POBox: An Efficient Text Input Method for Handheld and Ubiquitous Computers

Toshiyuki Masui

Sony Computer Science Laboratories, Inc.
3-14-13 Higashi-Gotanda
Shinagawa, Tokyo 141-0022, Japan

Abstract. We introduce an efficient text input technique that can be used in various environments where conventional full-sized keyboards cannot be used. The technique, called POBox, consists of two steps for entering a word or a phrase. First, a user enters a small part of the word or some other attribute, and POBox dynamically searches a dictionary for candidate words and shows them to the user for selection. The user then selects the desired word from the candidate list, and POBox enters the word into the user's document. POBox uses the context of the user's document to help identify likely candidates. Many times POBox can predict the desired word based on the context. This allows the user to skip the first step and enter text even more efficiently. We show that the same technique can be applied to various handheld and ubiquitous computers including PDAs and cellular phones, where conventional full-sized keyboards are inadequate.

1 Introduction

Full-sized keyboards have been by far the most common and efficient text input devices. By using a keyboard, a trained user can enter hundreds of characters per minute. However, a keyboard is efficient for text input only if there is enough space to hold the keyboard, only if both the user and the keyboard are in stable positions, only if the user can use both hands freely, and only if the user is well trained. These conditions are too restrictive these days, since computers are to be used by anyone at any place. People might want to check their e-mail in a restaurant, browse the Internet in the kitchen, write an e-mail in a commuter train, write a text on a whiteboard in a classroom, write down their schedule on their handheld computers while walking, etc. In all of these cases, the user cannot use a standard keyboard, either because it is too large or because the user cannot use both hands. There is no doubt that in the future, computers will be mainly used in environments where using full-sized keyboards are inadequate. It is therefore important to develop a technique in which characters and texts can easily be entered in any of these situations.

We propose an efficient text input technique called POBox (Predictive cOmposition Based On eXample) that can be used in various environments where conventional keyboards are difficult to use. With POBox, users can efficiently enter text in any language by using menus, word prediction, and approximate pattern matching. In this paper, we first introduce various existing text input techniques which does not require a standard keyboard, and then we introduce the POBox system by showing an application of it for pen-based computers. We also introduce implementations of POBox on other handheld machines including cellular phones and wearable computers.

2 Overview of Text Input Methods

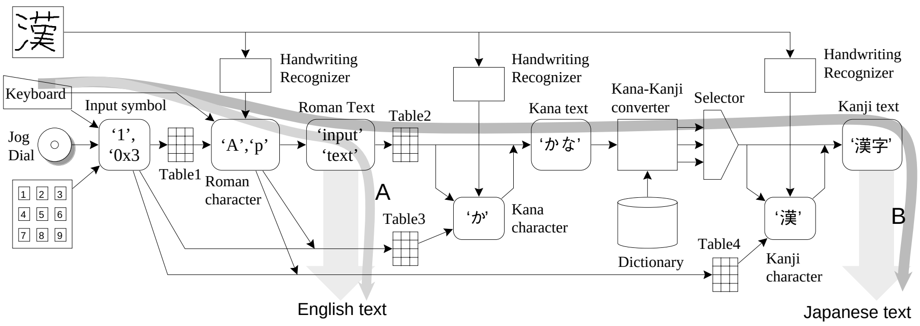


Fig. 1. Structure of various text input systems.

Figures 1 shows an overview of various existing text input systems. Line A shows how an English text is composed using a standard keyboard. Roman character codes are directly generated by the keyboard and concatenated to generate a text. Line B shows how a Japanese text is composed using a standard keyboard. Roman character strings are first converted to Kana texts which represent the pronunciation of Japanese words, and then they are converted to Kanji characters by a Kana-Kanji converter. Since multiple Kanji characters often have the same pronunciation, the user must choose the correct one by using the selector.

There are many reasons why text input on handheld computers is very slow. First, typing a key or writing a character is much slower than using a standard keyboard. Second, users have to type keys more times than when using standard keyboards, since small input devices often have smaller number of keys (e.g. cellular phones usually have only 20 keys). These keys can generate only a small number of input symbols, and combinations of the keypress must be converted to Roman characters using a mapping table (Table1.) In this way, input symbols must be converted more than once until the final text is composed. When entering Japanese text on a cellular phone, the input symbols must first be converted to Kana characters using Table3, and then Kana character string is converted to a Kanji character using the Kana-Kanji converter. A proper Kanji must then be selected using the selector. Hence, it takes a significant amount of time to follow these steps.

A variety of techniques for fast text input on handheld machines have been proposed. One approach is to make the speed of using a software keyboard faster. "QWERTY" layout is often used for the software keyboard, but QWERTY is not the best layout for a pen-based software keyboard, since frequently-used key combinations are sometimes laid out far apart and users must move the pen for a long distance to enter a text. "Fitaly" keyboard[11][3] is a layout for minimizing the pen movement on software keyboards. Since "e" and "n" often appear next to each other in many English words, they are put in an adjacent position on the Fitaly keyboard. Other layouts are also proposed to improve the input speed on software keyboards[2][4].

Another approach is to use fast handwriting recognition systems. Unistroke[1] was one of the first approaches in this direction, and similar techniques like Graffiti have become very popular on recent handheld computers including 3Com's PalmPilot. More sophisticated gesture-based techniques like T-Cube[13], Quikwriting[9], and Cirrin[5] have also been proposed.

Yet another approach is to give up entering characters one by one, and to use a word dictionary for composing a text. Textware's InstantText system[12] allows users to use an abbreviated notation of a sentence to reduce the number of input. For example, users can type "**`oot`**" to enter "**`one of the`**", or type "**`chrtcs`**" to enter "**`characteristics`**". These abbreviations are dynamically created and they do not have to be predefined. Tegic's T9 system[10] takes a different approach. T9 was originally developed for composing texts using only 9 keys on a standard telephone. On T9, instead of typing keys more than once to select an input character, more than one characters are assigned to the digit keys of a telephone so that users do not have to be concerned about the differences. Figure 2 shows a typical key assignment on a telephone keypad. When a user wants to enter "**`is`**", he pushes the "4" key first where "**`G`**", "**`H`**", and "**`I`**" are printed, and then pushes the "7" key where "**`P`**", "**`Q`**", "**`R`**", and "**`S`**" are printed. Using the combination of "4" and "7" corresponds to various two-character combinations including "**`hr`**", "**`gs`**", etc., but "**`is`**" appears most frequently in English texts, and the system guesses that "**`is`**" is the intended word in this case.

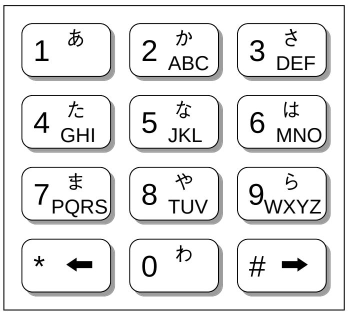


Fig. 2. Phone key assignment.

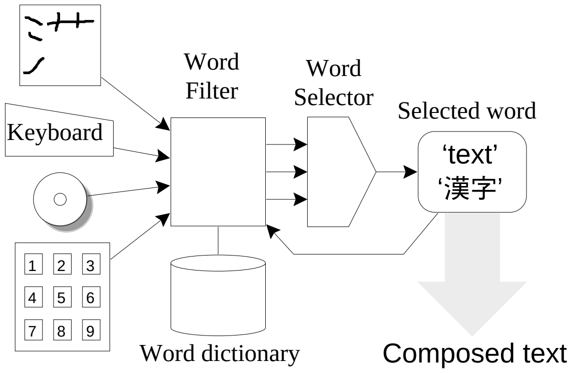


Fig. 3. POBox architecture.

3 POBox: Incremental and Predictive Text Input Method

POBox is a text input method for handheld and ubiquitous computers, with which only small number of user operations are required to compose a text. Figure 3 shows the architecture of POBox.

A text composition task with POBox consists of repetitions of the following two steps.

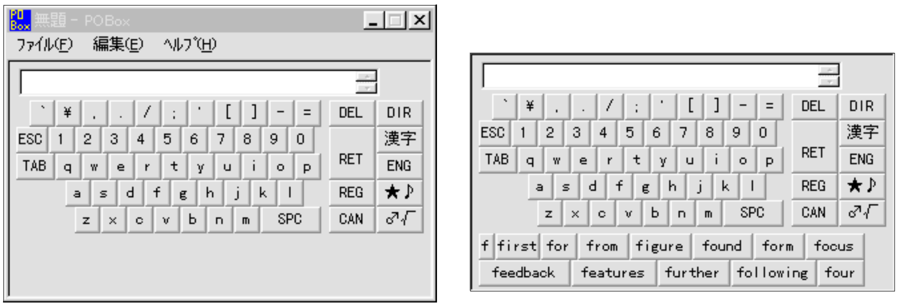
Filtering Step First, a user provides search keys for a word he wants to enter. Search keys can be the spelling, pronunciation, or shape of a character. As soon as he enters search keys, the system dynamically uses the keys to look for the word in the dictionary and shows candidate words to the user for selection.

Selection Step Second, the user selects a word from the candidate list and the word is placed in the composed text. Next input words are predicted from the context and are used in the next filtering step.

In most existing text input systems, users must provide all the information for the input text, either by specifying input characters or by showing the complete shape of characters by giving handwritten strokes. In POBox, users do not have to give all of them to the system; they only have to give information to the system which is enough for the search.

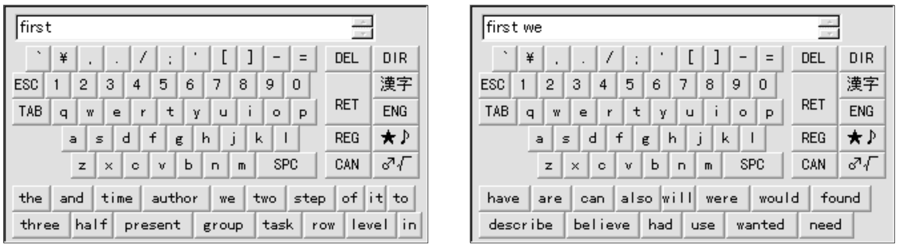
Users also do not have to specify all the characters or stroke elements which constitute a word; they only have to specify part of the input word and select it from the candidate list. This greatly reduces the amount of operations and time for composing a text, especially when selecting input characters is very slow or difficult.

This architecture can be applied to a variety of non-keyboard devices, including pen tablets, one-hand keyboards, and jogdial-based phones. We show how the same technique can be applied to many handheld and ubiquitous computers of different kind in the following sections.



(a) Initial Display (b) After tapping the ‘‘F’’ key

Fig. 4. Pen-based POBox.



(a) After selecting ‘**first**’ (b) After selecting ‘**we**’

Fig. 5. After selecting “**first**” and “**we**”.

3.1 Using POBox for Pen-based Computers

POBox was originally developed for pen-based computers[6], and it was extended to handle both software keyboards and handwriting recognition systems without changing modes[7]. We briefly review how POBox works on pen-based computers including pen-based Windows95 and 3Com’s PalmPilot.

Using POBox on Windows First, we show how POBox can be used on pen-based computers. We use the previous sentence as a sample input text.

Entering English Texts Figure 4(a) shows the startup display of POBox running on Windows95. When the user pushes the “F” key, the display changes to Figure 4(b), showing the frequently used words that start with “F” in a candidate word list.

Since the word “**first**” is a frequently used word and is found in the candidate list, the user can tap the word “**first**” so that it is put into the text area. After selecting “**first**”, the display changes to Figure 5(a). In the menu at the bottom, the words that often come after “**first**” are listed in order of frequency. The next word, “**we**”, often comes after “**first**”, and this word is again in the predicted list of candidate words, and the user can directly select “**we**” by touching it in the menu. After selecting “**we**”, the display changes to Figure 5(b).

In this way, users can repeatedly specify the search key and select a candidate word to compose a text.

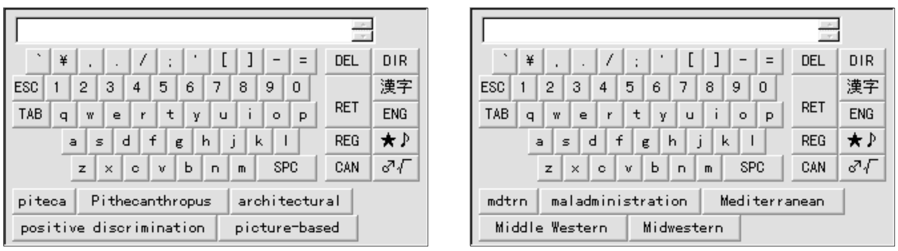
Using Approximate String Matching When no word is found from the keys given by the user, POBox automatically performs approximate pattern matching in which words closest to the given pattern are listed. With this feature, even when the user does not give the correct spelling of a word, there is a good chance of finding the desired word among the candidates. In Figure 6(a), a user is trying to input "**`Pithecanthropus`**" without knowing the exact spelling. Even so, the correct word is in the candidate because "**`Pithecanthropus`**" is the word which is closest to the pattern "**`piteca`**" given by the user.

Also, by using the same algorithm, the user can give 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 6(b).) This feature is similar to the automatic creation of abbreviations in the InstantText system[12], but InstantText cannot allow spelling errors like the previous example.

Using POBox on PalmPilot We show how handwriting recognition can be used for the selection step of POBox[7]. Figure 7(a) shows the display of POBox on PalmPilot after the user taps the "ま"(ma) key on the software keyboard. The words listed are candidate words which begin with the pronunciation "ma".

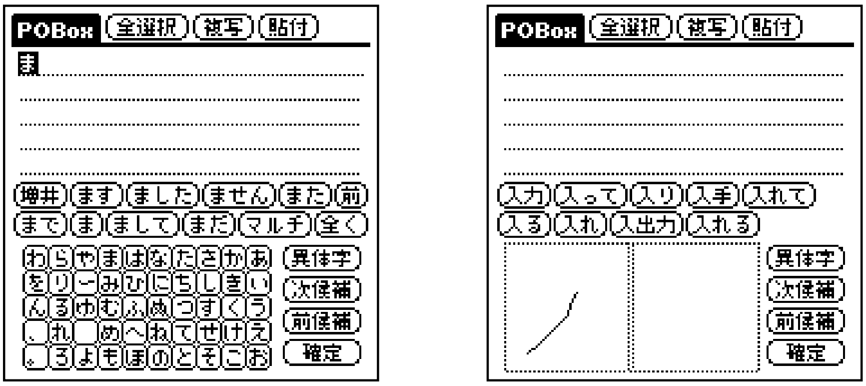
When the user moves the pen after touching the tablet instead of tapping the software keyboard, the system starts handwriting recognition and interprets the strokes incrementally, and shows candidate words that begin with the strokes. Figure 7(b) shows the display after the user has drawn a line from the center of the software keyboard to the lower left corner. This is the first stroke of the Kanji character "入," and those words that begin with the character are shown as candidates. Unlike existing handwriting recognition systems that recognize characters only after all penstrokes that constitute the character have been written, incremental recognition can greatly reduce the number of penstrokes the users have to draw.

In this way, software keyboards and handwriting recognition are seamlessly integrated in POBox for pen-based computers.



(a) Searching for '**`Pithecanthropus`**' (b) Searching for '**`Mediterranean`**'

Fig. 6. Performing approximate pattern matching.



(a) Using Japanese soft keyboard (b) Incremental handwriting recognition

Fig. 7. Using POBox on PalmPilot

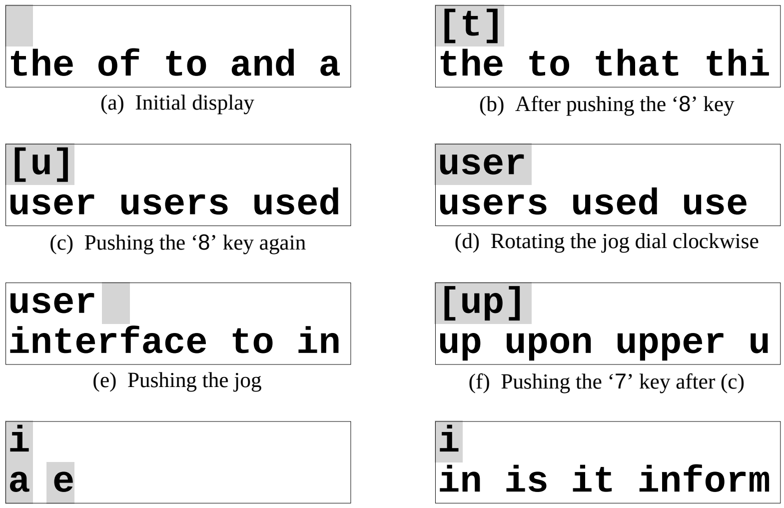
3.2 Using POBox for a Cellular Phone

POBox can be used for handheld devices which do not have pen tablets. Instead of using a software keyboard or pen operations on a cellular phone, digit keys and a jog-dial can be used for the filtering step and the selection step.

Since the number of keys on a cellular phone is much smaller than the number of keys on an ordinary keyboard, it is impossible to select an input character with one keypush. We took an approach that is currently popular in commercial cellular phones; pushing the “7” key once to input “p”, pushing the same key twice to input “q”, etc.

Fig. 8. CDMA cellular phone and POBox running on it.

Figure 8 shows the implementation of POBox on a CDMA cellular phone. The phone has about 20 keys on the surface and a jog-dial at the left side of the LCD display. Three or four alphabetical characters are assigned to each digit key (Figure 2) like standard push-phones in North America. Hiragana characters are also assigned to those keys in order to specify the pronunciation when used for Japanese text input.



(a) Initial display (b) After pushing the '8' key (c) Pushing the '8' key again (d) Rotating the jog dial clockwise (e) Pushing the jog (f) Pushing the '7' key after (c) (g) Rotating the jog dial counterclockwise (h) Pushing the jog

Fig. 9. Text input steps on CDMA phone.

Figure 9(a) shows the initial display of the phone. Frequently-used words are listed as candidates at the bottom of the display. When a user pushes one of the digit keys, the character printed on the key is shown at the cursor position and candidate words which begin with the character are shown at the bottom of the display (Figure 9(b)). When the user pushes the key again, the next character printed on the keytop is shown and corresponding candidate words are displayed (Figure 9(c)). A user can rotate the jog-dial clockwise at any time to select a candidate word. If "**user**" is the desired word, the user can rotate the jog-dial and display "**user**" at the top of the display. As the user changes the selection, more candidate words appear at the bottom of the screen for selection (Figure 9(d)). The user can then push the jog-dial to make the selection final. At this moment, next word is predicted just like pen-based POBox and next candidate words are displayed at the bottom. The user can again rotate the jog-dial to select a candidate from the list (Figure 9(e)). In Figure 9(c), if the user pushes the "7" key, "**p**" is selected as the next character for the input word (Figure 9(f)).

When a user begins rotating the jog-dial counter clockwise, the user can select input characters by the jog rotation. Input characters are sorted in frequency order; "**e**", "**a**", "**i**", etc. appear as the candidate input character as the user rotates the jog-dial. Figures 9(g) shows the display after the user rotated the dial three steps. When the user pushes the jog, the search character becomes fixed, and words which begin with the pattern are displayed as candidates (Figure 9(h)). The user can then rotate the jog-dial clockwise to select the candidate input word (e.g. "**information**".) Although using a jog-dial for

```
the of to and a
```
(a) Initial display

```
of with on we or
```
(b) After typing K4

```
work some would software so
```
(c) After typing K4

```
work some working works word
```
(d) After typing K2 and K3

```
word
words worked something
```
(e) Selecting '**word**'

```
word
     processors and processing
```
(f) Typing K5

```
word processing
     processor a as are
```
(g) Selecting '**processing**'

Fig. 10. Entering "**word prediction**" using one-handed POBox.

character input takes more time than using digit keys, using a jog-dial has an advantage; users do not have to touch the digit key at all and composing text only by one hand is possible.

3.3 Using POBox for One-hand Keyboards

A one-handed keyboard is sometimes more convenient for handheld or "wearable" computers. A large one-handed keyboard is sometimes more convenient for elderly people and handicapped people than using a full-sized keyboard designed for dextrous people. Some companies even claim that one-handed keyboards are the best input devices even when used on a desk.

One-handed keyboards have much less keys than full-sized keyboards, and the restrictions for text input seem to be the same as that for cellular phones. However, there are slight differences. When using a cellular phone, ordinary users cannot push more than three keys in one second. However, when using a well-designed one-handed keyboard, dextrous people can easily type more than 6 keys in one second. This means that, when using a one-handed keyboard, it is better for users to type as many keys as they can to perform an effective search of the dictionary.

With this consideration, we took an approach which is similar to the T9 system. We mapped all the alphabet keys to only four keys, each of which corresponds to each finger on one hand, excluding the thumb. All the keys which are pressed by both hands' forefinger ("f", "j", "t", "y", etc.) are mapped to one key (K2), and all the keys pressed by the middle finger are mapped to another key (K3)[1]. In this way, all the alphabetical keys are mapped to only four keys (K2–K5). So a word such as "**word**" is mapped to "K4 K4 K2 K3". On a standard ASCII keyboard, we can map the "a" key for K5, "s" for K4, "d" for K3, and "f" for K2. With this assignment, "**word**" is mapped to "**ssfd**".

Figure 10 shows how a text like "**word processing**" can be entered using this method. Figure 10(a) shows the initial display. By typing K4, K4, K2, and K3, the

[1] This number notation is derived from the notation used in piano scores.

display changes to (d), showing the first input word "**word**" in the candidate list. The space key is used for selecting candidates in this case, and when the user types the space key five times here, "**word**" comes to the text input area as shown in (e). When the user types K5 which corresponds to the first letter "**p**" for the next word "**processing**", the display will change to (f) and "**processing**" can be selected as the next word by typing the space key three times. From start to finish, in order to enter a text such as "**word processing**", 13 keystrokes were required in this case, which is not bad because "**word processing**" is made up of 15 characters.

This approach is similar to the approach used in the Half-QWERTY keyboard[8]. The Half-QWERTY keyboard can be considered as a "folded" keyboard where "k" and "d" are mapped to the same key position. Users can easily type "d" instead of "k" because of the symmetric nature of the human nerve system. Our method can be considered as folding the keyboard even more, until only four keys are left.

Just like T9, many different words fall into the same keystrokes in this approach. For example, "**word**", "**work**", and "**some**" are all mapped to the same keystroke "K4 K4 K2 K3". However, even when using the reduced keystrokes for the filtering part of POBox, it is not difficult to select a required word by typing the space key.

In our method, users who are able to touch-type do not have to remember new key-character mapping at all. With the key mapping on an ASCII keyboard shown above, the user only has to be careful not to move the fingers, and typing "f" instead of "r", "t", "g", etc. is not difficult.

Although this method looks similar to T9, there are significant differences. When T9 is used, users have to type n keys if they want to enter a word consisting of n characters. On the other hand, POBox does not use the length of a word as a key for searching a word, and users usually do not have to type as many as n keys before they find the desired word in the candidate list.

4 Discussions

4.1 Using POBox for Other Input Devices

We have already introduced variations of POBox using a pen tablet, a one-handed keyboard, and a jog-dial with a small number of keys on a cellular phone for the filtering step and the selection step of POBox. Virtually any input device can be used for POBox. It would also be possible to use gaze input, speech input, tilting input, etc. without big modifications.

4.2 Choosing the Best Combination

The most important design decision is to determine the interface for the filtering step and the selection step. Even for a device with very small number of input methods, this design decision is difficult. For example, in POBox for cellular phones, either a jog-dial or a keypad can be used for the filtering step and for the selection step. It is difficult to know which is better than the other. The ease of use depends on many factors including the input speed of the device, difference from existing methods, ease of understanding, etc.

4.3 Dictionary

Using an appropriate dictionary is another important point when using POBox. Since people use different phrases depending on the situations, it would be nice to be able to switch dictionaries according to the situations. For example, when replying to an e-mail, using the previous e-mail as a temporary dictionary would work well.

4.4 Other Advantages

There are more advantages of using POBox. We would like to mention other advantages of POBox not mentioned so far.

Avoidance of Spelling Errors Using POBox, all the input words are selected from the candidate list, so there is little chance of making spelling errors. This may be the reason why Japanese spell checkers are not as widely used as English spelling checkers.

Language-Independence The basic concept of POBox can be applied to any language, as long as the language has a finite set of notations. POBox can also be used for drawing symbols and diagrams if they are in a dictionary and can be searched by giving the appropriate keys for the search. Writing Kanji text with handwriting recognition is actually not very different from writing symbols.

Ease of Use POBox is an example-based input method, and users do not have to know how to write characters to compose a text as long as they know how to read characters. People who can read English are almost surely able to write English as well, but being able to read Kanji characters is completely different from being able to write them. The average Japanese person can easily read a word like "憂鬱"(melancholy), but only a portion of the population can actually write it without making a mistake. Even a pre-school child can compose text using POBox as long as he can read some of the character elements. We gave POBox on PalmPilot to a 6-year old boy without explaining what it was, and found that he was able to write a very long story without asking how to use it.

5 Conclusions

We have introduced an efficient text input technique called POBox, that can be applied to various handheld and ubiquitous computers where conventional keyboards are difficult to use. We have shown three examples of applying POBox to handheld devices, and have also shown that the same technique can be used for a wide range of handheld and ubiquitous computers.

References

1. 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.

2. Hashimoto, M., and Togasi, M. A virtual oval ieyboard and a vector input method for pen-based character input. In *CHI'95 Conference Companion* (May 1995), Addison-Wesley, pp. 254–255.
3. MacKenzie, I. S., Zhang, S. X., and Soukoreff, R. W. Text entry using soft keyboards. *Behaviour & Information Technology*. http://www.uoguelph.ca/ imackenz/BIT3.html.
4. MacKenzie, I. S., and Zhang, S. Z. The design and evaluation of a high performance soft keyboard. In *Proceedings of the ACM Conference on Human Factors in Computing Systems (CHI'99)* (May 1999), Addison-Wesley. to appear.
5. Mankoff, J., and Abowd, G. D. Cirrin: A word-level unistroke keyboard for pen input. In *Proceedings of the ACM Symposium on User Interface Software and Technology (UIST'98)* (November 1998), ACM Press, pp. 213–214. http://www.cc.gatech.edu/fce/pendragon/cirrin.html.
6. Masui, T. An efficient text input method for pen-based computers. In *Proceedings of the ACM Conference on Human Factors in Computing Systems (CHI'98)* (April 1998), Addison-Wesley, pp. 328–335. http://www.csl.sony.co.jp/person/masui/papers/CHI98/CHI98.pdf.
7. Masui, T. Integrating pen operations for composition by example. In *Proceedings of the ACM Symposium on User Interface Software and Technology (UIST'98)* (November 1998), ACM Press, pp. 211–212. http://www.csl.sony.co.jp/person/masui/papers/UIST98/UIST98.pdf.
8. Matias, E., MacKenzie, I. S., and Buxton, W. Half-qwerty: Typing with one hand using your two-handed skills. In *CHI'96 Conference Companion* (April 1996), ACM Press, pp. 51–52. http://www.dgp.toronto.edu/people/ematias/papers/chi96/.
9. Perlin, K. Quikwriting: Continuous stylus-based text entry. In *Proceedings of the ACM Symposium on User Interface Software and Technology (UIST'98)* (November 1998), ACM Press, pp. 215–216. http://mrl.nyu.edu/perlin/demos/quikwriting.html.
10. Tegic Communications. *T9*. 2001 Western Avenue, Suite 250, Seattle, WA 98121. http://www.t9.com/.
11. Textware Solutions. *The Fitaly Keyboard*. 83 Cambridge St., Burlington, MA, 01803 USA. http://www.twsolutions.com/fitaly/fitaly.htm.
12. Textware Solutions. *Instant Text version III*. 83 Cambridge St., Burlington, MA, 01803 USA. http://www.twsolutions.com/overview/overview.htm.
13. 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.