Efficient Mid-Air Text Input Correction in Virtual Reality

John J. Dudley*
University of Cambridge

Amy Karlson
Reality Labs Research
Meta Inc.

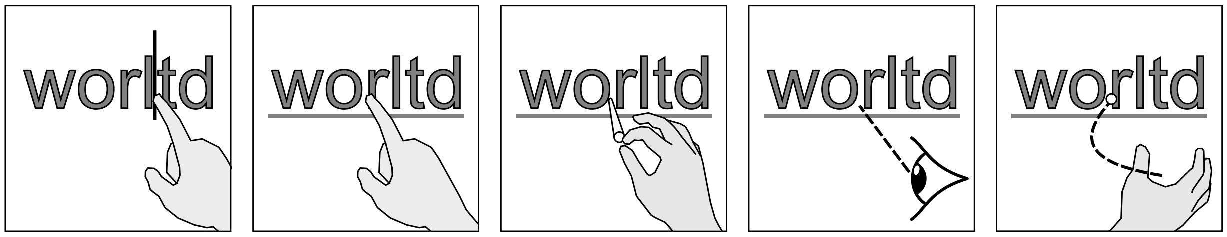
Kashyap Todi
Reality Labs Research
Meta Inc.

Hrvoje Benko
Reality Labs Research
Meta Inc.

Matt Longest
Reality Labs Research
Meta Inc.

Robert Wang
Reality Labs Research
Meta Inc.

Per Ola Kristensson
University of Cambridge



(a) Caret (b) Touch (c) Raycast (d) Gaze (e) Indirect

Figure 1: The five alternative interaction methods for text input correction in virtual reality that are investigated in this study. The Caret and Touch methods are familiar interactions closely related to established conventions for text input correction on a smartphone. Raycast, Gaze and Indirect are less familiar interactions but exploit the rich array of input available on modern head-mounted displays.

Abstract

The task of inputting text within virtual reality has attracted significant research attention over the last five years. Less well explored is the related task of correcting inputted text when errors are made. This is despite the fact that considerable time and frustration stems from efforts to correct text. In this paper, we bridge this gap in prior research and explore efficient methods for supporting text input correction in virtual reality. We present a characterization of the types and frequencies of errors encountered when inputting text in virtual reality and an analysis of effective editing strategies. We also present the results of a user study evaluating the performance and usability trade-offs for several interaction methods leveraging the unique capabilities of modern head-mounted displays.

1 Introduction

Mistakes are common in text input, regardless of the means of entry. Nevertheless, text entry research typically focuses on transcription tasks and sometimes provides no mechanism for, or actively discourages, error correction. Even more fundamentally, there is limited prior work on the types and frequencies of errors committed by users under different settings.

A wide range of text entry methods have been proposed and studied for use in virtual reality [6]. Considerably less well explored, however, are mechanisms compatible with these entry methods that support efficient correction of errors. In this paper, we address this gap by focusing on the task of correcting errors in text in a mid-air setting within virtual reality. This is a relevant and timely problem to investigate since recent work [10] has shown that entry rates for mid-air typing in virtual reality are reduced by up to 30% when input correction is required.

Text input systems on smartphones allow users to correct text via various methods. One method is derived from computer-based word processing applications and involves editing at a character level with a text caret. Another method involves interacting at the word or phrase level, by indicating the erroneous word or phrase and selecting corrections from a list of displayed alternatives. These same approaches can, in theory, be ported to text input systems in virtual reality to deliver a familiar interaction experience. However, these derived approaches do not fully exploit the additional input channels and display space afforded by modern virtual reality HMDs. In response to this point, we investigate what alternative methods are afforded by the unique capabilities of modern virtual reality hardware, and how do these methods trade-off between performance and usability in a mid-air interaction setting. We provide context to this investigation by first presenting a structured description of the text input correction task within virtual reality, and estimate the frequency and impact of input errors. Our analysis suggests that word-level error correction is likely to be more efficient than character-level error correction in a mid-air interaction setting. This motivates our primary focus on techniques that facilitate a word-level correction strategy as we consider potential instantiations from within the design space.

We implement and compare word-level correction interactions by means of touch, raycast, gaze and an indirect cursor, but also include a caret-based character-level correction interaction as an alternative baseline. These five techniques are illustrated in Figure 1. In a controlled user study with 20 participants, we find that word-level correction via a touch-based interaction provides the fastest method for correcting errors in a review correction task (where the task is simply to correct pre-existing errors). However, in an input correction task (where the task is to input text and correct any errors that occur) the character-level caret-based interaction method results in the shortest entry durations. In terms of overall participant preferences, the touch-based and gaze-based interaction methods were most preferred.

In summary, this paper makes the following contributions:

1. We characterize the types and frequencies of errors encountered when inputting text in virtual reality.

2. We evaluate and compare five alternative text input correction methods to deliver insight on the design of effective mid-air text input correction interactions in virtual reality.

*e-mail: jjd50@cam.ac.uk

2 Related Work

There has been considerable research effort devoted to understanding the principles for efficient and usable text entry within virtual reality (VR) and augmented reality (AR). The topic of text input correction in mid-air interaction settings is less-well explored in the literature. In this section we first briefly review prior work focused on text input in VR and AR, and then review work on text input error mitigation, and text input correction more broadly.

2.1 Text Input in VR and AR

A wide variety of interaction methods for text input in VR and AR have been proposed and studied [2, 9, 18, 25, 37, 43]. Dube et al. [6] offer an informative review. Most relevant to this current work are input methods that support efficient mid-air text entry on Qwerty layouts without the need for controllers [33–36]. The VISAR keyboard [9] was one of the first studies to examine mid-air touch typing on a Qwerty layout using integrated hand-tracking. Speicher et al. [37] compared the use of tracked hands to five other alternative selection-based text input methods in VR. The use of tracked hands was, however, outperformed by a ray-based method using two tracked controllers. Frutos-Pascual et al. [13] demonstrated a functional mid-air AR keyboard but their study focused on individual character selections and no entry rates are reported.

Using a simulated auto-correction functionality and high-precision hand tracking, Dudley et al. [8] demonstrated that mid-air entry rates without controllers could feasibly reach 40 wpm. In a subsequent study, Dudley et al. [10] demonstrated that headset-based hand tracking can enable users to achieve peak entry rates in the range of 40 to 45 wpm. This recent work highlights the potential of controller-free mid-air text input and the corresponding need for input correction techniques that are consistent with these methods.

2.2 Mitigating Errors During Mid-Air Text Input in VR

Mid-air text input is generally associated with lower entry rates and higher error rates than typing on a physical keyboard or smartphone [8, 10, 14, 26]. This has motivated work exploring methods that aim, in part, to reduce errors by giving better feedback to users or intelligently correcting errors. For example, the VISAR keyboard [9] evaluated a method for allowing users to preemptively designate key selections as unalterable by the auto-correction function in order to help users avoid auto-correction errors. This method was found to reduce error rates (although not significantly) but was associated with a marginal reduction in entry rates. The work by Foy et al. [12] sought to address an error type unique to mid-air typing caused by spurious touches generated by involuntary movement of fingers next to the finger intentionally performing a touch. Foy et al. [12] developed a component able to recognize such spurious touches that could be integrated with the keyboard input decoder in order to reduce error rates.

Offering better feedback is another way to potentially help users to regulate their behaviors in order to reduce errors. Gupta et al. [15] investigated different means of providing vibrotactile feedback in mid-air text entry and observed reduced, although not significantly so, error rates with an on-finger vibrotactile feedback configuration. To support efficient touch typing in a mid-air setting, a separate body of work by Gupta et al. [16] also investigated the use of 'squeeze haptics' to reinforce correct posture. The squeeze feedback provided by a band worn on each wrist was found to significantly reduce the need for expert typists to look at the keyboard, without any significant detrimental effect in terms of error rates. Dube et al. [7] evaluated the potential benefit of mid-air ultrasonic feedback and observed a significant effect associated with haptic feedback, both for increasing entry rates and reducing error rates. Frutos-Pascual et al. [13] investigated different visual feedback settings to assist with touching virtual keys in mid-air such as glow, a guiding ray and combining glow with a guiding ray but found no significant benefits in terms of improving the accuracy of key presses.

The various efforts reviewed above can be classed as preemptive or design focused efforts aimed at reducing errors during entry. Despite such efforts, error rates remain above zero highlighting the need for consideration of reactive methods for correcting errors when they occur. This is the focus of our current paper.

2.3 Text Input Correction

There has been some scattered work examining various aspects of text input correction, both in an immersive setting as well as on more conventional devices. Wobbrock's review [42] of text entry performance measures includes two measures directly related to the efficiency of error correction methods. Adhikary and Vertanen [1] and Vertanen and Kristensson [39] have a similar focus and approach in presenting methods to facilitate the correction of errors in speech-to-text input. Both works allow the user to interact with the word confusion network output by the speech recognition system, with Adhikary and Vertanen [1] presenting this interface in VR and Vertanen and Kristensson [39] using a smartphone. The user is presented with likely alternatives for the recognized words in the transcribed utterance and they can make corrections by directly selecting from these alternatives.

Hu et al. [17] and Li et al. [27] both examine caret navigation for error correction in AR and VR respectively. Hu et al. [17] evaluate different methods for navigating the caret within a body of text, including direct and indirect touch, raycast and gaze as well as leveraging a magnifier to assist with positioning. The task evaluated by Hu et al. [17] simply involved placing the caret at an indicated target location, and no actual edits were made. Li et al. [27] instead focus on interactions supported by a controller but do also examine these interactions within the context of a text correction task. The best performing interaction method allowed users to backspace at the word-level and move the caret in a continuous rather than discrete manner.

The majority of recent work on enhancing correction interactions has focused on text error correction on smartphones. Cui et al. [4] demonstrates the concept of simply retyping an erroneous word and relying on an inference step to determine which word in the previously typed text should be replaced. Zhao et al. [45] exploit gaze and speech-to-text to allow users to look at an erroneous word and then re-speak this word to insert a correction. This builds on prior work from Zhao et al. [44] where the approximate error location is indicated by touching on the screen. Proactive approaches seeking to fuse speech and keyboard input to reduce error rates at input time have also been explored [23], as well as speech only solutions [30, 38, 40]. These various methods could conceivably be readily ported to an immersive interaction setting. Even if effective, however, it is likely that such methods would still need to be paired with an explicit error correction interaction to overcome circumstances where the inference step or re-speaking does not deliver the desired outcome.

Also relevant to the broader understanding of text input correction behaviors is the work by Du et al. [5]. Du et al. [5] compile and analyze a dataset of human text revisions of Wikipedia, ArXiv and Wikinews text. Du et al. [5] present a taxonomy of different types of edits, e.g. for fluency, style or clarity, and their relative frequency within the dataset.

In summary, although there has been some work seeking to understand and facilitate text input correction, the general principles that make it effective or ineffective remain unclear. The work presented in this paper serves to advance our understanding in this regard.

3 Text Input Correction in VR

In this section we seek to establish common terminology and structure for describing the design and evaluation of text input correction

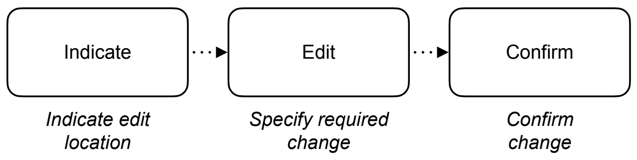


Figure 2: A function model for text input correction.

methods in virtual reality. Also relevant to this goal is an approximate understanding of the frequency and impact of errors encountered within this setting.

First, we make an important distinction between error correction and the less well constrained task of editing. As Robertson and Black [32] observe, text editing is a, "complex cognitive skill that requires significant planning." Error correction is more tightly constrained to the task of correcting errors in a body of text. The more clearly defined task of error correction is the most sensible starting point for evaluating alternative interaction methods for use in virtual reality

Also relevant to the evaluation of interactions methods for error correction is the point in time at which errors are identified and corrected. This timing may influence the strategies employed by users, and correspondingly, the most appropriate interface and interactions to provide. We therefore make a distinction between *input* correction and *review* correction. Input corrections are those occurring at the time of entry, e.g. selecting a word alternative or backspacing to correct a typographical error. Review corrections are those occurring after initial entry, e.g. discovering a typographical error in a word then fixing it.

Finally, corrections can be performed at various levels in the textual hierarchy. The level at which corrective actions are performed may also dictate which is the most appropriate interaction to provide. We define two edit levels in line with Arif and Stuerzlinger [3]. *Character-level* corrections are made to characters within a word. *Word-level* corrections are made to words within a phrase/sentence Some interaction techniques may be better suited to operating at the word-level as opposed to the character-level. There are also potential implications in terms of the most efficient strategy as analyzed later within this section.

3.1 A Generic Function Model for Text Input Correction

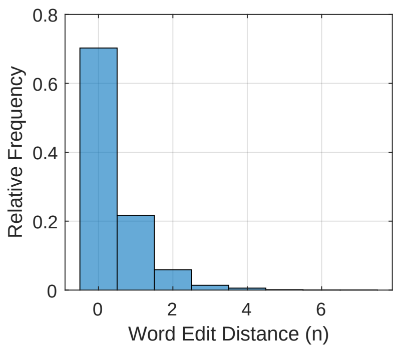
We can further describe the text input correction task in abstract terms by means of a generic function model [20–22]. Figure 2 presents a function model with three key functions: *Indicate*, *Edit* and *Confirm*. Indicate refers to the function of designating the location of the error to be corrected. Edit refers to the function of expressing the change to be made in order to correct the error. Confirm refers to the function of committing the change. Arif and Stuerzlinger [3] introduce a similar decomposition of the error correction task into a sequence of steps and decisions. By contrast, however, the function model in Figure 2 is focused on the generic functions that should be supported by the interface in order to allow the user to correct identified errors. As we will show later in Section 4, this generic function model provides a helpful conceptual framework for considering alternative interaction methods for delivering these specific functions.

3.2 Characterization of Errors and their Impact

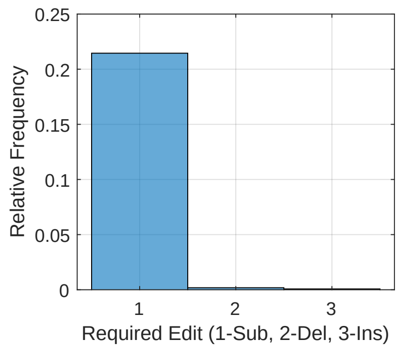
Also critical to understanding the design considerations relevant to supporting text input correction in a mid-air setting within VR is an awareness of the frequency of different types of errors and their impact on performance. There are several important factors that distinguish text input correction in a controller-free mid-air setting from conventional 2D interaction settings and correction in VR with controllers. These factors primarily relate to the frequency at which errors occur, and the precision with which one can control the cursor or articulate the error location. We analyzed the dataset collected by Dudley et al. [10] of 16 participants performing a transcription-based text input task using both gesture and single character typing modalities. This analysis highlighted the fact that the majority of input and review corrections in the dataset relate to word-level corrections where the current and target text differ by only one or two characters.

For example, Figure 3a plots the relative frequency of different word edit distance counts for the entered phrases in the dataset. This plot suggests that there is a power law distribution describing the frequency of word errors in an entered phrase. The majority of phrases contain no errors, while approximately 20% contain one error, and less than 10% contain two errors. If we now examine the type of edit required to address single word errors (i.e. word edit distance of 1), we see in Figure 3b that the vast majority of these errors require a substitution to complete the correction, i.e. there are the correct number of words in the phrase but one of the words is incorrect. Figure 3c plots the character edit distance for these words requiring a substitution. This plot also suggests there is a power law distribution describing the frequency of character errors in a word, with the vast majority of word errors containing one or two character errors.

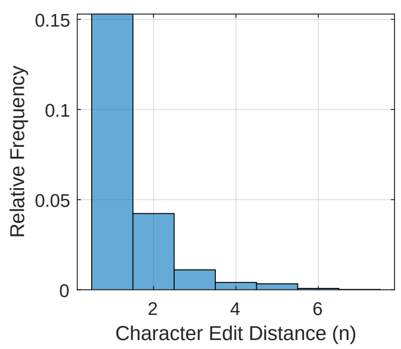
The three plots of Figure 3 viewed in combination highlight the fact that the vast majority of phrases containing a single word error can be resolved by a word substitution, and that the incorrect word is in most cases only one or two characters away from the desired



(a) Relative frequency over word edit distance between stimulus and entered phrase.



(b) Relative frequency over different required edit types for erroneously entered word.



(c) Relative frequency over character edit distance between stimulus and entered word requiring substitution.

Figure 3: Degree, frequency and type of text input errors in the dataset collected by Dudley et al. [10].

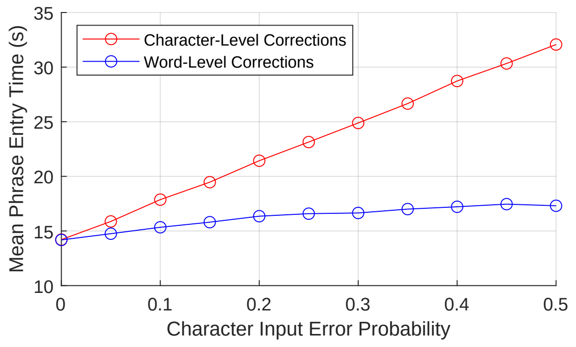


Figure 4: Analytical comparison of a character-level and word-level correction strategy as the probability of character substitution errors increases.

word. The fact that the erroneous word and correct word differ by only one or two characters suggest that there may be opportunities to exploit this similarity to infer the correct replacement word.

Our analysis of the dataset also provides rough estimates of the time required to perform edit actions such as entering a character, pressing the backspace key or selecting word alternatives. This facilitates estimation of the approximate frequency and impact of the different types of correction subtasks as detailed in the following subsection.

3.3 Analytical Evaluation of Correction Strategies

We built a simple interactive tool[1] in MATLAB to help examine the consequential effect of different input correction strategies under different operating assumptions. The tool allows us to examine the estimated effect of uncontrollable parameters such as how prone the input technique is to character selection errors, and how likely the user is to notice these errors. Our analysis using this tool suggests that a strategy based around word-level edits is likely to be more efficient than a strategy based around character-level edits. Furthermore, the benefit of the word-level edit strategy becomes more pronounced as the underlying input method becomes more error prone (i.e. more character-level errors are likely to be encountered). By a word-level strategy in this context, we mean waiting until after each word has been typed, and potentially auto-corrected, before correcting any errors. By a character-level strategy in this context, we mean fixing any incorrect characters immediately after they are entered by pressing backspace and then re-typing the correct character.

To illustrate this effect of strategy, we estimate the mean time to enter a set of 500 phrases taken from the Enron dataset [41], each repeated 10 times. We consider a range of variables that contribute to performance and strategy. Most critically for the purpose of this example, we consider: (i) the amount of time it takes to enter a single character; (ii) the amount of time it takes to press the backspace key to remove a character; (iii) the probability of making a substitution error; (iv) the probability that a word with a substitution error is auto-corrected; and (v) the amount of time it takes to substitute a word using some form of input correction interaction. In practice, we have varying levels of influence or control over these different variables but this does not prevent us from investigating their impact. A very similar approach was used by Kristensson and Müllners [22] in their analytical investigation of the impact of strategy on the performance benefit provided by word completions. Where possible we use values estimated from the previously mentioned dataset [10] but the interactive tool allows for these values to be freely adjusted to evaluate different hypotheses. Figure 4 shows the influence on mean phrase entry time under both a character-level and word-level correction strategy as the probability of making a substitution error increases. We can observe that the rate of increase in entry time, as the input method becomes more error prone, is considerably greater under the character-level strategy than under the word-level strategy. The intuitive explanation for this is that the word-level strategy can benefit from auto-corrections of word errors that would otherwise be fixed in-place in the character-level strategy. Additionally, the total time spent on a given word due to repeated cycles of pressing the backspace key and re-entering the desired character may approach and eventually exceed the amount of time required to interact at the word-level and replace the whole word. This analysis motivates prioritization of input correction methods that facilitate word-level correction.

4 Text Correction System

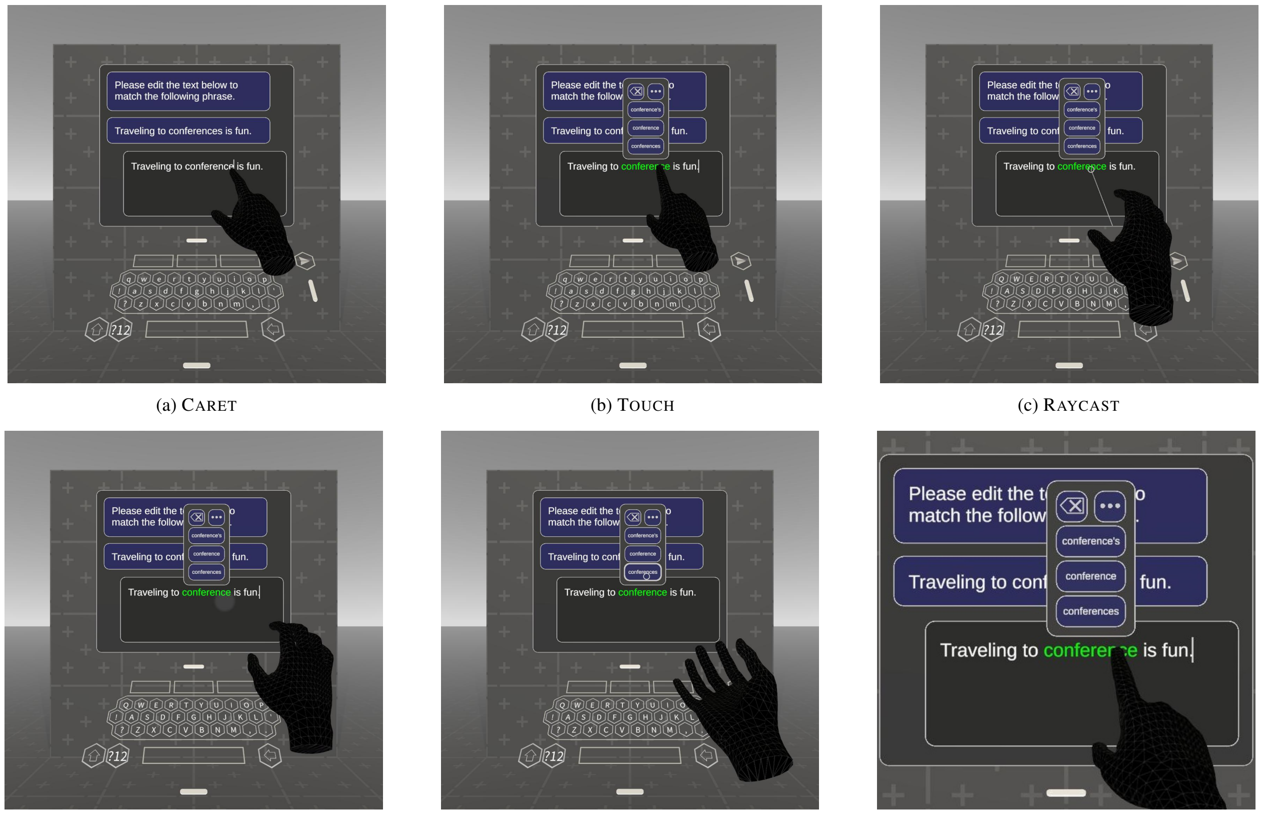
We developed an application to facilitate experimentation with different correction tasks and different interaction methods. The interface, shown in Figure 5, was designed to approximately replicate the visual appearance of the *Messenger* application on Meta Quest devices. The primary components of the interface relevant to this work are the instruction and stimulus text fields (blue background), the input field (dark gray background), and the keyboard. The keyboard implemented in this study supports ‘touch’ typing with the index finger of each hand. The keyboard was placed in the world frame and did not move automatically with the user. However, the user could choose to reposition the keyboard for comfort if they wished. By default, the keyboard was placed at a distance of 25 cm from the user, had a tilt of 45° relative to the vertical plane, and had a width of approximately 30 cm.

An input decoder is incorporated into the keyboard to provide auto-corrections. Selectable alternative auto-corrections are dis-

[1]https://github.com/Jojadud/EditStrategyAnalysisTool

Table 1: Five alternative interaction designs to support mid-air text input correction in VR.

Interaction	Indicate	Edit	Confirm
CARET	Directly touch point in word/sentence to place caret	Press keys on keyboard	
TOUCH	Directly touch word to display word alternatives	Directly touch on desired word alternative	
RAYCAST	Point ray at word, then pinch to display word alternatives	Point ray at desired word alternative	Pinch to insert
GAZE	Gaze at word, then pinch to display word alternatives	Gaze at desired word alternative	Pinch to insert
INDIRECT	Point indirect cursor at word, then pinch to display word alternatives	Point indirect cursor at desired word alternative	Pinch to insert


Please edit the text below to match the following phrase.

Traveling to conferences is fun.

Traveling to conference is fun.

(a) CARET (b) TOUCH (c) RAYCAST

(d) GAZE (e) INDIRECT (f) Context menu detail

Figure 5: Interaction conditions as viewed in the headset.

played above the keyboard as on a typical smartphone keyboard. The input decoder is also leveraged to provide word alternatives during the correction interactions described later in this section. Note that the keyboard does provide a backspace key and this can be used at any time to remove the previously entered character.

We implemented five alternative interaction methods leveraging the different input channels available on modern virtual reality HMDs. For the study reported in this paper, we used the Meta Quest Pro. All five methods are hand-based. This choice reflects the widening support of controller-free interactions on mixed reality devices and the need to provide input correction methods that are consistent with the input method under use. The five designs are summarized in Table 1 and described with reference to the function model introduced in Section 3.1. Each method is now briefly described below. Please also refer to the video figure for demonstrations of the interactions.

4.1 CARET

The CARET method replicates the character-level correction technique available on smartphones. The user directly touches the input field with their index finger to place the caret as shown in Figure 5a. Character-level edits can then be made using the keyboard. The caret can be dragged across the input field or placed using discrete touches.

4.2 TOUCH

The TOUCH method involves touching on a word to display possible corrections for that word. The context menu containing the possible corrections appears above the indicated word as shown in Figure 5b. If present, the user can touch the correct word which will then replace the erroneous word in the input field. If no suitable corrections are available, the user may also delete, by pressing the backspace button on the top left of the context menu, and then re-enter the word using the keyboard.

4.3 RAYCAST

The RAYCAST method involves indicating a word for correction by pointing the raycast cursor at the word, and then performing a pinch gesture to display word alternatives. The desired replacement is inserted by pointing the raycast cursor at the word in the context menu, and using the pinch gesture to confirm insertion. The source of the raycast is the `PointerPose` transform that is attached to, and rotates with, the hand, as per Meta Quest developer guidance[2]. This interaction is shown in Figure 5c.

4.4 GAZE

The GAZE method involves focusing the eyes on the word to be corrected, and then performing a pinch gesture to display alternatives. The core of this method is based on the Gaze + Pinch interaction technique described by Pfeuffer et al. [31]. The replacement word is inserted by looking at the desired word, and using the pinch gesture again to confirm insertion. We sought to incorporate emerging guidance[3] on gaze-based interaction when implementing the GAZE method. Specifically, we: (1) used a rounded visual style to help avoid user's eyes being involuntarily drawn to corners; (2) placed visual elements at a comfortable viewing distance with separation between interactive elements; and (3) used subtle visualizations to provide feedback on highlighted and selected elements. The gaze

[2]https://developer.oculus.com/documentation/unity/unity-handtracking/

[3]https://developer.apple.com/design/human-interface-guidelines/eyes

cursor (see Figure 5d) was implemented as a spot light with soft edges rather than a distinct visual cursor. Gaze hover events triggered subtle visual highlighting of UI elements.

The Quest SDK gives two independent vectors for the eye gaze. We cast these vectors onto the UI elements and take the average of the two intersection points over the 10 most recent observations (at 72 Hz). This buffer size was chosen to balance between responsiveness and jitter. The Quest system eye gaze calibration was performed by all participants prior to launch of the experiment application.

4.5 Indirect

The Indirect method introduces a temporary indirect cursor that is controlled by hand movements. To trigger display of the indirect cursor, the user must first turn over their hand such that the palm is facing up. Rotating the hand back to a neutral posture will hide the cursor. Each time the indirect cursor is triggered, it is initialized to be at the center of the input field. The initial position of the hand when the indirect cursor is triggered then serves as the reference point for controlling the cursor.

Otherwise, the interactions with the word correction context menu are the same. The word in need of correction is indicated by pointing the indirect cursor and pinching. The replacement word is then selected and confirmed by pointing the indirect cursor and pinching.

5 User Study: Comparing Correction Methods

We now describe our user study designed to evaluate the performance and usability trade-offs for the five alternative interactions introduced in Section 4. We examine performance by means of task completion time and usage rates. We capture usability by means of a questionnaire in which users rate the different interactions in terms of speed, accuracy and comfort. These quantitative metrics are complemented by user feedback and observations collected during the study.

5.1 Protocol

Participants were exposed to each interaction condition in sequence and performed corrections in two stages. Stage 1 was a review correction task and required the participant to correct a pre-existing phrase containing errors. Stage 2 was an input correction task and required the participant to transcribe a stimulus phrase and correct any errors that occurred. Text entry was always performed using the mid-air keyboard described in Section 4. There were 3 practice phrases and 10 test phrases in each stage, giving a total of 30 practice phrases and 100 test phrases across the two stages and five conditions. The sequence of conditions was balanced over participants using a fully balanced Latin square (10 possible order permutations with five conditions). The Quest system eye gaze tracking calibration was completed by all participants before the experimental application was launched.

Participants were required to correct all errors in the phrase in order to proceed to the next phrase. This was enforced by preventing submission and playing an error tone if the participant tried to submit the phrase with errors remaining. The error tone prompted the participant to identify and correct any remaining errors.

5.2 Stimulus Phrases

The phrases for the review correction task were taken from real erroneous submissions collected in the mid-air typing study conducted by Dudley et al. [10]. The original source of these phrases was the MacKenzie phrase set [29]. The three practice phrases remained the same throughout the study but all test phrases were unique and randomly assigned to conditions based on a stratified allocation of the number of errors in each phrase. To approximate the relative frequency of word edit distance shown in Figure 3a, we allocate: (i) seven phrases with a single word error, (ii) two phrases with two word errors, and (iii) a single phrase with three word errors. The

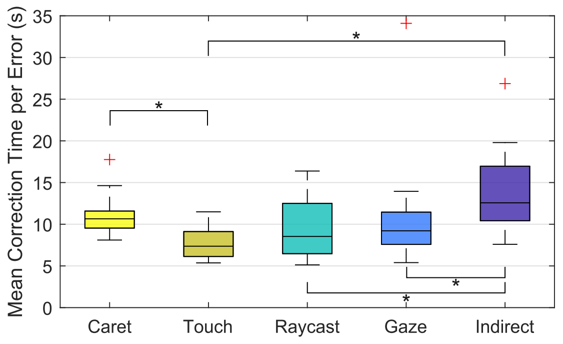


Figure 6: Mean correction time per error in the review correction task. The '*' symbol indicates a significant difference between conditions.

ordering of phrases within a condition was such that all single word error phrases would appear before two word error phrases and the single phrase with three word errors would appear last. The 50 unique stimulus phrases used in the input correction task were taken from the Enron mobile message dataset [41] after filtering based on phrases containing four words or more, and 40 characters or less.

5.3 Participants

20 participants were recruited for the study by convenience sampling and poster-based advertisement. The recruited participant group was comprised of 5 females and 15 males with a mean age of 26.5 years [19,55]. All participants were right handed (this was not a criteria for participation). When completing the pre-study demographics questionnaire, participants also rated their prior experience with VR on a scale from 1 (*Very Inexperienced: I have only ever used VR once or twice before, if at all.*) to 5 (*Very Experienced: I use VR several times a month*). The median response was 3, suggesting a good mix of participants with varying levels of familiarity.

6 Results

6.1 Stage 1: Review Correction

As described in Section 5.2, the review correction task stimulus phrases were selected to contain a set number of word errors within each phrase. To understand the correction efficiency of each interaction method, we compute the mean correction time per error. The correction time per error is simply the total time taken to fix all errors in a phrase divided by the number of initial word errors.

Figure 6 plots the distribution of mean correction time per error over the participant group. The mean correction times per error were Caret: 11.02 s, Touch: 7.82 s, Raycast: 9.55 s, Gaze: 10.55 s, and Indirect: 13.94 s. A repeated measures analysis of variance shows a significant effect for the interaction method on correction time ($F_{4,76} = 9.729$, $\eta_p^2 = 0.339$, $p < 0.001$). A multiple comparisons test with Bonferroni correction finds that Touch is significantly faster than Caret ($p = 0.024$) and Indirect ($p < 0.001$) but there is no significant difference between Touch and Raycast or Touch and Gaze.

This result suggests that the Touch method is more efficient at correcting errors than Caret. This finding is consistent with the analytical results from Section 3.2 which suggests that word-level correction is likely to be more efficient that character-level correction. Although correction time per error was lower in Raycast and Gaze compared with Caret, this difference was not significant. The Indirect interaction method performed worst which may stem from the lack of user familiarity with such an interaction method in a VR setting.

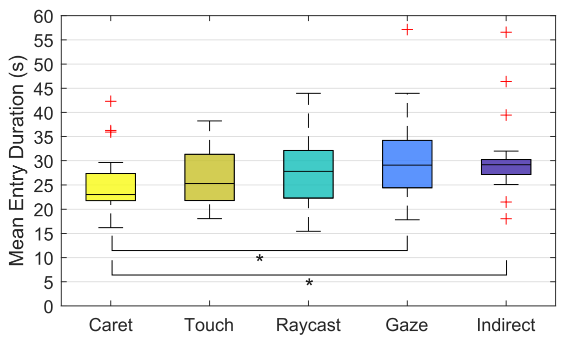


Figure 7: Mean entry duration in the input correction task. The '*' symbol indicates a significant difference between conditions.

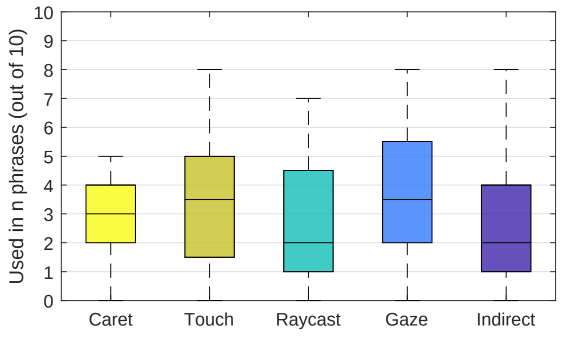


Figure 8: Count of phrases (out of 10) where the primary input correction interaction was used.

6.2 Stage 2: Input Correction

In the input correction task, the participant must both enter the phrase and correct any errors that occur. The entry duration therefore reflects the combined impact of text entry and correction actions. Figure 7 plots the distribution of entry durations over the participant group. The mean entry durations were CARET: 24.82 s, TOUCH: 26.47 s, RAYCAST: 27.06 s, GAZE: 30.09 s, and INDIRECT: 30.54 s. A repeated measures analysis of variance shows a significant effect for the interaction method on entry duration ($F_{4,76} = 4.058$, $\eta_p^2 = 0.176$, $p = 0.005$). A multiple comparisons test with Bonferroni correction finds that CARET is significantly faster than GAZE ($p = 0.030$) and INDIRECT ($p = 0.013$) but there is no significant difference between the other conditions.

The primary correction interaction method was fixed for a given condition within the input correction task, however, it was not uncommon for participants to fallback on using backspace to correct errors or indeed to enter the phrase without making any errors. Figure 8 is a boxplot of the number of phrases in which the primary correction interaction method was used by participants out of the 10 test phrases in the input correction task. We can observe that the primary correction interactions were typically used in less than half of the entered phrases. It is interesting to note that the median usage counts for TOUCH and GAZE are marginally higher than CARET while the usage counts for RAYCAST and INDIRECT are marginally lower.

6.3 Post-Study Questionnaire Responses

At the conclusion of the study, participants completed a questionnaire examining their experience of the five interaction conditions and their preferences. Participants responded to three statements relating to their experience of the interaction methods in terms of speed, accuracy and comfort. Responses were collected on a five-point Likert scale (1-Strongly Disagree (SD), 2-Disagree (D), 3-Neutral (N), 4-Agree (A), 5-Strongly Agree (SA)). Figures 9 to 11 show the distribution of responses for speed, accuracy and comfort respectively. A Friedman's test shows a significant effect for interaction method in terms of speed ($\chi^2(4) = 16.46$, $p = 0.0025$) and accuracy ($\chi^2(4) = 10.29$, $p = 0.0358$). Running multiple comparisons with Bonferroni correction reveals that responses are significantly more positive in terms of speed for TOUCH ($p = 0.0034$) and GAZE ($p = 0.0405$) relative to CARET. For accuracy, a significant difference was only found between TOUCH and RAYCAST ($p = 0.0213$).

Participants were also asked to indicate their most preferred interaction condition for the review correction task, the input correction task, and overall. The captured responses are summarized in Table 2. The most preferred method in the review correction task was Gaze (10) followed by Touch (7). For input correction, however, Caret (9) was the most preferred method. Overall, the Gaze (8) and Touch (8) methods were equally preferred.

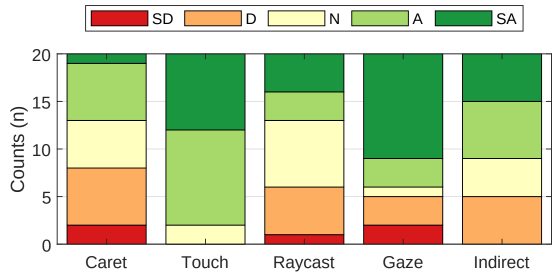


Figure 9: Participant responses to statement, "The technique made it easy to edit text quickly." on a five-point Likert scale from 1-Strongly Disagree (SD) to 5-Strongly Agree (SA).

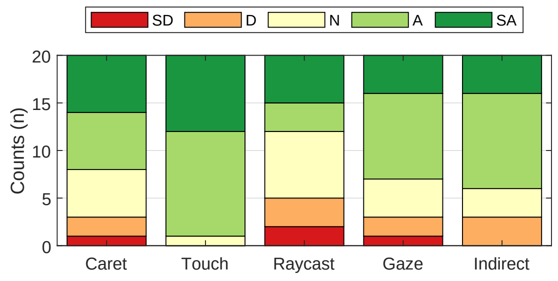


Figure 10: Participant responses to statement, "The technique made it easy to edit text accurately." on a five-point Likert scale from 1-Strongly Disagree (SD) to 5-Strongly Agree (SA).

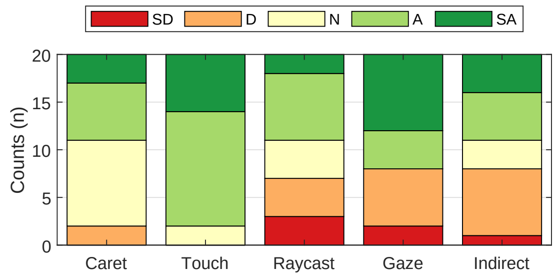


Figure 11: Participant responses to statement, "The technique was comfortable to use." on a five-point Likert scale from 1-Strongly Disagree (SD) to 5-Strongly Agree (SA).

Table 2: Interaction method preferences for the review and input correction tasks, as well as overall.

Condition	Review	Input	Overall
CARET	0	**9**	0
TOUCH	7	4	**8**
RAYCAST	3	2	2
GAZE	**10**	5	**8**
INDIRECT	0	0	2

6.3.1 Qualitative Feedback and Observations

Participants were provided with space to write comments reflecting on the positive and negative aspects of each method. Two of the most prominent themes common to these comments were the familiarity of the CARET method ("Similar to mobile phone so very familiar" (*P6*); "Similar experience to phone" (*P9*); "Same as smartphone" (*P14*); "Similar to everyday typing on the phone" (*P15*); "Follows the design of the smartphone interaction" (*P18*)) and the reduced physical effort required by the GAZE method ("Super fast and accurate compared to physical movement of hand" (*P4*); "Does not require physical access to text" (*P8*); "Felt an extension to the brain" (*P13*); "Least physical movement involved" (*P16*); "Less effort to edit" (*P20*)). The CARET method is indeed very similar to the interaction provided on most smartphones. The importance of familiarity is a critical factor for the success of text entry methods [19] and it seems reasonable to expect that this may extend to aspects of the supported input correction interactions. The feedback on the GAZE method is positive and corroborates the preference results and ratings in terms of speed. A key challenge that we observed for participants in the GAZE condition, however, was that experienced accuracy of the gaze tracking varied from person to person. When tracking issues were encountered during the experiment, we did run the eye gaze tracking calibration process again for the participant. This resolved most cases but three participants in the study had persistent issues.

7 DISCUSSION

This paper has presented an investigation of interaction design for text input correction in VR. We observed that word-level error correction is more efficient than character-level error correction when fixing errors in previously inputted phrases, but that this efficiency does not necessarily translate to fixing errors at input time. Our results suggest that a comprehensive text input system for VR should provide both word-level and character-level interaction methods. Fortunately, the interaction methods evaluated in this study could in fact be delivered in combination. For example, CARET and TOUCH could be distinguished by dwell or double-tap during the indicate phase of the interaction. The preference results presented in Table 2 also suggest that users may appreciate the seamless integration of complementary interaction methods. For example, TOUCH and GAZE could be delivered in a complementary manner by toggling between modes depending on whether the hand is in a preparatory posture for performing the pinch gesture. Similarly, although the Raycast method was generally found to be less efficient than Touch, Gaze and Caret, there may be value in supporting Raycast-based error correction if the text input method also employs a Raycast interaction. This as an example of the more general anticipated requirement for consistency between input and correction methods.

As reviewed in Section 2, Li et al. [27] focused on caret-based correction using a controller. The fastest controller-based method (WBs-CCc) in Li et al.'s study produced a mean correction time of 7.98 s. This is faster than the mean correction time for the Caret method (11.02 s) in our study leveraging the headset hand tracking, but comparable to the Touch method (7.82 s). This suggests that, with careful design, correction methods leveraging hand tracking can achieve efficiency comparable to controller-based methods focused on caret control.

The results of the user study also provide some preliminary evidence of the potential for gaze in delivering efficient text correction interactions. In a review correction scenario, the task of locating word errors in a phrase requires sequential fixation on each word. In the ideal case, the user can then simply pinch to display the context menu when they encounter an erroneous word. This is a more implicit use of gaze input than, for example, the task of forcibly moving one's eyes to each key in a Qwerty layout to enter characters with a gaze-based dwell timeout.

The five methods examined differ in terms of the hand and arm posture that they require the user to maintain. Figure 11 suggests that Raycast, Gaze and Indirect were generally perceived as less comfortable which is consistent with the observation that all three methods required the arm to be held cantilevered out from the body with the hand in a posture ready to pinch. By comparison, the Caret and Touch methods allowed the hand and arm to be placed in the user's preferred posture in between interaction events. As briefly reviewed in Section 2, there may also be opportunities for incorporating haptic feedback to promote more relaxed or effective postures [16] and to improve user feedback to limit unnecessary motions [7].

8 LIMITATIONS AND FUTURE WORK

We acknowledge several limitations of this work that should be considered when interpreting the findings. First, the typing and error correction behavior analyzed in Subsection 3.2 is based on data from a transcription task. This does not necessarily represent typical input or correction behavior. Second, due to the number of methods examined, the sample size per method per stage is relatively small and should be expanded in future work. Nevertheless, this current work does inform prioritization of methods for further examination, and also offers some insight regarding immediate efficacy given short exposure and limited training. Third, the participants in our study were mostly young and male and so the efficiency measures reported may not generalize to the wider population.

Finally, the gaze-based interaction method described in Subsection 4.4 represents a relatively simple implementation that may have been negatively impacted by complex considerations around the design of gaze-based interactions. Our approach to filtering the gaze signal will have introduced some lag that can be detrimental to user performance [11] and exacerbate late-trigger errors [24]. Further, although our visual representation of the gaze intersection point was designed to avoid drawing focus, it remains possible that participants experienced the 'fleeing cursor' problem if the eye gaze calibration was ineffective of drifted over time. Future work involving eye gaze interactions in this setting should consider follow up testing post-calibration and at intervals to assess the accuracy experienced by participants [28].

This work also highlights three key avenues requiring further research. First, we have chiefly focused in this paper on character and word-level correction. Supporting sentence and paragraph-level edits, demanding selection and context commands like copy, cut and paste, is likely to require other forms of interaction. Second, there are likely opportunities for improving the efficiency of caret control in a mid-air setting, building upon work such as that by Hu et al. [17]. Third, the study presented in this paper intentionally avoided highlighting potential errors in the input field. In practice, highlighting very likely errors can be helpful to the user and may provide the basis for modulating the behavior of the correction interactions. For example, a very likely error fixated on with gaze might bring up the context menu without the need for performing the pinch gesture.

9 Conclusions

In comparison with the considerable research that has introduced and evaluated diverse methods for inputting text in VR, the task of correcting text input errors has received very limited attention. In this work, we seek to address this gap in several key ways. First, we describe the task of designing and evaluating a text input correction system for use within VR, defining standard terminology and characterizing the frequency and impact of text input errors in VR. This analysis suggests that a word-based correction strategy is likely to outperform caret-based text correction in a mid-air setting. We then present an empirical investigation of five alternative interaction methods for correcting errors in VR. We find that word-level touch based error correction provides the most efficient way to correct pre-existing errors. However, when performing input correction during text entry then a character-level caret-based may be preferable. These contributions and findings advance our understanding of the requirements for efficient and usable mid-air text input correction methods in VR.

References

[1] J. Adhikary and K. Vertanen. Text Entry in Virtual Environments using Speech and a Midair Keyboard. *IEEE Transactions on Visualization and Computer Graphics*, 27(5):2648–2658, May 2021. Conference Name: IEEE Transactions on Visualization and Computer Graphics. doi: 10.1109/TVCG.2021.3067776 2

[2] J. Adhikary and K. Vertanen. Typing on Midair Virtual Keyboards: Exploring Visual Designs and Interaction Styles. In C. Ardito, R. Lanzilotti, A. Malizia, H. Petrie, A. Piccinno, G. Desolda, and K. Inkpen, eds., *Human-Computer Interaction – INTERACT 2021*, pp. 132–151. Springer International Publishing, Cham, 2021. 2

[3] A. S. Arif and W. Stuerzlinger. 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, p. 5–14. Association for Computing Machinery, New York, NY, USA, 2010. doi: 10.1145/1753326.1753329 3

[4] W. Cui, S. Zhu, M. R. Zhang, H. A. Schwartz, J. O. Wobbrock, and X. Bi. Justcorrect: Intelligent post hoc text correction techniques on smartphones. In *Proceedings of the 33rd Annual ACM Symposium on User Interface Software and Technology*, UIST '20, p. 487–499. Association for Computing Machinery, New York, NY, USA, 2020. doi: 10.1145/3379337.3415857 2

[5] W. Du, V. Raheja, D. Kumar, Z. M. Kim, M. Lopez, and D. Kang. Understanding iterative revision from human-written text. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 3573–3590. Association for Computational Linguistics, Dublin, Ireland, May 2022. doi: 10.18653/v1/2022.acl-long.250 2

[6] T. J. Dube and A. S. Arif. Text Entry in Virtual Reality: A Comprehensive Review of the Literature. In M. Kurosu, ed., *Human-Computer Interaction. Recognition and Interaction Technologies*, Lecture Notes in Computer Science, pp. 419–437. Springer International Publishing, Cham, 2019. doi: 10.1007/978-3-030-22643-5_33 1, 2

[7] T. J. Dube and A. S. Arif. Ultrasonic Keyboard: A Mid-Air Virtual Qwerty with Ultrasonic Feedback for Virtual Reality. In *Proceedings of the Seventeenth International Conference on Tangible, Embedded, and Embodied Interaction*, TEI '23. Association for Computing Machinery, New York, NY, USA, 2023. doi: 10.1145/3569009.3573117 2, 8

[8] J. J. Dudley, H. Benko, D. Wigdor, and P. O. Kristensson. Performance Envelopes of Virtual Keyboard Text Input Strategies in Virtual Reality. In *2019 IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*, pp. 289–300, Oct. 2019. doi: 10.1109/ISMAR.2019.00027 2

[9] J. J. Dudley, K. Vertanen, and P. O. Kristensson. Fast and Precise Touch-Based Text Entry for Head-Mounted Augmented Reality with Variable Occlusion. *ACM Transactions on Computer-Human Interaction*, 25(6):30:1–30:40, Dec. 2018. doi: 10.1145/3232163 2

[10] J. J. Dudley, J. Zheng, A. Gupta, H. Benko, M. Longest, R. Wang, and P. Kristensson. Evaluating the Performance of Hand-Based Probabilistic Text Input Methods on a Mid-Air Virtual Qwerty Keyboard. *IEEE Transactions on Visualization and Computer Graphics*, 29(11):4567–4577, nov 2023. doi: 10.1109/TVCG.2023.3320238 1, 2, 3, 4, 6

[11] A. M. Feit, S. Williams, A. Toledo, A. Paradiso, H. Kulkarni, S. Kane, and M. R. Morris. Toward Everyday Gaze Input: Accuracy and Precision of Eye Tracking and Implications for Design. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems*, CHI '17, p. 1118–1130. Association for Computing Machinery, New York, NY, USA, 2017. doi: 10.1145/3025453.3025599 8

[12] C. R. Foy, J. J. Dudley, A. Gupta, H. Benko, and P. O. Kristensson. Understanding, Detecting and Mitigating the Effects of Coactivations in Ten-Finger Mid-Air Typing in Virtual Reality. In *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems*, CHI '21. Association for Computing Machinery, New York, NY, USA, 2021. doi: 10.1145/3411764.3445671 2

[13] M. Frutos-Pascual, C. Gale, J. M. Harrison, C. Creed, and I. Williams. Character Input in Augmented Reality: An Evaluation of Keyboard Position and Interaction Visualisation for Head-Mounted Displays. In C. Ardito, R. Lanzilotti, A. Malizia, H. Petrie, A. Piccinno, G. Desolda, and K. Inkpen, eds., *Human-Computer Interaction – INTERACT 2021*, pp. 480–501. Springer International Publishing, Cham, 2021. 2

[14] J. Grubert, L. Witzani, E. Ofek, M. Pahud, M. Kranz, and P. O. Kristensson. Text Entry in Immersive Head-Mounted Display-Based Virtual Reality Using Standard Keyboards. In *2018 IEEE Conference on Virtual Reality and 3D User Interfaces (VR)*, pp. 159–166, Mar. 2018. doi: 10.1109/VR.2018.8446059 2

[15] A. Gupta, M. Samad, K. Kin, P. O. Kristensson, and H. Benko. Investigating Remote Tactile Feedback for Mid-Air Text-Entry in Virtual Reality. In *2020 IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*, pp. 350–360, Nov. 2020. ISSN: 1554-7868. doi: 10.1109/ISMAR50242.2020.00062 2

[16] A. Gupta, N. Sendhilnathan, J. Hartcher-O'Brien, E. Pezent, H. Benko, and T. R. Jonker. Investigating Eyes-away Mid-air Typing in Virtual Reality using Squeeze haptics-based Postural Reinforcement. In *Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems*, CHI '23. Association for Computing Machinery, New York, NY, USA, 2023. doi: 10.1145/3544548.3581467 2, 8

[17] J. Hu, J. J. Dudley, and P. O. Kristensson. An Evaluation of Caret Navigation Methods for Text Editing in Augmented Reality. In *2022 IEEE International Symposium on Mixed and Augmented Reality Adjunct (ISMAR-Adjunct)*, pp. 640–645, 2022. doi: 10.1109/ISMAR-Adjunct57072.2022.00132 2, 8

[18] F. Kern, F. Niebling, and M. E. Latoschik. Text Input for Non-Stationary XR Workspaces: Investigating Tap and Word-Gesture Keyboards in Virtual and Augmented Reality. *IEEE Transactions on Visualization and Computer Graphics*, 29(5):2658–2669, 2023. doi: 10.1109/TVCG.2023.3247098 2

[19] P. O. Kristensson. Next-Generation Text Entry. *Computer*, 48(7):84–87, July 2015. Conference Name: Computer. doi: 10.1109/MC.2015.185 8

[20] P. O. Kristensson. Designing Virtual and Augmented Reality User Interfaces using Parameterized Function Structure Models. In *2024 IEEE Conference on Virtual Reality and 3D User Interfaces Abstracts and Workshops (VRW)*, pp. 419–423. IEEE, 2024. doi: 10.1109/VRW62533.2024.00081 3

[21] P. O. Kristensson, J. Lilley, R. Black, and A. Waller. A Design Engineering Approach for Quantitatively Exploring Context-Aware Sentence Retrieval for Nonspeaking Individuals with Motor Disabilities. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems*, CHI '20, p. 1–11. Association for Computing Machinery, New York, NY, USA, 2020. doi: 10.1145/3313831.3376525 3

[22] P. O. Kristensson and T. Müllners. Design and Analysis of Intelligent Text Entry Systems with Function Structure Models and Envelope Analysis. In *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems*, CHI '21. Association for Computing Machinery, New York, NY, USA, 2021. doi: 10.1145/3411764.3445566 3, 4

[23] P. O. Kristensson and K. Vertanen. Asynchronous Multimodal Text Entry Using Speech and Gesture Keyboards. In *Twelfth Annual Conference of the International Speech Communication Association*, 2011. 2

[24] M. Kumar, J. Klingner, R. Puranik, T. Winograd, and A. Paepcke. Improving the accuracy of gaze input for interaction. In *Proceedings of the 2008 Symposium on Eye Tracking Research & Applications*, ETRA '08, p. 65–68. Association for Computing Machinery, New York, NY, USA, 2008. doi: 10.1145/1344471.1344488 8

[25] Y. Lee and G. J. Kim. Vitty: Virtual Touch Typing Interface with Added Finger Buttons. In S. Lackey and J. Chen, eds., *Virtual, Augmented and Mixed Reality*, pp. 111–119. Springer International Publishing, Cham, 2017. 2

[26] L. A. Leiva, S. Kim, W. Cui, X. Bi, and A. Oulasvirta. How We Swipe: A Large-Scale Shape-Writing Dataset and Empirical Findings. In *Proceedings of the 23rd International Conference on Mobile Human-Computer Interaction*, MobileHCI '21. Association for Computing Machinery, New York, NY, USA, 2021. doi: 10.1145/3447526.3472059 2

[27] Y. Li, S. Sarcar, Y. Zheng, and X. Ren. Exploring Text Revision with Backspace and Caret in Virtual Reality. In *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems*, CHI '21. Association for Computing Machinery, New York, NY, USA, 2021. doi: 10.1145/3411764.3445474 2, 8

[28] D. Lohr, S. Aziz, L. Friedman, and O. V. Komogortsev. GazeBaseVR, a large-scale, longitudinal, binocular eye-tracking dataset collected in virtual reality. *Scientific Data*, 10(1):177, 2023. 8

[29] I. S. MacKenzie and R. W. Soukoreff. Phrase Sets for Evaluating Text Entry Techniques. In *CHI '03 Extended Abstracts on Human Factors in Computing Systems*, CHI EA '03, p. 754–755. Association for Computing Machinery, New York, NY, USA, 2003. doi: 10.1145/765891.765971 6

[30] A. E. McNair and A. Waibel. Improving recognizer acceptance through robust, natural speech repair. In *3rd International Conference on Spoken Language Processing (ICSLP 1994)*. ISCA, 1994. 2

[31] K. Pfeuffer, B. Mayer, D. Mardanbegi, and H. Gellersen. Gaze + pinch interaction in virtual reality. In *Proceedings of the 5th Symposium on Spatial User Interaction*, SUI '17, p. 99–108. Association for Computing Machinery, New York, NY, USA, 2017. doi: 10.1145/3131277.3132180 5

[32] S. P. Robertson and J. B. Black. Structure and Development of Plans in Computer Text Editing. *Human–Computer Interaction*, 2(3):201–226, 1986. doi: 10.1207/s15327051hci0203_2 3

[33] J. Shen, J. Dudley, and P. O. Kristensson. Simulating Realistic Human Motion Trajectories of Mid-Air Gesture Typing. In *2021 IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*, pp. 393–402, 2021. doi: 10.1109/ISMAR52148.2021.00056 2

[34] J. Shen, J. Dudley, and P. O. Kristensson. Fast and Robust Mid-Air Gesture Typing for AR Headsets using 3D Trajectory Decoding. *IEEE Transactions on Visualization & Computer Graphics*, 29(11):4622–4632, nov 2023. doi: 10.1109/TVCG.2023.3320218 2

[35] J. Shen, J. Hu, J. J. Dudley, and P. O. Kristensson. Personalization of a Mid-Air Gesture Keyboard using Multi-Objective Bayesian Optimization. In *2022 IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*, pp. 702–710, 2022. doi: 10.1109/ISMAR55827.2022.00088 2

[36] Z. Song, J. J. Dudley, and P. O. Kristensson. Efficient Special Character Entry on a Virtual Keyboard by Hand Gesture-Based Mode Switching. In *2022 IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*, pp. 864–871, 2022. doi: 10.1109/ISMAR55827.2022.00105 2

[37] M. Speicher, A. M. Feit, P. Ziegler, and A. Krüger. Selection-Based Text Entry in Virtual Reality. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*, CHI '18, p. 1–13. Association for Computing Machinery, New York, NY, USA, 2018. doi: 10.1145/3173574.3174221 2

[38] K. Vertanen and P. O. Kristensson. Automatic selection of recognition errors by respeaking the intended text. In *2009 IEEE Workshop on Automatic Speech Recognition & Understanding*, pp. 130–135, 2009. doi: 10.1109/ASRU.2009.5373347 2

[39] K. Vertanen and P. O. Kristensson. Parakeet: A Continuous Speech Recognition System for Mobile Touch-Screen Devices. In *Proceedings of the 14th International Conference on Intelligent User Interfaces*, IUI '09, pp. 237–246. Association for Computing Machinery, New York, NY, USA, 2009. doi: 10.1145/1502650.1502685 2

[40] K. Vertanen and P. O. Kristensson. Getting it right the second time: Recognition of spoken corrections. In *2010 IEEE Spoken Language Technology Workshop*, pp. 289–294, 2010. doi: 10.1109/SLT.2010.5700866 2

[41] K. Vertanen and P. O. Kristensson. 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, pp. 295–298. Association for Computing Machinery, New York, NY, USA, Aug. 2011. doi: 10.1145/2037373.2037418 4, 6

[42] J. O. Wobbrock. Measures of text entry performance. In I. S. MacKenzie and K. Tanaka-Ishii, eds., *Text Entry Systems: Mobility, Accessibility, Universality*, pp. 47–74. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2007. 2

[43] X. Yi, C. Yu, M. Zhang, S. Gao, K. Sun, and Y. Shi. ATK: Enabling Ten-Finger Freehand Typing in Air Based on 3D Hand Tracking Data. In *Proceedings of the 28th Annual ACM Symposium on User Interface Software and Technology*, UIST '15, p. 539–548. Association for Computing Machinery, New York, NY, USA, 2015. doi: 10.1145/2807442.2807504 2

[44] M. Zhao, W. Cui, I. Ramakrishnan, S. Zhai, and X. Bi. Voice and Touch Based Error-tolerant Multimodal Text Editing and Correction for Smartphones. In *The 34th Annual ACM Symposium on User Interface Software and Technology*, UIST '21, p. 162–178. Association for Computing Machinery, New York, NY, USA, 2021. doi: 10.1145/3472749.3474742 2

[45] M. Zhao, H. Huang, Z. Li, R. Liu, W. Cui, K. Toshniwal, A. Goel, A. Wang, X. Zhao, S. Rashidian, F. Baig, K. Phi, S. Zhai, I. Ramakrishnan, F. Wang, and X. Bi. EyeSayCorrect: Eye Gaze and Voice Based Hands-free Text Correction for Mobile Devices. In *27th International Conference on Intelligent User Interfaces*, IUI '22, p. 470–482. Association for Computing Machinery, New York, NY, USA, 2022. doi: 10.1145/3490099.3511103 2