Dasher - a Data Entry Interface Using Continuous Gestures and Language Models

David J. Ward
Cavendish Laboratory
Madingley Road
Cambridge CB3 0HE
United Kingdom.
djw30@mrao.cam.ac.uk

Alan F. Blackwell
Computer Laboratory
Cambridge CB2 3QG
United Kingdom.
Alan.Blackwell@cl.cam.ac.uk

David J.C. MacKay
Cavendish Laboratory
Madingley Road
Cambridge CB3 0HE
United Kingdom.
mackay@mrao.cam.ac.uk

ABSTRACT

Existing devices for communicating information to computers are bulky, slow to use, or unreliable. Dasher is a new interface incorporating language modelling and driven by continuous two-dimensional gestures, *e.g.* a mouse, touchscreen, or eye-tracker. Tests have shown that this device can be used to enter text at a rate of up to 34 words per minute, compared with typical ten-finger keyboard typing of 40–60 words per minute.

Although the interface is slower than a conventional keyboard, it is small and simple, and could be used on personal data assistants and by motion–impaired computer users.

KEYWORDS: Adaptive, Text, Entry, Language, Modelling

1 INTRODUCTION

A conventional Scholes (QWERTY) keyboard is often inconvenient. The interaction device may be too small or too big [8], or may have to be operated with one hand because it is wrist-mounted or hand-held [17].

There are many competing methods of text entry for hand-held devices, but none is clearly superior. This is largely because text entry systems involve two significant tradeoffs: between potential efficiency and training time, and between device size and character-set size. Research on keyboards for conventional typing has addressed both efficiency versus training [19] and the effect of keyboard size [23]. In conventional keyboards, efficiency depends on relative key size (according to Fitts' law [13]) and on the mapping between movement over the keyboard configuration and English letter sequences. Learning speed appears to be a function of similarity to the QWERTY layout [9], which has almost ubiquitous familiarity, even in non-typists [19].

The difficulty of resolving these tradeoffs has led to predominance of the QWERTY keyboard, even in the face of improvements such as the Dvorak keyboard. However, in the case of hand-held devices no single text-entry system is dominant.

The system described in this paper uses a dynamically modified display and an adaptive language model to provide reasonably high efficiency. We have found that the system is also very easy to learn, compared to previous solutions.

2 PREVIOUS SOLUTIONS

Approaches to text entry in hand-held devices can be grouped into three broad categories; miniature and rearranged keyboards; gestural alphabets; and dynamic selection techniques.

2.1 Miniature and Rearranged Keyboards

The challenge in this approach is to present as much of the alphabet as possible while retaining sufficiently large keys. The number of keys is often reduced, and some selection mechanism is used to toggle between alternate sets. The 'half-QWERTY' device uses half of the QWERTY keyboard and the space bar toggles between the two halves. Evaluation showed typing speeds of 34.7 words per minute after 10 × 50 minute sessions [17]. The 'Fitaly' keyboard arranges keys in a compact square optimized for minimal hand travel with one-finger typing [12]. The T9 text input system uses the conventional mapping of alphabetic characters onto a telephone keypad, with a dictionary to help disambiguate alternative readings [25]. Both Fitaly and T9 are available as alternative text entry systems for the Palm Pilot. A mathematically optimised keyboard has recently been proposed in which keys are arranged on a hexagonal grid [10]. Predicted text entry speed is 43.7 wpm, but no empirical testing has been done.

Some evaluation has been done to measure the effect of using standard QWERTY keyboard layouts reduced in size on a one–finger touch screen [23]. Typing speed for experts after 30 minutes was 32.5 words per minutes with a large keyboard (246 mm wide), and dropped to 21.1 words per minute on a very small one (68 mm wide).

It is also possible to reduce keyboard size with 'chorded' schemes having fewer keys, but in which multiple keys are pressed at one time. Performance of the ternary chorded keyboard (TCK) [11] was about 16 words per minute after 600 minutes training.

2.2 Gestural Alphabets

Most hand-held pen devices are supplied with software to recognise handwritten characters, possibly using a special alphabet. The gesture sets range from those that are designed to be very efficient, to those that emphasize ease of learning. Commercial systems are generally designed to be easy to learn. Early devices attempted, with limited success, to recognize users' natural handwriting - and hence required no learning at all. Current systems – Graffiti [16] for the Palm Pilot, and Jot for Windows CE (also available on the Pilot) – improve efficiency while maintaining some resemblance to alphabetic shapes to assist learning. Researchers have also proposed more efficient alphabets such as Unistrokes, which maps simple gestures to common characters regardless of mnemonic similarity [8]. The inventors of Unistrokes estimate a rate of 40.8 words per minute, without any empirical testing [8].

2.3 Dynamic Selection

These techniques share characteristics of miniature keyboards and gestural alphabets. The user moves the pen in the direction of a selection region, which either selects a character or reveals a further set of alternatives. Examples include T-Cube [27] and Quikwriting [21]. Quikwriting is commercially available for the Pilot.

An unusual alternative approach has been to start with a standard QWERTY keyboard, but then rearrange the keys after each keystroke to place the most probable next letters near the centre. Results with the FOCL keyboard were around 10 words per minute after 10 sessions of 15 minutes [4].

These dynamic selection devices are subject to a tradeoff between efficiency and ease of learning. The arrangement of nested regions or sets of alternatives must favour the selection of common characters, while still being learnable. Dasher offers an alternative dynamic selection technique which is easier to learn (it uses simple alphabetic ordering), but improves efficiency by dynamically optimizing the selection targets according to a language model. The system can be used with arbitrary alphabets; special characters and capitalization can be included without modification.

3 THE INFORMATION CONTENT OF TEXT AND OF HAND MOVEMENTS

Conventional keyboards require the user to make one or two gestures per character (one for lower-case and two for upper-case characters). Each gesture is a selection of 1 from 80 keys, so the keyboard has the capacity to read about $\log_2(80) = 6.3$ bits per gesture. However, the entropy of English text is roughly 1 bit per character [24], so existing keyboards are inefficient by a factor of six. This inefficiency manifests itself in the fact that when typing, people make errors that are clearly not English.

The keyboard is inefficient for two reasons. First, text is usually highly redundant, yet users are forced to type most or all of the characters in a document. Second, keyboards register only contact events. Human motor capabilities dictate an upper limit for finger tapping frequency, and hence the information rate achievable from key presses. Continuous pointing gestures have the potential to convey more information through high-resolution measurement of position.

HCI research describes pointing performance in terms of Fitts' Law [13], which was originally proposed as a predictor of information rate in the human motor system. A typical empirical study providing coefficients for Fitts' law in a pointing task for data entry is that by Drury and Hoffmann [6], who find an information rate in visually-controlled pointing tasks of about 14 bits per second.

Thus in principle it should be possible to use *continuous gestures* to write English at a rate of 14 characters per second with just one finger, if only we could connect the finger to a good probabilistic model of the English language. This character rate corresponds to about 170 words per minute, which is about as fast as talking. How close we can get to this figure depends on the quality of our model of English, and on whether we can map finger gestures to text in a way that people can easily learn to use.

4 DASHER

4.1 Overall Users' View

We first describe how Dasher is used to enter the word '`the`'. Figure 1(a) shows the initial configuration, with an alphabet of 27 characters displayed alphabetically in a column. There are 26 lower case letters and the symbol '␣' represents a space. The user writes the first letter by making a gesture towards the letter's rectangle. The trails show the user moving the mouse towards the letter '`t`'.

In a continuous motion, the point of view zooms towards this letter (figure 1(b)). As the rectangles get larger possible extensions of the written string appear within the rectangle that we are moving towards. So if we are moving into the '`t`', rectangles corresponding to '`ta`', '`tb`',. . . ,'`th`',. . . ,'`tz`' appear in a vertical line like the first line. The heights of the rectangles correspond to the probabilities of these strings, given the language. In English, '`ta`' is quite probable; '`tb`' is less so; '`th`' is very probable. So it is easy to gesture our point of view into '`th`' (figure 1(c)), and from there into '`the`' (figure 1(d)).

There is an animated demonstration on the web [29].

4.2 How the Probabilistic Model Determines the Screen Layout

In a given context, we display the alphabet of possible continuations as a column of characters as shown in figure 2. The

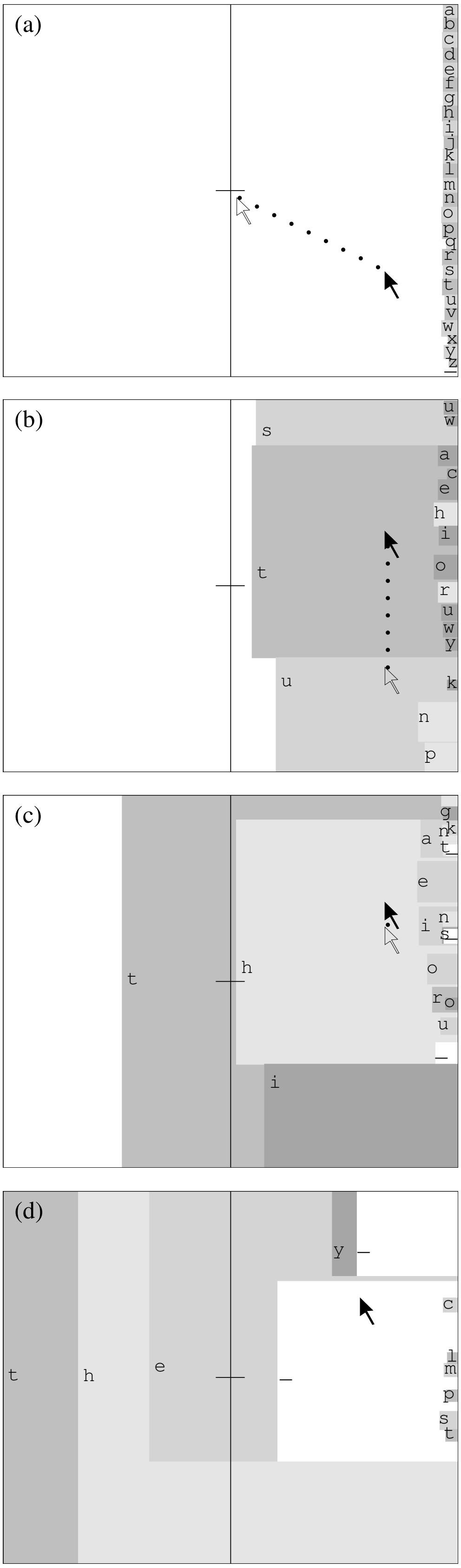

Figure 1: (a) Initial configuration. (b),(c),(d) Three successive screen shots from Dasher showing the string ‘the’ being entered. The symbol ‘_’ represents a space.

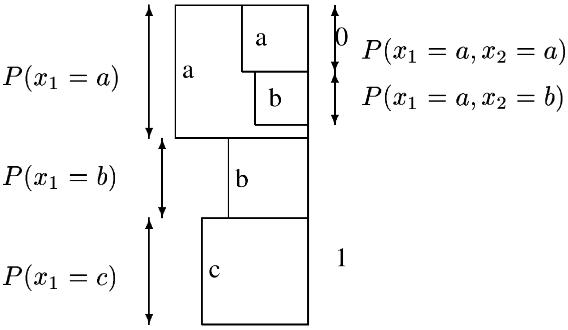

Figure 2: Arithmetic Coding

division of the right-hand vertical is analogous to arithmetic coding [2]. Let our alphabet be $A_X = \{a_1, a_2, \ldots, a_I\}$. We divide the real line [0,1) into I intervals of lengths equal to the probabilities $P(x_i = a_i)$. We subdivide the interval a_i into intervals denoted $a_i a_1, a_i a_2, a_i a_3, \ldots, a_i a_I$, such that the length of the interval $a_i a_j$ is

$$P(x_1 = a_1, x_2 = a_2) = P(x_1 = a_i) P(x_2 = a_j | x_1 = a_i) \tag{1}$$

We use a language model, described in section 4.5, to calculate the probabilities. The interval [0,1) can be divided into a sequence of intervals corresponding to all possible finite length strings $x_1 x_2 \ldots x_n$, such that the length of an interval is equal to the probability of the string given our model.

This sequence of intervals corresponds to the sequence of books in Borges’s ‘Library of Babel’ [5], with the size of each book being proportional to the probability of its contents under the language model. The user writes the book by gliding towards the alphabetically ordered books and selecting the desired book.

In figure 2, the rectangles are shown as squares, but we can choose any aspect ratio. In the version of Dasher illustrated in figure 1, the aspect ratio depends on the height of the rectangle on the screen.

4.3 Dynamics of the Interface

The two-dimensional pointer controls a continuous zooming–in of the users’ point of view. The interface zooms in so that the place under the pointer passes through the central crosshair S frame updates later (figure 3). S is the ‘Steps’ parameter.

Essentially, the left-right coordinate controls the rate of zooming–in, and the vertical coordinate determines the point on the right hand vertical that is being zoomed into. The intuitive idea is that the users should point where they want to go.

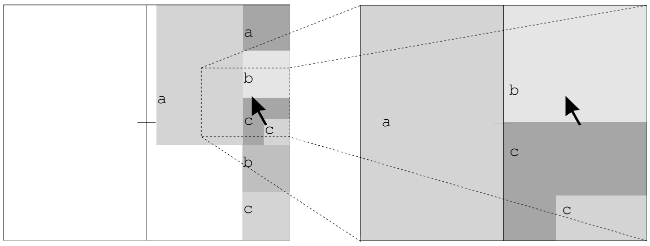

Figure 3: Dynamics of the interface, assuming the user is not moving the pointer. After S frame updates, the point of view has zoomed in so that the point under the mouse is now under the central cross-hair.

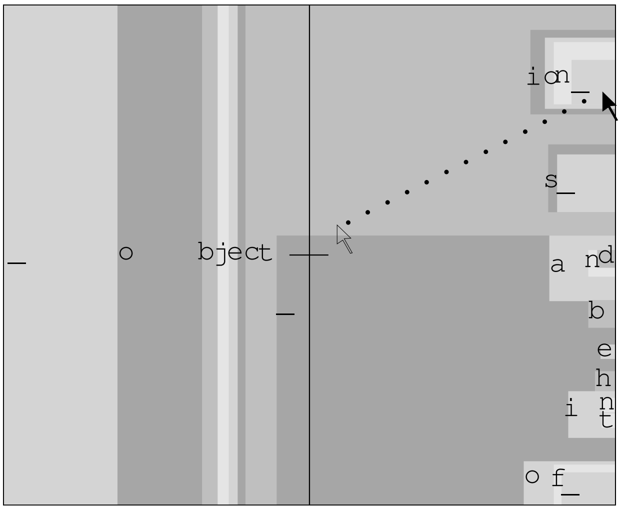

Figure 4: A single gesture can select several characters. Continuations of the string ‘object’ include ‘objection_’, ‘objects_’, ‘object_and’ and ‘object_of’.

4.4 Benefits of Continuous Gestures and Language Modelling

Dasher is driven by continuous gestures, so inaccurate gestures can be compensated for by later gestures, the way a driver keeps a car on the road. When using a conventional keyboard we select one character per gesture but in Dasher some gestures select more than one character and therefore Dasher has the potential to convey information at higher rates. Figure 4 shows completions of the string ‘object’. Grammatically correct continuations can easily be selected with a single gesture.

The language model influences the user to write coherent text so it is harder to make spelling mistakes.

4.5 The Language Model

When choosing a language model for Dasher we considered two qualities; how well the language model compresses text, and the time taken to compute the probability of a character. In Dasher, we need to calculate many probabilities each time the screen updates.

The model in the present version of Dasher is based on a popular text compression algorithm called prediction by partial match (PPM) [2]. PPM is a context-based algorithm; it uses statistics of the characters which follow identical strings. We use a variant of the algorithm called PPM5D+ [26], which can compress most English text to around 2 bits per character. There are slightly better algorithms [7] but PPM5D+ was chosen because it is simple and fast.

When using Dasher, it is difficult to select a character with a very small probability. Therefore, we modified the language model by adding a small fixed probability δ to every character and then renormalized the probabilities.

Given a context $\{c\} = x_1 x_2 \ldots x_n$, we can predict the probabilities $P(x|\{c\})$ for all symbols x. These probabilities determine the intervals in equation (1).

As the user enters text the characters can be fed back into the PPM algorithm, hence adjusting the future probabilities in accordance with that user's vocabulary.

5 EMPIRICAL EVALUATION

5.1 The Power Law of Practice

A universal observation from psychology is the power law of practice [18]. This law states that the logarithm of the reaction time τ for a particular task decreases linearly with the logarithm of the number of practice trials, N:

$$\log\left(\tau_N\right) = A - B \log\left(N\right), \qquad (2)$$

where A and B are constants. This law applies to a wide variety of behaviours; immediate-response tasks, motor-perceptual tasks, recall tests, text editing, and high-level, deliberate tasks such as chess and bridge.

It is of interest to see whether Dasher and other text entry system obey this law. We could consider N to be the number of written characters and τ_N, the time to enter a character.

The elapsed time, t, is the sum of the time of all completed tasks. This sum cannot be computed explicitly, but we can put $\tau = \frac{dt}{dN}$ and integrate equation (2) to get an alternative form of the power law:

$$\log\left[\frac{dN}{dt}\right] = C + D \log\left(t\right), \qquad (3)$$

where C and D are constants. We can measure the writing speed $\frac{dN}{dt}$ after time t and see if it follows this relationship.

5.2 Evaluation Approach

An objective of this research was to assess Dasher in a realistic text-entry task. Dasher requires uninterrupted visual attention, so it would be inappropriate to assess performance by entering text from hardcopy. Instead we used dictated text.

5.3 Pilot Experiment

The initial approach to the evaluation was to dictate text passages as synthesized speech. As the subject completed writing each word, the synthesizer would pronounce the text two words ahead of the one just entered.

This approach was tried with three experimental subjects, but they found the synthesized speech stressful and hard to follow. The results of the pilot experiment were used to estimate an appropriate length for each experimental session, to design the session format, and estimate the number of subjects and training sessions that would be required to observe significant learning.

5.4 Method

Subjects The main evaluation experiment involved ten experimental subjects, recruited from the students and staff of the Cavendish Laboratory. All subjects had vision corrected to normal, spoke English as their first language, and were free to use the mouse in their preferred hand. Subjects were paid for participating in the experiment. None of the subjects had previous experience with Dasher.

Task The experimental task, as in the pilot experiment, was to enter text dictated from Jane Austen's *Emma*, the text of which is available from Project Gutenberg [1]. Austen writes in a sophisticated but identifiable writing style. This allowed us to test Dasher in a reasonable simulation of typical usage where the language model would have been trained on previous prose written by the user. We selected 18 extracts at random from the total 883 Kbytes of text in *Emma* to form the test set. The remaining 866 Kbytes of text was used to train the language model.

Apparatus The dictated passages were recorded as a series of audio files, with each file containing a short phrase. The experimental control software had access to these audio files, and also to the test text. The text entered by the subject was monitored so that as he or she wrote the penultimate word in each phrase, the next audio file was played. This allowed subjects to enter text continuously, with dictation proceeding at a comfortable speed. A simple synchronisation algorithm compensated for misspelt words when identifying the end of a dictation phrase. Subjects could press a key with their free hand to repeat the last phrase if it was forgotten or misheard. There was no limit to the number of repeats.

Procedure Subjects completed 6 experimental sessions of approximately half an hour each. Each session consisted of three exercises. Subjects first took dictation using Dasher for five minutes. They then took dictation using the keyboard (conventional typing) for two minutes. Finally they took dictation using Dasher for another five minutes. Sessions were generally spaced at daily intervals, with no more than two on any day and no more than three days between sessions. Subjects were instructed at the start of each exercise to write as fast as possible, and were told that they could correct simple mistakes within the current word. They were told not to correct mistakes in previous words.

Configuration Dasher was configured with the following parameters:

Width of display	600 pixels
Height of display	500 pixels
Minimum threshold for plotting rectangles	4 pixels
δ	0.002
Steps parameter, S	Adaptive

The δ parameter, (defined in section 4.5), specifies a lower limit to the probability of a character.

For each exercise, the Steps parameter S (defined in section 4.3) was initialised to 60. It was automatically adapted during the test to suit the ability of the user. The timescale of adaptation was about 15 seconds.

Although Dasher has been used with a variety of position controllers, a standard mouse was used in this experiment.

Analysis The control software counted the characters entered in each period of dictation. It also counted word-level errors; a deletion, insertion, or replacement of a word (possibly with a misspelt version of the word) each counted as an error. The Viterbi algorithm [3, 28] was used to determine the minimum number of errors which when applied to the target text, produced the text entered by the user.

Data on character entry speed and proportion of errors were collected for the twelve periods of Dasher use, and for the six periods of conventional typing. We compared improvement in performance over the twelve periods of Dasher use in order to evaluate the effect of practice on Dasher performance. The interpolated conventional typing tests allowed us to estimate what proportion of this practice effect could be attributed to practice with the experimental dictation software, as opposed to practice with the Dasher text entry method.

6 RESULTS

6.1 Text-entry Speeds

All subjects successfully completed the 6 sessions within the given time constraints (figure 5). The dictation texts were used in the same order for all subjects. As a result, variation in vocabulary between the texts resulted in consistently lower speeds for some exercises. For example exercise 8 included a particularly uncommon word which caused users to hesitate. As a result, the raw data shown figures 5, 8, and 9 include apparent dips in writing speed during those exercises.

To investigate these correlations further, we plot information rate rather than writing speed (figure 6). The correlations between subjects over exercises 7,8 and 12 are no longer apparent because uncommon words have a higher information content.

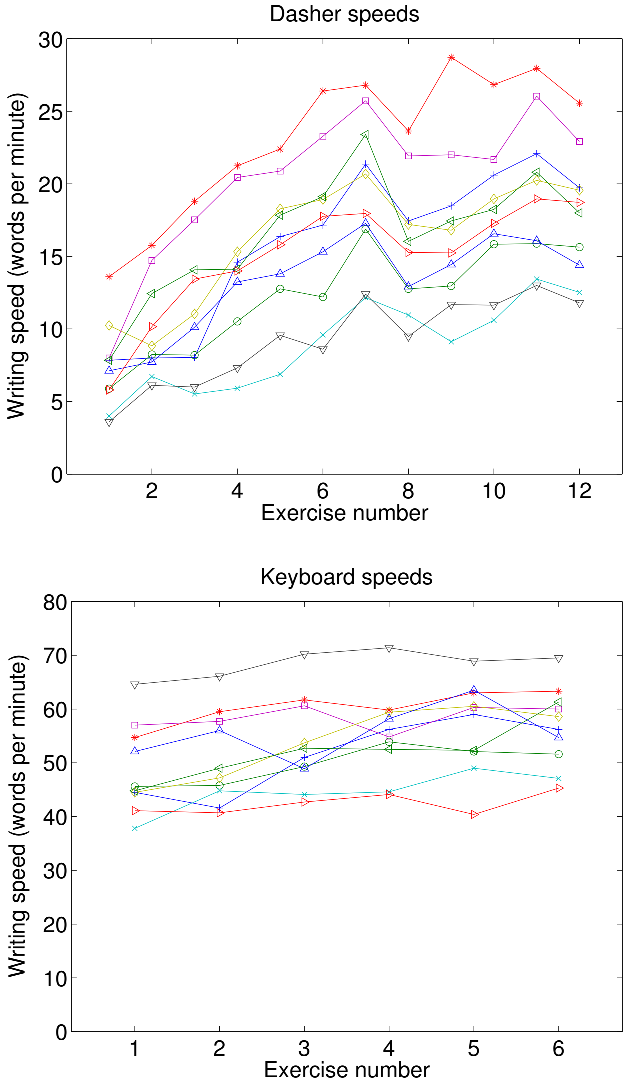

Figure 5: Effect of practice on writing speeds for ten subjects. Each exercise was 5 minutes long.

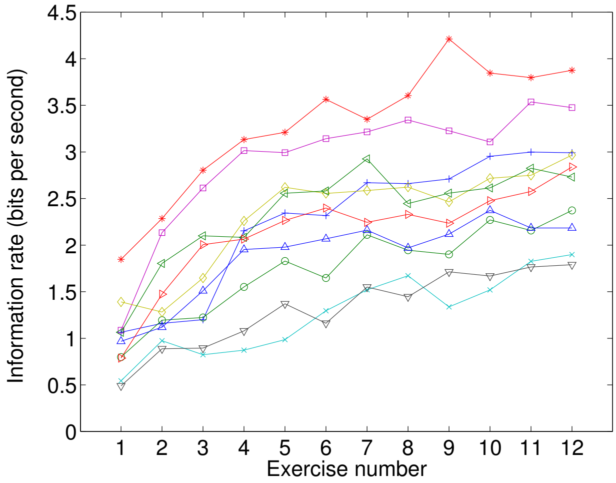

Figure 6: Dasher information rate. Each exercise was 5 minutes long.

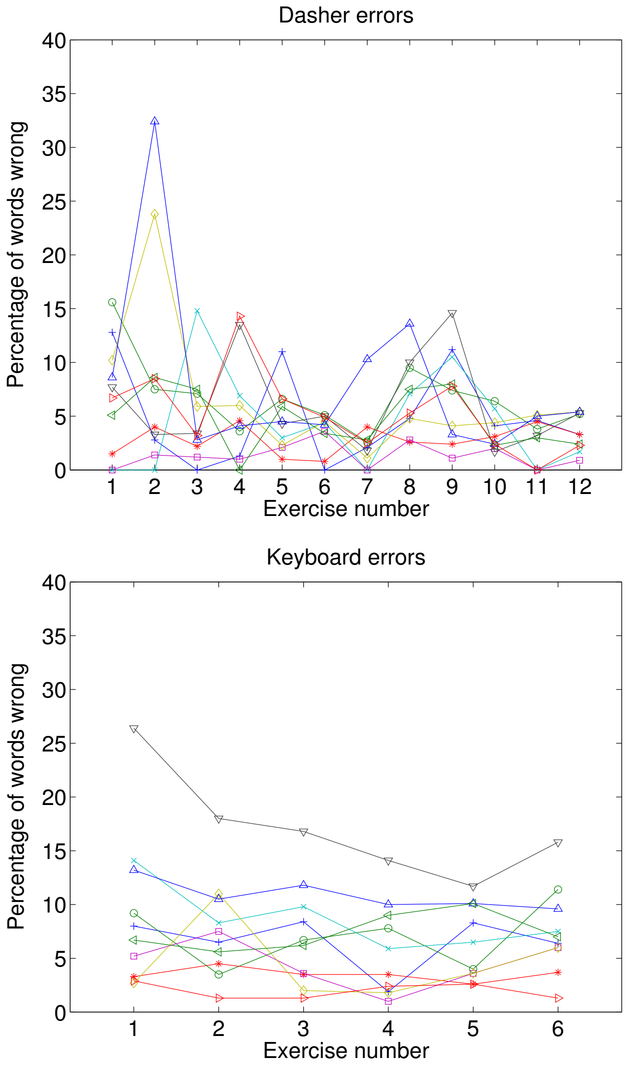

Figure 7: Writing Errors

6.2 Writing Errors

Errors were expressed as the percentage of words written differently from the test text. From the comments of subjects and from the written texts, it is clear that some errors could be attributed to the dictation system. These errors fell into the following categories:

- subjects were unsure of the uttered words, even after repeating the phrase;
- spelling errors, including common words as well as those peculiar to the text (such as proper names);
- homophones such as ‘too’ and ‘to’.

The rate of keyboard errors in figure 7 show that users took a little while to get used to the dictation system (assuming that their writing skill did not improve during the experiment). However, after the third keyboard exercise, the errors in both exercises are fairly consistent. The improvement of Dasher errors can therefore be attributed to the users becoming familiar with the dictation system as well as learning to use the

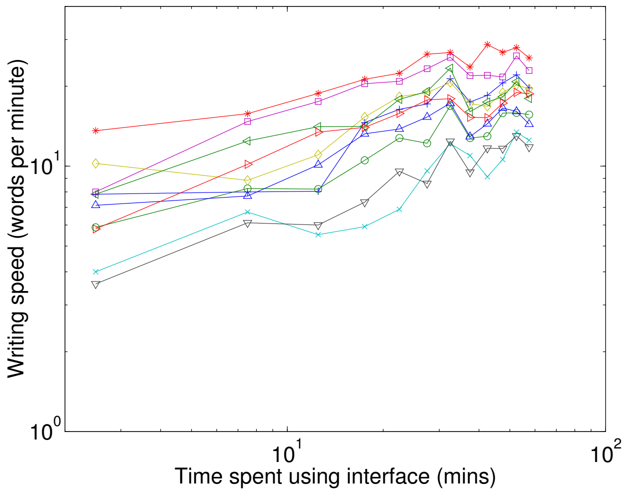

Figure 8: Logarithmic plot to test the Power Law of Practice. Dasher results are plotted for all ten subjects

interface correctly.

Towards the end of the evaluation the majority of users have an error rate of less than 5% when using Dasher, while keyboard errors vary from $2\% - 16\%$. On 10 occasions, Dasher users made no errors while keyboard users invariably made errors.

6.3 The Power Law

From figure 8 we can see that the data give a reasonably good fit to the power law (equation 3). The gradient $D = 0.32 \pm 0.05$ (related to the learning rate) is similar for all 10 subjects, but $C = 3.22 \pm 0.45$ (the log rate after one minute) varies more among subjects. The users with faster initial speeds were generally the fastest after an hour training. Consequently, we can estimate a users' future typing speed with reasonable accuracy, based on a small amount of use, say 15 minutes.

7 DISCUSSION

7.1 Comparison to Other Devices

In the introduction, we observed that novel text entry methods suffer from a tradeoff between potential efficiency and training time. This means that reporting a single text entry speed in an empirical evaluation is of limited scientific value. Where previous researchers have reported improvement of performance over time, we can compare Dasher with other devices according to the power law of practice.

We have extracted data from a number of earlier published evaluations of text entry interfaces and compare the results to Dasher (figure 9).

In a relatively short training time, writing performance with Dasher already exceeded that of many methods. If we extrapolate a power law function fit (a straight line on figure 9), it seems that with further practice, Dasher performance may also equal the fastest method using QWERTY keyboard derivatives.

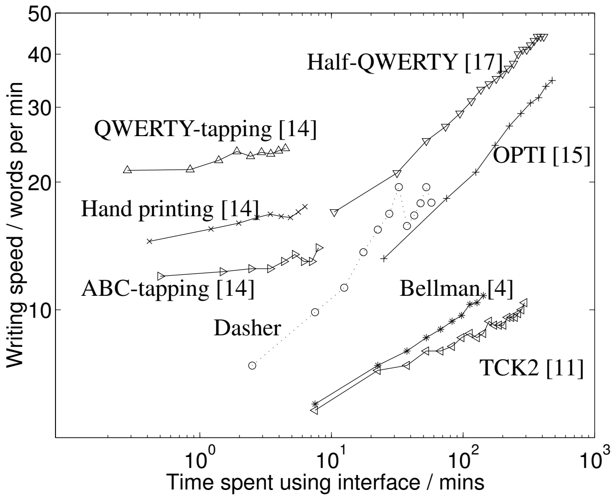

Figure 9: Comparison of Dasher to other devices. Hand printing required the user to print letters on a touchscreen. In QWERTY-tapping, a stylus selects characters from an on-screen QWERTY keyboard. ABC-tapping uses a stylus with letters in alphabetical order. Half-QWERTY uses half a QWERTY keyboard, with the other half accessible with a special key. Bellman is a keyboard designed with a probabilistic character layout strategy. TCK2 is a version of the ternary chorded keyboard. OPTI is a soft-keyboard with an optimized layout.

7.2 Attention Demand

It is clear that Dasher requires sustained visual attention from the user. This requirement is in contrast to conventional keyboards which can be used without visual attention after sufficient training. There have been attempts to produce novel text entry methods which do not require visual attention[8]. We believe that this factor is subject to a tradeoff between practice and performance.

7.3 Current Information Rate of Dasher

One of the authors has used Dasher for substantially longer than the experimental training period; a total of a few hours use, mostly during development testing rather than sustained text entry. During a number of trials with the experimental dictation text from Jane Austen's *Emma*, he achieved an average speed of 170 characters per minute (34 wpm).

The information content of the text can be described with respect to the model as the number of bits per character which are required to specify a given text. The experimental texts have an information content of 1.7 bits per character with respect to the model. Expert performance of 170 characters per minute therefore corresponds to an information rate of 4.8 bits per second.

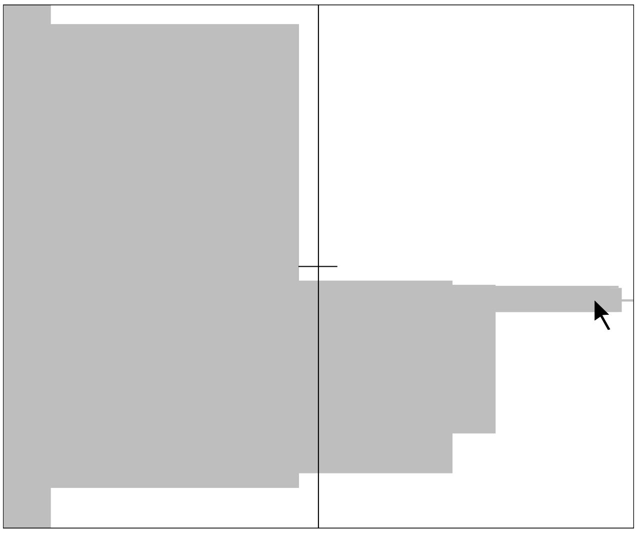

Figure 10: Modified version of Dasher to measure writing speed without visual search

7.4 Measuring Potential Information Rate of Dasher Interface

A limit in Dasher might be the time required for the user to search among the presented strings. We performed an experiment to estimate the maximum writing speed of Dasher when this visual search is not required. The required sequence of squares was highlighted in a strongly-contrasting colour (figure 10), and the 'expert' user guided Dasher along this sequence as fast as possible. The required sequence of rectangles, and their sizes, were identical to those displayed when writing the experimental text from *Emma*. The information content of the required writing path is therefore identical to that of the experimental text.

When operating Dasher as shown in figure 10, the same user was able to write at a rate of 228 characters per minute. This rate corresponds to an information rate of 6.5 bits per second. The information rate when Dasher is used in this way is 1.3 times faster than dictation speed when visual search is required so it seems that visual search is not a major limit in the present implementation.

7.5 Potential Writing Speed

The performance of Dasher is determined by two factors. The first is how well the language model can compress the text being entered. Shannon [24] estimated the entropy of English to be 1 bit per character but PPM typically compresses to 2 bits per character. A perfect language model should increase the speed of Dasher by a factor of 2 which would result in a writing speed similar to that of a QWERTY keyboard.

The second factor is the user interface, which can currently convey 4.8 bits per second to the computer. Earlier we cited an upper bound estimate of 14 bits per second for human pointing performance. Dasher uses only one dimension of movement to select letters, so perhaps we would expect a maximum rate of 7 bits per second. The following parameters are important:

- the aspect ratio of Dasher and/or the individual rectangles;
- the positioning of the characters on the screen;
- the pointing device;
- the frame-rate of Dasher (averaging 40 frames per second in the experiment);
- the rate at which we can display new letters.

The last two are dependent on the performance of the system being used for the testing. There is a tradeoff between the two quantities. If we open up new squares too eagerly then the frame-rate will be un-usably slow.

8 FURTHER WORK

8.1 Eye-tracking

An eye-tracking device is currently being tested as an alternative input device to Dasher, enabling hands-free text entry. The system could be especially beneficial to users with limited hand movement.

When Dasher is used with an eye-tracker, users simply look at the string they want to write. We hope that this will be easier and more natural to use than eye-tracking interfaces in which the users must consciously direct their gaze toward action targets, for example [22].

8.2 Modelling

The language model could be improved in several ways, for example, by the use of a dictionary to supplement the statistical model. We are exploring other models.

9 CONCLUSION

Dasher is a novel text entry interface that exploits the information redundancy of the English language, and the human capacity to convey high information rates through fine motor control.

The operation of Dasher is simple, and immediately evident to new users. Furthermore Dasher has a rapid learning rate that is comparable to alternative text entry methods.

Dasher is still under development; as yet it is unclear whether the principal benefit will be increased absolute writing speed or ease of use and learning for keyboard-less text entry.

ACKNOWLEDGEMENTS

Dasher was conceived by David MacKay and Mike Lewicki and would not have been possible without Ousterhout's Tcl language [20].

We would like to thank the experimental subjects.

This work was supported by the Gatsby foundation and by Applied Science Laboratories.

Alan Blackwell's research is funded by the Engineering and Physical Sciences Research Council, under EPSRC grant

GR/M16924 'New paradigms for visual interaction'.

REFERENCES

1. Project Gutenberg. Fine literature digitally re-published. http://www.promo.net/pg/.

2. T.C. Bell, J.G. Cleary, and I.H. Witten. *Text Compression.* Prentice Hall, NJ, 1990.

3. R. Bellman. *Dynamic Programming.* Princeton University Press, Boston, 1957, 1957.

4. T. Bellman and I.S. MacKenzie. A probabilistic character layout strategy for mobile text entry. In *Proc. Graphics Interface '98*, pages 168–176, 1998.

5. Jorge Luis Borges. *The Library of Babel.* David R. Godine, 2000.

6. C. G. Drury and E. R. Hoffmann. A model for movement time on data-entry keyboards. *Ergonomics*, 35(2):129–147, 1992.

7. G. Gilchrist. Archive Comparison Test. Available at http://www.geocities.com/SiliconValley/Park/4264/act.html.

8. D. Goldberg and C. Richardson. Touch-typing with a stylus. In *Proceedings of the INTERCHI '93 Conference on Human Factors in Computing Systems.*, pages 168–176. ACM, New York, 1993.

9. M.E. Gordon, H.G. Henry, and D.P. Massengill. Studies in typewriter keyboard modification: I. effects of amount of change, finger load, and copy content on accuracy and speed. *Journal of Applied Psychology*, 60:220–226, 1975.

10. M. Hunter, S. Zhai, and B. A. Smith. Physics-based graphical keyboard design. In *Proceedings of CHI'2000.*, volume 2, pages 157–158.

11. K.H.E. Kroemer. Performance on a prototype keyboard with ternary chorded keys. *Applied Ergonomics*, 23(2):83–90, 1992.

12. I. S. MacKenzie, S. X. Zhang, and R. W. Soukoreff. Text entry using soft keyboards. *Behaviour & Information Technology.*, 18:235–244.

13. I.S. Mackenzie. Movement time prediction in human-computer interfaces. In *Readings in Human-Computer Interaction (2nd ed.)*, pages 483–493. Kaufmann, Los Altos, CA., 1992. R. M. Baecker, W. A. S. Buxton, J. Grudin and S. Greenberg (Eds.).

14. I.S. MacKenzie, B. Nonnecke, C. McQueen, S. Riddersma, and M. Meltz. A comparison of three methods of character entry on pen-based computers. In *Human Factors and Ergonomics Society 38th Annual Meeting*, pages 330–334. Santa Monica, CA. Human Factors Society, 1994.

15. I.S. MacKenzie and S.X. Zhang. The design and evaluation of a high-performance soft keyboard. In *Proceedings of the ACM Conference on Human Factors in Computing Systems - CHI '99*, pages 25–31. New York: ACM.

16. I.S. MacKenzie and S.X. Zhang. The immediate usability of Graffiti. In *Proceedings of Graphics Interface '97*, pages 129–137. Toronto: Canadian Information Processing Society.

17. E. Matias, I.S. MacKenzie, and W. Buxton. Half-QWERTY: A one-handed keyboard facilitating skill transfer from QWERTY. In *Proceedings of the INTERCHI '93 Conference on Human Factors in Computing Systems.*, pages 88–94. ACM, New York.

18. A. Newell and P.S. Rosenbloom. Mechanisms of skill acquisition and the power law of learning. In *Cognitive Skills and Their Acquisition* , edited by J. R. Anderson, pages 1–55. L. Erlbaum Associates, NJ, 1981.

19. D.A. Norman and D. Fisher. Why alphabetic keyboards are not easy to use: Keyboard layout doesn't much matter. *Human Factors*, 24:509–519, 1982.

20. J. K. Ousterhout. *Tcl and the Tk toolkit.* Addison-Wesley, Reading, Mass., 1994.

21. K. Perlin. Quikwriting: continuous stylus-based text entry. In *ACM Symposium on User Interface Software and Technology*, pages 215–216, 1998.

22. D. D. Salcucci and J. R. Anderson. Intelligent gaze-added interface. In *CHI 2000*, pages 273–280.

23. A. Sears, D. Revis, J. Swatski, R. Crittenden, and B. Shneiderman. Investigating touchscreen typing: the effect of keyboard size on typing speed. *Behaviour and Information Technology*, 12:17–22, 1993.

24. C. E. Shannon. *Collected Papers.* IEEE Press, New York, 1993. Edited by N. J. A. Sloane and A. D. Wyner.

25. M. Silfverberg, I. S. MacKenzie, and P. Korhonen. Predicting text entry speed on mobile phones. In *proceedings CHI 2000*, pages 9–16.

26. W.J. Teahan. Probability estimation for PPM. In *Proceedings NZCSRSC'95.* Available from http://www.cs.waikato.ac.nz/~wjt/papers/NZCSRSC.ps.gz.

27. D. Venolia and F. Neiberg. T-Cube: A fast, self-disclosing pen-based alphabet. In *Proceedings CHI '94*, pages 265–270, 1994.

28. A. J. Viterbi. Error bounds for convolutional codes and an asymptotically optimum decoding algorithm. *IEEE Transactions on Information Theory*, IT-13:260–269, 1967.

29. David Ward. Non-interactive Dasher demonstration. `http://wol.ra.phy.cam.ac.uk/djw30/dasher/`.