Schomaker, L.R.B. (1994). User-interface aspects in recognizing connected-cursive handwriting. Proceedings of the IEE Colloquium on Handwriting and Pen-based input, 11 March, London: The Institution of Electrical Engineers, Digest Number 1994/065, (ISSN 0963-3308).

User-interface Aspects in Recognizing Connected-Cursive Handwriting

L. Schomaker[1]

1 Introduction

There are at least two major stumbling blocks for user acceptance of pen-based computers: (1) the recognition performance is not good enough, especially on cursive handwriting, and (2) the user interface technology has not reached a mature stage. The initial reaction of product reviewers and potential user groups to pen-based computers varies. The realistic assessment is that this is a technology with a large potential, but still too brittle for serious real-world use. The application area of form filling, for instance, is characterized by acceptable digit recognition rates, but the essential punctuations such as decimal points and commas are still handled poorly. Similarly, there are problems in handling text input. Novice users seem to have a very high expectation of the pen technology. They start quickly writing a paragraph of text in their style (mostly a mix of handprint and cursive) and are amazed at the resulting strings of question marks and characters which they do not remember having entered. Also, whereas users seldomly blame the keyboard for their typing errors, the pen computer is blamed for not recognizing script which shows evident shape errors after close inspection. It seems to be clear that there are problems both at the system and at the user side.

As long as the pattern recognition systems fail to recognize clearly-written material and keep coming up with counter-intuitive recognition results, there is room for refining the algorithms. Current problems have to do with the multi-cultural handwriting styles and methods for adapting to new users. The diversity of handwriting styles, with often conflicting shapes, makes the notion of a true user-independent recognition system seem unrealistic. In a project on within- and between-writer variability it was found that the number of stroke-shape interpretations in cursive script keeps increasing with each new writer in a trainable system, and the existence of an asymptote was not apparent. The problem is that in a large set of templates, a substantial portion of the templates will act as noise with respect to the handwriting style of the current user. Given the fact that training is useful to ameliorate this problem, how can we be sure that end users are able to train the system in a consistent way? Already, the importance of user interface design becomes evident.

Furthermore, there are several indications - to be discussed below - that the upper limit of the recognition rate in practical applications is lower than the 100% goal. In order to increase the user acceptance, this deficit must be compensated for, and the place where this must happen is the user interface. This is not so strange, since even the keyboard only became usable after the addition of edit control keys like "Rubout".

[1]NICI, Nijmegen University, P.O. Box 9104, 6500 HE Nijmegen, The Netherlands

2 Handwriting glitches: common errors

In what follows, we will concentrate on the entering of textual material, but much of it probably holds for other forms of input (numbers, gestures) as well. A collected set of handwritten words from a text copying task can be categorized in a number of ways.

- There may be canceled material, e.g., writers starting a word and scribbling over it in an unpredictable way.
- The input may consist of discrete noise events, like dots or short lines caused by inadvertently dropping or tapping the pen on the writing surface.
- Also, the input may consist of badly formed shapes, illegible by both human and machine.
- Still another type of input is legible by humans but not by the algorithm, e.g., in case of fused characters. If an algorithm was designed with the constraint that individual characters should be present and correctly shaped, this type of input is deemed inappropriate for testing. However, there may exist another algorithm which can handle it.
- Individual characters may be legible but the words are badly spelled.
- Words are written correctly, but are unsollicited (not prompted for) in the data collection process.
- The handwriting input may contain device-generated errors: random noise, stiction of pen-down switches, unresponsiveness of switches, pen tilt errors.

This list is not exhaustive: more types of errors exist. The difference between reported academic pattern recognition rates and the real-world recognition rates which an end user does experience comes from the useful academic practice to discard samples falling in any of the above categories. Thus, a typical test set is of the generic handwriting style the algorithm was designed for; it does not contain spelling or device errors; and is what the scientific community considers to be "clean data".

3 Usability problems in "truthing" as indicators of the recognition performance asymptote

Indeed, the laboratory benchmarking of pattern recognition algorithms, as envisaged in the UNIPEN project [2], will have to be based on neat data. Such data are obtained during a truthing process, where human operators assess the data prior to execution of an actual test of recognition performance. Already at this level, problems arise, despite the expected high recognition and cognition capabilities of the truther. In fact, the truthing job in cursive-connected handwriting is virtually impossible without prior knowledge of the text or a list of words which were presumably entered by the subject. This problem was studied in four small-scale experiments, using different types of contexts, presenting human readers with samples of cursive handwriting or handprint written with ballpoint on white paper (Table 1).

Exp.	Context	Style	Writers	Target Words	Readers	Words	Recog-nized
A	frequently-used Dutch words	handprint	1	30	12	N=360	98%
	frequently-used Dutch words	neat cursive	1	30	12	N=360	88%
B	sentence fragments, same writer	cursive	3	15	12	N=180	85%
C	sentence fragments, same writer	cursive	4	13	20	N=260	85%
	unrelated words, same writer	cursive	4	13	20	N=260	77%
D	unrelated words, same writer	fast cursive	12	12	15	N=180	72%
	unrelated words, different writers	fast cursive	12	12	15	N=180	54%

Table 1. Human recognition rates of handwritten words on paper. Explanation of **Exp**eriments. A: single-word recognition; B-D: three-word sequences, middle-word recognition

As can be observed, only in handprint does human recognition reach a high recognition rate of 98%, whereas the recognition of cursive words varies from 54-88%. It is most likely that in the truthing of samples of handwriting the operator is confronted with similar problems like the subjects in these experiments (or worse, due to CRT screen resolution). The combination of sloppy writing and absence of linguistic and shape context leads to a poor 54%. These findings suggest that it is unrealistic to assume that the asymptote for cursive recognition is 100%. In real applications things are worse since there is no extra intelligence like a truther between the tablet and the recognition algorithm. Traditionally, this problem is approached by trying to improve the preprocessing of the data, but there will always be a portion of the input which is inherently problematic. As an example we can take the process which segments a stream of pen movements into words. Whatever heuristics are used, there will always be a portion of the input which is segmented erroneously (e.g., because the white space between two words happens to be 0.1 mm closer than the threshold used). The portion of badly segmented words will lower the net recognition rate considerably. Below, a number of solutions to these types of problems are proposed.

4 User-interface solutions for improving handwriting recognition rate

The solution to these types of problems is to design user interfaces which:

1. Constrain the writer where possible
2. Give the writer more control
3. Provide more powerful mechanisms for error handling
4. Clarify to the user what is going on.

4.1 Constrain the writer?

In order to obtain cleaner pen data, the writer's attitude may be constrained in some acceptable way. Boxes and so-called combs are useful in handprint but not in cursive script. A single guideline is helpful in imposing constraints on cursive script, but this does not by itself facilitate the temporal segmentation. Current cursive word recognition technology requires a segmentation into words due to the dependence on a lexicon. Since automatic word segmentation is almost as brittle as the word recognition itself, new pen-driven word segmentation techniques have to be defined. Again, in the case of the keyboard, it is a generally accepted practice to use the "Enter" or "Return" key for input validation. In our lab a number of pen-driven word segmentation methods are under study. Three methods are considered here:

1. Gesture-driven word segmentation where the user enters a tap on the digitizer on the right of the end of the current word ($> 2cm$);

2. "OK-button" word segmentation;

3. Time-out driven word segmentation.

Subjects performed a lowercase text-copying task. There were three texts, randomly distributed over the three conditions, and all subjects copied a different text in all three conditions. Results are given in Table 2. Results for three time-out values (0.8,1.0, and 1.4s) were combined since differences where statistically negligible. The entered words were off-line truthed and processed by a stroke-based cursive recognition program reported earlier [1]. It was trained on the handwriting of 32 multinational writers and used without extra training for the 18 subjects. The word list for lexical post processing contained the words present in the text to-be-copied. Subjects were allowed to write in their own style, which was mainly mixed cursive and handprint, and often contained small-written capital characters as "lower case". The recognizer was originally designed for connected cursive.

	Gesture	OK-button	Time-out	
Text reading time/word	2.0	1.9	1.7 [s]	N.S.
Writing time/word	2.9	2.5	2.5 [s]	p ¡ 0.001
Top word correct	56% (20-81)	62% (21-80)	60% (8-90)	N.S.
Correct word in top five	64% (24-88)	70% (21-88)	67% (13-94)	N.S.
Avg. words/subject	55	53	53	N.S.

Table 2. Human reading and writing times, and machine recognition rates as a function of word segmentation method (N=18 subjects)

Statistically, the only significant difference is in word writing time, due to slower writing in the gesture method. This may be due to cognitive overhead in anticipation of the newly learnt gesture movement. From the user-interfacing point of view, the "OK-button" method has the advantage that it is compatible and consistent with a "Cancel-button" option, for input which the user considers to be sloppy. In the other two methods, this can only be robustly solved by allowing for post-hoc editing.

4.2 Give the writer more control

New visual widgets and methods have to be designed for handling the input of material for which no acceptable recognition performance can be reached. Punctuations are often difficult to recognize, because they may interfere with the cursor placement gesture or text selection methods. A visible and limited-size toolbar for punctuations [. , ' ' : ;] is probably perfectly acceptable. Gestures are nice, but they require highly motivated users who must store and recall them in/from human memory. The success of todays desktop metaphors is largely due to the fact that functions and objects are identified by visual association, a much less demanding cognitive task than recall from memory. Also needed are easier methods for cursor placement and cutting and pasting. Currently, sometimes two modes of cursor control are implemented, confusingly mixed at the same time: relative (to the current cursor location) and absolute (below the pen tip) mode.

4.3 Improve the recovery from error

The rate of erroneous movement in a pen-based system can be very high, especially in novice users. Therefore, a single-level "Undo" does not suffice and a large "Undo" stack is necessary to restore an earlier uncorrupted state of the system.

4.4 Clarify to the user what is going on

Instead of entering directly a recognition result into the application, the user interface must allow for input validation by the user. Less intrusive, before executing gestures or entering characters, symbols may be graphically echoed in their shape-as-recognized, both to give the user confidence in control over the pen computer and to remind him/her of the shapes known by the system.

4.5 Give up the paper metaphor

Some of the problems in pen computing experienced today are most probably due to the fact that the paper-mimicking approach without on-line guidance leads to unrealistic user expectations and consequently low real-life recognition rates.

5 References

1. Schomaker, L.R.B. (1993). Using Stroke- or Character-based Self-organizing Maps in the Recognition of On-line, Connected Cursive Script. Pattern Recognition , 26(3), 443-450.

2. Guyon, I., Schomaker, L., Plamondon, R., Liberman, M. & Janet, S. (1994, in press). The UNIPEN project of data exchange and recognizer benchmarks. To Appear in Proceedings of the 12th ICPR, October 9-13, Jerusalem.