Shared Control for Text Editing in Virtual Reality with Voice, Gaze and Touch

John J Dudley
Department of Engineering
University of Cambridge
Cambridge, United Kingdom
jjd50@cam.ac.uk

Amy Karlson
Reality Labs Research
Meta Inc.
Redmond, Washington, USA
akkarlson@meta.com

Kashyap Todi
Reality Labs Research
Meta Inc.
Redmond, Washington, USA
kashyap.todi@gmail.com

Matt Longest
Reality Labs Research
Meta Inc.
Redmond, Washington, USA
matt.longest@meta.com

Hrvoje Benko
Reality Labs Research
Meta Inc.
Redmond, Washington, USA
benko@meta.com

Robert Wang
Reality Labs Research
Meta Inc.
Redmond, Washington, USA
rywang@meta.com

Per Ola Kristensson
Department of Engineering
University of Cambridge
Cambridge, United Kingdom
pok21@cam.ac.uk

(a) Voice Only (b) Voice + Touch (c) Voice + Gaze

Figure 1: The three alternative interaction methods for voice-based text editing investigated in this study.

Abstract

Despite its centrality to the overall text entry experience, the task of editing text in virtual reality has received limited attention in the literature. In this paper, we focus on the task of editing text in virtual reality and explore the opportunities afforded by Large Language Models (LLMs) in supporting this task. We specifically investigate how an LLM can be employed to mediate free-form speech-based text editing commands given by users. This flexible approach is inspired by the concept of shared control and allows users to potentially focus less on the specific edit operations required, and more on the desired outcome. However, such flexibility in the user commands can introduce potential ambiguity regarding the desired target of the edit. We therefore study three different interaction conditions representing promising alternative methods for expressing the intended location of an edit: (i) directly via voice commands; (ii) by highlighting words in the text field with a touch-based interaction; and (iii) implicitly via gaze fixations. We find that participants develop effective strategies for collaborating with the LLM-based agent to make edits but also appreciate the more explicit interaction based on touch for expressing edit context.

CCS Concepts

• **Human-centered computing** → **Virtual reality**; **Text input**.

Keywords

Virtual Reality, Text Editing, Shared Control

ACM Reference Format:
John J Dudley, Amy Karlson, Kashyap Todi, Matt Longest, Hrvoje Benko, Robert Wang, and Per Ola Kristensson. 2025. Shared Control for Text Editing in Virtual Reality with Voice, Gaze and Touch. In *ACM Symposium on Spatial User Interaction (SUI '25), November 10–11, 2025, Montreal, QC, Canada.* ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/3694907.3765928

SUI '25, Montreal, QC, Canada

ACM ISBN 979-8-4007-1259-3/25/11
https://doi.org/10.1145/3694907.3765928

1 Introduction

The efficiency and usability of text entry methods for virtual reality have improved considerably over the past decade [7, 9, 26, 31]. As

the achievable entry rates approach that of alternative interaction settings such as typing on a smartphone [9, 19, 24], there is a need to address persisting performance bottlenecks [14, 15], such as those related to the ancillary tasks of correcting and editing text. Furthermore, switching between entering and editing text should ideally be seamless. For example, the demonstrated efficacy of bare-hand mid-air text entry methods [9, 28] motivates the exploration of similarly unencumbered and mobile text editing methods.

In this paper, we investigate methods for improving efficiency and usability in the task of correcting and editing text in virtual reality. In particular, we explore the potential of collaboration with an intelligent text correction and editing system via voice commands. We view this introduction of a collaborative aspect into the text editing task as a form of *shared control*. Shared control has its origins in early work by Sheridan [30] in telerobotics. The shared control concept describes how interaction with an intelligent system, "may be controlled along a continuum from direct manual control to quite high-level supervisory command and control" [30, p. 488]. This framing in the context of text entry and editing reflects the fact that at times the user may focus on low level execution (e.g., inserting individual characters or adjusting the caret), and at other times offer high level strategic direction (e.g., specifying the goal of an edit).

Text editing is a "complex cognitive skill that requires significant planning" [27]. To explore how users formulate and apply text editing strategies, we developed a prototype system capable of making edits based on user-generated voice commands. These voice commands are passed to a Large Language Model (LLM) along with the original text to obtain a revised version of the text. This approach means that the form of user-generated voice commands is not constrained, thereby allowing users to experiment with different strategies. A key deficiency of this approach is that unconstrained voice commands may be ambiguous.

We hypothesize that users either consciously or unconsciously trade-off between brevity and specificity. This aspect implies an important sub-task when performing an edit with voice commands related to the indication of context, that is, where in the body of text the edit is to be made. To this end we developed three alternative approaches for indicating context in order to examine how this impacts user behavior and perceived utility: (i) indicating context with the voice command; (ii) indicating context by highlighting words in the text field with the fingertip; and (iii) indicating context implicitly based on gaze fixations. Each of these context indication methods offers unique strengths and weaknesses that are relevant to study within the text editing task. For example, the use of gaze can potentially offer a way to indicate context with low physical effort, but with reduced accuracy and agency relative to touch. While there has been some work exploring these approaches in the context of smartphones [37, 38], it remains unclear how these interactions should be delivered within an immersive setting.

In summary, this paper makes the following contributions:

- We demonstrate a shared control system that supports efficient text editing in virtual reality via flexible voice commands, and optionally leveraging additional contextual information expressed through touch or gaze.
- We evaluate this system and study how user behavior and preferences are influenced by edit type (e.g. insertion, deletion, extension) and expression of context in different ways.

2 Related Work

A wide variety of methods for text entry in virtual reality have been explored and recent works [9, 28, 31] have demonstrated entry rates approaching that of conventional input methods using a keyboard or smartphone. The ancillary tasks of text correction and editing have received relatively little attention in the literature, despite these tasks often being time consuming and often frustrating for users. The recent advances in large language models (LLMs) also offers opportunities for approaching the correction and editing task differently. Similarly, the advances in speech recognition also provide a potentially efficient and flexible method for correcting and editing text. In this section, we briefly review prior work on text correction and editing in virtual reality, before examining entry and editing in other settings with the assistance of LLMs, and finally review efforts using voice to support text editing.

2.1 Text Correction and Editing in VR

Among the limited prior work in this area, most investigations have primarily focused on text correction, where the main task is to correct word-level errors. Robertson and Black [27] make an important distinction between text correction and text editing, and note that text editing is a, "complex cognitive skill that requires significant planning." Nevertheless, a variety of strategies and interactions for performing text correction in VR have been explored.

Adhikary and Vertanen [2] examine correcting text inputted with voice by means of a word confusion network. The interface allows the user to select alternative hypotheses from the word confusion network to make corrections. This approach builds on prior work by Vertanen and Kristensson [33] who do the same using a smartphone.

Li et al. [22] and Hu et al. [13] explore the low level task of placing a text cursor in VR, as an essential precursor task in making corrections. Dudley et al. [8] explore five alternative strategies for correcting word errors in VR, including at the character level by using a text cursor, but also at the word level with interactions alternatively based on touch, gaze, raycast and another indirect hand-tracking based method. Li et al. [21] study the use of hand gestures to support text editing in virtual reality, assigning different gestures to common actions like cut, undo and delete. Compared to menu-based selection of editing actions, the approach using gestures was found to be more efficient.

While Adhikary and Vertanen [2] do employ speech for text input, text editing is performed without speech through mid-air selections. The other work reviewed above similarly focuses on both low and high-level interactions for text correction and editing but does not incorporate voice-based commands. To our knowledge, we are the first to specifically study the opportunities and challenges related to speech-driven text correction and editing in VR.

2.2 Text Editing with LLM-Based Assistance

LLMs offer great potential for facilitating the text editing task. Recent efforts have examined how the text editing capabilities of

LLMs can be more closely integrated into the text composition workflow [39, 40]. For example, Zindulka et al. [40] explore different interactions and visualizations for use on a smartphone to trigger an LLM to shorten or expand text. In the same vein, another work by Zindulka et al. [39] examines the opportunities for including an 'improve' button to support drafting of emails on a smartphone. Earlier work by Faltings et al. [10] demonstrated a custom transformer-based model that supported text editing by commands. For example, the user might provide a command to 'expand' the text or give more specific commands like 'add years in office' to a blurb being written about a president.

Tang et al. [32] provide a recent review of the different ways in which LLMs can be integrated to enhance interactions in extended reality. The review is mainly focused on the manipulation and understanding of the scene or objects in the scene but does briefly mention text entry citing three papers [4, 5, 12]. Examining these three papers in more detail, Cui et al. [5] do not actually employ a LLM but rather a conventional bi-gram language model to support gaze-based text input. Guo et al. [12] simply demonstrate that their text entry method using gaze and controller button presses can be used to make queries to ChatGPT. Chen et al. [4] do in fact incorporate an LLM and study the opportunities it affords for improving the efficiency of text input in VR. They study three alternative means of inputting text with reduced keystrokes: *simplified spelling* where one could type 'tnger' and still obtain 'teenager'; *content prediction* where the remainder of the sentence is predicted and available for insertion; and *keyword-to-sentence* where one could simply enter the most important words of a sentence and the LLM would generate a complete sentence. The prediction and keyword based approaches are found to offer superior typing efficiency compared to the simplified spelling method. This keyword approach is also demonstrated by Shen et al. [29] in a non-immersive setting.

Although not strictly using an LLM, prior work has explored methods for facilitating text editing on a smartphone through intelligent inference methods. JustCorrect [6] and Type, Then Correct [35] both demonstrate methods for making rapid corrections by retyping the erroneous word. In this paper, we examine a variety of editing tasks that bridge the more open ended text editing approaches demonstrated by Zindulka et al. [39, 40] and Faltings et al. [10] with the finer-grained editing techniques like JustCorrect [6] and Type, Then Correct [35]. We also study the implications of performing such tasks within an immersive setting with higher levels of input uncertainty.

2.3 Voice-Based Text Editing

Text editing via voice has been more widely studied in a smartphone context [36–38]. Zhao et al. [36] explored voice and touch for text editing on a smartphone. The voice and touch method was found to offer reductions in editing and correction relative to voice alone and touch alone methods. More recent work by Zhao et al. [38] incorporates an LLM to further facilitate correction. Similar to their earlier work without an LLM [36], Zhao et al.'s Tap&Say [38] allows the user to tap on or near an erroneous word and re-speak the word or phrase. Tap&Say has some similarities with an earlier approach by Fan et al. [11] given the moniker Just Speak it. Just Speak It [11] incorporates an LLM that helps process voice-based edits expressed by re-speaking or by what Fan et al. refer to as 'descriptive commands' such as 'delete A'. However, Just Speak It appears to specifically check for keywords like 'delete' in order to handle such descriptive commands, thereby imposing a constrained syntax. EyeSayCorrect [37] combines gaze with voice to support text correction on a smartphone. The user focuses their gaze on or near the erroneous word and re-speaks the correct word or phrase. Li et al. [20] investigate a more general task they refer to as interactive dictation. They demonstrate a technique that allows users to interleave spoken editing commands directly within their dictation. For example, a user can say, "the event on the 23rd, on Friday the 23rd" with the system inferring that the desired output should be "the event on Friday the 23rd."

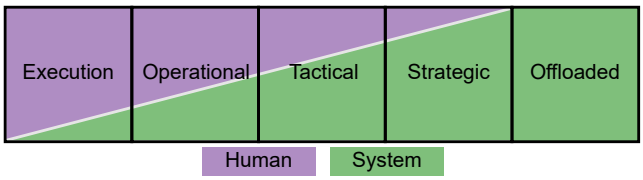


Figure 2: The spectrum of Shared Control for text editing.

There has been less work on voice-based editing in an immersive setting. Pick et al. [25] demonstrated a system for speech-based text input in a CAVE-like immersive environment. An interface was provided that allowed the user to select words or spaces in the transcribed text field. The user could then make edits by, for example, selecting a word and choosing from alternatives or speaking the word again, deleting a word, or inserting a word by pressing a button located at a space.

Our work complements the efforts of Zhao et al. [36–38] by exploring whether an LLM employed as an intelligent text correction agent can offer greater flexibility and accommodate a wider range of editing tasks. In contrast to Zhao et al. [36–38], we also study these interactions within an immersive setting that has different input characteristics from a smartphone, as summarized later in Section 4. Furthermore, we contextualize this investigation through the informative lens of shared control as detailed in the next section.

3 Shared Control for Text Editing

In this paper we suggest *shared control* [30] as a lens to assist us in understanding the collaboration between the user and an intelligent text correction and editing system interacted with through touch, gaze and speech. Shared control provides a spectrum of levels of control at which a user may operate when collaborating with a machine. Figure 2 offers a diagrammatic representation of how this spectrum of shared control might manifest in the context of text editing, adapted from the framework proposed by Abbink et al. [1].

We view shared control for text editing as five levels (Figure 2):

- **Execution.** At the execution level of shared control, the system may assist the user by giving insight or automating minor tasks such as correcting typos. The simplified spelling approach demonstrated by Chen et al. [4] and described in Section 2.2 provides a good example of this level of shared control.

- **Operational.** At the operational level of shared control, the user may give specific instructions to the system for it to perform. For example, the user may indicate that one word needs to be changed to a different word and the system completes this task. Tap&Say [38] and Just Speak It [11] (see Section 2.3) provide examples of an operational level of shared control, with the user only required to re-speak a word or phrase to correct an error.
- **Tactical.** At the tactical level of shared control, the user may give guidance to the system for it to incorporate. For example, the user may indicate that a body of text should be revised for clarity or to avoid acronyms and the system must determine which portions of text contravene this guidance, and make associated changes. Our system described later in Section 5 supports such free-form guidance commands.
- **Strategic.** At the strategic level of shared control, the user may express or the system may infer, an intent for the system to act upon. For example, the user may express an intent to give an example to accompany a specific point, and the system must then determine and insert a suitable example. This level of control is exemplified well by the work of Faltings et al. [10] (see Section 2.2) wherein the user can ask for specific details to be added to a body of text.
- **Offloaded.** The offloaded level is system intervention in editing that involves no user input at all. For example, it is conceivable that a system could automatically refine previously entered text for accuracy, consistency and style while the user moves on to other tasks. This offloaded level reflects a point of difference from the Abbink et al.'s [1] framing of the shared control spectrum.

4 A Function Model of Text Editing in VR

The high-level task of text editing in virtual reality is in many respects similar to editing text in conventional interaction settings, such as on a smartphone. However, there are several important points of distinction that can be reasonably expected to impact the ideal design of interactions for this task in virtual reality.

First, mid-air text input speeds in VR are generally slower than those achievable on a smartphone or physical keyboard. This rate limit affects the potential choice between following an edit strategy based on deletion then retyping versus an edit strategy that focuses on correction. Second, controller-free interaction relying on hand tracking can make it difficult to precisely indicate the desired edit location. This reduced precision in control influences the feasibility of low level editing strategies that require accurate placement of a text caret. Third, modern VR headsets provide additional input modalities, such as gaze, that are not typically available in conventional interaction settings. These additional input modalities can potentially be leveraged to improve the usability of interactions and the efficiency of the editing task.

To help tease out these issues we present a function model for the task of text editing. A *function model* is a diagram that identifies the key functions in a system and the interrelations of those key functions. By function we here mean *what* must be carried out—as opposed to a particular solution, which tells us *how* to achieve a particular function. Inspired by prior work using function models in HCI [18] and extended reality [16, 17], function models are lightweight models that allow us to reason about an interactive system at a largely solution-neutral level.

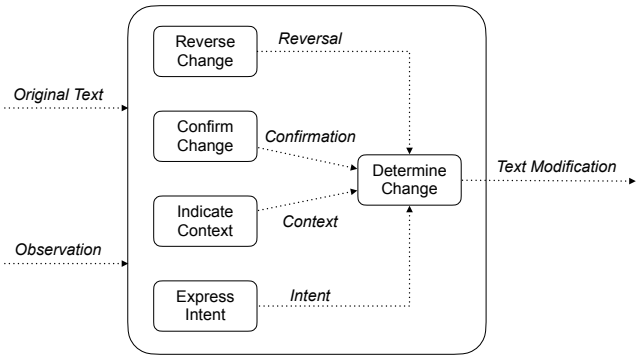


Figure 3: Function model of the text editing task.

Figure 3 shows a function structure model [23] of the text editing task. Functions are indicated in boxes and dashed arrows indicate signal flows. The central function is Determine Change, which acts based on four subfunctions: (1) Reverse Change; (2) Confirm Change; (3) Indicate Context; and (4) Express Intent. A user interaction with the system is captured as an *Observation*. The precise details of this observation determine which subfunction is triggered. For example, a user utterance of 'Undo' may trigger the Reverse Change subfunction and thereby issue a *Reversal* signal to the Determine Change function.

An effective interface and text editing system should implement these functions. The decomposition of the system in this way serves to highlight how abstract functions might potentially be implemented through different methods, and which functions offer the greatest design freedom. Within this work, we primarily study alternative design choices related to the core function of Indicate Context. We study how *Observation* signals captured through either touch, gaze, or speech can be decoded into some form of text modification based on the *Original Text*. We also examine how these choices influence the ways that users express intent. As detailed further in the following section, we implement three alternative methods for indicating the context of an edit.

5 Text Editing System

We developed a prototype text editing system to provide the basis for our study. The system consists of three key components: the interface, the voice command transcription system, and the edit interpretation system. Each of these system components is described in further detail in the following subsections.

5.1 Interface

The application interface is shown in Figure 4. The implemented interface is based on a simplified chat application. The top message provides the task instructions. The middle message is the ideal text. The bottom message is the text that the participant must modify. For the study described in this paper, the square chat window shown in Figure 4 was 360 × 360 mm, and positioned at a distance of 320 mm

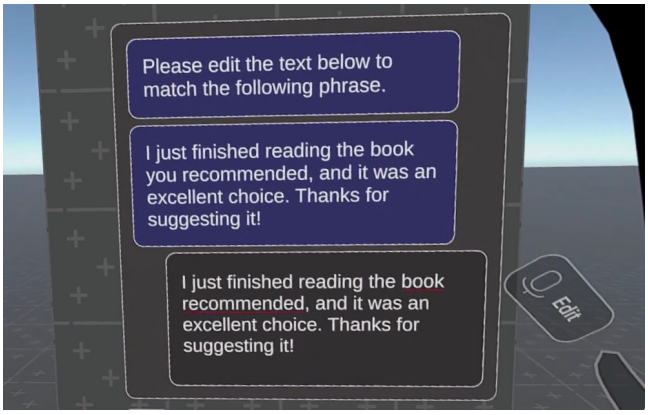


Figure 4: The task interface in the user study. The interface is based on a simplified chat application. The top message provides the task instructions. The middle message is the ideal text. The bottom message is the editable text field.

from the user. The text font size was set such that the capital 'T' character shown in Figure 4 was approximately 12 mm high.

To issue a voice command, the user performs a double pinch gesture. The double pinch gesture involves touching the index finger and thumb tips together twice in quick succession. An audible tone and visual label attached to the index fingertip indicates that the system is now listening for a voice command. The spoken voice command is transcribed and interpreted as detailed in the following subsections. The text field is then updated with the revised text.

As outlined in Section 4, the interface supports three alternative methods for indicating edit context. We subsequently refer to these methods as: VoiceOnly, Voice+Touch, and Voice+Gaze.

5.1.1 VoiceOnly. The VoiceOnly method provides no additional method for indicating context and so, if necessary, context must be expressed directly as part of the voice command. Users are free to express voice commands in any phrasing they choose.

5.1.2 Voice+Touch. The Voice+Touch method optionally allows the user to indicate context by highlighting words in the text field using a virtual touch-based interaction. The user moves their index finger into the text field plane and can draw one or more lines to highlight words. This interaction seeks to mimic the familiar user experience of marking up a document with a highlighter. The highlighted words are passed to the Edit Interpretation system. Figure 5 provides an example of a participant using the touch interaction seeking to indicate the word 'completed'.

5.1.3 Voice+Gaze. The Voice+Gaze method captures user fixations on the words in the text field using the gaze tracking functionality provided by the Meta Quest Pro. When the user issues a voice command, the word receiving the longest period of cumulative fixations is passed to the Edit Interpretation system. The accumulation of fixation times is reset after each edit interaction. The user's gaze location is visualized on the text field as a subtle cursor implemented as a weak spotlight. Fixations on a word in the text field also produce a subtle change in text color from white to light gray. This Voice+Gaze method therefore incorporates gaze

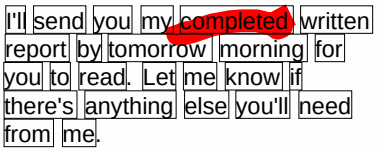


Figure 5: Context information obtained from touch. In this example, the participant has explicitly indicated 'completed' since it should be deleted. The word boundaries shown are not visible to the user, as seen in Figure 4.

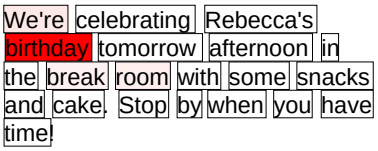


Figure 6: Context information obtained from gaze. In this example, the participant has implicitly indicated 'birthday' since it should be replaced with 'promotion'. The word boundaries shown are not visible to the user, as seen in Figure 4.

as an implicit signal for indicating context. This choice, in combination with the use of subtle visual feedback, is made to minimize user distraction and to avoid the inefficiencies associated with more explicit gaze-based interactions such as dwell. However, this does mean that the gaze method is generally characterized by less precise indication of context due both to the implicit capture of fixations as well as potential inaccuracies stemming from individual calibration. Figure 6 provides an example of the participant's gaze distribution over words in the message when the edit command is issued.

5.2 Voice Command Transcription

The voice commands are transcribed using the Meta Quest Voice SDK and Wit.ai. This configuration is consistent with developer guidance for adding voice-based interaction to Meta Quest applications.[1] We use the `AppDictationExperience` feature offered by the SDK which does not process text with Natural Language Understanding (NLU). We use this feature as opposed to the alternative `AppVoiceExperience` feature, as the spoken instructions given by users may not follow an expected format phrased as an explicit command. For example, in pilot testing we found it was common for users to re-speak just the words relevant to the target edit.

The transcription system automatically stops listening after a set period of silence, and a termination tone is played. The visual 'Edit' label attached to the fingertip also disappears. To trigger the system to listen again, the user must perform another double-pinch gesture. If the user performs a double-pinch gesture and no voice is detected, the system plays an alternative termination tone and the visual label disappears. We use the default configuration values provided for the `AppDictationExperience` component.

[1] Meta Voice SDK: Integrate Voice documentation.

Table 1: Examples of the five different edit types.

Edit Type	Target Text	Before Editing
Missing Word	I just finished reading the book you recommended, and it was an excellent choice. Thanks for suggesting it!	I just finished reading the book recommended, and it was an excellent choice. Thanks for suggesting it!
Extra Word	The meeting tomorrow afternoon starts at 3 PM and should last approximately two hours. Let me know if you will be available.	The meeting tomorrow afternoon starts at 3 PM and should last approximately ~~for~~ two hours. Let me know if you will be available.
Incorrect Word	I'm not sure whether I'll attend the conference this weekend. Do you think it will be worthwhile?	I'm not sure ~~weather~~ I'll attend the conference this weekend. Do you think it will be worthwhile?
Rephrasing	I updated the budget document with the latest figures, so please use the new version for your analysis. Let me know if you have questions.	I updated the ~~project outline~~ with the latest figures, so please use the new version for your analysis. Let me know if you have questions.
Extension	Could you send me your presentation slides before end of day? I'd like to review them tonight to prepare thoroughly.	Could you send me your presentation slides? I'd like to review them tonight to prepare thoroughly.

5.3 Edit Interpretation

The edit commands, original text, and any additional contextual information are passed to the Edit Interpretation system implemented using OpenAI's GPT-4o. We use OpenAI's `completions` API and send requests using a `UnityWebRequest`. The web request returns the edited sentence and this is inserted into the text field. The full prompt given with the API request is shown in Appendix A. There is no strict syntax imposed on the voice commands issued by the user. The flexibility afforded by the edit interpretation system thus allows users to freely explore different levels of shared control.

6 User Study

The system described in the previous section is used as the platform for exploring user behaviors and preferences for the task of text editing in virtual reality. Specifically, we seek to answer the following four research questions (RQs):

RQ1: What interaction behaviors do users exhibit when performing edits in VR with speech commands?
RQ2: How is this behavior influenced by the available means of indicating context?
RQ3: How is this behavior influenced by the type of edit required?
RQ4: What are users' attitudes towards offloading aspects of the task of editing text to an intelligent agent?

6.1 Task

The editing task completed by participants in the study required the modification of an existing body of text such that it matched a presented ideal form. The body of text was formulated as a two sentence text message. Much prior work evaluates text correction and editing interactions with a single sentence. We intentionally involve more than one sentence in the task due to the inherent complexity this introduces in resolving potentially ambiguous edit commands. The specific formulation of the messages was loosely based on the Enron mobile message dataset [34] and sought to reflect the type of two-sentence message one might send to a friend or colleague. The ideal form of the message was then corrupted according to one of the following five required edit types: (1) missing word, (2) extra word, (3) incorrect word, (4) rephrasing, and (5) extension. An example of each message type is given Table 1.

The full set of text messages used in the study is provided in the supplementary material. Participants completed five practice tasks and 20 test tasks in each condition. The five practice messages were the same for all conditions while three unique sets of 20 messages were randomly allocated to the three conditions. As shown in Figure 4, the location of the required edit was indicated at the start of each task by a red underline. Participants had to edit the text field until it exactly matched the ideal text. If the submit button was pressed without the texts matching, an error tone would play and the task would not advance. In pilot testing we found that participants would sometimes make a mistake or the voice command would be misinterpreted, leading to extensive unintended changes in the text field. To allow recovery from such situations, we also included a button that resets the current task to its original state.

6.2 Protocol

The experiment began with participants providing informed consent and completing a short demographics questionnaire. Participants were then provided with an introductory briefing and detailed task instructions. The order of the three conditions was fully counterbalanced across the 18 participants. The Quest Pro gaze tracking was calibrated immediately before commencing the Voice+Gaze condition. Participants were encouraged to take a short break after the first 10 test edit tasks in each condition.

At the end of each condition, participants completed the raw NASA-TLX and System Usability Scale surveys. After all three conditions, participants also completed a final questionnaire capturing their subjective ratings and views of each interaction method. Participants also responded to two free-text questions: (1) How did you feel about offloading some parts of the editing task to the intelligent agent? (2) What strategies did you apply? Were these strategies different for the different methods?

6.3 Participants

We recruited 18 participants (7 female, 11 male; median age of 27 [23,53]) from our institution through convenience sampling

and public postings. The study was approved by our departmental research ethics committee. The entire study took approximately one hour to complete and participants received an Amazon voucher as a token of appreciation for their time.

Participants were asked to indicate 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 2 and ranged from 1 to 5. All participants were right handed despite this not being a criterion for participating in the study.

7 Results

In this section we present the results of the user study, covering the different aspects of editing performance and editing experience. We also include a general synthesis of the subjective participant feedback, and our broader observations of participant behavior.

7.1 Editing Performance

A key starting point for examining the impact of the three alternative interaction methods is the mean edit time. The edit time reflects the relative efficiency of the different context indication methods. This edit time includes all time elapsed between when the message is first presented, until the participant presses the submit button. Since the corrected message must exactly match the stimulus, the edit time potentially also includes one or more follow up actions if the first command did not fully achieve the desired edit.

Figure 7 plots the mean edit time across participants for each interaction method. To test for a significant effect of the interaction method, we ran a repeated measures ANOVA. We found a significant effect associated with interaction method ($F_{2,34} = 5.425$, $\eta_p^2 = 0.242$, $p = 0.009$), with VoiceOnly found to be significantly faster than Voice+Gaze ($p = 0.009$). There was no significant difference between Voice+Touch and the other two methods. Nevertheless, Figure 7 suggests that Voice+Touch was slightly less efficient than VoiceOnly but slightly more efficient than Voice+Gaze.

It is reasonable to expect that Voice+Touch might be slower than VoiceOnly due to the fact that the condition involves an additional word selection action before speech commands are issued. By contrast, Voice+Gaze might in principle be expected to avoid this time penalty since gaze information is incorporated implicitly without the need for additional user interaction.

As mentioned above, the mean edit time incorporates any follow up actions required to complete the editing task. To offer insight on how efficiency is impacted by the need for rework, we examine the mean number of voice commands per task. Figure 8 plots the mean number of voice commands used by participants in each condition. We can see from Figure 8, that Voice+Gaze typically required more commands per task than VoiceOnly and Voice+Touch. A repeated measures ANOVA found a significant effect associated with interaction method ($F_{2,34} = 5.276$, $\eta_p^2 = 0.237$, $p = 0.010$), with participants using significantly more commands in Voice+Gaze than VoiceOnly ($p = 0.013$). This may in part explain the reduced efficiency observed in Figure 7. The most likely cause of the need for rework is reduced accuracy in the editing operations triggered by the voice commands and the additional gaze signal. Since the gaze signal is the main point of difference between VoiceOnly and Voice+Gaze, it is likely that inaccuracy in gaze-based context indication is the main deficiency.

We can examine a direct proxy for gaze-based context indication accuracy by examining the offset between the indicated word and the word or words that are known to require editing. We subsequently refer to this measure as the minimum offset distance. The minimum offset distance is the shortest distance between the edge of the bounding box for the indicated word or words and the edge of the bounding box for the word or words that must be edited. If any of the indicated or target edit words align, the minimum offset distance is 0. Figures 5 and 6 show examples of the word boundaries that are used as the bounding boxes. The height of the bounding boxes is fixed at approximately 20 mm and the horizontal spacing between words is typically in the range of 5 to 6 mm.

To constrain our analysis to easily interpretable user intent, we focus only on the first voice command within an edit task. Figure 9 plots the mean of the minimum offset distance for the Voice+Touch and Voice+Gaze conditions. The distances are significantly larger for Voice+Gaze, suggesting that participants experienced considerable difficulty in accurately indicating the relevant context word. As noted above, an adjacent word in the same row would yield a minimum offset distance of approximately 5 mm. Figure 9 shows that the median minimum offset distance across participants was 4.51 mm, suggesting that a large portion of gaze-based context indications were on the adjacent word or even further away. This may be due to issues with calibration or unexpected participant fixation behaviors on the text field. The extent to which participants faced difficulty in this condition clearly varied. For comparison we can look at the subjective ratings reported later in Section 7.2, and consider just the subset of five participants who strongly agreed with the statement that Voice+Gaze 'made it easy to edit text accurately.' For this group, the median offset distance was 2.65 mm.

Another aspect impacting efficiency is the verbosity of issued voice commands. Figure 10 plots the mean number of words used in each voice command. Participants used fewer words per command in the Voice+Touch and Voice+Gaze conditions, than in the VoiceOnly condition. A repeated measures ANOVA found a significant effect associated with interaction method ($F_{2,34} = 27.524$, $\eta_p^2 = 0.618$, $p < 0.001$), with participants using significantly fewer words per command in Voice+Touch ($p < 0.001$) and Voice+Gaze ($p < 0.001$) than in VoiceOnly. This suggests that at a command-level, participants were able to gain some efficiency from the alternative forms of context indication.

The performance analysis presented above groups all five edit types together. In practice, these different edit types represent a spectrum of difficulty. Figure 11 plots the mean edit time per participant for each edit type with all conditions grouped together. The plot shows that the *Extension* edit was by far the most involved, while fixing an *Incorrect Word* was the least time consuming.

7.2 Editing Experience

As outlined in Section 6.2, we capture measures of the usability of the three different interaction methods through the raw NASA-TLX and SUS questionnaires, as well as our targeted questionnaire examining speed, accuracy, comfort, and control. We also capture the participants' ranking of their preference over the three methods.

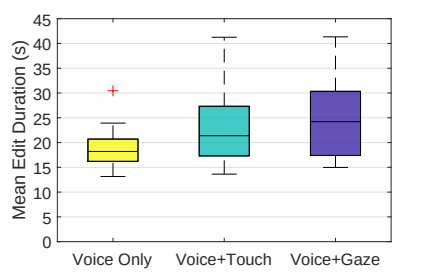


Figure 7: Mean edit time.

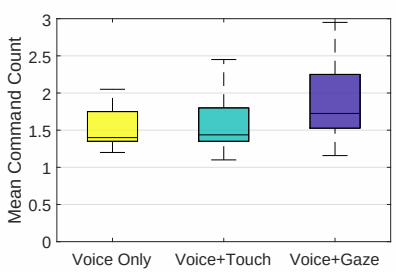


Figure 8: Mean command count.

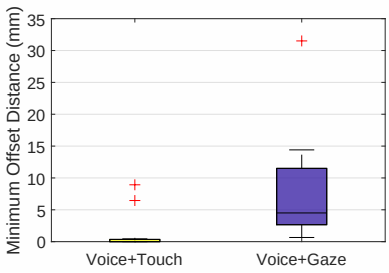


Figure 9: Minimum offset distance.

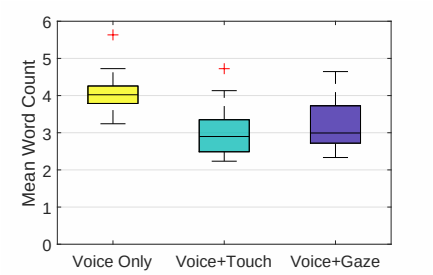


Figure 10: Mean word count.

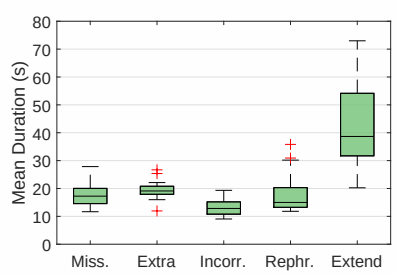


Figure 11: Mean edit time by edit type.

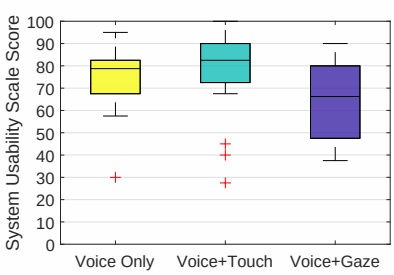
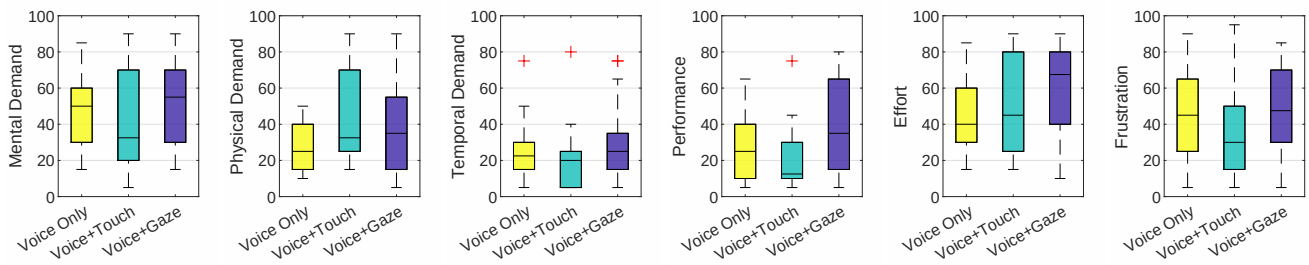


Figure 12: System Usability Scale scores.



(a) Mental Demand **(b) Physical Demand** **(c) Temporal Demand** **(d) Performance** **(e) Effort** **(f) Frustration**

Figure 13: Raw NASA-TLX responses.

Table 2: Median Likert ratings of statements covering perceived speed, accuracy, comfort and control from 1-Strongly Disagree to 5-Strongly Agree.

Aspect	VoiceOnly	Voice+Touch	Voice+Gaze
Speed	4	4.5	3
Accuracy	4	5	3.5
Comfort	4	4	3
Control	4	5	3

Table 3: Participant preference rankings.

Ranking	VoiceOnly	Voice+Touch	Voice+Gaze
1st	3	12	3
2nd	12	3	3
3rd	3	3	12

The raw NASA-TLX results are plotted in Figure 13. No statistically significant effects were observed. Nevertheless, the plots over the different scales of perceived workload suggest that Voice+Gaze is generally associated with elevated ratings compared to VoiceOnly and Voice+Touch, i.e. higher workload. Compared with VoiceOnly, Voice+Touch was rated as having less mental and temporal demand, causing less frustration, and yielding better perceived performance, but being more physically demanding and requiring more effort. This result is consistent with the additional physical effort required in Voice+Touch to highlight words in the text field.

The SUS results are plotted in Figure 12. Voice+Touch received a median SUS rating of 82.5, which reflects very good usability [3]. VoiceOnly was the next highest in terms of SUS rating, and then with Voice+Gaze receiving a median rating of less than 70. Since ratings for Voice+Touch were not normally distributed, we ran a Friedman test to determine if there was a significant effect associated with interaction method. The effect was found to be significant

($\chi^2(2) = 6.609, p = 0.037$), with Voice+Touch yielding significantly higher scores than Voice+Gaze ($p = 0.032$).

Table 2 summarizes the participant ratings on statements related to speed, accuracy, comfort and control. There are some observable parallels between the NASA-TLX ratings and the participant responses to the statements. Voice+Gaze was generally found to be weak across all aspects. Voice+Touch was generally found to be very accurate and offer good controllability, but lacked somewhat in terms of speed and comfort.

Table 3 summarizes the participant preference rankings for the three interaction methods. Voice+Touch was ranked first by 12 of 18 participants. The next most preferred method was VoiceOnly with 12 of 18 participants ranking it second. Voice+Gaze was then the least preferred method. These preference ratings are largely consistent with the SUS ratings, where Voice+Touch received the highest SUS score and Voice+Gaze received the lowest.

The written comments from the post-study questionnaire also offer various insights about the experience of working with the intelligent agent, and the strategies developed. As P14 commented, "[offloading some parts of the editing task] felt convenient and efficient, especially because I didn't have to give specific commands e.g. I could just say 'fix spelling' or 'change spelling' without specifying exactly to what." P15 noted that the, "intelligent agent could infer context automatically so only simple commands were sufficient." While these comments are generally positive, this was not the universal experience of participants. P18, whose subjective experience ratings were an outlier among the participants in the study commented that, "The agent is not intelligent. It is hallucinating and sometimes it manages to hallucinate correctly." This observation highlights the fact that different users have very different attitudes towards involving an intelligent agent in the text editing task. Some participants also experienced difficulty with voice commands due to their accent. As P16 noted, "Really helpful but it could be better if the different accent will be understandable."

Multiple participants commented that the strategy they employed in each condition mainly differed in terms of how many words they used. For example, P6 noted that, "For Voice+Touch, I avoided saying contextual info. For Voice and Voice+Gaze, I had to provide more context especially for adding text." Another common observation was that in instances of the system failing to provide the correct outcome, participants would use more words to express their desired action. As commented by P5, "I would repeat with a longer chunk of the sentence."

7.3 Editing Behaviors

The lack of constraints on the syntax of the voice commands allowed participants to explore different approaches and strategies. Table 4 summarizes the variation in voice commands for the extra word edit type that required a deletion to resolve. The count of each command form is given for each interaction method. The '<other>' placeholder represents other command forms that are not easily categorizable. These are mainly command forms based on re-speaking the portion of text around the required edit location or commands arising due to the need for subsequent rework. Table 4 highlights how the different conditions impacted the framing of commands in terms of whether or not to include additional context about the edit target. For Voice+Touch, 50.6% of the commands used by participants for this edit type did not include any spoken context (e.g. 'Delete', 'Get rid of that', and 'Remove'). For Voice+Gaze, the proportion was 27.4%. Some context-free commands were used in the VoiceOnly condition (2.6%), but this is likely due to user error.

Table 4: Variation in voice command for extra word edit type. Counts shown in square brackets. '*' indicates wildcard. VO: VoiceOnly, V+T: Voice+Touch, V+G: Voice+Gaze.

Command	VO	V+T	V+G
Delete	1	37	31
Delete *	50	19	34
Delete the word *	5	-	5
Remove *	18	13	18
Remove the word *	1	2	1
Remove	2	7	-
Get rid of that	-	1	-
Remove this word	-	-	1
<other>	39	10	27

8 Discussion

The results presented in the previous section reveal several interesting trade-offs between efficiency, usability and control. We now revisit the research questions posed in Section 6.

RQ1. What interaction behaviors do users exhibit when performing edits in VR with speech commands? We observed considerable variability in the strategies employed by participants in completing the task. Some participants expressed edit commands with high specificity, while others depended more on the intelligence of the correction system. This is most apparent in Figure 10, which highlights how some participants issued more verbose commands, averaging more than four words per command, while others were more terse, approaching an average of two words per command. Table 4 also highlights the variation in voice commands used by participants for the extra word edit type. This variation illustrates how eliminating the need for specific syntax in voice commands allowed participants to explore their own preferred formulations. The general strategy observed and mentioned by participants was to be as concise as possible in the first instance. If the command did not produce the desired outcome, participants would then add more and more context (see comments of P5 in Section 7.2).

RQ2. How is this behavior influenced by the available means of indicating context? Participants exhibited different behaviors depending on the interaction method available to them. As expected, the VoiceOnly method generally required participants to be more verbose in their commands due to the need to more explicitly express context. This is most apparent in the results for the mean word count per command (see Figure 10). Nevertheless, the VoiceOnly condition was found to be the most efficient in terms of mean edit time (see Figure 7). Voice+Touch allowed participants to be more concise as the context could be explicitly indicated through the touch interaction. Voice+Gaze also resulted in more concise commands but the inaccuracies in the gaze-based context indication meant that some participants would often also include spoken context (see Voice+Gaze column in Table 4).

RQ3. How is this behavior influenced by the type of edit required? The edit type is a key factor influencing how long it takes to perform an edit. Figure 11 shows how the mean edit time varied for each of the five different edit types. The most complicated edit type studied required the extension of text. This task is inherently more involved as it requires editing beyond the individual word level. This influenced user behavior by demanding more effort to indicate the context of the edit. For the simpler edit types, such as extra word and incorrect word, participants demonstrated that single word commands such as 'Delete' could be effective (see Table 4).

RQ4. What are users' attitudes towards offloading aspects of the task of editing text to an intelligent agent? Participants showed good willingness to offload aspects of the editing task to the intelligent correction system. The written comments summarized at the end of Section 7.2 highlight how participants appreciated the convenience and efficiency afforded by the system. Participants indicated strong agreement with the statement that they had control over the editing process for the VoiceOnly and Voice+Touch conditions. The median SUS scores were also strong for both the VoiceOnly and Voice+Touch conditions. The majority of participants (12/18) rated Voice+Touch as their most-preferred condition, suggesting that the ability to explicitly indicate context with touch was appreciated. This preference was observed despite the fact that VoiceOnly was marginally more efficient in terms of mean edit time. In summary, we find favorable attitudes to offloading aspects of the task, but with participants also eager to retain control over other aspects.

8.1 Implications

Our findings suggest several implications for the design of efficient and usable text editing methods for use in VR.

Participants favored explicit interaction for context indication. While the VoiceOnly condition resulted in the fastest editing times, participants preferred complementing voice with touch indications, indicating that users desire additional control. We therefore suggest text editing interfaces provide users with an alternative modality that allows users to provide additional context explicitly.

Participants developed their own voice command formulations. The flexibility of the voice transcription and LLM-supported editing system meant that participants did not have to follow a strict syntax. This gave freedom to participants to explore and identify command formulations that they found to be effective. We suggest that this approach offers good walk-up usability for unfamiliar users. This study protocol required participants to perform only five practice edit tasks, and the SUS scores obtained for VoiceOnly and Voice+Gaze suggest good usability despite the limited familiarity.

The method used to implicitly capture context from gaze was not effective. The Voice+Gaze condition yielded the slowest editing times and was also least preferred by participants. We conjecture this is due to the uncertainty inherent in using gaze as an implicit information channel. It is possible that improvements to gaze tracking and calibration procedures, as well as more advanced methods for extracting implicit context may address this observed deficiency.

8.2 Limitations and Future Work

While our findings offer insight for the development of voice-based text editing techniques in VR, we recognize several key limitations and avenues for future work.

First, in this paper we studied text edits for messages comprising exactly two sentences. Further, the five edit tasks explored were selected to give some diversity in the user actions required and variation in difficulty, but they do not exhaustively represent the range of all possible text editing tasks. We have observed that performance and user strategies differ for different edit tasks. We also anticipate that different editing tasks are likely to have different ideal operating points on the spectrum of shared control described in Section 3. Longer form text also has the potential to give rise to greater ambiguity in user inputs. It would therefore be interesting to expand the scope of this work and investigate longer text and different types of edits, such as moving one sentence from one paragraph to another. As highlighted by the function model presented in Figure 3, there are also two other key functions to consider: Reverse Change and Confirm Change. It is important to study how providing users with the ability to reverse and confirm their edit actions may impact the interaction strategies applied.

Second, there are several possible design choices inherent in bestowing users with control capabilities based on explicit touch input and implicit gaze fixations. Despite the generally poorer performance of Voice+Gaze, Table 3 highlights that three participants still listed the condition as their top preference. This suggests there are some redeeming qualities that may be better exploited if the context indication accuracy can be improved. In addition, it would be interesting to investigate whether touch and gaze may be combined to provide higher performance or increase user satisfaction. Other common interaction paradigms, such as ray-based cursor control and text selection, may also be studied and compared.

Third, text editing is embedded within wider working practices. Although the experimental task used sought to reproduce the experience of editing text in a messaging context, participants were not editing text they had generated themselves. This was necessary to exercise experimental control and is similar to conventional text entry studies that employ a transcription task rather than a composition task. Nevertheless, the fact that users held no prior knowledge of the text they were asked to edit likely impacted their behavior and performance. Future work should study user behaviors when editing self-composed text. Extending this further, we also see value in studying LLM-assisted text editing in a variety of use contexts, ranging from replying to a message while mobile to revising text for a presentation. Although the focus of the current study was on the evaluation of alternative methods for indicating context to inform voice-based text editing, a broader study would also benefit from a comparison against conventional character and word-level editing approaches [8].

9 Conclusions

Text editing is an important but understudied area of text entry, in particular in virtual reality. In this paper we explore the task of editing text through a shared control lens. Specifically, we have investigated LLM-assisted text editing in VR in three conditions: (i) using voice commands; (ii) indicating words using touch; and

(iii) implicitly indicating words using gaze. Our results show that users develop effective strategies for collaborating with the LLM-based agent to make edits. Voice-only resulted in the fastest edit times, followed by voice with touch indications. However, participants preferred voice with touch overall. Implicit gaze assistance yielded the slowest edit times and was the least preferred text editing method. Our findings suggest that voice-based collaboration with an LLM can provide an effective text editing workflow in VR that is complementary to conventional editing approaches.

References

[1] David A. Abbink, Tom Carlson, Mark Mulder, Joost C. F. de Winter, Farzad Aminravan, Tricia L. Gibo, and Erwin R. Boer. 2018. A Topology of Shared Control Systems—Finding Common Ground in Diversity. *IEEE Transactions on Human-Machine Systems* 48, 5 (2018), 509–525. doi:10.1109/THMS.2018.2791570

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

[3] Aaron Bangor, Philip Kortum, and James Miller. 2009. Determining What Individual SUS Scores Mean: Adding an Adjective Rating Scale. *J. Usability Studies* 4, 3 (may 2009), 114–123.

[4] Liuqing Chen, Yu Cai, Ruyue Wang, Shixian Ding, Yilin Tang, Preben Hansen, and Lingyun Sun. 2024. Supporting Text Entry in Virtual Reality with Large Language Models. In *2024 IEEE Conference Virtual Reality and 3D User Interfaces (VR)*. 524–534. doi:10.1109/VR58804.2024.00073

[5] Wenzhe Cui, Rui Liu, Zhi Li, Yifan Wang, Andrew Wang, Xia Zhao, Sina Rashidian, Furqan Baig, IV Ramakrishnan, Fusheng Wang, and Xiaojun Bi. 2023. GlanceWriter: Writing Text by Glancing Over Letters with Gaze. In *Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems* (Hamburg, Germany) *(CHI '23)*. ACM, New York, NY, USA, Article 719, 13 pages. doi:10.1145/3544548.3581269

[6] Wenzhe Cui, Suwen Zhu, Mingrui Ray Zhang, H. Andrew Schwartz, Jacob O. Wobbrock, and Xiaojun Bi. 2020. 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)*. ACM, New York, NY, USA, 487–499. doi:10.1145/3379337.3415857

[7] John J. Dudley, Hrvoje Benko, Daniel Wigdor, and Per Ola Kristensson. 2019. Performance Envelopes of Virtual Keyboard Text Input Strategies in Virtual Reality. In *2019 IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*. 289–300. doi:10.1109/ISMAR.2019.00027

[8] John J. Dudley, Amy Karlson, Kashyap Todi, Hrvoje Benko, Matt Longest, Robert Wang, and Per Ola Kristensson. 2024. Efficient Mid-Air Text Input Correction in Virtual Reality. In *2024 IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*. 893–902. doi:10.1109/ISMAR62088.2024.00105

[9] J. J. Dudley, J. Zheng, A. Gupta, H. Benko, M. Longest, R. Wang, and P. Kristensson. 2023. 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 (nov 2023), 4567–4577. doi:10.1109/TVCG.2023.3320238

[10] Felix Faltings, Michel Galley, Gerold Hintz, Chris Brockett, Chris Quirk, Jianfeng Gao, and Bill Dolan. 2020. Text Editing by Command. arXiv:2010.12826 [cs.CL] https://arxiv.org/abs/2010.12826

[11] Jiayue Fan, Chenning Xu, Chun Yu, and Yuanchun Shi. 2021. Just Speak It: Minimize Cognitive Load for Eyes-Free Text Editing with a Smart Voice Assistant. In *The 34th Annual ACM Symposium on User Interface Software and Technology (UIST '21)*. ACM, New York, NY, USA, 910–921. doi:10.1145/3472749.3474795

[12] Jiajing Guo, Andrew Benton, Nan Tian, William Ma, Nicholas Feffer, Zhengyu Zhou, and Liu Ren. 2023. EyeClick:A Robust Two-Step Eye-Hand Interaction for Text Entry in Augmented Reality Glasses. In *Adjunct Proceedings of the 36th Annual ACM Symposium on User Interface Software and Technology* (San Francisco, CA, USA) *(UIST '23 Adjunct)*. ACM, New York, NY, USA, Article 91, 4 pages. doi:10.1145/3586182.3615814

[13] Jinghui Hu, John J. Dudley, and Per Ola Kristensson. 2022. 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)*. 640–645. doi:10.1109/ISMAR-Adjunct57072.2022.00132

[14] Per Ola Kristensson. 2009. Five Challenges for Intelligent Text Entry Methods. *AI Magazine* 30, 4 (2009), 85–85.

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

[16] Per Ola Kristensson. 2024. 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)*. IEEE, 419–423. doi:10.1109/VRW62533.2024.00081

[17] Per Ola Kristensson. 2024. Five Ways Function Models Enable Accessible Mixed Reality Interfaces. In *2024 IEEE International Symposium on Mixed and Augmented Reality Adjunct (ISMAR-Adjunct)*. IEEE, 223–227.

[18] Per Ola Kristensson and Thomas Müllners. 2021. 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* (Yokohama, Japan) *(CHI '21)*. ACM, New York, USA. doi:10.1145/3411764.3445566

[19] Per Ola Kristensson and Keith Vertanen. 2014. The Inviscid Text Entry Rate and Its Application as a Grand Goal for Mobile Text Entry. In *Proceedings of the 16th International Conference on Human-Computer Interaction with Mobile Devices & Services*. ACM, New York, NY, USA, 335–338. doi:10.1145/2628363.2628405

[20] Belinda Z. Li, Jason Eisner, Adam Pauls, and Sam Thomson. 2023. Toward Interactive Dictation. arXiv:2307.04008 [cs.CL] https://arxiv.org/abs/2307.04008

[21] Xiang Li, Wei He, and Per Ola Kristensson. 2025. Evaluating the Usability of Microgestures for Text Editing Tasks in Virtual Reality. arXiv:2504.04198 [cs.HC] https://arxiv.org/abs/2504.04198

[22] Yang Li, Sayan Sarcar, Yilin Zheng, and Xiangshi Ren. 2021. 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)*. ACM, New York, NY, USA, 12 pages. doi:10.1145/3411764.3445474

[23] Gerhard Pahl and Wolfgang Beitz. 2013. *Engineering Design: A Systematic Approach*. Springer Science & Business Media.

[24] Kseniia Palin, Anna Maria Feit, Sunjun Kim, Per Ola Kristensson, and Antti Oulasvirta. 2019. How do people type on mobile devices? Observations from a study with 37,000 volunteers. In *Proceedings of the 21st international conference on human-computer interaction with mobile devices and services*. 1–12.

[25] Sebastian Pick, Andrew S. Puika, and Torsten W. Kuhlen. 2016. SWIFTER: Design and evaluation of a speech-based text input metaphor for immersive virtual environments. In *2016 IEEE Symposium on 3D User Interfaces (3DUI)*. 109–112. doi:10.1109/3DUI.2016.7460039

[26] Mark Richardson, Fadi Botros, Yangyang Shi, Pinhao Guo, Bradford J Snow, Linguang Zhang, Jingming Dong, Keith Vertanen, Shugao Ma, and Robert Wang. 2024. StegoType: Surface Typing from Egocentric Cameras. In *Proceedings of the 37th Annual ACM Symposium on User Interface Software and Technology*. ACM, New York, NY, USA, Article 83, 14 pages. doi:10.1145/3654777.3676343

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

[28] Junxiao Shen, John Dudley, and Per Ola Kristensson. 2023. Fast and Robust Mid-Air Gesture Typing for AR Headsets Using 3D Trajectory Decoding. *IEEE Transactions on Visualization and Computer Graphics* (2023).

[29] Junxiao Shen, Boyin Yang, John J Dudley, and Per Ola Kristensson. 2022. KWickChat: A Multi-Turn Dialogue System for AAC Using Context-Aware Sentence Generation by Bag-of-Keywords. In *Proceedings of the 27th International Conference on Intelligent User Interfaces* (Helsinki, Finland) *(IUI '22)*. ACM, New York, NY, USA, 853–867. doi:10.1145/3490099.3511145

[30] T.B. Sheridan. 1989. Telerobotics. *Automatica* 25, 4 (1989), 487–507. doi:10.1016/0005-1098(89)90093-9

[31] Paul Streli, Mark Richardson, Fadi Botros, Shugao Ma, Robert Wang, and Christian Holz. 2024. TouchInsight: Uncertainty-aware Rapid Touch and Text Input for Mixed Reality from Egocentric Vision. In *Proceedings of the 37th Annual ACM Symposium on User Interface Software and Technology* (Pittsburgh, PA, USA) *(UIST '24)*. ACM, New York, NY, USA, Article 7, 16 pages. doi:10.1145/3654777.3676330

[32] Yiliu Tang, Jason Situ, Andrea Yaoyun Cui, Mengke Wu, and Yun Huang. 2025. LLM Integration in Extended Reality: A Comprehensive Review of Current Trends, Challenges, and Future Perspectives. In *Proceedings of the 2025 CHI Conference on Human Factors in Computing Systems (CHI '25)*. ACM, New York, NY, USA, Article 1054, 24 pages. doi:10.1145/3706598.3714224

[33] Keith Vertanen and Per Ola Kristensson. 2009. Parakeet: A Continuous Speech Recognition System for Mobile Touch-Screen Devices. In *Proceedings of the 14th International Conference on Intelligent User Interfaces (IUI '09)*. ACM, New York, NY, USA, 237–246. doi:10.1145/1502650.1502685

[34] 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*. ACM, New York, NY, USA, 295–298. doi:10.1145/2037373.2037418

[35] Mingrui Ray Zhang, He Wen, and Jacob O. Wobbrock. 2019. Type, Then Correct: Intelligent Text Correction Techniques for Mobile Text Entry Using Neural Networks. In *Proceedings of the 32nd Annual ACM Symposium on User Interface Software and Technology* (New Orleans, LA, USA) *(UIST '19)*. ACM, New York, NY, USA, 843–855. doi:10.1145/3332165.3347924

[36] Maozheng Zhao, Wenzhe Cui, IV Ramakrishnan, Shumin Zhai, and Xiaojun Bi. 2021. 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)*. ACM, New York, NY, USA, 162–178. doi:10.1145/3472749.3474742

[37] Maozheng Zhao, Henry Huang, Zhi Li, Rui Liu, Wenzhe Cui, Kajal Toshniwal, Ananya Goel, Andrew Wang, Xia Zhao, Sina Rashidian, Furqan Baig, Khiem Phi,

Shumin Zhai, IV Ramakrishnan, Fusheng Wang, and Xiaojun Bi. 2022. EyeSayCorrect: Eye Gaze and Voice Based Hands-free Text Correction for Mobile Devices. In *27th International Conference on Intelligent User Interfaces* (Helsinki, Finland) *(IUI '22)*. ACM, New York, NY, USA, 470–482. doi:10.1145/3490099.3511103

[38] Maozheng Zhao, Michael Xuelin Huang, Nathan G Huang, Shanqing Cai, Henry Huang, Michael G Huang, Shumin Zhai, IV Ramakrishnan, and Xiaojun Bi. 2025. Tap&Say: Touch Location-Informed Large Language Model for Multimodal Text Correction on Smartphones. In *Proceedings of the 2025 CHI Conference on Human Factors in Computing Systems (CHI '25)*. ACM, New York, NY, USA, Article 649, 17 pages. doi:10.1145/3706598.3713376

[39] Tim Zindulka, Sven Goller, Florian Lehmann, and Daniel Buschek. 2025. Content-Driven Local Response: Supporting Sentence-Level and Message-Level Mobile Email Replies With and Without AI. In *Proceedings of the 2025 CHI Conference on Human Factors in Computing Systems (CHI '25)*. ACM, New York, NY, USA, 23 pages. doi:10.1145/3706598.3713890

[40] Tim Zindulka, Jannek Maximilian Sekowski, Florian Lehmann, and Daniel Buschek. 2025. Exploring Mobile Touch Interaction with Large Language Models. In *Proceedings of the 2025 CHI Conference on Human Factors in Computing Systems (CHI '25)*. ACM, New York, NY, USA, 21 pages. doi:10.1145/3706598.3713554

A Edit Interpretation Prompts

The following prompt is used to obtain the revised text based on the instructed command. The command inserted into the placeholder <command> is the direct transcription of the user's utterance. The lighter text in the prompt is only included for the Voice+Touch and Voice+Gaze interaction modes. The focus word or words are inserted into the <context> placeholder. Using Figure 6 as an example, the context is formatted as 'birthday [4]'.

> You are a helpful assistant who edits text for the user based on their command. The user's command is, '<command>.' Give the most likely intended revision of the text based on the command. If unclear, the user may be referring to the portion of text on or near the word '<context>' where the number in square brackets indicates that it is the nth word in the body of text. Don't make any other edits to the sentence.