Text Entry Performance of State of the Art Unconstrained Handwriting Recognition: A Longitudinal User Study

Per Ola Kristensson and Leif C. Denby
Cavendish Laboratory, University of Cambridge
JJ Thomson Avenue, CB3 0HE, Cambridge, United Kingdom
{pok21, lcd33}@cam.ac.uk

ABSTRACT
We report on a longitudinal study of unconstrained handwriting recognition performance. After 250 minutes of practice, participants had a mean text entry rate of 24.1 wpm. For the first four hours of usage, entry and error rates of handwriting recognition are about the same as for a baseline QWERTY software keyboard. Our results reveal that unconstrained handwriting is faster than what was previously assumed in the text entry literature.

Author Keywords
Handwriting, handwriting recognition, software keyboard

ACM Classification Keywords
H5.2. User interfaces: Input devices and strategies.

INTRODUCTION

Unconstrained handwriting recognition means that the recognizer simultaneously accepts hand-printed characters, cursive script, and a combination of both. It is not until recently that unconstrained handwriting recognition has become accurate enough to be practical. In 1995, Frankish et al. [4] concludes: "…at present, handwriting recognition is unlikely to be an effective method for unconstrained text input…" LaLomia [5] found that users typically require that only one character out of 100 is wrong (1% error rate) before sending a text message to their superior. This goal has been unattainable for unconstrained handwriting recognition until very recently.

Interestingly, ignoring the problem of recognition errors, the text entry rate for unconstrained handwriting recognition has remained an open issue. The early user studies of handwriting recognition (e.g. [4,6]) only studied aspects of isolated character-recognition. Many sources claim that unconstrained handwriting is fundamentally slow, citing e.g. Card et al. [2] or Seibel [9]. However, none of the above sources did actual research on handwriting performance. Instead, they in turn cite Devoe [3]. What Devoe [3] actually did was to let three participants copy a 100 word long text message using four different text entry methods over ten sessions. The unconstrained handwriting method was only used at the first two sessions and at the last one (reasons unknown). Devoe [3] found that his three participants had a grand mean text entry rate of 16.4 wpm for unconstrained handwriting (calculated based on Table 1 in [3]). No handwriting recognizer was used in [3] so this data point only covers articulation time and ignores any error correction time (which would inevitably occur if a handwriting recognizer was used). Occasionally Bailey [1] is cited as giving a 25 wpm upper bound of unconstrained handwriting (often as a secondary or even tertiary source). However, the only plausible source for this number in [1] is estimation from a log-scale diagram of text entry measurements in [1], which again refers back to Devoe [3].

In the recent text entry surveys it has been either implicitly [10] or explicitly [7] assumed that studies of unrecognized handwriting speed represent an upper bound for handwriting recognition text entry performance. The argument is centered on the fact that unrecognized handwriting lacks a verification and correction phase (since there cannot be any recognition errors). Therefore, it is assumed participants are writing as fast as they could if they were using a 100% accurate handwriting recognizer. Under this hypothesis, the average handwriting entry rate of 16.4 wpm in Devoe's [3] study would classify handwriting recognition as a relatively slow text input method, well below the QWERTY software keyboard baseline [7,10]. However, without empirical measurements this is just an educated guess based on a small-scale study.

In this paper we answer two research questions. First, what is the actual text entry performance of state of the art unconstrained handwriting recognition? Second, how does handwriting recognition compare as a text entry method to the well-understood status-quo QWERTY software keyboard?

METHOD

Participants

We recruited 12 volunteers from the university campus. We intentionally wanted a rather broad sample and recruited participants from many different departments with many

CHI 2009, April 4–9, 2009, Boston, Massachusetts, USA.

different backgrounds. Six were men and six were women. Their ages ranged between 22-37 (mean = 27, sd = 4). Participants were screened for dyslexia and repetitive strain injury (RSI). Seven participants were native English speakers and five participants had English as their second language. No participant had used a handwriting recognition interface before. One participant had used a software keyboard before. No participant had regularly used a software keyboard before. Participants were compensated £10 per session.

Apparatus and Software

We used a Dell Latitude XT Tablet PC running Windows Vista Service Pack 1. The 12.1" color touch-screen had a resolution of 1280 × 800 pixels and a physical screen size of 261 × 163 mm. Participants used a capacitance-based pen to write directly onto the screen in both conditions.

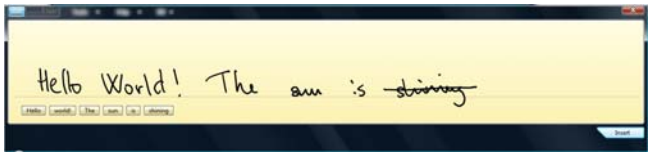

Figure 1. The handwriting recognition interface. Recognized words are displayed as buttons below the writing area.

The handwriting recognizer (Figure 1) was configured to learn and adapt to participants' handwriting style (the default setting on Windows Vista). Each participant performed the experiment in a separate user account on the machine to ensure handwriting adaptation was carried out on an individual basis. There was a potential confound in enabling handwriting adaptation since it caused the system, as well as the user, to learn as a function of usage. In the interest of external validity we enabled adaptation since in actual use users would most likely have adaptation turned on.

Figure 2. A word-edit box is displayed when the user presses the button for a recognized word.

The handwriting recognizer had two basic modes for correction. One was to strike out one or more words by crossing them and then rewriting the misrecognized text (see Figure 1). Another method was to click on the button for a recognized word. This action brought up a correction interface that enabled the user to edit the word character by character using a letter recognizer (see Figure 2).

For the software keyboard condition we used the default QWERTY software keyboard on Windows Vista.

Both the handwriting recognizer and the software keyboard were docked to the lower part of the screen. The dimensions of the software keyboard were 1266 × 244 pixels and 257 × 50 mm. The dimensions of the handwriting recognizer writing area measured 1266 × 264 pixels and 257 × 55 mm. The slight additional height difference was because the handwriting recognizer had a small INSERT button below the writing area to the right (cf. Figure 1). The dimensions of both text entry methods' areas were held constant during the experiment. A letter key on the keyboard measured 14 × 11 mm, SPACEBAR measured 72 × 10 mm and BACKSPACE measured 23 × 11 mm.

Procedure

The experiment consisted of one introductory session and ten testing sessions. In the introductory session the experimental procedure was explained to the participants. Participants were shown how to use the software keyboard and the handwriting recognizer, including demonstrations of how to correct errors. They also completed the built-in Tablet PC tutorial.

Each testing session lasted slightly less than one hour. Testing sessions were spaced at least 4 hours from each other and subsequent testing sessions were maximally separated by two days. In each testing session participants did both conditions (software keyboard and handwriting recognition). The order of the conditions alternated between sessions and the starting condition was balanced across participants. Each condition lasted 25 minutes. Between conditions there was a brief break. Participants were also instructed that they could rest at any time after completing an individual phrase.

In each condition participants were shown a phrase drawn from the phrase set provided by MacKenzie and Soukoreff [8]. Each participant had their own randomized copy of the phrase set. Participants were instructed to quickly and accurately write the presented phrase using either the software keyboard or the handwriting recognizer. Participants were instructed to correct any mistakes they spotted in their text. In the handwriting condition we instructed participants to write using their preferred style of handwriting (e.g. printed, cursive or a mixture of both). After they had written the phrase they pressed a SUBMIT button and the next phrase was displayed. The SUBMIT button was a rectangular button measuring 248 × 16 mm. It was placed 9 mm above the keyboard and handwriting recognizer writing area.

RESULTS

In total we collected 100 hours of data. In each session participants inputted on average 83.2 (sd ≈ 13.7) phrases in the software keyboard condition and 81.5 (sd ≈ 16.7) phrases in the handwriting recognition condition. All statistical analyses were conducted using repeated-measures analysis of variance (ANOVA) at significance level $\alpha = 0.05$.

Entry Rate

Entry rate was calculated as words-per-minute, with a word defined as five consecutive characters. The time required to write each phrase was defined as the interval between when

the participants first pressed down the stylus on either writing interface (software keyboard or handwriting area) until they pressed the SUBMIT button.

As expected, participants became faster with practice ($F_{9,99} = 2.4616$, $p < 0.05$), see also Figure 3. In the first session, the mean entry rate was 19.6 wpm (sd ≈ 3.2) for software keyboard and 21.5 wpm (sd ≈ 3.7) for handwriting recognition. In the last session, the mean entry rate was 24.9 wpm (sd ≈ 5.0) for software keyboard and 24.1 wpm (sd ≈ 5.0) for handwriting recognition. As is evident in Figure 3, the mean entry rate increased faster for the software keyboard than for the handwriting recognizer. At session six the software keyboard became faster than the handwriting recognizer. The mean entry rate difference between software keyboard and handwriting recognition was not significant ($F_{1,11} = 0.0699$, $p = 0.7963$).

Error Rate

Error rate was calculated as the minimum edit-distance between the phrase shown (stimuli) and the phrase actually entered by the participant (response), divided by the number of characters in the stimuli phrase. This means that if a participant entered the phrase completely correct the error rate was minimized to zero. If a participant entered a phrase completely incorrect the error rate was maximized to unity. Note that this was the corrected error rate, and it was not a measure of recognition accuracy.

Participants' error rates did not significantly vary with practice ($F_{9,99} = 0.8982$, $p = 0.53$). The grand mean error rate was 1.4% (sd ≈ 1.6%) for software keyboard and 1.0% (sd ≈ 1.0%) for handwriting recognition. The mean error rate difference between software keyboard and handwriting recognition was not significant ($F_{1,11} = 4.668$, $p = 0.05468$).

Error Correction Time

The handwriting recognizer we used did not enable us to measure recognition accuracy directly. Instead we opted for estimating participants' error correction time: the proportion of their writing time spent correcting errors. Note that this class of errors includes both recognition errors and errors caused by participants (e.g. misspellings).

In the handwriting recognition condition we stored participants' pen traces. These were later played back so we could detect when users were crossing out words or invoking the word-edit box. We built a software tool to automatically detect participants' error correction actions using model fitting techniques. Thereafter we semi-automatically marked up the log file segments when participants were correcting the text. Using this data we then calculated the proportion of participants' writing time that was spent correcting errors.

In the keyboard condition the procedure went as follows. For each phrase we calculated the ratio between unnecessary keystrokes and total number of keystrokes needed to form the input the user committed when pressing SUBMIT. This gave us an estimate of the proportion of time users spent correcting text because superfluous keystrokes (that didn't contribute to the committed text) must have been either BACKSPACE presses or additional letters inserted by the participant to make up for previous BACKSPACE presses. We assumed the time taken to hit any key was uniformly distributed. In addition to simply correcting errors by pressing BACKSPACE multiple times, the participant could also select an entire block of text and delete it by pressing BACKSPACE once. These cases were automatically detected and in these instances we retrieved the correction time by manual inspection of the log files.

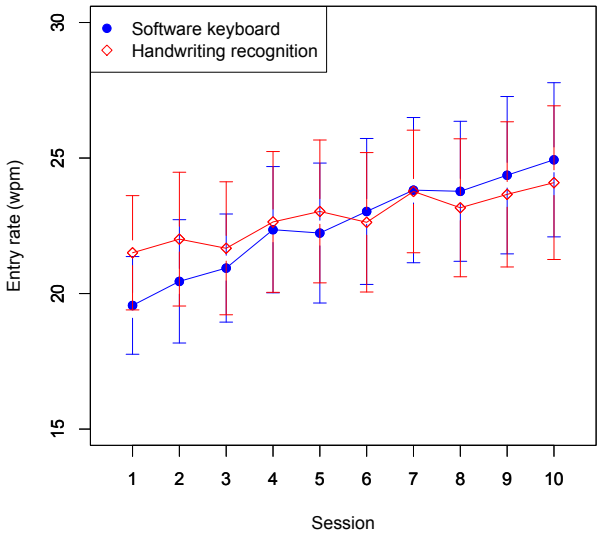

Figure 3. Mean entry rate (wpm) and 95% confidence intervals as a function of session number. Note that the y-axis is cut off at 15 wpm at the bottom.

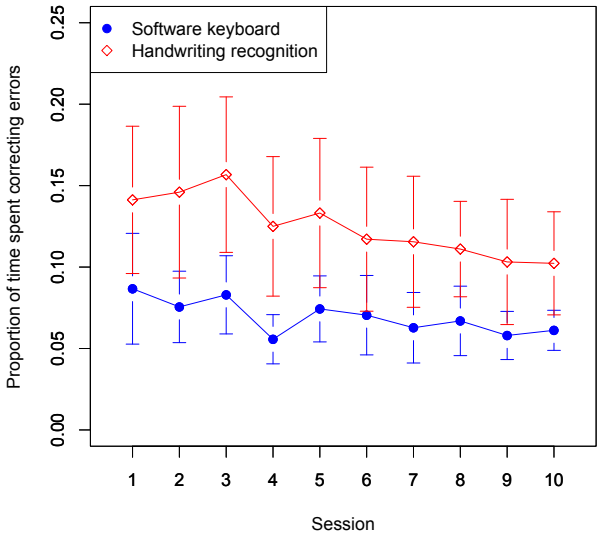

Figure 4. Mean proportion of time spent correcting errors and 95% confidence intervals as a function of session number.

Figure 4 plots the proportion of error correction time for both software keyboard and handwriting recognition as a function of session. The difference is significant ($F_{1,11} = 5.469$, $p < 0.05$). Participants spent more time correcting

errors using handwriting recognition than with the software keyboard. This indicates that there is still room for improvements in handwriting recognition accuracy.

Subjective Ratings

Table 1 summarizes the mean subjective ratings ranging from 1 ("Strongly Disagree") to 7 ("Strongly Agree") on a Likert scale. The only clear difference was that participants perceived handwriting as more fun than software keyboard.

Statement	SK	HW
It was easy to correct errors.	4.9 (1.4)	4.6 (1.7)
The SK/HW was accurate.	4.7 (1.6)	4.5 (1.5)
It is fun to use SK/HW	4.2 (1.6)	5.3 (1.5)

Table 1. Mean subjective ratings (standard deviations in parentheses) aggregated over all sessions.

DISCUSSION

As alluded in the introduction, text entry performance of unconstrained handwriting recognition has not been well studied before. In comparison to Devoe [3] our participants were much faster, even after the first session (21.5 wpm vs. 16.4 wpm in [3]). At the last session, our participants had a mean entry rate of 24.1 wpm. Last, our participants' final text had a character-level mean error rate of 1%. This error rate level has been previously described by users as the acceptable error level for communicating with superiors [5].

We believe the difference in relation to Devoe [3] can be attributed due to primarily two factors. First, Devoe [3] studied fewer participants (n = 3) and let them write less text. Second, we hypothesize that the presence of recognition feedback resulted in a positive feedback-loop between the recognizer's output and participants' behavior. This pushed our participants to write faster and less precise as they observed how much the recognizer could tolerate. In comparison, in [3] participants received no such feedback and most likely felt they had to write legibly enough for another person to be able to transcribe their handwriting.

CONCLUSIONS

In the recent text entry surveys [7,10] handwriting recognition is hypothesized to be relatively slow at around 16 wpm (excluding a 25 wpm upper-bound estimation derived from a mistaken secondary reference to [1] in [7]) and the QWERTY software keyboard to be faster at 25-40 wpm. However, we found that handwriting recognition performs almost identical to a QWERTY software keyboard, at least for the first hours of usage. During the first 25 minutes of use, participants wrote text using the handwriting recognizer at 21.5 wpm. This was 2 wpm faster than the software keyboard. After 250 minutes of writing, participants wrote text at 24.1 wpm with the handwriting recognizer and 24.9 wpm with the software keyboard. We found no statistical difference between handwriting recognition and software keyboard in either text entry rate or error rate. Further, the 95% confidence intervals of both methods' mean entry rates were very close (Figure 3). Taken together, text entry performance of state of the art unconstrained handwriting recognition appears to have been underestimated in the literature. In addition, our participants also thought handwriting recognition was more fun to use than the software keyboard. We hope our findings inspire designers, developers and researchers to re-consider the role of handwriting recognition in interactive systems.

ACKNOWLEDGEMENTS

We express our gratitude towards the participants. We also thank Keith Vertanen for his assistance. L.D.'s undergraduate internship, the apparatus, and compensation to participants were funded by a donation from Nokia. The following applies to P.O.K. only: The research leading to these results has received funding from the European Community's Seventh Framework Programme FP7/2007-2013 under grant agreement number 220793.

REFERENCES

1. Bailey, R.W. *Human Performance Engineering: Using Human Factors/Ergonomics to Achieve Computer System Usability*. Prentice Hall (1989).
2. Card, S.K., Moran, T.P. and Newell, A. *The Psychology of Human-Computer Interaction*. Lawrence Erlbaum Associates (1983).
3. Devoe, D.B. Alternatives to handprinting in the manual entry of data. *IEEE Transactions on Human Factors in Electronics HFE-8*, 1 (1967), 21-32.
4. Frankish, C., Hull, R. and Morgan, P. Recognition accuracy and user acceptance of pen interfaces. *Proc. CHI 1995*, ACM Press (1995), 503-510.
5. LaLomia, M.J. User acceptance of handwritten recognition accuracy. *Conference Companion CHI 1994*, ACM Press (1994), 107.
6. MacKenzie, I.S. and Chang, L. A performance comparison of two handwriting recognizers. *Interacting with Computers 11*, 3 (1999), 283-297.
7. MacKenzie, I.S. and Soukoreff, R.W. Text entry for mobile computing: models and methods, theory and practice. *Human-Computer Interaction 17* (2002), 147-198.
8. MacKenzie, I.S. and Soukoreff, R.W. Phrase sets for evaluating text entry techniques. *Ext. Abstracts CHI 2003*, ACM Press (2003), 754-755.
9. Seibel, R. Data entry devices and procedures. In Van Cott, H.P. and Kinkade, R.G. (Eds.). *Human Engineering Guide to Equipment Design*. John Wiley & Sons (1972), 311-344.
10. Zhai, S., Kristensson, P.O. and Smith, B.A. In search of effective text input interfaces for off the desktop computing. *Interacting with Computers 17*, 3 (2005), 229-250.