Linguistically Optimized Text Entry on a Mobile Phone

*submitted to CHI 2001

Hedy Kober[1], Eugene Skepner[1] , Terry Jones[1], Howard Gutowitz[1,2], and Scott MacKenzie[3]

[1]Eatoni Ergonomics, Inc.
171 Madison Avenue
New York, New York 10016
USA
+1 212 569 0262
{hedy,eu,terry}@eatoni.com

[2] Département de Mathématiques
Ecole Supérieure de Physique et de Chimie Industrielles de la Ville de Paris
10, rue Vauquelin
Paris V, France 75005
(on leave)
hag@eatoni.com

[3]Dept. of Mathematics & Statistics
York University
Toronto, Ontario, Canada M3J 1P3
+1 416 736 2100
smackenzie@acm.org

ABSTRACT

We present an analysis of linguistically optimized text entry on mobile phones. This analysis compares the behavior of a linguistically optimized system with word-based disambiguation methods. Through theoretical analysis, it is shown that in real-world situations in which typing errors are common and dictionaries are incomplete, the speed of text-entry using a word-guessing method degrades to the speed of a multi-press method. Linguistically optimized text entry speed, however, decays gracefully in the face of significant noise, and remains close to those of unambiguous text-entry methods.

Keywords

Mobile computing, text entry, linguistic optimization, Fitts' law, input devices.

INTRODUCTION

A central reason to explore and examine the efficacy of telephone keypad text entry is the growth of Short Message Services (SMS) messages on mobile phones, especially in Europe. According to the GSM association, 3 billion SMS messages were sent in December 1999, with a prediction of 10 billion messages a month by the end of 2000 [8]. Currently, the majority of SMS users use multi-press as their method of text entry, although some cellular phones do feature word-based disambiguation systems, including T9® Text Input ("T9®") by Tegic Communications, Inc., iTAP™ Intelligent Keypad Entry System by Motorola, Inc., and eZiText™ by Zi Corporation.

This growing phenomenon, accompanied by a greater demand for mobile internet access, calls for efficient means of text entry to encourage further growth in the mobile internet by addressing user needs. In this paper, we present an overview of the development and expected performance of word-based disambiguation mechanisms and of a linguistically optimized text entry mechanism. Their efficacy will be determined as a function of correct word throughput, as a function of typing speed, and typing error rates.

The Evolution of Text Entry on Cell Phones

On a typical telephone keypad, groups of letters in alphabetical order are associated with number keys. For example, "a," "b," and "c," are typically associated with number 2. Thus, any single press of a key is ambiguous, as it may represent any of the associated sets of three or four letters.

Early text-entry methodology concentrated on explicitly disambiguated entries: two-key input (chording) and multi-pressing. Two-key entry methods activate a combination of keys, simultaneously or in sequence, to encode each symbol unambiguously. Such systems require the user to press the key associated with the desired letter, and then follow with a second key to specify the position of the letter on the first key. The second key is usually one of the keys on the top row: for example, to enter "g", a user would press 4 followed by 1 [5,10]. Multi-press works differently. It requires multiple taps on the same key to disambiguate an entry: the user taps the key the number of times corresponding to the position of the letter in the standard ordering. For example on the 2 key, the user taps once for "a," twice for "b", etc. [2,3]. Although these methods offer perfectly unambiguous entries, they involve the significant disadvantage of requiring more than one keystroke per letter, which results in cumbersome and laborious typing.

In view of this perceived disadvantage, and since ease of use and typing speed are essential components of effective text entry, methods enabling one keystroke per letter emerged as early as the 1970s, using the standard ambiguous code and a database of stored responses, represented by their numerical sequences. Early testing and implementations of such systems established their accuracy [16], and ease of use [13].In recent years, dictionary-based disambiguation mechanisms have appeared in various forms, often aided by N-gram frequencies, syntactical information, or other statistical information of letter and word frequencies [1, 9, 11, 14]. When used, such systems compare the numerical code of an entry, which is treated as a discrete unit, with those found in a database, and guess the intended letters or words. However, many dictionary words share the same numerical code, and in these cases the system will present alternatives in a list (a query). The user selects the intended word from the list. This requires extra taps for the word entry to be correct and complete.

Nevertheless, even if dictionary words are correctly disambiguated with just a few extra taps, a greater problem is created in practice because it is necessary to allow entry of non-dictionary words, which are in everyday use (e.g., proper names, slang, abbreviations, technical and professional terms, etc.). For example:

a collection of text from the 1988 Wall Street Journal containing 20,691,239 words, was found by James Raymond Davis [4] to contain not only 8,633,941 ambiguous words, but also 4,007,375 words which were not in Webster's seventh dictionary, to which it was compared.

With a perfect dictionary, multi-press methods and word-guessing methods are points on a continuum. For both, extra keystrokes beyond one per intended letter are required for a word to appear correctly. In the case of multi-press, the extra keystrokes are entered throughout the word to select each intended letter, while in a word-guessing method the extra keystrokes all occur at the end of the word. Delaying the extra keystrokes until the end of the word has the advantage of reducing the total number of keystrokes which must be entered. It has the disadvantage of causing the display to be unstable as algorithms typically present their current best guess based on partial information after each keystroke is entered.

With an imperfect dictionary, word-guessing methods fail catastrophically on many words which are not in the dictionary. Since a word guessing algorithm cannot match a word not in its dictionary, it resorts to default rules which do not involve complete words in order to make a guess. When the underlying code is the highly ambiguous standard ambiguous code, these default rules will typically render letter sequences which have little relationship with the intended letter sequence.

In some implementations of word-guessing systems, such as Tegic Communications' implementation on the Ericsson model 280, a failure of the word-guessing algorithm places the phone into multi-press mode so that the word can be re-entered unambiguously using multi-press. This solution enables users to enter any letter sequence, but slows typing speed significantly.

Measuring Ambiguity

A system's ambiguity, or its efficacy in disambiguating entries, can be measured in at least two ways: the query rate and the lookup error rate. Both are easy to measure given a complete list of words from a language, their numeric equivalents, and their probabilities.

The *query rate* measures how often the same keystroke pattern yields multiple words. It is calculated as the reciprocal sum of all probabilities of all the words with identical codes. Using the standard ambiguous code, a query occurs, on average, every three words, which at an average typing speed of 20 wpm, means every 9 seconds.

The *lookup error rate* measures how often the desired word is not the first in the list of alternatives in a query. It is calculated as the reciprocal sum of probabilities of words in

queries, except the first. Using the standard ambiguous code, a lookup error occurs every 28 words.

Clearly, queries slow typing speed by demanding attention and cognitive processing time from the user. Lookup errors exact the same demands, in addition to requiring extra taps to select the right word from the list.

In this paper, we present an optimized system, which reduces the number of queries and lookup errors significantly, thus eliminating most of the distractions that slow typing speed.

Linguistically Optimal Predictive Text Entry (LOPT)

Optimal coding is in some ways the opposite of cryptography. While cryptography seeks codes that are as difficult as possible to decipher, the optimal coding method seeks codes which are as easy as possible to decipher.

To overcome the main difficulties with the standard ambiguous code, we have developed an optimal coding method for text entry on a telephone keypad. The method substantially improves on all previous text entry methods, as it was designed to optimize primarily the query rate and lookup error rate with respect to other ergonomic criteria, thus creating a touch typable device requiring little attention from the user and offering fewer distractions.

The idea behind LOPT is that some of the letter entries are unambiguous. These letters are entered unambiguously using an auxiliary key, much like a shift key on the standard Qwerty keyboard. And so, the system is an evolved hybrid of chording and ambiguous coding.

The essential insight of the chording aspect of this invention is that substantially simultaneous activation of a pair of keys is unified into a single gesture. Thus, a pair of keystrokes is no more, or only a little more difficult to master than a single keystroke. Yet, a pair of keystrokes contains substantially more information than a single keystroke. Thus, chording creates easily operable, low-ambiguity codes.

Although selection of letter pairs to be entered unambiguously on some keys was considered, it was rejected in light of ergonomic criteria such as learnability and ease of use, because a regular and predictable division of letters into shift and non-shift sets of letters makes the keyboard easier to learn. One could consider more complicated partitions in which each set contains irregular sized groups of letters, but this would help very little to reduce ambiguity. Further, only on keys which contain more than three letters would irregularities be possible. There are only two such keys on the telephone keypad (the 7 key and the 9 key). Therefore, a single letter from each key was selected, in the following way: Query rates and Lookup error rates were calculated for all 11,664 possible combinations of letters. The "cloud" graph in Figure 1 shows the results:

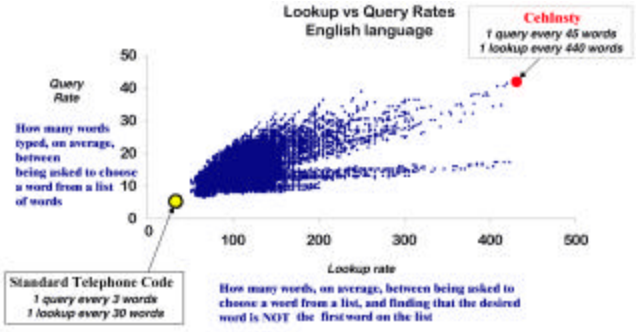

Figure 1: Query vs. Lookup error rates for English.

Each point on the graph corresponds to the results for a distinct letter combination. As clearly seen, the letter combination providing the optimal (lowest) rates of both queries and lookup errors in English (and many other languages in fact) is C, E, H, L, N, S, T, and Y.

Figure 2: A CEHLNSTY labeled mobile keypad.

Using this selection of letters as a base, we have constructed a disambiguation system called WordWise™ [i]. Using WordWise™, fully 45% of all letters entered are entirely unambiguous, and so many words such as "yes," "style," "cheese," "sentence," and "senselessness" can be entered completely unambiguously. This effect helps reduce queries. In WordWise™, queries occur only every 45 words, which at an average typing speed of 20 wpm means only once in two minutes of typing. Lookup errors occur every 440 words, or once in every two pages of typed text. These rates are 15 times better than those achieved by the standard ambiguous code.

ANALYSING TYPING SPEED

Fitts' law has provided good quantitative estimates of task execution speeds in a variety of contexts [6,12]. In the present instance, Fitts' law relates typing speed to the distance traveled by the fingers. Silfverberg, MacKenzie, and Korhonen [15], recently provided estimates of text entry speed using a standard telephone keypad. They compared several input techniques, multi-press, two-key input, and the T9® text input system, for each using one-

hand thumb input and two-handed index finger input. Their estimates are built upon three assumptions, limiting their examination to users' motor performance alone:

1) There were no typing, spelling, or other errors requiring time and effort to correct.
2) All words entered are unambiguous.
3) All words typed are included in the dictionary.

These three hypotheses will be referred to as the *perfect typist hypothesis*, the *no ambiguity hypothesis*, and the *perfect dictionary hypothesis*, respectively. Under these hypotheses, Fitts' law, combined with a linguistic probability model, predicts that word-based disambiguation will provide expert users with the capability to type at rates of 41 or 46 wpm, for one-handed thumb input and two-handed index finger input respectively, while multi-press will allow text entry only at rates of 25 or 27 wpm, even in the hands of expert users.

The No Ambiguity Hypothesis

The validity of the assumption that there is no ambiguity and thus no queries depends highly on the code used. The standard ambiguous code contains significant ambiguity, as will be shown in this section.

Figure 3 shows the probability of queries with a given number of words in a query. The graph is based on the 100 million words in the British National Corpus. Results are shown for the linguistically optimized system WordWise™, and a system based on the standard ambiguous code, such as the T9® text input system :

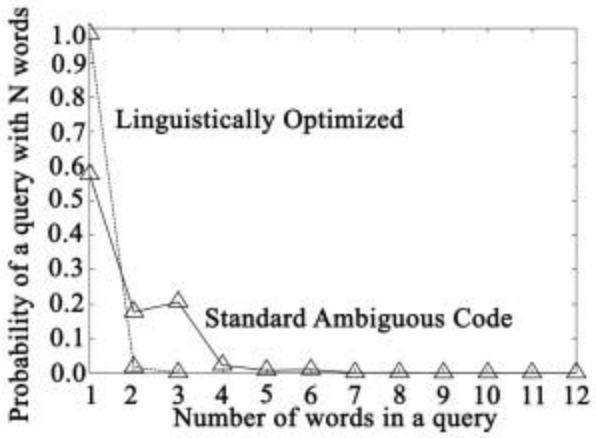

Figure 3: Occurrence probabilities for queries of differing sizes.

For the word-based disambiguation mechanism, there is about a 20% chance of having a query with either 2 or 3 words, which might require extra taps. Non-zero probabilities exist for queries with up to 12 words. With WordWise™, however, the probability for a query of 2 words is close to zero, and there are never queries with more than 3 alternatives.

The prediction of 41-46 wpm for the word-guessing method, based on the aforementioned hypotheses, was calculated for a single keystroke, a single movement, per letter, disregarding the potential need for additional keystrokes. However, in light of these data, text entry in such a system would require, on average, 1.03 keystrokes per letter.

Silfverberg et al. briefly examined the effects of queries on typing speed. Their original model ignores factors that might further decrease text entry speed in empirical tests, other than ambiguity itself, such as the time (i) to visually verify input; (ii) to cognitively register that activation of the disambiguation mechanism is required; and (iii) to visually scan the list, register its contents, and verify a selection. Estimating that users visually verify input 50% of the time, and that each inspection takes 500 ms, they modified their prediction for the T9® text input system to 35 wpm.

By contrast, we consider the no ambiguity hypothesis accurate for WordWise™. The low probability of queries enables the linguistically optimized system to require only 1.002 keystrokes per letter, a rate which virtually relieves the user from the need to visually verify input. In accordance with Fitts' law, this leads to a predicted typing speed similar to that of the word-guessing mechanism, before the modification of the no ambiguity hypothesis.

Further analysis of the no ambiguity hypothesis, and analysis of the perfect dictionary hypothesis is reserved for future discussion, as will empirical studies. In the present report, we present an expanded theoretical analysis of the perfect typist hypothesis, and its effects on text entry speed.

The Perfect Typist Hypothesis

Even highly trained professional typists do not type perfectly. An extensive literature exists on the variety of typing errors and their rates [7].

Typing errors on mobile phone keypads can be expected to be more frequent than those on Qwerty keyboards, since telephone keypads are conducive to typing errors due to their small and difficult-to-operate keys.

Typing Error Amplification

When typing on an unambiguous keyboard, such as a Qwerty keyboard, a one-keystroke typing error results in a one-letter difference in the displayed text. By contrast, when using an ambiguous keyboard, such as the standard telephone keypad or the linguistically optimized keypad, a one-stroke typing error may result in many letters being different in the typed text. Disambiguation works by using context to choose the letter to display. If the context is altered in one place, it can generally affect the letters displayed in many places. Potentially, a single keystroke error can affect the entire word. We call this *typing error amplification*.

In general, the more ambiguous the code, the more a single typing error will amplify into multiple errors. The amplification of typing errors for ambiguous keyboards implies that the perfect typist hypothesis demands close examination. To evaluate the importance and suitability of the perfect typist hypothesis, we must have some way to

measure the effect of imperfections in typing on disambiguation mechanisms.

Since a disambiguation mechanism correlates letters within a word, multiple typing errors within a single word may be correlated in their effects. The correlations make it very difficult to supply a full analysis, in which typing errors of any kind, including multiple errors within a single word, are considered. Some approximations must be made. This will be undertaken in the next section.

A Mean Field Theory of Typing Errors

In physics, problems involving complicated correlations are often attacked using mean field theory. In mean field theory, correlations between sites are decoupled. The result is a good approximation provided correlations are small. In the present case, correlations between typing errors are small if the probability of making a single typing error is small. If so, then the probability of making more than one keystroke error within a given word is small and there can be little correlation. For single keystroke error rates in the range studied here (0% to 10%), we expect the mean field approximation to be very good.

The procedure for building a mean field approximation is as follows:

1) Construct the set of possible 1-keystroke typing errors.
2) Weight all possible 1-keystroke typing errors equally.
3) Determine the average effect of a one-keystroke error.
4) Use this average effect to calculate the expected amplification of typing errors.

Possible typing errors

To model the way typing errors are made on a telephone keypad, we assume the following:

1) Typing errors are due to hitting keys adjacent to the intended key, either horizontally or vertically.
2) All ways of making typing errors occur with equal probability.

Thus, we will not consider double typing errors (a keystroke is mistakenly repeated), inversion errors (two keystrokes are reversed), insertion or deletion errors (a keystroke is spuriously inserted or omitted), etc. Inclusion of these types of errors complicates the analysis, but does not change the conclusions.

Given these assumptions, we can say that for each word w, there are M_w ways of mistyping the word with a one-keystroke error.

For instance, the word "so" is typed using the standard ambiguous code with the key sequence 76. There is one key vertically adjacent to the 7 key: the 4 key, which corresponds to the letters g, h, and i. There is one key horizontally adjacent to the 7 key: the 8 key, corresponding to the letters t, u, and v. Similarly, there are two keys vertically adjacent to the 6 key: the 3 key, corresponding to the letters d, e, and f; and the 9 key, corresponding to the letters w, x, y, and z. There is one key horizontally adjacent to the 6 key: the 5 key, corresponding to the letters j, k, and l. Each adjacent key might be mistakenly hit. The possible key combinations in a mistyping are: 46 and 86, where the first keystroke is in error, and, 73, 75, and 79, where the second keystroke is in error. In the T9® system, these keystroke combinations give rise to the letter combinations "in", "to","re", "pl", and "ry" respectively. The difference between the letters intended and the letters displayed will be referred to as display errors. The number of display errors for these mistypings is: 2, 1, 2, 2, and 2, respectively. The average display error over all of these mistypings is 1.8. We call this average number the *sensitivity* of the word "so" under T9®.

To discuss the corresponding calculation for WordWise™, we indicate the unambiguous shifted letters in bold. Thus, the word "so" is written "**s**o". The mistypings for "**s**o" are: "**h**o", "**t**o", "**s**f", "**s**k", and "**s**w". In each case, there is only one display error. Thus the average is 1, and the sensitivity of the word "so" under WordWise™ is 1, S_w=1.

Continuing to compute the sensitivity of all words in the BNC corpus in the same way, for both T9® and WordWise™, we obtain the distributions shown in Figure 4.

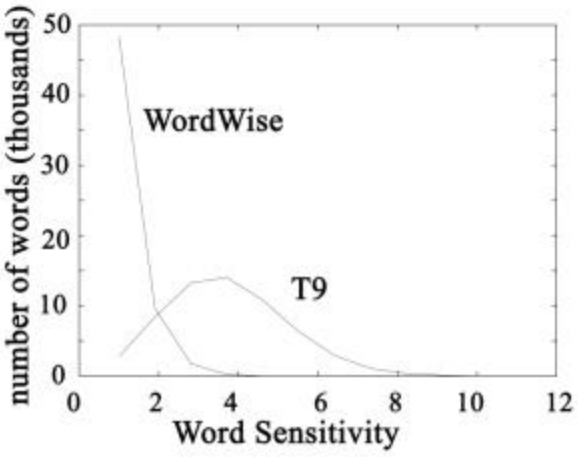

Figure 4: Distribution of word sensitivities for WordWise™ and T9®.

We see that word sensitivities for T9® peak around 4, while most words have a sensitivity at or near 1 for WordWise™. Since the average word length for English is about 5.5 letters, these data imply that single keystroke errors often cause display errors throughout the word.

Average Sensitivity

By averaging with respect to the probability of words, we can compute the sensitivity as a function of the length of words, and also, the average sensitivity of the language, S. For WordWise™, the average sensitivity is approximately 1.27, meaning that, on average, a single keystroke error will result in 1.27 letter differences in the displayed text. For the word-guessing algorithm, the sensitivity, S, is 2.6. The sensitivity as a function of word length is shown in the Figure 5, for both disambiguation mechanisms.

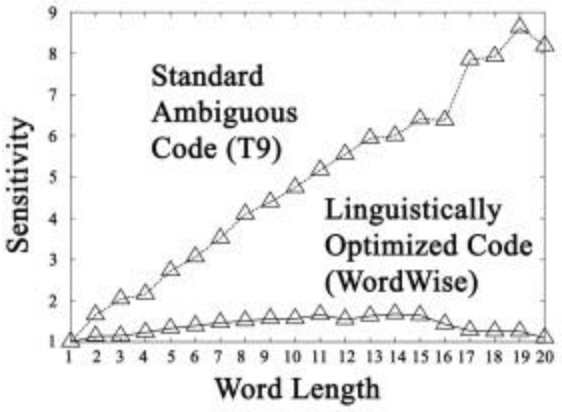

Figure 5: A comparison of the relationship between word length and sensitivity in WordWise ã and T9 â .

This figure shows that for WordWise™ the sensitivity is small, and relatively independent of word length. For the word-guessing algorithm, by contrast, the sensitivity grows linearly with the length of the word. This means that most single-keystroke errors are catastrophic: that is, they propagate through the entire word. This behavior is expected, since a single keystroke error is sufficient to ensure that no match is found in the word-guessing dictionary. Since no appropriate match can be found, the likelihood that any of the letters displayed correspond to the intended word is small. With WordWise™, however, errors remain bounded within a word. This is due to the fact that roughly every other letter, on average, is an unambiguously entered letter (c, e, h, l, n, s, t, and y). For these letters, the display does not depend on other keystrokes. Furthermore, the unambiguous letters provide a solid framework to contain errors.

We are now in a position to connect the new concept of sensitivity with the established concept of throughput.

Throughput

Throughput, as defined here, is the rate at which correct words are produced by a typist. For a perfect typist, the throughput is the same as the typing speed. For an imperfect typist, however, throughput is determined not only by raw typing speed, but also by the rate of production of errors, and the means for correcting errors.

On a mechanical typewriter, the only way to correct a word is to backspace over the word and retype it. The extra keystrokes have a significant effect on throughput. To improve throughput, computerized word processing software provides a single-keystroke mechanism to erase the preceding word.

With the computerized mechanism, correction of a mistyped word of length N requires $(N + 1)$ keystrokes, 1 to delete the word, and N to retype the word. An even faster mechanism would be to skip the erasure step, and simply retype the word. With this error correction method, the throughput is simply the typing speed times the probability that a word contains no errors. We use this idealized correction method in the following analysis. By doing so, we unrealistically reduce the effect of typing errors on throughput. Nonetheless, we will see that these effects are quite substantial.

Dependence of Throughput on Keystroke Error Probability

The probability of producing a word needing correction — that is, the probability of producing a word with at least one display error — depends on the keystroke error probability, and the sensitivity of the disambiguation mechanism.

If the keystroke error probability is p, and an unambiguous keyboard is used, then the probability that there are no display errors in a word is given by:

$$C_0 = (1 - p)^N$$

where N is the length of the word. If the text entry method is ambiguous, then the probability of a display error is p times S. The probability of typing a word with no display errors is:

$$C_0 = (1 - Sp)^N .$$

In each case, the throughput T is $T = (C_0 * \text{speed})$, where speed is the raw typing speed.

The throughput for Qwerty, Multi-press, the T9® input system, and WordWise™ is plotted in Figure 6. In the no ambiguity case, the raw typing speed is taken as 46 words per minute for all methods of text entry, except multi-press., which has a speed of 27 words per minute under the perfect typist hypothesis. As we have discussed, Qwerty, WordWise™, and Multi-press are not substantially affected by ambiguity. However, in the case of T9®, Silfverberg et al. argue that ambiguity can drive the effective typing speed for T9® down from 46 to 35 words per minute. Thus, for T9® we have plotted two curves, one, labeled "T9® (no ambiguity)" assumes a raw typing speed of 46 words per minute, the other, labeled "T9® (with ambiguity)" assumes a typing speed adjusted for ambiguity of 35 words per minute.

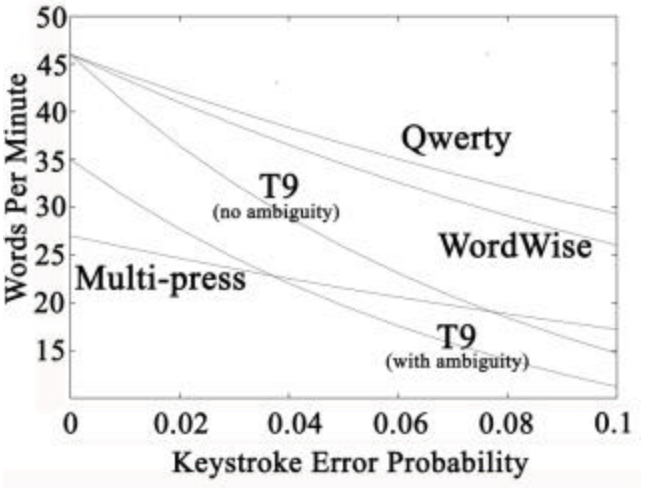

Figure 6: The degradation of throughput with increasing per-letter error probability, for a perfect

dictionary and, in the case of T9â , either with no ambiguity or with ambiguity. Qwerty, WordWiseã , and Multi-press are substantially unaffected by ambiguity.

We see that the larger the keystroke error probability, the larger the difference between Qwerty and the T9® text input system, while Qwerty and WordWise™ remain close throughout the range of keystroke error probabilities considered. With no ambiguity, the performance of T9® degrades to that of multi-press at a keystroke error rate of about 0.08. When both ambiguity and imperfections in typing are taken into account, the performance of T9® degrades to that of multi-press at a keystroke error rate of around 0.04. This is a keystroke error rate which is commonly observed in practice [7].

CONCLUSIONS

Throughput in a disambiguation system is a function of many factors beyond raw typing speed. We have shown that due to the typing error amplification property, throughput in real-world application of a disambiguation system may be significantly less than the throughput achieved by a perfect typist.

When the relaxation of the perfect typist hypothesis is combined with relaxation of the no ambiguity hypothesis, we can expect that, at any reasonable rate of typing errors, the throughput of the T9® text input system will degrade significantly.. When the relaxation of the no ambiguity and perfect typis t hypotheses is further combined with relaxation of the perfect dictionary hypothesis, we expect the gap between Qwerty and WordWise™ on one hand, and T9® on the other to widen further. In future reports we will examine these predictions both theoretically and in empirical tests.

REFERENCES

[1] Bentley, J. The littlest keyboard. Unix Review, December, 1994.

[2] Coles, G. A. Telephone-coupled visual alphanumeric communication device for deaf persons, U.S. patent 4,191,854, March 10, 1980.

[3] Danish, A., Danish, S., and Kimbrough, K. W. *Entry of alphabetical characters into a telephone system using a conventional telephone keypad*, U.S. patent 5,392,338, February 21, 1995.

[4] Davis, J.R. Let your fingers do the spelling: Disambiguating words spelled with the telephone keypad. *Avios Journal*, *9*: 57-66, March 1991

[5] Detweiler, M. C. Schumacher R. M. Jr., Gattuso, N. L. Jr. Alphabetic input on a telephone keypad, *Proceedings of the Human Factors Society 34th Annual Meeting*, 1990.

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

[7] Gentener, D. R., Grudin, J. T., Larochelle, S., Norman, D. A., and Rumelhart, D. E. A glossary of terms including a classification of typing errors, ***Cognitive aspects of skilled typing***, ed. W. E. Cooper. (New York: Springer-Verlag, 1983) 39-44.

[8] GSM Association statistics: http://www.gsmworld.com/membership/mem_stats.html

[9] James, L. E. *Processor-assisted communication system using tone-generating telephones*, U.S. Patent 4,650,927, March 17, 1987.

[10] Knowlton, K. C. *Method and apparatus for using pushbutton telephone keys for generation of alpha-numeric information*, U.S. patent 3,967,273, June 29, 1976.

[11] Kondraske, G. V. *Character pattern recognition and communications apparatus*, U.S. Patent 4,674,112, June 16, 1987.

[12] MacKenzie, I. S. Fitts' law as a research and design tool in human-computer interaction, *Human-Computer Interaction 7* (1992), 91-139.
[13] Richie, D. Research has always had to pay its way. *Bell Labs News*, September 10, 1984.

[14] Riskin, B. N. *Method and apparatus for identifying words entered on DTMF pushbuttons*, U.S. Patent 5,031,206, July 9, 1991.

[15] Silfverberg, M., MacKenzie, I. S., and Korhonen, P. Predicting text entry speed on mobile phones, In *Proceedings of the ACM Conference on Human Factors in Computing Systems - CHI 2000*. New York: ACM, 2000, pp. 9-16.
[16] Smith, S. L., Goodwin, N. C. Alphabetic data entry via the touch tone pad: A comment. *Human Factors*, *13*(2) 189-190, 1971.

---

[i] WordWise™ is a trademark of Eatoni Ergonomics, Inc. All other trademarks referred to herein are the property of their owners.