RJ10393 (A0611-006) November 1, 2006
Computer Science

IBM Research Report

Introduction to Shape Writing

Shumin Zhai
IBM Research Division
Almaden Research Center
650 Harry Road
San Jose, CA 95120-6099

Per Ola Kristensson
Linköpings Universitet
Linköping, Sweden

Research Division
Almaden - Austin - Beijing - Haifa - India - T. J. Watson - Tokyo - Zurich

Introduction to Shape Writing

Shumin Zhai IBM Almaden Research Center, San Jose, CA, USA

Per Ola Kristensson Linköpings universitet, Linköping, Sweden

Introduction

This article introduces "shape writing" – a novel form of writing that uses pen strokes on graphical keyboards to write text. We first describe the basic concept of shape writing, followed by the human-performance considerations and rationales that guided its iterative design and development process in the past few years (Zhai & Kristensson, 2003; Kristensson & Zhai, 2004). We conclude the article by analyzing shape writing in relation to the multiple dimensions of goodness of text entry.

The Basic Concept of Shape Writing

Shape writing, also known as shorthand aided rapid keyboarding (SHARK), is a writing method designed to enable users to enter text efficiently at a faster rate than previously possible on mobile phones, handheld computers and other mobile devices. It can also be used with external tablets connected to desktop computers for those who need an effective alternative to two handed touch typing on a physical keyboard. Unlike the hand writing systems which rely on pre-defined symbols based on either widely practiced scripts (i.e. natural handwriting) or novel symbols (such as Graffiti), shape writing is defined on and used with a graphical keyboard.

The basic concept of shape writing is rather simple. A shape writing system (our current system is named ShapeWriter) displays a graphical keyboard to the user. Instead of tapping each individual letter key explicitly and precisely, the user slides the pen over all the letter keys in a word sequentially on the graphical keyboard. Figure 1 shows the ideal trace (left) and one actual and acceptable input stroke (right) for the word "the". The user's pen trace can deviate and sometimes not even cross some of the intended keys as long as it is closer to the intended word's ideal trace than any other word traces in a lexicon.

The ideal trace is called a "sokgraph" (shorthand on keyboard as a graph) (Kristensson & Zhai, 2004). In other words, a sokgraph is the continuous trace that is formed by serially connecting the center points of the keys on a graphical keyboard for a string of letters (normally a word). For shape writing, a graphical keyboard is simply an array of characters arranged either in the traditional QWERTY layout or in an optimized layout. To shape write a word, the user draws a pen stroke on the keyboard that approximates the sokgraph of the word. Once the pen stroke terminates (by lifting the pen for example), the shape writing

system in principle compares the pen stroke on the keyboard with all sokgraphs generated from a lexicon and returns the closest match. Figure 2 shows ShapeWriter in action ─ a user is shape writing the word “writing” on an optimized keyboard layout. Shape writing is inherently error tolerant, allowing noises such as hand tremors or faster and more “sloppy” writing.

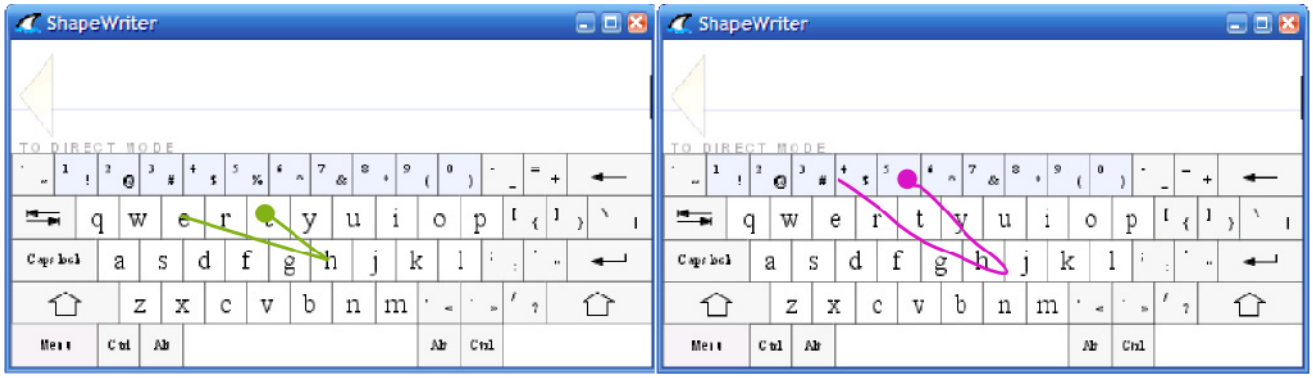


Figure 1. The word “the” as an ideal *sokgraph* (left) and a user’s actual pen input (right). The dot indicates the start of the pen-stroke.

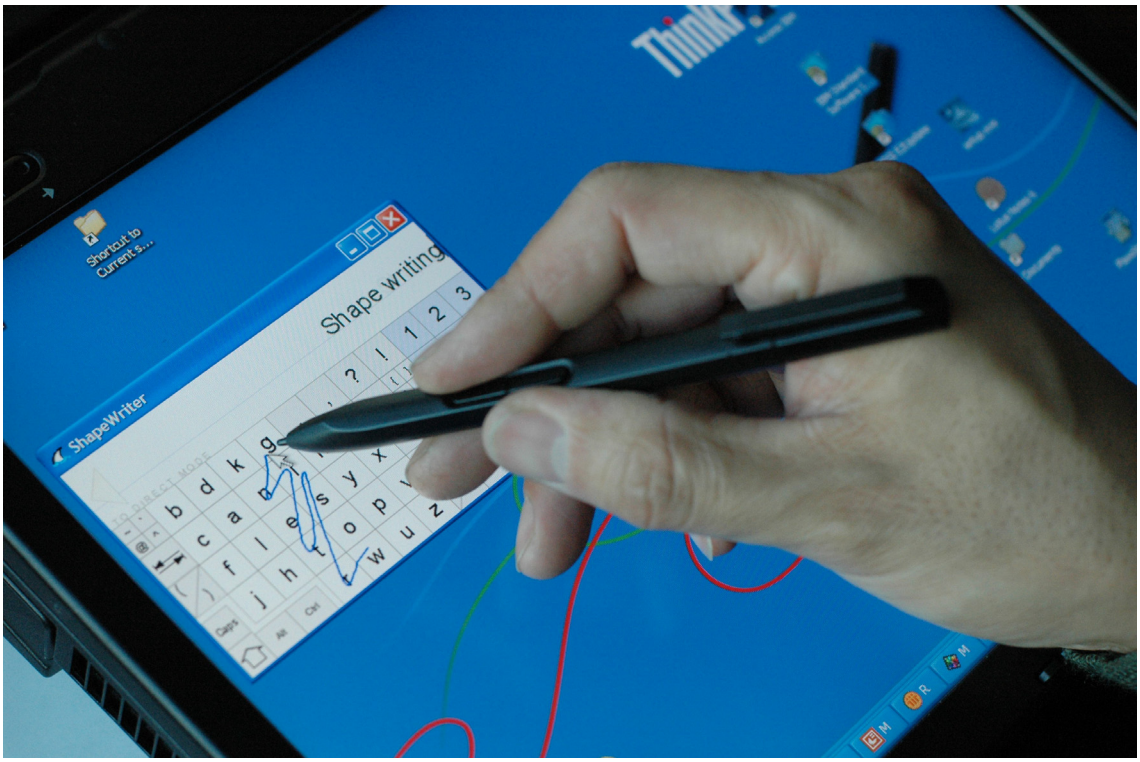


Figure 2. ShapeWriter on a Tablet PC

Information and Constraints

Text entry can be viewed as a communication system through which information is transmitted from the user to the computer. Such a system can be represented by Shannon's noisy channel model. With shape writing the writer encodes his message (a word) with geometric shapes defined on a letter map (the graphical keyboard). Such a process is "noisy" both because one often has to cross irrelevant letters in order to reach the intended letter and because the writer can be sloppy and miss some relevant letters. In other words the stroke drawn is often imperfect. The shape writing recognizer decodes the message from the code + noise and outputs the message (the word).

The reason a shape writing system can decode the noisy "shape code" fundamentally lies in the regularities and redundancies in language (and word compositions in particular). In fact in his seminal paper that laid the mathematical foundation of modern communication systems (the information theory) Shannon (1948) introduced his work by first examining the constraints and redundancy in the English language. The statistical regularities in natural language form the basis for much of today's speech recognition and other language processing technologies. The most basic way of capturing the lexical level of language regularities is the notion of a lexicon that consists of all permissible letter combinations including words, parts of words, names and acronyms. The use of a lexicon is a critical aspect of shape writing. Using lexical constraints a shape writing system allows many irrelevant letters to be crossed and still have the intended word returned. Taking Figure 2 as an example, the letters intended to be crossed were "w-r-i-t-i-n-g", but the actual pen stroke crosses "w-r-t-o-s-i-s-e-t-e-s-i-n-g.". ShapeWriter can still return the word "writing" because the lexical constraints eliminate all illegitimate letter strings. Consequently, ShapeWriter takes advantage of the regularities of word formation and recognizes the user's pen stroke on the keyboard with flexibility and error tolerance. An intended word can still be recognized although irrelevant letters between intended letters are crossed or even if some of the letters in a word are missed by the pen stroke.

To give a sense of the amount of lexical constraint existing in a natural language, such as English, consider only words with the length of 5 letters. The letter permutations of a 5 letter string would be in the range of tens of millions. This number is even greater if we can consider permutations of letter strings with different lengths. In practical writing an individual's active vocabulary is several orders of magnitude smaller. For instance, in the Enron email corpus (Klimt & Yang, 2004) the most frequent email sender wrote 8,926 emails in two years, totalling around 400,000 words. But the number of unique words in all of his emails written is only 10,858. Hence the vast majority of letter combinations can be considered as shape writing error tolerance "white space" (illegitimate letter combinations not in the lexicon).

A lexicon is only a special and limited form of language modeling for shape writing. Other types of either lower or higher order regularities, such as word level *N*-grams, can potentially also help to further relax the precision requirement of shape writing. Also, the smaller or more specific the lexicon is, the more flexibility and error tolerance shape writing can provide. This is a factor that can be taken advantage of in certain domains, for example in medical applications.

Shape Writing Recognition

Due to space limitations and the introductory nature of this article, we refer the reader to (Kristensson & Zhai, 2004) for a detailed presentation of shape writing recognition in particular and the vast literature such as (Duda, Hart, & Stork, 2001) and (Theodoridis & Koutroumbas, 1999) for pattern recognition technologies in general. Here we only highlight some specific characteristics of shape writing recognition derived from the idiosyncrasies of the shape writing paradigm.

Template-based recognition

Unlike the majority of the popular contemporary handwriting recognition methods that rely on data-training-based statistical pattern recognition (Tappert, Suen & Wakahara, 1990) we chose a template-based approach where the user's pen-stroke is compared to each word's *sokgraph* template. The template-based approach allows new words to be added to the lexicon easily. For instance, by tapping a new word on the graphical keyboard the system can instantly build the template of the added word. In contrast a training based statistical pattern recognition method may require the user to provide several sample pen-strokes of the word. A template-based approach is also more easily implemented for shape writing whose "orthography" – the canonical format of a script (Coulmas, 1989) – is clearly defined by the letter map. For natural handwriting, there is often a great deal of variation for writing each letter all within the acceptable bound.

Multi-channel recognition

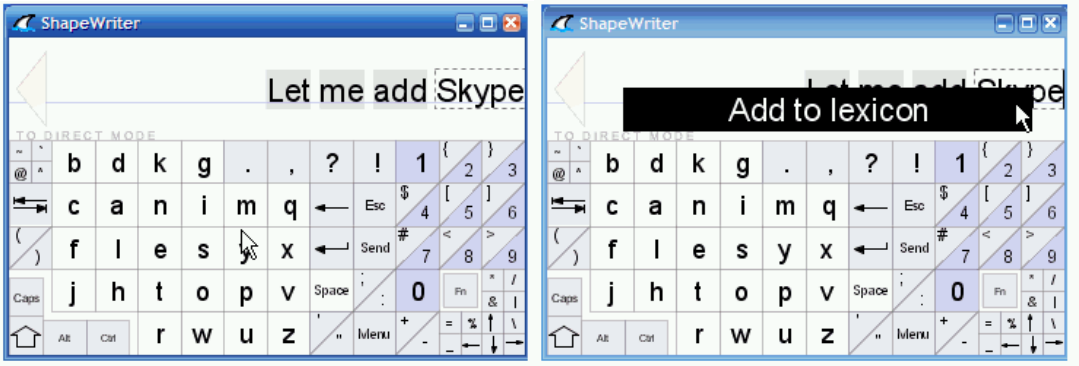
Shape writing also poses some unique challenges to its recognition. One of them is the large number of *sokgraphs* to be recognized. The two primary features we decided to include in our recognition system to address the challenge are total shape similarity between the user's pen trace and the ideal *sokgraph*, and the user's pen trace location on the graphical keyboard in relation to the location of the letter keys of the intended word. These shape and location features are treated as parallel recognition channels that are integrated using a probabilistic approach. To improve recognition accuracy several weighting schemes are used within each channel and between the channels.

Out of Lexicon Input, Ambiguity and Error Handling

Handling words outside the lexicon

What happens if the user wants to shape write a word that is not in the lexicon? No matter how large the lexicon is or how closely the lexicon matches the individual user's writing vocabulary, occasionally the user may still need to write a word, such as a rare name, an acronym, or a foreign word that is not pre-stored in the lexicon. Fortunately, shape writing is used atop of a graphical keyboard. One can use the conventional graphical keyboard input method and tap one letter at a time to enter a new word. ShapeWriter will mark the tapped unknown word with, for example, a box (Figure 3) to show the user that this is not a word in

the lexicon. The user can choose to add the word to the lexicon so it can be shape written the next time.



(a) (b)

Figure 3. A tapped word not in the lexicon is marked with a box (a); and the user can add the new word to the lexicon by clicking on it (b). Shown in pictures is the ATOMIK layout (Zhai, Hunter, & Smith, 2002) available in ShapeWriter

Ambiguity

Due to the nature of mapping words onto a graphical keyboard, it is inevitable that the *sokgraph*s of some pairs of words are almost or completely identical. Our study (Kristensson & Zhai, 2004) shows that in a lexicon containing 20,000 words, 1117 pairs of words (5.6%) are identical in normalized shapes (independent of scale and location) on the ATOMIK keyboard layout (See Figure 3). For example "root" vs. "heel", and "ben" vs. "buy". Many of these conflicts are not natural or complete English words. For example, the word "as" conflicts with "lo", "oz", "by", "ny" and "ft". If we consider only confusion pairs with the same starting and ending key positions, the number reduces to 493 (2.5%). Examples of confusion pairs with identical start and ending positions include "refuge" vs "refugee" "webb" vs "web", and "traveled" vs. "travelled". 284 of these confusions pairs are Roman numerals, such as "lxvi" vs. "lxxxvi". Excluding Roman numerals there are 209 pairs of confusion in the 20000 word lexicon (1.0%). The number of complete ambiguous word pairs on a QWERTY layout is only slightly higher (Kristensson & Zhai, 2004). Techniques for resolving ambiguity are discussed in the next section.

Error Recovery

As in all human skills, a text entry method operates on a speed accuracy trade-off curve (Figure 4). As one shape writes faster (to the left on Figure 4), she is increasingly more likely to draw a shape that is closer to an unintended word than the intended word, resulting in an entry error.

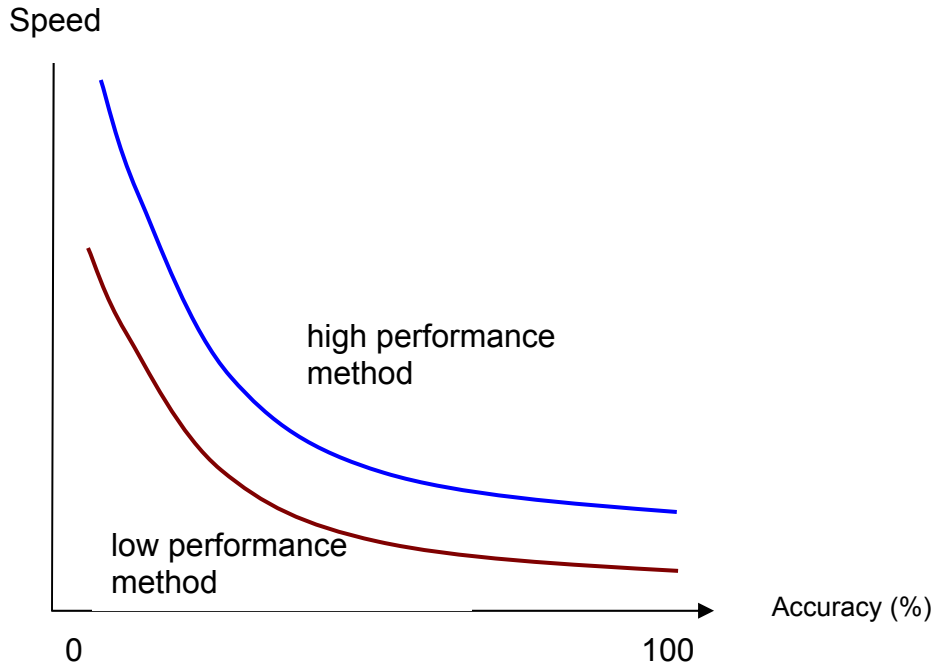


Figure 4. Hypothetical speed accuracy trade-off curves of text entry

Error recovery is a critical aspect of any text entry system design. It presents a problem that is very difficult to solve in speech recognition since it requires mode switching between dictation and command (for editing), both of which use voice. Errors occur even in the most common text entry method, two-handed touch typing on a keyboard, but there error correction is quite easy – hit the backspace key and re-type. One can easily visualize the intensity of the use of the backspace key with a keyboard monitoring program (e.g. *RSI Guard*) which displays the hit plots on each individual key. Computer users who are not rigorously trained touch typists can see that backspace is one of the most frequently hit keys.

In shape writing, the basic method of dealing with errors, due either to the small number of confusion pairs or to the deviation of the user's stroke from the intended *sokgraph* to an unintended one, is using the alternative list menu. If the user clicks on a displayed word, a correction menu is shown, in which alternative candidates are listed according to their probability of being the intended target word (Figure 5). From a certain (simplified) motor action standpoint, two pen strokes (one for shape writing one occasionally for menu selection) are still more efficient than writing a word with longhand, which on average consists of writing five letters with more than five strokes. With ShapeWriter one can also erase a word in the editor by, for example, a stroke crossing the word (Figure 6) and re-write.

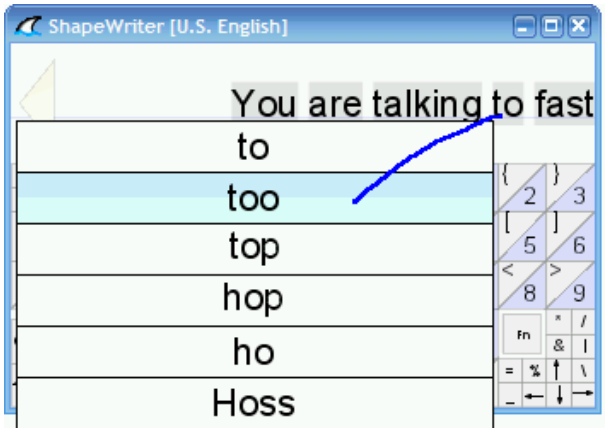


Figure 5. The user can correct an unintended word with ShapeWriter by means of the alternative list

Figure 6. The user can delete an unwanted word with a stroke crossing the word. One can also cross out multiple words with one stroke.

Human Sensitivity to Shape as an Encoding Modality and the Progression from Tracing to Direct Shape Writing

As mentioned earlier in relation to the communication model, shape writing uses geometric shapes as an encoding mechanism for words (See Figure 9 for a few examples). Humans are very good at "geometric encoding". A mundane example is using shape as a mnemonic aid to memorize pass codes. For example, it is much more difficult to remember the number sequence *1478963* than to remember the keys forming a U shape on the telephone keypad as a password.

Quantitatively, one experiment on people's ability to learn, memorize and reproduce *sokgraph*s showed that people could on average learn 10 to 15 *sokgraph*s in each hour of practice and correctly reproduce them later without seeing the keyboard (Zhai & Kristensson, 2003).

In practice, learning shape writing is expected to be more gradual and implicit, with the graphical keyboard serving as a map that guides pen movement. One does not have to memorize any *sokgraph* to begin shape writing. Instead, one traces the intended word from letter to letter by visual guidance from the graphical keyboard. The disadvantage of visual guidance based input is that it is relatively slow since entering each letter requires closing the perception-action loop. The advantage, on the other hand, is that it makes shape writing easier to begin without needing to memorize a system of symbols as in other shorthand writing systems. Ease of use, or more appropriately the low requirement of learning, made graphical user interfaces (GUI) the standard way of interacting with computers today, although it is in fact slower than the earlier command based interaction method at expert skill levels.

The transition from visual tracing to direct shape writing is modeless. After a few times of tracing a word, the user begins to remember parts or the entire shape of the *sokgraph* of the word. In the same laboratory experiment mentioned earlier (Zhai & Kristensson, 2003), in which the interval of practice of each word was gradually increased, the study participants could completely remember the shape of a *sokgraph* after 7 to 15 trials of practice. Once parts or the entire shape is remembered, the user's performance shifts towards the expert end of direct shape writing (Figure 7). As a result, the writing speed of the user increases due to two related factors: chunking (Miller, 1956; Buxton, 1986) and open-loop behaviour: chunking in the sense the user does not have to plan, search and execute the pen strokes from letter to letter on the graphical keyboard as separate units. Instead, the user plans and executes several letters or the entire word as one shape. The action is open-loop in the sense that much of the behaviour is driven by memory recall rather than visual guidance, as in all skilled and rapid performance. In our "stress tests", if a user practices and repeats a sentence until all the shapes of the words in the sentence are remembered, the shape writing speed can reach beyond 100 words per minute.

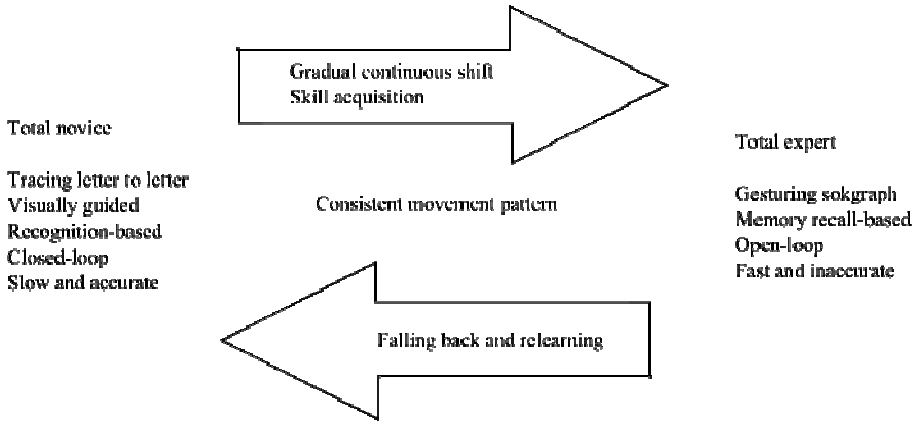


Figure 7. With practice, the user's behaviour shifts from tracing to direct shape writing.

Obviously, the longer the word, the more practice it will take to remember the entire shape of its *sokgraph*. Before that, the user is somewhere in between the two ends of shape writing expertise: complete visual tracing vs. complete direct shape writing (Figure 7). Common segments or syllables of a long word may be chunked and these parts can be directly gestured. For less common segments, the user may rely more on visual guidance of the graphical keyboard. Forgetting a *sokgraph* is also possible, which means the user has to fall back to the novice behaviour of visual tracing, which gives the user an opportunity to rehearse the shape of the *sokgraph* again. See (Fuchs, 1962) for an early observation of the progression and regression of human motor skills in general.

What bridges the novice end to the expert end of shape writing performance is that their actual movement actions are consistent. The same (consistent) movement pattern is used both in novice tracing behaviour and in expert recall-based direct shape writing. Psychology research has shown that a key to developing automaticity or less attention demanding performance in human behaviour (Schneider & Shiffrin, 1977) is consistency. The concept of facilitating novice to expert transition can also be related to Kurtenbach and Buxton's inspiring work and their compelling articulation of marking menus (Kurtenbach, Sellen, & Buxton, 1993; Kurtenbach & Buxton, 1994). They observed that the disconnection between the novice GUI menu operations and the expert hot-key-based operation prevents a natural transition from novice to expert. As an alternative solution they designed marking menus which are pie menus with a delayed pop-up of the menu items. Users make angular pen-marks to select the menu items with (in case of a novice) or without (in case of an expert) the guidance of the pie menu display.

There is an interesting difference between marking menus and shape writing (particularly in its later design) in how the novice to expert transition is designed. Via the delayed pop-up display marking menus enforce a binary transition from novice to expert behaviour. A novice pauses and waits for the delayed menu display to pop-up whereas an expert would not wait for the pop-up but produces the angular pen-marks from memory. Shape writing takes a more subtle approach – the transition is more gradual and controlled by the user rather than by the interface. With shape writing, the keyboard is always displayed. The user can choose to look at the keyboard a lot, a little, or not at all. It is conceivable that the marking menu approach "pushes" the user to jump to the expert mode sooner while the shape writing approach is less forceful, less explicit, but less rapid in helping the user to progress towards an open-loop mode. We feel the more subtle approach is necessary for shape writing due its large vocabulary size. Understanding and devising effective methods to aid the novice to expert transition is an important but under-investigated HCI research topic.

Efficiency and Layout Matters

Shape writing's high efficiency comes from several related factors. First and foremost, it is a form of speed writing in that it enables writing one word with one stroke. "Natural" handwriting is longhand, articulating one letter at a time. The weighted average length of English words is 4.7 letters (according our analysis based on the American National Corpus, http://americannationalcorpus.org/), with each letter's articulation complexity on par with a *sokgraph* of a common (often short) word. Figure 8 shows the *sokgraphs* of the words "shape writing is fun" on the ATOMIK and QWERTY layout.

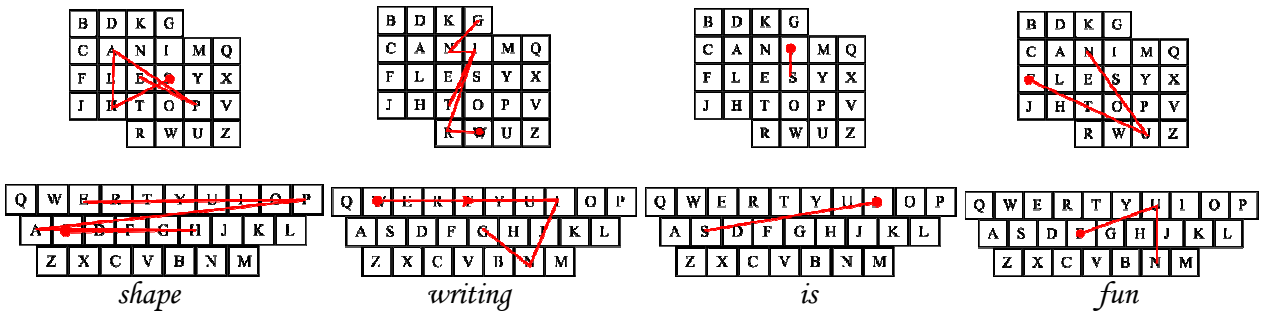


Figure 8. Some *sokgraph* examples on ATOMIK and on QWERTY

The second efficiency factor is due to the "intelligence" built into shape writing recognition. Since ShapeWriter takes advantage of the constraints of the total shape of *sokgraphs* it allows the user to deviate from the precise trajectory. The user can be as "sloppy" as the white space allows, enabling the user to trade a certain amount of accuracy for speed.

The third efficiency factor comes from layout optimization. Shape writing is in principle independent of layout, and indeed can use any number of layouts. The subjectively familiar layout is the QWERTY layout invented in the 1860's for minimizing mechanical jamming (Yamada, 1980). However, touch typing uses motor memory instead of visual spatial memory so the learning advantage of QWERTY as a map for shape writing is not as great as one might think. The disadvantage of using QWERTY for shape writing is the loss of potential efficiency and greater difficulty in long term learning. We have experimented with an class of optimized layout called ATOMIK (alphabetically tuned and optimized mobile interface keyboard (Zhai & Smith, 2001; Zhai, Hunter, & Smith, 2002)). We started with the classic ATOMIK optimized for tapping (Figure 9) as a more efficient shape writing map. *Sokgraph*s defined on ATOMIK are both more efficient and easier to remember because the *sokgraph*s are more diverse than those defined on QWERTY which tend to zig-zag in a similar fashion (See Figure 8) for many words due to the fact that common consecutive letters tend to be arranged on the opposite sides of the QWERTY layout (See Yamada 1980 for an thorough review of QWERTY design). Two mechanisms in the ATOMIK layout help a novice user find the letters. First, due to the optimization effect the next letter needed is more likely to be in the vicinity of the current pen position. Second, letters from A to Z tend to run from the upper left to lower right corner of the ATOMIK layout, giving the novice user another clue for finding a particular letter (Figure 9 and 10) (Smith & Zhai, 2001) .

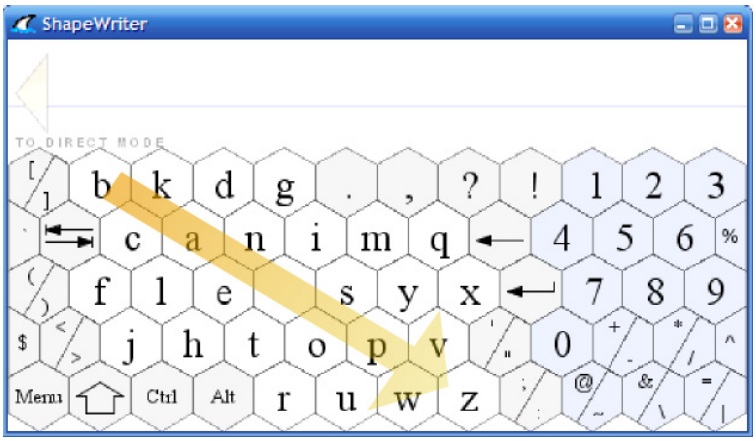


Figure 9. Classic ATOMIK layout with ABC to Z order tuning illustration

Figure 10. The ATOMIK layout revised for shape writing with ABC to Z order tuning illustration.

There are a few aspects of ATOMIK that were improved specifically for shape writing, resulting in a revised ATOMIK layout (Figure 10). As shown in Figure 9 and 10, D and K keys are swapped so that the common suffix ED is more distant from the common suffix ING. Hexagon keys are changed to the simpler and cleaner squared keys. In the classic ATOMIK layout (Figure 9) keys were hexagonal to maximize tapping efficiency, which is no longer needed for shape writing. Similarly, the space key is moved off the center since spaces are automatically filled after every *sokgraph* is recognized. The last row is slightly shifted to the left to make the layout more compact and some *sokgraph*s less stretched.

Keyboard optimization research (Getschow, Rosen, & Goodenough-Trepagnier, 1986; Lewis, Kennedy, & LaLomia, 1992; MacKenzie & Zhang, 1999; Zhai, Hunter, & Smith, 2000; Smith & Zhai, 2001) has focused on the core Roman letter set. Auxiliary keys in fact also matter to the overall efficiency and usability of the keyboard. The main rationale for the auxiliary keys in the revised ATOMIK in Figure 10 is as follows: $ is close to the number

keys, since it is usually followed by digits. % is below the number keys, since it is usually placed after digits. Similar pairs such as "{ }" and "[ ]" are aligned and grouped together. Similar keys \ | / are aligned and grouped. Arrow keys are set according to convention (location and shape). The Caps, SHIFT, Ctrl, Alt, and Fn keys cause a state change as soon as a stylus lands on them, even if the user is in the middle of drawing a stroke and lands on one accidentally. To minimize unintentional activation of these keys, they are made slightly smaller with a gap between them and the adjacent keys. The smaller Fn and Ctrl keys also make the user more careful and deliberate to trigger "command strokes" ─ *sokgraph*s starting from a command key so they will be interpreted as a command. For example, Ctrl-C-O-P-Y issues a Copy command rather than entering the word "copy". The Fn key is placed as close to the numeric keys as possible, so that function shortcuts such as F1 and F3 are more separated. Ctrl is placed more to the center (swapped with Alt) for a similar reason.

The Multiple Dimensions and Guidelines of Efficient Text Entry

Developing effective text entry methods is a challenging task, partly because the goodness of the solution is multidimensional. In our view, there are at least the following dimensions that can be desirable in a text entry method: High performance (high product of speed and accuracy); Ease of entry / ease of learning; Low effort; Mobility; and Fun. In this section we discuss shape writing along these dimensions of goodness as design guidelines.

To achieve perfection in all of these dimensions in one method is an impossible task, since some of them can be contradictory in design choices. The relative importance of each of these dimensions depends on application requirements and individual preference. Our primary goal of developing ShapeWriter is to enable mobile devices to take on common personal computing tasks efficiently (e.g. email, text messaging, note taking, etc), which dictated the trade offs we made, described as follows.

High performance

High performance should not be measured by high speed only, but rather by a high speed accuracy trade off curve (Figure 4). High performance is one of our primary goals in developing shape writing. This was achieved due to the shorthand nature, the built-in "intelligence" and error tolerance, and the layout optimization. The relative ease of error correction should also contribute to average performance.

Ease of entry and ease of learning

This dimension is typically at odds with the first. For example character-based longhand writing is easy to begin for any literate user, but it is not high performance. On the other hand, traditional shorthand writing (such as Pitman or Gregg's system) can give very high speed, but they require the user to memorize a large number of symbols to begin, making its learning requirement impractical for ordinary computing and communication use. One of the core ideas in shape writing is to bridge ease of entry with high performance. One can start with no memory at all (tracing) and gradually move to (partially and increasingly) memory recall-based high performance gesturing. Interestingly, the standard desktop text entry method, two-

handed touch typing, can afford quite high performance but its learning requirement would be hardly acceptable if it were a new computer user interface that had not been widely adopted. Learning shape writing is much easier than learning touch typing.

Effortless

A good text entry solution should require as little effort as necessary. Effort in fact can be further divided along multiple sub-dimensions of human performance: motor, perceptual, and cognitive. A method can optimize for one of the sub-dimensions at the cost of another. For example, by dynamically positioning letter keys according to context as in Dasher (Ward, Blackwell, & MacKay, 2000), the amount of stylus moment for text entry can be minimized. The cost of dynamically arranging letters, however, is the greater perceptual effort needed to constantly monitoring the changing letters. Such a trade-off of motor effort with perceptual effort could well be justified for some applications, for example, interfaces for users with certain types of motor impairments. In contrast, methods like Unistrokes (Goldberg & Richardson, 1993) and Graffiti can be potentially eyes-free but they require much more motor effort since they are fundamentally longhand methods (character level writing). Shape writing balances the motor and perceptual effort with a stationary layout augmented with fluid pen strokes. In our early implementation of shape writing, named SHARK at the time (Zhai & Kristensson, 2003), an important goal was making shape writing scale and location independent so an expert user did not have to look at the keyboard at all if the *sokgraph* is remembered completely. This was achieved by limiting shape writing only to a small set of common words. For all other words, the user was expected to tap one letter at a time. The Zipf's law effect was expected to skew the impact of the small number of common words and therefore speed up the average writing time. A cost to this hybrid approach, we later realized, could be cognitive: the user has to decide between shape writing the whole word or tapping individual letters. We hence decided to expand shape writing to all words needed (Kristensson & Zhai, 2004), although it means that a certain degree of visual monitoring of the strokes on the keyboard is needed since scale and location independence cannot be guaranteed for such a large number of words. This perceptual cost is much lower than that of conventional stylus tapping requiring high precision taps within each individual key. Overall, the design of ShapeWriter minimizes the user's motor effort without demanding an unacceptable level of perceptual and cognitive effort.

Mobility

Today's research interest in developing new text entry methods is largely driven by the need to carry out computing and communication functions on small, handheld mobile devices. A small and flexible form factor, with a small footprint and an near zero start up time (including the time of setting-up the device in case of an external attachment solution) is desirable. An extreme case is one handed (or no hand at all) input while walking or driving (which is probably unsafe in any case). Speech offers a unique advantage in this regard and is making inroads into these extreme mobile conditions such as command systems in cars, but suffers other problems such as error correction difficulty for large vocabularies and cognitive load if used as a dictation method (Karat, Halverson, Horn, & Karat, 1999). Shape writing requires one hand (if the device is rested on a surface) or both hands (with one holding the device). The size of ShapeWriter is scalable. How human performance of shape writing precisely

changes with the keyboard size requires future research. Previous research on path steering performance shows that the best motor control performance is achieved at a scale in which finger joints and hand wrist carry out the motion (Accot & Zhai, 2001).

Fun and aesthetics

Ultimately, users' subjective experience of using a text entry method dictates the market demand. Not all the factors that make an interface fun and enjoyable are well understood today. All of the previous four dimensions (high performance, ease of learning, low effort, and mobile form factor) can contribute to a fun shape writing experience. In addition, simple and clean visual design, effective feedback, and fluid stroking actions may also contribute to a fun experience. More explicitly, we have also developed a practice game embedded in ShapeWriter so the user can master shape writing skills in a playful mode (Figure 11). In this game, balloons carrying different words float upward and are popped when the user enters the correct *sokgraph* for each word. The words that appear in the game are driven by an expanding rehearsal interval (Landauer & Bjork, 1978; Zhai & Kristensson, 2003) so the practice impact is optimized. There is also an auto-play mode in the game in which ShapeWriter draws the *sokgraph*s automatically so the user can learn by watching. The benefit of "observational practice" has been shown in human motor control research (Kohl & Shea, 1992) .

Figure 11. An embedded ShapeWriter game based on memory research

Since the many dimensions of an efficient text entry interface sometimes conflict with one another there will be probably never exist a single text entry method that addresses all needs in all situations for all users. Many of the recently developed methods, such as EdgeWrite (Wobbrock, Myers, & Kembel, 2003), offer their unique strengths hence may find particular applications. Shape writing is an evolving novel technology and we are conducting more empirical and analytical studies which may give us more insights for further improvement. Clearly, we have made various design trade-offs in order to address as much and as many important dimensions of goodness as possible. In its current form ShapeWriter is already practical enough for writing daily email and parts of this article (when a tablet is available), but much more can be done to further improve it. Obviously shape writing also needs to be adapted to other languages than English. This is relatively easy for alphabetical languages. For non-alphabetical languages such as Chinese, many creative steps are needed since Chinese characters[1] are not easily parsed into regular elements (See Chapter 6 of Coulmas 1989 for an introduction of Chinese writing). An indirect approach is to use *pinyin*, the Roman-letter-based phonetic equivalent of Chinese characters, which is how Chinese is entered into computers today by most users in mainland China. Unfortunately there are numerous homophonic characters corresponding to each phonetic spelling. This is solved typically by displaying multiple choices for the user to choose, which demands additional visual attention and slows down the input speed due to the perceptual and cognitive burden on the user (Wang, Zhai, & Su, 2001).

Finally, some questions as food for thought. First, what can be the highest possible performance in articulating words through shape writing as we have developed it or through some variant form of it in the future? There is no straightforward theoretical estimate on this due to the lack of an empirical law on stroke gesturing that can be used in the way Fitts' law is used to estimate tapping on graphical keyboard (Lewis, 1992; Soukoreff & MacKenzie, 1995; Zhai, Sue, & Accot, 2002). As a reference point 150 to 160 words per minute is the recommended rate for easily understandable speech, such as recording books on tape (Williams, 1998). Speech is also a form of muscle articulation although not by hand. Can shape writing be improved to the point that it is as fast as this level of speech? Another interesting and difficult question is how a text entry technology or some of its components may or may not be adopted. In economics there has been a lively debate between David (1985; 1998-2000) vs. Liebowitz and Margolis (1990; 1996) regarding "Qwertynomics" – the path dependence theory that a suboptimal rather than the optimal technology can be locked into the society. The various writing systems of the world, as a very special type of technology in different civilizations, in fact also underwent complicated creation (often by borrowing from other cultures), evolution or extinction processes that are shaped by economical (efficiency), compatibility with previous practice, and socio-political (such as nationalism) factors (Coulmas, 1989). Many interesting and important lessons can be learned there in order to influence the future of digital mobile society with creative technology solutions.

---

[1] Unlike Roman characters, Chinese characters, or *zi*, are at a higher level than alphabet but lower than words. The closest counterpart to a Chinese zi in English is a syllable although most Chinese characters have more defined semantics than English syllables.

Further Readings

Zhai and Kristensson (2003) presents the initial motivation, implementation, and experimentation of shape writing on graphical keyboards. Kristensson and Zhai (2004) describes the rationale and implementation of a large scale shape writing system. Zhai, Kristensson and Smith (2005) gives an overview of both shape writing and ATOMIK, the preceding project that led to the creation and development of shape writing.

References

Accot, J., & Zhai, S. (2001). Scale effects in steering law tasks. *Proc. CHI 2001: ACM Conference on Human Factors in Computing Systems, CHI Letters 3(1)*, 1-8.

Buxton, W. (1986). Chunking and phrasing and the design of human-computer dislogues. *Proc. IFIP World Computer Congress*, 475-480.

Coulmas, F. (1989). *The writing systems of the world*. Oxford: Blackwell.

David, P. A. (1985). Clio and the Economics of QWERTY. *American Economic Review, 75*, 332-337.

David, P. A. (1998-2000). *Path Dependence, its critics, and the quest for 'historical economics'* (No. JEL-codes: A1 B0 C4 D9 N0 O3). Stanford: Stanford University.

Duda, R. O., Hart, P. E., & Stork, D. G. (2001). *Pattern Classification*. New York: John Wiley & Sons.

Fuchs, A. H. (1962). The progression-regression hypothesis in perceptual-motor skill learning. *Journal of Experimental Psychology, 29*, 39-53.

Getschow, C. O., Rosen, M. J., & Goodenough-Trepagnier, C. (1986). A systematic approach to design a minimum distance alphabetical keyboard. *Proc. RESNA (Rehabilitation Engineering Society of North America) 9th Annual Conference*, 396-398.

Goldberg, D., & Richardson, C. (1993). Touching-typing with a stylus. *Proc. INTERCHI, ACM Conference on Human Factors in Computing Systems*, 80-87.

Karat, C.-M., Halverson, C., Horn, D., & Karat, J. (1999). Patterns of entry and correction in large vocabulary continuous speech recognition systems. *Proc. CHI'99: ACM Conference on Human Factors in Computing Systems*, 568-574.

Klimt, B., & Yang, Y. (2004). Introducing the enron corpus. *Proc. Conference on Email and Anti-Spam (CEAS)*.

Kohl, R. M., & Shea, C. H. (1992). Pew(1966) revisited: Acquisition of hierarchical control as a function of observational practice. *Journal of Motor Behavior, 24*(3), 247-260.

Kristensson, P.-O., & Zhai, S. (2004). SHARK2: A Large Vocabulary Shorthand Writing System for Pen-based Computers. *Proc. ACM Symposium on User Interface Software and Technology*, 43 - 52.

Kurtenbach, G., & Buxton, W. (1994). User Learning and Performance with Marking Menus. *Proc. CHI: ACM Conference on Human Factors in Computing Systems*, 258-264.

Kurtenbach, G., Sellen, A., & Buxton, W. (1993). An empirical evaluation of some articulatory and cognitive aspects of "marking menus". *Human Computer Interaction, 8*(1), 1-23.

Landauer, T. K., & Bjork, R. A. (1978). Optimum rehearsal patterns and name learning. In M. M. Gruneberg, P. E. Morris & R. N. Sykes (Eds.), *Practical Aspects of Memory* (pp. 625-632). London: Academic Press.

Lewis, J. R. (1992). *Typing-key layouts for single-finger or stylus input: initial user preference and performance* (Technical Report No. 54729). Boca Raton, FL: International Business Machines Corporation.

Lewis, J. R., Kennedy, P. J., & LaLomia, M. J. (1992). *Improved typing-key layouts for single-finger or stylus input* (Technical Report No. TR 54.692): IBM Technical Report TR 54.692.

Liebowitz, S., & Margolis, S. E. (1996). Typing Errors. *Reason Magazine* http://reason.com/9606/Fe.QWERTY.shtml.

Liebowitz, S. J., & Margolis, S. E. (1990). The Fable of the Keys. *Journal of Law and Economics, XXXIII*.

MacKenzie, I. S., & Zhang, S. X. (1999). The design and evaluation of a high-performance soft keyboard. *Proc. CHI'99: ACM Conference on Human Factors in Computing Systems*, 25-31.

Miller, G. A. (1956). The Magical Number Seven, Plus or Minus Two: Some Limits on Our Capacity for Processing Information. *The Psychological Review, 63*, 81-97.

Schneider, W., & Shiffrin, R. M. (1977). Controlled and automatic human information processing: I. Detection, search, and attention. *Psychological Review, 84*(1), 1-66.

Shannon, C. E. (1948). A mathematical theory of communication. *The Bell System Technical Journal, 27*, 379-423, 623-656.

Smith, B. A., & Zhai, S. (2001). Optimised Virtual Keyboards with and without Alphabetical Ordering - A Novice User Study. *Proc. INTERACT'2001 - IFIP International Conference on Human-Computer Interaction*, 92-99.

Soukoreff, W., & MacKenzie, I. S. (1995). Theoretical upper and lower bounds on typing speeds using a stylus and keyboard,. *Behaviour & Information Technology, 14*, 379-379.

Tappert, C. C., Suen, C. Y., & Wakahara, T. (1990). The State of the Art in On-Line Handwriting Recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 12*(8).

Theodoridis, K., & Koutroumbas, K. (1999). *Pattern Recognition*. San Diego: Academic Press.

Wang, J., Zhai, S., & Su, H. (2001). Chinese Input with Keyboard and Eye Tracking - An Anatomical Study. *Proc. of CHI 2001 - ACM Conference on Human Factors in Computing Systems*, 349-356.

Ward, D., Blackwell, A., & MacKay, D. (2000). Dasher - A data entry interface using continuous gesture and language models. *Proc. The 13th ACM Symposium on User Interface Software and Technology (UIST)*, 129-136.

Williams, J. R. (1998). Guidelines for the use of multimedia in instruction, 1447-1451.

Wobbrock, J. O., Myers, B. A., & Kembel, J. A. (2003). EdgeWrite: a stylus-based text entry method designed for high accuracy and stability of motion. *Proc. ACM symposium on user interface software and technology*, 61-70.

Zhai, S., Hunter, M., & Smith, B. A. (2000). The Metropolis Keyboard - an exploration of quantitative techniques for virtual keyboard design. *Proc. The 13th Annual ACM Symposium on User Interface Software and Technology (UIST)*, 119-218.

Zhai, S., Hunter, M., & Smith, B. A. (2002). Performance optimization of virtual keyboards. *Human-Computer Interaction, 17*(2,3), 89-129.

Zhai, S., & Kristensson, P.-O. (2003). Shorthand Writing on Stylus Keyboard. *Proc. CHI 2003, ACM Conference on Human Factors in Computing Systems, CHI Letters 5(1)*, 97-104.

Zhai, S., & Smith, B. A. (2001). Alphabetically biased virtual keyboards are easier to use - layout does matter. *Proc. CHI 2001: ACM Conference on Human Factors in Computing Systems*, 321-322.

Zhai, S., Sue, A., & Accot, J. (2002). Movement model, hits distribution and learning in virtual Keyboarding. *Proc. CHI 2002: ACM Conference on Human Factors in Computing Systems, CHI Letters 4(1)*, 17-24.