Use of Chatroom Abbreviations and Shorthand Symbols in Pen Computing

William B. Huber[1], Sung-Hyuk Cha[1], Charles C. Tappert[1], and Vicki L. Hanson[2]
[1] *Computer Science Department, Pace University, Pleasantville, NY, USA*
[2] *IBM T.J. Watson Research Center, Hawthorne, NY, USA*
wbhuber@aol.com, scha@pace.edu, ctappert@pace.edu, vlh@us.ibm.com

Abstract

This paper describes an efficient system for entering data into a pen-enabled computer, particularly handheld devices. While keyboard input typically is faster than handwriting input, this is not true for the small PDA interfaces. For this reason, we designed and developed a prototype that uses chatroom abbreviations and shorthand symbols to increase the speed of data entry for these devices. This system was also developed as a prototype system that would enable persons with speech impairments to rapidly convert hand-drawn symbols on a pen-enabled device into speech output. We created a library of chatroom abbreviations and shorthand symbols, and developed a k-nn classification system to recognize the symbols. Experimental results show the effectiveness of the system in terms of speed and accuracy.

Keywords: pen computing, chatroom abbreviations, online character recognition

1. Introduction

Human input to computer systems has been widely studied [1]. While keyboards have been the most popular means, speech recognition and pen computing, among others, have been considered as alternative means that are more natural. However, limited speech capabilities prevent a large number of people from being able to use speech recognition for input. Even for persons who have perfectly intelligible speech, noisy conditions can limit recognition, and using speech input can be socially inappropriate under conditions such as business meetings. In considering an alternative to keyboarding and speech recognition, we turn to handwriting recognition.

The common keyboard layout is QWERTY. Typing with these keyboards requires memory of the keyboard layout and learning the associated motor skills. When learned, however, the input speed, especially with optimized keyboard layouts such as DVORAK, tends to be faster for keyboards than for input from speech or handwriting recognition [1].

As speed becomes more critical in our fast paced lives, users are likely to write faster using word abbreviations. Chatroom users, in fact, have created an abbreviation vocabulary – for example, CU (see you), GA (go ahead), etc. – to increase communication speed. Here, we describe a prototype system that utilizes chatroom abbreviations for pen-computing systems. We hypothesize that using symbols for words will increase both input speed and recognition rates.

The remaining sections are organized as follows. Section 2 reviews the history of Internet chat, focusing on the abbreviations which appear to be growing in popularity. We then discuss online handwritten character recognition offering solutions and dilemmas of the recognition procedures. Section 3 describes the proposed system. The experimental results are to be illustrated in Section 4. Last of all, after we investigate, assess and retrieve results of the use of using Internet chat abbreviations developed for handwriting recognition, we generate a conclusive outlook for the potential new handwriting to speech recognition simulator.[1]

2. Background

In this section, we examine some of the efforts to capture human input in computer systems. In particular, we consider chatroom abbreviations, shorthand alphabets, user-defined symbols, and InkML.

2.1. Chatroom Abbreviations

Real-time chat is one of today's most popular online activities, and several systems allow users to communicate synchronously through text or a combination of text and graphics. Chatroom shorthand uses characters and character sequences to abbreviate words and phrases. Table 1 lists 20 examples of such abbreviations and their corresponding meanings from "The New Hacker's Dictionary" [11].

[1] This paper summarizes a dissertation [6].

Proceedings of the 9th Int'l Workshop on Frontiers in Handwriting Recognition (IWFHR-9 2004)
0-7695-2187-8/04 $20.00 © 2004 IEEE

Table 1. Examples of Chatroom Abbreviations.

Abbr	Word/Phrase	Abbr	Word/Phrase
IMHO	In my humble opinion	BRB	Be right back
BCNU	Be seeing you	LOL	Laughing out loud
BTW	By the way	CU	See you
BYE	End the conversation	CU L8R	See you later
ENQ	Are you busy?	ROTF	Rolling on the floor
FWIW	For what it's worth	TTYL	Talk to you later
FYI	For your information	TTFN	Ta-ta for now
FYA	For your amusement	B4	Before
GA	Go ahead	AFK	Away from keyboard
HHOJ	Ha ha only joking	NHOH	Never heard of him/her

2.2. Shorthand in Pen Computing

Shorthand is any brief, rapid system of writing that may be used in transcribing, or recording spoken words [5]. Perhaps the two most popular shorthand alphabets used in Personal Digital Assistants (PDAs) are Graffiti [9], used in the Palm OS devices, and Papyrus Allegro [10], used in Microsoft OS devices. These alphabets allow for the writing of anything, and they are designed for ease of learning and speed of input. In this study we use the Allegro alphabet (Figure 1). Each Allegro letter is written with a single stroke (pen down to pen up) – thus, i and j are not dotted, and t and x are formed without raising the pen [4].

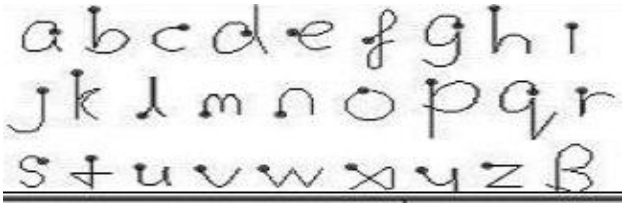

Figure 1. Allegro alphabet.

The chatroom abbreviations described above can be made using the Allegro alphabet symbols. We added chatroom abbreviations to further speed the input by using abbreviations for words and phrases. This combination of chatroom symbols and Allegro allows for even faster input of anything users want to write.

2.3. User-defined symbols

A third type of symbol that we will use is application-dependent, user-defined symbols. For instance, we could have a set of user-defined symbols for hearing impaired persons, for medical diagnosis, for stock market transactions, etc. The user-defined symbols we use here in our prototype system are reasonably general. Several symbols are derived from mathematics, such as "<" and ">" indicating less and more, respectively. Some, for example "ily" (derived from signed handshape for "I love you"), are similar to chatroom abbreviations. Others, such as "Z" meaning "sleep" are from common associations.

The total inventory of symbols consists of the Allegro alphabet and the user-defined symbols. The user-defined symbols must be chosen carefully because there are trade-offs between these symbols and the Allegro alphabet. For example, the user-defined symbols must be sufficiently unlike the Allegro symbols in order to maintain reasonable recognition accuracy. Also, the number of user-defined symbols should be limited for ease of learning and remembering, as well as for recognition accuracy.

2.3. InkML

We briefly mention *InkML*, which is currently under development by W3C [7], because it represents a standardization of digital ink format that is essential for applications such as that proposed here. *InkML* can store representations of hand-drawn digital ink on pen-enabled devices. For instance, in addition to the pen position over time, *InkML* allows recording of information about transducer device characteristics and detailed dynamic behavior to support applications such as handwriting recognition and authentication.

3. Proposed Recognition System

The proposed system has two phases: training and recognition. Figure 2 shows the GUI for these tasks. There are two modes of training, one for the Allegro alphabet and the other for the user-defined symbols. For the latter, users are asked to draw a single stroke and provide the truth (correct character, word, or phrase). The program then records the stroke with its truth into the reference set in the *InkML* format or rejects it if it finds a similar stroke with another meaning.

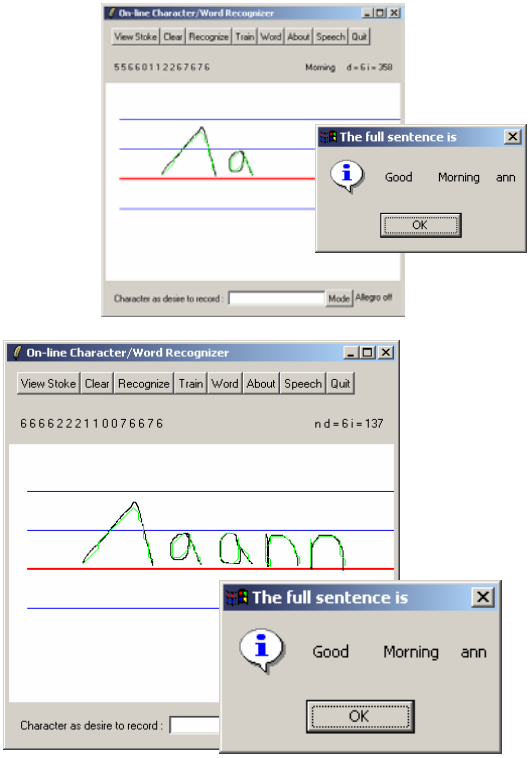

Figure 2. The prototype GUI.

In the recognition phase, users can write a combination of allegro and user-defined symbols but they must indicate mode shifts between the two with the toggle "Mode" button (shown at the bottom of the GUI in Figure 2) with feedback provided to the user (left panel shows "Allegro off" and right shows "Allegro on"). Thus, it is possible to use the same symbol with two different meanings, one meaning in the allegro mode and one in the user-defined mode, as illustrated in the figure with the "a" symbol. The recognition system uses this mode information to direct the recognizer to either the allegro or the user-defined library of symbols. The approximate stroke sequence matching technique [2, 3] is used to recognize the symbols using the k-nearest neighbor classifier. This technique uses eight stroke directions, numbered 0-7, beginning with the left-to-right direction labeled 0, and ordered counter clockwise. For debugging purposes, the sequence of stroke directions is shown in the GUI – for example, the sequence on the right panel, that is "6666222110076676," corresponds to the most recent input symbol "n" and shows the direction sequence of the stroke as down (direction 6), up (direction 2), curving to the right (direction 0), and finally down (direction 6). Upon recognition, the recognized symbol is displayed on the screen and saved in the sentence accumulator. When the desired sentence input is complete, the user touches the 'Speech' button (shown in the top portion of the GUI in Figure 2) to display the full sentence and speak it. A flow diagram of the procedure is shown in Figure 3.

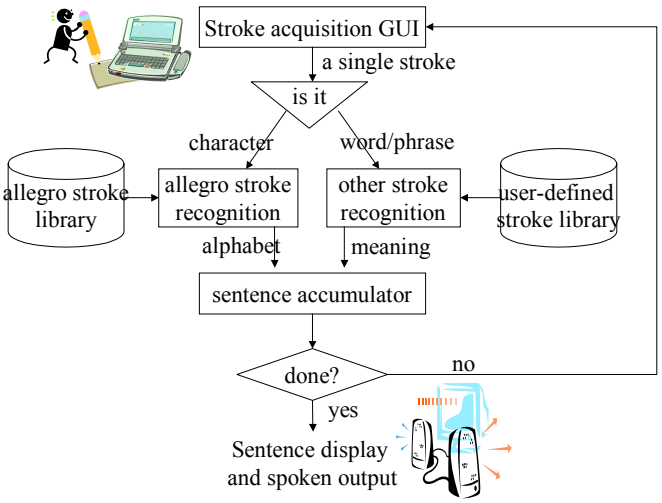

Figure 3. Procedure of the shorthand recognition system.

Figure 4 shows the 65 symbols defined by one user. For each of these user-defined symbols we present the corresponding keyboard symbol(s), the meaning of the symbol, and a handwriting representation of the symbol with an indication of the stroke direction.

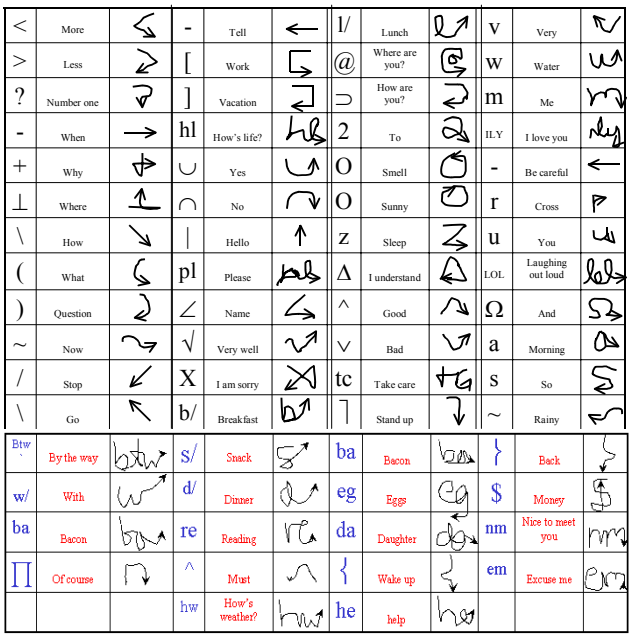

Figure 4. User defined library of shorthand words/phrases.

4 Experimental results

In order to assess the effectiveness of the proposed system relative to existing models in terms of speed and

accuracy, we completed two experiments with the person who had created the symbols shown in Figure 4.

The first experiment evaluated the time for the user to enter sentences in four modes: typing on a standard keyboard, tapping with a stylus on a handheld "soft" keyboard, drawing the complete sequences of alphabet symbols on a pen-enabled device (in this case, an IBM ThinkPad TransNote, a pen-enabled laptop), and finally drawing the sequence of proposed shorthand symbols on the pen-enabled device. The writer input four sentences ten times in each of these four modes, and the time to input each sentence was recorded. The four sentences were derived from sample cases from a book for learning German [8]. For clarification of the shorthand mode, we show the sequences of shorthand symbols corresponding to the four sentences (cases) in Figure 5. Table 2 presents the experimental results, the average time to enter each sentence in each mode. The speed of sentence input using the proposed shorthand symbols is not much greater than standard keyboard input.[2] Notice on the 4th case, the reversing user-defined "m" symbol means "wrong".

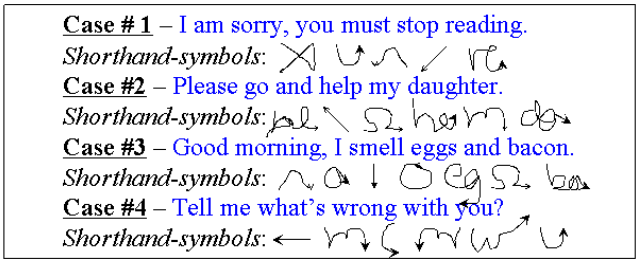

Figure 5. The four case sentences in the proposed shorthand system.

Table 2. Average time to enter each sentence in each mode.

Sentence	Standard PC Keyboard	"Soft" Keyboard on Handheld	Pen Longhand	Pen Shorthand
Case #1	3.92 sec	> 30 sec	8.50 sec	3.92 sec
Case #2	3.75 sec	> 30 sec	8.67 sec	4.17 sec
Case #3	5.03 sec	> 30 sec	9.89 sec	7.39 sec
Case #4	4.39 sec	> 30 sec	8.88 sec	6.78 sec
Mean times	4.27 sec	> 30 sec	8.98 sec	5.56 sec

The second experiment was a performance evaluation. For each input, the recognizer stores the 1st, 2nd, 3rd, 4th, and 5th choices. We computed the accuracy of the handwriting recognizer program for each of these five choices. The 100 symbols used in this experiment were comprised of the 26 alphabet characters + digits 1 to 9 + the 65 user-defined symbols that were stored in the library. The user from the previous experiment drew 10 instances of each symbol on the prototype GUI system. Recognition accuracy was obtained by using a previously developed string matching method [2, 3]. With these 100 symbols the performance was 86% first choice. When looking at the other choices, accuracy increased 100% when the correct symbol was within the top five choices. These results are shown in Table 3. Thus, although the first choice was not always correct, the correct symbol was always within the top five choices. This suggests that improvement in future versions can be enhanced by allowing users to take advantage of the multiple choices or by developing a syntactic postprocessor that could automatically do so.

[2] Larger experiments are planned to establish more meaningful statistical results.

Table 3. Performance evaluation. Shown is the percentage of time in which the correct symbol is recognized when the cumulative choices are taken into consideration.

Choice	Accuracy
1	86.5%
2	92.0%
3	96.4%
4	97.7%
5	100.0%

5 Conclusions and Future Work

This paper describes a unique system of chatroom abbreviations and shorthand alphabet symbols in pen computing, and verifies that it is more efficient than several existing alternatives for computer input. The speed of sentence input using the proposed shorthand symbols is comparable to that of standard keyboard input and is considerably faster than state-of-the-art handheld input modes. The handwriting recognizer does a reasonably good job of recognizing the symbols of the proposed system. Furthermore, we anticipate that a syntactic postprocessor will resolve most of the classification errors and thus make a stronger case that this accuracy is sufficient for a usable system.

Future work should involve eliminating the mode shift to speed input, developing a syntactic postprocessor to increase recognition accuracy, and creating a more user-friendly GUI. For the GUI we recommend using a high-

end programming language such as Java, XML, or WML to provide a state-of-the-art technological environment, perhaps using new solution tools that allow the users the option of clicking buttons, tuning track wheels, or using a pen.

References

[1] Buxton, W., "Human input to computer systems: theories, techniques and technology," Book manuscript 1994/2002.

[2] Cha, S. and Srihari, S. N., "Approximate String Matching for Character Recognition and Analysis," *Pattern Recognition and String Matching* edited by Dechang Chen, ISBN 1-4020-0953-4, December 2002, COMBINATORIAL OPTIMIZATION series Volume 13

[3] Cha, S. and Srihari, S. N., "Writing Speed and Writing Sequence Invariant On-line Handwriting Recognition," *Pattern Recognition From Classical to Modern Approaches* edited by S. Pal and A. Pal, World Scientific Publishing Co. Nov. 2001, p 559-574.

[4] December, John, "Characteristics of Oral Culture in Discourse on the Net," Paper presented at the twelfth annual Penn State Conference on Rhetoric and Composition, University, Park, Pennsylvania, July 8, 1993. http://www.december.com/john/papers/pscrc93.txt

[5] Glatte, H., *Shorthand Systems of the World*, Philosophical Library, 1959.

[6] Huber, William B., "Shorthand and chatroom abbreviations in pen computing," M.S. Dissertation, School of CSIS, Pace University, 2004.

[7] InkML Documents, http://www.w3.org/TR/InkML/

[8] Lasting, I., and Singer, H. "German All the Way" Living Language, 1994.

[9] Palm Computing, "Palm Pilot: Graffiti Reference Card."

[10] Papyrus Associates, "Recognition by Papyrus for Microsoft Windows: User Reference Guide," 1995.

[11] Raymond, Eric S., *The New Hacker's Dictionary*, MIT Press, 1996, also available at http://catb.org/~esr/jargon/oldversions/jarg443.txt