

Common Chatroom Abbreviations Speed Pen Computing

William B. Huber

IBM
2455 South Road
Poughkeepsie, NY 12601 USA
wbhuber@us.ibm.com

Vicki L. Hanson

IBM Watson Research Center
19 Skyline Drive
Hawthorne, NY 10532 USA
vlh@watson.ibm.com

Sung-Hyuk Cha and Charles C. Tappert

CS Dept., Pace University
861 Bedford Road
Pleasantville, NY 10570 USA
{scha, ctappert}@pace.edu

Abstract

While keyboard input typically is faster than handwriting input, this is not true for small PDA interfaces. This paper describes an efficient system for entering data into pen-enabled devices. The prototype uses single-stroke chatroom abbreviations and shorthand symbols to increase the speed of data entry. We created a library of chatroom abbreviations and shorthand symbols, and developed a k-*nn* classification system to recognize the symbols. Preliminary results show the effectiveness of the system in terms of speed and accuracy when compared with traditional input techniques.

1 Introduction

While keyboards have been the most popular means of inputting information to computer systems, more natural systems for input are desirable. Papers at this conference deal with a variety of interfaces including, for example, speech interfaces (e.g., Goldman, R., Price, K J., Lin, M., Feng, J., & Sears, 2005), vision based interfaces (e.g., Kjeldsen, 2005), haptics (e.g., Hwang, Langdon, Clarkson, & Keates, 2005), brain interfaces (e.g., Moore & Allison, 2005) and technologies designed to accommodate needs of users with disabilities (e.g., Myers & Wobbrock, 2005; Trewin, Vanderheiden, & Zimmerman, 2005). The present paper examines another input technology -- handwriting recognition.

Arguably, speech is the most common alternative to keyboard input. Pen technology, however, offers advantages in some specific situations. Most notably, pen technology is more practical in a noisy environment where speech recognition is difficult. It is also less distracting to others in crowded environments such as shared spaces, classrooms, or meetings. Additionally, it provides a practical alternative to many potential users who have no speech ability or poorly intelligible speech. For all users, pen technologies present a valuable alternative to the keyboard for input on small devices such as Personal Digital Assistants (PDAs).

Like speech recognition, handwriting recognition also has some drawbacks. For both, recognition accuracy is an issue, influenced by a number of factors. Input speeds can also compare unfavorably to keyboard input, particularly if error correction is needed. We describe a prototype pen-computing system that utilizes single-stroke chatroom and user-defined symbols, to increase both input speed and handwriting recognition accuracy.

This paper begins with a brief introduction of shorthand alphabets and symbols used in pen computing, followed by a discussion of common chatroom abbreviations and user-defined abbreviations that could be used to facilitate handwriting input. A prototype system is described that uses these techniques, preliminary evaluations of the prototype are presented and future directions are discussed.

2 Using Abbreviations in Pen Computing

Shorthand is any brief, rapid system of writing that may be used in transcribing, or recording spoken words (Glatte, 1959). Single-stroke systems, such as Graffiti and Allegro, have proven particularly popular as shorthand systems for PDA input (Isokoski, 2001). These alphabets provide an alternative symbol set for the Roman alphabet. Each character is written with a single stroke (pen down to pen up) and the alphabet is designed for ease of learning and

speed of input. Specifically, to conform to the single stroke constraint Allegro's *i* and *j* are not dotted, and *t* and *x* are formed without raising the pen. This solves the character level segmentation problem that previously plagued handwriting recognition. Single-stroke recognition algorithms can be relatively simple because there is no need to decide which parts of the curve belong to which character or to wait for more strokes that belong to the same character as is often the case when we try to recognize conventional handwriting.

In the work to be presented here, we use the Allegro alphabet as the foundation to our system, and then add chatroom abbreviations and user-defined symbols. 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 users have created an abbreviation vocabulary to increase communication speed. Abbreviations such as BTW ("By the way"), TTFN ("Ta-ta for now"), and LOL ("Laughing out loud") are common in chatroom conversations, text messaging, and instant messaging. These shorthand abbreviations use characters and character sequences to abbreviate words and phrases.

Table 1 lists additional examples of such abbreviations and their corresponding meanings from *The New Hacker's Dictionary* (Raymond, 1996).

Abbreviation	Word / Phrase
IMHO	In my humble opinion
CU	See you
ROTF	Rolling on the floor
TTYL	Talk to you later
AFK	Away from keyboard
NHOH	Never heard of him / her
GA	Go ahead
FWIW	For what it's worth
HJOJ	Ha ha only joking
FYA	For your amusement
BRB	Be right back

Table 1. Examples of chatroom abbreviations.

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, although reasonably general, were designed primarily for a hearing impaired user to input symbol sequences for conversion to synthetic speech output. 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.

In sum, the total inventory of symbols consists of

- the single-stroke Allegro alphabet
- single-stroke user-defined symbols based on chatroom abbreviations (e.g., IMHO)
- single-stroke user-defined symbols that are application dependent (e.g., ily)

3 Experimental Recognition System

The proposed recognition system has two phases: training and recognition. For the Allegro training, it is easy to learn and fast to execute Allegro alphabets because they are similar to the basic letter shapes. The figure below depicts the first three letters of the Allegro alphabet. Each letter is drawn in a single-stroke, starting at the start-point, shown by a solid dot.

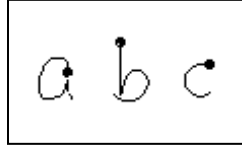


Figure 1. First three Allegro letters.

As with the Allegro alphabet, user-defined symbols had to be single-stroke. Table 2 shows 12 of the 65 symbols defined by one user. For each of these user-defined symbols, the Table shows the keyboard symbol(s), the meaning of the symbol(s), and the single-stroke handwriting representation of the symbol with an indication of the stroke direction.

Keyboard symbol(s)	Meaning	Handwriting representation
>	More	
<	Less	
ily	I love you	
lol	Laughing out loud	
tc	Take care	
\	How	
(What	
--	When	
⊥	Where	
+	Why	
u	You	
@	Where are you?	

Table 2. User defined library of shorthand words/phrases.

For both the Allegro and user-defined symbols, training involves the subject drawing a single stroke and providing 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. *InkML* represents a standardization of digital ink format that is essential for applications such as that proposed here (Chee et al., 2004). It 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.

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 interface in Figure 1) with feedback provided to the user. 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. Figure 1 shows “Allegro on”.

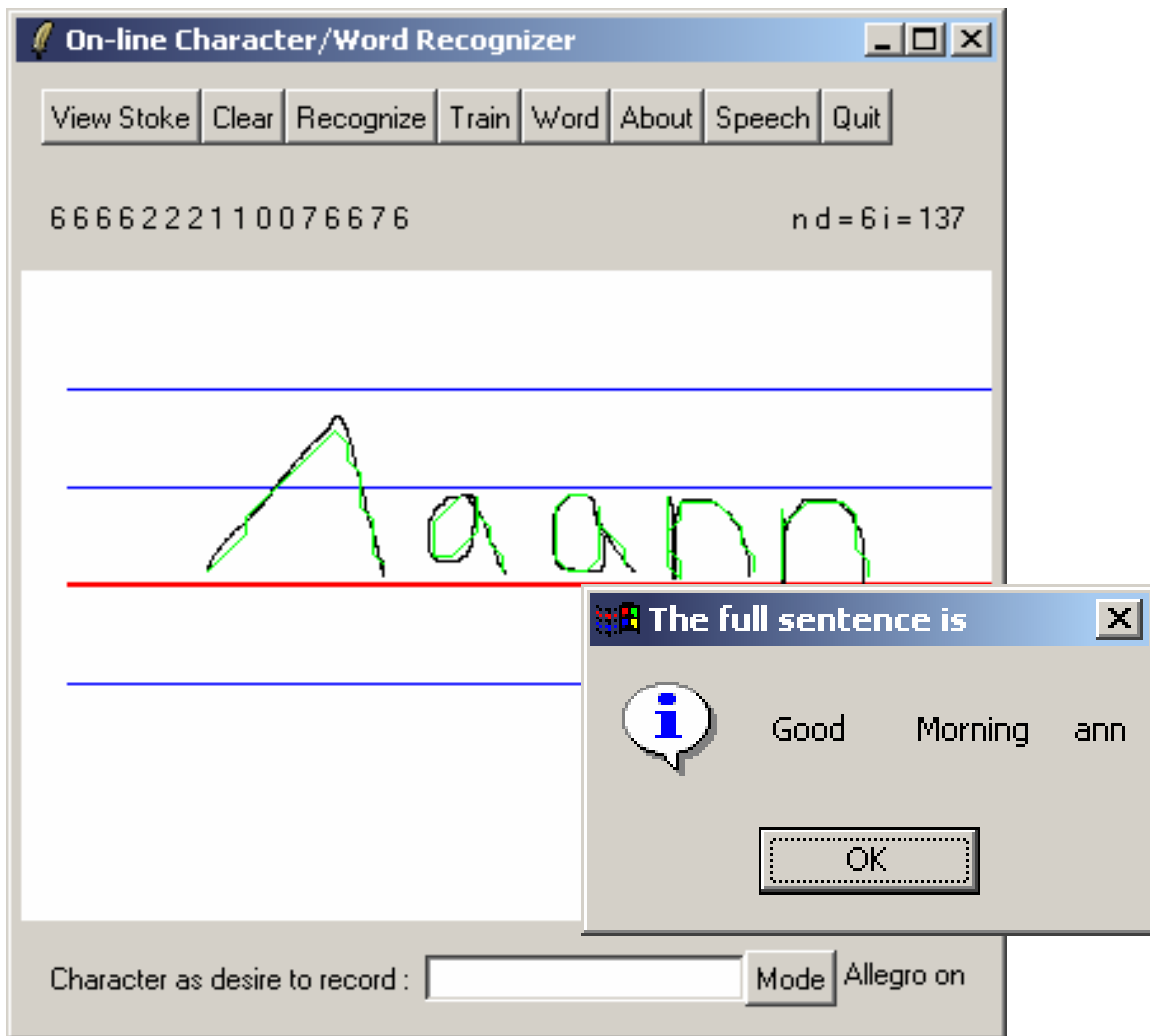


Figure 2. User interface for recognition system.

The approximate stroke sequence matching technique (Cha & Srihari, 2001, 2003) 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 example, the sequence

“6666222110076676” (Figure 1) 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).

In the prototype system, 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 interface in Figure 1) to display the full sentence and speak it.

4 Evaluation

In order to assess the effectiveness of the proposed system relative to existing models in terms of speed and accuracy, we completed two preliminary evaluations with the person who had created the symbols shown in Table 2.

For the first evaluation, the time for the subject to enter sentences was examined using an IBM ThinkPad TransNote, a pen-enabled laptop. The time to enter the sentences was compared drawing the complete sequences of alphabet symbols on the devices and when using the experimental shortcut system. For comparisons, times were also obtained for entering these sentences by typing on a standard PC desktop keyboard and by tapping with a stylus on a PalmPilot “soft” keyboard. The subject input four sentences ten times in each of these four modes, and the time to input each sentence was recorded. Four simple sentences were used (Huber, 2004). Figure 2 shows the shorthand symbols corresponding to one of the test sentences. The resulting data consisted of 160 data points for each of the four conditions.

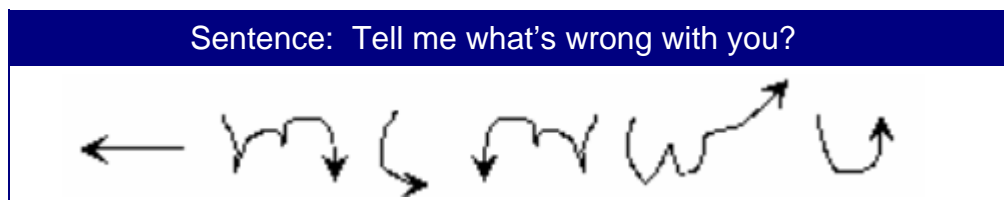


Figure 3. An example test sentence and the shorthand symbols.

Keyboard typing was the fastest condition, with the mean data-entry time being only 4.27 sec. The subject for this evaluation was skilled with using a PC, so this result is not surprising. Entering the same sentences on the PDA ‘soft’ keyboard was considerably longer, with the mean data-entry time being greater than 30 sec.

Of interest here is the comparison of the two conditions using the pen-enabled laptop. The pen longhand condition took considerably longer for data entry than the pen shorthand condition. The mean times were 8.98 sec and 5.56 sec, respectively to enter the sentences in these two conditions. Notably, the speed of sentence input using the shorthand symbols was considerably shorter than the soft keyboard PDA times and not much greater than standard keyboard input. While further work with additional subjects is need to confirm these findings, this pilot work suggests that the shorthand system has the potential to bring pen input rates nearly comparable to keyboard typing. Research using a PDA device comparing inputs rates across ‘soft’ keyboard entry, longhand entry, and shorthand entry is needed.

For the second evaluation, symbol recognition accuracy was examined. For each input symbol, the recognizer stores the 1st, 2nd, 3rd, 4th, and 5th choices. The accuracy of the handwriting recognizer program was computed 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 subject from the previous evaluation produced 10 instances of each symbol on the prototype system, making a total of 1000 data inputs.

Recognition accuracy was obtained by using a previously developed string matching method (Cha & Srihari, 2001, 2003). With these 1000 drawings of the symbols, the 1st choice was correct 86.5% of the time, and accuracy

increased to 100% when the correct symbol was within the top five choices. Table 3 shows the percentage of time in which the correct symbol is recognized from the cumulative choices.

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

Table 3. Results of symbol recognition evaluation.

5 Conclusions

Although further experiments with a larger number of subjects, and possibly more samples per subject, are needed to confirm the results of this pilot study, the results presented here indicate a promising approach to solving problems that have plagued handwriting recognition systems. A unique system of chatroom abbreviations and shorthand alphabet symbols in pen computing was described and appears to be more efficient than existing longhand input. The speed of sentence input using the proposed shorthand symbols was nearly comparable to that of standard keyboard input and was considerably faster than state-of-the-art handheld input modes. The handwriting recognizer has done a reasonably good job of recognizing the symbols of the proposed system. Moreover, we anticipate that a syntactic postprocessor would resolve most of the classification errors and thus make a stronger case that this accuracy is sufficient for a usable system.

Future work should eliminate the mode shift to speed input, develop a syntactic postprocessor to increase recognition accuracy, and create a more user-friendly interface. For the interface 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 give the user the option of clicking buttons, tuning track wheels, or using a pen. Of particular interest would be further evaluations by persons with speech impairments who would benefit from the ability to rapidly convert hand-drawn symbols on a pen-enabled device into speech output.

6 Acknowledgements

This research was conducted as part of an M.S. Dissertation by the first author for the School of CSIS, Pace University.

7 References

- Cha, S. & Srihari, S. N. (2001). Writing Speed and Writing Sequence Invariant On-line Handwriting Recognition. In S. K. Pal & A. Pal (Eds), *Pattern Recognition From Classical to Modern Approaches* (pp 559-574). World Scientific Publishing Co.
- Cha, S. & Srihari, S. N. (2003). Approximate String Matching for Character Recognition and Analysis. In D Chen (Ed), *Pattern Recognition and String Matching*. Springer.
- Glatte, H. (1959). *Shorthand Systems of the World; A Concise Historical and Technical Review*, London: Selbstverl.

- Goldman, R., Price, K J., Lin, M., Feng, J., & Sears, A. (2005, in press). Speech interactions for mobile devices: A natural solution to a contextual conundrum. In *Proceedings of Human-Computer International HCII'05*, Mahwah, NJ: Erlbaum.
- Huber, William B., (2004). Shorthand and chatroom abbreviations in pen computing. M.S. Dissertation, School of CSIS, Pace University, Pleasantville, NY.
- Hwang, F., Langdon, P., Clarkson, J., & Keates, L. S. (2005, in press). A haptic toolbar for motion-impaired users. In *Proceedings of Human-Computer International HCII'05*, Mahwah, NJ: Erlbaum.
- Chee, Y., Magaña, J., Franke, K., Froumentin, M., Russell, G., Madhvanath, S., Seni, G., Tremblay, C., & Yaeger, L. (2004). Ink Markup Language. Retrieved February 16, 2005, from <http://www.w3.org/TR/InkML/>
- Isokoski, Poika. (2001). Model for unistroke writing time. In *Proceedings of the SIGCHI conference on Human factors in computing systems* (pp. 357 – 364). New York, ACM.
- Kjeldsen, R. (2005, in press). Exploiting the flexibility of vision-based user interactions. In *Proceedings of Human-Computer International HCII'05*, Mahwah, NJ: Erlbaum.
- Moore, M., & Allison, B. (2005, in press). Brain-Computer Interfaces for People with Severe Disabilities. In *Proceedings of Human-Computer International HCII'05*, Mahwah, NJ: Erlbaum.
- Myers, B., & Wobbrock, J. (2005, in press). Text Input to Handheld Devices for People with Physical Disabilities. In *Proceedings of Human-Computer International HCII'05*, Mahwah, NJ: Erlbaum.
- Raymond, Eric S. (1996). The new hacker's dictionary. Cambridge, MA: MIT Press. Also, available as *The Jargon File Version 4.4.3*, retrieved February 16, 2005, from <http://catb.org/~esr/jargon/oldversions/jarg443.txt>
- Trewin, S., Vanderheiden, G., & Zimmermann, G. (2005, in press). Hello, what do you do?: Natural Interaction with Unfamiliar Technologies. In *Proceedings of Human-Computer International HCII'05*, Mahwah, NJ: Erlbaum.