ReType: Quick Text Editing with Keyboard and Gaze

Shyamli Sindhwani
University of Auckland
Auckland, New Zealand
ssin820@aucklanduni.ac.nz

Christof Lutteroth
University of Bath
Bath, UK
c.lutteroth@bath.ac.uk

Gerald Weber
University of Auckland
Auckland, New Zealand
g.weber@auckland.ac.nz

ABSTRACT

When a user needs to reposition the cursor during text editing, this is often done using the mouse. For experienced typists especially, the switch between keyboard and mouse can slow down the keyboard editing workflow considerably. To address this we propose ReType, a new gaze-assisted positioning technique combining keyboard with gaze input based on a new 'patching' metaphor. ReType allows users to perform some common editing operations while keeping their hands on the keyboard. We present the result of two studies. A free-use study indicated that ReType enhances the user experience of text editing. ReType was liked by many participants, regardless of their typing skills. A comparative user study showed that ReType is able to match or even beat the speed of mouse-based interaction for small text edits. We conclude that the gaze-augmented user interface can make common interactions more fluent, especially for professional keyboard users.

CCS CONCEPTS

• **Human-centered computing** → **Text input**; **Pointing**.

KEYWORDS

Eye gaze tracking; text editor; positioning; keyboard; typographical error; patching metaphor; natural user interfaces

ACM Reference Format:
Shyamli Sindhwani, Christof Lutteroth, and Gerald Weber. 2019. ReType: Quick Text Editing with Keyboard and Gaze. In *CHI Conference on Human Factors in Computing Systems Proceedings (CHI 2019), May 4–9, 2019, Glasgow, Scotland UK*. ACM, New York, NY, USA, 13 pages. https://doi.org/10.1145/3290605.3300433

CHI 2019, May 4–9, 2019, Glasgow, Scotland UK

ACM ISBN 978-1-4503-5970-2/19/05…$15.00
https://doi.org/10.1145/3290605.3300433

1 INTRODUCTION

With advances in hardware and software, eye gaze tracking may soon be built into most of our devices, especially on the desktop. Now imagine you spot a typo you need to correct. What would you expect from a natural user interface (NUI)? That you still have to use the mouse to point to the typo, although you are already looking at it? We present ReType, a NUI paradigm integrating keyboard with gaze: you look at the place you want to edit, and "re-type" the text as you want it, including a short lead-up to the changed characters. This is similar to ways we speak about typos while we correct them.

Eye gaze trackers are used to measure and possibly respond to human eye gaze. While interacting with the computer screen, eye gaze is a good indicator of the user's point of interest and is considered one of the most natural forms of input [36, 37, 44]. Hence it seems natural to use gaze input for pointing and positioning tasks. There have been numerous attempts to utilize gaze in everyday computer interaction; gaze has been used for pointing in various applications [1, 12, 21] and can also be used as an augmented input in addition to mouse or keyboard [18, 19, 47].

However, gaze input comes with several design constraints which need to be addressed in order to design user interfaces with minimum eye input inaccuracies [11]. Avoiding the Midas Touch problem [41] is one such constraint. If we want to use gaze movements as intentional commands, it can be difficult to distinguish them from unintentional gaze movements necessary to consume the information on the user interface. This makes the use of eye gaze in a NUI challenging. Another obstacle is the inaccuracy of gaze input. This is partly due to hard physiological limitations, namely the one degree visual angle covered by the fovea, the area of the retina responsible for sharp central vision [38, 42]. This creates problems for selecting small and closely positioned targets such as text. Furthermore, holding the gaze, e.g., to indicate a selection (dwell), requires a specific novel effort from the user, which can be difficult and frustrating. In this paper we investigate a gaze-augmented NUI for cursor positioning and text editing with a traditional keyboard which avoids these problems.

Many users who work intensively with important productivity applications, specifically for text input, still use keyboard and mouse as their prime input devices. Sometimes users with different tasks or preferences choose not

to use those input devices and look for other viable alternatives. For an experienced typist, moving the hand from the keyboard and using a mouse for pointing & clicking hinders the typing flow and can cause discomfort and fatigue [46]. This repetitive context switch between keyboard and mouse is not only time-consuming but also distracts the typist from the actual task, leading to reduced productivity [7, 34, 45]. The overuse of these pointing devices can also lead to RSI (Repetitive Strain Injury) [6, 29]. People spending more time using keyboard than mouse at work are at lower risk for Carpal Tunnel Syndrome [2].

Scope: We present a novel gaze-augmented method for text editing called ReType. It aims to facilitate the process of editing text by removing the aforementioned context switch between keyboard and mouse, allowing typists to make the best use of their typing skills. ReType combines gaze with keyboard input to replace common pointing operations; it can be used to make small changes in text anywhere on the screen, such as for typo correction, as well as to navigate in the text using only gaze and keyboard input. Beside these new operations, the keyboard can be used for text editing as usual. A variation of navigation can be used to select text, so that copy & paste operations can also be used as usual. We investigate ReType by addressing the following research questions:

RQ1 How can gaze be naturally combined with keyboard input for text editing?
RQ2 How does ReType compare with conventional mouse & keyboard interaction?
RQ3 How does the typing speed of a user affect ReType compared to conventional mouse & keyboard interaction?

For small text edits such as typo correction, ReType merges the positioning process (within the visible text) with the actual change process. We show that ReType offers an opportunity to include gaze tracking as a positioning cue in such a way as to become second nature to the users while they focus on the change they want to make ("automatic ReType"). We also consider variants of ReType which rely on more explicit, modal positioning cues ("modal ReType"), including a variant which works without gaze ("keyboard-only ReType").

Correcting typos with a keyboard usually involves placing the cursor at the right location and using procedural backspace or delete for editing. By contrast, ReType uses a new process which we call 'patching' or 'replacing' the typo. One can think of ReType as a metaphor of patching over a typo with a band-aid: the user re-types some correct characters preceding the typo (prefix), then continues with the correct spelling in place of the erroneous characters, and continues a little past the typo (suffix). Prefix and suffix allow the new patch to 'stick' to the matching bits in the old text, allowing ReType to locate the desired position of the patch without mouse or cursor keys.

We evaluated the usability of the different ReType variants, comparing them against keyboard & mouse. We did not consider using a trackpad or solely navigation keys for cursor positioning as the mouse is more productive [5, 22]. We performed a qualitative free-use study, involving proofreading with a range of different text editing tasks. We also conducted a quantitative study evaluating the central operation of ReType: positioning and editing small changes by re-typing. We compared ReType's ability in correcting typos – as typical examples of small, localised changes – with that of keyboard & mouse.

In summary, we make the following contributions:

(1) We present the design and implementation of ReType, a novel gaze-assisted input method allowing keyboard-only work, which uses a metaphor of changing text by patching.
(2) We show that in ReType, gaze tracking is essential for achieving a performance which can compete with mouse & keyboard.
(3) We show that ReType with its patching metaphor can benefit experienced keyboard users in particular.

2 RELATED WORK

Jacob et al. [13] explored technical and physiological limitations of gaze tracking such as eye tracker accuracy, sensor lag and fixation jitter. These are responsible for problems when using gaze for interaction, such as the Midas Touch problem [11, 12]. In order to overcome these problems, gaze interaction methods have been proposed which are tolerant to fairly low accuracy [18, 24, 26, 31, 32, 48].

Some research explores selection mechanisms for explicitly given targets, such as buttons and links [24, 31, 32, 44, 48]. These approaches derive speed advantages from the limited number of targets, but they are not suitable for fine positioning such as character-wise cursor placement in a text editor. Ware et al. [44] proposed selection with gaze and a physical button, noting that while this approach worked well for larger targets, a reduced speed and increase in number of errors were reported for smaller targets. Zhang et al. [48] focused on improving the stability of an eye gaze cursor using multiple techniques such as force fields, which were suitable mainly for button-sized targets. In our workgroup we proposed a number of two-stage selection approaches which increase the resolution to a degree suitable for smaller links [24, 31, 32]. The fastest technique, Actigaze [24], approaches the speed of the mouse, but is explicitly not aimed at fine-grained positioning.

Gaze-enhanced input methods use existing input devices and employ gaze to improve their usability. Zhai et al. [47] presented MAGIC ("manual and gaze input cascaded"), the first gaze-enhanced mouse pointing approach, where the mouse pointer moves quickly to the user's gaze position and a physical mouse is used to make finer movements of the mouse pointer. MAGIC proved to be slightly faster than the mouse alone, with similar accuracy. The Rake Cursor [3] and Ninja Cursors [33] use an alternative gaze-enhanced mouse technique: the mouse controls a grid of cursors while gaze is used to select the active cursor from within the grid.

The GUIDe (Gaze-enhanced User Interface Design) project explored how gaze can be effectively used as an augmented input in addition to keyboard and mouse for improving the user interfaces for everyday computing tasks [19, 20]. They presented practical applications for pointing & selection [18], application switching [17] and scrolling. Fono et al. [8] proposed eye pointing with either dwell time or key activation for focusing a GUI window. Results indicated that while automatic activation was slightly faster than keyboard activation, the latter was preferred by most participants.

Kumar et al. [18] proposed EyePoint, which allows precise pointing & clicking with gaze using magnification and a press-release button. EyePoint allows the user to keep their hands on the keyboard, using the keyboard essentially just as a mouse button. However, the magnification has a high visual impact on the user experience.

There are some approaches for using gaze input for text based user-interfaces, with most text entry and error correction applications specifically designed for people suffering from motor impairment. GazeTalk [9] and Dasher [40] are examples of text entry systems. GazeTalk uses specialized on-screen keyboards with dwell time for key activation while Dasher lets users enter text directly using continuous pointing gestures. The gaze-only text entry system proposed by Kishi et al. [15] includes support for copy & paste editing using a two-step cursor control mechanism. Both initial cursor setting and fine adjustment are done using dwell time.

Some work focuses on increasing the typing speed for on-screen keyboards either by adjusting dwell [25, 27] or proposing virtual on-screen keyboards with novel layouts [10, 35, 43]. Some "dwell-free" typing approaches [16, 23, 30] have been proposed to mitigate the effort and time taken by gaze dwelling on keys. Majaranta et al. [26] presented a system for gaze-only text editing using dynamic and static gaze menus. A menu pops up when the user fixates on text to be edited, offering dwell-activated buttons for editing functions such as positioning the cursor within the text. The approach supports basic text editing (cut, copy, paste) and text formatting (bold, italic, underline).

OverlapKeys [39] works as a software filter which focuses specifically on overlap errors by identifying and filtering out additional key presses. On the other hand, TrueKeys [14], an auto-correction system, combines models of word frequency, keyboard layout and typing error patterns to identify and correct text automatically while typing.

3 MODAL RETYPE

The initial interaction design for ReType, which we call "modal ReType," is a classic temporary mode that can be activated with a hot key and added to a standard ASCII style or WYSIWYG text editor. The state machine for modal ReType is shown in Figure 1. A user can switch to "ReType mode" by pressing a pre-defined hotkey (we chose the Windows key). Pressing the Escape key exits ReType mode and returns the editor to the same state it was in before entering ReType mode; we will assume that the original state is the typical editing mode of the text editor and call it the "normal mode." From this point a user can switch into ReType mode again. To support ReType, the text editor has an added text field – the ReType field – for reviewing the input entered during ReType mode. While in ReType mode, all user input is shown in the ReType field. We call the text shown in the ReType field the *edit string*. The edit string may contain spaces, and control keys such as backspace have their usual effect in the ReType field.

The first defining property of ReType is that immediately after each key stroke it performs a match of the current edit string against the original text. The match is approximate, i.e. it uses a measure of similarity to match text. ReType replaces the best match in the text with the current edit string, based on the assumption that this is the user's intended target. If the edit string is different from the matched text, then the edit string is considered the desired edited text to be used in place of the original text (i.e. resulting in a text edit). If the edit string is an exact match with part of the text, no modification occurs and the cursor is merely moved right after the match (i.e. resulting in cursor positioning).

In principle this mechanism works well without any further input in unambiguous situations, i.e. if the best approximate match in the text is indeed the desired target of the ReType edit operation. If the user is content with the current replacement performed by ReType, then the user can confirm and embed this change by pressing Enter. This ends the ReType mode, which is designed as a temporary mode active only during the editing process, and places the cursor right after the edited text. In this optimistic case, a first contribution of ReType lies in making a search/replace dialogue a bit less modal by combining the find and the replace phases. This makes ReType more attractive for positioning and small correction tasks.

In many cases, however, the approximate match with the edit string will not be unique, e.g. if the same word is used repeatedly in the text and misspelled in one place, or if several

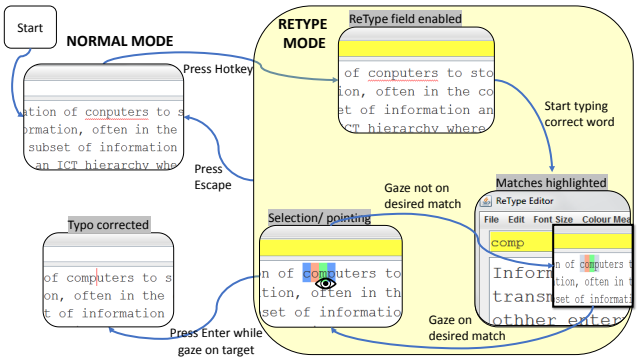


Figure 1: State machine for modal ReType.

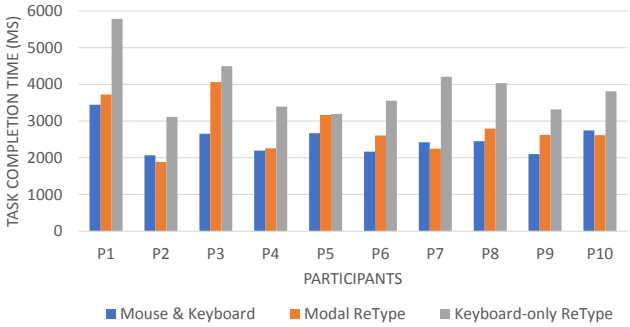


Figure 2: Average task completion times in milliseconds of all pilot study participants for the three tested interaction methods.

similarly spelled words are present. The key idea of ReType, in the spirit of NUIs, is that in ambiguous cases the user's gaze can be used to disambiguate the user's intention, without an explicit pointing approach. Gaze is simply used as an additional positioning cue, based on the reasonable assumption that the user will look at the text she wants to modify (or navigate to). As a result, ReType creates a synergy between gaze input, the fact that the user's gaze rests naturally on the point of interest, and the approximate match of the edit string.

Keyboard-Only ReType

The fact that the keyboard input alone gives a good positioning cue, in some cases even an unambiguous position, leads to a natural counterposition. In the interests of a critical discussion of modal ReType, a detractor may argue that a still rare and moderately pricey gaze tracker could be replaced by disambiguating between potential matches with a go-to-next hotkey. This leads to a variant we call "keyboard-only ReType," which works solely with the keyboard without having to use eye gaze or mouse input. To make a point for ReType as a gaze-augmented interaction method, we will show empirically that the combination of ReType with gaze tracking is necessary to deliver good usability and compete with the mouse. Keyboard-only ReType does not deliver the full advantage.

The working of keyboard-only is similar to ReType with gaze, with one difference: amongst all the potential matches in the text, the desired match is selected by moving a highlight marker with arrow keys instead of gaze input. The user presses the Windows key, enters the ReType mode and types the desired edit string (e.g. a corrected snippet of a word with a typo) into the ReType field. Then she uses the arrow keys (Left, Right, Up, Down) to move between the matches found in the text file. After the correct match is highlighted, the user presses Enter to lock the changes in the document.

Pilot Study

The pilot studies played an important role in the design ideation of the ReType prototype. We followed the approach of "learning by doing" and performed many rounds of iterative testing during the design. After the initial prototype (modal ReType) was ready, a fully fledged pilot study was conducted with ten participants, which helped us to gain an overall idea of the prototype's usability. The users were asked to perform typo editing with eight text files containing five typos each, using a methodology similar to the full evaluation described below. Three different interaction methods were used: 1) modal ReType, 2) keyboard-only ReType, and 3) mouse & keyboard. The mouse & keyboard editing was the usual editing without gaze where a user clicks at the typo, deletes the wrong characters and types the correct ones.

Figure 2 depicts the task completion times averaged over all 40 typo correction tasks for each participant. From these results it is evident that keyboard-only ReType was markedly slower than the other two interaction methods. The 95% confidence interval for the median relative ratio of the task completion times of modal ReType and mouse & keyboard was [0.93, 1.25], with 1 signifying equal speed. This is an indication that modal ReType is likely not a lot slower than mouse & keyboard. Compared to other alternatives for the mouse such as trackpads this can already be considered a good result. Participants' comments and observations made during the pilot study indicate that users were excited about the idea of a gaze-augmented interface for text editing.

Further analysis of the results showed that a major part of the task completion time for modal ReType was spent on the mode switch. Supporting this, several participants pointed out that the multiple steps required for editing with ReType were not optimal.

This led us to the hypothesis that the ReType hotkey created a mental load as users were not used to it in normal text editing. As a result, we looked for an interaction design

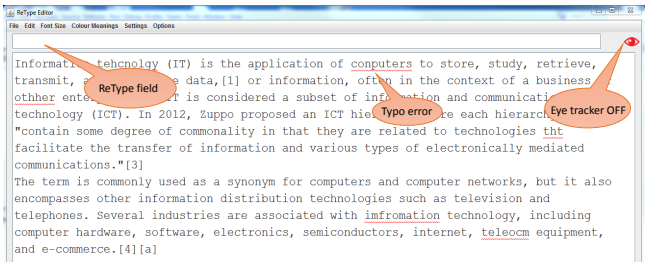


Figure 3: Screenshot of text editor in normal mode. The editor behaves as usual until a ReType operation is performed.

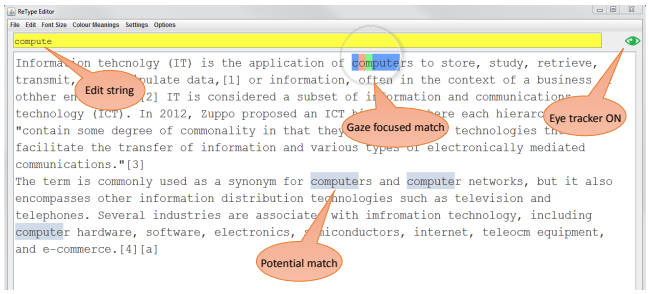


Figure 4: Screenshot of text editor in ReType mode. Matches of the edit string (yellow box) in the text are highlighted.

which would avoid the mode switch. This design is presented in the next section.

4 AUTOMATIC RETYPE

Our latest design is called "automatic ReType" as it removes explicit mode switching. In the spirit of natural interaction, ReType tries to detect the user's intention to switch into ReType mode and then switches to ReType mode automatically. The most promising solution we identified is a mode switch depending on the position of the user's gaze.

In order to avoid interference with normal typing, there is no mode switch if the user looks at (or near) the current position of the cursor. Furthermore, while typing, the user can naturally move her eyes between the keyboard and the screen without triggering a mode switch. ReType will be only activated if 1) the user looks away from the cursor and 2) types a string that is an approximate match with text at (or near) the gaze position.

In order to reduce the number of unintended approximate matches, we introduced the following strategy: for a match to be considered, a fixed number of characters at the start of the edit string, which we call *lead characters*, need to match the characters in the text. The number of lead characters is a configurable parameter of the ReType system. It should not be too large; the only realistic values are 1C, i.e. a single lead character, and 2C, i.e. two lead characters. One can also think of this in terms of the patching metaphor: for a patch to stick at the target position, the edit string should start with one or two characters exactly matching the text.

Figure 3 depicts the editor in normal mode. The red eye in the top right corner of the editor indicates that the editor is not listening to gaze events (either the tracker is off or not connected). A green eye in the top right corner indicates that the editor is listening to gaze events and ready for gaze-augmented text editing. Figure 4 depicts the editor in ReType mode. The ReType field turns yellow to indicate that the user has entered this mode.

Matching Algorithm

As discussed above, in ReType mode the user input constitutes the edit string in the ReType field. This edit string is then used to find the potential matches in the document using an edit distance measure. All potential matches with the lowest edit distance are immediately highlighted with a gray background, see Figure 4.

ReType uses a one-way edit distance, as is common in approximate string matching into a continuous text. The distance is presented as an insert/delete edit script which transforms the text into the edit string, because this aligns with the preferred visualization of the edit (described in more detail below). For example, a character inserted into the edit string should be either a matching character or an insert in the text. One-way edit distance means that leading and trailing deletes are ignored (i.e. have no cost). As said before, ReType requires the first letters in the edit string, the lead characters, to be an exact match with the visible text. This simplifies the calculation of potential matches and also keeps the number of potential matches small, aiding visualization. Consequently the matching algorithm can just loop through the occurrences of the lead characters in the text and compute the one-way edit distance if a match is possible. So in contrast to a general one-way edit distance computation, we know possible intended start points of the match.

We use the well-known Myers' greedy algorithm for fast computation of diff and/or edit distance [28] with one modification: an early termination criterion once the edit string has been completely inserted. In the edit graph used to visualize the Myers' algorithm, this criterion is reached as soon as the algorithm reaches the lower boundary of the graph. At this point the one-way edit distance has been computed and can be returned. Since the only change to Myers' algorithm is an early termination criterion, the good asymptotic runtime properties remain.

Highlighting of Matches

When highlighting matches, they are shown as the text of the edit string without changing the actual document yet, i.e. the displayed text is showing what will happen if the ReType operation is confirmed. To make the temporary changes more informative, inserted characters are highlighted in green, and a character before a deleted character is highlighted in red, indicating the positions where changes would be made, see Figure 4. For example, if the user has just entered 'quick' to correct a word 'qujck', then this potential match in the text is shown as 'quick' ('i' with green highlight and 'u' with red highlight). When confirming the ReType operation by pressing Enter, then the potential match that is closest to the gaze point (if there is one within a certain radius from the point of gaze) gets edited according to the operation. In ReType mode, the system always shows the match closest to the gaze point in a dedicated colour (here with a blue background) to indicate where the current ReType operation will be applied once it is confirmed.

State Machine

Figure 5 illustrates how the coordination is maintained between the gaze input and the automatic mode switchover. In normal mode a user looks at the part of the text to edit and starts typing a correctly edited snippet of the text. This gives the system an indication that the typing is not intended as normal typing in the document, but that a switchover to ReType mode is desired for editing. The system automatically switches to ReType mode, and whatever the user types is entered into the ReType field. Based on the user input in the ReType field (i.e. the edit string), the system starts to look for matches within the visible text. The matches get continually updated with each new character the user types. When the user observes the desired change being applied to the right match, while selecting the match with gaze, she presses Enter to confirm the change. The selected match gets changed leaving all other matches unchanged and the system returns to the normal mode with the modified text. The user can now start the ReType operation again for the next edit (or to position the cursor). Pressing Escape at any stage while performing a ReType operation exits the ReType mode and leaves the text unchanged. Pressing Escape right after a ReType operation reverts the change and brings the system back to the previous, unedited state in normal mode.

ReType for Positioning

ReType is always also performing navigation: the cursor remains at the new position after patch. For pure positioning, an unaltered patch can be re-typed and confirmed with return. Once the cursor is positioned, other editing tasks such as keyboard-based insertion and deletion of text can be

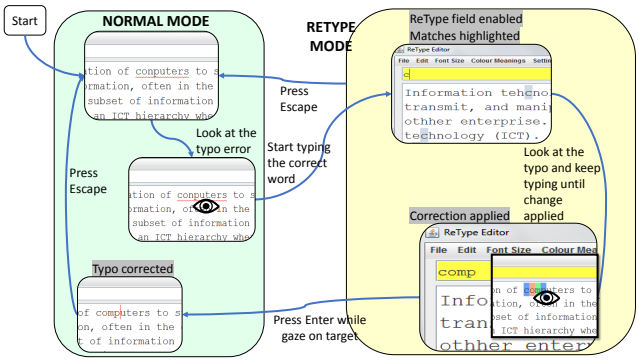


Figure 5: State machine for automatic ReType.

performed normally. This is similar to the way some typists use search functions to navigate quickly in a text.

ReType also supports text selection, which is another common text editing task which usually involves the mouse, in a similar manner to correcting typos. Selection of a text block in ReType is based on navigation, and hence indirectly on patching: it involves navigating to the start position of the desired selection, pressing a special key (we chose the Windows key) to mark this position, navigating to the end position of the desired selection, and finalizing the selection by pressing Enter. This is similar to the mark-select workflow of text selection in some text editors such as Emacs.

5 QUALITATIVE EVALUATION

A free-use study was conducted with seven participants in order to evaluate the use of ReType as a general text editing method together with usual keyboard-based editing. We were interested in observing participants' behaviour and interaction with the system and analyzing its usability. For this purpose, we tried to create a natural text editing setup by providing participants with tasks for proofreading. Two task files (approx. 700 words each) were used: Task 1 contained 60 errors and Task 2 contained 80 errors. A mix of typical types of errors was used, including typos, incorrect words, misplaced text blocks, incorrect text blocks, and missing text.

Procedure

The session started with a demonstration of both ReType variants, modal and automatic ReType. Participants were given several minutes to become familiar with each of them and choose one of the variants for proofreading the texts, depending on their preference. Since the focus of study was on qualitative aspects of usability, we did not ask participants to be as fast as possible; instead, they were asked to correct the text freely in their own time. It was observed that participants spent about 15-20 mins on each task. The study

session was followed by a semi-structured interview. Data was also collected from observation of the study sessions.

Results

Variant Preference. We observed that skilled typists mostly chose automatic ReType as they felt it was more natural and intuitive, while average typists preferred modal ReType as they felt it gave them more control. Two participants who chose modal ReType stated that with more practice they would prefer automatic ReType because of its "naturalness."

Ease of Use. Three participants said ReType required less effort than mouse & keyboard and was therefore easier to use. The other four participants mentioned that ReType required some initial learning; however, after using ReType for a few minutes, they were able to perform editing tasks quickly and efficiently. Three participants were able to complete the second task faster than the first task while making very few mistakes, even though the second task contained more errors than the first. The other four participants completed both tasks in roughly the same time.

Perceived Speed. Four participants felt they were able to complete the tasks faster with ReType than they would have with mouse & keyboard, while maintaining their usual keyboard typing rhythm. Two were uncertain whether ReType or mouse & keyboard were faster. One found ReType slower.

Usage Intentions. All participants were inclined towards using ReType in their own text editing work if it were available. Five participants said they would definitely use ReType, while two said they would consider it after more practice.

Is Keyboard-Mouse Switch a Problem? All participants confirmed that they find the hand switch between keyboard and mouse problematic, with five participants calling it a "hassle." When asked what they do to avoid it, they said they used keyboard shortcuts, arrow keys, and find/replace functions.

Overall Experience. Two participants mentioned ReType's "intuitiveness" in moving the cursor within the visible text. Two most liked not having to move their hands between keyboard and mouse. Text selection, highlighting and typo correction in ReType were also mentioned positively. Six said they felt no fatigue when using ReType; one reported fatigue on the same level as with mouse & keyboard.

6 QUANTITATIVE EVALUATION

We conducted a quantitative experimental study in order to understand the promising qualitative results in more detail. At the heart of ReType use, underlying all other operations, is the "patching" operation; therefore in this study we focused exclusively on patching of small text segments through re-typing, using words with typos as representative examples. We compared automatic ReType against the mouse & keyboard, choosing the mouse as the control pointing device in line with other studies [8, 18, 24, 36] in the field, for comparability with the literature. Previous results show that keyboard-only ReType performed a lot worse than gaze-augmented ReType. Therefore we did not consider it further.

Methodology

The first independent variable of the comparative study was *Method*. It had two levels, ReType and Mouse, and used a within-participant design. The order of the levels was counterbalanced to compensate for order bias and training effects. In order to gain a differentiated insight into ReType, we considered a second independent variable, *Lead Characters*, i.e. the number of preceding characters required for a ReType operation to start, with two levels 1C (single preceding character) and 2C (two preceding characters). We considered it better to expose every participant only to one Lead Characters configuration as it may have been confusing otherwise. We therefore investigated this variable using a between-participant design, dividing our 24 participants into two groups of 12. Participants were pseudo-randomly assigned to groups in a way which would balance groups by typing skill. An overview of the four conditions formed by the independent variables is shown in Table 2. For further analysis, we considered *Error Type* as a third independent variable, i.e. the type of error (typo) a user corrected, with two levels, Single (single error with one affected letter) and Multiple (combination of different types of error and/or multiple affected letters), see Table 1.

The main dependent variables measured were the task completion time, the number of errors (accuracy), System Usability Scale (SUS) scores [4] and a preference ranking. Usability was further measured with additional questionnaires based on a five-point Likert scale and including questions related to specific characteristics of each method. The typing speed of each participant was measured as a covariate. Two variants of task completion time were considered: operating time and motor time. Operating time ('op time') is the overall time spent in correcting a typo. It includes think time, which is the time a participant spends in mental preparations such as identifying the typo and planning actions to be taken, before performing any motor action in the typo correction process. Think time was measured as the time between the instant one typo was corrected and the instant a motor action was taken to correct the following typo. Motor time is the time from the first to the last motor action a participant takes to correct a typo. As a result, operating time is the sum of think time and motor time. In the following, we use the following terms for ReType task completion times: RN

for ReType motor time and RT for ReType operating time. Similarly, for Mouse MN is used for motor time and MT for operating time. The letters T and N indicate whether think time is considered (T) or not (N).

Apparatus. All experiments were conducted on a 14 inch, 2.3 GHz HP laptop with Intel Core i5-6200U processor and 1366×768 resolution. The laptop was configured with the Windows 7 environment (default settings), using a Dell optical mouse and a Tobii 4C gaze tracker. The ReType editor was developed in Java. Identical experimental equipment and settings were used for all conditions. All task text files were small enough to be completely visible in the editor without scrolling. Gaze, key press and mouse events were recorded automatically for accurate measurement of task completion times.

Participants. 24 (17 men, 7 women) participants were recruited, aged from 21 to 38 years with a variety of ethnicities and from a range of disciplines including Computer Science, Engineering and Psychology. All were regular computer users, using computers between two and ten hours per day and reading English text for between two and twelve hours per day. The typing speeds recorded for the participants were between 25 and 83 WPM. Ten participants wore either glasses or contact lenses of varying strengths. Eight participants had prior experience with gaze tracking.

Task. For each method participants were given the same five texts, which contained six typos each, resulting in each participant editing 30 typos in total. The five texts were taken from Wikipedia and typos were introduced artificially. The density of typos is relatively high for reasons of practicality in administering the study. The typos were chosen for representativeness to cover all positions in a word as well as covering complex misspellings, a mix of incorrect letters, additional letters, missing letters, reverse order of letters or a combination of these. Table 1 gives an overview of the categories of typos used in the study. In order to reduce seek time for the typos, the open source Spell Check library was used to highlight the incorrect words in the text file; it was not used to perform the actual correction. As we want to measure interaction time and accuracy, and not a participant's skill in correcting spelling mistakes, all participants were provided with printouts of the task text files before the start of each task. These printouts showed each typo highlighted and corrected, so that the participant did not have to think much about the correct spelling while interacting with the interface.

Procedure. The procedure is illustrated in Figure 6. After being comfortably seated in an adjustable chair, participants were first asked to fill out a demographics questionnaire, followed by a one-minute typing speed test. Participants

Table 1: Typo categories (error type and letters affected) with frequencies.

Type of Error	1 letter	2 letters
Incorrect	5	1
Additional	5	1
Missing	4	2
Order	6	-
Additional + Missing	-	3
Order + Incorrect	-	1
Order + Missing	-	2

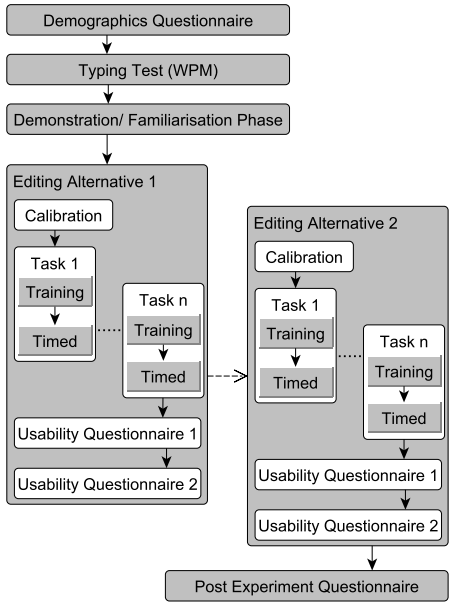


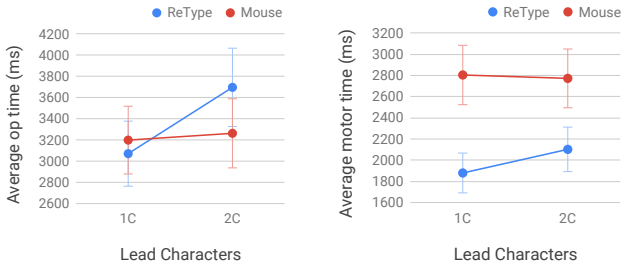
Figure 6: Procedure used in the quantitative evaluation.

were then briefed about the nature of the experiment and were familiarized with the input methods, editor functions and experimental tasks. The gaze tracker was calibrated for each participant using the standard Tobii Eye Tracking Core software. Participants were instructed to correct typos quickly and accurately. For each condition, we had a pre-task training where participants were trained on the text before the actual measurement.

After their interaction in each condition, participants were asked to fill out the SUS and additional usability questionnaires. The experiment concluded with the participants filling out a post-experiment questionnaire where they ranked the input methods based on their preference and provided comments explaining the ranking. The experiment took approximately 60-70 minutes for each participant.

Table 2: Task completion times (milliseconds, $M \pm SD$).

Method	Lead Characters	n	Op time	Motor time
ReType	1C	12	3070.92±591.64	1878.58±461.31
	2C	12	3696.92±807.69	2102.42±668.05
Mouse	1C	12	3198.50±504.80	2803.50±524.48
	2C	12	3262.58±432.13	2771.92±356.17



(a) Operating time **(b) Motor time**

Figure 7: Comparison of ReType and Mouse a) operating time, and b) motor time for 1C and 2C conditions.

Results

We validated that the data satisfies the assumptions of an analysis of variance (ANOVA), such as normality of the dependent variables and homoscedacity. All tests for significance were made at the $\alpha = 0.05$ level. The error bars in the graphs show the 95% confidence intervals of the means.

Task Completion Time. The results for average operating time and motor time across all typos are summarized in Table 2 and illustrated in Figure 7. A two-way within-subjects ANOVA was conducted to test the influence of Method and Lead Characters on average op time. The main effects for Method ($F(1, 22) = 1.44, p = .24$) and Lead Characters ($F(1, 22) = 2.72, p = .11$) were not significant. However, the interaction effect was significant ($F(1, 22) = 4.82, p = .03$). An independent samples t-test showed a significant difference in ReType 1C and 2C conditions ($t(22) = -2.17, p = .04$) with a 'large' effect size (Cohen's d > 0.8). The results indicate that ReType with 1C is notably faster than with 2C (see Figure 7a). Therefore we focus our analysis on the more promising 1C.

A two-way within-subjects ANOVA was conducted to test the influence of Method and Lead Characters on average motor time. The main effect for Method was significant ($F(1, 22) = 61.22, p < .001$) but the main effect for Lead Characters ($F(1, 22) = 0.27, p = .60$) and the interaction effect ($F(1, 22) = 1.57, p = 0.22$) were not significant. Post hoc t-tests with Bonferroni correction showed a significant difference between ReType motor time and Mouse for 1C ($t(11) = -8.06, p < .001$) and 2C ($t(11) = -3.98, p < .01$) with a 'large' effect size (both Cohen's d > 1). These results show that ReType motor time is faster than Mouse for both 1C and 2C conditions (see Figure 7b).

The frequency distributions of task completion times for the promising ReType 1C condition and Mouse are shown in Figure 8. ReType was faster than Mouse (Figure 9); however, a paired samples t-test comparing average op time (ReType (RT) and Mouse (MT)) for 1C was not significant ($t(11) = -0.84, p = 0.42$). A paired samples t-test comparing average motor time (ReType (RN) and Mouse (MN)) for 1C showed that ReType was significantly faster than Mouse ($t(11) = -8.06, p < .001$) with a 'large' effect size (Cohen's d = 2.3).

Effect of Error Type. The results for average op time and motor time for Error Type S (single affected letter) and M (multiple affected letters) are summarized in Table 3 and illustrated in Figure 10. A two-way within-subjects ANOVA was conducted to test the influence of Method and Error Type on average op time (Figure 10a). The main effect for Method ($F(1, 28) = 0.004, p = .99$) was not significant. The main effect for Error Type ($F(1, 28) = 39.36, p < .001$) and the interaction effect ($F(1, 28) = 21.34, p < .001$) were significant. A paired samples t-test comparing average op time of ReType (RT) and Mouse (MT) for Error Type M was not significant ($t(9) = -1.84, p = 0.09$).

A two-way within-subjects ANOVA was conducted to test the influence of Method and Error Type on average motor time (Figure 10b). The main effect for Method was significant ($F(1, 28) = 97.95, p < .001$), with ReType faster than Mouse. The main effect for Error Type ($F(1, 28) = 41.21, p < .001$) and the interaction effect ($F(1, 28) = 27.89, p < .001$) were also significant. A paired samples t-test comparing average motor time (ReType (RN) and Mouse (MN)) for Error Type M showed that ReType was significantly faster than Mouse ($t(9) = -6.04, p < .001$) with a 'large' effect size (Cohen's d = 1.9).

Influence of Typing Speed. We conducted a Pearson correlation analysis on ReType and Mouse task completion times (op times RT/MT and motor times RN/MN) with typing speed (WPM), which showed significant correlations with RT ($r = -0.65, p < .001$), RN ($r = -0.72, p < .001$), MT ($r = -0.58, p = .003$) and MN ($r = -0.67, p < .001$). A simple linear regression analysis showed that there is a significant effect of typing speed on ReType op time (RT) and on Mouse op time (MT). A significant regression equation for RT $5096 - 34.47$ WPM ($F(1, 22) = 15.94, p < .001, R^2 = 0.42$) indicates that typing speed is a significant predictor. Similarly, a significant regression equation for MT $4160 - 18.72$ WPM ($F(1, 22) = 11.30, p = .003, R^2 = 0.34$) indicates that typing speed is a significant predictor. Looking only at ReType with

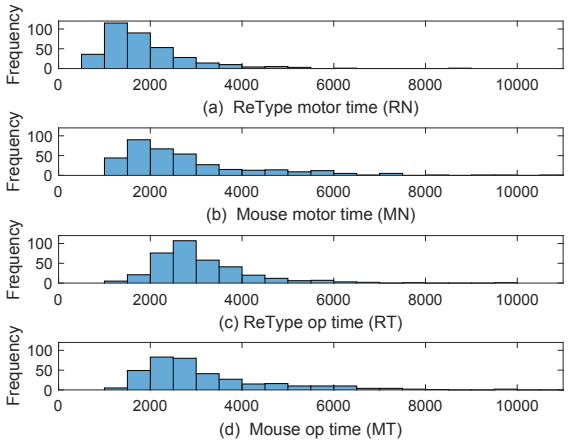
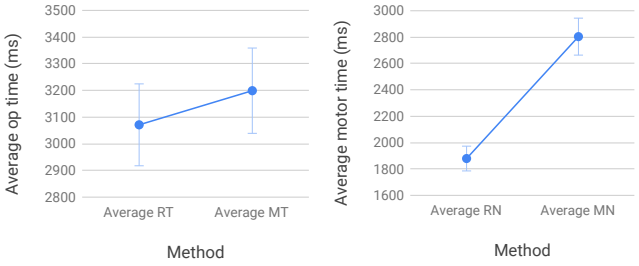


Figure 8: Frequency distributions of task completion times in milliseconds for ReType 1C and Mouse.



(a) Operating time **(b) Motor time**

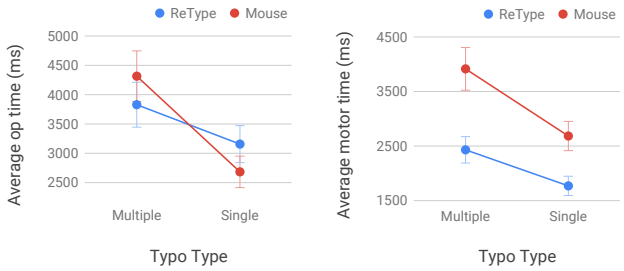
Figure 9: Comparison of a) ReType op time (RT) and Mouse op time (MT), and b) ReType motor time (RN) and Mouse motor time (MN) for the 1C condition.

Table 3: Task completion times (milliseconds, $M \pm SD$).

Method	Error Type	n	Op time	Motor time
ReType	M	10	3832.90±488.41	2431.60±433.99
	S	20	3159.60±358.00	1769.95±310.30
Mouse	M	10	4320.00±950.29	3918.80±967.95
	S	20	2686.05±456.03	2222.20±471.34

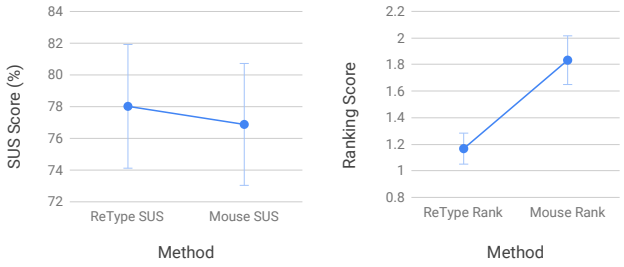
1C, a significant regression equation for RT $4373 - 25.86$ WPM ($F(1, 10) = 5.75, p = .04, R^2 = 0.37$), indicates typing speed is also a significant predictor for the 1C condition. In summary, typing speed reduces op time more for ReType than for Mouse ($\beta = -34.47$ and $\beta = -25.86$ vs. $\beta = -18.72$), which indicates that ReType allows typists to make better use of their skills than Mouse.

Accuracy. ReType errors were counted if the user made an unintentional (incorrect) change in one of the target words



(a) Operating time **(b) Motor time**

Figure 10: Comparison of ReType and Mouse a) operating time, and b) motor time for Mutiple (M) and Single (S) error type conditions.



(a) SUS scores **(b) Preference ranks**

Figure 11: Comparison of ReType and Mouse a) SUS scores and b) preference rankings.

or anywhere else while doing the correction. ReType had an error rate of 1.80% (13 errors) vs. 0.42% (3 errors) for Mouse. Considering the 1C and 2C groups separately, the error rate for ReType 1C was 0.83% (3 errors) and that of ReType 2C was 2.78% (10 errors). We observed that the corrections at the beginning of a word were particularly error-prone with 2C.

Usability. A paired samples t-test was conducted to compare the SUS scores (%) for ReType and Mouse across all 24 participants, illustrated in Figure 11a. ReType ($M = 78.02, SD = 12.62$) scored higher than the Mouse ($M = 76.87, SD = 11.96$), but the difference was not significant ($t(23) = 0.32, p = 0.75$). A paired Wilcoxon Signed-Rank test showed that participants rated their enjoyment of ReType ($M = 4.67, SD = 0.57$) significantly higher than that of Mouse ($M = 3.71, SD = 1.16$), $W = 50, p < .01$, and that participants felt they "had to type too much to correct typos" with Mouse ($M = 3.13, SD = 0.99$), more than with ReType ($M = 2.00, SD = 1.14$), $W = 30.5, p < .01$. One-sample Wilcoxon signed-rank tests indicate that participants agreed (median score greater than 4 on a 5-point Likert scale) that with ReType "it was easy to see what changes will be made in the text" ($M = 4.38, SD = 0.58, V = 60, p < .01$), "it was easy to select the word [they]

wanted" ($M = 4.42, SD = 0.65, V = 90, p < .01$), that they "could relate to the metaphor of patching" ($M = 4.63, SD = 0.58, V = 144, p < .001$) and that they are "willing to type a bit more" if it saves them switching between keyboard and mouse ($M = 4.46, SD = 0.78, V = 112, p = .01$). A one-sample Wilcoxon signed-rank test shows that participants tended to agree (median greater than mid-scale point 3) that they had "to make too many switches between keyboard and mouse to correct typos" when using Mouse ($M = 4.13, SD = 0.74, V = 190, p < .001$), and that "correcting typos just with the keyboard using ReType requires less effort than using a context menu with suggested spelling corrections with the mouse" ($M = 4.04, SD = 1.20, V = 240.5, p = .001$).

Preference. Figure 11b illustrates the participants' preference rankings. Twenty-two participants preferred ReType and two participants preferred the Mouse. A Wilcoxon Signed-Rank test on the preference rankings showed that ReType ($M = 1.17, SD = 0.38$) was ranked significantly better than Mouse ($M = 1.83, SD = 0.38$), $W = 120, p < .01$. A one-sample Wilcoxon signed-rank test shows that participants tended to agree (median greater than mid-scale point 3) that they "prefer using ReType over using keyboard and mouse for correcting typos" ($M = 3.92, SD = 1.06, V = 223, p < .001$).

Qualitative Results. The overall response to ReType was very positive (e.g. *"Once I got going with ReType, I found it quite sensible and easy to use on the whole. I like both systems, but I could certainly see myself happily using ReType a lot."*, *"I like to be able to practice my spelling rather than rely on the computer to tell me the spelling – this is a benefit of ReType."*). Ten participants used words such as "easy to use" or "effortless" and seven participants used words such as "saves time" or "fast" when describing their experience. Seven participants mentioned that ReType was "accurate" and "efficient" in identifying a typo's location. Five other participants said that they found ReType very "intuitive" and that it worked well for correcting typos. Five participants reported that they enjoyed correcting typos with ReType. Six participants agreed that with more practice, ReType will perform better than Mouse. Four participants pointed out that they were more comfortable with Mouse as they had years of experience with it, and that they would need more practice with ReType to reach the same level of proficiency (e.g. *"Given that I have been using mouse and keyboard to type it feels more natural. I think I might change my mind after using ReType for more time."*). All in all, participants liked the fact that ReType saves them manually placing the cursor and considered it as a viable text editing alternative.

7 DISCUSSION

The results indicate that it is possible to combine gaze and keyboard for text editing, in particular for small changes such as typos, but that it requires some consideration to make this 'natural' (RQ1). In terms of task completion time, modal ReType performed worse than the non-modal, 'automatic' ReType as it required a start signal (key press) which was not part of the typical text editing process. Similarly, using two lead character for matching target text (2C) did not work as well as using just one (1C). This is most likely because having to type additional characters for patching deviates from the typical editing process and may introduce friction, and because choosing the characters created additional mental load.

Looking only at motor time, automatic ReType was clearly superior to Mouse for small changes (RQ2). However, while the average op time of automatic ReType with 1C came out better than Mouse, there was no clear superiority. It is likely that the limited experience participants had with ReType, as compared to the Mouse, contributed to a longer think time. Based on comments, many participants believed that they could become more proficient with ReType given more practice. Overall, the responses indicate that ReType's patching metaphor was intuitive for most participants, even though they were not used to it.

The regression analysis indicates that higher typing speeds reduce op time more for ReType than for Mouse, making ReType particularly attractive for typists (RQ3). The regression equations predict that from about 30 WPM upwards, users will be faster with ReType than with Mouse, with average op time for correcting typos below 3600ms. This is supported by observations that skilled typists became fluent with ReType much more quickly than average typists. However, although several participants were skilled typists, the effect on op time was not strong enough to be statistically significant given the limited sample size. Based on the results, the difference between ReType 1C and Mouse would likely become significant with more ReType training.

Limitations

The study presented has some limitations. It was not possible to give participants extensive training with ReType during the experiment, which may have disadvantaged it relative to Mouse. Future work could consider the use of ReType over a longer period of time. Moreover, there may have been a novelty bias in the responses as many participants had not used gaze tracking before. Since it was not feasible to hide from the participants that the experimenter had an interest in ReType, social desirability bias may have influenced the subjective results, in particular the ranking. This was mitigated through the use of standardised, neutrally formulated participant instructions, objective measures (task completion times, accuracy), additional Likert-scale questions to elicit differentiated feedback, and by encouraging genuine feedback. It is likely the study was underpowered to detect

differences in speed between ReType 1C and Mouse; future studies investigating these differences should focus on the more promising ReType 1C.

In this study ReType was used with texts which fitted on a single screen for reasons of study design (reducing parameter space). However, the ReType editor supports editing of texts that are larger than one screen. Many proficient keyboard users are already able to navigate quickly across screens using the 'page up' and 'page down' keys. Furthermore, gaze-supported techniques such as gaze scrolling could be used. In particular, ReType could be extended to show potential off-screen matches of re-typed words of the text together with some context, e.g. at the bottom of the text editor, so that these can be selected by gaze and navigated to as usual.

8 CONCLUSIONS

We have presented the design and evaluation of ReType, a gaze-assisted text editing technique for keyboard, which can be used to make changes quickly in text shown on the screen. Our results show that gaze is crucial for the user experience of ReType, and that once gaze is available, ReType achieves a high user satisfaction. In order to explore the performance potentials, we have implemented automatic ReType, where positioning happens automatically once the system has inferred that the user wants to edit at a particular position. We have ring-fenced the parameter space for ReType which delivers the best results for the user by showing that a single lead character (1C) allows good positioning and ensures fast and reasonably accurate interaction. ReType received positive feedback for small text changes such as correcting typos, as well as in the context of more general text editing, involving text insertion, removal, selection and cut & paste.

ReType is notable as it achieves improvements in particular for professional and already fluent keyboard users. In this sense ReType targets different scenarios than many other gaze interaction techniques which are aimed at situations where a user needs to replace an existing interaction technique. A key conclusion of this study is that users who care about typing speed and efficiency in their work felt that ReType, and therefore the use of gaze, offers an improvement. They considered it plausible that they would use it frequently if it were available.

Future work could investigate the use of ReType 'in the wild,' and its performance after usage over a longer time. Extensions of ReType to support gaze-augmented navigation in longer documents also appear plausible and should be explored further.

REFERENCES

[1] Michael Ashmore, Andrew T Duchowski, and Garth Shoemaker. 2005. Efficient eye pointing with a fisheye lens. In *Proceedings of Graphics Interface 2005*. Canadian Human-Computer Communications Society, 203–210.

[2] Isam Atroshi, Christina Gummesson, Ewald Ornstein, Ragnar Johnsson, and Jonas Ranstam. 2007. Carpal tunnel syndrome and keyboard use at work: A population-based study. *Arthritis & Rheumatism: Official Journal of the American College of Rheumatology* 56, 11 (2007), 3620–3625.

[3] Renaud Blanch and Michaël Ortega. 2009. Rake cursor: improving pointing performance with concurrent input channels. In *Proceedings of the CHI Conference on Human Factors in Computing Systems*. ACM, 1415–1418.

[4] John Brooke et al. 1996. SUS-A quick and dirty usability scale. *Usability Evaluation in Industry* 189, 194 (1996), 4–7.

[5] Stuart K Card, William K English, and Betty J Burr. 1978. Evaluation of mouse, rate-controlled isometric joystick, step keys, and text keys for text selection on a CRT. *Ergonomics* 21, 8 (1978), 601–613.

[6] Suparna Damany and Jack Bellis. 2000. *It's not carpal tunnel syndrome!: RSI theory and therapy for computer professionals*. Jack Bellis.

[7] Franck Dernoncourt. 2014. Replacing the computer mouse. *arXiv preprint arXiv:1410.5907* (2014).

[8] David Fono and Roel Vertegaal. 2005. EyeWindows: evaluation of eye-controlled zooming windows for focus selection. In *Proceedings of the CHI Conference on Human Factors in Computing Systems*. ACM, 151–160.

[9] John Paulin Hansen, Anders Sewerin Johansen, Dan Witzner Hansen, Kenji Itoh, and Satoru Mashino. 2003. Command without a click: Dwell time typing by mouse and gaze selections. In *Proceedings of Human-Computer Interaction–INTERACT*. 121–128.

[10] John Paulin Hansen, Kristian Tørning, Anders Sewerin Johansen, Kenji Itoh, and Hirotaka Aoki. 2004. Gaze typing compared with input by head and hand. In *Proceedings of the 2004 Symposium on Eye Tracking Research & Applications*. ACM, 131–138.

[11] Robert Jacob and Sophie Stellmach. 2016. What you look at is what you get: gaze-based user interfaces. *Interactions* 23, 5 (2016), 62–65.

[12] Robert JK Jacob. 1991. The use of eye movements in human-computer interaction techniques: what you look at is what you get. *ACM Transactions on Information Systems (TOIS)* 9, 2 (1991), 152–169.

[13] Robert J Jacob and Keith S Karn. 2003. Eye tracking in human-computer interaction and usability research: Ready to deliver the promises. *Mind* 2, 3 (2003), 4.

[14] Shaun K Kane, Jacob O Wobbrock, Mark Harniss, and Kurt L Johnson. 2008. TrueKeys: identifying and correcting typing errors for people with motor impairments. In *Proceedings of the 13th International Conference on Intelligent User Interfaces*. ACM, 349–352.

[15] Reo Kishi and Takahiro Hayashi. 2015. Effective gazewriting with support of text copy and paste. In *IEEE/ACIS 14th International Conference on Computer and Information Science (ICIS)*. IEEE, 125–130.

[16] 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 & Applications*. ACM, 241–244.

[17] Manu Kumar, Andreas Paepcke, and Terry Winograd. 2007. *EyeExpose: Switching Applications with Your Eyes*. Technical Report CSTR 2007-02. Stanford University.

[18] Manu Kumar, Andreas Paepcke, and Terry Winograd. 2007. EyePoint: practical pointing and selection using gaze and keyboard. In *Proceedings of the CHI Conference on Human Factors in Computing Systems*. ACM, 421–430.

[19] Manu Kumar and Terry Winograd. 2007. GUIDe: gaze-enhanced UI design. In *CHI'07 Extended Abstracts on Human Factors in Computing Systems*. ACM, 1977–1982.

[20] Manu Kumar, Terry Winograd, Andreas Paepcke, and Jeff Klingner. 2007. *Gaze-enhanced user interface design*. Technical Report. Stanford InfoLab.

[21] Chris Lankford. 2000. Effective eye-gaze input into windows. In *Proceedings of the 2000 Symposium on Eye Tracking Research & Applications*. ACM, 23–27.

[22] Michael YC Lin, Justin G Young, and Jack T Dennerlein. 2015. Evaluating the effect of four different pointing device designs on upper extremity posture and muscle activity during mousing tasks. *Applied Ergonomics* 47 (2015), 259–264.

[23] Yi Liu, Chi Zhang, Chonho Lee, Bu-Sung Lee, and Alex Qiang Chen. 2015. Gazetry: Swipe text typing using gaze. In *Proceedings of the Annual Meeting of the Australian Special Interest Group for Computer Human Interaction*. ACM, 192–196.

[24] Christof Lutteroth, Moiz Penkar, and Gerald Weber. 2015. Gaze vs. Mouse: a fast and accurate gaze-only click alternative. In *Proceedings of the 28th Annual ACM Symposium on User Interface Software & Technology*. ACM, 385–394.

[25] Päivi Majaranta, Ulla-Kaija Ahola, and Oleg Špakov. 2009. Fast gaze typing with an adjustable dwell time. In *Proceedings of the CHI Conference on Human Factors in Computing Systems*. ACM, 357–360.

[26] Päivi Majaranta, Niina Majaranta, Gintautas Daunys, and Oleg Spakov. 2009. Text editing by gaze: static vs. dynamic menus. In *Proceedings of the 5th Conference on Communication by Gaze Interaction (COGAIN)*. 19–24.

[27] Martez E Mott, Shane Williams, Jacob O Wobbrock, and Meredith Ringel Morris. 2017. Improving dwell-based gaze typing with dynamic, cascading dwell times. In *Proceedings of the CHI Conference on Human Factors in Computing Systems*. ACM, 2558–2570.

[28] Eugene W. Myers. 1986. An O(ND) Difference Algorithm and Its Variations. *Algorithmica* 1 (1986), 251–266.

[29] Emil F Pascarelli and Deborah Quilter. 1994. *Repetitive strain injury: a computer user's guide*. Wiley.

[30] 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 Transactions on Accessible Computing (TACCESS)* 6, 1 (2015), 3.

[31] Abdul Moiz Penkar, Christof Lutteroth, and Gerald Weber. 2012. Designing for the eye: design parameters for dwell in gaze interaction. In *Proceedings of the 24th Australian Computer-Human Interaction Conference*. ACM, 479–488.

[32] Abdul Moiz Penkar, Christof Lutteroth, and Gerald Weber. 2013. Eyes only: navigating hypertext with gaze. In *IFIP Conference on Human-Computer Interaction (INTERACT)*. Springer, 153–169.

[33] Kari-Jouko Räihä and Oleg Špakov. 2009. Disambiguating ninja cursors with eye gaze. In *Proceedings of the CHI Conference on Human Factors in Computing Systems*. ACM, 1411–1414.

[34] Julian Ramos, Zhen Li, Johana Rosas, Nikola Banovic, Jennifer Mankoff, and Anind Dey. 2016. Keyboard surface interaction: making the keyboard into a pointing device. *arXiv preprint arXiv:1601.04029* (2016).

[35] Sayan Sarcar, Prateek Panwar, and Tuhin Chakraborty. 2013. EyeK: an efficient dwell-free eye gaze-based text entry system. In *Proceedings of the 11th Asia Pacific Conference on Computer Human Interaction*. ACM, 215–220.

[36] Linda E Sibert and Robert JK Jacob. 2000. Evaluation of eye gaze interaction. In *Proceedings of the CHI Conference on Human Factors in Computing Systems*. ACM, 281–288.

[37] Dave M Stampe and Eyal M Reingold. 1995. Selection by looking: a novel computer interface and its application to psychological research. *Studies in Visual Information Processing* 6 (1995), 467–478.

[38] Sophie Stellmach and Raimund Dachselt. 2012. Look & touch: gaze-supported target acquisition. In *Proceedings of the CHI Conference on Human Factors in Computing Systems*. ACM, 2981–2990.

[39] Shari Trewin. 2002. An invisible keyguard. In *Proceedings of the Fifth International ACM Conference on Assistive Technologies*. ACM, 143–149.

[40] Outi Tuisku, Päivi Majaranta, Poika Isokoski, and Kari-Jouko Räihä. 2008. Now Dasher! Dash away!: longitudinal study of fast text entry by eye gaze. In *Proceedings of the 2008 Symposium on Eye Tracking Research & Applications*. ACM, 19–26.

[41] Boris Velichkovsky, Andreas Sprenger, and Pieter Unema. 1997. Towards gaze-mediated interaction: collecting solutions of the "Midas touch problem". In *Human-Computer Interaction (INTERACT)*. Springer, 509–516.

[42] Brian A Wandell. 1995. *Foundations of vision*. Sinauer Associates.

[43] David J Ward and David JC MacKay. 2002. Artificial intelligence: fast hands-free writing by gaze direction. *Nature* 418, 6900 (2002), 838–838.

[44] Colin Ware and Harutune H Mikaelian. 1987. An evaluation of an eye tracker as a device for computer input. In *ACM SIGCHI Bulletin*, Vol. 17. ACM, 183–188.

[45] Wayne Westerman, John G Elias, and Alan Hedge. 2001. Multi-touch: a new tactile 2-d gesture interface for human-computer interaction. In *Proceedings of the Human Factors and Ergonomics Society Annual Meeting*, Vol. 45. SAGE Publications, 632–636.

[46] Victoria Woods, Sarah Hastings, Peter Buckle, and Roger Haslam. 2002. *Ergonomics of using a mouse or other non-keyboard input device*. Health and Safety Executive.

[47] Shumin Zhai, Carlos Morimoto, and Steven Ihde. 1999. Manual and gaze input cascaded (MAGIC) pointing. In *Proceedings of the CHI Conference on Human Factors in Computing Systems*. ACM, 246–253.

[48] Xinyong Zhang, Xiangshi Ren, and Hongbin Zha. 2008. Improving eye cursor's stability for eye pointing tasks. In *Proceedings of the CHI Conference on Human Factors in Computing Systems*. ACM, 525–534.