Swap: A Replacement-based Text Revision Technique for Mobile Devices

Yang Li[1], Sayan Sarcar[2], Sunjun Kim[3,4], Xiangshi Ren[1]
[1]Kochi University of Technology, Japan; [2]University of Tsukuba, Japan;
[3]Aalto University, Finland; [4]DGIST, Republic of Korea

ABSTRACT
Text revision is an important task to ensure the accuracy of text content. Revising text on mobile devices is cumbersome and time-consuming due to the imprecise caret control and the repetitive use of the backspace. We present *Swap*, a novel replacement-based technique to facilitate text revision on mobile devices. We conducted two user studies to validate the feasibility and the effectiveness of Swap compared to traditional text revision techniques. Results showed that Swap reduced efforts in caret control and repetitive backspace pressing during the text revision process. Most participants preferred to use the replacement-based technique rather than backspace and caret. They also commented that the new technique is easy to learn, and it makes text revision rapid and intuitive.

Author Keywords
Text revision, mobile device, virtual keyboard, backspace, caret control.

CSS Concepts
• **Human-centered computing~Human computer interaction (HCI)**; *Interaction Design,* Interaction design process and methods

INTRODUCTION
Text revision is a ubiquitous and vital process for text-related tasks such as email composition and instant messaging on mobile devices. Typo free contents contribute to both formal and informal scenarios to help the reader convey information accurately. Currently, virtual keyboard interfaces offer backspace and caret control tools for erasing characters and navigating the caret. With the help of automatic techniques (e.g., spell checker [1, 5] and input guidance [26]) and writing assistant software (e.g., Grammarly [25]), users can also avoid, detect, and/or correct typos and grammar mistakes.

Despite many existing techniques, users still face significant usability issues when revising text on mobile devices. First, navigating the caret with the finger is error-prone and time-consuming, due to finger occlusion [36] and small target size [13, 16]. Second, auto-correction sometimes introduces confusing, embarrassing, and hard-to-observe errors [1, 7, 11, 32], that diminish the benefits of those techniques and introduce extra revision workload to users [14]. Third, the character-level backspace makes the revision process time-consuming and error-prone when the revision contains multiple characters and/or happens in the middle of an input string [10]. Other than the backspace and caret, there lacks efficient tools (and methods) when revising the sentence itself (e.g., modifying the meaning of the sentence by inserting, deleting, or substituting word(s)).

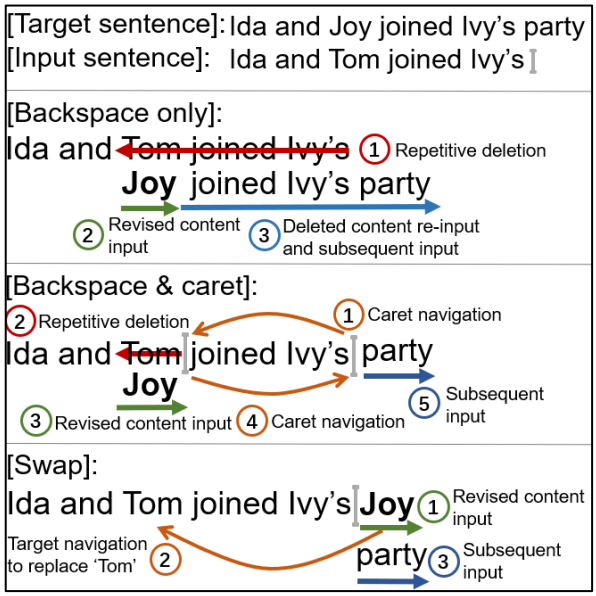

Figure 1. Operation sequences (shown with enclosed numbers) for different techniques when revising the sentence. Swap minimizes the use of backspace and caret by first entering the revised content and then using it to replace the target.

In this paper, we present Swap, a novel technique that is designed to facilitate text revision on mobile devices by using a unified replacement-based process to address various text revision tasks (e.g., substituting/deleting/inserting a word). Instead of using backspace and caret repetitively, Swap allows users to do the revision by first entering the correct content, and then using it to replace the target (see Figure 1). Swap 1) reduces the time consumption and the workload when using the backspace, and 2) avoids moving the caret in order to ensure the quick recovery from revision to the regular text entry process. With Swap, users can focus

CHI '20, April 25–30, 2020, Honolulu, HI, USA.
 ACM 978-1-4503-6708-0/20/04...$15.00.
DOI: https://doi.org/10.1145/3313831.3376217

on the revision itself rather than spending time on auxiliary steps.

The paper is organized as follows. First, we conducted a workshop to investigate the pain points of the current text revision interaction on mobile devices. After that, we illustrate the design of the Swap. Then, we make the first implementation (Dot Swap) to validate the feasibility of Swap with a user study. Based on that study, we further improve the design, propose an iterative implementation (improved Swap), and evaluated its effectiveness through a user study compared with the conventional backspace and caret. Finally, we discuss Swap technique insights and future work.

RELATED WORK

In this section, we first review previous research on text error correction, including the tool (backspace and caret) improvement, and current solutions (techniques, algorithms, and commercial products) for enhancing users' error correction performance. Then, we make a summary based on the review to reveal the gap between error correction and text revision.

Text Revision Procedure on Mobile Devices

Generally, the design of text revision on current mobile devices obeys Object-Action model [15, 33]: first, users need to navigate the caret to the target position, and then finish the revision with repetitive backspace pressings (and extra caret control). As text revision often happens during a text entry process, users usually have to relocate the caret for other operations (another text revision or the following text entry). As a result, we summarize the current text revision procedure as "navigate the caret, revise the target, and navigate the caret".

Improvement of the Backspace

The conventional backspace serves users with the character-level function of both deleting a character and moving the caret backward [40]. Researchers attempt to use hotkeys [34] and touch pressure [2] to extend the deletion granularity of the backspace and decrease repetitive use of the backspace.

Gestures are often used to facilitate error correction in both research [21] and industry field [20]. Smart-restorable backspace [8] allows users to quickly delete and store the text after the typo position by swiping left on the backspace key. A similar solution also occurs in [1]. Instead of only having one backspace key on the keyboard, Arif et al. [9] removed the backspace key and integrate its function to the whole keyboard. Users can swipe left on any key on the keyboard to trigger the deletion.

Caret Control Enhancement

Caret control on mobile devices has been well investigated by researchers. Caret is an indicator for both mobile devices and users to remind them where interactions may happen. The trackpad mode on a virtual keyboard [27], magnifier [22] (an assistive widget used in Android and iOS), and selection handle [23] can help users navigate the caret with flexibility and precision. iOS 13 [6] uses the mouse to navigate the caret on tablets. However, those tools introduce extra interaction steps, cognitive load, and the possibility of increased errors.

Users can navigate the caret in both direct and indirect ways. Widgets such as virtual stick [35] and arrow keys [24, 32] are integrated into virtual keyboards to simulate the caret control on physical keyboards. By holding one key on the keyboard, Ando et al. leveraged the device tilt [3] and slide gesture [4] to complete the caret control, target selection, and a particular command (e.g., copy) at the same time. Eady et al. [19] built a deformable interface and proposed a bend gesture on the corner of the device to control the caret. Sindhwani et al. [37] used a matching algorithm to highlight potential correction position(s) based on users' input. Then, eye movement was used to identify the intended correction position. Instead of moving the caret, Suzuki et al. [38] attempted to move the caret by moving the background text.

Error Correction

Algorithms and AI-based methods (e.g., natural language processing) are widely used in current typing applications: 1) to detect typos and grammar mistakes and 2) to prevent those issues from appearing in the final text [12]. Arif et al. [8] used Levenshtein distance [30] to calculate the nearest position for their smart error correction technique. Retype [37] used a simple string matching method to filter the potential correction positions by entering a few characters. Variants of visual feedback were used to inform users of detected problems. Maxie Keyboard [29] and Wisetype [1] used colored shades to highlight typos. Grammarly [25] used underlines to remind users of detected errors. Users can do a quick correction by tapping on the error and select a required option provided by the system. Additionally, applications such as Microsoft Word and Gmail also provide auto-correction functions. These systems automatically correct typos as the users tap the space key after entering a word.

Summary

To date, extensive research mentioned above were focused on addressing typos and grammar mistakes. Most of those errors can be solved (semi-)automatically with corpora, machine learning, and auto-correction techniques. However, it may be noted that typos and grammar mistakes are not the only targets for text revision. Other than backspace and caret, no efficient tool deals with conditions such as revising words with unintended meaning, or changing unintended and/or confusing words inserted by incorrect auto-correction [8]. Furthermore, through relevant literatures, we found that most researches were focused on only one part (e.g., caret control) of the text revision procedure rather than questioning the text revision procedure itself as well as the rationality of the current procedure and tools used in mobile devices.

PRE-STUDY: HOW DO USERS DO TEXT REVISION

Before designing the new text revision technique, we conducted a workshop to investigate strategies of using the backspace and caret in daily text revision scenarios and to explore the challenges when revising text content on mobile

devices. All user studies conducted in this work were approved by the Ethics Review Committee of the university. The workshop involved 20 participants (ten females, ten males, ages ranged from 24 to 32 years), all of whom have used smartphones for over 6 years. Seven of them currently use iOS smartphones, the others use the Android smartphones. All participants are familiar with both systems. All participants used instant messaging applications (e.g., Line and WhatsApp) every day. Eight of them compose emails daily on their smartphones. We designed several text revision scenarios (e.g., changing a word in the middle of a sentence when composing an email) and asked participants to simulate them on their smartphones. After that, we organized a free discussion to collect their feedback on the use of backspace and caret with a view of improving our design.

Most (18 of 20) participants preferred to use two thumbs for typing on the virtual keyboard. All participants used only backspace for quick revisions if there were only a few characters away from the caret. When editing was required in the middle of the sentence, three participants tended to use backspace only, while the others navigated the caret, executed the correction, and then navigated the caret again to other positions for further actions. When navigating the caret, 10 participants used assistive techniques (e.g., selection handles and the magnifier). All participants reported that it is difficult to navigate the caret with their fingers due to the occlusion.

In the free discussion session, most participants regarded the text revision process as "*using the proper content to **replace** the improper one*". They commented that it was simple but cumbersome to perform that on mobile devices. They felt it "simple" because they had used backspace and caret for years. Participants subconsciously turned the intention to revise into a sequence of steps. They felt the process "cumbersome" because of the limitations of backspace and caret. Participants must spend time on extra but essential steps (e.g., caret control and consecutive deletion) to revise the text. All participants commented that frequent caret control interrupted their typing flow. They must stop typing, move their fingers out of the virtual keyboard to navigate the caret, finish the correction, and restart typing after that (with navigating the caret once more). This situation became worse if participants have multiple words to correct.

THE DESIGN OF SWAP

Based on the pre-study, we regard "replacement" as the core design principle to simplify the mobile text revision process. There are two ways to implement the replacement process. One is "type and then select", while the other is "select and then type". We chose the first one to enable participants to enter revision content quickly without the obvious interruption caused by the selection process.

We took the view that "replacement" can happen not only between words (e.g., replace "*John*" with "*Sam*" in "*John will come and visit Bob*") but also between functions and words (e.g., deleting "*almost*" by using a symbol (representing the delete function) to replace "*almost*" in "*I am almost ready to go there*").

However, it is difficult to realize the aforementioned "replacement" in current conditions because: 1) the current backspace and caret control methods cannot provide a unified procedure to deal with different revision conditions (e.g., inserting, deleting, or substituting a word); 2) the current backspace function cannot appear on the screen as a regular character. Thus, we designed the replacement-based text revision process and symbolized the backspace as a character to make it visible on the screen.

Replacement-based Text Revision Process

Swap changes the revision into two steps: *Content Preparation* and *Replacement Execution*. With them, Swap unifies various revision procedures into one: enter the content and then replace the target with it.

Content Preparation: The first step of revision is to enter the content for the replacement. The content includes characters and symbolized backspace (this will be elaborated later in the paper).

Replacement Execution: After the content preparation, the following step is to use the entered content to finish the revision. Specifically, participants can navigate the content and finish the replacement by tapping the target. After the replacement, the caret remains at the end of the input string and ready for further input.

Taking the phrase "*… text revision tools show vital importance in text entry tasks.*" as an example, where users attempt to:

1) delete "*tools*", they first enter a symbolized backspace, then use it to replace "*tools*";

2) substitute "*tools*" to "*methods*", they first enter "*methods*", then use it to replace "*tools*";

3) insert "*mobile*" between "*in*" and "*text*", they first enter "*mobile*", then use it to replace the space between "in" and "*text*".

Symbolized Backspace

Compared with the conventional backspace, we visualize the function of deletion on the screen with the symbolized backspace. The symbolized backspace represents the function of "deleting multiple characters". When participants want to delete one word, they can enter a symbolized backspace, and then use it to replace the word. It may be noted that the symbolized backspace will disappear after the replacement and it will not appear in the final text.

DOT SWAP

Two issues are needed to solve before implementation. First, we need a suitable symbol to represent the function of "deleting multiple characters". Second, we need an approach that helps identify different revision intentions (e.g., inserting or deleting a word) during the revision process. It

should also be noted that, when entering the symbol, it should not bring much extra cost to participants (e.g., mode switch, visual search). We addressed those issues by redesigning the use of the dot/period (".") because it is the common symbol directly shown on the default virtual keyboard. To 1) validate the feasibility of the new text revision process and the symbolized backspace and 2) improve the design of Swap, we implemented Dot Swap and evaluated it with a user study.

Dot Swap used a dot as both the symbolized backspace and the identifier. As shown in Figure 2, when participants observed the target for revision, they would first enter the content at the end of the sentence. If participants entered only one dot, this dot would be regarded as the symbolized backspace. If the entered content didn't contain any dot at the beginning, it can be used to substitute the target word. Content marked with two dots as a prefix meant that the content will be inserted into the sentence. After entering, participants tapped the intended target (when inserting a word, Dot Swap calculated the nearest space to the touch point) to finish the revision. If participants want to enter a period, they can type the dot and press the "enter" button to leave it at the end of the string.

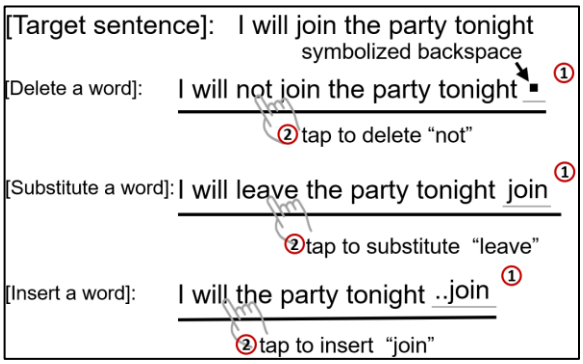

Figure 2. The use of Dot Swap under three revision conditions. Participants first enter the content at the end of the string, then use that content to revise the target.

STUDY 1 - FEASIBILITY EVALUATION OF DOT SWAP

We conducted a user study to 1) examine the feasibility of the replacement-based process for text revision and 2) compare the text revision efficiency of Dot Swap with the conventional backspace & caret and magnifier.

Apparatus

A custom application was designed to implement the text revision techniques with Android P (SDK version 28) in Java on a Huawei P20 smartphone (5.8 inches, 2244 × 1080 pixel, 149.1 × 70.8 × 7.65 mm). The smartphone recorded participants' keypress and caret control events during the experiment. We also implemented a custom virtual keyboard without the functions of auto-correction and auto-completion.

Participants

We recruited nine participants (three females, six males, average age 24.67 years, *SE*=1.15) for this study. One participant is left-handed. All participants had experience over three years of using mobile devices and preferred to use two thumbs when typing on the touch screen.

Procedure and Task

The study used a within-subjects design to compare text revision performance of conventional backspace & caret (as the baseline), magnifier [22] (when participants move the caret, a virtual lens appears and follows the finger position with the content below the fingertip), and Dot Swap. We used the memorable test set from the Enron Mobile Email Dataset [39].

We set two phases for each trial to simulate the revision behavior during the text entry process (e.g., sending short messages). In the first phase, part of the target sentence appeared on the screen (see Figure 3a). Participants were requested to read and enter the sentence part as fast and accurate as possible. Error correction in the first phase was mandatory. Then, participants pressed the "enter" key on the keyboard to trigger the second phase.

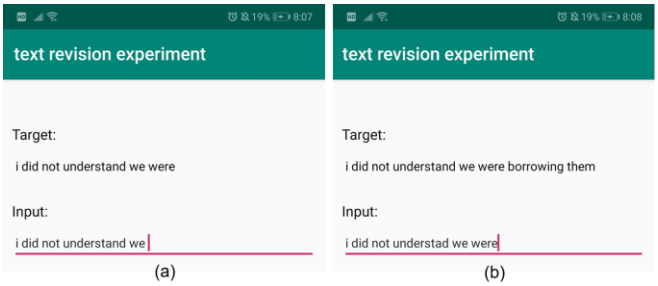

Figure 3. The experimental interfaces used for study 1. (a) 1st phase of the trial. (b) 2nd phase of the trial.

In the second phase, the system showed the rest part of the target sentence (see Figure 3b). As instructed in [8], we injected an error randomly in the front, middle, or end of the participant's transcription (in real-life scenarios, text revision usually happens unexpectedly during typing). Participants were requested to correct the error and finish the transcription as fast and accurate as possible. After that, participants submitted the input by pressing the enter button (and then started a new trial). The injected error had two main types: character-level and word-level. Each included the following subtypes: insertion, substitution, and omission. The distribution of each type of error is shown in Table 1 (distribution of character-level errors followed the statistical results from [18]).

	Character-level	Word-level
Insertion	21%	33%
Substitution	53%	33%
Omission	26%	33%

Table 1. Proportions for various types of errors inserted during the experiment.

We counterbalanced the order of the techniques across participants using Latin Square. For investigating the

potential learning effect, three blocks of trials were completed by each participant over three consecutive days (with an approximate time gap of more than 24 hours). Every day, participants needed to finish one block of trials for each technique (3 blocks a day). Each block consisted of 24 target sentences randomly chosen from the corpus.

Participants practiced sufficiently with all techniques. During the study, participants were seated in front of the desk in comfortable postures. All participants used two hands to hold the smartphone and typed on the virtual keyboard. Participants got a 5-minute rest between blocks. As a result, the total number of trials was:

9 participants × 3 techniques (conventional backspace & caret, magnifier, Dot Swap) × 24 sentences × 1 block/day × 3 days=1944.

IVs and DVs

The independent variables in this study were *technique* and *error type* (character-level, word-level). We evaluated text revision performance with the following dependent variables: 1) *correction time* (duration between the first action and the final action when revising the target, measured in milliseconds (ms)), 2) *number of caret control operations* (for Dot Swap, content navigation was regarded as the caret control operation), 3) *caret control time* (duration of the finger navigating the caret or the content), 4) *number of backspace keystrokes*, and 5) *backspace time* (duration between the previous keystroke and the backspace keystroke). For Dot Swap, the behavior of navigating the content for replacement was also calculated as the number of caret control operations and caret control time.

Results

We did the log-transformation operation and validated the data normality of correction time, caret control time, and backspace time. Further, we performed the repeated measures ANOVA on those DVs (α=0.05, post-hoc tests with Bonferroni correction). For the number of caret control operations and the number of backspace keystrokes, data did not satisfy the normality, thus we performed the Friedman test and Wilcoxon Signed-Rank test.

The average correction time for conventional backspace & caret, magnifier, and Dot Swap (same order hereinafter) were 7158 ms (*SE*=323.45), 7069 ms (*SE*=313.38), and 6441 ms (*SE*=381.81), respectively. An ANOVA showed a significant effect of technique ($F_{2,16}$=8.21, p<.001, η_p^2=0.03), error type ($F_{1,8}$=81.63, p<.001, η_p^2=0.21) and technique × error type ($F_{2,16}$=5.18, p<.01, η_p^2=0.02) on the average correction time. A post-hoc showed that Dot Swap took significantly less correction time than the other two techniques (all p<.01).

For the average number of caret control operations, participants navigated the caret most times with the conventional backspace & caret (2.62, *SE*=0.12), then with the magnifier (2.27, *SE*=0.08). Dot Swap navigated the caret with the least number (1.29, *SE*=0.05). The Friedman test showed a larger effect of technique (χ^2(2)=536.05, p<.001) for the number of caret control operations. A post-hoc using the Wilcoxon Signed-rank test with Bonferroni correction showed that Dot Swap used significantly less caret control operations than the other two techniques (all p<.001).

The average caret control time for three techniques was 1813 ms (*SE*=103.72), 1754 ms (*SE*=122.8), and 1100 ms (*SE*=86.16), respectively. An ANOVA showed a significant effect of technique ($F_{2,16}$=186.72, p<.001, η_p^2=0.37) and error type ($F_{1,8}$=15.88, p<.001, η_p^2=0.05) on the average caret control time. The interaction effect of technique × error type ($F_{2,16}$=2.84, p=0.06, η_p^2=0.01) was not significant. A post-hoc showed that Dot Swap took significantly less caret control time than the other two techniques (all p<.001).

For the average number of backspace keystrokes, the conventional backspace & caret used the most 4.19 (*SE*= 0.17), then the magnifier 4.02 (*SE*=0.17). Dot Swap used the least number of backspace keystrokes 2.44 (*SE*=0.66) during the revision. The Friedman test showed a larger effect of technique (χ^2(2)=190.14, p<.001) for the number of backspace keystrokes. A post-hoc using the Wilcoxon Signed-rank test with Bonferroni correction showed that Dot Swap used significantly less backspace than the other two techniques (all p<.01).

The average backspace time for three techniques was 543 ms (*SE*=38.59), 773 ms (*SE*=76.1), and 693 ms (*SE*=163.8), respectively. An ANOVA showed a significant effect of technique ($F_{2,16}$=9.67, p<.001, η_p^2=0.03), error type ($F_{1,8}$=84.7, p<.001, η_p^2=0.21) and technique × error type ($F_{2,16}$=51.05, p<.001, η_p^2=0.14) on the average backspace time. A post-hoc showed that Dot Swap took significantly less backspace time than other the two techniques (all p<.001).

Discussion

Based on the results, the magnifier can help participants to decrease the effort when using backspace and caret. However, it mainly focused on improving the controllability of the caret instead of changing the text revision procedure. When revising character-level errors, all participants commented that instead of navigating the caret, they preferred to enter the whole word.

Though Dot Swap showed an overall reduction in the average correction time, it didn't achieve the improvement we expected. There were three reasons for that. First, participants needed time to memorize and adapt to the rules of using Dot Swap. Therefore, more cognitive effort was taken to decide the correction strategy and the following procedures during revisions. Three participants commented that sometimes there was a confusion entering the dot as the dot was often regarded as the 'period' symbol indicating the end of a sentence. Second, the text in the text area was small. Thus, due to the finger occlusion, it was difficult to navigate the content for the correction towards small targets (1 or 2 characters) with their fingertips. Third, when participants

made an unintended correction (to the wrong place) or the intended correction was misspelled, extra time and steps were required to recover from the compounded mistakes. This can lead to cascading errors.

IMPROVED SWAP

Based on results and concerns revealed in study 1, we made the following improvements in the iterative technique (named improved Swap).

First, we used a text buffer to receive and temporarily hold the entered content (see Figure 4). The text buffer is a widget commonly used in Chinese text entry methods (to hold pinyin strings) [31]. Here, the content entered by participants would first go into the text buffer instead of directly appearing at the end of the string as the final input. With the text buffer, the content would be: 1) used for revision (replace the target text) or 2) confirmed as the final input (e.g., by pressing the "enter" key on the keyboard). After either operation, the text buffer would be emptied for further input. In case of long-text revision (e.g., inserting a sentence), we set the text buffer size as 150 characters to satisfy most conditions (on average 70-100 characters for one sentence [17]).

Second, we designed an expanded layout to make words and spaces easy to select during the revision. After entering the content for revision, participants can tap the input string to trigger an expanded layout (see Figure 4). In the expanded layout, words and spaces near the tap position are turned into buttons. Users can press the corresponding button to replace its content with the input in the text buffer (e.g., replace the space with "join" in Figure 4). After that, the expanded layout disappeared with the caret remained at the end of the input string with the text buffer emptied. If participants triggered the expanded layout in the wrong place, they can press the "CANCEL" button to revoke the expanded layout. If there was no content in the text buffer when tapping the input string, the expanded layout would not be triggered.

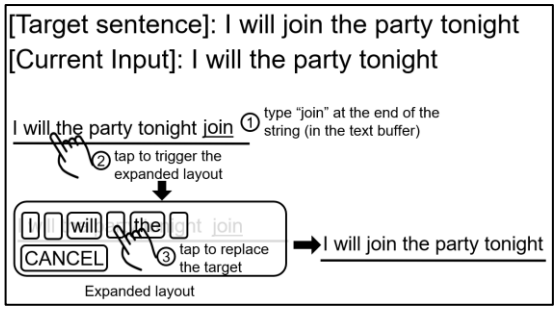

Figure 4. The schematic of using the improved Swap. The enclosed numbers refer to the operation steps.

Last, we integrated the symbolized backspace on the virtual keyboard as a button. When participants pressed the symbolized backspace, an emoji (a red cross, see Figure 5b) appeared at the end of the text.

In summary, when using improved Swap, participants first entered the content at the end of the text (into the text buffer), tapped the target position to trigger the expanded layout, and then pressed the target button in the expanded layout to finish the revision (replacement).

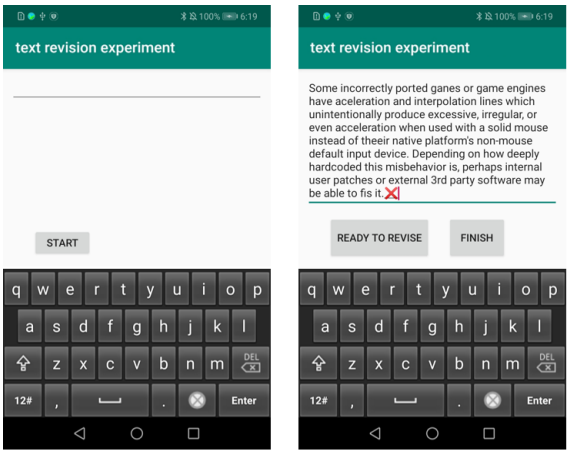

Figure 5. The experiment interfaces for Study 2. (a) is the interface before the experiment. (b) is the interface for the experiment. The symbolized backspace key is on the left side of the "Enter" key. When participants enter the symbolized backspace, a red cross appears on the screen.

STUDY 2 - COMPARATIVE USER STUDY

We conducted a user study to examine the performance of improved Swap and compared its text revision efficiency with the conventional backspace & caret. In this study, participants were requested to revise paragraphs, which contained multiple words to correct. In addition to the number and time consumption when using the backspace and navigating the caret, we also explored participants' entry speed with different text revision techniques. Furthermore, we collected participants' feedback and preferences regarding those techniques.

Apparatus

We used the same apparatus in Study 1 with a custom application for the paragraph revision task. For the virtual keyboard, we added a symbolized backspace (see Figure 5).

Participants

Eighteen participants (eight females, ten males, average age 26.33 years, *SE*=0.9, one left-handed) participated in the study. Before the experiment, we asked all participants to do simple typing speed tests on their smartphones (https://10fastfingers.com/). The average typing speed was 23.94 WPM (*SE*=1.79). No participant had physical problems with their hands or eyes. All participants had at least 3 years of experience in using mobile devices. None of the participants was native English speaker. And, none of them knew the replacement-based text revision techniques before. Each participant received a voucher equivalent to $10 as compensation.

Procedure and Task

To further validate the capacity of the replacement-based text revision process in case of heavy revision conditions, we

used a paragraph revision task in this study. For each technique, participants were given six paragraphs (with eight revision targets in each paragraph) to revise. All paragraphs were selected from Wikipedia. The error types and their distribution are shown in Table 1.

The study lasted about 70 minutes for each participant. Participants first signed the informed consent form. Then we demonstrated two techniques to participants and asked them to adjust themselves into a comfortable sitting position in front of the desk. Then, participants can get sufficient practice to ensure that they were familiar with the techniques. After that, participants were requested to revise each paragraph with the assigned technique as fast and accurate as possible with two thumbs.

When revising the paragraph, we provided the corresponding printout with all revision targets marked on it to the participant, because 1) it is difficult for them to detect word-level errors (e.g., a word without typos but fails to express the participants' intention) and 2) we mainly focus on investigating the revision efficiency of different techniques, not the participants' text comprehension ability. With marked targets on the printout, participants could reduce the effort of observing targets during text revision.

The experiment began when the participant pressed the "START" button (see Figure 5a). Then the paragraph (with revision targets) appeared on the touch screen (see Figure 5b). We requested participants to revise all targets. For each target, when the participant observed it and prepared to revise, s/he needed to press the "READY TO REVISE" button and then start the revision. When all targets were corrected, s/he pressed the "FINISH" button to finish the revision of the given paragraph. No help was provided from the experimenter during the experiment.

Participants could get enough rest between paragraphs. After revising all paragraphs, we asked participants to fill in the NASA-TLX and a 7-point Likert Scale questionnaire (1 for the less, 7 for the most, the lower, the better) for subjective evaluations on factors including complexity, fatigue, difficulty, and the dislike for the two techniques.

We used the within-subjects design in this study. The order of the paragraphs and the techniques for each participant was totally randomized. The total number of revisions in this study was:

18 participants × 2 techniques (conventional backspace & caret, improved Swap) × 6 paragraphs × 8 revision targets per paragraph=1728.

IVs and DVs

The independent variables in this study were *technique* (conventional backspace & caret, improved Swap) and *error type*. Error type included character-level ("C") and word-level ("W"). Each had three subtypes: insertion ("i"), substitution ("s"), and omission ("o").

Except for calculating the number of caret control operations, caret control time, and number of backspace keystrokes as in Study 1, we also calculated the following metrics:

1) *Keystroke per minute*: the number of keystrokes during the revision process divided by the input time.

2) *Pre-action time*: the time taken to detect the revision target and plan the sequence of actions. It is calculated as the time from the moment that the previous target is revised to the moment the "READY TO REVISE" button is pressed.

3) *Action time*: the time taken to do the actual revision (after pressing the "READY TO REVISE" button). It is the duration between the first and the last action required for the actual revision of the target.

4) *Navigation time*: for improved Swap, it is the total time taken to trigger the expanded layout, press the corresponding button for replacement, and press the "CANCEL" button if the participant mis-triggered the layout; for conventional backspace & caret, it is calculated as the total time taken to control the caret.

5) *Input time*: the time taken to enter characters (including the symbolized backspace).

Results

Sixty-seven (3.88%, sixty-four outliers, and three missing data) revision data was removed before the quantitative analysis. We did the log-transformation operation and validated the data normality for DVs in Study 2. Further, we performed the repeated measures ANOVA (α=0.05, post-hoc tests with Bonferroni correction). For the number of caret control operations and the number of backspace keystrokes, data did not satisfy the normality, thus we performed the Wilcoxon Signed-Rank test (also for the subjective evaluation) and Friedman test.

Keystroke per Minute

For each correction, the average keystroke per minute for conventional backspace & caret and improved Swap were 87.43 (SE=1.39) and 124.55 (SE=2.91), respectively. An ANOVA identified a significant effect of technique ($F_{1,17}$=116.07, p<.001, η_p^2=0.55), error type ($F_{5,85}$=6.79, p<.001, η_p^2=0.07), and their interaction ($F_{5,85}$=76.15, p<.001, η_p^2=0.44) on average keystroke per minute.

Time Consumption During Correction

For each correction, the average pre-action time for conventional backspace & caret and improved Swap were 3616 (SE=125.77) and 3042 (SE=154.66) ms, respectively. An ANOVA identified a significant effect of technique ($F_{1,17}$=18.16, p<.001, η_p^2=0.16) on the average pre-action time. The effect of error type ($F_{5,85}$=2.08, p=0.07, η_p^2=0.02) and technique × error type ($F_{5,85}$ =1.17, p=0.32, η_p^2=0.01) was not significant. Figure 6 illustrates the details.

For each correction, the average action time for conventional backspace & caret and improved Swap were 7420 (SE=143.4) and 6343 (SE=145.99) ms, respectively. An ANOVA

identified a significant effect of technique ($F_{1,17}$=27.53, p<.001, η_p^2=0.22), error type ($F_{5,85}$=24.44, p<.001, η_p^2=0.2) and their interaction ($F_{5,85}$=21.47, p<.001, η_p^2=0.18) on the average action time. The pairwise comparison revealed that all three word-level errors were different from each other (all p<.001). There was no significant difference among the three character-level errors. Ws also showed significant differences from all other error types (all p<.001). See Figure 7 for details.

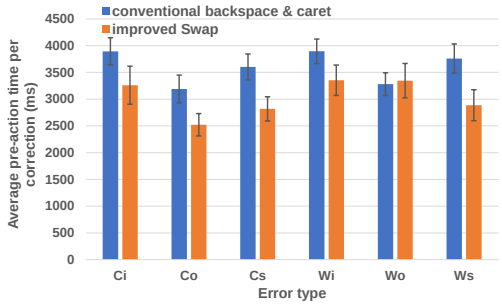

Figure 6. Average pre-action time for different error types.

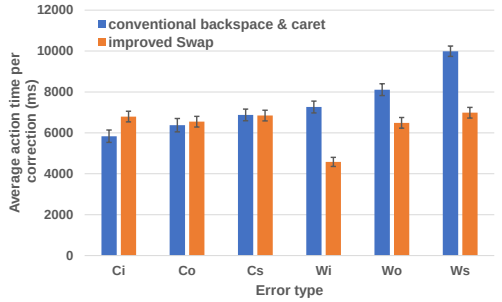

Figure 7. Average action time for different error types.

For each correction, the average navigation time and input time for conventional backspace & caret and improved Swap are shown in Figure 8. No significant effect of technique was found on average navigation time ($F_{1,17}$=2.21, p=0.14, η_p^2= 0.02). An ANOVA found a significant effect for error type on average navigation time ($F_{5,85}$=8.13, p<.001, η_p^2=0.08). For average input time, an ANOVA identified a significant effect of technique ($F_{1,17}$=125.63, p<.001, η_p^2=0.57), error type ($F_{5,85}$=43.4, p<.001, η_p^2=0.31) and their interaction ($F_{5,85}$=37.64, p<.001, η_p^2=0.28) on the average input time. Correcting the Ws targets with conventional backspace & caret took the longest average input time (M=6678 ms, SE=196.5), and while correcting the Wi targets, the improved Swap took the shortest average input time (M=1289 ms, SE=83.27).

Caret Control

For each correction, the average number of caret control operations for conventional backspace & caret and improved Swap were 2.61 (SE=0.1) and 1.04 (SE=0.01). Figure 9 shows the average number of caret control operations per correction for each error type. The Wilcoxon Signed-rank test showed a significant effect between the two techniques (W=16, Z=-18.65, p<.001, r=0.46). The Friedman test showed a significant effect among error types ($\chi^2(5)$=34.01, p<.001) for the number of caret control operations. A post-hoc using the Wilcoxon Signed-rank tests with Bonferroni correction showed significant differences between Ci and Cs, Co and Cs, and Cs and Ws (all p<.001).

For each correction, the average caret control time for conventional backspace & caret and improved Swap were 3297 (SE=102.09) and 2346 (SE=75.66) ms. Figure 10 shows the average caret control time when correcting different types of errors. An ANOVA identified a significant effect of technique ($F_{1,17}$=56.49, p<.001, η_p^2=0.37) and error type ($F_{5,85}$=6.21, p<.001, η_p^2=0.06) on the average caret control time. However, the interaction effect was not significant ($F_{5,85}$=1.65, p=0.15, η_p^2=0.02). Pairwise comparisons identified the significant difference between Co and Ci (p<.05), Wi (p<.05), Wo (p<.001), and Ws (p<.01).

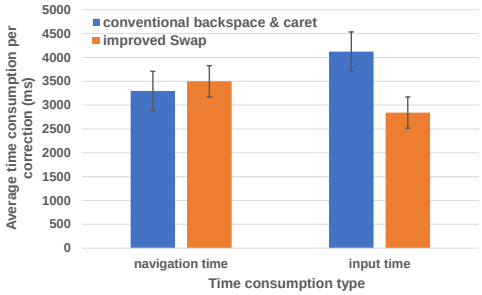

Figure 8. Average navigation time and input time for two text revision techniques.

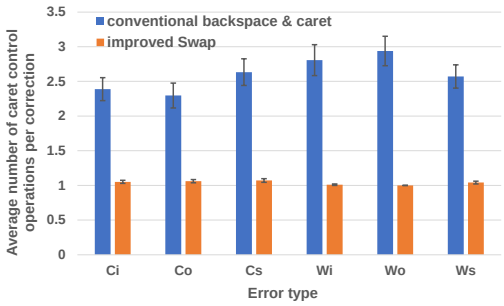

Figure 9. Average number of caret control operations for different error types.

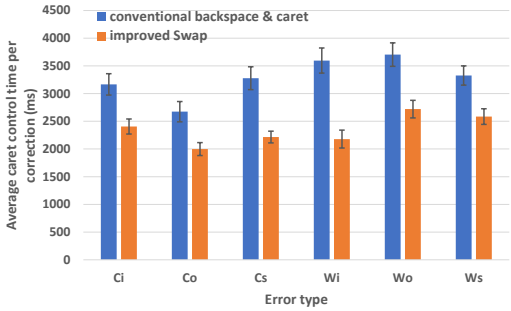

Figure 10. Average caret control time for different error types.

The Use of Backspace

For each correction, the average number of backspace keystrokes for conventional backspace & caret and improved Swap were 3.24 (*SE*=0.08) and 0.08 (*SE*=0.02) respectively. The Wilcoxon Signed-rank test showed a significant effect among techniques (W=17, Z=-21.75, p<.001, r=0.54). The Friedman test showed a significant effect among error types ($\chi^2(5)$=445.46, p<.001) for the number of backspace keystrokes. A post-hoc using the Wilcoxon Signed-rank tests with Bonferroni correction showed significant differences between all error type pairs (all p<.01).

Subjective Evaluations

For the weighted NASA-TLX rating scores (the lower, the better), improved Swap (M=45.3, SE=2.7) scored lower than the conventional backspace & caret (M=53.37, SE=3.84). However, a Wilcoxon Signed-Rank test did not find the significant difference between two techniques (W=6, Z=6, p=.05, r=0.33). For all factors measured in the Likert Scale (see Figure 11), improved Swap got lower scores than conventional backspace & caret. A Wilcoxon Signed-Rank test revealed that, for improved Swap, the score for fatigue (W=2, Z=-2.99, p <.01, r=0.5), difficulty (W=1, Z=-3.22, p<.01, r=0.54), and dislike (W=1, Z=-3.03, p<.01, r=0.51) was significantly lower than conventional backspace & caret.

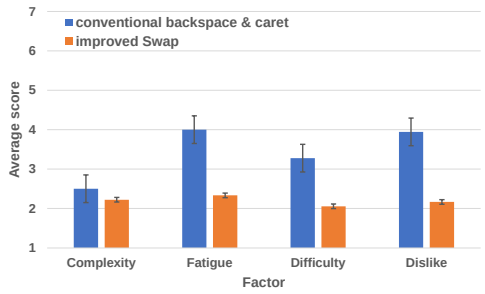

Figure 11. Average Likert Scale scores for each factor.

Overall, most participants gave positive responses to improved Swap (*P3:* "*It is easy to understand the logic of the new technique and to get used to it quickly.*", *P5:* "*I prefer to use the new technique because it frees me from using the backspace multiple times.*"). Eight participants described the experience of using improved Swap with the word "intuitive" (*P11:* "*The new technique liberates me from controlling the small caret. I can quickly type the content, locate, and finish the revision.*"). Five participants pointed out that improved Swap showed advantages, especially when revising word-level targets in long paragraphs. Six participants commented that, in a short period of time, it was hard to change to the new technique due to the years of experience with backspace and caret. Participants regarded that long-term use of improved Swap would change their habits of using backspace and caret. All participants agreed that improved Swap could decrease the effort required by navigating the caret and pressing the backspace repetitively. Six participants expect to integrate improved Swap into their smartphones.

Discussion

Based on results, improved Swap took less action time when correcting word-level errors, whereas it didn't show advantages when correcting character-level errors. The reason is that improved Swap did the replacement only in word-level. Thus, participants needed to type the whole word and then replace the target. Participants expressed their attitudes towards the character-level revision performance with improved Swap: "*For character-level errors, if the word length is within 5 characters, it would be fine to type the whole word and do the replacement. If the length exceeds that, it might cost extra time for revision. Despite that, I still tend to type because it is easier than navigating the caret.*"

In Study 2, we found that participants adapted two strategies to navigate the caret. One strategy was to persevere in locating the caret with repetitive caret control attempts until it arrived at the intended position. The other strategy was that when the caret was located before the intended position, the participant would re-navigate the caret; if the caret was located 2-3 characters after the intended position, the participant would use the backspace multiple times to delete to the intended place and then perform the revision. One participant gave a reason for adapting the second strategy: "*I usually avoid navigating the caret on the smartphone as the caret is too difficult to locate. Therefore, I would rather spend time deleting and retyping the content than controlling the caret multiple times.*" The expanded layout made it easier for participants to replace the word as 1) it enlarged the target size for easier selection and 2) it made the replacement intuitive and easy to understand.

Figure 8 showed that improved Swap took longer navigation time than the conventional backspace & caret. There are two reasons for that: 1) after typing the content for replacement, it still needs some time for participants to locate the target; 2) it is challenging for participants to fully accept a new technique in a short period of time. Though participants reported that they did sufficient practice, they still need time and effort to get used to the replacement operation.

With improved Swap, participants can type faster than the conventional backspace & caret during the text revision. The average keystroke per minute for improved Swap was similar to the average typing speed for all participants measured before the study. This indicated that improved Swap could leverage participants' regular typing speed to finish the text entry quickly during the revision.

We calculated the total time (sum of pre-action time and action time) to evaluate the time consumption when revising a target. Results proved that improved Swap took the less total time (M=9195 ms, SE=133.71) than the conventional backspace & caret (M=10654 ms, SE=114.38). For word-level errors, it took on average 9248 ms (SE=189.85) to finish the revision, while on average 11688 ms (SE=152.05) for the conventional backspace & caret. The reason for the difference was that the improved Swap minimized the use of

backspace and caret, and thus decreased the time consumption.

Figure 9 revealed that participants were more likely to navigate the caret multiple times when using conventional backspace & caret for revision compared with improved Swap. Through the data, we found 499 revisions navigated the caret more than once with the conventional backspace & caret. For improved Swap, only 35 revisions were found to trigger the expanded layout more than once. It indicated that selecting a target in the expanded layout was easier than navigating the small caret.

GENERAL DISCUSSION

Swap simplifies the caret control when redesigning the text revision interaction on mobile devices to focus on the revision task itself. This produces the following benefits to text revision efficiency. First, Swap allows participants to enter content for revision as they observe the revision target (with entry speed similar to that of regular text input). This is less disruptive to the flow of mind and input because participants can quickly turn their revision intention into real actions without breaking their focus in order to navigate the caret. Second, Swap turns high-precision caret control between characters into target selection with the expanded layout, which reduces the accuracy requirements of the caret control.

Swap provides a new perspective ("replacement") to interpret the text revision task and to unify various text revision conditions into "type first and then replace". With the symbolized backspace and the expanded layout, Swap visualizes all contents (words, spaces between words, and the symbolized backspace) during the revision and making the replacement intuitive and easy to understand by participants.

Swap improves the deletion efficiency with the symbolized backspace. This diminishes the time consumption and the number of repetitive backspace keystrokes when deleting multiple characters (e.g., correcting Wi errors). In addition, with the symbolized backspace, the conventional backspace could also reveal advantages in quick corrections (e.g. correcting the typo near the caret [28]).

Two user studies showed that Swap can handle both light (e.g., revision in one sentence) and heavy-load (e.g., revising a paragraph) revision conditions. It indicated that Swap could handle most daily text-related conditions, such as instant messaging and long text composition (e.g., email, online notes, blogs, etc.). The concept of replacement and the symbolized backspace also shed light on symbolizing (or visualizing) more commands for various text revision requirements (e.g., bold/italic/underline words, change font size/color, etc.).

To gather more user feedback after studies, we removed most control settings in the user study and asked participants to revise a paragraph with improved Swap. All participants commented that improved Swap could enhance efficiency and fluency when revising multiple targets. Five participants showed their expectations about integrating improved Swap with their own mobile devices such as iPads and laptops with multi-touch screens (e.g., Microsoft Surface).

Limitations and Future Work

Our current study mainly focused on the tool design and interaction design for text revision on mobile devices. Therefore, it encountered several limitations which inspire us to perform further evaluations and development. First, quantitative results and subjective evaluations validated the feasibility and the advantages of the replacement-based text revision techniques, whereas a long-term in-the-wild study is still needed to investigate text revision efficiency and changes in user behaviors and how to facilitate them efficiently.

Second, to make the replacement-based text revision technique compatible with the real-life scenarios, more factors could be considered in the following research, such as mobility (e.g., walking), workload (e.g., one-hand occupied), keyboard layout, and the size of the touchscreen.

Moreover, it should be noted that multiple intelligent text entry aid techniques and algorithms have been imbedded into current text input methods. These can also be integrated with Swap for quick and seamless text revision operations.

CONCLUSION

In this paper, we have presented Swap, a replacement-based technique to facilitate text revision on mobile devices. To simplify the text revision interaction, Swap designed the symbolized backspace and the replacement-based revision process. The Swap allowed users to input the content first and then use it for revision. We implemented two techniques (Dot Swap and improved Swap) and compared their text revision efficiency with the conventional text revision techniques through user studies. Results revealed that Swap improves revision efficiency and fluency by significantly reducing the use of backspace and caret during the text revision. Most participants showed their preference on Swap as it is easy to learn, and it is more efficient. Some participants expect to integrate Swap with their own mobile devices for long-term use.

REFERENCES

[1] Ohoud Alharbi, Ahmed Sabbir Arif, Wolfgang Stuerzlinger, Mark D Dunlop, and Andreas Komninos. 2019. WiseType: A Tablet Keyboard with Color-Coded Visualization and Various Editing Options for Error Correction. In Proceedings of Graphics Interface 2019 (GI 2019). https://doi.org/10.20380/GI2019.04

[2] Zaib Ali. 2017. Increase Text Delete Speed On Stock iOS Keyboard With 3D Touch. Retrieved from https://ioshacker.com/how-to/adjust-text-delete-speed-stock-ios-keyboard-3d-touch

[3] Toshiyuki Ando, Toshiya Isomoto, Buntarou Shizuki, and Shin Takahashi. 2018. Press & Tilt: One-handed Text Selection and Command Execution on Smartphone. In Proceedings of the 30th Australian

Conference on Computer-Human Interaction (OzCHI '18), 401–405. https://doi.org/10.1145/3292147.3292178

[4] Toshiyuki Ando, Toshiya Isomoto, Buntarou Shizuki, and Shin Takahashi. 2019. One-handed Rapid Text Selection and Command Execution Method for Smartphones. In Extended Abstracts of the 2019 CHI Conference on Human Factors in Computing Systems (CHI EA '19), LBW0224:1--LBW0224:6. https://doi.org/10.1145/3290607.3312850

[5] Android. 2017. Spell checker framework. Retrieved from https://developer.android.com/guide/topics/text/spell-checker-framework

[6] Apple. 2019. Caret navigation features in iOS 13. Retrieved from https://www.apple.com/ios/ios-13-preview/features

[7] Samuel Arbesman. 2017. Overcomplicated: Technology at the Limits of Comprehension. https://www.xarg.org/ref/a/0143131303/

[8] Ahmed Sabbir Arif, Sunjun Kim, Wolfgang Stuerzlinger, Geehyuk Lee, and Ali Mazalek. 2016. Evaluation of a Smart-Restorable Backspace Technique to Facilitate Text Entry Error Correction. In Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems (CHI '16), 5151–5162. https://doi.org/10.1145/2858036.2858407

[9] Ahmed Sabbir Arif, Michel Pahud, Ken Hinckley, and Bill Buxton. 2014. Experimental Study of Stroke Shortcuts for a Touchscreen Keyboard with Gesture-redundant Keys Removed. In Proceedings of Graphics Interface 2014 (GI '14), 43–50. http://dl.acm.org/citation.cfm?id=2619648.2619657

[10] Ahmed Sabbir Arif and Wolfgang Stuerzlinger. 2010. Predicting the Cost of Error Correction in Character-based Text Entry Technologies. In Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '10), 5–14. https://doi.org/10.1145/1753326.1753329

[11] Ahmed Sabbir Arif and Wolfgang Stuerzlinger. 2013. Pseudo-pressure Detection and Its Use in Predictive Text Entry on Touchscreens. In Proceedings of the 25th Australian Computer-Human Interaction Conference: Augmentation, Application, Innovation, Collaboration (OzCHI '13), 383–392. https://doi.org/10.1145/2541016.2541024

[12] Kenneth C Arnold, Krzysztof Z Gajos, and Adam T Kalai. 2016. On Suggesting Phrases vs. Predicting Words for Mobile Text Composition. In Proceedings of the 29th Annual Symposium on User Interface Software and Technology (UIST '16), 603–608. https://doi.org/10.1145/2984511.2984584

[13] Stephen Brewster. 2002. Overcoming the Lack of Screen Space on Mobile Computers. Personal Ubiquitous Comput. 6, 3: 188–205. https://doi.org/10.1007/s007790200019

[14] Daniel Buschek, Benjamin Bisinger, and Florian Alt. 2018. ResearchIME: A mobile keyboard application for studying free typing behaviour in the wild. In Conference on Human Factors in Computing Systems - Proceedings. https://doi.org/10.1145/3173574.3173829

[15] Stuart K Card, Allen Newell, and Thomas P Moran. 1986. The Psychology of Human-Computer Interaction. CRC Press. https://www.xarg.org/ref/a/B0074195FY/

[16] Herbert A Colle and Keith J Hiszem. 2004. Standing at a kiosk: Effects of key size and spacing on touch screen numeric keypad performance and user preference. Ergonomics 47, 13: 1406–1423. https://doi.org/10.1080/00140130410001724228

[17] Martin Cutts. 2013. Oxford Guide to Plain English (Oxford Paperback Reference). OUP Oxford. Retrieved from https://www.xarg.org/ref/a/B00FZSX76G/

[18] Vivek Dhakal, Anna Maria Feit, Per Ola Kristensson, and Antti Oulasvirta. 2018. Observations on Typing from 136 Million Keystrokes. In Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems (CHI '18), 646:1--646:12. https://doi.org/10.1145/3173574.3174220

[19] Alexander Keith Eady and Audrey Girouard. 2015. Caret Manipulation Using Deformable Input in Mobile Devices. In Proceedings of the Ninth International Conference on Tangible, Embedded, and Embodied Interaction (TEI '15), 587–591. https://doi.org/10.1145/2677199.2687916

[20] Fleksy. 2019. Fleksy keyboard. Retrieved from https://www.fleksy.com

[21] Vittorio Fuccella, Poika Isokoski, and Benoit Martin. 2013. Gestures and Widgets: Performance in Text Editing on Multi-touch Capable Mobile Devices. In Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '13), 2785–2794. https://doi.org/10.1145/2470654.2481385

[22] Google. 2018. Magnifier widget. Retrieved from https://developer.android.com/guide/topics/text/magnifier

[23] Google. 2018. Selection handle for Android Textview. Retrieved from https://developer.android.com/reference/android/widget/TextView.html#getTextSelectHandle()

[24] Google. 2019. Gboard. Retrieved from https://play.google.com/store/apps/details?id=com.google.android.inputmethod.latin

[25] Grammarly. 2019. Grammarly: free writing assistant. Retrieved from https://www.grammarly.com

[26] Toshiyuki Hagiya, Toshiharu Horiuchi, and Tomonori Yazaki. 2016. Typing Tutor: Individualized Tutoring in Text Entry for Older Adults Based on Input Stumble Detection. Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems - CHI '16: 733–744. https://doi.org/10.1145/2858036.2858455

[27] idownloadblog. 2018. How to use iOS 12 virtual keyboard in trackpad mode on iPhones without 3D Touch. Retrieved from https://www.idownloadblog.com/2018/08/22/howto-iphone-keyboard-trackpad-mode/

[28] Andreas Komninos, Mark Dunlop, Kyriakos Katsaris, and John Garofalakis. 2018. A Glimpse of Mobile Text Entry Errors and Corrective Behaviour in the Wild. In Proceedings of the 20th International Conference on Human-Computer Interaction with Mobile Devices and Services Adjunct (MobileHCI '18), 221–228. https://doi.org/10.1145/3236112.3236143

[29] Andreas Komninos, Emma Nicol, and Mark D Dunlop. 2015. Designed with Older Adults to SupportBetter Error Correction in SmartPhone Text Entry: The MaxieKeyboard. In Proceedings of the 17th International Conference on Human-Computer Interaction with Mobile Devices and Services Adjunct (MobileHCI '15), 797–802. https://doi.org/10.1145/2786567.2793703

[30] V I Levenshtein. 1966. Binary Codes Capable of Correcting Deletions, Insertions and Reversals. Soviet Physics Doklady 10: 707.

[31] Microsoft. 2018. Using an Input Method Editor in a Game. Retrieved from https://docs.microsoft.com/en-us/windows/win32/dxtecharts/using-an-input-method-editor-in-a-game

[32] Microsoft. 2019. SwiftKey: The Smart Keyboard. Retrieved from https://www.microsoft.com/en-us/swiftkey

[33] Donald A Norman. 1995. The Psychopathology of Everyday Things. Elsevier. https://doi.org/10.1016/b978-0-08-051574-8.50006-6

[34] Raymond. 2007. Whose idea was it to make Ctrl+Backspace delete the previous word. Retrieved from https://devblogs.microsoft.com/oldnewthing/20071011-00/?p=24823

[35] Jean-Baptiste Scheibel, Cyril Pierson, Benoît Martin, Nathan Godard, Vittorio Fuccella, and Poika Isokoski. 2013. Virtual Stick in Caret Positioning on Touch Screens. In Proceedings of the 25th Conference on L'Interaction Homme-Machine (IHM '13), 107:107--107:114. https://doi.org/10.1145/2534903.2534918

[36] Katie A Siek, Yvonne Rogers, and Kay H Connelly. 2005. Fat Finger Worries: How Older and Younger Users Physically Interact with PDAs. In Proceedings of the 2005 IFIP TC13 International Conference on Human-Computer Interaction (INTERACT'05), 267–280. https://doi.org/10.1007/11555261_24

[37] Shyamli Sindhwani, Christof Lutteroth, and Gerald Weber. 2019. ReType: Quick Text Editing with Keyboard and Gaze. In Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems (CHI '19), 203:1--203:13. https://doi.org/10.1145/3290605.3300433

[38] Kenji Suzuki, Kazumasa Okabe, Ryuuki Sakamoto, and Daisuke Sakamoto. 2015. Fix and Slide: Caret Navigation with Movable Background. In Adjunct Proceedings of the 28th Annual ACM Symposium on User Interface Software & Technology (UIST '15 Adjunct), 79–80. https://doi.org/10.1145/2815585.2815728

[39] Keith Vertanen and Per Ola Kristensson. 2011. A Versatile Dataset for Text Entry Evaluations Based on Genuine Mobile Emails. In Proceedings of the 13th International Conference on Human Computer Interaction with Mobile Devices and Services (MobileHCI '11), 295–298. https://doi.org/10.1145/2037373.2037418

[40] Wikipedia. 2019. Backspace. Retrieved from https://en.wikipedia.org/wiki/Backspace