a mouse button. Dr. Card said that selection time had been taken into account in his experiments. He had found that clicking a mouse button is similar, in terms of efficiency, to typing a character on a keyboard.

Mr. Nichols wondered what effect mouse buttons have on the behaviour patterns of the user. **Dr. Card** replied that a source of "bloody struggle" is deciding on how many buttons to have on a mouse. The number can vary from 1 to 5 (one for each finger). Dr. Card had two points to make:

> Firstly that the best number of buttons is task and method dependent. For example, maybe the mouse is to be used in the task of editing. Different editing methods exist, and a particular method will require a certain number of buttons. In short, first one must chose the task to be performed using the mouse, and the method of performing the task. This will then dictate the number of mouse buttons required.
>
> Secondly, the best number of buttons depends on the type of user. For a naive user, perhaps a mouse with a single button would be ideal, because it is easy to learn how to use. However, only a single button means that the mouse has less functionality. A more expert user might require a mouse with more functionality, and thus more buttons. More learning is required in this case.

Professor McCarthy pointed out that using the mouse is generally part of some larger task. He wondered whether the nature of this larger task could affect how skilfully/efficiently the mouse is used.

Someone supplied an example: if the larger task also involves using the keyboard, then the hand must be moved back and forth between the keyboard and the mouse. The time involved in doing this might mean that the mouse is used less efficiently. Dr. Card explained that moving the hand between a keyboard and a mouse takes about the same amount of time as moving it between two sets of keys on the same keyboard (e.g. the normal typewriter keys and the digital pad). A mouse is fairly big and easy to get hold of. If the user has to keep getting hold of the mouse as a part of some larger task, this has little effect on how efficiently the mouse is used.

The speaker was asked if anyone has experimented with a pointing device that can be controlled by the feet, thus leaving the hands free to use a keyboard. Dr. Card replied that a pointing device controlled by the knee had been used.

On the subject of editing and methods of editing, **Professor McCarthy** wondered what difference it would make if the task of editing was secondary to the task of thinking. In such a case one would need an editing method that requires little thinking. Such methods are usually simple ones.

Regarding Dr. Card's experiments on people learning how to select items from a menu, the speaker was asked whether all items needed to be selected with equal frequencies. If some items need to be selected more frequently than others, what sort of differences would this make to the design of the menu and to the ease with which users could learn how to select items from the menu?

Dr. Card replied that in all his experiments, the menu items were used with equal frequencies. The more frequently an item is needed, the more quickly a user learns its position in the menu. Items could be ordered so that more frequently used ones are at the beginning of the menu. Of course, this means that the probable frequencies of use of each item must be known before the menu is designed. If the frequency of usage of different menu items changes over time, then this complicates matters further.