FreePad: A Novel Handwriting-based Text Input for Pen and Touch Interfaces

Bharath A. and Sriganesh Madhvanath
HP Laboratories India, Bangalore
HPL-2007-203
December 20, 2007*

Text input, pen and touch interfaces, handwriting recognition,

The last decade has seen tremendous growth in mobile devices such as Pocket PCs, mobile phones, Tablet PCs and notebooks. Most of these devices enable interaction through a stylus or touch interface, powered by handwriting recognition (HWR) capability. In this paper, we propose a novel input method that addresses some of the issues that arise due to the constraints posed by these devices in accepting handwriting input. For instance, many of the devices have a small writing area making "continuous" input difficult if not impossible, and the process of handwriting input demands significant user attention. The proposed solution is inspired by touch-typing, and appreciably reduces user's effort in the interaction, and it is especially suited for very small writing areas. The approach has been demonstrated using a prototype system that recognizes handwritten English words, and its accuracy has been evaluated using a standard dataset of handwritten words. A preliminary user study has also been carried out to understand user acceptance of the proposed technique.

Internal Accession Date Only Approved for External Publication
Submitted to 2008 International Conference on Intelligent User Interfaces (IUI'08), Canary Island, Spain, Jan 13-16, 2008

FreePad: A Novel Handwriting-based Text Input for Pen and Touch Interfaces

Bharath A. and Sriganesh Madhvanath
Hewlett-Packard Labs India
Bangalore, India
{bharath.a, srig}@hp.com

ABSTRACT
The last decade has seen tremendous growth in mobile devices such as Pocket PCs, mobile phones, Tablet PCs and notebooks. Most of these devices enable interaction through a stylus or touch interface, powered by handwriting recognition (HWR) capability. In this paper, we propose a novel input method that addresses some of the issues that arise due to the constraints posed by these devices in accepting handwriting input. For instance, many of the devices have a small writing area making "continuous" input difficult if not impossible, and the process of handwriting input demands significant user attention. The proposed solution is inspired by touch-typing, and appreciably reduces user's effort in the interaction, and it is especially suited for very small writing areas. The approach has been demonstrated using a prototype system that recognizes handwritten English words, and its accuracy has been evaluated using a standard dataset of handwritten words. A preliminary user study has also been carried out to understand user acceptance of the proposed technique.

Author Keywords
Text input, pen and touch interfaces, handwriting recognition,

ACM Classification Keywords
H.5.2 [Information interfaces and presentation]: User Interfaces---input devices and strategies, interaction styles

1. INTRODUCTION

Many interfaces have been developed for handwriting-based text input for different devices, because of the intrinsic advantages of handwriting input over hard or soft keyboards (such as naturalness, suitability for scripts with large character sets, absence of seek time). Broadly these may be classified as discrete symbol based, and continuous. Examples of discrete symbol based methods include character input interfaces popular on PDAs and stylus equipped mobile phones. These require symbols to be entered one at a time into a specific writing area, and generally use a timeout to indicate the end of a symbol. This considerably slows down the process of writing. Unistroke alphabets such as Graffiti use the end of stroke to signify the end of a character, but require the user to learn special sets of symbols. Continuous input interfaces on the other hand allow words to be written as a continuous stream of strokes (i.e. without explicit indication of character transitions), often support cursive styles, and allow several words to be entered at once. However these require larger digitizer surfaces as found on TabletPCs, external tablets, and electronic paperclip devices. Limited forms are also found on some PDAs. In general, however, continuous input is difficult if not impossible when the writing area on the device is small and the stylus is substituted for by a finger.

A second issue with continuous handwriting input is the attention required by the process. Existing continuous input interfaces have been designed taking into account the conventional writing order of the script (for example, left to right in the case of English), so that the spatial positioning of the characters and words that make up the input is maintained. For example, the characters and words need to be approximately the same size, written next to one another, aligned with a (imaginary or displayed) baseline. Along with adhering to these requirements, users also unconsciously tend to read what they have written before sending the ink for recognition. As a result, such input methods require user attention to writing, i.e. the user has to look at the writing surface to make sure that the input is "spatially correct" and the position of each unit of writing with respect to the others is maintained.

The cognitive load may increase further where the writing (digitizing) surface is different from the application display surface. The user has to look at writing surface while writing, and then at the display to see the results of recognition. This constant switching between the display device and writing surface after each writing unit (which could be a gesture, character, word, or larger unit,

IUI'08, January 13-16, 2008, Maspalomas, Gran Canaria, Spain.

depending on the recognizer used) results in significant cognitive load on the user, and greatly reduces the speed of text entry.

The problem we are addressing in this paper is achieving the throughput of continuous handwriting input, but using a small digitizer such as the touchpad of a mobile phone/PDA, digital camera or a home printer, where it is practically impossible to write a complete word using the finger in the conventional way. We would also like the user to pay as little conscious attention to writing as possible, much like the touch-typist who does not pay attention to the process of typing, but rather concentrates only on the end result (text on the screen). However as we have already observed, continuous handwriting input requires conscious attention, as well as sufficient space to write on.

This paper is organized as follows. In the following section, we propose a novel handwriting-based text input method that addresses the issues described. Section 3 describes the core of the proposed solution – a modified handwriting recognition system. Section 4 describes FreePad, an IME that we have prototyped based on the proposed technique, and Sections 5 and 6 present some preliminary results from empirical evaluation and user studies respectively. The final section presents some conclusions and directions for future work.

2. PROPOSED SOLUTION

Our solution to the problems stated in the previous section is for the user to "overwrite" repeatedly on the same area without pausing between characters, as shown in Figure 1.

Figure 1. Overwriting on the same area – relative position between characters and even strokes is not maintained

We leverage the fact that unlike paper, the system in our case captures the temporal order of handwriting input, and hence laying handwriting out spatially becomes less significant. Further, removing the need to produce "spatially correct" writing also automatically reduces the need for user attention to the writing process, to a point where it becomes akin to "writing blind". It is important to note that our proposal is different from entering a character at a time, since the user does not pause between characters. Rather, the user writes continuously and maintains his/her normal speed of writing, entering a sequence of (largely overwritten) strokes. In this process, several characteristics of normal writing are lost – among them, the alignment and relative position between characters. In fact, even the relative positions between strokes of a multi-stroke character become unreliable. Ascenders and descenders disappear. The shape of the character as a whole becomes worse. The result of all of the above is considerably more ambiguity between characters, with some confusions impossible to resolve without additional context (Figure 2).

At the core of our solution, is a handwriting recognition (HWR) engine which ignores those aspects of writing that are lost or rendered unusable as a result of "writing blind". In order to resolve the resulting ambiguity, our solution uses contextual information in the form of a dictionary to differentiate characters using the word context. With this solution, we are able to achieve high throughput text input using a limited writing area. At the present time, our solution deals with recognition of one overwritten-word at a time. In other words, the input to the system is a sequence of strokes corresponding to an isolated word.

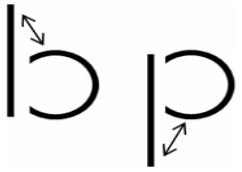

Figure 2. Ambiguity between 'b' and 'p' due to loss of relative position

3. HANDWRITING RECOGNITION SYSTEM

In order to handle the issues arising from overwriting as described earlier, the HWR engine needs to be made immune to size, shape and positional variations in the input. In this section, we describe the the architecture of our HWR engine (Figure 3).

3.1 Preprocessing and Feature Extraction

In any HWR system, preprocessing is the initial step that eliminates variations in the input due to noise and then normalizes the input for further processing. Conventional preprocessing techniques maintain the relative position between characters in a word and positions of strokes within the character. However, since our input is prone to position and size variations, preprocessing is carried out at the stroke level. Each stroke in the input ink, irrespective of its position on writing surface, is first translated to the origin and then scaled in such a way that the larger dimension is fixed to a constant value while the smaller dimension is rescaled according to the original aspect ratio of the stroke. Due to this stroke-level preprocessing, the recognition system views handwriting input as a sequence of strokes where each stroke is of similar size and written exactly at the same location. This sequence of preprocessing steps is carried out on both the strokes of character samples used for training the system, as well as on the (overwritten) strokes of the word to be recognized.

Once the strokes are preprocessed, the features extracted from the X-Y coordinates of the ink trajectory include normalized X-Y values, normalized first and second derivatives and curvature, as described in [3].

3.2 Word Modeling

In our system for recognizing handwritten words entered by overwriting, we have used Hidden Markov Models (HMM)

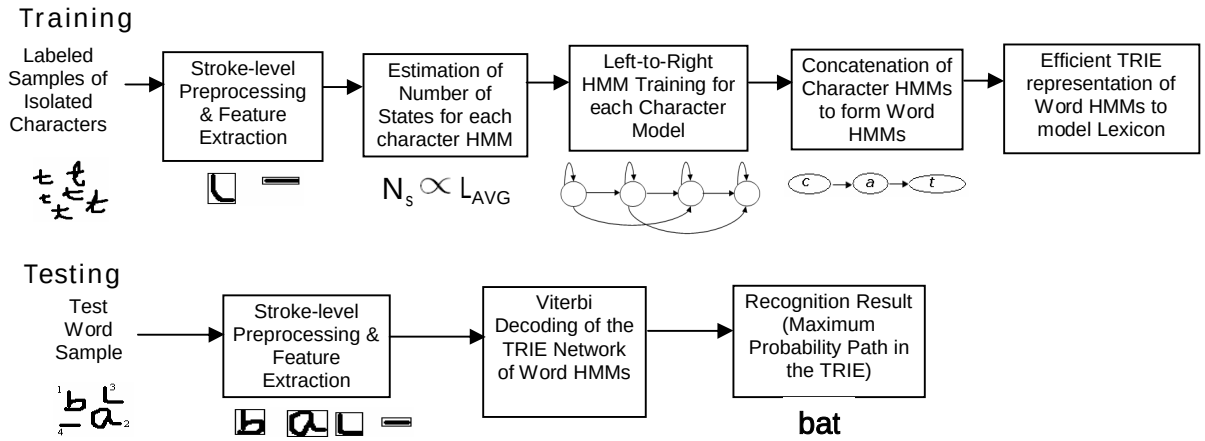

Figure 3. HMM-based architecture for overwritten word recognition

for modeling characters and thereby words. HMMs are stochastic models and hence are able to cope with noise and variations in the handwriting. A few hundred samples of isolated characters written by different writers were used to train the HMMs. These were subjected to stroke-level preprocessing and feature extraction as explained in the previous section. Subsequently the samples of a character were used to train a left-to-right HMM to model the character using the Baum-Welch algorithm [4]. The number of states in the HMM was determined based on the shape complexity of the character, measured as the average length of the character in the training set after preprocessing, and the number of Gaussians per state was empirically determined as *15*. Once the character models were formed, word models were built simply by concatenating the constituent character HMMs. The lexicon that contains the list of words to be recognized is then modeled as a Trie (also known as prefix tree) where each node in the Trie corresponds to a character. The Trie representation affords lower computational and space complexity at the time of recognition, when compared to the linear list representation.

During testing, the system recognizes the input handwritten word using the models learnt during the training phase, and the lexicon, and assigns a word label to the input using the Viterbi algorithm.

4. FREEPAD IME

Based on the HMM-based architecture for handwritten word recognition described in Section 3, a prototype word recognition system that recognizes lowercase English words was developed. The resulting Input Method Editor (IME) prototype, nicknamed "FreePad", runs on a notebook computer, and captures overwriting input as a sequence of finger strokes from the notebook's touchpad, as shown in Figure 4(a). Figure 4(b) shows the captured ink after preprocessing. One may observe that all the strokes have been moved to the same location irrespective of where they were written on the touchpad, and rescaled to a fixed size while maintaining the aspect ratio. Figure 4(c) shows the Graphical User Interface (GUI) of the IME displaying the recognition result. As mentioned earlier, the unit of recognition is a word, and hence the user writes one word at a time. Once the user has written the word, he/she presses the left-button below the touchpad to trigger recognition, and the recognition result is shown on the GUI. If the word is misrecognized, the user can scroll through other recognition choices by repeatedly pressing the left button. The user may also turn off the visual feedback of the writing (Figure 4(b)) based on his/her preference. The UI also allows the user to clear the last written ink or the last recognized text using the middle button.

(a) **(b)** **(c)**

Figure 4. (a) User overwrites the word "cat" on the touchpad (b) Stroke-level preprocessing (c) Recognition result

The FreePad IME and its user interface described above are only meant to illustrate the core idea of overwriting input. Since the concept of overwriting and the recognition system developed to recognize words are both independent of the front-end GUI, one may customize the user interface based on the task at hand.

5. EXPERIMENTAL EVALUATION

The HMM-based word recognition engine (Figure 3) was trained using the IRONOFF dataset [1]. Isolated lowercase character samples from the dataset were divided into training and test sets, and the training set was used to train character models. In order to evaluate word recognition accuracy, a lexicon containing 1000 most frequently occurring words in the English language was obtained. For each word in the lexicon, three writers were randomly selected from the 137 writers represented in the test character set and a word sample for each writer was then "composed" by simply "concatenating" the test samples of each constituent character written by the same writer. Such

a manner of generating word samples from isolated character samples is conceivable only because of the stroke level preprocessing carried out later, that discards all of the scale and positional information. As a result, 3000 word samples were created for testing. The accuracy of the recognition engine was evaluated on the test word samples for a wide range of lexicon sizes, as shown in Table 1. The decrease in accuracy as the lexicon size increases may be attributed to the inherent perplexity in recognition as the number of words increases. Given the substantially lossy nature of overwriting input, the observed accuracy of 89% with a 20K word lexicon is a promising result to build upon. For comparison, the accuracy of online handwriting recognition on *normal* handwriting input was 93.4% with a 20K word lexicon [2] which is comparable with the top-2 accuracy with overwriting input for the same size of lexicon.

Lexicon Size	Accuracy % (zero rejection)	
	Top 1	Top 2
2K	93.60	95.63
5K	91.97	94.43
10K	90.63	93.87
20K	89.17	93.17
30K	88.5	92.93

Table 1. Word recognition accuracy

6. USER STUDY

As an IME, FreePad is well-suited for scenarios requiring small amounts of text input - for instance SMS messages, notes, chat messages, etc on mobile devices and other devices with limited writing area. However, since overwriting requires a change in the mental model of writing, we subjected the IME to a preliminary user study involving twelve participants to determine user reactions. In particular, we compared the user experience of using FreePad with T9®, a popular text input mechanism for SMS messaging, for the task of composing SMS messages. The users had moderate prior experience with T9 and were asked to enter pre-designed sample messages as promptly as they could without errors. A T-Test on the average time taken by users to enter messages indicates that FreePad is significantly faster than T9. Subjective ratings by the users on a Likert scale of 1(lowest) to 7 (highest) corresponding to three attributes: naturalness, ease of use, and overall experience, are shown in Figure 5. FreePad scores higher than T9 on all three counts, indicating a high acceptance rate for the proposed technology. While limited in scale and scope, this study suggests that overwriting with a finger is easy to use and effective, and users are able to cope with the change in the model of writing.

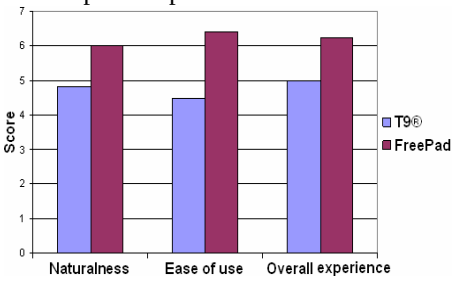

Figure 5. Initial user study – FreePad vs. T9®

7. CONCLUSIONS AND FUTURE DIRECTIONS

In this paper, we proposed a novel text input method for small writing surfaces. The technique based on overwriting in place, allows the throughput of writing continuously on small writing areas using pen or finger, and considerably reduces the cognitive load on the user. The relatively high accuracy of 89% obtained on a 20K lexicon, suggests that with additional context such as that from a language model, accuracies comparable to normal handwriting input may be feasible.

In addition to subjecting the technique to a more rigorous user study and studying user acceptance and throughput vis-a-vis competing approaches such as soft-keyboard and existing handwriting-based text input mechanisms, a number of improvements and extensions to the present system are being considered. Instead of triggering recognition after completing a word, recognition may be performed after each stroke to predict word completions. The approach may also be extended to recognize continuous input (phrases or sentences as opposed to words). Our solution may be extended for entering Indic scripts such as Devanagari and Tamil, which currently do not have any standard IMEs on Pocket PCs and mobile platforms. On the handwriting recognition front, we would like to investigate incorporating additional features, less-reliable information such as relative position between strokes of a character, language models, learning from fewer samples, recognizing mixed (discrete and cursive) styles of writing and writer adaptation.

REFERENCES

1. Christian Viard-Gaudin, Pierre Michel Lallican, Philippe Binter, Stefan Knerr. The IRESTE on/off (IRONOFF) dual handwriting database. *Proc. of the 5th International Conference on Document Analysis and Recognition (ICDAR)*, (1999).
2. Jaeger, S., Manke, S., Reichert, J., and Waibel, A. Online handwriting recognition: The NPen++ recognizer. *International Journal on Document Analysis and Recognition*, 3, (2001), 169–180.
3. Pastor, M., Toselli, A. and Vidal, E. Writing speed normalization for on-line handwritten text recognition. *Proc. of the 8th International Conference on Document Analysis and Recognition (ICDAR)*, (2005), 1131–1135.
4. Rabiner, R. A tutorial on Hidden Markov Models and selected applications in speech recognition. *Proc. of IEEE*, 79(2), (1989), 257-286.