0743-4618/98/1401-0025 $3.00/0. Volume 14, March 1998
AAC Augmentative and Alternative Communication

Model Simulations of User Performance with Word Prediction

Heidi Horstmann Koester and Simon P. Levine
Rehabilitation Engineering Program, Graduate Bioengineering Program, Department of Physical Medicine and Rehabilitation, University of Michigan, Ann Arbor, Michigan, USA

Previous work has demonstrated that, under well-defined conditions, a quantitative model can accurately represent user performance with word prediction systems (Koester, 1994; Koester & Levine, 1994, 1995, 1997). This paper illustrates the use of this model to simulate user performance across a broad range of conditions. Examples of simulation results are presented to show how user characteristics, strategy of use, and system characteristics combine to determine overall performance. The use of model simulations to inform clinical decision-making and research questions is discussed.

KEY WORDS: augmentative communication, rate enhancement, user modeling, word prediction

Development of quantitative models of user performance has been a longstanding research interest in augmentative and alternative communication (AAC) and computer access (Dabbagh & Damper, 1985; Damper, 1984; Gibler & Childress, 1982; Horstmann & Levine, 1990; Rosen & Goodenough-Trepagnier, 1989; Vanderheiden, 1988). Accurate simulations of user performance can potentially aid developers and clinicians with:

- System design, by predicting the effect of system characteristics on performance;
- System recommendation, by simulating the expected performance of a particular user with candidate systems; and
- System delivery and training, by simulating how system configuration and strategy of use might affect performance.

Our work to date has focused on word prediction systems. The goal of this paper is to use model simulations to address some longstanding questions about user performance with word prediction. The broad question to be addressed is under what conditions will word prediction provide faster text generation rate than letters-only typing? Conditions to be examined include the characteristics of the user, the strategy used to search the word list, and the configuration of the word prediction system. The relative importance of each of these conditions in determining text generation rate will also be discussed.

BACKGROUND

An important prerequisite to realizing the potential benefits of a modeling approach is model validation. The accuracy of model simulation results relative to actual performance must first be demonstrated before the model can be applied to new situations with confidence. Early model simulations of user performance with word prediction were limited because they were never validated against actual performance (Horstmann & Levine, 1990, 1992; Newell, Arnott, & Waller, 1992).

To guard against this limitation, the model presented in this paper has been tested quite extensively within a pool of 14 subjects (Koester, 1994; Koester & Levine, 1994, 1995, 1997). In the model validation study, a simple model of user performance was developed and tested against the actual performance of eight able-bodied (AB) and six spinal cord injured (SCI) subjects. Given the search strategy to be employed and measured estimates of the user's keypress and list search times, model error for simulations of overall text generation rate with word prediction averaged 9.2% with a 95% confidence interval of 5.2, 13.2. Model error was not significantly influenced by strategy or SCI. These results provide confidence that text generation rate can be simulated with an error of less than 10% when the search strategy to be employed is known and measured estimates of the user's keypress time and list search are available.

The accuracy of these model simulations provides some confidence that the model can help determine when word prediction will provide faster text genera-

617 407 1009

tion rate than letters-only typing. The key factors represented in the model are the characteristics of the user, the strategy used to search the word list, and the configuration of the word prediction system. While many types of simulations are possible, we present examples of simulations across each of these dimensions to illustrate the basic approach.

SIMULATION METHODS

Model Equations

Equations for the text generation rates with letters-only typing and word prediction were derived as follows. Text generation rate with letters-only typing (TGR_{lo}), expressed in characters per minute, is simply:

$$TGR_{lo} = 60/t_k \text{ char/min}, \quad (1)$$

where t_k is the user's keypress time during letters-only typing.

To model word prediction use, the time the user requires to search the word list (t_s) must be considered, in addition to the keypress time during word prediction use. The average time necessary to generate each character is modeled as the sum of two components: the number of list searches per character multiplied by the search time and the number of keypresses per character multiplied by the keypress time during word prediction use.[1] This is expressed in equation form as follows:

$$T_{wp} = (S)(t_s) + (1\text{-ksav})((t_k)_{wp}) \quad \text{sec/char}, \quad (2)$$

where S is the number of list searches per character, 1-ksav is the number of keypresses per character, t_s is the search time, $(t_k)_{wp}$ is the keypress time during word prediction use.

The model must also account for the empirical observation that keypress time during use of word prediction can be significantly slower than during letters-only typing (Koester & Levine, 1994, 1996). This "keypress delay" effect is represented by an additional user parameter, d, defined as the amount of extra time required for the keypress action during use of word prediction relative to letters only, so that:

$$(t_k)_{wp} = t_k + d \text{ seconds}. \quad (3)$$

Substituting Equation 3 into Equation 2, the resulting equation for text generation rate with word prediction (TGR_{wp}) is:

$$TGR_{wp} = 60/T_{wp} = \frac{60}{(S)(t_s) + (1\text{-ksav})(t_k + d)} \text{ char/min}. \quad (4)$$

As can be seen from Equation 4, text generation rate is a multidimensional space, which makes it difficult to determine the unique effects of each parameter. As shown in Table 1, three parameters (t_s, t_k, d) define the characteristics of the user, and two parameters (S, ksav) are used to describe the word prediction system. The user parameters are independent of each other, and the two system parameters are jointly determined by the system configuration and the search strategy used. To keep the simulations manageable and interpretable, parameters for the example simulations presented here were constrained as defined below.

Simulation Constraints

User Characteristics

Measurements from the model validation study provide the primary basis for defining a range of expected user parameter values (Koester & Levine, 1994, 1996). Given the empirically observed ranges of 0.2 to 1.6 seconds for search time and 0.4 to 1.0 seconds for keypress time, a range of 0 to 2 seconds was chosen for both t_s and t_k. For keypress delay (d), the range for these simulations is 0 to 0.3 seconds, which is quite similar to the empirically observed range, although several instances of d larger than 0.4 seconds were observed. Note that these ranges, while empirically observed in small numbers of subjects, are not all-inclusive, as values for some actual users certainly could exceed these range limits.

Two "user types" have been defined for simulations when it is necessary to fix user characteristics while varying another condition of interest, such as keystroke savings. These user types are based on the average empirical measurements observed in AB and SCI subjects at the end of a seven-session protocol, rounded to the nearest 50 milliseconds (Koester & Levine, 1994, 1996). The values are shown in Table 2. Note that these user types are intended only to repre-

TABLE 1: **Notation and Definition of User System Parameters Used in the Performance Model of Equation 4**

Parameter Type	*Symbol*	*Definition*
User	t_s	List search time
	t_k	Keypress time
	d	Keypress delay
System	S	Searches per character
	ksav	Keystroke savings

[1]The number of keypresses per character is expressed as 1-ksav, where ksav is the keystroke savings provided by the word prediction system, expressed as a proportion rather than a percent.

TABLE 2: **User Parameter Values for Two Defined User Types**

	User Type	
Parameter (sec)	*AB*	*SCI*
List Search, t_s	0.50	1.10
Keypress, t_k	0.80	0.50
Keypress delay, d	0.05	0.20

User AB is based on performance of AB subjects, and user SCI is based on performance of subjects with SCI, after seven experimental sessions (Koester & Levine, 1994, 1996).

sent the two broad clusters of user performance, rather than being rigid expectations of parameter values characterizing any given user.

Word Prediction Strategies

Six text entry methods were examined: letters-only and five word prediction strategies. Strategies are of interest because they represent a practical means of influencing user performance. In previous work, strategy has been observed to affect the improvement in text generation rate as well as the cognitive cost associated with the use of word prediction (Koester & Levine, 1994, 1996). Model simulations provide an opportunity to further explore those empirical findings by investigating a variety of strategies and their relative performance across a range of conditions. Five different search strategies were examined, as defined below:

- Always-search (AS)—search the list before each selection;
- 1-then-search (1S)—select one letter, then search the list before each subsequent selection;
- 2-then-search (2S)—select two letters, then search before subsequent selections;
- 3-then-search (3S)—select three letters, then search before subsequent selections; and
- 4-then-search (4S)—select four letters, then search before subsequent selections.

Two variations of each word prediction strategy were studied. The "simple" variation involves rigid adherence to the search rules, unless the word list is empty. The "smart" variation is slightly more sophisticated, in that the search is discontinued if the word is not found by the fifth letter.

System Configurations

Three system configurations were examined, representing "low," "average," and "high" keystroke savings. "Average" is approximately the average keystroke savings reported for commercial word prediction systems (42%) (Higginbotham, 1992), and "high" and "low" are deviations of roughly 10 percentage points from average. System parameters (S, ksav) for all variations of the word prediction strategies under each level of keystroke savings were computed through the use of software developed for that purpose. This software simulated the entry of the supplied text using either word prediction with a given search strategy rule or letters-only typing. "Low," "average," and "high" keystroke savings were represented by texts from the validation study (Koester & Levine, 1994, 1996). The resulting parameters for "simple" search are shown in Table 3, with those for "smart" search in Table 4.

Performance Profiles

The performance profile was the main tool used to determine the relative performance of the six text entry methods examined (five word prediction strategies plus letters-only typing). In the profile, shaded regions for each text entry method show the combinations of user characteristics (t_k, t_s, and d) for which that method gives better performance than the other five.

Performance profiles were generated by simulating the text generation rates (Equations 1 and 4) for each strategy and letters-only typing across the defined range of user keypress and list search times. The resulting rates were then compared over the space of user parameter values to determine the fastest method in each region of the space. A mathematical

TABLE 3: **Word Prediction System Parameters Under "Simple" Search Conditions**

	Low		*Average*		*High*	
Strategy	*S*	*(1-ksav)*	*S*	*(1-ksav)*	*S*	*(1-ksav)*
Always-search	0.542	0.682	0.518	0.587	0.457	0.475
1-then-search	0.390	0.705	0.368	0.617	0.316	0.510
2-then-search	0.258	0.765	0.251	0.701	0.194	0.585
3-then-search	0.169	0.854	0.156	0.790	0.128	0.687
4-then-search	0.080	0.931	0.099	0.862	0.090	0.790

S = number of searches per character generated; ksav = keystroke savings.

TABLE 4: **Word Prediction System Parameters Under "Smart" Search Conditions**

	Low		*Average*		*High*	
Strategy	*S*	*(1-ksav)*	*S*	*(1-ksav)*	*S*	*(1-ksav)*
Always-search	0.501	0.682	0.476	0.587	0.442	0.475
1-then-search	0.350	0.705	0.326	0.617	0.301	0.510
2-then-search	0.218	0.765	0.210	0.701	0.179	0.585
3-then-search	0.129	0.854	0.114	0.790	0.113	0.687
4-then-search	0.040	0.931	0.057	0.862	0.075	0.790

S = number of searches per character generated; ksav = keystroke savings.

software package called MatLab[2] was used to generate these performance profiles.

SIMULATION RESULTS

Core samples of the simulation graphs that were produced are presented below. For each, examples of how the graph could be used are discussed, and general trends are identified where possible.

"Baseline" Performance Profile

Figure 1 shows an example performance profile, referred to as the "baseline" profile since it represents simple search under average keystroke savings. The profile also assumes no keypress delay, which defines a target that might be achievable with long-term word prediction use.

Each of the six wedges in the profile represents the region in (t_k, t_s) where a particular text entry method gives better performance than the other five. For example, for any combination of t_k and t_s that falls in the "AS" wedge, the "always-search" strategy is predicted to give faster performance than either letters-only typing or any of the other word prediction strategies examined. According to the profile, a user whose keypress time is 0.8 seconds would need a search time of less than 0.2 seconds in order for "always-search" to be the preferred strategy. A user with a keypress time of 0.8 seconds and a search time of 0.5 seconds should use the "1-then-search" strategy for the best performance. The "1-then-search" strategy has the biggest region in which it is dominant, which suggests it may be a good, all-purpose strategy for users with a variety of characteristics. The "4-then-search" strategy has the smallest region, so it is applicable to the fewest number of users. These examples illustrate how the profile can provide prescriptions for specific word prediction strategies.

It is also possible to use the profile to make more general comparisons between letters only and word prediction. The region labeled "LO" shows where letters only provides better performance than any of the five word prediction strategies. The remaining shaded region is where some form of word prediction use will give the best performance. For a user with t_k = 0.8 seconds, letters only should be used if t_s exceeds 1.1 seconds (under the assumption of no keypress delay). As keypress time gets faster, the cut-off search time also decreases; for t_k = 0.5 seconds, for example, search time would have to be below 0.65 seconds for some form of word prediction to be beneficial.

Impact of User Parameters

The baseline profile provides valuable information on how list search and keypress time affect relative performance. However, it assumes that the keypress time during word prediction use is the same as during letters-only typing (i.e., d = 0). However, significant keypress delay has been observed consistently across subjects (Koester & Levine, 1994, 1996), so it

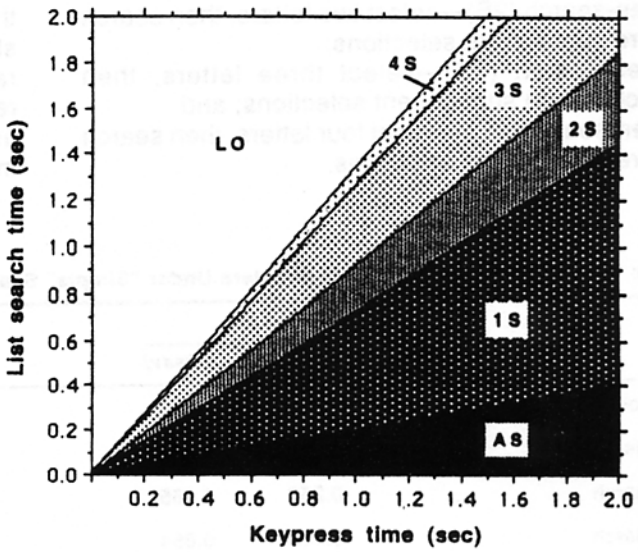

Figure 1. "Baseline" performance profile, showing regions where each of the five word prediction strategies and letters-only typing is predicted to provide the best performance. Average keystroke savings, simple search, and no keypress delay are assumed.

[2]The Math Works, Inc., 24 Prime Park Way, Natick, MA, USA 01760.

is important to be able to simulate its effect. Model simulations show that the region in which word prediction provides better performance than letters-only typing shrinks with increasing keypress delay. Even a keypress delay as small as 50 milliseconds (d = 0.05 sec) has a significant impact. A user with t_k = 0.8 seconds and d = 0.05 seconds must have a search time faster than 0.8 seconds for word prediction to give better performance, as compared to 1.1 seconds when d = 0.

A keypress delay of 0.2 seconds has an even more dramatic impact, as illustrated in Figure 2. The region in which word prediction is better than letters only has shrunk substantially, to the point where a user with t_k = 0.8 seconds must have a list search time below 0.45 seconds for word prediction to be better than letters only. And for a user with t_k = 0.5 seconds, word prediction under these conditions would not generally be a practical option, since search time must be less than 0.2 seconds.

The "1-then-search" strategy continues to have a large, favored region even for larger keypress delays. But the profile provides no information about how the absolute rate achieved is affected by keypress delay and other user parameter values. Figure 3 illustrates how the expected improvement in text generation rate with word prediction relative to letters-only typing depends on the user's keypress and list search times. Average keystroke savings, simple search, and a keypress delay of 0.1 seconds are assumed.

Based on Figure 3, a user with a keypress time of 1.2 seconds, a list search time of 0.5 seconds, and a keypress delay of 0.1 seconds should be able to generate text over 20% faster with word prediction as compared to letters only. If this same user's search time were actually 1.1 seconds, there would be no predicted benefit to using word prediction relative to letters only.

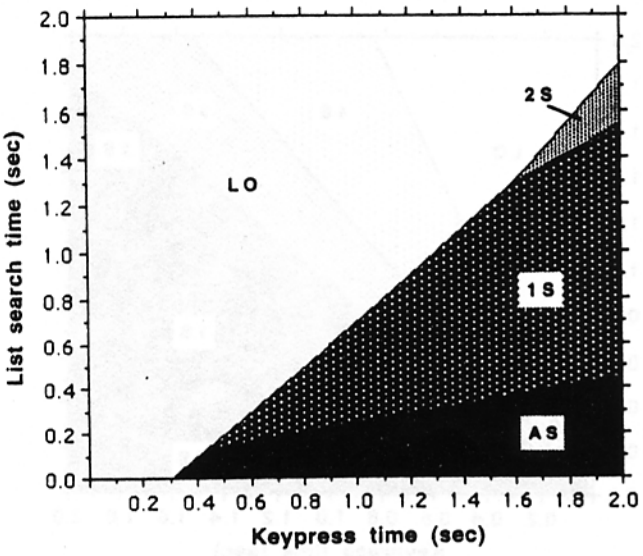

Figure 2. Performance profiles with keypress delay set at 0.2 seconds. Average keystroke savings and simple search are assumed.

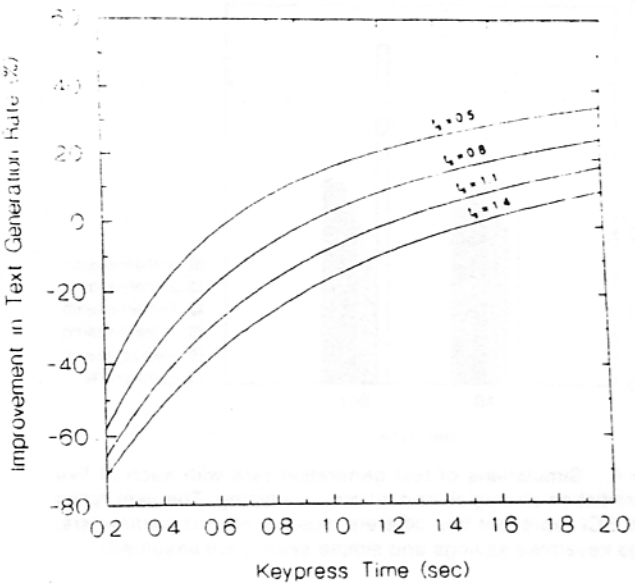

Figure 3. Predicted improvement in text generation rate with word prediction relative to letters-only typing, as a function of keypress time. The "1-then-search" strategy, with average keystroke savings, simple search, and a keypress delay of 0.1 seconds, is assumed. Each curve represents a fixed level of list search time t_s in seconds.

Impact of Search Strategy

Relative Advantage of Strategies

The performance profiles give detailed information on when each of the five word prediction strategies is better than the others, but they do not answer the question of how much difference choosing the best strategy really makes. The answer for any particular set of user and system characteristics can be found graphically, as shown in the example in Figure 4. This example assumes simple search and average keystroke savings. For the AB user type (with t_k = 0.8, t_s = 0.5, and d = 0.05 sec), all five of the tested word prediction strategies are predicted to provide better performance than letters-only typing, with the "1-then-search" strategy predicted to be the best. This strategy is predicted to be 10 characters per minute (cpm), or 13%, faster than letters only, and 8 cpm, or about 10%, faster than the slowest word prediction strategy, which is "4-then-search." "Always-search" is predicted to have the next to worst performance, at 6.5% slower than "1-then-search," even though its search rule is quite similar to "1-then-search." Finally, the predicted difference between "1-then-search" and the next best, "2-then-search," is only 1.5 cpm, or 1.8%.

For the SCI user type (t_k = 0.5, t_s = 1.1, d = 0.2 sec), the situation is quite different (see the right half

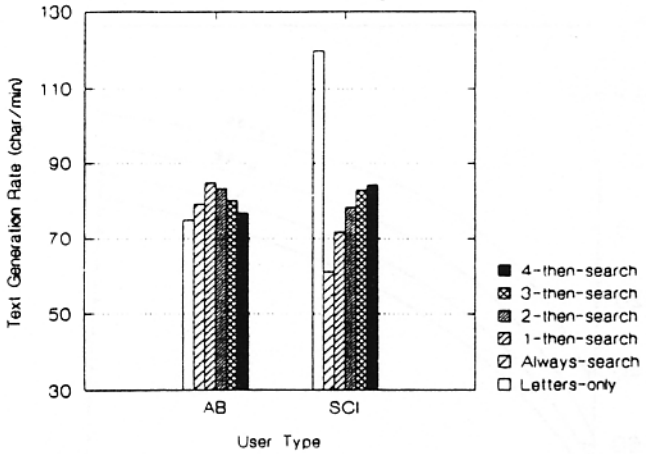

Figure 4. Simulations of text generation rate with each of five word prediction strategies and letters-only typing. The user types AB and SCI represent two different clusters of user parameters. Average keystroke savings and simple search are assumed.

of Fig. 4). None of the word prediction strategies is predicted to provide an advantage over letters-only typing, which is simulated at 30% faster than the fastest word prediction strategy. There is also a large difference between strategies, up to 23 cpm (or 27%) between the best ("4-then-search") and worst ("always-search") strategies. The difference between best and second-best ("3-then-search") is much smaller, at 1.4 cpm (or 1.6%).

Given the differences in the effect of strategy for these two examples, it is difficult to draw general conclusions about the impact of word prediction strategy. The difference between the best and worst strategy can be appreciable, and while the difference between best and second best is usually fairly small, it is not always obvious what the second-best strategy is. Since strategy may matter a fair amount in many cases, it is most sensible to simply use the dominant strategy recommended by the performance profile. If a precise answer regarding the relative advantage of the dominant strategy is desired, it can be obtained by constructing a graph like the one in Figure 4.

Smart vs. Simple Search

The simulations for word prediction strategies presented above assumed "simple" search, or rigid adherence to the strategy rules. In contrast, "smart" search recognizes that if a word has not been found by the fifth letter, the chances of it being found are very low (in fact, the chances are zero for the texts used in this study). "Smart" search amends the rules for any strategy by dictating that if a word is not found in the list by the fifth letter, it should be spelled out with no further list searches. The benefit of this search variation is that it reduces the number of unsuccessful searches made for words that are not in the dictionary. A potential drawback is a small loss of keystroke savings, when the assumption that the word will not appear in the list after the fifth letter proves incorrect. For the texts in this study, however, this assumption turns out to be correct 100% of the time.

The following model simulations estimate how smart search might affect overall performance. Figure 5 shows a performance profile with smart search, at average keystroke savings and no keypress delay. Comparison to the baseline profile of Figure 1 shows that the region in which some strategy of word prediction is expected to outperform letters-only typing has greatly expanded. For example, for a user with a keypress time (t_k) of 0.8 seconds, the maximum search time for word prediction to provide rate enhancement has increased from 1.1 seconds under simple search to 1.9 seconds.

The expanded region is due to an increase in the size of the wedge corresponding to the "4-then-search" strategy. Wedge size for the other strategies is basically the same for simple and smart search, which suggests that smart search provides the same amount of performance increase for each strategy. In other words, there is no interaction between smart search and word prediction strategy, so the strategy effects discussed above hold for both simple and smart search conditions.

The benefit of smart search decreases substantially as keypress delay becomes a factor, as shown in Figure 6. Comparison with Figure 2 illustrates that when d = 0.2 seconds, the region in which word prediction provides better performance than letters only is only slightly larger with smart than simple search.

The expected increase in absolute performance with smart search is modest. Calculating the expected rates shows that for the "1-then-search" strategy and an AB type user, smart search increases text generation rate by 2.6 cpm (3%) over simple search. The increase for

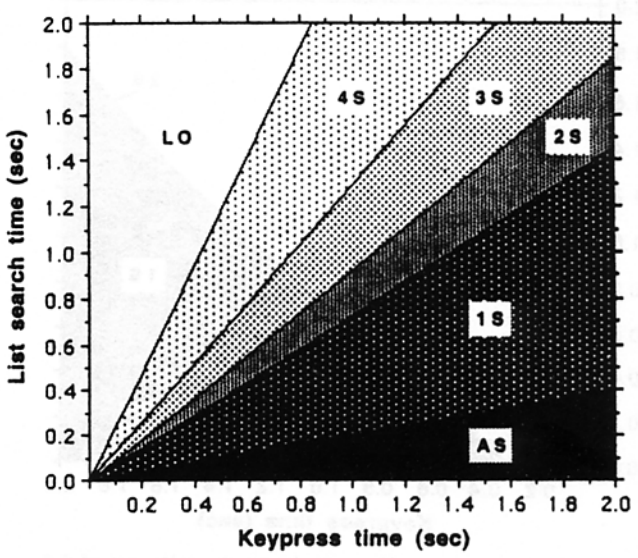

Figure 5. Performance profile with smart search, assuming average keystroke savings and no keypress delay.

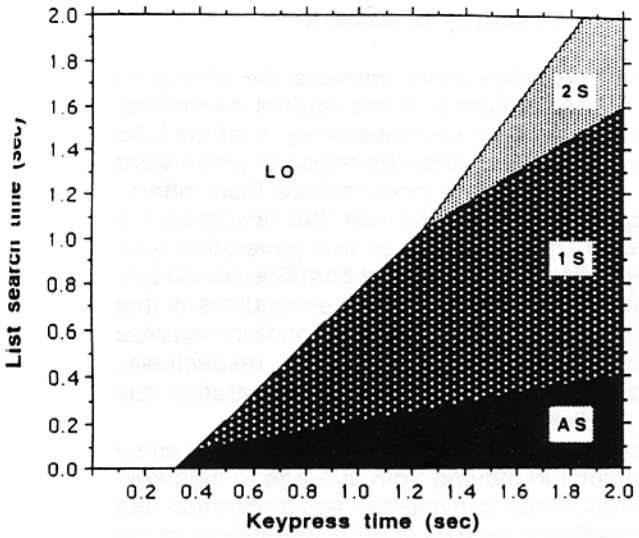

Figure 6. Performance profile with smart search, assuming average keystroke savings and d = 0.2 seconds.

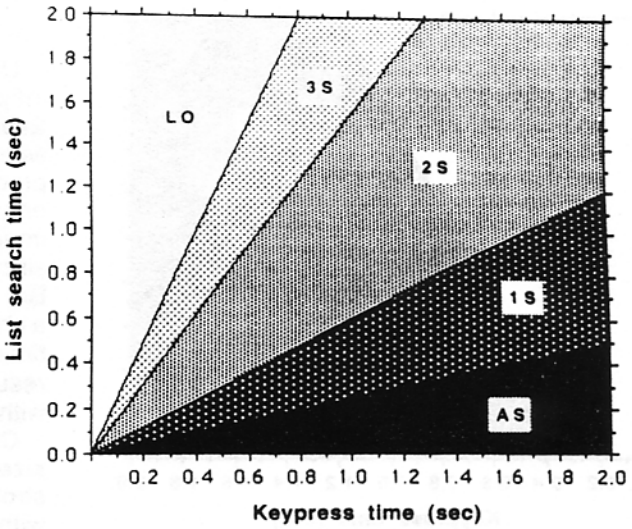

Figure 8. Performance profile for high keystroke savings. Simple search and no keypress delay are assumed.

an SCI-type user is 4.2 cpm (6%). However, for users who can comply with the additional search rule, it does provide a fairly painless way to gain some speed. Additionally, use of smart search does not affect the choice of the best strategy, since it affects all strategies equally.

Impact of System Keystroke Savings

Figures 7 and 8 show the performance profiles for low and high keystroke savings, respectively, under the assumptions of simple search and no keypress delay. Recall that with average keystroke savings, a user with t_k = 0.8 seconds must have a search time faster than 1.1 seconds for some form of word prediction to be faster than letters-only typing. For low keystroke savings, the maximum search time decreases to 0.7 seconds, while for high keystroke savings, it increases to 2.0 seconds.

The factor of keypress delay plays a significant mediating role on the effect of keystroke savings. In Figure 8, the steepness of the boundary between the letters-only and word prediction region supports an optimistic prediction that almost any user will benefit from word prediction. The figure shows that for a user with t_k = 0.5 seconds, word prediction is expected to provide better performance than letters only for t_s faster than 1.1 seconds under high keystroke savings conditions. However, as shown in Figure 9, with a keypress delay of 0.2 seconds, this same user must now have a t_s faster than 0.4 seconds for word prediction to be faster than letters-only typing. For users with slower keypress times, of course, there remains a large space in which word prediction is predicted to be faster.

Figure 10 illustrates the text generation rates predicted for the five word prediction strategies at each level of keystroke savings for the AB and SCI user types. For both types of users, high keystroke savings (average + 10%) has a larger effect than low (average − 10%), relative to average keystroke savings. The "2-then-search" strategy is the most sensitive to higher keystroke savings, resulting in 18 cpm (22%) improvements for both the AB and SCI user types, respectively. The "4-then-search" strategy benefits the least from increased keystroke savings, yielding a 7 cpm (9.2%) and 8 cpm (9.2%) improvement for the AB and SCI user types, respectively.

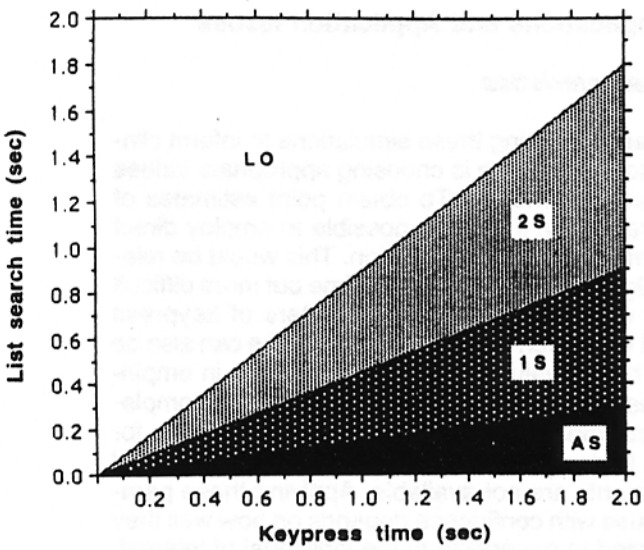

Figure 7. Performance profile for low keystroke savings. Simple search and no keypress delay are assumed.

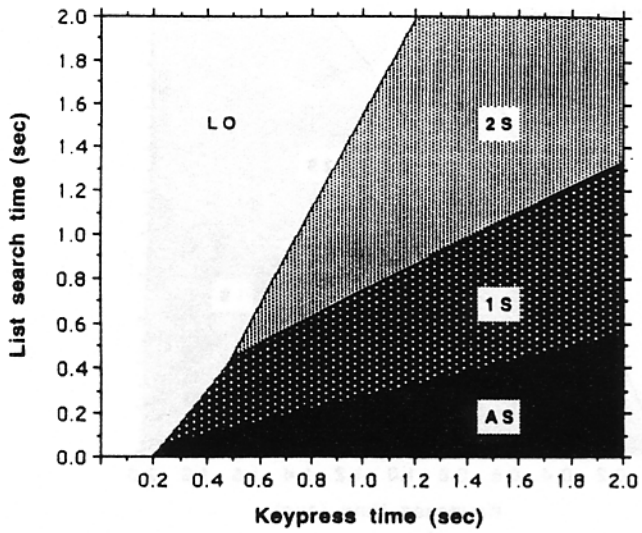

Figure 9. Performance profile for high keystroke savings, d = 0.2 seconds. Simple search is assumed.

DISCUSSION

The model simulations presented here estimate the effects of three categories of usage conditions on user performance with word prediction: user characteristics, search strategy, and system characteristics. The specific results and their implications are discussed below, with the caution that they apply most directly to the range of conditions and the type of word prediction system examined here. Several questions that were not explicitly addressed in these simulations are also considered, and while the answers are largely speculative, it should be noted that these simulation techniques provide a tool that can be used to provide more specific answers, by using different parameter values to represent the condition of interest.

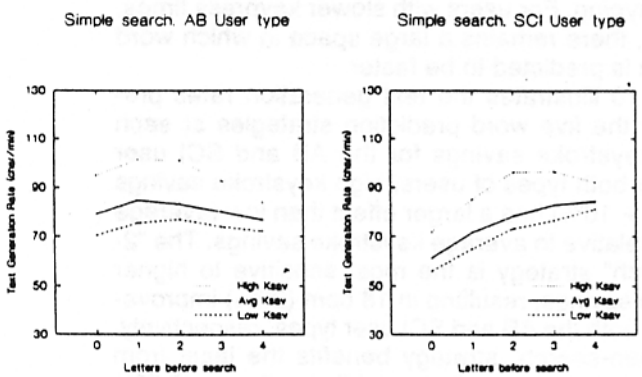

Figure 10. Text generation rates predicted for the five word prediction strategies at each level of keystroke savings for the AB and SCI user types.

Summary of Effects

User characteristics have perhaps the strongest influence on performance of the factors examined. Even a small amount of keypress delay, such as 0.05 seconds, significantly shrinks the region in which word prediction provides better performance than letters-only typing. There is no fixed rule that describes the impact of user characteristics on text generation rate, since it depends on the choice of baseline conditions. But a general guideline from these simulations is that a change of either 0.25, 0.3, or 0.6 seconds in keypress time, keypress delay, or list search time, respectively, results in a 10% to 30% change in text generation rate with word prediction.

Changes in keystroke savings have a similar effect size, particularly in moving from average to high keystroke savings, which can improve text generation rate with word prediction by 10% to 30%, depending on the search strategy used. High keystroke savings by itself does not guarantee success with word prediction for all users, however. In Figure 9, for example, there is a fairly large region in which letters-only typing provides better performance than word prediction. But higher keystroke savings does greatly expand the region in which word prediction is best, particularly for users with moderate to slow keypress times.

Finally, strategy of use with word prediction *can* have a substantial effect on performance, but again the size of the effect depends on other conditions, such as the user characteristics. There are generally at least two strategies that give roughly the same performance for a given individual, but the difference between text generation rate with best and worst strategies can be as high as 30%. Using a "smart search" variation with the chosen strategy can yield a modest boost in performance, about 3% to 6%.

Implications and Application Issues

User Characteristics

A challenge in using these simulations to inform clinical and design practice is choosing appropriate values for the user parameters. To obtain point estimates of these parameters, it may be possible to employ direct measurement in a clinical situation. This would be relatively straightforward for keypress time but more difficult and time consuming for the parameters of keypress delay and list search time. Parameter choice can also be informed by the values reported for subjects in empirical studies (Koester & Levine, 1994, 1996), to complement direct measurement and to provide a basis for modeling in design situations where specific individual measurements are not available. Applying these parameter values with confidence depends on how well they are expected to generalize to the individual of interest. In Koester and Levine (1994), for example, two different clusters of user parameter values were found, one gen-

erally describing the AB subjects, who were novice mouthstick typists, and one for the SCI subjects, who were proficient mouthstick or hand splint typists. In applying these values to other individuals, a careful decision must be made as to which cluster, if any, represents the individual with reasonable accuracy. This issue will continue to complicate the process of applying these values in new situations until the source of the difference is more clearly understood and parameter measurements are made for a wider range of potential users.

Parameter values reported from empirical studies at least provide a range of possible values and reveal the need for caution in assuming homogeneity among users. Additionally, the uncertainty in obtaining accurate point estimates for each parameter matters less in some situations than others. For example, across all of the performance profiles presented above, any user with a keypress time slower than about 1.5 seconds should do better with word prediction than letters-only typing, except for those with unusually slow search times. If a user's keypress time is measured directly and found to be above 1.5 seconds, accuracy in estimating search time and keypress delay becomes less important to the decision between word prediction and letters-only typing.

Uncertainty also exists in estimating dynamic changes in user parameter values, such as those due to cyclic factors like fatigue or long-term improvements with practice. Again, parameters measured across time in empirical studies provide some limited guidelines regarding expected improvements. For example, Koester and Levine (1994) reported that list search time for the SCI subjects improved an average of only 3% over seven test sessions. The performance profiles provide a way to estimate the improvement that would be required before word prediction would become faster than letters-only typing. For example, for the SCI user type, if it is assumed that keypress delay could improve to d = 0.05 seconds, while maintaining a keypress time of 0.5 seconds, search time with average keystroke savings would have to improve from the observed 1.1 seconds to less than 0.4 seconds for word prediction to provide a rate advantage over letters-only typing. While further empirical work is necessary to provide more insight into whether this improvement would be possible, the empirical results suggest that it may not be easy to achieve. The goal of improving parameter values may make more sense for users whose parameters even as novices make word prediction the preferred choice. Practice could be relied on as one potential mechanism for improvement, with the advantage that the user would not have to practice on a system that initially provides relatively poor performance (as would be the case with the SCI-type subjects).

Beyond practice effects, there may be clinical and/or design interventions that facilitate improvement in user parameter values. For keypress time, clinical interventions might include choice of the most appropriate mouthstick or typing splint, proper positioning of the keyboard, or proper seating to stabilize the rest of the body during typing. Design interventions might include more efficient keyboard layouts (Chubon, 1988) or flexible location of the keys used to make selections from the list. Improving keypress delay may be more challenging, especially since its source is not entirely clear. Simple coaching in rules about when to search the list may help those decisions become more automatic and therefore faster. From a design standpoint, it might help to display the list only when it is supposed to be searched, providing less distraction during keypress-only selections. Although data on the efficacy of these approaches requires future empirical work, the model simulations provide confidence that even a moderate improvement in keypress-related parameters will provide a meaningful boost in performance.

Regarding list search time, analyses of search time during use of word prediction suggest that anticipation and serial search both play a role in the search process (Koester, 1994; Koester & Levine, 1997). Anticipation of the list contents would be expected to become more successful with practice, shortening list search time, and there may also be ways of actively facilitating its development. For example, the use of fixed prediction lists, as compared to dynamic lists, may be important in allowing the user to gradually develop expectations about the list contents. There may also be ways to reduce reliance on serial search in favor of more direct search methods. For example, imposing alphabetical order on the list might decrease reliance on serial search, as compared to the common practice of ordering words by their frequency. Or, color coding the list based on syntax (e.g., blue for verbs, green for nouns) might improve list search time for some users.

System Characteristics

Since the model simulations above showed that user performance was relatively sensitive to the keystroke savings provided by word prediction, several potential means of influencing keystroke savings and their possible effects on performance are discussed below. The discussion is largely speculative since it touches on factors that have not been explicitly investigated here from either an empirical or modeling perspective. It should be noted that in the empirical study that formed the basis for this modeling work, keystroke savings was manipulated by adjusting the text to be transcribed, rather than by modifying the word prediction system itself (Koester & Levine, 1994, 1996). This meant that the search task remained basically the same regardless of the keystroke savings provided. In real-world attempts to influence keystroke savings, the possibility of affecting the search task must be considered.

A common method of enhancing keystroke savings is to employ an adaptive dictionary that continuously updates the frequency information for each word to adapt to the user's linguistic style and also uses the recency of a word's use in making its predictions

(Swiffin, Arnott, Pickering, & Newell, 1987). One effect of an adaptive dictionary is that the list contents are dynamic. For example, before any letter is entered, the list would contain the words most likely to follow the previous word, instead of the most likely words in English overall. In comparison to a system that uses fixed word lists, dynamic lists provide more keystroke savings (Swiffin et al., 1987), but they may also limit the ability of the user to anticipate the list contents, leading to longer search times. To model this trade-off accurately, more information is needed regarding the expected increase in keystroke savings and list search time.

A second means of increasing keystroke savings is to use syntax information to offer more grammatically appropriate predictions (Swiffin et al., 1987). This algorithm has been found to increase keystroke savings by about 5 percentage points (Swiffin et al., 1987). Given the sensitivity of performance to increases in keystroke savings, this would be expected to lead to a noticeable improvement in text generation rate if list search time remained constant. Again, further work is necessary to test these hypotheses about expected performance.

A third general strategy to influence keystroke savings is to manipulate the number of words presented in the prediction list (i.e., list length). Increasing the list length increases keystroke savings but might also lead to changes in list search time. The simulation results suggest that when keystroke savings increases from "average" to "high" (about 10 percentage points), the allowable search time for word prediction to be faster than letters only increases from 1.1 to 1.9 seconds for a user with a keypress time of 0.8 seconds. Previous analyses of list search suggest that each additional word would add 150 milliseconds to the list search time (Koester, 1994; Koester & Levine, 1997). This provides for an increase of up to 5 additional words, or 11 total. The question is then whether an 11-word list would increase keystroke savings more than 10 percentage points. The exact relationship between keystroke savings and list size depends on the system implementation, but, based on the results of Swiffin et al. (1987), the increase in keystroke savings is quite flat in the range of 6 to 11 words, at roughly 2 percentage points improvement. This suggests that increasing the list length will not provide a keystroke savings large enough to counteract the increase in required search time. This conclusion is empirically supported by a study in which text generation rate with word prediction was no different for list lengths of 5, 10, and 15 words (Venkatagiri, 1994).

In the opposite direction, decreasing the list length has the effect of decreasing keystroke savings and would also be expected to decrease the list search time. As above, the net trade-off between these two effects dictates whether or not overall performance will improve. According to Swiffin et al. (1987), reducing the list from six to five words has almost no effect on keystroke savings. Reducing the list from six to five words could improve list search time by 150 milliseconds, during unsuccessful searches (Koester, 1994; Koester & Levine, 1997). This would result in a modest improvement in performance, assuming no loss in keystroke savings. The actual loss in keystroke savings would have to be calculated from simulations and balanced against the faster search time to determine the overall effect.

A more dramatic approach is to consider the effect of a list that contained only one word. Swiffin et al. (1987) suggest that this would cause a drop of only about 10 percentage points in keystroke savings, similar to the decrease from "average" to "low" in the simulations above. For a user with a keypress time of 0.8 seconds, the allowable search time for word prediction to be faster than letters only is 0.7 seconds for "low" keystroke savings. The list search time for a one-word list would almost certainly be faster than 0.7 seconds, which suggests that the one-word list approach might be more effective than letters only for a wide range of users. Indeed, the list search time might be as low as 0.2 seconds with this approach, since evaluating the single word is essentially a simple pattern match, which would more than compensate for the loss in keystroke savings. It would be intriguing to test these possibilities empirically.

Strategy of Use

While finding ways to accelerate improvement in user parameters or provide substantial increases in keystroke savings is fairly challenging, strategy of use is relatively easy to manipulate. Ideally, the appropriate performance profile should be used to select the best strategy, in order to avoid the possibility of a costly mistake, especially since strategy is one of the few factors that a user or clinician can control. In the absence of good information about the user or system configuration, a strategy such as "1-then-search" could be chosen, since it generally has a large dominance region. Whatever strategy is chosen, the model simulations suggest that smart search can provide a means of modest performance enhancement.

There are two main ways of manipulating strategy of use. First, users can be taught appropriate strategy rules, as was done successfully in the empirical study. However, depending on the individual and the complexity of the rules, strategy compliance may be more difficult if the word list is continually present. The second means of manipulating strategy addresses this potential problem by having the system facilitate strategy compliance. The system could be configured to display the list only after a defined number of letters have been entered, so to support a "1-then-search" strategy, the list would be hidden until the user entered the first letter of a word. This feature is provided by at least one commercial word prediction system,[3] and is recommended as an enhancement to those systems

[3] Co:Writer®, Don Johnston Developmental Equipment, Inc., 1000 N. Rand Road, Wauconda, IL, USA 60084.

that do not have this feature. A "smart search" type of strategy could be supported similarly by configuring the system to hide the list after a defined number of letters have been entered, although commercial systems do not currently provide this option.

Configuring the system to match a particular strategy does restrict the user's freedom to search the list at any time, but this restriction may actually be an advantage. Providing a user with more than one method to achieve a goal can actually decrease performance, as the user may spend considerable time deciding which method to use (Olson & Nilsen, 1988). Therefore, if a strategy that is likely to provide the best performance can be identified, the system should be configured to support that strategy.

Limitations

It should be noted that these simulation results (and others based on this model) apply with the greatest confidence to conditions similar to those under which the model was validated. Further work is necessary to determine the extent to which the modeling approach is valid for users who have variable motor control and/or cognitive impairments, natural strategies of system use, and systems other than word prediction.

CONCLUSIONS

The simulation graphs presented here represent a family of predictions regarding performance with word prediction relative to letters-only typing across a wide range of usage conditions. The results highlight the importance of keypress delay as a factor in user performance and provide a better understanding of the role of other user parameters, search strategies, and keystroke savings. These simulations have led to many new questions regarding performance that could be addressed through a combination of new model simulations and empirical research. This illustrates that modeling does not replace empirical work, but it may influence and complement the type of empirical work that is done.

ACKNOWLEDGMENTS

This work was supported by the National Science Foundation, The University of Michigan Rackham School of Graduate Studies, and the U-M Rehabilitation Engineering Program. This paper is based in part on a doctoral dissertation entitled "User Performance with Augmentative Communication Systems: Measurements and Models," written by the first author in partial fulfillment of the requirements for a Ph.D. degree in bioengineering at the University of Michigan. Simon P. Levine was the chair of the dissertation committee. Preliminary results of this work were presented at the RESNA '95 Annual Conference.

Address reprint requests to: Heidi Horstmann Koester, 368 Oak Harbor Court, Holland, MI 49424, USA; E-mail: hhk@umich.edu.

REFERENCES

Chubon, R. A. (1988). An inexpensive, off-the-shelf approach to increasing computer keyboard entry rate for single finger and typing stick typists. In *Proceedings of ICAART 88* (pp. 368–369). Arlington, VA: RESNA Press.

Dabbagh, H. H., & Damper, R. I. (1985). Average selection length and time as predictors of communication rate. *Proceedings of the 8th Annual RESNA Conference* (pp. 404–406). Washington, DC: RESNA.

Damper, R. I. (1984). Text composition by the physically disabled: A rate prediction model for scanning input. *Applied Ergonomics, 15*, 289–296.

Gibler, C. D., & Childress, D. S. (1982). Language anticipation with a computer-based scanning communication aid. *Proceedings of the IEEE Computer Society Workshop on Computing to Aid the Handicapped* (pp. 11–15). Charlottesville, VA: IEEE.

Higginbotham, D. J. (1992). Evaluation of keystroke savings across five assistive communication technologies. *Augmentative and Alternative Communication, 8*, 258–272.

Horstmann, H. M., & Levine, S. P. (1990). Modeling of user performance with computer access and augmentative communication systems for handicapped people. *Augmentative and Alternative Communication, 6*, 231–241.

Horstmann, H. M., & Levine, S. P. (1992). Modeling AAC user performance: Response to Newell, Arnott, and Waller. *Augmentative and Alternative Communication, 8*, 92–97.

Koester, H. H. (1994). *User performance with augmentative communication systems: Measurements and models.* Doctoral dissertation, Bioengineering Program, University of Michigan, Ann Arbor, Michigan.

Koester, H. H., & Levine, S. P. (1994). Modeling the speed of text entry with a word prediction interface. *IEEE Transactions on Rehabilitation Engineering, 2*, 177–187.

Koester, H. H., & Levine, S. P. (1995). Validating quantitative models of user performance with word prediction systems. In A. Langton (Ed.), *Proceedings of the RESNA '95 Annual Conference* (pp. 127–129). Arlington, VA: RESNA Press.

Koester, H. H., & Levine, S. P. (1996). The effect of a word prediction feature on user performance. *Augmentative and Alternative Communication, 12*, 155–168.

Koester, H. H., & Levine, S. P. (1997). Keystroke-level models for user performance with word prediction. *Augmentative and Alternative Communication, 13*, 239–257.

Newell, A. F., Arnott, J., Waller, A. (1992). On the validity of user-modeling in AAC: Comments on Horstmann and Levine (1990). *Augmentative and Alternative Communication, 8*, 89–92.

Olson, J. R., & Nilsen, E. (1988). Analysis of the cognition involved in spreadsheet software interaction. *Human Computer Interaction. 3*, 309–350.

Rosen, M., & Goodenough-Trepagnier, C. (1989). The Tufts-MIT prescription guide: Assessment of users to predict the suitability of augmentative communication devices. *Assistive Technology, 1*, 51–61.

Swiffin, A. L., Arnott, J. L., Pickering, J. A., & Newell, A. F. (1987). Adaptive and predictive techniques in a communication prosthesis. *Augmentative and Alternative Communication, 3*, 181–191.

Vanderheiden, G. C. (1988). A unified quantitative modeling approach for selection-based augmentative communication systems. In L. Bernstein (Ed.), *The vocally impaired* (pp. 40–83). Philadelphia: Grune and Stratton.

Venkatagiri, H. S. (1994). Effect of window size on rate of communication in a lexical prediction AAC system. *Augmentative and Alternative Communication, 10*, 105–112.