INTERNATIONAL JOURNAL OF COGNITIVE ERGONOMICS, 2000, *4*(2), 107–124

Designing Multifunction Displays: An Optimization Approach

Gregory Francis
Department of Psychological Sciences
Purdue University

ABSTRACT

A variety of modern devices use multifunction displays to present hierarchically ordered information. Users interact with the devices by pushing buttons to move through the hierarchy and access needed information. Several studies have shown that the mapping of hierarchy labels to buttons effects user performance. This article describes a quantitative methodology that optimizes such mappings, relative to a defined cost function. Methods for defining a model of user search, measuring necessary model parameters, and creating an optimal mapping are described and tested. In an experimental test of the method, the optimized mapping resulted in a 32% reduction in search time.

1. INTRODUCTION

This article introduces a computational method of assigning labels and functions to user selections in a multifunction display (MFD). An MFD consists of a computer display screen that provides information in response to a user's request. Through interaction with an interface (usually button pushes), the user moves through a hierarchical representation of information. Automated teller machines, medical devices, aircraft cockpits, electric typewriters, retail registers,fax machines, and many other devices utilize MFDs, with varying degrees of complexity.

At some point in the design of an MFD, decisions must be made about how to map the various parts of the information hierarchy to user actions (e.g., button pushes). These decisions can affect user performance, as demonstrated by Rogers, Cabrera, Walker, Gilbert, and Fisk (1996) for automated tellers; Obradovich and Woods (1996) and Cook and Woods (1996) for medical devices; Cuomo, Borghesani, Khan, and Violett (1998) for Intranet information servers; and Reising and Curry (1987), Dohme (1995), and Sirevaag et al. (1993) for aircraft flight controls.

Mapping hierarchy information to MFD buttons is a challenging task. The human–computer interactions involved in accessing information from an MFD are complicated and not entirely understood. Moreover, even a small hierarchy database can be mapped to hardware buttons in a vast number of ways (see Fisher, Yungkurth, & Moss, 1990, for a discussion),

Requests for reprints should be sent to Gregory Francis, Department of Psychological Sciences, Purdue University, 1364 Psychological Sciences Building, West Lafayette, IN 47907–1364. E-mail: gfrancis@psych.purdue.edu

so combinatorial explosion quickly precludes an exhaustive search of all possible mappings. As a result, such mappings are, at best, created by experts who rely on experience and general guidelines (Calhoun, 1978; Department of Defense, 1981; Holley & Busbridge, 1995; Lind, 1981; Spiger & Farrell, 1982; Williges, Williges, & Fainter, 1988). For example, one guideline suggests that frequently used labels should be the most accessible to allow users to quickly access those pieces of information needed most often. A similar guideline suggests that frequently used labels should be mapped to "ideal" buttons that are accessed more quickly than other buttons. Likewise, another guideline suggests that sequences of button presses should be designed to minimize movements. Unfortunately, it is unclear how to satisfy these, and other, guidelines. Applying the guidelines is problematic because they often conflict with each other, and mapping one label to a button can force movement of many other labels to accommodate the guidelines.

Recently, there has been interest in computational methods that can optimize the design of computer consoles (Farrell, 1997; Fisher, 1993; Hwa, Marks, & Shieber, 1995; Roske-Hofstrand & Paap, 1986; Sargent, Kay, & Sargent, 1997). The general approach is to gather relevant data about the human–computer interaction, build a quantitative model of that interaction, and then find a design that provides the best model performance. This article demonstrates how to apply this approach to assigning MFD labels to button presses.

A slightly different goal in the design of MFDs has also involved quantitative modeling, and it is worth discussing because the current approach is partly an outgrowth of these ideas. MFDs utilize a type of menu interface, and earlier quantitative analyses of menu search explored trade-offs between depth and breadth of hierarchy menus. At issue is whether it is better to have a broad design (with many menu options per page) or a deep design (with many levels of the hierarchy). By imposing several restrictions on hierarchy types and user behavior, Lee and MacGregor (1985) found an analytical form for expected search time in a menu interface. They used this analysis to conclude that deep hierarchies are preferred over broad hierarchies because the latter require substantial time searching for appropriate labels. Lee and MacGregor's analysis implicitly assumed that users did not know which button they should push and that there was no organization among the labels to guide the search. Paap and Roske-Hofstrand (1986) demonstrated that grouping labels could dramatically decrease the time required to find a desired label on a menu page. As a result, they concluded that menus should be broad rather than deep. Fisher et al. (1990) extended this line of analysis to a larger class of hierarchy structures. Norman (1991) provided an excellent summary of this and related lines of research. Unfortunately, the restrictions on hierarchy types and the model of user search preclude applying these analyses to the problem of optimizing the assignment of labels to MFD buttons.

The next section describes a quantitative model that considers the mapping of hierarchy labels to MFD buttons. A method for defining the model parameters is described as is an optimization approach that allows a computer to find the mapping between labels and MFD buttons, which minimizes expected search time. An experiment to test the methodology is then described. The final section discusses extensions to the approach taken here.

2. AN OPTIMIZATION METHOD

Creating an optimal mapping of hierarchy labels to MFD buttons requires two steps. First, the designer must build a quantitative model of the time needed to search through the hierar-

chy. Second, an optimization algorithm must be used to identify the best mapping, relative to the model. These steps are described next.

2.1. A Quantitative Model of User Search Time

The methodology described in this article most naturally applies to situations in which the number of available buttons and labels, and their hierarchical organization, are known. In these situations, the remaining task is to map the labels to buttons while preserving the established hierarchical order. A generalization of this approach to allow optimization of the hierarchical organization along with button mapping will be discussed in Section 5.

The model assumes that the time to reach a target label consists of independent button-path time and label time. A button path is the sequence of buttons that must be pushed to reach a desired label. Button-path time includes the time for the user to physically reach and push each of the buttons needed to move through the hierarchy. For example, one might expect that a path consisting of repeated selection of a single button would have a short button-path time. Let $B(i)$ correspond to the time needed to move through the button path assigned to label *i*.

Label time includes everything that is not button-path time. Label time may include reading and categorization time if the user is unfamiliar with the labels and must visually search for target labels. It may include time for memory recall if the user has previous experience searching for the target label. Label time may include computer response time for situations when displaying information requires substantial time. Label time may also include movement time that is specific to a label rather than to a button path. For example, the functions associated with some labels may only be needed under certain contexts (e.g., evasive actions in military aircraft). In those contexts, it may take more (or less) time to reach a button than is normal. Let $L(i)$ correspond to the label time for *i*. An assumption of the model is that label time is unrelated to the sequence of buttons that might make up the button path assigned to a label. This is an assumption that surely does not hold for all situations but can be experimentally tested.

The model proposes that the total time to reach a target label *i* is the sum of button path and label times:

$$T(i) = B(i) + L(i) \tag{1}$$

The mean search time to find a label is then

$$E(T) = \sum_{i=1}^{m} B(i)p_i + \sum_{i=1}^{m} L(i)p_i \tag{2}$$

where m is the number of labels in the hierarchy and p_i is the probability of searching for label *i*. For simplicity, assume that the design goal is to minimize mean search time. Because the second summation does not vary with button path, finding the optimal assignment of button paths to target labels is equivalent to finding the set of button paths that minimizes the first summation. Designate the button path for label *i* as a vector $\mathbf{B_i}$ and let $\beta = \bigcup_{i=1}^{m} \mathbf{B_i}$ be the set of buttons paths with assigned labels. Let β^* be the set of button paths that minimizes expected search time. Then

$$\beta^* = \arg\min_{\beta} \sum_{i=1}^{m} B(i)\, p_i \tag{3}$$

where $B(i)$ is the button-path time associated with button-path $\mathbf{B_i}$. Different choices of β correspond to different mappings of labels to button paths. The summation in Equation 3 is the expected button-path time and can be interpreted as the cost for choosing a mapping. Within the context of the model, the task of optimally mapping labels to buttons is the same as the task of finding a mapping that minimizes this cost. The next section describes a computational method for finding the mapping that minimizes this cost.

2.2. Optimizing the Design

It might seem that with a quantified model of search times, one could simply let a computer consider all possible mappings of labels to button paths and find the mapping that minimizes expected search time. Unfortunately, the computational magnitude of the problem precludes such an exhaustive search. Even a small MFD hierarchy with eight buttons and three levels, for instance, provides 512 unique button paths that reach to the third level. Suppose every label is filled at every level so that there are eight choices at the top level (one button press), eight choices at each second level (two button presses), and eight choices for each bottom level (three button presses). If the problem is constrained so that all the hierarchical relations between labels must be maintained in their button paths (as in the discussion next), then at each level there are 8! = 40,320 (8 factorial) possible assignments of labels. Thus, the total number of possible mappings that still maintain the hierarchical relations among labels is the cube of 8! or approximately 6.55×10^{13}. As the size of the MFD increases, the number of mappings becomes so enormous that even high-speed computers cannot consider all the possibilities in a timely manner (for example, MFDs with 20 buttons and five levels are not excessively large in real world applications, but the number of mappings is approximately 8.52×10^{91}). Thus, an approach other than exhaustive trial and error is needed.

For complex optimization problems of this type, computer scientists often apply a technique called *hill-climbing*. Given an initial mapping, a computer can calculate its cost using the sum in Equation 3. If the designer modifies the mapping and the new cost is smaller than the old cost, then the new mapping replaces the older mapping. Iterating this process eventually leads to a mapping for which the mapping cannot be reduced further. Intuitively, each possible mapping of labels to buttons corresponds to a position in a (multidimensional) space. The cost at each position defines a surface in the space. Changing the mapping of labels to buttons is, then, like moving through the space. Hill-climbing techniques start at an initial point in space and move in a direction that goes down the cost curve (thus, *hill-descending* might be a better term for these techniques).

This methodology was applied to the problem of mapping labels to button paths. An initial mapping consisted of randomly assigning labels to button paths (in a way that preserved the appropriate hierarchical relations). A pair of labels with a common parent node (i.e., each node was in the same position in the hierarchy structure but with button paths that differed in the last button) were randomly chosen and swapped button paths. The mean button-path time was calculated both before and after the swap. If the swap resulted in a reduction in search time, it was kept. Otherwise the labels were returned to their original

button paths, and a new random pair was selected. As it made each swap, the program also automatically updated the button paths for children (i.e., those labels that are hierarchically below a label) of moved labels, as their paths were modified by the change of their parent. This procedure was repeated many times until there seemed to be no further changes in mean button-path time. The final assignment of labels to button paths was taken as the best.

A problem with the hill-climbing approach is that because it never allows changes that go up the cost curve, the system can become trapped in local valleys and, thus, not find a deeper global minimum over the next hill. Simulated annealing (Geman & Geman, 1984) avoids this problem by introducing extra "energy" into the system. Initially, movement can occur in directions that go up or down the cost curve. The relative probability of moving up the cost curve is related to the cost itself so that higher costs have smaller probabilities of allowing upward movements. A temperature parameter gradually lowers the overall probability of moving up the cost surface so the system is likely to become stuck in the deepest valley of the curve, a global minimum. The system is more likely to climb out of local cost minima because they are not as deep. By decreasing the temperature gradually, the system is very likely, but not guaranteed, to rest in a state that corresponds to the minimum of the cost function. In this application, this means the system will likely choose a mapping of labels to button paths that minimizes the average model button-path time.

3. MODELING BUTTON-PATH TIME

Minimizing the mean button-path time requires knowing all possible B_i terms. That is, the designer must know the button-path time for every possible combination of button presses that can lead to a label. One way to do this is to directly measure movement times for every possible set of button pushes. This is feasible for a small set of buttons with few hierarchy levels. However, as the number of levels or number of buttons increases, the number of paths increases on the order of n^v, where n is the number of buttons on each MFD page (breadth), and v is the number of levels (depth) in the hierarchy. Such increases in problem size preclude attempts to measure all button paths. For example, a system with eight buttons and five levels would have 37,448 different button paths, and each path time would need to be estimated separately.

Landauer (1987) suggested that Fitts's (1954) law could be used to model movement times in MFD-type devices. Fitts's law proposes that mean movement time is a function of the logarithm of distance moved divided by target width. If Fitts's law held for MFD searches, a designer could use its predictions of motor times to calculate the time needed to move through a path of buttons. An implication of Fitts's law would be that movements to and from corner buttons of an MFD should take longer than movements to and from other buttons because the corner buttons are relatively distant from other buttons. On the other hand, one could imagine that corner buttons, by their isolation, are easier to identify and strike, thereby leading to the opposite results. So it is unclear whether Fitts's law should adequately describe movement times in this context.

To be able to consider situations in which Fitts's (1954) law might not hold, this study used a model of button-path times that is more likely to apply than Fitts's law but is simpler than estimating each path time individually. The path to a label in an MFD hierarchy can be identified by the necessary sequence of button pushes from the top hierarchy level. Number

the buttons of the MFD 1,…,*n*. Each button path can be represented by an ordered set [b_1 …, b_v], where *v* is the number of buttons that need be pushed to reach the end of the path. The model approximates button-path time by summing the time needed to move from one button to strike the next. Let *s*[*j, k*] indicate the time needed to move from button *j* to strike button *k.* Then, an estimate of the time needed to travel through the button path of label *i* is

$$B(i) = \sum_{j=0}^{v-1} s[b_j(i), b_{j+1}(i)] \tag{4}$$

The first term in the summation $s[b_0(i), b_1(i)]$ is the time taken to strike the first button in the path. The rest of the summation then adds the time needed to move from the first button to the second, the second to the third, and so on until the button related to the label is pressed.

Such a scheme greatly reduces the effort needed to build the model because the experimenter needs only to estimate the *s*[*j, k*] terms. Such estimation requires measuring the time between every pair of buttons. For a hierarchy with eight buttons, this requires only 64 measures—regardless of the number of hierarchy levels.

4. EXPERIMENT

4.1. Method

4.1.1. Participants. All participants received course credit at Purdue University. There were two groups of participants. Group 1 (16 participants) was used to gather data to define the model of button-path times. That data was then used to build an optimal mapping of labels to button paths, and Group 2 (11 participants) was used to validate the resulting hierarchy. Participants were run individually, and an experimental session lasted approximately 25 min.

4.1.2. Materials. An MFD was emulated with a program written in Java and run on a Silicon Graphics workstation. It consisted of a frame with buttons along the left, right, and bottom sides. The buttons on the left and right sides had modifiable labels that changed as the user traversed (with mouse clicks) through the virtual space of the hierarchy. Figure 1 shows screen shots of the frame.

The top level of the information hierarchy consisted of music types. The middle level consisted of music acts appropriate for the selected music type. The bottom level of the hierarchy consisted of songs by the appropriate music act. All totaled there were 585 labels in the hierarchy database. The participant's task was to search through the MFD to click on the button containing a requested target label. Each target label was given in the middle of the frame (see Figure 1), and the search always started from the top level. To aid in the search, if the target label was a song title, the participant was also given the appropriate music act.

Each participant sequentially went through three MFD searches: practice, button-path measurements, and test. In the practice search, a random subset of 10 labels was selected as targets. Each target label was randomly assigned a presentation frequency between 1 and 10 and was fixed across participants and groups. All the labels were then randomly assigned

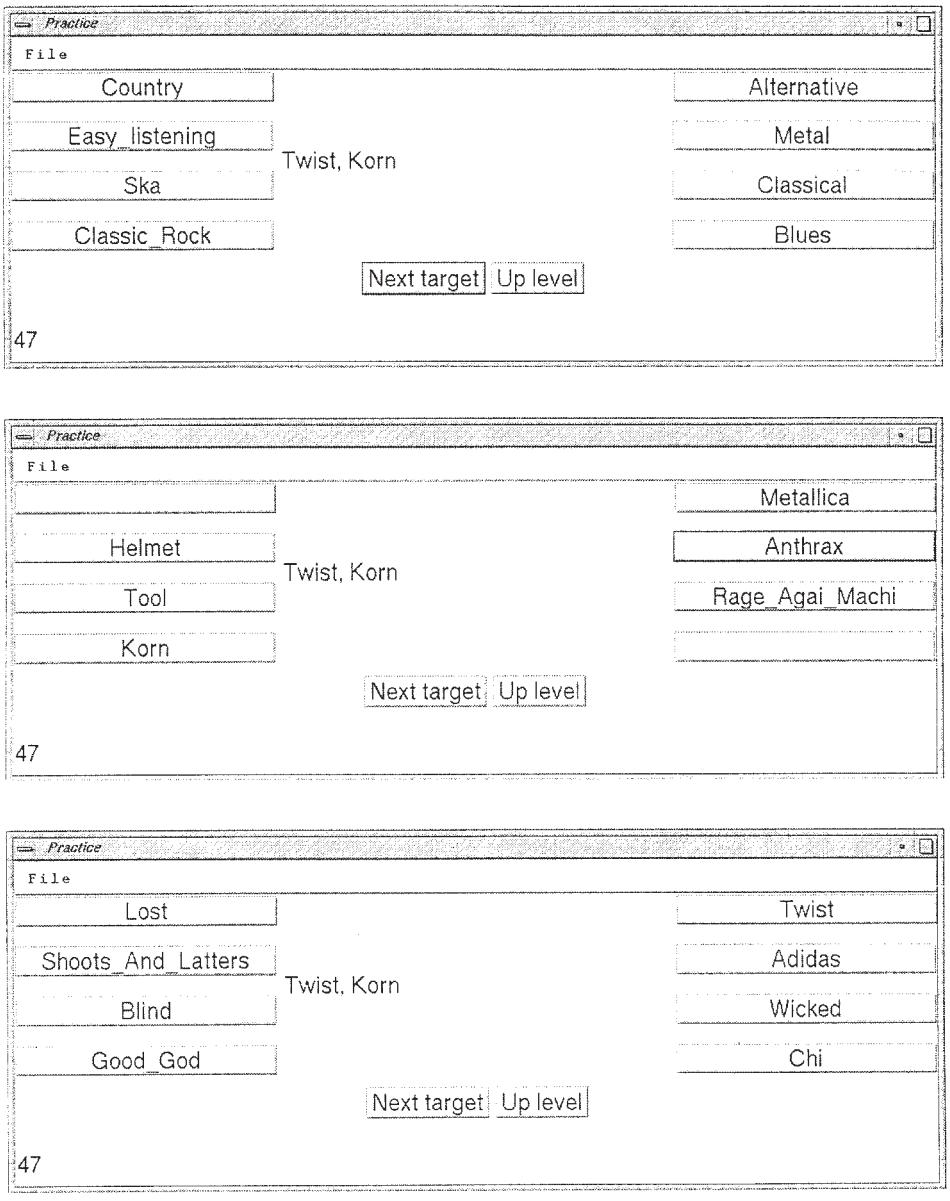

FIGURE 1 An emulated multifunction display interface. Users moved through the information hierarchy by selecting buttons. A target label, *Twist* (a song by the artist Korn), is presented on the middle of the frame. The number on the bottom left indicates the number of remaining trials. (Top) The top hierarchy level consisted of music categories. (Middle) Selecting *Metal* from the top levels shows artists for the selected music category. (Bottom) Selecting *Korn* shows songs by the artist.

button paths with the restriction that the hierarchical organization of information was preserved in the button paths. In the practice condition, participants searched for every label with its appropriate frequency (55 searches total). The practice searches were designed to provide the participant with experience moving the mouse to press buttons and to partly familiarize the participant with the hierarchical relations among the labels in the database.

After finishing the practice search, button pair movement times were estimated for the participant. This was done with a modified version of the MFD frame. For the modified frame, the labels were replaced with numbers, and the participant's task was to move as quickly as possible between the indicated buttons (see Figure 2). Each participant made two movements to every pair of buttons (except movements to the Next target button and movements to and from the Up level button). For each movement, the time between the first button press and the second button press was measured and stored by the computer. These times, when averaged across participants, provided the $s[b_j, b_j + 1]$ times used in Equation 4.

Finally, the test was much like the practice search except that a new set of target labels were selected, and the mapping of labels to button paths was modified. For Group 1, the labels were mapped to button paths in a random order that was fixed across participants. For Group 2 the labels were mapped to button paths as prescribed by the optimization procedure. Every participant went through the target set twice, and search times were recorded only on the second set.

4.2. Results

The search and movement time data were used to explore four main topics: (a) to explore whether or not searches in the optimized hierarchy were faster than searches in the nonoptimized hierarchy, (b) to see if Fitts's (1954) law could be applied to the movement time data, (c) to explore learning effects by looking at search time differences between target labels that were repeated many times compared to those rarely searched for, and (d) to explore whether the assumed independence between button path and label times is valid.

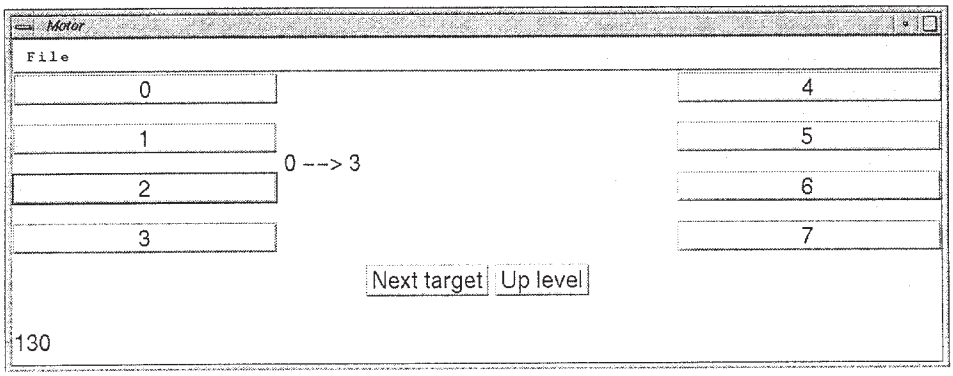

FIGURE 2 A modified version of the multifunction display interface that measures the time to move between button pairs. On each trial, users moved as quickly as possible between the indicated buttons.

4.2.1. Effects of optimization. Participants in Group 1, with random assignments of labels to button paths, required an average of 3.7 sec to reach a target label. Participants in Group 2, with the calculated optimal assignments, required an average of 2.5 sec. Searching the optimized MFD was 32% faster than the random MFD. This finding demonstrates that the optimization approach assigned labels to buttons in a manner that reduced overall search time. Looking through the optimized hierarchy reveals why it produced shorter times. The optimal hierarchy placed the most frequently searched for labels on paths that involved repeated pressing of the same button. Figure 3 demonstrates such a path for a commonly accessed label. Those labels that could not be placed along a repeating path had paths that minimized movement time. In general, this organization is consistent with the guidelines suggested by design experts.

Details of the optimized mapping can be seen in Figure 4, which plots the average time needed to move between buttons as a function of label search frequency. For every label, the optimized MFD has shorter between-button movement times. Averaged across all target labels, the random mapping had a between-button movement time of 701 ms, whereas the optimized mapping had an average of 469 ms. The reduction in movement time for all target labels indicates that the target set was sufficiently distributed across the hierarchy topics so that nearly every label could be placed on a button path involving repeated selection of a single button. If the target set increased in size or used only labels for common topics, the optimization approach would assign the most frequently searched for labels to button paths with short times, thereby forcing less frequently searched for labels to be mapped to longer button paths. There is a slight tendency for this phenomenon in Figure 4, as those labels searched for most often tend to have slightly shorter between-button times than labels that are rarely searched for.

Participants who worked with the optimized hierarchy reported that the overall feel of searching for information was that the target label would likely be found where they expected it to be and would be easily accessible. It seems likely, but was not explicitly investigated, that in addition to reducing search times, such optimal mappings will produce fewer errors and increase overall user satisfaction.

4.2.2. Details of button press times. Figure 5a plots between-button movement times for buttons averaged across both groups. The numbers on the abscissa correspond to the buttons as in Figure 2 except that number 8 corresponds to the Next target button. Separate curves are averages for movements ending with the button on the abscissa (To), movements starting with the button on the abscissa (From), and movements that involved two successive strikes of the same button (Repeat). Two properties are noteworthy. First, for button paths that involve repeated selection of the same button, all buttons give essentially the same time and that time is substantially shorter than is needed to move between buttons. Second, both the time to and the time from curves show a scalloped shape with longer reaction times for buttons on the corners of the frame (buttons 0, 3, 4, and 7). This suggests that buttons in the middle of the MFD have an advantage over corner buttons because middle buttons are, on average, closer to more buttons than corner buttons.

These findings suggest a reduction in search time for those paths that involve repeated selections of the same button. For those paths that must involve selecting different buttons, middle buttons generally have an advantage over corner buttons. The latter property is qual-

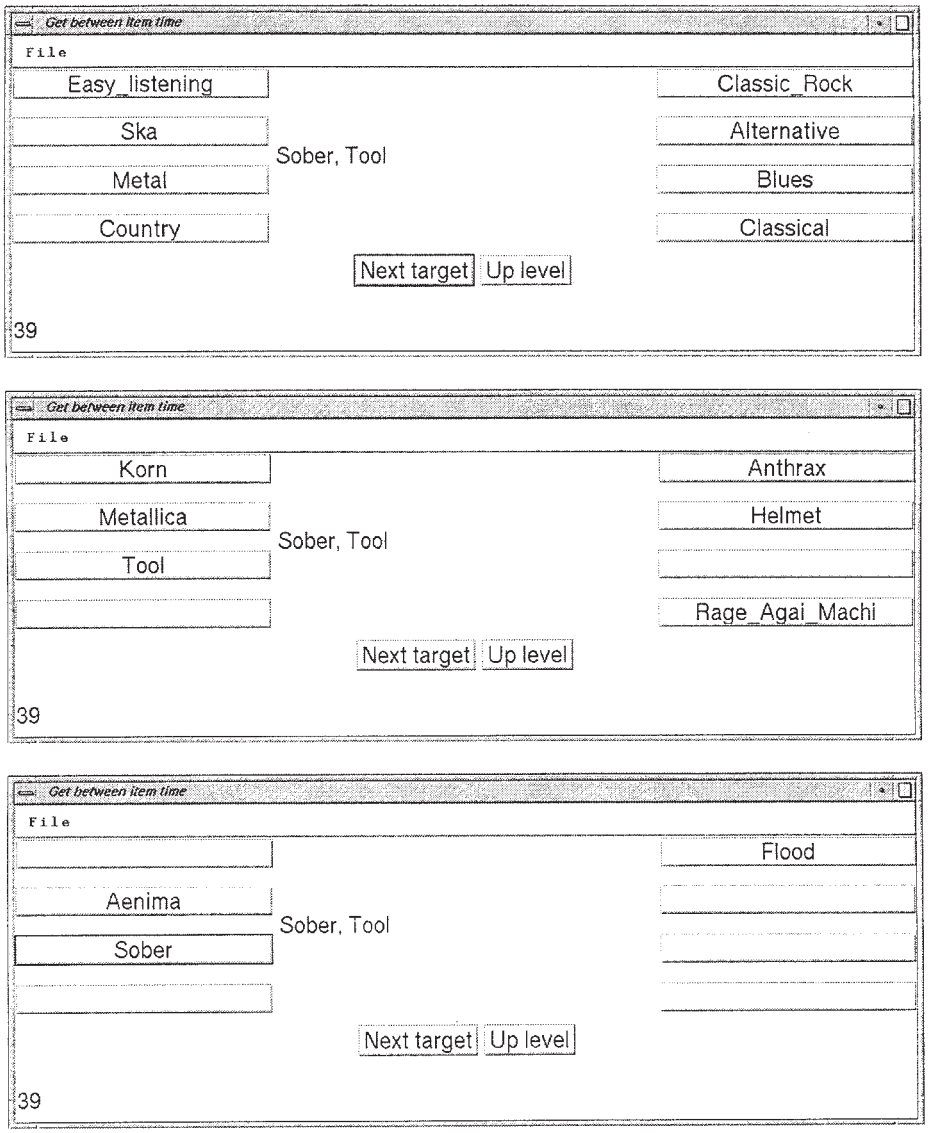

FIGURE 3 The optimal multifunction display interface. The user needs only to press one button repeatedly to reach the target song, *Sober*. Tool is the artist. Music categories are shown in the top box, artists in the *Metal* category are shown in the middle box, and songs by Tool are shown in the bottom box.

itatively consistent with Fitts's (1954) law, which suggests that the time to move between buttons should obey the equation

$$a \log_2 \left(\frac{D}{S} \right) \tag{5}$$

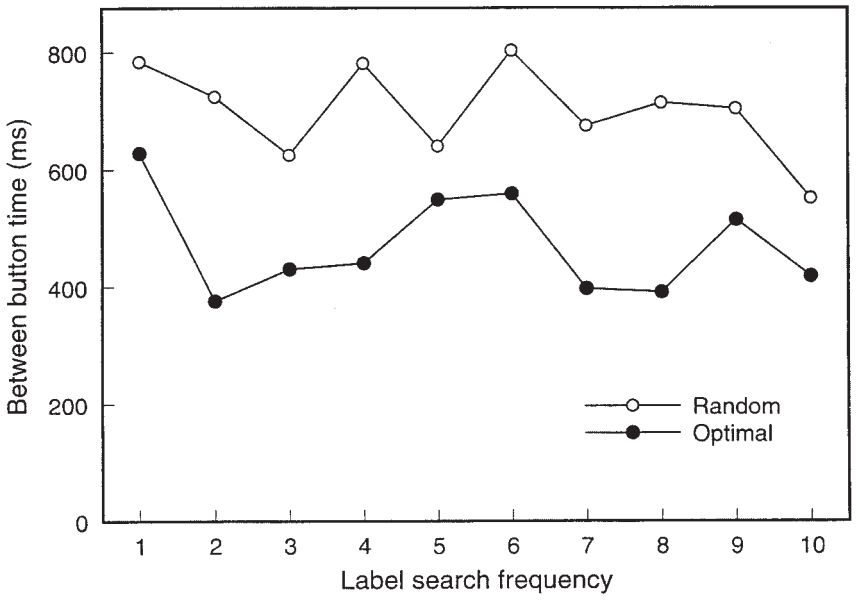

FIGURE 4 The average time needed to move between buttons for the paths of target labels. The optimal multifunction display design used button paths with shorter movement times than the random design.

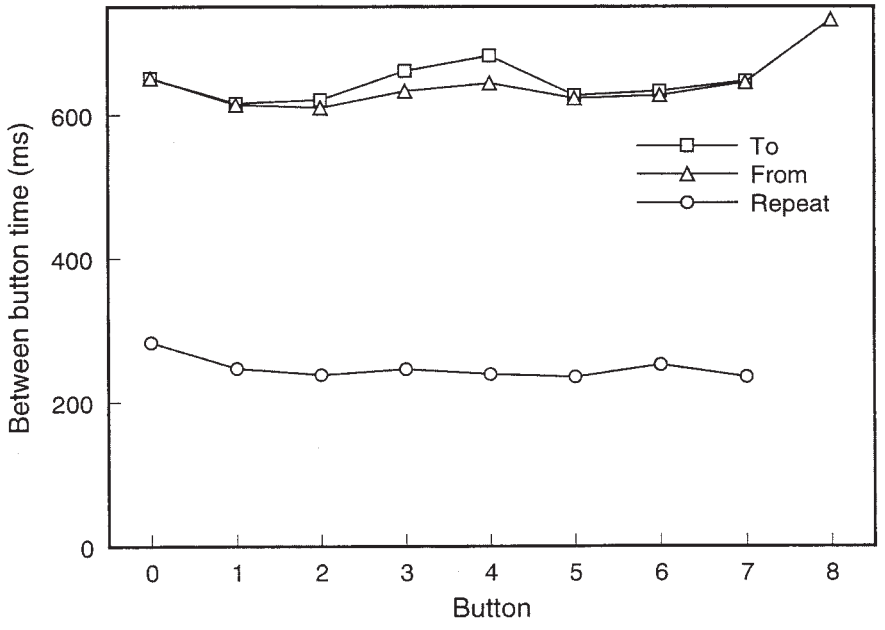

FIGURE 5 The average time needed to move to, move from, or repeat pressing a button. The relatively flat shape of the repeat curve indicates that no single button was repeated more quickly than any other. The scallop shape of the top curves indicates that the corner buttons (0, 3, 4, and 7) required longer times to move to and from; probably because of their relative isolation compared to the other buttons.

where D is the distance between a pair of buttons, S is the size of the button that is being moved to, and a is a constant scaling factor estimated from the data. Using Fitts's law to model between-button times in an MFD requires some additional terms. Fitts's law describes the time to move between buttons but does not necessarily include time needed to then strike a button. Likewise, in repeated selection of a button, there is no movement, so Fitts's law does not apply. A simple model of the time to move between buttons i and j would sum these terms:

$$s[j,k] = a\left[\log_2\left(\frac{D_{jk}}{S}\right)\right]^+ + c \tag{6}$$

where D_{jk} is the distance between buttons j and k, the notation $[x]^+ = \max(0, x)$ corresponds to half wave rectification, and c is the average time needed to strike a button in rapid succession:

$$c = \frac{1}{8}\sum_{k=0}^{7} s[k,k] \tag{7}$$

Using linear regression, it is possible to derive a least squares estimate of a as

$$a = \frac{\sum_{j=0}^{8}\sum_{k=0,k\neq j}^{7}(s[j,k]-c)\log_2\left(\frac{D_{jk}}{S}\right)}{\sum_{j=0}^{8}\sum_{k=0,k\neq j}^{7}\log_2\left(\frac{D_{jk}}{S}\right)} \tag{8}$$

Here, $j = 8$ corresponds to movement from the Next target button, and the restriction $k \neq j$ avoids those situations in which the user strikes a button twice. Measuring the center-to-center distance for each pair of buttons and plugging in the appropriate data results in a model of between-button time that obeys the equation

$$s[j,k] = 127\left[\log_2\left(\frac{D_{jk}}{S}\right)\right]^+ + 244 \tag{9}$$

The value of $a = 127$ is similar to other estimates (Card, Moran, & Newell, 1986). The $s[j, k]$ terms can be used in equation (4) to predict path times.

Figure 6 plots $s[j, k]$ values produced by the Fitts's (1954) law model and values measured experimentally for every pair of buttons. For both the model and the data, the hills correspond to movements that went from a button on one side of the display to the other side, and the valleys correspond to movements between buttons on a common side of the display. The eight data points on the far right correspond to movements involving the Next target button, which is located slightly left of the center along the bottom of the display. Fitts's law does a good job capturing the basic rise and fall of between-button time and has an overall correlation with the data of $r = 0.93$. However, the model fails to capture much of the fine structure at the peaks and valleys. The Fitts model tends to underestimate between-button

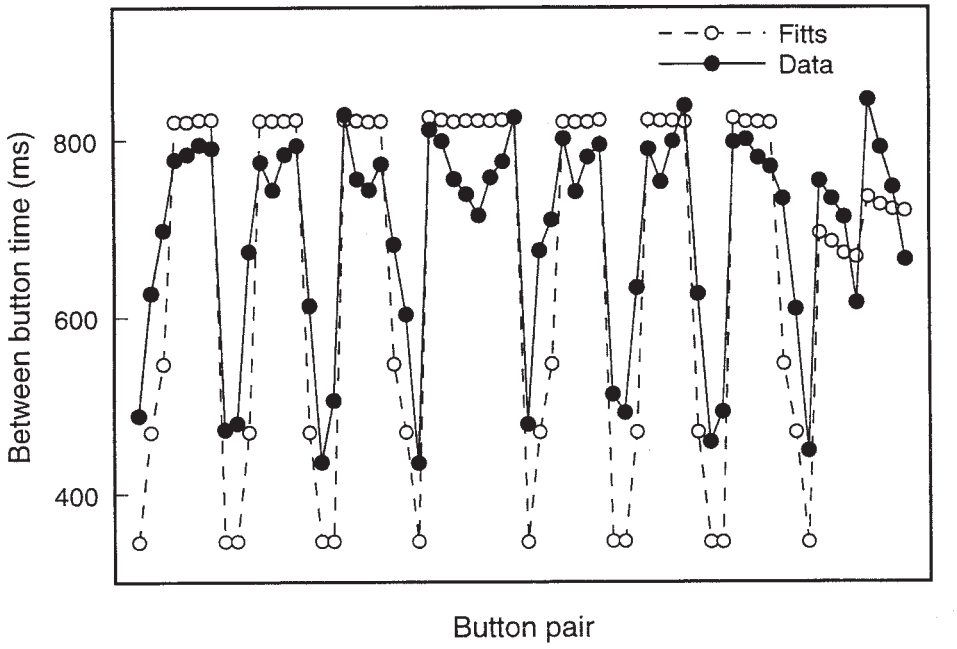

FIGURE 6 The between-button time for every button pair as measured experimentally and as modeled by Fitts's (1954) law. Fitts's law tends to underestimate the shortest times and overestimate the longest times. The Fitts law model also fails to differentiate movements directly across the multifunction display from oblique movements across the MFD. The graph shows these as smaller dips in the hills.

time for neighboring buttons (bottom of a valley) by more than 100 msec. Likewise, the Fitts model tends to overestimate between-button time for buttons on opposite sides of the display and fails to match the scallop shape at the peaks. The scallop occurs as a result of differences in movements that also includes a vertical component (longer time). The vertical distances are small compared to the horizontal distances, so Fitts's law does not properly distinguish those movement times.

Determining whether Fitts's (1954) law provides an adequate model of between-button time depends on whether the details missed by Fitts's law will have a significant impact on the optimization process. For the current optimization approach, the 100-msec discrepancies at the longest and shortest between-button times are likely too big to be tolerated. This discrepancy may disappear or may be tolerable in other contexts. However, there is reason to expect that Fitts's law will not apply to all cases of MFD hierarchy search times. Fitts's law describes the time needed to quickly move from one position to another. Applying this law to MFD searches assumes that the user physically moves a pointer or a finger from one button to another. This may not always be the case. If the buttons are close enough, the user may use one finger to push one button and another finger to push another button. Likewise, the user may use two hands should the physical spacing of buttons allow it. Fitts's law is unlikely to apply in such cases because there is no movement between buttons. Despite these limitations, for those situations in which Fitts's law does apply, it provides an easy means to model button movement times.

4.2.3. Learning and independence. The model assumes that the time taken to move through a path of buttons toward a label can be broken down into independent factors that are specific to the button path and to the label. This assumption can be tested. Rewriting Equation 1 shows that, in the model, the label time obeys the equation

$$L(i) = T(i) - B(i) \tag{10}$$

Both $T(i)$ and $B(i)$ were measured during the course of the experiment, so it is possible to estimate $L(i)$. If $L(i)$ does not vary when the mapping of labels to button paths is changed to the optimal mapping, then adding the $L(i)$ values back to the button-path times should provide an accurate prediction of search time:

$$T(i)^* = B(i)^* + L(i) \tag{11}$$

where the asterisks indicate values for the optimized condition. For the optimized mapping, the predicted mean search time was 3.0 sec, which is longer than the 2.5-sec actual mean search time. The most likely explanation for this discrepancy is that users of the optimized mapping more easily learned the button sequence needed to reach a target label. For example, remembering that a label is reached by pressing button 2 three times in rapid succession is likely easier than remembering that a label is reached by pressing button 2, then button 6, and then button 5. Such learning effects violate the assumed independence of button paths and label times.

To compare recall times for different labels, $L(i)$ was divided by the level of the label in the hierarchy. This gives the average between-level time for each target label, which corresponds roughly to the time spent searching for the correct labels along the target's button path. That learning plays a role can be seen in Figure 7, which compares the between-level times for the random and optimized mappings as a function of the search frequency for a label. There are two important demonstrations of learning. First, between-level times are shorter as label search frequency increases. This is likely due to improved recall of button paths for those labels that are searched for more often. This finding is consistent with earlier studies of learning in menu searches (Vandierendonck, Van Hoe, & De Soete, 1988) and is unrelated to the independence issue. Second, the between-level time for the optimized mapping is substantially shorter than for the randomized mapping (M = 512 and 992 msec, respectively). The difference between optimized and randomized mappings demonstrates that the assumed independence between button-path times and label times is not an accurate description of MFD searches in the experiment.

The lack of independence between button-path times and label times prevents the model from accurately predicting search time because Equation 11 is not valid. However, the lack of independence may not prevent the system from finding an optimal mapping. In particular, the shape of the curves in Figure 7 are very similar, thereby indicating that the optimized mapping simply makes it easier to learn the button path for every label in the target set. Because every target label is affected nearly equally, it is as if $L(i)$ for every term was subtracted by a constant amount. Plugging that constant into Equation 2 shows that, again, the optimal hierarchy is found by minimizing expected button-path times and Equation 3 again applies. Of course, there is no guarantee that the change in $L(i)$ will be constant for other situations, and if it is not, then Equation 3 must be modified to include the $L(i)$ terms. The constant change in between-level times demonstrated in Figure 7 also suggests that it may be

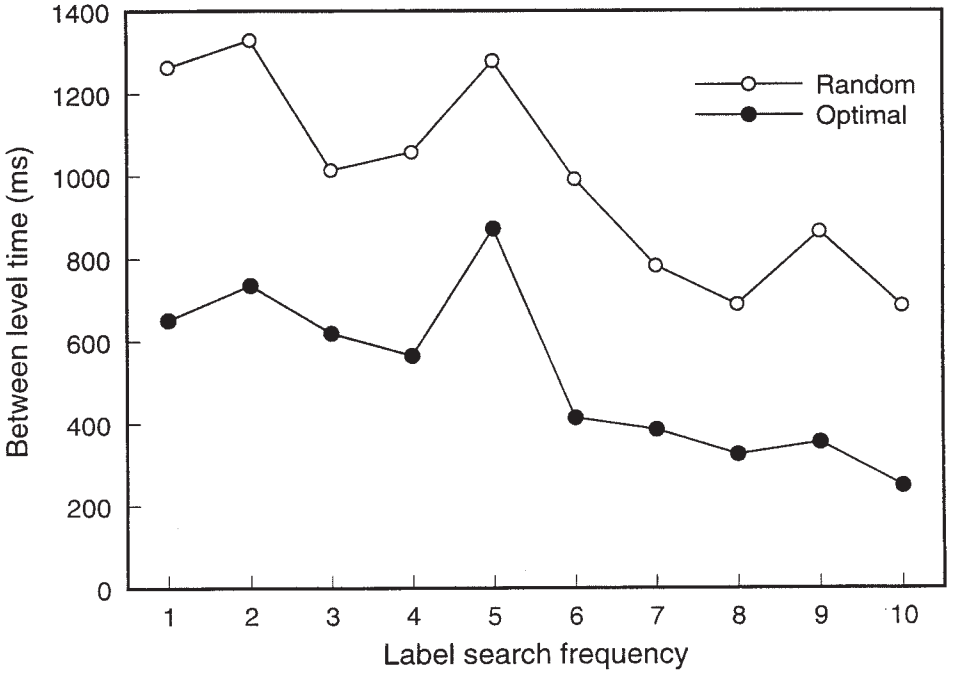

FIGURE 7 The average time needed to move between levels, after factoring out button-path time, as a function of frequency of search. For both the random and optimal mappings, between-level time generally decreased with label frequency, suggesting memory effects. The between-level times were smaller for the optimal mapping, thereby rejecting the hypothesized independence between button-path and label times.

possible to predict the way that button paths interact with labels. Further study is necessary to explore this possibility.

This discussion has assumed that shorter than predicted search times are due to learning effects. An alternative explanation is that participants made more errors in the random condition than the optimized condition. After an error, the user needed to move back up a level and then move to the correct button. The time to recognize the error and execute compensatory movements was added into the $T(i)$ times. If the optimized button paths had fewer errors, the $T(i)^*$ times would be shorter than the model would predict, and this would account for the observed changes in $L(i)$. The software, unfortunately, did not keep track of error data, so additional experiments will be needed to explore these effects.

5. DISCUSSION

One advantage of treating the mapping of labels to buttons as a cost minimization optimization problem is that it can be easily extended to include other influences on the design process. For example, instead of simply minimizing expected search time, a new cost function could be introduced that sums a variety of individual cost functions:

$$C = \sum_{k=1}^{n} \lambda_k C_k \tag{12}$$

where C_k is a cost function and λ_k is a weight assigned by the designer. For example, Francis and Reardon (1997) described cost functions for consistent placement of a label in multiple places in a hierarchy to ensure that time critical functions are easily accessed, keep related functions close to each other, and minimize the depth of the hierarchy. Other efforts to optimize interface designs (Hwa et al., 1995; Kleiss, 1997; Roske-Hofstrand & Paap, 1986; Sargent et al., 1997) can probably also be incorporated into this scheme by recasting the problem as one of cost minimization. A cost could also include relevant data from cognitive psychology on stimulus response compatibility (Proctor & Van Zandt, 1994). Likewise, if a designer wishes to impose special restrictions that derive from other constraints (e.g., carry over design properties from an earlier MFD), it is easy to define a cost function to impose that constraint. A designer can also impose hard constraints on the optimization procedure and force it to optimize the remaining parts of the system subject to those constraints.

A challenge of introducing additional cost functions is to scale them properly so that each cost function contributes to the final design appropriately. This can be done by setting the λ_k weight of each cost appropriately. Determining the appropriate weight requires knowledge of how the different cost terms affect usability of the device. As these influences have only been investigated in a piecemeal fashion, there remains some substantial experimental work before the general method can be used to its full ability.

Even without the elaboration, the current methodology offers substantial benefits. The optimization approach was shown to generate a mapping of hierarchy labels to buttons that reduced search time by 32%. This validates the basic approach. Although additional experiments are needed to further explore the strong effects of learning and their interactions with mapping type, the methodology proposed in this article promises to dramatically and easily improve the design of MFD devices.

ACKNOWLEDGMENTS

This work was supported in part by the Army Aeromedical Research Laboratory (Dr. Kent A Kimball) under the auspices of the U.S. Army Research Office Scientific Services Program administered by Battelle (Delivery Order 1832, Contract No. DAAL03–91–C–0034; Delivery Order 63, Contract No. DAAH04–96–C–0086; and Delivery Order 224, Contract No. DAAH04–96–C–0086).

Thanks to Charles James and Joshua Killey for assistance designing the hierarchical database and running experimental participants.

The views, opinions, and /or findings contained in this report are those of the author and should not be construed as an official Department of the Army position, policy, or decision, unless so designated by other documentation.

REFERENCES

Calhoun, G. (1978, October). Control logic design criteria for multifunction switching devices. *Proceedings of the Human Factors Society 22nd Annual Meeting,* 383–387.

Card, S. K., Moran, T. P., & Newell, A. (1986). The model human processor: An engineering model of human performance. In K. R. Boff, L. Kaufman, & J. P. Thomas (Eds.), *Handbook of perception and human performance: Vol. II. Cognitive processes and performance* (pp. 45-1 to 45-35). New York: Wiley.

Cook, R. I., & Woods, D. D. (1996). Adapting to new technology in the operating room. *Human Factors, 38,* 593–613.

Cuomo, D. L., Borghesani, L., Khan, K., & Violett, D. (1998). Navigating the company web. *Ergonomics in Design, 6,* 7–14.

Department of Defense. (1981). *Military standard: Human engineering design criteria for military systems, equipment, and facilities* (Report No. MIL–STD–1472D). Washington, DC: Author.

Dohme, J. (1995). The military quest for flight training effectiveness. In W. Larsen, R. Randle, & L. Poposh (Eds.), *Vertical flight training* (NASA Reference Publication 1373). Washington, DC: National Aeronautics and Space Administration.

Farrell, P. (1997). A human-machine interaction analysis using layered protocol theory. *Proceedings of the 29th Annual Conference of HFAC/ACE,* 39–41.

Fisher, D. L. (1993). Optimal performance engineering: Good, better, best. *Human Factors, 35,* 115–139.

Fisher, D. L., Yungkurth, E., & Moss, S. (1990). Optimal menu hierarchy design: Syntax and semantics. *Human Factors, 32,* 665–683.

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

Francis, G., & Reardon, M. (1997). *Aircraft multifunction display and control systems: A new quantitative human factors design method for organizing functions and display contents* (USAARL Report, No. 97–18). Fort Rucker, AL: United States Army Aeromedical Research Laboratory.

Geman, S., & Geman, D. (1984). Stochastic relaxation, Gibbs distributions, and the Bayesian restoration of images. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 6,* 721–741.

Holley, C., & Busbridge, M. (1995, May). Evolution of the Venom variant of the AH–1W super-cockpit. *Proceedings of the American Helicopter Society 51st Annual Forum,* 1436–1449.

Hwa, R., Marks, J., & Shieber, S. (1995). *Automatic structuring of embedded hypermedia documents* (Tech. Rep. TR–95–6). Cambridge, MA: Mitsubishi Electric Research Laboratory.

Kleiss, J. (1997). Identifying users' conceptual organization of menu functions in an automotive electronic navigation product. *Proceedings of the Human Factors and Ergonomics Society 41st Annual Meeting, 2,* 944–948.

Landauer, T. K. (1987). Relations between cognitive psychology and computer system design. In J. M. Carroll (Ed.), *Interacting thought: Cognitive aspects of human computer interaction*. Cambridge, MA: MIT Press.

Lee, E., & MacGregor, J. (1985). Minimizing user search time in menu retrieval systems. *Human Factors, 27,* 157–162.

Lind, J. (1981). *Evaluation of cockpit procedures, displays, and controls for stores management in the advanced aircraft armament system (AAAS)* (Naval Weapons Center Technical Memorandum No. 4538). China Lake, CA: Naval Weapons Center.

Norman, K. L. (1991). *The psychology of menu selection: Designing cognitive control at the human/computer interface*. Norwood, NJ: Ablex.

Obradovich, J. H., & Woods, D. D. (1996). Users as designers: How people cope with poor HCI design in computer-based medical devices. *Human Factors, 38,* 574–592.

Paap, K., & Roske-Hofstrand, R. (1986). The optimal number of menu options per panel. *Human Factors, 28,* 377–385.

Proctor, R. W., & Van Zandt, T. (1994). *Human factors in simple and complex systems*. Boston: Allyn & Bacon.

Reising, J., & Curry, D. (1987). A comparison of voice and multifunction controls: Logic design is the key. *Ergonomics, 30,* 1063–1077.

Rogers, W., Cabrera, E., Walker, N., Gilbert, K., & Fisk, A. (1996). A survey of automatic teller machine usage across the adult life span. *Human Factors, 38,* 156–166.

Roske-Hofstrand, R., & Paap, K. (1986). Cognitive networks as a guide to menu organization: An application in the automated cockpit. *Ergonomics, 29,* 1301–1311.

Sargent, T. A., Kay, M. G., & Sargent, R. G. (1997). A methodology for optimally designing console panels for use by a single operator. *Human Factors, 39,* 389–409.

Sirevaag, E., Kramer, A., Wickens, C., Reisweber, M., Strayer, D., & Grenell, J. (1993). Assessment of pilot performance and mental workload in rotary wing aircraft. *Ergonomics, 36,* 1121–1140.

Spiger, R., & Farrell, R. (1982). *Survey of multi-function display and control technology* (Report No. NASA–CR–167510). Washington, DC: National Aeronautics and Space Administration.

Vandierendonck, A., Van Hoe, R., & De Soete, G. (1988). Menu search as a function of menu organization, categorization and experience. *Acta Psychologica, 69*, 231–248.

Williges, R., Williges, B., & Fainter, R. (1988). Software interfaces for aviation systems. In E. Wiener & D. Nagel (Eds.), *Human Factors in Aviation* (pp. 463–491). San Diego, CA: Academic.