Context Switching Eye Typing Using Dynamic Expanding Targets

Carlos H. Morimoto
Department of Computer Science
University of São Paulo
hitoshi@ime.usp.br

Jose A. T. Leyva
Department of Computer Science
University of São Paulo
jtula@ime.usp.br

Antonio Diaz-Tula
Department of Computer Science
University of São Paulo
diaztula@ime.usp.br

Figure 1: Dense single line keyboard layout with dynamic expanding targets. Keys are selected by dwelling.

ABSTRACT

Text entry by gazing on a virtual keyboard (also known as eye typing) is an important component of any gaze communication system. One of the main challenges for efficient communication is how to avoid unintended key selections due to the Midas' touch problem. The most common selection technique by gaze is dwelling. Though easy to learn, long dwell-times slows down the communication, and short dwells are prone to error. Context switching (CS) is a faster and more comfortable alternative, but the duplication of contexts takes a lot of screen space. In this paper we introduce two new CS designs using dynamic expanding targets that are more appropriate when a reduced interaction window is required. We compare the performance of the two new designs with the original CS design using QWERTY layouts as contexts. Our results with 6 participants typing with the 3 keyboards show that the use of smaller size layouts with dynamic expanding targets are as accurate and comfortable as the larger QWERTY layout, though providing lower typing speeds.

CCS CONCEPTS

• **Human-centered computing** → **Empirical studies in interaction design**; **Accessibility systems and tools**;

KEYWORDS

Gaze interaction, eye tracking, eyetyping, context switching, expanding targets.

COGAIN '18, June 14–17, 2018, Warsaw, Poland

ACM ISBN 978-1-4503-5790-6/18/06…$15.00
https://doi.org/10.1145/3206343.3206347

ACM Reference Format:
Carlos H. Morimoto, Jose A. T. Leyva, and Antonio Diaz-Tula. 2018. Context Switching Eye Typing Using Dynamic Expanding Targets. In *COGAIN '18: Workshop on Communication by Gaze Interaction, June 14–17, 2018, Warsaw, Poland*. ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3206343.3206347

1 INTRODUCTION

Text entry by gaze (eye typing) on a virtual keyboard is an important tool for people with motor disabilities to communicate and several different methods have been proposed in the literature. The usability of an eye typing method basically depends on the method used to select the key. Though most methods in the literature focus on typing speed, in this paper we consider other dimensions such as comfort, learnability, size, robustness to avoid unintended selections, and overall user experience.

Dwell-time [Majaranta et al. 2009] is probably the simplest eye typing method to learn if the user is already accustomed to a QWERTY keyboard, since it only requires the user to look at the desired key and wait (dwell) for it to be selected. Though simple to learn, a short dwell-time increases the number of typing errors (due to the Midas' touch problem [Jacob 1990]), and a long dwell-time slows down the communication rate.

Besides dwelling, gaze gestures have been suggested for eye typing to avoid the Midas' touch problem. A discrete gaze gesture is defined as a sequence of eye fixations. Examples of discrete gaze gesture interfaces are Eye-S [Porta and Turina 2008], EyeWrite [Wobbrock et al. 2008], Pie-menus [Huckauf and Urbina 2008], and [Chakraborty et al. 2014].

[Porta and Turina 2008] have suggested the use of "graffiti" style gestures for gaze communication. A graffiti like gesture were performed on a 3×3 grid of hotspots. A similar idea was suggested by [Wobbrock et al. 2008], that proposed graffiti like gestures using the four corners of a squared window as hotspots. Though such gaze gestures can be performed on a relatively small virtual window, learning the gesture for each character requires significant effort.

To improve learnability, the interface should provide visual information and feedback of each gesture. Therefore, it is not surprising

that the most common typing interface shows a virtual keyboard with the QWERTY layout. In-out-in gestures where the user focus on a key and quickly looks outside and then back to select it have suggested by [Chakraborty et al. 2014]. Another idea borrowed from typing in mobile devices is the use of swiping gestures for gaze interaction, as suggested by [Kristensson and Vertanen 2012]. To type a word using a swipe gesture the user should look at all the letters of the word in the correct sequence. One of the main challenges for using swipe gaze gestures is the identification of the beginning and end of the swipe. On a mobile device, the gesture starts by touching the first letter and lifting the finger once the swipe reaches the last letter. For eye-typing, [Pedrosa et al. 2015] used short dwell-times for each key and a filtering strategy based on a language model to filter the most probable word, while [Kurauchi et al. 2016] proposed the use of reverse-crossing gestures to mark the start and end of the swipe. Reverse-crossing is similar to an in-out-in gesture but shows explicit visual feedback.

The virtual keyboard is not required to follow the QWERTY layout. For example, pie menus have been suggested by [Huckauf and Urbina 2008]. Groups of letters of the alphabet are arranged within each pie slice. When the user fixates at one of the slices, the items in that slice are displayed on another pie, hierarchically, until a single item is contained in one slice to be selected. The use of pie menus saves screen space when compared to a full QWERTY keyboard, but depending on the number of hierarchical levels, typing can be slow.

Continuous gaze gestures, such as Dasher [Ward et al. 2000] and Stargaze [Hansen et al. 2008], allows the user to type by just following the intended letter as they move along the display. Stargaze has been developed to increase robustness to noise during typing, and Dasher has shown that the use of a language model for word prediction can significantly improve the eye typing performance. They are examples of zooming interfaces that require the user's continuous attention to navigate through the interface. Language models have been used in gaze swipe gesture interfaces by [Kurauchi et al. 2016] and also with dwell-time interaction by [Tula and Morimoto 2016], demonstrating that other interaction methods can also achieve typing rates over 20 words per minute.

Typing at high speeds using any method requires the user to maintain a high cognitive load. Though allowing high speed typing is certainly an important factor, the ability to instantly adjust to the user's typing rhythm improves comfort and the overall user experience. For example, methods based on discrete gestures adjust to the user's rhythm more easily than dwell-time based methods, and in Dasher the user can explicitly control the typing speed by looking towards the left or right side of the interface.

Eye typing using context switching (CS) was suggested by [Morimoto and Amir 2010]. Instead of pointing and dwelling, selection in CS is activated by a saccade (a fast eye movement between fixations). The method consists of two identical regions called "contexts", separated by a region called "bridge". A full QWERTY keyboard is shown in each context, as seen in Figure 2a.

Using CS, the user is allowed to gaze anywhere within a context for as long as desired, without worrying about unintended selections. To make a selection, the user must focus on the desired key within one of the contexts and saccade to the other context, crossing the bridge. The typing rhythm and speed is determined by the saccades between contexts. The bridge displays the typed text and works as a safety zone between the contexts. If the saccade stops within the bridge, i.e., the bridge is not crossed, no letter is typed.

Compared to a QWERTY dwell-time keyboard, context switching trades space for speed and comfort. It allows the user to freely explore the content of a context, freeing the user's mind from the Midas' touch problem caused by long fixations or having to look left or right on a zooming interface. Selection is fast because it depends on a single saccade, and it is comfortable because the user determines her typing speed to be as slow or as fast as she wishes to type.

The focus of this paper is to exploit the use of expanding targets with context switching. As shown in Figure 1, expanding targets allows significant reduction of the keyboard size when used with dwelling, because it reduces the keyboard to a single line. To avoid the inherent problems of dwelling, in this paper we investigate how increasing the density of keys using expanding targets affects the performance of key selection by context switching.

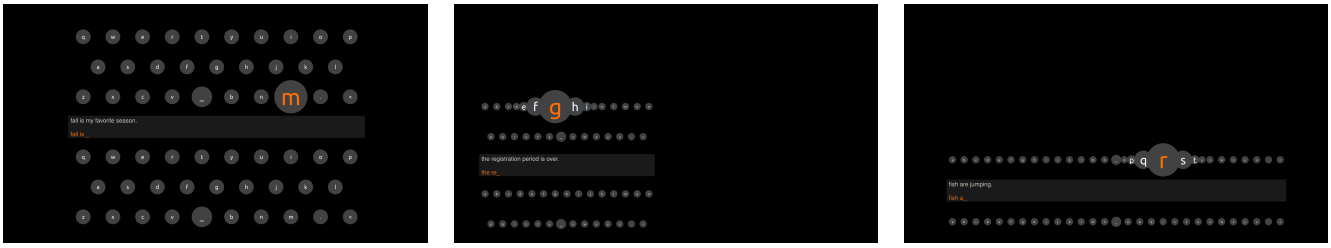
1.1 Gaze interaction with expanding targets

Typically, virtual keyboards for gaze interaction have to use visual targets that are larger than the accuracy of the eye tracker, which is about 1 degree of visual angle. The distance between keys must also be configured to allow users to reliably point and keep their gaze on the target despite eye jitter and imprecise gaze data from the eye tracker. When the size of the keyboard is reduced below this safety margin, pointing becomes unreliable and therefore the user is unable to control the interface.

Because screen space is an important interface resource, applications might require gaze interaction to be restricted to a smaller screen region. Reduced size virtual keyboards such as pie menus and other composed of hotspots for discrete gestures, might require long gestures (with several selections) to reach the desired key. [Ŝpakov and Majaranta 2008] investigated the use of a scrollable keyboard to save screen space by only showing part of a QWERTY keyboard (one or two lines). Though saving space, the method slows down the typing rate. In [Ŝpakov and Majaranta 2008] about 39% of the total number of selections were scroll commands using a 1-row keyboard and and 16.5% using a 2-row keyboard. A single line design such as shown in Figure 1 avoids scrolling but requires a finer control of the positioning, bellow the accuracy of any typical eye tracker.

[Tula et al. 2012] have proposed the use of dynamic sized contexts to optimize the use of space in CS, i.e., the context being focused is maximized while the context without focus is minimized. Because the minimized context had to be focused to be expanded, selections in practice required at least two saccades, one to expand the minimized context and the second to focus on the desired key, slowing down the interaction.

Expanding targets have been suggested to facilitate manual pointing tasks [McGuffin and Balakrishnan 2002; Zhai et al. 2003] in small interfaces and also for gaze interaction [Ashmore et al. 2005; Miniotas et al. 2004]. The basic idea is to display small targets and expand the targets within the user's focus of attention.



(a) (b) (c)

Figure 2: Context Switching using different layouts. a) uses a full QWERTY keyboard as context; b) uses a denser dual line keyboard and; c) uses a single line keyboard.

[Miniotas et al. 2004] showed that the use of static target expansion in motor space only (i.e., invisible to the user) facilitates pointing both in terms of speed and accuracy when using small targets for gaze interaction. In a later publication, [Spakov and Miniotas 2005] suggested that the use of dynamic target expansion (i.e., visible to the user) could also be advantageous for gaze interaction. They successfully applied the technique for selecting menu items, significantly reducing the error rate though increasing the selection time. Their technique expanded the target under focus after a dwell time. Eye jitter and calibration issues might cause the focused target to be incorrect, since targets are small and close together on a menu list. When a target is expanded, its neighboring targets are shifted up and down the list. In case the target is incorrect, the gaze will naturally move up or down, so the selection is adjusted till the desired target is under focus and can be selected by dwell-time.

A similar dynamic target expansion technique using fisheye lens was proposed by [Ashmore et al. 2005]. Instead of using dwell-time for expansion, the fisheye lens worked similar to grab-and-hold, i.e., fisheye expansion occurs at the fixation onset and, during the fixation, the lens would not follow the eye to avoid the distraction caused by a jittery lens locked onto the eye. The authors also tested locking the lens movement with the gaze, but using an attenuation factor inversely proportional to the magnification factor at the measured gaze point within the lens to reduce the jitter effect.

Virtual keyboards including all letters with expanding targets have been included in gaze-based interfaces for browsing the web [Kumar et al. 2017] and in games [Menges et al. 2017]. Nonetheless, those papers focused on tasks different from eye typing, hence no investigation was made about the effect of expanding targets on eye typing speed and comfort. They also used dwell-time activation, that is known to suffer from the Midas touch problem [Jacob 1990].

In this paper we present two new CS-based designs that use smaller contexts with expanding targets to help visualization and control of the keys. The next section presents the designs called single line and dual line context switching.

2 CONTEXT SWITCHING USING DYNAMIC EXPANDING TARGETS

The single line CS displays the whole alphabet horizontally, and in the dual line CS keyboard the alphabet is split into two rows. We will refer to the single and dual line layouts as L1 ans L2 respectively throughout this text. By reducing the sizes of the keyboards, significant screen space can be saved, as seen in FigureËIJ2. These two layouts allow the interface to display other significant information, such as the contents of a browser or email application, simultaneously with the keyboard.

2.1 Single Line Context Switching Layout (L1)

Our first reduced size layout for CS, L1, is shown in detail in Figure 3. Each context contains a 29-key line keyboard, corresponding to the 26 letters of the English alphabet, plus the space bar, backspace, and period, as seen in Figure 3.

The space bar was placed at the center of the line, and the period and backspace at the end (right side) of the line. The size of the space bar is also a little larger than the other keys to facilitate its localization and serve as a reference point.

The diameter of the small keys of the prototype used in the experiments was 0.5^o, with 0.5^o separation between keys. We use a stepwise magnification for the visual interface. The focused key is linearly scaled by a factor of 6, and its 2 immediate neighbors on the left and right hand sides are scaled by 3 and the neighbors of the neighbors are scaled by 1.5. The remaining keys are not scaled. The center of the focused key remains the same, and the neighboring keys are shifted.

In the experiment presented by [Ashmore et al. 2005] a 9×9 grid was used to position the targets. The large separation between targets allow their system to use fixation onset to activate the magnification on each target. In a similar way, the magnification within a context in L1 can also be activated at the fixation onset at a key. Therefore, distant keys can be selected quickly. The key distribution in L1 is much denser than in the fisheye experiment, and more similar to the menus used by [Spakov and Miniotas 2005]. To select neighboring keys and avoid jitter in L1, a distance threshold T computed from the center of the focused key can be adjusted by the user to control when the transition to neighboring keys occur. Typically, T is set to 60% of the radius of the focused key. This allows smooth transitions from the focused key to neighboring keys. When a neighbor key receives focus, all the expanded keys are shifted accordingly to have the largest key always centered at focused key.

The height of the bridge was set to 1^o and the distance between the center of the top to the center of the bottom keyboard lines was 3^o.

Figure 3: Layout using a single scanning line with dynamic expanding targets.

2.2 Dual Line Context Switching Layout (L2)

Our second reduced size layout for CS is shown in detail in Figure 4. The same 29 keys were divided into two lines. The space bar was placed at the center of the lower line, and the period and backspace at the lower right.

The same key dimensions and overall behavior from L1 was used in L2. The line separation between lines of the same context was set to 2^o.

L1 and L2 are designed to be used in scenarios with limited screen space. Compared to the original CS layout, L1 can save up to 75% of screen space. Besides providing a full keyboard within smaller screen space, a second advantage of the novel designs is the reduction of the average saccade length required for each selection. Shorter saccades produce less eye fatigue when used for prolonged periods, i.e., requires less effort overall, improving comfort.

There are also key design differences between L1 and L2. In L1 the distance between contexts is minimized to make vertical saccades that crosses from one context to the other as short as possible. This comes at the cost of a larger horizontal space to search for letters within one context. Because of that, in L2 we split the horizontal line into two lines. Though saccades between contexts are a bit larger the search distance between letters within the same context is smaller.

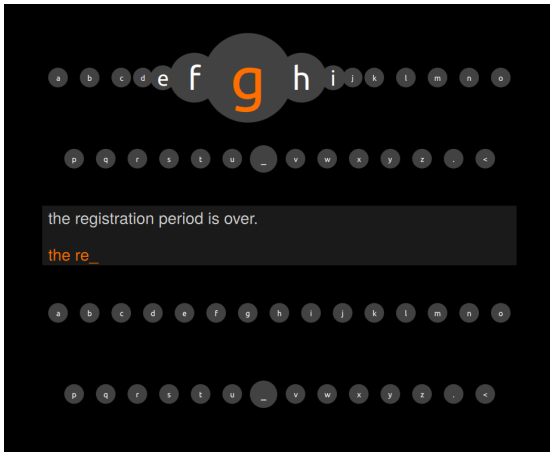


Figure 4: Layout using two scanning lines with dynamic expanding targets.

In order to compare L1 and L2 we conducted an eye typing experiment. We included the original CS QWERTY layout as the baseline condition.

3 METHODOLOGY

We have developed three CS keyboard prototypes for evaluation of the new designs and comparison with the original CS design. We will call the original design L3. Observe that the number associated to each layout name corresponds to the number of lines displaying keys in each context.

In L3, each context contains a QWERTY keyboard as shown in Figure 2. The keys in L3 do not require expansion in motor space because the separation between keys in a horizontal line was large, set to 2^o. The separation between two neighboring lines was also 2^o, the same used for L2.

3.1 Apparatus

All three virtual keyboards were implemented in C++ with Qt/QML. The experiments were conducted using a desktop computer with an 3.2 GHz Intel I5 microprocessor with 8 GB RAM.

Eye movements were tracked using a 60 Hz Tobii EyeX remote eye tracker. The eye tracker was placed at the lower edge of a 23" LCD monitor with a resolution of 1920 × 1080 pixels. The experiment was conducted in a room with artificial illumination and no direct sun light.

3.2 Experimental design

To evaluate how users perform when using each of the layouts in an actual eye-typing task, a within-subjects, repeated measures 3-factorial design was employed where each condition was defined by one of the layouts, L1, L2, and L3.

3.3 Hypothesis

We do not expect significant differences in typing error rate if our expanding target method solves the problem of selecting small targets in crowded spaces, just a reduction in selection time since selection using expanding targets might require adjustments when the saccade does not end exactly on the desired key as pointed by

[Spakov and Miniotas 2005]. Therefore, we hypothesize that eye-typing using the original CS design (L3) will be faster than using the other layouts and that L1 will be the slowest of the 3 layouts because the average distance between keys is longer that in L2. Because L2 follows a mixed design that combines features from L1 (horizontally) and L3 (vertically), we would expect its typing speed to fall between L1 and L3.

3.4 Subjects

A total of 6 unpaid volunteers (all male) with mean age 35 (±7) years old participated in the experiment. All were able-bodied and had normal or corrected to normal vision using contact lenses or glasses. Four subjects had previous experience with other eye-typing methods and had participated in other eye-typing experiments. All subjects were familiar with the QWERTY layout for at least 10 years.

3.5 Procedure

The experimenter first introduced participants to the different layouts and explained the objective of the experiment. After the introduction, participants signed an informed consent form. A regular chair without wheels were used during the experiments. Subjects were seated at a viewing distance of about 60 cm from the monitor. No chin nor a forehead rest were used.

The experiment began with the calibration of the eye tracker using its built-in software. Calibration accuracy was verified by looking at nine points on the screen. After a successful calibration (as evaluated by the experimenter) participants took a training session of about 5 minutes to learn how to use the three layouts. The first eye-typing session followed the training session. Each subject completed 8 sessions, taking breaks of at least 5 minutes at every other session. The order in which the layouts were used in each session followed a Latin square to counterbalance the order of the layouts to reduce learning or fatigue effects. Some participants completed their sessions in two days. After they completed their sessions, they were asked to answer a questionnaire and comment on their experience using each layout.

3.6 Task

The task consisted of transcribing at least 5 phrases per session per layout, or at most 3 minutes of eye-typing per layout. Therefore each session lasted for at least 9 minutes of eye-typing (3 per layout) and collected at least 5 phrases per method.

A new phrase was presented in the text area (the bridge located between the 2 contexts) by pressing the space bar in the computer keyboard. Subjects were instructed to read and memorize the phrase, and to eye-type it as fast as possible, correcting errors only when detected within the word being typed. Every phrase ended with a period. The typed text was shown below the original phrase as it was being entered.

The set of phrases used in the experiment are based in [MacKenzie and Soukoreff 2003]. The order of presentation of the phrases was randomized for each participant.

3.7 Data recording and analysis

During the experiment we recorded all selections made by participants in log files. The time to complete a phrase was computed as the time from the first selected letter to the final dot ".". The following metrics were used to evaluate the performance of the three layouts:

(1) **words per minute** (WPM): number of typed words per minute (one word is considered as a sequence of 5 characters including white spaces [Arif and Stuerzlinger 2009]).
(2) **keystrokes per character** (KSPC): number of key selections needed to produce one character in the final text [Soukoreff and MacKenzie 2001].
(3) **not corrected error rate**: (NER) errors left in the final text using the Minimum String Distance [Soukoreff and MacKenzie 2001].
(4) **corrected error rate**: (CER) errors that have been corrected using backspace and do not appear in the final text [Soukoreff and MacKenzie 2003].

4 RESULTS

The grand mean of WPM considering all participants and all sessions was 8.1 (±2.48) for L1, 8.68 (±2.71) for L2, and 13.1 (±4.39) for L3. Hence, L3 provides the fastest typing experience among the three tested layouts.

Figure 5 shows the results of WPM in each session averaged for all participants. The horizontal axis represents the session number. Each session has 3 points corresponding to each layout. Vertical bars represent one standard deviation. As can be observed, L3's WPM was larger compared to both L1 and L2 in all sessions.

We can also observe in Figure 5 a learning effect of session number on WPM. In the first session, the WPM grand mean was 11.87 for L3, 7.11 for L1, and 7.16 for L2. In the last session, WPM grand mean was 13.42 for L3, 8.69 for L1, and 9.54 for L2. A one-way repeated-measures ANOVA showed a significant effect of session, $F(7, 35) = 3.55$, $p < 0.05$, $\eta_p^2 = 0.42$.

Because in the last 2 sessions the learning curve had already reached a plateau, we analyzed the layouts' performance in those sessions. A one-way repeated-measures ANOVA showed a significant effect of layout, $F(2, 10) = 31.4$, $p < 0.01$, $\eta_p^2 = 0.86$. A paired post-hoc test with Holm correction showed that WPM was significant larger for L3 compared to both L1 and L2 ($p < 0.05$ in both cases). There was not any significant difference between L1 and L2, $p = 0.22$.

With respect to KSPC, the grand mean considering all participants and sessions was 1.18 (±0.15) for L1, 1.16 (±0.15) for L2, and 1.18 (±0.17) for L3. Figure 6 shows the results of KSPC in each session averaged for all participants. We did not observe any learning effect of session over KSPC, as confirmed by ANOVA results: $F(7, 35) = 0.62$, $p = 0.74$, $\eta_p^2 = 0.11$.

Comparing the KSPC in the last 2 sessions, ANOVA test did not reveal any difference between the layouts, $F(2, 10) = 0.65$, $p = 0.54$, $\eta_p^2 = 0.11$. It implies that participants committed approximately the same number of errors to produce the text using each layout.

Regarding NER, the grand mean considering all participants and sessions was 2.12 (±1.91) for L1, 2.47 (±2.02) for L2, and 2.69 (±2.8) for L3. Figure 7 shows the results in each session averaged for all

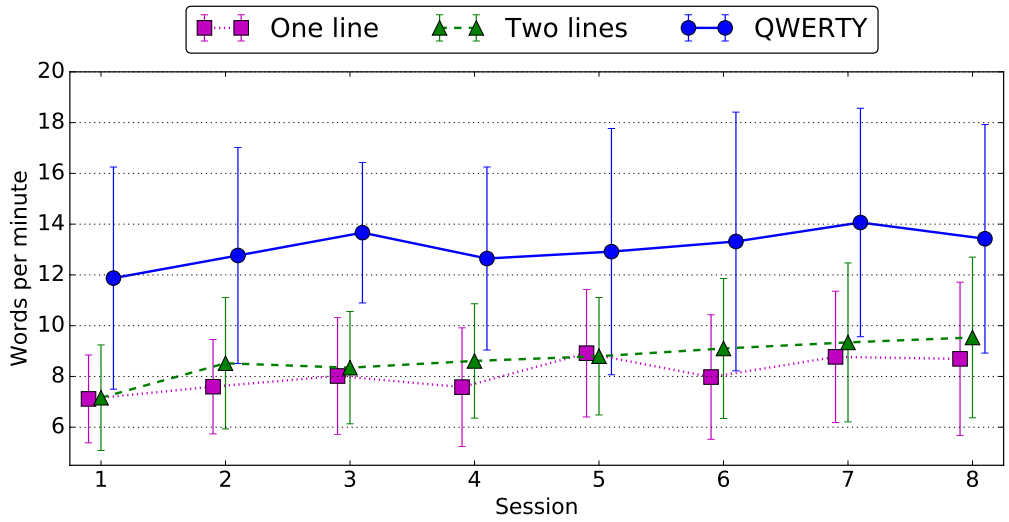


Figure 5: Results of words per minute for all layouts and sessions computed with data from 6 participants.

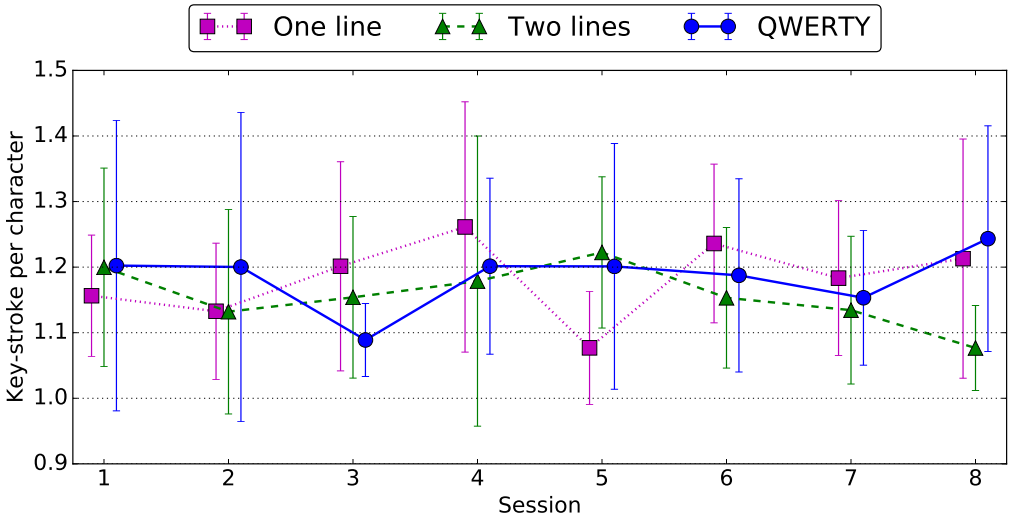


Figure 6: Results of key-stroke per character for all layouts and sessions computed with data from 6 participants.

participants. ANOVA did not reveal any learning effect, $F(7, 35) = 0.78$, $p = 0.61$, $\eta_p^2 = 0.13$.

We compared the layouts NER in the last 2 sessions. ANOVA test did not reveal any difference between the layouts, $F(2, 10) = 0.65$, $p = 0.54$, $\eta_p^2 = 0.11$. Thus all participants typed carefully in all sessions using the 3 layouts.

When analyzing the CER, the grand mean considering all participants and sessions was 7.73 (±5.63) for `L1`, 6.72 (±5.62) for `L2`, and 7.76 (±6.49) for `L3`. As can be observed, CER was larger than NER. This implies that participants were more likely to correct errors while typing than leaving errors in the final text.

Figure 8 shows the results in each session averaged for all participants. We did not observe any learning effect of session, as confirmed by the ANOVA result, $F(7, 35) = 0.54$, $p = 0.8$, $\eta_p^2 = 0.097$.

We compared the layouts in the last 2 sessions regarding CER. ANOVA test did not reveal any difference between the layouts, $F(2, 10) = 1.84$, $p = 0.21$, $\eta_p^2 = 0.27$.

Figure 9 shows the compilation of the results from the user questionnaire. We have asked users to grade, according to a Likert scale from 1 (strongly disagree) to 7 (strongly agree), to the following statements:

- `Learnability`: It was easy to learn to use the system.
- `Speed`: This method allows me to type fast.
- `Accuracy`: It was easy to select the keys.
- `Comfort`: I felt comfortable typing using this method.
- `Fatigue`: Typing using this method made me tired.
- `Cognitive Load`: Typing using this method demanded high cognitive load.

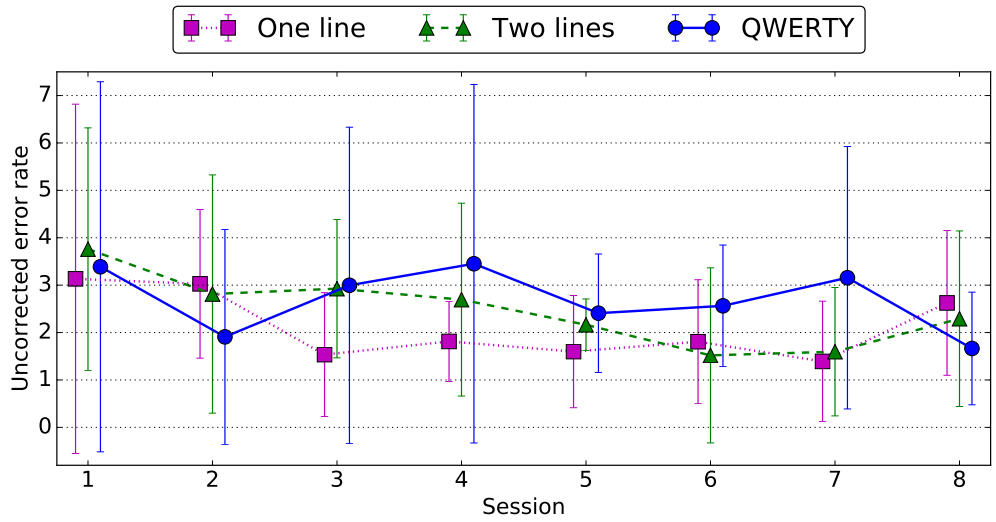


Figure 7: Results of not corrected error rate for all layouts and sessions computed with data from 6 participants.

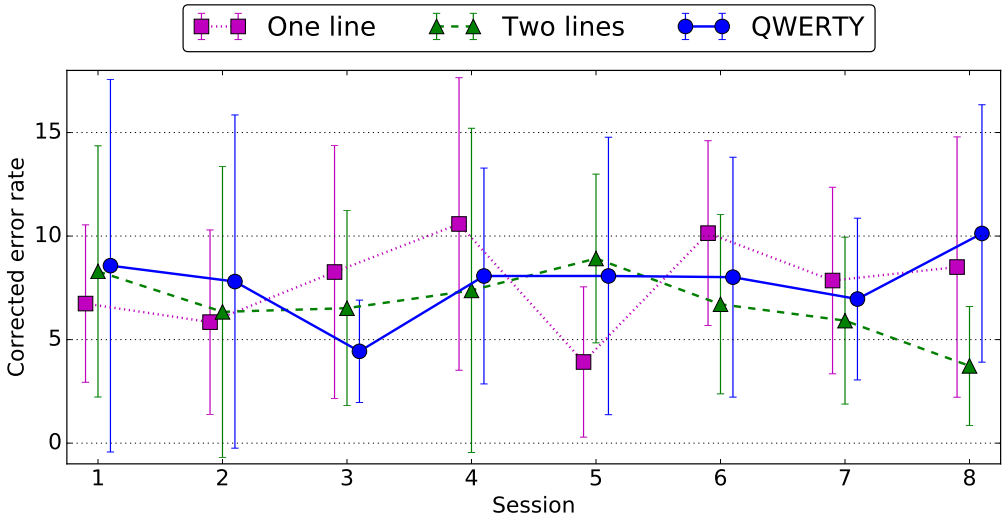


Figure 8: Results of corrected error rate for all layouts and sessions computed with data from 6 participants.

- `Like`: I enjoyed typing using this method.
- `Overall`: This is a good eye-typing method and I would not mind using it if I had to.

The radar graph shows 8 labeled rays, each associated to one of the questions. Points closer to the center correspond to disagreement, and farther from the center to agreement.

Most users (4 out of 2) preferred `L3` over the other methods and, in case the choice were between `L1` and `L2`, most users (4 out of 2) preferred `L2`.

5 DISCUSSION

As we have hypothesized, despite the large eye movements required to navigate throughout the contexts in `L3`, the large targets facilitate pointing and selection so users were able to type faster using `L3`, at an average speed of 14 WPM in the last session. The average final speed of the other methods were about 9 WPM. Though we believe the familiarity with the QWERTY layout might be partially responsible for the difference in performance, it is interesting to notice that there was a significant learning effect in all methods, including `L3`, i.e., despite the familiarity, switching contexts became more efficient along the sessions.

Although the results also indicate that `L1` presents the lowest speed and `L2` shows intermediate performance as we have expected, there was no significant difference between the performance of the methods using expanding targets. Probably the time spent searching and adjusting the gaze to select the correct expanded target dominates any advantage of having a smaller average key distance

in L2. The performance achieved by the expanding methods is compatible with pEYEwrite by [Huckauf and Urbina 2008], that could probably work well in a reduced size window about 25% of the screen, with reported speeds of 8 WPM for novices and 13 WPM for experts.

From the results in Figure 6 we notice that the KSPC was about 1.2 for all methods and along all sessions, implying that the number of errors in each layout was about the same all the time. The total number of error is the sum of the corrected (CER) and the not corrected (NER) errors.

It is interesting to notice that the amount of error of each type did not vary much along the sessions or methods either, as shown in Figures 7 and 8. Statistical tests also did not reveal significant difference, showing that subjects were equally careful in all sessions for all methods. Nonetheless, while on average subjects left 2 to 3 uncorrected errors per session, they corrected about 7 errors per session.

One of the subjects mentioned that while typing using L3 the system detected a wrong key for long saccades. We have used an finite state machine to detect the saccades between contexts [Diaz-Tula and Morimoto 2017]. This machine considers a window of 6 gaze samples (about 200 ms) to detect a saccade and, for longer saccades, this parameter might not work well and will be the subject of future investigation.

Another subject commented for L1 that, when the calibration is not good, some letters were typed while the user was reading the text. This might be caused the bridge height of 1^o was not appropriate for the accuracy achieved by the subject's calibration. This might be solved in future versions of the keyboard by dynamically scaling the motor space of the bridge as well.

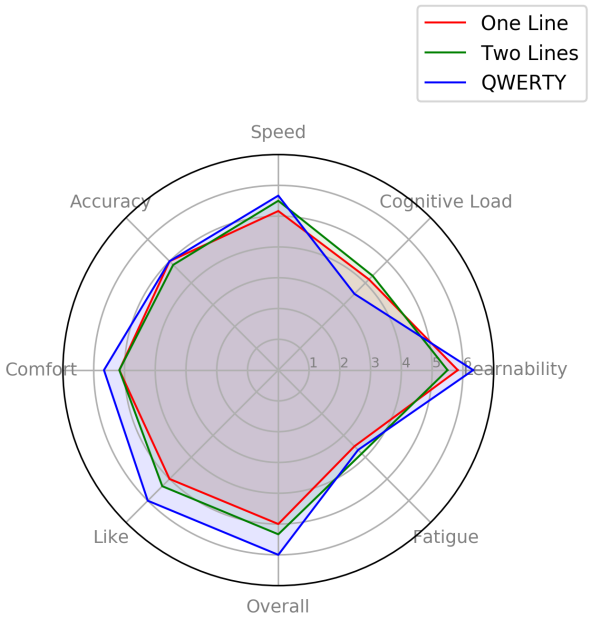


Figure 9: Compilation of the user feedback from the questionnaire.

Regarding the subjective evaluation from the questionnaires, it is clear that the all methods offered a similar user experience, though L3 got better scores overall, which is expected given the subjects' familiarity to the QWERTY layout and its best performance. It does not seem to be a significant difference between the experience using L1 and L2, though there is a slight preference for L2. When asked which was their preferred method if they had to choose between L1 or L2, 4 out of the 6 subjects chose L2.

6 CONCLUSION

In this paper we introduced two new keyboard designs based on context switching for efficient and comfortable selection and expanding targets to facilitate control using reduced sized keyboards.

The original context switching keyboard [Morimoto and Amir 2010] shows two 3-line QWERTY layouts that, though familiar to most users, used most of the screen for text-entry purposes. This problem is also shared by other efficient eye-typing methods such as Dasher [Ward et al. 2000] and limits or make them more difficult to be integrated with computer applications such as web browsers and text editors. Existing space saving methods that use smaller windows for eye-typing, such as pie-menus [Huckauf and Urbina 2008] or graffiti like gestures [Porta and Turina 2008; Wobbrock et al. 2008] are not as fast.

Our first design, L1, reduces each context to a single line with small close together keys. The keys are expanded in visual and motor space on fixation onset for quickly selection of distant keys, but also allows fluid motion to neighboring keys. About 75% of the screen can be saved for other purposes using L1.

Our second design, L2, splits the single line into two. While L1 is meant to use a horizontal slice of the screen, L2 uses about a quarter of the screen. Though the space saving is about the same as L1, its squared shape would allow its use in different application layouts.

We have conducted a pilot eye-typing experiment with 6 users to compare these new designs with a CS keyboard based on the QWERTY layout and without expanding targets. We called this layout L3. Our results show that despite the longer saccades needed in L3 to switch contexts, target selection is easier and, as a result, users were able to type faster using L3, at about 15 WPM, while using the other methods users were able to type at about 9 WPM.

These results have shown that context switching virtual keyboards with expanding keys might be an efficient alternative for smaller keyboards. In future work, we will investigate the effect of expanding targets in two dimensions and include language models for word prediction. A single or dual line keyboard with expanding targets could be also used with dwelling, reducing even further the size of the keyboard. Future work will also investigate the use of dwell-time and compare the user experience with context switching in reduced size keyboards.

ACKNOWLEDGMENTS

This work is partially supported by the São Paulo Research Foundation (FAPESP) under Grant No.: 2016/10148-3.

REFERENCES

Ahmed Sabbir Arif and Wolfgang Stuerzlinger. 2009. Analysis of text entry performance metrics. In *2009 IEEE Toronto International Conference Science and Technology for Humanity (TIC-STH)*. IEEE, 100–105. https://doi.org/10.1109/TIC-STH.2009.5444533

Michael Ashmore, Andrew T. Duchowski, and Garth Shoemaker. 2005. Efficient Eye Pointing with a Fisheye Lens. In *Proceedings of Graphics Interface 2005 (GI '05)*. Canadian Human-Computer Communications Society, School of Computer Science, University of Waterloo, Waterloo, Ontario, Canada, 203–210. http://dl.acm.org/citation.cfm?id=1089508.1089542

Tuhin Chakraborty, Sayan Sarcar, and Debasis Samanta. 2014. Design and Evaluation of a Dwell-free Eye Typing Technique. In *Proceedings of the Extended Abstracts of the 32Nd Annual ACM Conference on Human Factors in Computing Systems (CHI EA '14)*. ACM, New York, NY, USA, 1573–1578. https://doi.org/10.1145/2559206.2581265

A. Diaz-Tula and C.H. Morimoto. 2017. Robust, real-time eye movement classification for gaze interaction using finite state machines. In *Electronic Proceedings of the 2017 COGAIN Symposium*. The COGAIN Association, Wuppertal, Germany.

Dan Witzner Hansen, Henrik H. T. Skovsgaard, John Paulin Hansen, and Emilie Møllenbach. 2008. Noise tolerant selection by gaze-controlled pan and zoom in 3D. In *Proceedings of the 2008 symposium on Eye Tracking Research & Applications (ETRA '08)*. ACM, New York, NY, USA, 205–212. https://doi.org/10.1145/1344471.1344521

Anke Huckauf and Mario H. Urbina. 2008. Gazing with pEYEs: towards a universal input for various applications. In *Proceedings of the 2008 symposium on Eye Tracking Research & Applications (ETRA '08)*. ACM, New York, NY, USA, 51–54. https://doi.org/10.1145/1344471.1344483

Robert J. K. Jacob. 1990. What you look at is what you get: eye movement-based interaction techniques. In *Proceedings of the SIGCHI conference on Human factors in computing systems: Empowering people (CHI '90)*. ACM, New York, NY, USA, 11–18. https://doi.org/10.1145/97243.97246

Per Ola Kristensson and Keith Vertanen. 2012. The Potential of Dwell-free Eye-typing for Fast Assistive Gaze Communication. In *Proceedings of the Symposium on Eye Tracking Research and Applications (ETRA '12)*. ACM, New York, NY, USA, 241–244. https://doi.org/10.1145/2168556.2168605

Chandan Kumar, Raphael Menges, Daniel Müller, and Steffen Staab. 2017. Chromium Based Framework to Include Gaze Interaction in Web Browser. In *Proceedings of the 26th International Conference on World Wide Web Companion (WWW '17 Companion)*. International World Wide Web Conferences Steering Committee, Republic and Canton of Geneva, Switzerland, 219–223. https://doi.org/10.1145/3041021.3054730

Andrew Kurauchi, Wenxin Feng, Ajjen Joshi, Carlos Morimoto, and Margrit Betke. 2016. EyeSwipe: Dwell-free Text Entry Using Gaze Paths. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems (CHI '16)*. ACM, New York, NY, USA, 1952–1956. https://doi.org/10.1145/2858036.2858335

I. Scott MacKenzie and R. William Soukoreff. 2003. Phrase Sets for Evaluating Text Entry Techniques. In *CHI '03 Extended Abstracts on Human Factors in Computing Systems (CHI EA '03)*. ACM, New York, NY, USA, 754–755. https://doi.org/10.1145/765891.765971

Päivi Majaranta, Ulla-Kaija Ahola, and Oleg Špakov. 2009. Fast gaze typing with an adjustable dwell time. In *Proceedings of the 27th international conference on Human factors in computing systems (CHI '09)*. ACM, New York, NY, USA, 357–360. https://doi.org/10.1145/1518701.1518758

M. McGuffin and R. Balakrishnan. 2002. Acquisition of expanding targets. In *Proc. CHI 2002*. ACM Press, New York, NY, USA, 57 – 64.

Raphael Menges, Christoph Schaefer, Chandan Kumar, Tina Walber, Ulrich Wechselberger, and Steffen Staab. 2017. Schau genau! A Gaze-Controlled 3D Game for Entertainment and Education. In *Electronic Proceedings of the 2017 COGAIN Symposium*. The COGAIN Association, Wuppertal, Germany, 1–3.

Darius Miniotas, Oleg Spakov, and I. Scott MacKenzie. 2004. Eye Gaze Interaction with Expanding Targets. In *CHI '04 Extended Abstracts on Human Factors in Computing Systems (CHI EA '04)*. ACM, New York, NY, USA, 1255–1258. https://doi.org/10.1145/985921.986037

Carlos H. Morimoto and Arnon Amir. 2010. Context switching for fast key selection in text entry applications. In *Proceedings of the 2010 Symposium on Eye-Tracking Research & Applications (ETRA '10)*. ACM, New York, NY, USA, 271–274. https://doi.org/10.1145/1743666.1743730

Diogo Pedrosa, Maria Da Graça Pimentel, Amy Wright, and Khai N. Truong. 2015. Filteryedping: Design Challenges and User Performance of Dwell-Free Eye Typing. *ACM Trans. Access. Comput.* 6, 1, Article 3 (March 2015), 37 pages. https://doi.org/10.1145/2724728

Marco Porta and Matteo Turina. 2008. Eye-S: a full-screen input modality for pure eye-based communication. In *Proceedings of the 2008 symposium on Eye Tracking Research & Applications (ETRA '08)*. ACM, New York, NY, USA, 27–34. https://doi.org/10.1145/1344471.1344477

R. William Soukoreff and I. Scott MacKenzie. 2001. Measuring Errors in Text Entry Tasks: An Application of the Levenshtein String Distance Statistic. In *CHI '01 Extended Abstracts on Human Factors in Computing Systems (CHI EA '01)*. ACM, New York, NY, USA, 319–320. https://doi.org/10.1145/634067.634256

R. William Soukoreff and I. Scott MacKenzie. 2003. Metrics for Text Entry Research: An Evaluation of MSD and KSPC, and a New Unified Error Metric. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '03)*. ACM, New York, NY, USA, 113–120. https://doi.org/10.1145/642611.642632

O. Špakov and P. Majaranta. 2008. Scrollable Keyboards for Eye Typing. In *Proceedings of the 2008 COGAIN Symposium*. Prague, Czech Republic, 63–66.

Oleg Spakov and Darius Miniotas. 2005. Gaze-based Selection of Standard-size Menu Items. In *Proceedings of the 7th International Conference on Multimodal Interfaces (ICMI '05)*. ACM, New York, NY, USA, 124–128. https://doi.org/10.1145/1088463.1088486

Antonio Diaz Tula, Filipe M. S. de Campos, and Carlos H. Morimoto. 2012. Dynamic Context Switching for Gaze Based Interaction. In *Proceedings of the Symposium on Eye Tracking Research & Applications (ETRA '12)*. ACM, New York, NY, USA, 353–356. https://doi.org/10.1145/2168556.2168635

Antonio Diaz Tula and Carlos H. Morimoto. 2016. AugKey: Increasing Foveal Throughput in Eye Typing with Augmented Keys. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems (CHI '16)*. ACM, New York, NY, USA, 3533–3544. https://doi.org/10.1145/2858036.2858517

David J. Ward, Alan F. Blackwell, and David J. C. MacKay. 2000. Dasher&Mdash;a Data Entry Interface Using Continuous Gestures and Language Models. In *Proceedings of the 13th Annual ACM Symposium on User Interface Software and Technology (UIST '00)*. ACM, New York, NY, USA, 129–137. https://doi.org/10.1145/354401.354427

Jacob O. Wobbrock, James Rubinstein, Michael W. Sawyer, and Andrew T. Duchowski. 2008. Longitudinal evaluation of discrete consecutive gaze gestures for text entry. In *Proceedings of the 2008 symposium on Eye Tracking Research & Applications (ETRA '08)*. ACM, New York, NY, USA, 11–18. https://doi.org/10.1145/1344471.1344475

S. Zhai, S. Conversy, M. Beaudouin-Lafon, and Y. Guiard. 2003. Human On-line Response to Target Expansion. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems. CHI '03*. ACM Press, New York, NY, USA, 177 – 184.