An Efficient Text Input Method for Pen-based Computers

Toshiyuki Masui

Sony Computer Science Laboratory Inc.
3-14-13 Higashi-Gotanda
Shinagawa, Tokyo 141-0022, Japan
+81-3-5448-4380
masui@csl.sony.co.jp

ABSTRACT

Pen-based computing has not yet taken off, partly because of the lack of fast and easy text input methods. The situation is even worse for people using East Asian languages, where thousands of characters are used and handwriting recognition is extremely difficult. In this paper, we propose a new fast text input method for pen-based computers, where text is not composed by entering characters one by one, but by selecting words from a menu of candidates created by filtering the dictionary and predicting from context. Using our approach, users can enter Japanese text more than twice as fast as recognition-based and other existing text input methods. User studies and detailed analysis of the method are also given.

KEYWORDS: Input devices, Pen-based input, Predictive interface, Hand-held devices, International interfaces, POBox

INTRODUCTION

Although a variety of pen-based computers are available these days, they are not as widely used as keyboard-based computers, partly because entering text is much harder on pen-based machines. Traditionally, handwriting recognition techniques and the soft keyboard (virtual keyboard displayed on the tablet of a pen computer) used to be the main techniques for entering characters on pen-based computers, although other techniques have also been proposed[4][6]. However, using any of these techniques takes much longer to enter text than with a standard keyboard.

The situation is worse for East Asian languages such as Chinese, Japanese, etc. These, unlike European languages, have thousands of character faces. Even with a keyboard, it is not easy to enter a character. A variety of techniques for entering text into computer have been investigated. The most widely-used Japanese input technique is "Roman-Kanji conversion" (RKC), in which a user specifies the pronunciation of a word with an ASCII keyboard, and the system shows the user a word with the specified pronunciation[1]. If the word was not the one that the user intended to use, the user types a "next candidate key" until the correct word appears as the candidate.

On almost all the pen-based computers available in Japan, either RKC or handwriting recognition is supported. Text input is slow and tiring using either of the techniques, for the following reasons. Specifying the pronunciation of every input word using a soft keyboard takes a lot of time, and the user must convert the pronunciation to the desired Kanji strings with extra keystrokes. Handwriting recognition has more problems. First, the recognizer has to distinguish between thousands of characters, often making errors. Many of the characters in the character sets have similar shapes, so it is inherently difficult to make recognition reliable. Second, in many cases, users do not remember the shape or the stroke order of Kanji characters, even when they have no problem reading them. Finally, writing many characters with many strokes on a tablet is very tiring. With these difficulties, it is believed to be difficult to enter Japanese text faster than 30 characters a minute on pen-based computers, which is several times slower than using keyboards.

We have developed a new pen-based text input method called *POBox* (Pen-Operation Based On eXample), where users can efficiently enter text in any language, using menus, word prediction and approximate pattern matching. The remainder of this paper demonstrates the details of POBox.

STRATEGIES FOR RAPID TEXT ENTRY

There is a big difference between the speed of typing on keyboards and pointing to characters on soft keyboards of pen-based computers. Computer users can easily type more than five characters per second, while it is very difficult to touch three character keys per second, accurately on the soft keyboard of a pen-based computer. In contrast, the speed of selecting an item from a list is faster with a pointing device, and many keyboard-oriented text editors (e.g. Emacs) now have mouse interfaces. For this reason, forcing the user to enter many characters should be avoided on pen-based computers, while a better approach should allow the user to select a word from a list of candidates, in a minimum number of penstrokes. We took the following approach.

[1]Japanese characters consist of two character sets. Kanji characters, imported from China, contain both meaning and pronunciation, while Kana characters only represent pronunciation.

CHI 98 Los Angeles CA USA

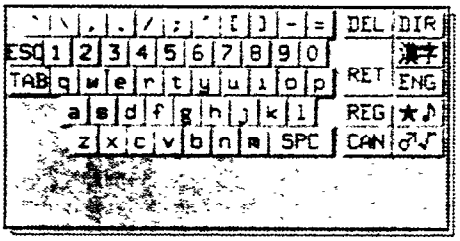

Figure 1: Initial display.

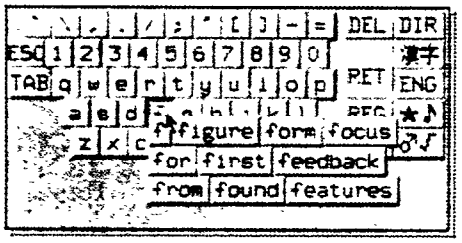

Figure 2: Selecting the "F" key.

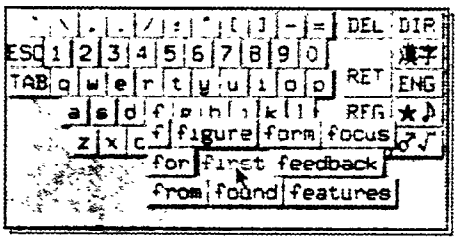

Figure 3: Selecting "first" by dragging.

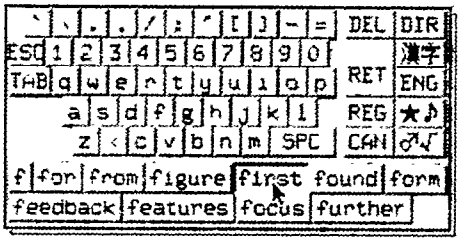

Figure 4: Selecting "first" after releasing the pen from the tablet.

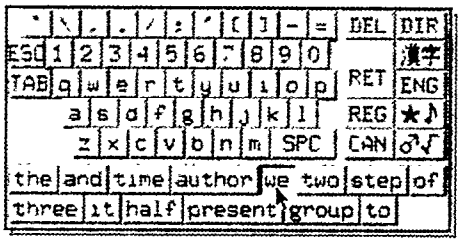

Figure 5: Selecting "we" after selecting "first".

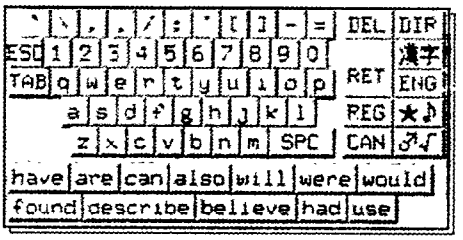

Figure 6: After selecting "we".

Using dynamic menus to show candidates and select words: The desired word can be picked up directly from a pulldown or popup menu.

Dynamic query for dictionary search: As soon as the user specifies a portion of the pronunciation or the spelling of a word using the soft keyboard on the tablet, POBox shows a menu of candidate words that match the input.

Using term frequency and example phrases: The words which are most likely to appear at the insertion point in the text are shown at the top of the menu. The likelihood is calculated from the term frequency and context. For example, since the word "interface" tends to come after "user," it appears at the top of the menu after the user has selected "i" as the first character following "user."

Dynamic approximate string matching for selecting candidate words: If the pattern specified by the user does not exactly match any of the words in the dictionary, POBox automatically performs approximate string search based on the following two strategies. One is *spatial approximation*, where adjacent characters on the soft keyboard are treated equally in the search. This strategy is effective especially when the soft keyboard is small and precise selection is difficult. For example, if the user failed to tap the right position of a soft keyboard and selected "**dtns**" to enter "**dynamic**," no word in the dictionary matches "**dtns**" and POBox automatically searches the dictionary using the less strict pattern "**[ersdfxc] [rtyfg] [hjbnm] [weasdzx]**," based on the arrangement of ASCII keyboard. ("**d**" key is surrounded by "**e**," "**r**," "**s**," "**f**," "**x**" and "**c**" keys.) This pattern matches words like "**synergy**" and "**dynasty**," but since "**dynamic**" has higher term frequency than these words, it is shown in the candidate word list for the selection. The other is *pattern matching allowing errors*. This strategy is effective when the user does not remember the correct spelling or the pronunciation of a word. In this case, POBox automatically looks for words whose spelling or pronunciation is closest to the pattern and shows them as candidates. Users can even specify only a portion of a word to get the desired word in the candidate list.

Simple dictionary adaptation: Newly selected words are put at the top of the dictionary, and are likely to be shown at the top of the menu so that the dictionary reflects the characteristics of the current text.

EXAMPLES

Entering English Text

First, for explanatory purpose, we show how to use POBox for entering English text, although POBox is more effective for entering Japanese and other East Asian languages. We used the ACM CHI'95 Electronic Proceedings CD-ROM to create an English dictionary with term and phrase frequencies. We extracted plain text files from all the HTML files in the CD-ROM, counted the occurrences of words and word combinations, and created the dictionaries by sorting the entries by frequency order. The remainder of this section uses the sentence (*"First, we show our technique for entering English text."*) as the sample input text for our example.

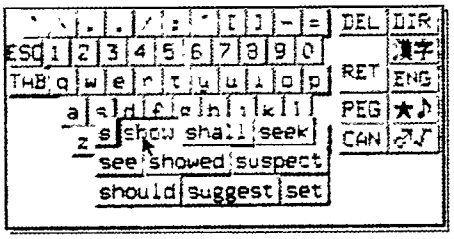

Figure 7: Selecting "show" from the menu.

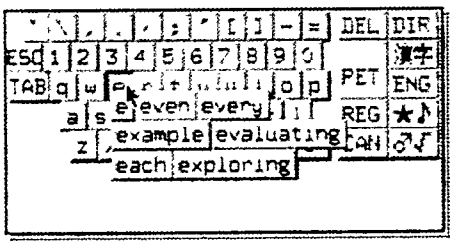

Figure 8: Selecting the "E" key.

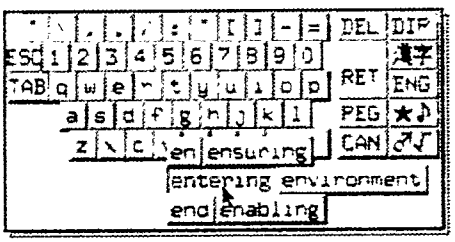

Figure 9: Moving to the "N" key and selecting "entering".

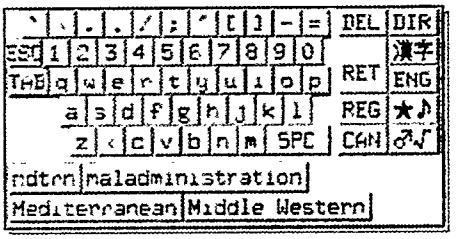

Figure 10: Specifying "mdtrn" to get "Mediterranean".

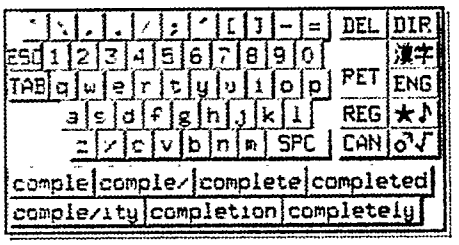

Figure 11: After specifying "comple".

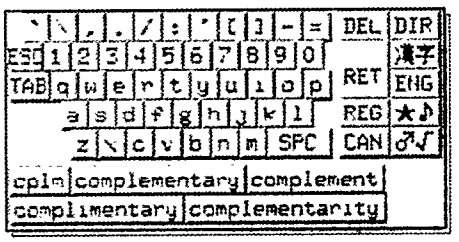

Figure 12: After specifying "cplm".

Figure 1 shows the startup display of POBox. When the user touches the "F" key, the display changes to Figure 2, showing the frequently used words that start with "F" in a pulldown menu. Since the word "first" is a frequently used word and is found in the menu, the user can drag the pen and highlight the word "first" as shown in Figure 3, and then take the pen off the tablet to complete the selection. Alternatively, if the user does not make a selection from the pulldown menu of Figure 3, he can choose the desired word from the popup menu as shown in Figure 4.

After selecting "first", the display changes to Figure 5. In the menu at the bottom, the words that often come after "first" are listed in order of frequency. The word combination "first the" appears 27 times in the CHI'95 CD-ROM, "first and" and "first time" appear 20 times, etc. Since the next word, "we," happens to be in the list because "first we" appears 13 times in the CD-ROM, the user can directly select "we" by touching it in the menu. After selecting "we", the display changes to Figure 6. In this case, "show" is not found in the menu, but it can be selected from the pulldown menu by touching the "S" key as shown in Figure 7.

After this, "our", "technique" and "for" can be selected in a similar manner. Touching the "E" key does not make the system display the next intended word ("entering") as shown in Figure 8, but touching the "N" key next narrows the search space of the dictionary and "entering" then appears in the menu for the selection (Figure 9).

From start to finish, the user only had to tap the tablet 15 times to enter the phrase "First, we show our technique for entering." Notice that the user made no spelling errors with this method, since all the input words were taken from the dictionary.

Using Approximate String Matching

With the approximate string matching feature, even when the user does not specify the correct spelling of a word, there is a good chance of finding the desired word among the candidates. Also, the user can specify only part of the spelling to find the desired word. For example, if the user does not remember the spelling of "Mediterranean," he can specify "mdtrn" to see the list of words which are close to the pattern and then can find the right word in the list (Figure 10.)

The same technique can be used to enter a word that has a common prefix. If the user tries to enter "complementary" and specifies "comple," he still cannot find the word in the candidates in Figure 11, since there are many commonly used words that begin with "comple." Instead, the user can specify the characters that better represent the word. As shown in Figure 12, the user can obtain "complementary" by specifying "cplm," although other patterns such as "cpmt" will also work.

Entering Japanese Text

With POBox, users can enter Japanese text much more easily than RKC and handwriting recognition systems. We show the example by using "以下に本手法を用いた文章入力例を

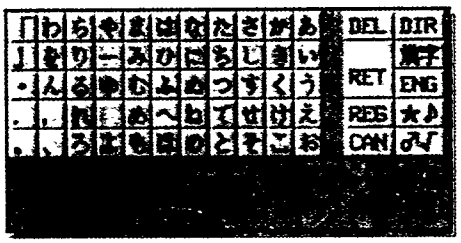

Figure 13: Initial display in Japanese input mode.

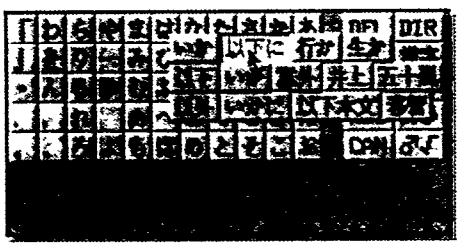

Figure 14: Selecting "以下に".

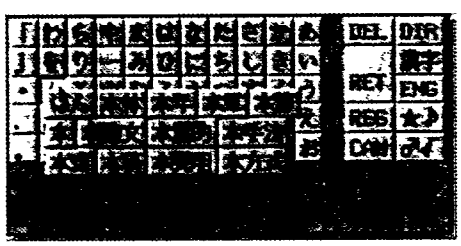

Figure 15: Before selecting "本手法".

示す"[2] as a sample Japanese input text. Figure 13 shows the initial display of POBox in Japanese input mode. A Hiragana character table is displayed for entering pronunciations, instead of the Roman alphabet in English mode.

The pronunciation of the first word "以下に" is "いかに"(i-ka-ni), and the user can select the word by choosing "い"(i) and "か"(ka) from the Hiragana keyboard, just like in the English example. Figure 14 shows how the user can select the word "以下に" with the pulldown menu. The user can select the next word "本手法" (pronounced "hon-shuhou") after selecting its pronunciation "ほ"(ho) and "ん"(n).

In this way, the user can enter Japanese text by specifying the pronunciation of the first portion of the word and then selecting the desired word from the menu, just like specifying the spelling for English words. The user can input the phrase "以下に本手法を" in 7 penstrokes, whereas the ordinary RKC method requires at least 20 penstrokes.

DETAILS OF THE ALGORITHM

Dictionaries and Word Prediction

The word dictionary is a set of 2-tuples {word, spelling/pronunciation} sorted by the term frequency of the word. The top portion of the English word dictionary is shown in Figure 16. Since "the" appears more often than any other word in the corpus, it resides at the top of the dictionary, with its spelling "THE." The phrase dictionary is a set of 3-tuples {context, word, spelling/pronunciation} sorted by the phrase

[2]"Here, we show an example of entering text using this method"

Word	Spelling/Pronunciation
the	THE
of	OF
to	TO
and	AND
...	...

Figure 16: Word dictionary.

Context	Word	Spelling/Pronunciation
of	the	THE
in	the	THE
to	the	THE
...	...	...
as well	as	AS
into	the	THE
...	...	...

Figure 17: Phrase dictionary.

frequency. Here, "context" means the word(s) that precede the input word. The top portion of the initial phrase dictionary is shown in Figure 17. Of all the phrases (lists of more than one words), "of the" occurs most often and hence appears at the top of the phrase dictionary.

Whenever possible, POBox checks the context and the characters specified by the user, and generates the list of candidate words for the next user input. First, it checks the phrase dictionary and looks for the dictionary entries whose context match the current context and whose spelling match the user input. If such entries are found, POBox puts them into the candidate list. Then it checks the word dictionary and looks for entries whose spelling match the user input. If no entry is found in both of the dictionaries, POBox tries to find more candidate words by performing approximate string matching described in the next section. After the user selects a word from the menu, the newly selected word and phrase are put at the top of the dictionaries.

A middle-sized natural language dictionary usually has 20,000 to 50,000 word entries, which occupies less than 500KB of memory without compression. With appropriate compression and indexing techniques, a word dictionary plus a phrase dictionary can easily be packed into 1MB of memory.

Approximate String Matching

Our approximate string matching algorithm is based on Baeza-Yates' "shifter algorithm"[1], with our extensions for allowing errors and handling simple wildcard characters. The shifter algorithm is also used in an approximate string matching program **agrep**[7] (an extension to **grep** on UNIX), where wildcard characters are treated differently from ours. In our algorithm, we limit the wildcard to the basic "**.***" pattern in order to achieve simple and fast processing.

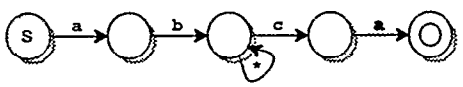

Figure 18: A state transition machine which accepts "**ab.*ca**".

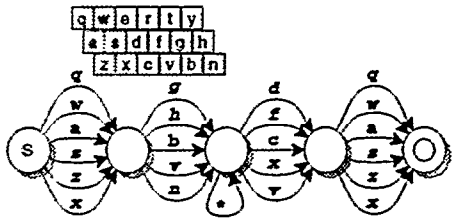

Figure 19: A state transition machine with spatial approximation

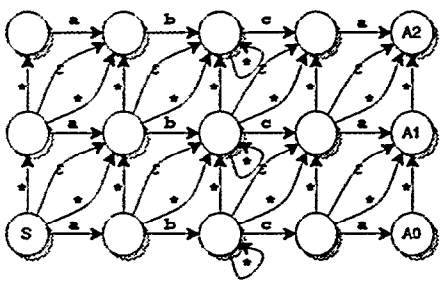

Figure 20: State transition machine which allows errors.

Figure 18 shows a nondeterministic state transition machine which accepts a regular expression "`ab.*ca`". In the shifter algorithm, a bit string is used to represent the status of this state machine. For example, the initial state is represented as "`10000`", and it becomes "`11000`" after accepting an "`a`".

The state machine can be extended to perform spatial approximate search by adding transitions by adjacent characters (Figure 19.) The state machine can also be extended to allow errors by adding extra rows of states as shown in Figure 20. A0 is the accept state with no errors, and A1 and A2 are the accept states with one and two errors, respectively. Like most spelling correctors, POBox treats character insertion, deletion and substitution as errors. Figure 21 shows the state transition by "`abracadabra`". After reading "`ab`", state A2 becomes active, showing that "`ab`" matches "`ab.*ca`" with two errors. After reading "`abra`", state A1 also becomes active, showing that "`abra`" matches "`ab.*ca`" with one error.

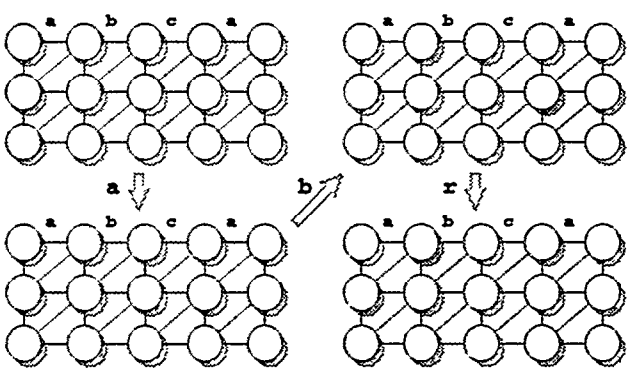

Figure 21: State transition by "`abracadabra`".

This state transition can be calculated with simple logic and shift operations. For a short pattern with small ambiguity, POBox first creates a deterministic state transition table from the nondeterministic state transition diagram like the one shown in Figure 20, and uses the transition table instead, for faster processing. For example, the state machine in Figure 20 can be converted to a deterministic state transition table with 32 states.

EVALUATION

POBox currently runs on UNIX(X11), Windows95, Newton, Java VM, and Pilot. POBox for Pilot is the latest version, distributed to the public on the Web[3] since July 1997, and downloaded by more than 10,000 people in two months. Since it is the most widely-used version of POBox, we used it for the evaluation, although it lacks the pulldown menu feature because of its limited processing power.

A set of inquiries asking the user's background and impressions of POBox was also presented on the Web page for downloading POBox, and 1,057 people answered the questions. Among the 967 people with experience in both POBox and Japanese handwriting recognition systems, 126 people (13.0%) said they feel that POBox is as efficient as handwriting recognition systems, and 796 people (82.4%) said POBox is more efficient. Among the 899 people with experience in both POBox and RKC systems, 118 people (13.2%) said they feel that POBox is as efficient as conversion-based systems, and 718 people (80.1%) said POBox is more efficient. Several people sent back comments saying that they feel POBox is the most effective pen-based Japanese input method they have ever used.

To obtain more reliable data, we asked POBox users who answered the inquiry to compare the text input time using POBox and other handwriting recognition systems[4]. Of these users, we selected approximately 300 people who seemed to have reasonable experience with both POBox and handwriting recognition systems, independent of their performance on the two systems, and 31 people agreed to perform the experiment and sent back the test results. All of them are adult male, and most of them are engineers in various Japanese companies. About half are in their thirties, three are in their forties, all of them having enough experience on both POBox and handwriting recognition systems.

We asked the participants to measure the entry time of a sample Japanese text consisting of 53 Kanji/Kana characters and 2 punctuation characters, under the following conditions:

1. writing the text on paper.
2. entering the same text using POBox.
3. entering the text using conventional RKC.
4. entering the text using the participants' favorite Kanji handwriting recognition systems on any architecture.

[3] http://www.csl.sony.co.jp/person/masui/POBox/pilot.html

[4] We offered calling cards (a value of approximately $5) to the participants as a token incentive to perform the test seriously.

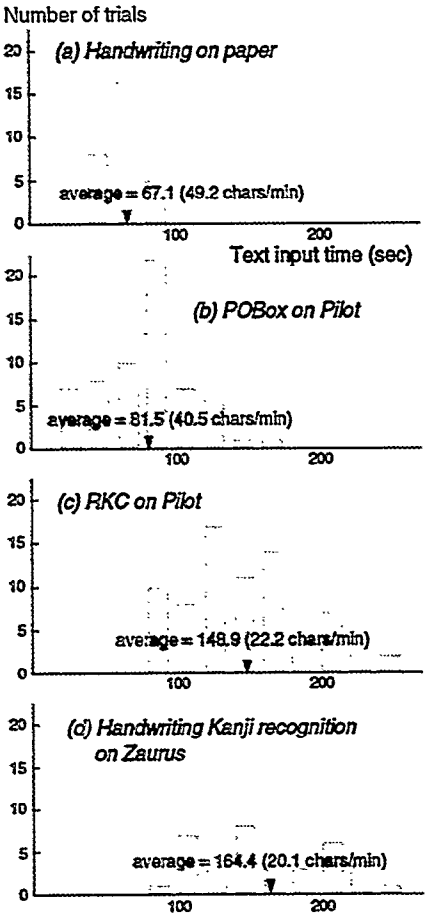

Figure 22: Distribution histograms of text input time using different methods.

The second and third tests were performed on the Pilot, which does not have a Kanji handwriting recognition system. Therefore, we asked the participants to use their favorite recognition systems, instead.

Among the 31 participants, 10 people used the same handwriting recognition system available on a Zaurus PDA[5] (made by Sharp). Other people used various handwriting recognition systems on PCs and other PDAs, but the recognition time was longer than on the Zaurus. The summary of the test result is shown in Figure 22. Since not all participants completed all experiments for the same number of times, the area of the histogram differ among the tests.

Input Speed Comparison

Most of the participants could write the sample text on paper faster than with any of the electronic text input methods. (The average was about 50 chars/min.) Writing speed does not vary significantly between people. On the other hand, the text input speed using Zaurus' Kanji handwriting recognition system does vary considerably from person to person, the average being about 20 chars/min. This is much slower than writing on paper, because of the recognition error and difficulty of writing on a tablet. No correlation was observed between the speed of writing on paper and the speed of entering text using handwriting recognition systems.

The average text input speed using POBox was about 40 chars/min, which is approximately twice as fast as conventional RKC or Zaurus' handwriting recognition system. While the fastest handwriting recognition times observed were shorter than the slowest POBox users, every individual tested performed better with POBox than with the handwriting recognition system.

Approximate String Matching

We have not advertised the approximate string matching feature very much on the Web page, but 448 people (43.4%) of the users noticed this feature. Of these 448 users, only 30 of them (6.7%) answered that approximate string matching was not useful for them.

DISCUSSIONS

Stochastic Analysis of the Dictionary

The total number of words in the CHI'95 CD-ROM is about 650,000, and the distribution of the frequency conforms well to Zipf's rank-frequency law. From the data, the probability of finding the desired word in the candidate menu after entering the top portion of the spelling can be calculated by summing up all the frequencies of words that appear in the menu after each penstroke. This is the case when using POBox without the prediction from context feature. The result is shown in Figure 23.

When the system shows 10 candidates after each penstroke, about 53% of the input words can be found in the menu after specifying one character, and about 92% of words can be found after three penstrokes. This means that 92% of the words can be entered with four penstrokes, while about 50% of the words in the CHI'95 CD-ROM consist of more than four letters. This result shows that the menu-based text input method of POBox is effective even without the prediction mechanism.

The same analysis for the Japanese dictionary is shown in Figure 24. Since about 50 Hiragana characters are used for

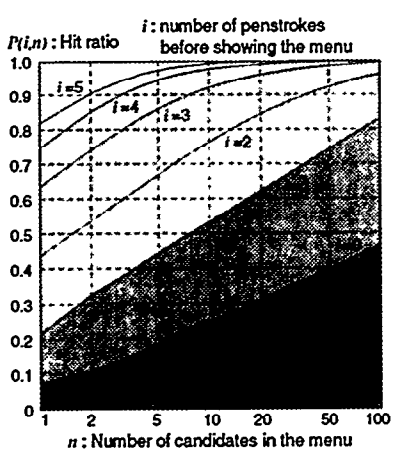

Figure 23: Probability of finding the desired word in the menu (English text).

[5]Zaurus was the most popular PDA in Japan at the time this experiment was performed.

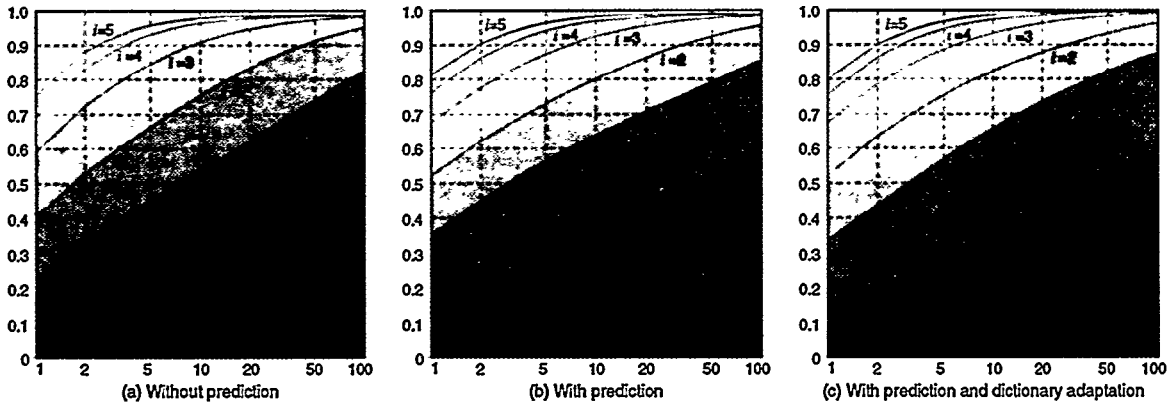

Figure 25: Probability of finding the desired word in the menu.

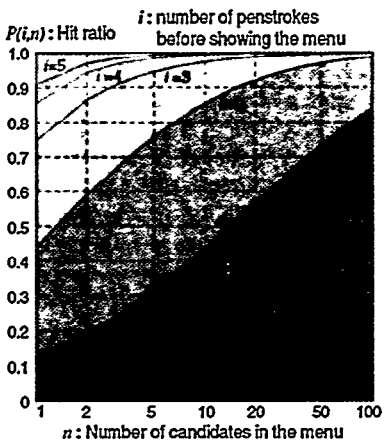

Figure 24: Probability of finding the desired word in the menu (Japanese text).

Japanese text input, most of the desired words can be found in the menu after two or three penstrokes, while more than four penstrokes are required using ordinary Kanji-conversion methods.

Dynamic Analysis

More accurate hit ratio of POBox menus can be calculated by simulating the prediction and adaptation mechanisms of POBox with real English text. Figure 25(a) shows the hit ratio calculated by using all the texts in the CHI'95 CD-ROM. The hit ratio with the prediction from context feature is shown in Figure 25(b), and the hit ratio with prediction and dictionary adaptation is shown in Figure 25(c). Prediction from context is effective for increasing the hit ratio, especially when no input is specified for selecting words ($i = 0$). In this case, POBox displays the correct word among its 10 candidates 38% of the time, whereas this number drops to 26% when prediction is not used.

Input Speed Estimation

Text input speed can also be estimated by dynamic analysis if the character input speed using the soft keyboard and the speed of menu selection is known.

From the dynamic analysis shown above, the hit ratio $P(i,n)$ of finding a word in the menus with n items after selecting i characters is known. If it takes T_k for a user to input one character and it takes $T_s(n)$ to select an item from the menu with n items, the average total time for entering a word ($T(i,n)$) can be calculated by the following formula:

$$\begin{aligned} T(i,n) &= T_s(n) \\ &+ (T_k + T_s(n))(1 - P(0,n)) \\ &+ (T_k + T_s(n))(1 - P(1,n)) \\ &+ \ldots \\ &= T_s(n) + \textstyle\sum_{j=0}^{\infty}(T_k + T_s(n))(1 - P(j,n)) \end{aligned}$$

If the user starts using the menu after entering at least i characters, the average total time $T(i)$ is calculated by the following formula:

$$T(i,n) = i \cdot T_k + T_s(n) + \sum_{j=i}^{\infty}(T_k + T_s(n))(1 - P(j,n))$$

We assume that $T_s(n)$ is proportional to n and T_k is a constant value, since POBox shows a menu of candidates according to the probability of the words, and the user cannot tell the ordering of the words in the menu beforehand. We calculated $T(i,n)$ using $P(i,n)$ for the two cases of slow and fast character input.

Slow Character Input: Figure 26 shows the calculated average time for entering a word where character input speed is slow and $T_s(n)$ can be estimated to be $n/10$ and T_k is the constant 1. In this case, without prediction, the minimum text input time is obtained when $i = 1$ and $n = 3$, which means using a three-entry menu after one penstroke without a menu. With prediction, the input time is minimized when $i = 0$ and $n = 3$, which means using a three-entry menu from the start. This is because frequently-used words are displayed at the top of the menu even before the user specifies characters for filtering the dictionary. The estimated average time for entering words is smaller with prediction than without prediction.

Faster Character Input: Figure 27 shows the average time for entering a word, where character input speed is faster than the previous example and $T_s(n)$ is estimated to be $n/3$. In this case, minimum input time is obtained when $i = 0$ and $n = 1$, which means predicting one word every time after entering a character.

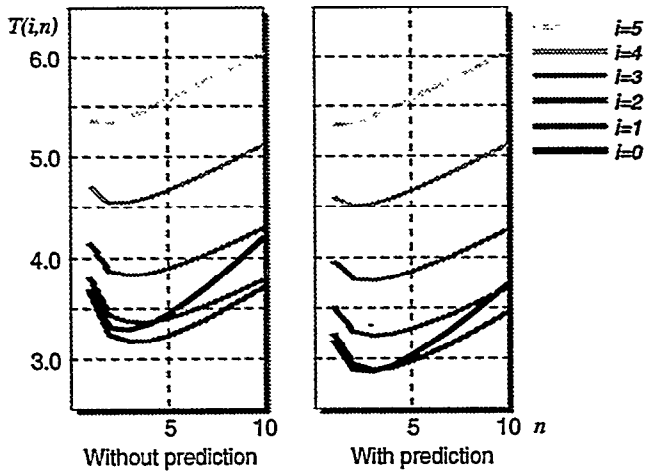

Figure 26: Text input speed estimation with slow character input. ($T_k = 1, T_s(n) = n/10$)

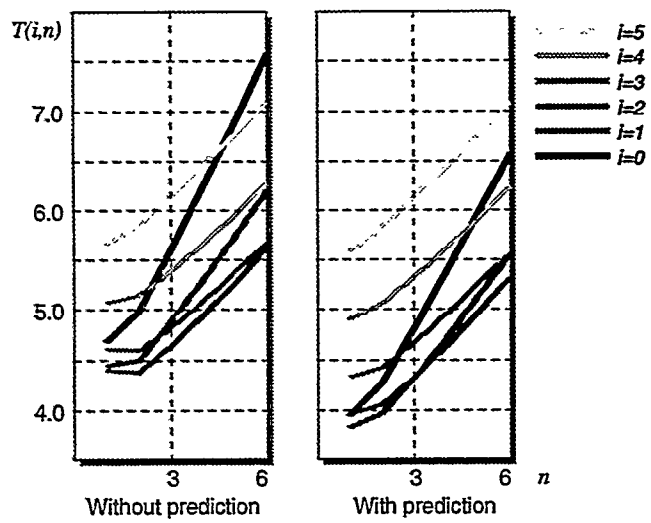

Figure 27: Text input speed estimation with faster character input. ($T_k = 1, T_s(n) = n/3$)

In this manner, the fastest method for entering text depends on the relation between $T_k/T_s(n)$ and $P(i,n)$. Roughly speaking, when $T_k/T_s(n)$ is very small (character input is very fast) as with a keyboard, the fastest way of entering text is entering characters without the use of menus. On the other hand, if $T_k/T_s(n)$ is very large (character input is very slow), using menus with many entries is faster. The two cases shown in Figure 26 and Figure 27 are between these extremes, and POBox supports the entire spectrum.

Related Work

Darragh's Reactive Keyboard[2] predicts the user's next keystrokes from the statistical information gathered by the user's previous actions and shows the predicted data for the selection. Unfortunately, the Reactive Keyboard is not usually useful for experienced computer users, since they can type much faster than selecting candidates from the menu. On pen-based computers, however, people cannot enter characters as fast as with keyboards, thus predictive methods like POBox and the Reactive Keyboard are useful. By integrating existing common GUI tools with the prediction mechanism, POBox can greatly reduce the time for text input on pen-based computers, especially for Japanese and other languages where direct text input is not possible.

Greenberg[5] argued that it is convenient to put frequently used tools close at hand, and showed that this technique is useful for issuing text commands in his WORKBENCH system. POBox resembles the WORKBENCH system in that both frequently used words and recently used words always appear close at hand at the top of the candidate list for quick selection.

Fukushima et al.[3] showed that input word prediction can reduce the search space and the number of penstrokes for handwriting recognition of Japanese texts. Although they reported that their prediction system could reduce input penstrokes from 10 to 40 percent, problems with handwriting recognition still remain and the text input speed does not increase dramatically.

CONCLUSIONS

We developed a new fast text input method for pen-based computers based on dynamic query of the dictionary and word prediction from context. With our method, the speed of text input on pen-based computers greatly increases and for the first time, pen computing becomes a viable alternative to keyboard-based input methods.

ACKNOWLEDGEMENTS

We would like to thank Jun Rekimoto and Jeremy Cooperstock for giving us many valuable suggestions. We also thank many POBox users who actually used it, sent comments to us, and performed the evaluation tests.

REFERENCES

1. Baeza-Yates, R. A., and Gonnet, G. H. A new approach to text searching. *Communications of the ACM 35*, 10 (October 1992), 74–82.
2. Darragh, J. J., Witten, I. H., and James, M. L. The Reactive Keyboard: A predictive typing aid. *IEEE Computer 23*, 11 (November 1990), 41–49.
3. Fukushima, T., and Yamada, H. A predictive pen-based Japanese text input method and its evaluation. *Transactions of Information Processing Society of Japan 37*, 1 (January 1996), 23–30. in Japanese.
4. Goldberg, D., and Richardson, C. Touch-typing with a stylus. In *Proceedings of ACM INTERCHI'93 Conference on Human Factors in Computing Systems (CHI'93)* (April 1993), Addison-Wesley, pp. 80–87.
5. Greenberg, S. *The Computer User as Toolsmith.* Cambridge Series on Human-Computer Interaction. Cambridge University Press, March 1993.
6. Venolia, D., and Neiberg, F. T-Cube: A fast, self-disclosing pen-based alphabet. In *Proceedings of the ACM Conference on Human Factors in Computing Systems (CHI'94)* (April 1994), Addison-Wesley, pp. 265–270.
7. Wu, S., and Manber, U. Agrep - a fast approximate pattern-matching tool. In *Proceedings of USENIX Technical Conference* (San Francisco, CA, January 1992), pp. 153–162.