GestureVoice: Enabling Multimodal Text Editing for Blind Users Using Gestures and Voice

Prerna Khanna
pkhanna@cs.stonybrook.edu
Stony Brook University, USA

Monalika Padma Reddy
mpadmareddy@cs.stonybrook.edu
Stony Brook University, USA

IV Ramakrishnan
ram@cs.stonybrook.edu
Stony Brook University, USA

Xiaojun Bi
xiaojun@cs.stonybrook.edu
Stony Brook University, USA

Aruna Balasubramanian
arunab@cs.stonybrook.edu
Stony Brook University, USA

ABSTRACT

Text editing on smartphones presents substantial difficulties for blind users, particularly in mobile situations where using the smartphone touch screen is challenging. While voice input allows for hands-free text creation, editing the text typically requires physical interaction with the touchscreen, negating the benefits of the hands-free input mechanism. This paper introduces *GestureVoice*, a novel multimodal approach that enables screen-free text editing for blind users. By leveraging smartwatch-based hand gestures for navigation and voice commands for correction, *GestureVoice* allows users to edit text without any contact with their smartphones. *GestureVoice* replaces cumbersome screen-based interaction for choosing the navigation granularity with an intuitive mid-air hand gesture. It also introduces an adaptive crown cursor (rotating the physical dial of the watch) to smoothly navigate to the edit location. A preliminary study highlighted the significant time spent by blind users correcting text errors using traditional methods. In contrast, our evaluation with 8 blind users demonstrates that *GestureVoice* achieves a 53.80% reduction in text editing time, offering a more efficient, intuitive, and screen-free solution for blind users.

CCS CONCEPTS

• **Human-centered computing** → **Accessibility design and evaluation methods**; **Accessibility systems and tools**; **User centered design**.

KEYWORDS

Text editing, Accessibility, Blind users, Wearables, Gestures, Voice

ACM Reference Format:
Prerna Khanna, Monalika Padma Reddy, IV Ramakrishnan, Xiaojun Bi, and Aruna Balasubramanian. 2025. GestureVoice: Enabling Multimodal Text Editing for Blind Users Using Gestures and Voice. In *The 27th International ACM SIGACCESS Conference on Computers and Accessibility (ASSETS '25), October 26–29, 2025, Denver, CO, USA.* ACM, New York, NY, USA, 15 pages. https://doi.org/10.1145/3663547.3746388

ASSETS '25, October 26–29, 2025, Denver, CO, USA

ACM ISBN 979-8-4007-0676-9/2025/10
https://doi.org/10.1145/3663547.3746388

1 INTRODUCTION

Text editing on smartphones is extremely challenging for blind users, often accounting for more than 50% of the difficulties they encounter while using mobile devices [26, 43]. Although screen readers (e.g., TalkBack for Android and VoiceOver for iOS) are valuable accessibility tools that are often used for smartphone interactions, they are limited to reading text aloud. Editing text, which involves navigating to and modifying the text, is laborious and slow, especially because of the lack of visual cues [5, 9].

The difficulties in text editing are further exacerbated in the mobile context. A blind person often needs to use both hands to edit text since it requires touchscreen interactions. This is extremely challenging to perform when walking, when traveling in public transport, or while holding a dog or a cane with one hand. Of course, many blind people use voice for text input—voice input is hands-free and does not need any contact with the phone. However, voice input introduces numerous errors, and correcting these errors still requires touchscreen interactions. While voice-only text correction is possible, it presents significant challenges on mobile devices, particularly for precise cursor positioning and efficient error correction [19, 24, 38]. Simply re-dictating sections is also inefficient and can introduce new mistakes.

Instead, we introduce *GestureVoice*, a multimodal assistive technology that enables a *smartphone screen-free* text editing experience by combining hand gestures with voice commands (see Figure 1). A blind person using *GestureVoice* can edit a text message on-the-go even without removing their phones from their pockets and without any contact with their smartphone (hence the term *screen-free*). The enabling technology is a *smartwatch-based* gesture recognition algorithm that can accurately detect hand gestures using sensors captured from a smartwatch. Smartwatches are becoming increasingly popular [14, 28, 29] and many blind people own smartwatches, which means the user does not have to carry any additional custom device for gesture recognition. Khanna et al. [28] found that over 70% of their user study participants owned a wearable device. Other studies [1, 25, 54] have corroborated this trend, reporting that blind users favor wearable devices for everyday interactions. In our own user study with 8 blind participants (§3) we found that 6/8 users own a smartwatch and regularly use it for quick interactions.

As a first step, we conducted a preliminary user study with eight blind people to understand how participants input and edit text. Seven out of the eight participants primarily used voice for text input, with 6 participants also supplementing voice with keyboard-based text entry. However, all participants used touchscreen-based

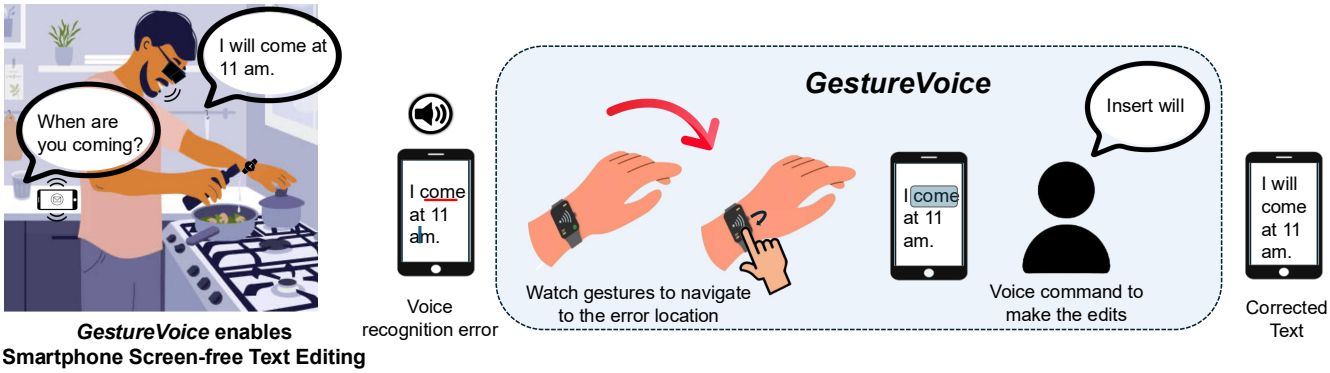

Figure 1: *GestureVoice* enables screen-free text editing for blind users.

text editing and found text editing and correction to be extremely challenging. Participants required an average of *4.5 minutes* to correct textual errors in a single sentence.

The key challenge in text editing for blind users is in navigating the cursor to the error location, especially in the absence of visual cues. These steps involved (i) choosing the right navigation granularity and (ii) positioning the cursor at the error location, which were both challenging. Many blind participants, particularly those using Apple devices, rely on the rotor [3] to choose the granularity. The rotor is a context-dependent wheel of commands that helps choose the navigation granularity— sentence-by-sentence, word-by-word, or character-by-character. While the rotor functionality is useful for text editing, interacting with the rotor requires a non-standard two-finger twisting gesture on the screen (Figure 2), which is cumbersome. Participants found rotor interaction to be an average of 3.38 difficulty on the Single Ease Question (SEQ) rating scale of 1 to 7, where 1 is difficult and 7 is easy (see § 3.2). Our study shows that navigation alone accounts for 3.28 minutes of the total 4.5 minutes editing time for a single sentence, making rotor a major bottleneck in the text correction process for blind users.

GestureVoice introduces two gesture-based techniques to efficiently navigate the cursor at the error location. First, *GestureVoice* decouples the rotor functionality from its interaction modality. *GestureVoice* uses a simple, intuitive mid-air hand gestures (e.g., circle gesture) performed with the watch-wearing hand to allow the user to easily switch navigation granularity (character, word, sentence), replacing the awkward two-finger twist gesture used by rotors. Our evaluation shows we can detect this mid-air circular gesture with a high accuracy of 97.5%, enabling robust contactless interaction.

Second, *GestureVoice* introduces an *adaptive cursor* navigation so that the user can modulate the movement towards the error location, instead of moving to the error location step-by-step. Users rotate the crown of the watch (physical dial on the side of a watch) with desired speed that determines how quickly they move through the content. A fast spin jumps quickly across multiple steps, while a slow turn allows for precise, fine-grained control—ideal when the error is nearby. This dynamic interaction makes navigating to the error location faster and smoother.

GestureVoice combines the gesture-based navigation with voice commands for text editing. The user can use simple voice commands issued via the watch microphone (e.g., "delete," "insert [text]," "replace [text]," "spell," "bold") to execute the desired edit, leveraging the speed of voice for the action itself. Figure 1 illustrates the approach of screen-free text editing for blind users.

We present the design, implementation, and rigorous evaluation of the *GestureVoice* system. Our evaluation shows that *GestureVoice* screen-free text editing approach reduced the task completion time for blind users by 53.80% when compared to the screen reader-based editing. For context, editing a document with 10 errors would require approximately 17 minutes with traditional screen readers but only about 6.5 minutes when edited with *GestureVoice*. Subjective evaluation shows that *GestureVoice* significantly enhanced editing efficiency while reducing cognitive load and improving intuitiveness among the users.

2 RELATED WORK

In this section, we review current methods designed for text editing for blind users.

2.1 Text editing for blind users

Blind users rely on screen readers such as *VoiceOver* [2] on iOS and *TalkBack* [21] on Android to interact with their smartphones [8]. Blind people use these screen readers not only to ingest information, but also for text editing. Both iOS and Android integrate an additional *rotor* feature in the screen readers for text editing that allows the user to choose their navigation granularity. For example, on iOS, the "two-finger rotational gesture" activates the rotor which is a context-dependent command wheel [3]. As the wheel rotates, the user can choose the navigation granularity as the character, word, or sentence level. Once the granularity is chosen, the user can navigate to the chosen word for editing. However, prior research shows that blind users often struggle with the two-finger rotor gesture on VoiceOver, with some opting to rewrite text entirely instead of using the rotor for text editing [5, 9]. Our own study (§3) confirms this finding.

As an alternate to screen readers, *BrailleSketch* [33] enables blind users to input text by sketching gestures that trace the dots corresponding to each Braille letter from any screen position, with word-level feedback and auto-correction. *Hybrid-Brailler* [48] combines physical chorded input for Braille entry on the back of the device with distinct touchscreen gestures dedicated to text editing tasks. Other Braille-based touchscreen input methods for blind

S.No.	Gender	Age	Text composition	Text correction	Frequency of texting on-the-go	Does voice recognition work for you?	Challenges in text correction
1	F	51	voice, keyboard	backspace and re-dictate/ re-type	not a lot	sometimes	backspace a lot, difficult to know the cursor location, voice doesn't work well in public
2	F	50	voice, keyboard	backspace and re-dictate	all the time	sometimes	backspace a lot, voice doesn't work well sometimes
3	M	58	voice	backspace and re-dictate	all the time	most of the time	backspace a lot, voice doesn't work well if background noise is present
4	F	37	voice, keyboard	rotor and re-type	all the time	sometimes	rotor action is very hard, unable to rotate the rotor sometimes, difficult to know the cursor location
5	M	44	voice, keyboard	rotor and re-type	all the time	all the time	difficult to know the cursor location
6	M	59	keyboard	backspace and re-dictate, sometimes rotor	sometimes	all the time	backspace a lot, voice is not consistent, rotor action is very hard, difficult to know the cursor location
7	F	40	voice, keyboard	rotor and re-type	sometimes	sometimes	-
8	M	60	voice, keyboard	rotor and re-type	all the time	sometimes	voice doesn't work well if background noise is present, backspacing a lot

Table 1: Summary of participants' text composition and correction strategies, including frequency of mobile text usage, how well voice recognition worked in practice, and the challenges encountered during text correction.

users include *BrailleTouch* [47], which uses a 3x2 key layout with flick gestures, and *Perkinput* [6], which leverages input finger detection to map touch patterns to 6-bit Braille with audio feedback. However, fewer than 40% of blind users use Braille, and even fewer do so on smartphones. Most prefer integrated mobile screen readers for their ease of use and real-time auditory feedback [37].

2.2 Voice-based interaction

Voice input is an efficient and accessible method for text entry particularly for blind users [5, 17, 45, 52]. However, text editing using voice remains challenging.

In the case of sighted users, voice-based text correction has been explored across a range of contexts, including document editing [19, 30, 40, 46]. More recently, it has expanded into mobile scenarios such as smartphone-based correction [13, 17] and hands-free composition using smart glasses [18], with the aim of supporting eyes-free interaction and reliance on touch-based input. Beyond direct editing commands, significant work has also focused on improving the underlying speech recognition for more natural and robust voice-only interactions. This includes addressing recognizer acceptance through robust speech repair mechanisms [36] and enabling automatic selection and correction of recognition errors by re-speaking the intended text [49, 50]. However, these systems often impose high cognitive demands, require precise and inflexible voice commands, and frequently have errors when interpreting ambiguous input [19, 24, 38]. These factors hinder their practicality in real-world use.

Khan et al. [27] developed a speech-based text editing system especially for blind users, using Google Speech API for voice-command-driven text entry and editing, with key phrases (e.g. activate word replace) enabling editing modes and providing real-time auditory feedback. However, editing remains challenging due to the linear and temporal nature of audio, which makes it hard to navigate and correct errors [5, 22].

2.3 Gesture-based interaction

Blind users find touchscreen interactions challenging due to issues such as the difficulty of one-handed interactions, the overload of gesture functions, and the susceptibility to shoulder-surfing attacks [5, 32, 53]. Recent studies have investigated alternative interaction modalities to replace traditional touchscreen gestures for blind users [11, 12, 14, 34, 35, 41]. Hand gesture interactions via smartwatches present a promising alternative by eliminating the need for specialized sensors. AccessWear [28] shows that blind users prefer smartwatch gestures as an alternative interaction method. For blind users, hand gestures offer a secure and convenient way to interact with their phones without needing to take them out. This not only allows for one-handed use but also significantly reduces the risk of shoulder surfing attacks. However, current gesture-based interactions do not extend to text editing.

2.4 Multi-modal text correction on mobile devices for sighted users

Finally, there has been considerable studies on text correction for sighted users on mobile devices. Modern smartphones employ autocorrection to automatically fix the word currently being typed but this feature is often limited in its ability to modify text that has already been entered [7, 20, 51].

Several works, designed for sighted users, edit previously entered text using prediction mechanisms [4, 10, 56]. In parallel, gesture-based systems have focused on improving text editing using different gestures [15, 16, 31, 42, 55]. Multimodal systems such as *VT* [58] and *LLM-VT* [57] combine touch gestures and voice commands for error-tolerant text editing and correction, leveraging language models to improve performance by handling complete phrases and tolerating imprecise inputs. Similarly, *EyeSayCorrect* [59] integrates eye gaze for word selection with voice input for corrections, inferring the user's correction intent through voice commands and contextual text analysis. However, these methods are designed and

tested only for sighted users; blind users face additional challenges in text editing because of the lack of visual cues.

3 TEXT EDITING METHODS USED BY BLIND USERS AND OPPORTUNITIES FOR SCREEN-FREE TEXT EDITING

We conducted an IRB-approved preliminary study to gather insights into how blind participants enter and edit text. Our study involved eight blind participants (4 female, 4 male) aged 37-60 years (see Table 1). Five of the participants were congenitally blind, meaning they had never had visual input, while three participants had lost their sight later in life. All participants were highly experienced with Apple's iPhone and proficient in using the built-in VoiceOver screen reader. For the study, we developed a custom iOS app that ran on an iPhone 15 Pro, specifically designed to be fully compatible with VoiceOver. This app served as the primary interface for text composition and correction tasks. All data was stored for later offline analysis on the MacBook M1.

3.1 Design

To observe participants' natural text composition and editing behavior, we conduct two tasks as described below.

Task 1: Text Composition and Editing. The first task allowed participants to compose and edit text as they would in a real-world setting. Participants were asked to transcribe five sentences presented to them via the screen reader. These sentences could be tapped on, allowing participants to hear the sentence repeatedly, ensuring that they understood the content. Participants were given the freedom to use either voice dictation or the on-screen keyboard for text composition. Additionally, they were instructed to correct any mistakes using their preferred method, whether that was backspacing or using the rotor functionality to navigate through the text.

Task 2: Correcting Induced Errors. In this second task, we introduced five sentences that contained intentional errors, such as omissions, substitutions, or insertions (see Table 5 for types of erroneous sentences). Participants were informed of these errors in advance, and the correct version of each sentence was made available through the screen reader. By tapping on the reference sentence, participants could hear the correct sentence spoken aloud. Participants were asked to correct these errors using any method they found most comfortable, whether that involved backspacing through the text and re-typing or re-voicing it, or using the rotor to navigate to specific parts of the sentence. The errors were strategically placed to simulate common typing mistakes that might occur during real-world use.

3.2 Key Findings

The key findings of our study are the following:

1. Voice preferred for composition, manual methods used for text editing.

We found a strong preference for voice input for initial text composition task. In general, all participants expressed a general preference for voice for text input, particularly valuing its hands-free convenience when outdoors or on the move (see Table 1).

Figure 2: Interaction of blind users with the screen reader rotor. (a) shows a user attempting to activate the rotor using a two-finger rotation gesture with both hands. (b) illustrates the user rotating both the phone and their hands simultaneously to activate the rotor. The mean SEQ score for rotor gesture difficulty was 3.38 across the users (1-difficult, 7-easy).

When corrections were needed, participants primarily resorted to manual, touch-based methods: using the screen reader rotor (4/8 participants) or repeatedly pressing the backspace key (4/8 participants).

When asked about their preference for text editing, half of the participants (4/8) stated they do not use voice for corrections, while the other half reported that their primary voice-based "correction" strategy was to delete the entire message and re-dictate it. This workaround avoids the difficulty of specifying the error location but is inefficient and frustrating, often requiring multiple attempts, especially if dictation accuracy suffers due to background noise (a problem reported by 5/8 participants in public settings, Table 1). Table 2 shows some specific user quotes that illustrate the problem with editing.

2. Screen-based text editing is hard even with rotors.

Even when users resorted to manual correction methods, they continued to face significant challenges—particularly with the rotor. Participants in our study often found the two-finger rotation gesture used to activate the rotor difficult to perform consistently. As a result, they frequently triggered accidental actions or failed to correct errors as intended. Performing the two-finger rotation gesture also required fine motor control, adding another layer of difficulty. Some even rotated the entire device in an effort to mimic the gesture. This underscores how unintuitive and physically demanding the rotor interaction can be (see Figure 2). The two-finger rotor interaction was rated: 2, 1, 7, 5, 4, 1, 2, 5 by the eight participants (on a SEQ scale where 1 = very difficult, 7 = very easy). Mean score 3.38±2.19 and many found rotor interaction difficult. User P3 who gave the rating 7 may have misinterpreted the scale (i.e., thought 7 is difficult) because their responses indicate that they found the rotor interaction extremely challenging. Four participants expressed strong dissatisfaction, preferring the tedious backspace method over the frustrating rotor interaction. Apple's fixed two-finger touch-and-rotate gesture for the rotor can't be customized or remapped, making changes difficult. Even if this rotor touch gesture could be altered, it wouldn't enable screen-free text editing, which is a core benefit of *GestureVoice*. This allows a

"I use the rotor, don't like it though. If you are not careful while doing the correction, you can edit the wrong thing. Sometimes while I am speaking, it does not understand me."
"Swiping up/down using a rotor does not tell me exactly where I am."
"The actual rotor is very cumbersome to rotate."
"I can not understand where the cursor is placed, start of the word, end of the word, before or after the space. This is very confusing."
"I like using voice to edit text, but sometimes I have to redo the entire message because it picks up something different from what I actually said. That is annoying."
"Dictation doesn't always work reliably. Voice works better for short texts, but with longer messages, it often misses nuances. It's frustrating to make corrections in long paragraphs, and I usually make fewer mistakes when typing than using voice, but typing takes more time."
"I usually use voice to send short texts — like saying I'll arrive in 10 minutes, but background noise often causes errors. I end up having to retype or erase the whole thing and say it again. It usually takes 3–4 tries to get it right. It'd be easier if making small corrections to the text were simpler."

Table 2: User responses to the question 'Is text editing a difficult task?' highlighting various challenges encountered during text editing.

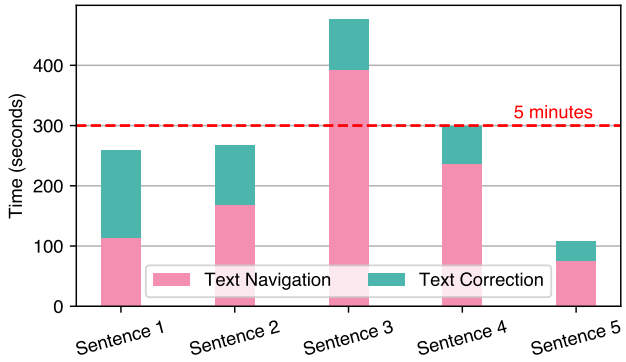

Figure 3: Mean time taken by blind users to correct induced errors in sentences, comprising the time spent navigating to the error (text navigation) and correcting the error (text correction). Pink bars represent time spent navigating to errors, green bars represent time spent making corrections once errors are located, and the red dashed line represents the 5-minute threshold, beyond which the task was considered incomplete.

S.No.	Do you have a smartwatch?	What type of watch?	Frequency of use
1	no	-	-
2	yes	Pixel	everyday
3	no	-	-
4	yes	Apple	everyday
5	yes	Apple	sometimes
6	yes	Apple	sometimes
7	yes	Apple	rarely
8	yes	Apple	sometimes

Table 3: Summary of participant responses regarding smartwatch usage, watch type and the frequency of use.

blind person to edit text on their phones without needing to take them out.

Our analysis also showed that navigating to the error location (see Figure 3) often took considerably longer (average 3.28 minutes) than performing the correction itself (average 1.2 minutes) to correct a single sentence. This issue became even more pronounced when navigating through longer pieces of text, where precise cursor placement is essential. In such cases, users found themselves having to move back and forth repeatedly to figure out where the cursor was located, increasing both the time and effort required to complete the correction.

3. Wearables offer potential for screen-free interaction combined with voice. A majority of participants (6 out of 8) reported owning a smartwatch, with most using them frequently (Table 3). They use their watches for quick interactions such checking notifications, getting directions, or sending short dictated messages, particularly in public or while mobile. Several noted that using the watch felt safer and less cumbersome than taking out their phone in potentially risky environments. One participant mentioned preferring the watch after dropping their phone, stating, "I just use the watch now, pulling out the phone again feels too risky."

Despite the convenience for quick tasks and voice dictation, participants universally agreed that performing text corrections on the smartwatch was currently impractical or impossible. This limitation, combined with occasional voice recognition errors, often led to users sending messages with errors rather than attempting to fix them on the watch or retrieving their phone. As one user put it, "If the watch gets it wrong, I just send it anyway — there's no way I'm fixing it there..." This highlights a critical gap: wearables are convenient for mobile input but lack effective editing capabilities. However, this widespread adoption and preference for mobile use suggest wearables are a promising platform for new interaction techniques.

Thus, voice input and wearable interaction compliment each other and, when combined, may overcome the limitations of each modality alone. This motivates the exploration of novel wearable-based interaction models for text editing, such as the one proposed in this work.

4 *GESTUREVOICE*: SCREEN-FREE TEXT EDITING FOR BLIND USERS

In this section, we describe *GestureVoice*, a multimodal assistive technology that enables a *screen-free* text editing experience by combining hand gestures with voice commands. With *GestureVoice*, one can use hand gestures to navigate to the error location and use short voice commands to execute the editing action, without touching their smartphone.

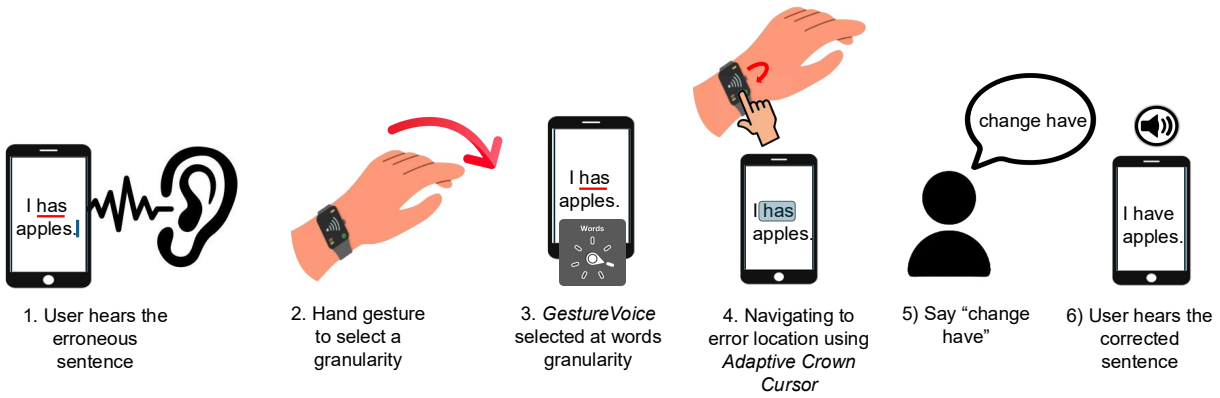

Figure 4: Illustration of the interaction with the *GestureVoice* system for blind users. (1) Step 0: an example where the user attempts to change "has" to "have" in the sentence "I has apples." (2) Step 1: use the mid-air circle gesture to select a granularity (3) Step 2: Set to "word" granularity. (4) Step 3: Use the watch crown rotation to move to the error location quickly instead of word-by-word and select the entire word "has." (5) Step 4: use the voice command to replace has with have, and (6) Step 5: User hears the corrected sentence.

To achieve this, we implement three core components: (1) Gesture-based rotor interaction that enables hands-free granularity selection using mid-air gestures, (2) Adaptive crown navigation that allows the user to move to the error location quickly, and (3) Integration with voice commands for editing. Figure 4 shows the flow of this screen-free text editing experience.

4.1 Mid-Air Gesture for Granularity Selection

Our preliminary user study shows that rotors provide an intuitive way for navigation by allowing the user to choose the right granularity. However, touch interaction associated with the rotor is non-intuitive and difficult. Instead, *GestureVoice* decouples the rotor's underlying functionality from the interaction modality by replacing the touch interaction with gesture based interaction. In *GestureVoice*, we choose the "mid-air circle" gesture for interaction. Analogous to the touch rotor, this circle gesture cycles through the available granularities (character → word → line → sentence → character) with each rotation. The system confirms the selection via VoiceOver audio feedback.

We chose the circle gesture as it is intuitive and mimics the natural use of the touch screen rotor, and we design a gesture detection algorithm to accurately detect this gesture. However, users can have varied gesture preferences. We envision that one can replace the circle gesture with a different hand gesture if needed; however, the system will need to design the corresponding gesture detection algorithm.

Gesture Detection Process: The goal is to recognize the mid-air circle gesture accurately and robustly using the Inertial Measurement Unit (IMU) sensor data captured from the smartwatch. In this work, we only use data from the gyroscope, which captures the rotational motion of the gesture.

We borrow ideas from related work [28] and use a time-series template matching algorithm focused on identifying the "gesture nucleus". The gesture nucleus represents the core of the gesture after the user gets into position to perform the gesture and before the user gets out of the position. Nucleus is the core pattern of movement that occurs during the main execution phase of a gesture, excluding the preparation and retraction phases. Previous work [28] has shown that this nucleus is user invariant and has a consistent pattern. Therefore, once we extract the nucleus, we only need to match it to a pre-determined circle template to recognize the gesture.

The process begins by calculating the magnitude of the 3-axis gyroscope signal:

$$magnitude = \sqrt{gx^2 + gy^2 + gz^2} \tag{1}$$

where gx, gy, gz are the instantaneous gyroscope readings.

Next, to identify the nucleus in this magnitude signal, we use energy-based thresholding techniques [28]. We use an overlapping sliding window technique to iterate over the magnitude signal. For each window position, we calculate the energy change compared to the previous window. Windows that show maximum energy change are marked, and they indicate the boundaries where the gesture nucleus begins and ends. We discard any windows marked that are too close. The intuition here is that the speed of the gesture changes after the user gets into position to perform the gesture and gets out of position after the gesture; by identifying these change points, we can identify the nucleus.

After isolating the nucleus segment, we compare it against a predefined template representing the canonical mid-air circle gesture. For this comparison, we implement the Fast Dynamic Time Warping (DTW) algorithm [44]. DTW algorithm excels at handling temporal variations, allowing us to recognize gestures performed at different speeds or with slight timing variations across the users. The DTW algorithm computes a distance score reflecting the similarity between the detected *nucleus* and the template, accommodating variations in speed and timing. If this distance is below an empirically set threshold ($\theta_{DTW} = 10.0$), a valid gesture is confirmed. A significant advantage of our detection model is that it eliminates the need for individual user calibration or training. The template matching focuses specifically on the gesture nucleus sequence, which remains consistent across different users [28]. We implement the

nucleus detection algorithm on the smartphone that pairs with the smartwatch. The user initiates the process by tapping on the watch. On recognizing a tap, the watch begins streaming gyroscope data to the phone via Bluetooth. This ensures that non-intentional hand movements do not trigger the gesture detection algorithm.

4.2 Adaptive Crown-Based Cursor Navigation

The current navigation method requires users to move step-by-step—whether that's by word, sentence, or character—depending on the selected granularity. While precise, this approach can be frustratingly slow, as we observe in our preliminary study.

To address this, *GestureVoice* leverages the watch crown to offer a more fluid and intuitive navigation experience. Instead of rigid steps, the speed at which the user rotates the crown determines how quickly they move through the content. A fast spin jumps quickly across multiple steps, perfect for covering long distances. A slow, deliberate turn allows for precise, fine-grained control—ideal when the error is nearby.

We chose the watch crown because it's a physical dial that provides natural feedback as a user turns it. Users can feel distinct "clicks" as they rotate the crown, which match the movement through text at their chosen level (character, word, or sentence). This physical feedback helps users understand exactly how far they're moving. We combine this physical feedback with voice output from the screen reader. As the user turns the crown and the cursor moves, the screen reader reads out each selection in real-time. For example, when moving word by word, each click of the crown advances to the next word, which is immediately spoken aloud. This combination of physical feedback and sound helps the user to know their position in the text.

Other gesture options could work alongside or instead of the crown. Gestures like rotating on the edge of the watch dial or multi-finger circular motion gestures could be potentially used. In future versions, we plan to let users choose their preferred gesture method based on their own needs and abilities.

Mapping Crown Rotation to Cursor Movements: The main challenge now is to map the rotation of the watch crown to the speed of navigation for the cursor. The system converts the continuous turning of the watch crown into a sequence of discrete steps. Every crown rotation produces a raw angle value ($\Delta\theta$), capturing both how far and in which direction (forward/ backward) the crown is turned. We smoothen the raw values to prevent over-sensitive reactions. Smoothening is achieved by dampening this raw input value using an empirically determined threshold $\alpha = 0.5$:

$$\Delta\theta_{smoothed} = \alpha \cdot \Delta\theta \tag{2}$$

The smoothed value, $\Delta\theta_{smoothed}$, determines the intended cursor displacement direction and magnitude. The smoothed rotation is then converted into discrete cursor steps. This is where adaptive speed comes in:

$$\Delta cursor\_steps = \text{sign}(\Delta\theta_{smoothed}) \cdot \min(|\Delta\theta_{smoothed}|, MaxStepsPerUpdate) \tag{3}$$

In this equation, direction is determined by the sign of the rotation (forward or backward). Step Size depends on how fast the crown was turned. A slow, small turn results in a small smoothed value, typically producing a single step. A faster rotation yields a larger $\Delta\theta_{\text{smoothed}}$, requiring more steps. Capping with $MaxStepsPerUpdate$ (e.g., 1 or 2) ensures stability. Even rapid crown spins won't cause the cursor to jump unpredictably far. Instead, fast rotations lead to several small updates in succession, preserving user control.

The computed number of steps is sent from the smartwatch to the smartphone, which then updates the cursor position using accessibility APIs. As the cursor lands on each word, the system highlights it and announces the word, mitigating cursor ambiguity.

4.3 Voice Command Interface for Editing

Once the user navigates to the error location, *GestureVoice* uses a simple voice command to specify the text editing action. This process begins with capturing speech via the watch microphone. The audio is transcribed to text on the smartphone (using *SFSpeechRecognizer* on iOS). The system parses this text to identify supported commands and associated parameters from a predefined grammar, which includes:

- **Basic editing:** "delete", "type [text]", "insert [text]", "change [text]"
- **Formatting:** "bold", "italic", "underline"
- **Punctuation:** "period", "comma", "question mark"
- **Navigation/System:** "cursor position", "read", "unselect", "correct"

If a valid command is found, the corresponding action is executed in the text view at the current cursor position. Most commands in our system implement straightforward logic directly coded into the application. For example, "delete" removes the selected text, while "italic" transforms the selected text to italic style formatting. However, the "correct" command works differently. Instead of using fixed rules, we leverage a Large Language Model (LLM) to provide intelligent text correction. The system sends the current sentence to our integrated LLM, which analyzes the text for grammatical errors, contextual inconsistencies, and other language issues. The model considers both grammar rules and the surrounding context to generate the most appropriate correction. We use an external T5 grammar model [23]. Running the LLM inference on the phone for sentence correction takes 4.5 sec on average. Once processed, VoiceOver announces the specific changes made (such as fixing verb tense or adjusting punctuation), giving users clear feedback about how their text was modified.

4.4 Implementation Details

The *GestureVoice* prototype is built for iOS, using an Apple Watch and an iPhone. The two devices communicate via Apple's `WatchConnectivity` framework for real-time transfer of IMU data, crown events, and commands. On the Apple Watch, IMU data is buffered and analyzed for gesture detection (Section 4.1). Crown input is captured using the `digitalCrownRotation` API. Both gesture and crown events are packaged and sent to the iPhone.

The iPhone app handles incoming messages, tracks system state (e.g., current granularity, voice mode), and interacts with a `UITextView` and the VoiceOver screen reader. Cursor navigation is implemented using the `UIAccessibilityCustomRotor` API, enabling crown-driven, programmatic cursor control. Voice input (Section 4.3) is transcribed using `SFSpeechRecognizer`, then parsed to determine

ID	Gender	Age	Phone	Screen Reader Familiarity	Text Preference	
					Composition	Editing
P1	F	53	Android	Proficient	Voice	Backspace and re-type
P2	M	60	Android	Proficient	Voice	Rotor, backspace, re-voice
P3	M	45	iOS	Proficient	Keyboard (short text) and Voice (long texts)	Rotor and re-voice
P4	F	37	iOS	Proficient	Keyboard and Voice (short messages)	Rotor and re-type
P5	M	60	iOS	Proficient	Keyboard and Voice (long texts)	Rotor and re-type
P6	M	58	iOS	Proficient	Voice	Backspace and re-voice
P7	F	50	iOS	Proficient	Keyboard and Voice (long texts or outdoors)	Backspace and re-voice
P8	F	41	iOS	Expert	Keyboard and Voice (hands occupied or at home)	Rotor and re-type

Table 4: The first five columns in the table show the demographics and smartphone usage information. The last two columns show participants' text correction preferences (discussed in §5.8).

commands. Text edits—cursor movement, deletion, insertion, formatting—use standard iOS text APIs. Audio feedback is generated via: `UIAccessibility.post(notification: .announcement, argument:...)` Text corrections associated with the "correct" command are performed using async REST calls to a Hugging Face model endpoint. As mentioned earlier, we use an external T5 grammar model (Grammarly coedit-large via Hugging Face API [23]). The CoEdIT (Collaborative Editing) model is built upon FLAN-T5 and fine-tuned on diverse text revision instructions, offering grammar, spelling, and stylistic corrections. Input to the T5 model, an encoder-decoder architecture fine-tuned on instruction following, is the erroneous sentence and output is the corrected sentence. We use the prompt: *grammar: [text]*. For user feedback, the system announces the corrected sentence and correction type. T5 was selected for its lightweight architecture and low on-device inference time (~ 4.5 seconds); the correction model can be substituted with any equivalent language model.

While the current implementation targets iOS, the architectural principles and interaction modalities could be adapted to Android using corresponding APIs like *AccessibilityService* (for screen reader interaction), *WearableListenerService* (for watch communication), Android's Speech Recognizer, and standard text view manipulations. Additionally, to enable other researchers to explore this domain, we have made the iOS mobile and companion watch app implementation of *GestureVoice* available at: https://github.com/SBUNetSys/GestureVoice.

5 EVALUATION AND RESULTS

We conducted a comprehensive user study based evaluation of *GestureVoice*. The IRB-approved user study involved eight (8) blind participants. Participants were asked to correct sentences with induced errors using two methods: (1) their standard screen reader editing techniques that they typically use (called "default screen reader editing"). All eight participants in our study were familiar with iOS screen reader, with seven being proficient and one being an expert, and (2) the *GestureVoice*-based text editing we described in §4.

Our evaluation focused on three key aspects: (1) task completion rate and time taken for error correction, (2) accuracy of gesture detection, and (3) subjective user feedback. The key findings are:

- *GestureVoice* improved task completion time by 53.80% compared to VoiceOver and achieved a 100% task completion rate.
- The gesture recognition algorithm achieved a high accuracy of 97.5% for mid-air circle gestures, enabling reliable screen-free interaction.
- While proficient users benefit most significantly from our system, even expert screen reader users recognize the value of *GestureVoice* for text editing on-the-go

5.1 Participants and Apparatus

The study involved 8 blind participants (4 female, 4 male) aged 41–60 years. Six participants were highly experienced iPhone users and proficient with Apple's VoiceOver screen reader, while two participants primarily used an Android phone but were familiar with VoiceOver (see Table 4). 7 participants from the previous pilot user study were part of this study. For the experiment, we developed a custom iOS app running on an iPhone 15 Pro, fully compatible with VoiceOver, which served as the primary interface for the text correction tasks. We developed a companion app for the Apple Watch Series 10 to stream gyroscope sensor data to the iPhone when a tap was detected. When a long press was detected, the smartawatch used the microphone to recognize commands. For collecting the gesture traces, we sampled IMU data at 100 Hz and transmitted via Bluetooth to a MacBook M1 laptop, where a custom data logger application stored the data for offline analysis.

5.2 Design and Procedure

We designed a within-subject experiment to compare the two text correction methods for blind users. The independent variable were the text correction methods. Participants were tasked with correcting errors in sentences containing intentional omissions, substitutions, or insertions (see Table 5). The correct version of each sentence was made available through the screen reader, and participants could hear it by tapping the reference sentence. Sentences

Error Type	Sentence with Error	Corrected Sentence
Omission	She went to ^ store yesterday.	She went to **the** store yesterday.
	She ^ going to buy new car next month.	She **is** going to buy new car next month.
	They ^ waiting for bus at ^ corner.	They **are** waiting for bus at **the** corner.
Substitution	He **gived** me the book yesterday.	He **gave** me the book yesterday.
	We **buyed** groceries for dinner tonight.	We **bought** groceries for dinner tonight.
	The students **writed** their essays last week.	The students **wrote** their essays last week.
Insertion	My sister **she** works at the hospital downtown.	My sister works at the hospital downtown.
	The document that I wrote **it** contains important information.	The document that I wrote contains important information.
	When the professor **he** explained the concept, I understood it.	When the professor explained the concept, I understood it.
	John **he** is going to **to** visit his parents tomorrow.	John is going to visit his parents tomorrow.

Table 5: Types of errors used in the text correction tasks during the user study

for the error correction task were picked using the error distribution used in Palin et al.'s mobile typing dataset [39]. This dataset is widely used for other text editing studies [58, 60]. We used these sentences to standardize the evaluation across users and alternate techniques. The order of sentences was randomized for each participant.

Default Screen Reader Editing: We presented a 5-minute training session on VoiceOver and screen reader editing techniques for uniformity, but all 8 participants were already familiar with these functionalities. As shown in Table 4, participants used various editing methods matching their real-world preferences, including: 1. Backspace and re-typing (P1, P6): These participants deleted text using the backspace key and then re-entered the correct content. 2. Rotor navigation with re-typing (P4, P5, P8): These participants used the VoiceOver rotor to navigate to specific locations in the text and then manually typed corrections. 3. Rotor with voice re-dictation (P2, P3): These participants combined rotor navigation with voice input for corrections. 4. Backspace with voice re-dictation (P6, P7): These participants preferred deleting erroneous text with backspace and then using voice to re-dictate the correct content.

All participants were proficient with VoiceOver, with one (P8) self-identifying as an expert user.

***GestureVoice* Text Editing:** In the *GestureVoice* method, participants received a 5-minute training session on how to use the contacltess text editing platform. The participants were given a test sentence: "I *has* apples." We then demonstrated how to perform the mid-air circle gesture while wearing the smartwatch to change granularity and how to use the adaptive crown cursor with variable speed to navigate to the error location. They were presented with the list of voice commands that could be used to perform the edit action. To initiate the session, participants tapped on the smartwatch, activating the gesture detection algorithm. As they performed the mid-air circle gesture, the screen reader provided audio feedback, guiding them through various granularity levels (character, word, line). Once a granularity level was selected, participants used the watch crown to navigate through the text at that granularity level, rotating clockwise to move forward and counterclockwise to move backward. For text operations such as read, deletion, replacement, or insertion, participants could use voice commands triggered by a long press on the watch. Participants familiarized themselves with the audio and haptic feedback provided in the system to perform screen-free text editing.

For each participant, the experiment included two correction methods (default rotor and *GestureVoice*) and ten trials per method (10 erroneous sentences), resulting in a total of $8 \times 2 \times 10 = 160$ trials. We used counterbalancing to control for order effects between default screen reader editing and *GestureVoice*, alternating starting condition across participants.

Finally, to evaluate the gesture detection accuracy, during the study, we collected gesture traces using the IMU sensors on the smartwatch as participants performed various gestures and evaluated them offline. To obtain ground truth data, we also recorded videos (showing only participants' hands) during the text correction and gesture tasks.

5.3 Task Completion Rate

We report a trial as complete if the user successfully corrected the erroneous sentence to match the target sentence. Any trial that exceeded 5 minutes was marked as incomplete. Using the default screen reader editing, participants completed 158 out of 160 trials, resulting in a 98% task completion rate. With *GestureVoice* all users successfully corrected the sentences, achieving a 100% task completion rate.

The two incomplete trials with the default screen reader method show specific interaction barriers. In one case, the user struggled to navigate the rotor menu, repeatedly cycling through commands accidentally without being able to select the desired granularity. In the second case, the user selected the correct granularity but edited incorrect portions of the text due to cursor position ambiguity - a common challenge with screen readers, where users lose track of their exact position within the text.

5.4 Task Completion Time

The task completion time was measured from the moment the experimenter clicked the "record" button to the time they clicked

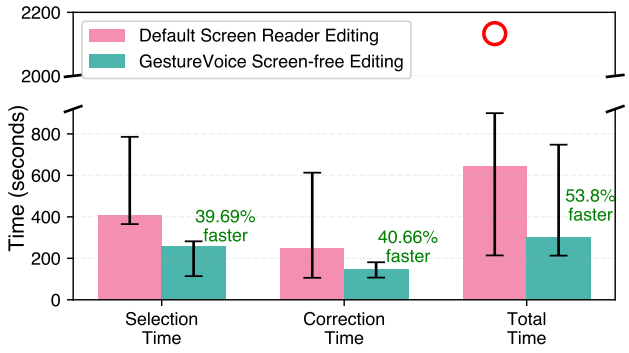

Figure 5: Comparison of task time between the Default screen reader based editing (pink) and the screen-free *GestureVoice* (green) for blind users. *GestureVoice* is 39.69% faster for moving the cursor to error location, 40.66% for making the edits.

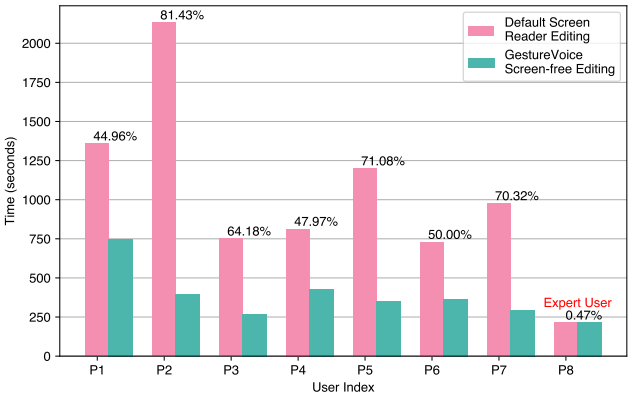

Figure 6: Time reduction for correction tasks using the *GestureVoice* (green) compared to the Default screen reader based editing (pink) across different users. The proposed *GestureVoice* is 53.80% faster than the default screen reader for text editing. The time reduction % is annotated on top of each bar.

"stop recording" after completing the correction. This time metric captures the duration users took to navigate to and correct errors. Figure 5 shows that our proposed gesture rotor method resulted in a 53.80% reduction in average correction time compared to the default rotor method. *GestureVoice* significantly accelerates the text editing process for blind users. It enables 39.69% faster cursor movement and selection, due to intuitive mid-air gestures for quick navigation granularity adjustments and the precise control offered by adaptive crown-based navigation. Furthermore, the use of efficient voice commands for edits results in a 40.66% speedup compared to touch-based screen reader interactions.

Figure 6 shows the percentage reduction in task completion times per user when using our proposed gesture rotor compared to the default rotor. The time savings were significant for 7 out of 8 users. The average time reduction was approximately 44.96%, 81.43%, 64.18%, 47.97%, 71.08%, and 50%, and 70.32% respectively for the 7 users. A paired-samples t-test confirmed that the time difference was statistically significant ($t_7 = 3.55$, $p = 0.0093$).

Participant P8, who had more experience with screen readers and accessibility tools demonstrated marginal improvement of 0.47%.

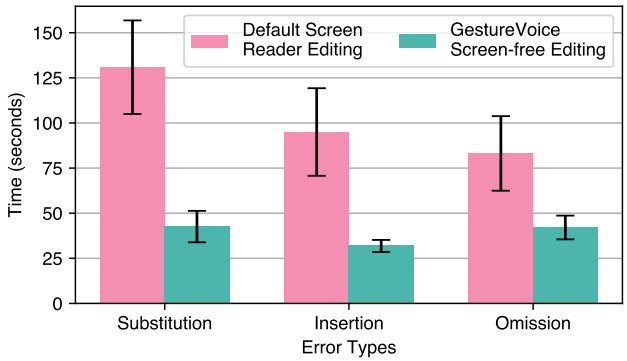

Figure 7: Mean time taken by blind users to correct different error types (Substitution, Insertion, Omission) using the default screen reader editing (pink) and the proposed *GestureVoice* screen-free editing (green). The proposed *GestureVoice* shows significant time reductions for Substitution, Insertion and Omission errors.

Due to their familiarity with the existing system and reliance on learned muscle memory for interacting with the default rotor, they completed the correction tasks in almost as quickly using the default rotor and did not see benefits in terms of time when using the *GestureVoice* for interaction. This expert user has years of experience teaching accessibility tools at the NYC Public Library. However, they appreciated the idea of *GestureVoice* and even commented how such a system could be useful for their students, as they see a lot of screen reader users struggle with the rotor interaction. They mentioned they would like to use *GestureVoice* when on-the-go or when they can not use their phone.

The 53.80% average reduction in task completion time highlights the frustrating editing process with current editing methods. For context, editing a typical document containing 10 errors like those shown in Table 5 would require approximately 17 minutes with traditional screen readers but only about 6.5 minutes with *GestureVoice*. This time-saving changes the texting workflow for blind users, shifting the editing process from a tedious to a more natural writing experience.

5.5 Error-Specific Performance Analysis

Different error types present unique challenges for blind users during text editing, each requiring specific interaction patterns and cognitive load. Understanding how *GestureVoice* impacts each error type provides insights into where the system offers the most significant benefits in real-world scenarios. We evaluated three common error types: substitutions, insertions, and omissions. As shown in Figure 7, the *GestureVoice* considerably reduced the time required for all the kinds of errors. Each error type required different editing operations, with our system showing distinct advantages for each scenario.

To correct the *substitution errors*, participants were required to precisely navigate and replace the incorrect text with the correct version ("change [text]" commands). Participants achieved the most significant improvement, reducing correction time by approximately 66% (42 seconds vs. 127 seconds average (see Table 6)). The

Error Type	Default (s)	*GestureVoice* (s)	Reduction (%)
Substitution	127	42	66%
Insertion	95	32	66%
Omission	83	42	49%

Table 6: Performance comparison by error type showing the average time (seconds) required to correct each type of error using the default screen reader versus *GestureVoice*.

combination of gestures and voice replacement commands eliminated the multiple rotor manipulations and keyboard interactions typically needed for text substitution.

Insertion errors primarily involve deletion operations. This error type showed the greatest relative efficiency gain. Users completed these corrections in nearly one-third of the time required with traditional methods (32 seconds vs. 95 seconds average (see Table 6)). Screen readers announce the location of the cursor or focused item, but users often struggle with cursor ambiguity, not knowing if the cursor is at the start or end of a word, which can lead to deleting the wrong content. Our system's straightforward "delete" voice command, paired with precise crown-based navigation, provided clear cursor positioning feedback, eliminating this ambiguity.

Even for *omission errors*, which require adding missing text through "insert [text]" commands, participants maintained significant performance advantages with our system (42 seconds vs. 83 seconds average (see Table 6)). In traditional screen readers, users navigate through content using touch gestures, but often face challenges determining exact cursor position—whether it's at the end of a word or after a space. *GestureVoice*'s algorithm figures out if users are trying to insert at the character or word level, and adds spaces in the right places. This makes it much easier for users to add missing text without having to worry about where the cursor is exactly placed or if they need to add spaces themselves.

These findings demonstrate that *GestureVoice*'s combination of gesture-based navigation and voice commands directly addresses the most challenging aspects of text editing for blind users—maintaining awareness about the location of the cursor and reducing the steps required for the editing operations.

5.6 Accuracy of Gesture Detection

For the alternate gesture interaction to be successful, gesture detection needs to be highly accurate. Any inconsistency or inaccuracy in gesture detection could lead to frustration and hinder the usability of the system. To evaluate the reliability of gesture detection, we analyzed the offline gesture traces collected during the user study.

Figure 8 shows the detection accuracy across the users with an average accuracy of 97.5%. Participants were asked to rate the ease of performing each gesture using the SEQ scale (where 7 represents "very easy" and 1 represents "very difficult"). The mid-air circle gesture received a 5.6 mean score. 5/8 users liked the mid-air circle gesture as it is hands-free and mimics the use of rotor. Partcipant P4 mentioned that she would prefer a *flick wrist* gesture over the circle gesture. The latency for gesture recognition on smartphone is 11 msec on average and the it takes 46 msec to stream gyroscope data from the watch to the phone. This shows that *GestureVoice* can work in real time with participants not perceiving any delays when performing text editing.

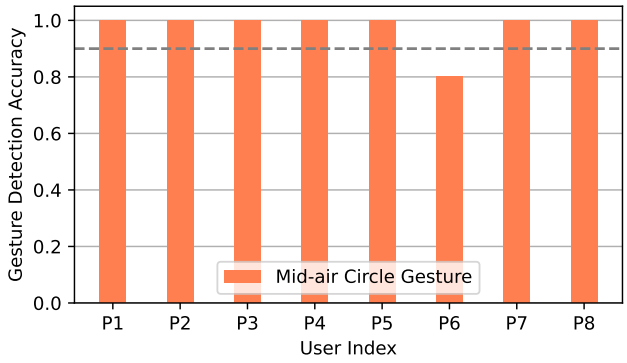

Figure 8: Gesture detection accuracy across blind users for the Mid-air Circle gesture (orange). The dashed line indicates the 90% accuracy threshold. Gesture detection accuracy varies across users, with both gestures being recognized with high accuracy for most participants.

5.7 Detailed Performance Analysis

To provide a more comprehensive understanding of user interactions and system performance, we analyzed additional metrics from the user study, including error rates, correction strategies, and command usage patterns.

5.7.1 Error Rates and Correction Attempts. Although *GestureVoice* achieved complete task success across all trials, users did experience occasional recognition errors during interaction. Speech recognition proved to be the primary source of errors, occurring in 22 of 80 trials (27.5%) when voice commands were misunderstood by the system. Most of these issues were quickly resolved—20 trials needed just 2 retry attempts, while only 2 trials required 3 attempts before the system correctly recognized the command. Gesture recognition was notably more reliable, with errors appearing in just 2 trials (2.5%), both of which were corrected after 2 retry attempts.

5.7.2 Navigation Granularity Preferences. When examining how participants navigated through text, a clear pattern emerged favoring word-level granularity. Participants chose word-level navigation in 98.75% of trials, suggesting this level offers the right balance between control and efficiency for correction tasks. Character-level and line-level options were rarely selected, indicating that word-by-word movement matches how users naturally think about editing text.

5.7.3 Comparison of Correction Strategies. The approach users took to fix errors varied significantly between systems. With standard VoiceOver, half of the participants (4 out of 8) resorted to deleting entire sentences and retyping them from scratch, whether through manual typing or dictation. While this method guarantees accuracy, it demands considerable time and mental effort. With *GestureVoice*, this approach disappeared entirely. Every participant instead made precise, targeted corrections using voice commands combined with gestures. This shift in behavior shows that *GestureVoice* opens up efficient editing techniques that were previously difficult or impractical with conventional screen readers.

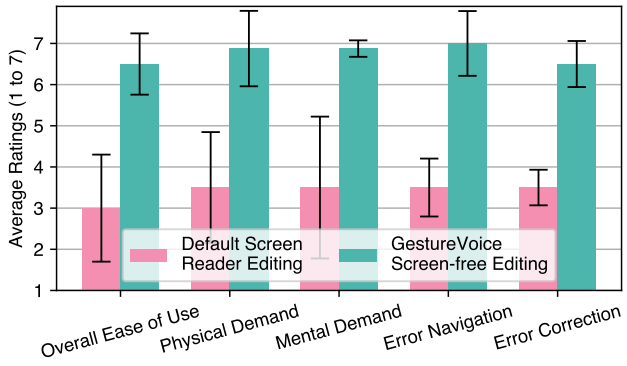

Figure 9: Mean SEQ ratings for text correction using the Default Screen Reader editing (pink) and *GestureVoice* Screen-free Editing (green), rated from 1 (very difficult) to 7 (very easy). The proposed *GestureVoice* Screen-free Editing shows higher ratings compared to the Default Screen Reader Editing across all categories.

5.7.4 Command Usage Patterns. Analysis of voice command usage in *GestureVoice* revealed distinct patterns in how participants approached text corrections. Once the appropriate granularity level was selected, the most frequently used commands were:

- *Delete* commands: 41.3% of all corrections
- *Insert* commands: 25.8% of all corrections
- *Replace* commands: 24.7% of all corrections
- LLM-based auto correction: 2.5% of all corrections

The predominance of delete commands aligns with the error types used in our study, where insertion errors (requiring deletion) were common. The balanced usage of insert and replace commands demonstrates that participants effectively utilized *GestureVoice*'s diverse correction capabilities rather than relying on a single editing strategy.

5.7.5 LLM-based Auto Correction Usage. The LLM-based auto correction feature was used rarely despite being covered in training: only 2 participants tried it across 2.5% of trials. This feature, activated by selecting the appropriate granularity and saying "correct," was designed to automatically handle common grammatical mistakes. The low usage suggests that users need more time to become comfortable with the feature during the study.

5.8 Subjective Feedback

We gathered subjective feedback from participants on several aspects of the text editing tasks, including overall ease of use, physical and mental demands, error navigation, and error correction (see Figure 9). The *GestureVoice* system received higher ratings across all categories compared to the default VoiceOver based text editing. Participants found the *GestureVoice* to be less physically demanding and easier to navigate, which corresponds with the faster task completion times observed. On a 7-point SEQ scale, where 7 indicates "very easy" and 1 indicates "very difficult," the *GestureVoice* consistently achieved higher average scores.

Furthermore, all the users expressed a preference for the *GestureVoice* over the default screen reader editing. Participant P8 (an expert user), however, noted that she preferred the default rotor due to her familiarity with it, but acknowledged, "For novice users, or those who struggle with rotating the rotor, the gesture-based system is very useful. When I first started using screen readers, this would have been extremely helpful. But now, I'm too comfortable with the default rotor." This underscores how familiarity with existing methods can shape user preferences, even when newer systems offer benefits, especially for users who are still building their skills.

P1 described the interface as "simple, not too many gestures — I would use it every day." P2 echoed this sentiment, saying, "Very helpful, I would use it daily. I don't have to touch my mobile." Participants generally found the system efficient for everyday text editing. Beyond efficiency, several users highlighted broader usability advantages. P3 emphasized that *GestureVoice* could help "people with blindness and low vision when writing text faster and more accurately," and believed the system could also help users with low motor skills, explaining that it "requires very little attention, compared to a lot of finger movement on the screen". They added that integrating the system with voice-supported platforms such as Amazon or YouTube would make searching and editing smoother and less physically demanding. P4 appreciated the tactile feedback of the crown-based cursor navigation, remarking, "I absolutely love the crown feature — it gives the needed control and feedback for selecting the text I want to edit." P5 also described the system as less stressful than default text editing method, saying, "This is easier to operate — I like it a great deal compared to the rotor on the phone for text corrections."

Additionally, participants reported being comfortable using *GestureVoice* in both public and home settings (mean = 6.5), and most preferred it over the default rotor for editing text (mean = 6.75), as shown in Table 7. It was especially favored for composing short messages (mean = 6.63), making it a strong fit for quick, everyday interactions. The consistently higher ratings for our gesture-based approach across all subjective measures validate our core design principle of mapping physical gestures to text navigation actions. The significant improvement in mental demand ratings (6.875 vs. 3.5) confirms our hypothesis that the abstraction layer present in traditional screen reader navigation has a substantial cognitive load, given the absence of visual feedback. Our design approach proves valuable for text editing tasks.

6 LIMITATIONS AND FUTURE WORK

In this section, we reflect on the current limitations of our system and highlight potential directions for future work.

1. Performance in Mobile Scenarios

While *GestureVoice* performed well in controlled environments, its effectiveness in mobile settings (e.g., walking, navigating public spaces, or using public transportation) remains untested. For instance, cursor control via the watch crown relies on finger movements, which requires a steady hand. This can be difficult to maintain while walking or trying to stay balanced on a moving vehicle. As part of future work, we will conduct more realistic experiments in outdoor environments (or simulate outdoor environments in controlled settings).

Questions	Mean Score (out of 7)	Median Score (out of 7)
I find performing hand gestures in *GestureVoice* challenging.	2.37	1
I find rotating the watch crown in *GestureVoice* challenging.	1.75	1.5
I would use *GestureVoice* for text editing in public settings.	6.5	7
I would use *GestureVoice* for text editing at home.	6.5	7
I would use *GestureVoice* to compose short messages.	6.63	7
I would use *GestureVoice* to compose longer text entries.	6.0	6.5
I prefer using *GestureVoice* over default screen reader rotor for text editing.	6.75	7

Table 7: Mean and median Likert scale scores across 8 blind participants. The scale ranges from 1 (strongly disagree) to 7 (strongly agree). Higher scores indicate stronger agreement with each statement.

2. Need for diverse gestures and wearable interfaces

GestureVoice relies on mid-air gestures and crown rotation to support text editing without requiring direct interaction with the touchscreen. While this input method reduces reliance on on-screen touch, it may introduce physical challenges for some users. In general, gesture preferences can vary widely depending on factors such as motor control, comfort, and what feels intuitive. Participant P4 remarked, "The circular gesture doesn't come naturally to me… I think it's a me thing. Something like flicking my wrist would be more my style." Participant P6 noted, "I'm a bit rough with my hand movements, so these gestures feel kind of weird for me. Something more subtle - like rotating just one finger would work better". These comments suggest that a single, fixed gesture may not suit everyone equally. Offering customizable or alternative gestures could better support users with different movement styles and physical needs.

Similarly, many users already use other wearables, such as headphones or earbuds, in their daily routines. Participant P2 mentioned, "I use a watch sometimes, but not as frequently as my headphones. I think it would be great if I could use that instead." Participant P4 expressed interest in wearable rings: "I like the idea of rings. I could just wear the ring, and instead of rotating my hand, I could rotate my finger." These preferences point to the value of supporting interaction through other wearable devices going beyond smartwatches.

3. Privacy and Recognition Constraints in Voice Input

GestureVoice supports voice input to perform simple operations such as deleting, inserting, or replacing text, making it easier to edit without direct interaction with the smartphone screen. While this approach works well indoors, voice input can be less practical in outdoor or public settings. Even with short simple commands, users may feel self-conscious using voice input in public - especially when editing something private. To avoid drawing attention, they may lower their voice, which can make it harder for the system to pick up commands accurately. As the advancements occur in the field of Automatic Speech Recognition (ASR), more powerful models can be explored for our use case. While *GestureVoice* relies on short voice commands and is therefore less susceptible to noise compared to longer sentences, in future work we will study the effect of various noise environments to identify limitations and potential areas for improvement.

4. Number and Diversity of the participants

Our findings are based on a user study with eight participants. While the study demonstrated that *GestureVoice* can improve the speed and accuracy of text editing for blind users, the small sample size limits the generalizability of this finding. The participant group did not represent a broad spectrum of physical abilities, such as limited arm mobility or hand tremors. Given the promising result, a study with a broader set of blind user population will make an even stronger case to deploy a system like *GestureVoice*. Also, our evaluation was conducted with researcher-generated sentences rather than user-generated content, which may not fully capture the complexity and variability of real-world text editing scenarios.

7 CONCLUSION

In this work, we present *GestureVoice*, a multi-modal, text editing system that makes editing significantly easier for blind people. Our formative study showed that text editing remains a significant challenge for blind people, primarily due to the difficulty in navigating to the editing location and because of the difficulty in performing certain touchscreen actions. To this end, we developed a multimodal gesture-based interaction paradigm where a user can use simple hand gestures to navigate to the error location quickly and easily and use short voice commands to make the edits. The hand gestures are detected using sensors on a commodity smartwatch, which is commonly available today. Our user study with 8 blind participants demonstrated a substantial 53.80% reduction in editing time compared to the default screen reader based text editing. Our qualitative evaluation showed that users preferred the gesture and voice based text editing interaction because they found it to be less physically and mentally demanding, and reported feeling comfortable using the system in a variety of mobile scenarios.

ACKNOWLEDGMENTS

We sincerely thank the reviewers for their insightful comments and suggestions. This work was supported in part by a Google Research Scholar award, the National Science Foundation under award numbers 2113485, 2153056, the National Institutes of Health under award numbers R01EY03568801, R01EY03008503, and the DoD-USAMRAA under award number HT94252410098.

REFERENCES

[1] Ali Abdolrahmani, Ravi Kuber, and Amy Hurst. 2016. An empirical investigation of the situationally-induced impairments experienced by blind mobile device users. In *Proceedings of the 13th International Web for All Conference*. 1–8.
[2] Apple VoiceOver. [n. d.]. https://www.apple.com/voiceover/info/guide/_1121.html
[3] Apple VoiceOver Rotor. [n. d.]. https://support.apple.com/en-us/111796
[4] Ahmed Sabbir Arif, Sunjun Kim, Wolfgang Stuerzlinger, Geehyuk Lee, and Ali Mazalek. 2016. Evaluation of a smart-restorable backspace technique to facilitate text entry error correction. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems*. 5151–5162.
[5] Shiri Azenkot and Nicole B Lee. 2013. Exploring the use of speech input by blind people on mobile devices. In *Proceedings of the 15th international ACM SIGACCESS conference on computers and accessibility*. 1–8.
[6] Shiri Azenkot, Jacob O Wobbrock, Sanjana Prasain, and Richard E Ladner. 2012. Input finger detection for nonvisual touch screen text entry in Perkinput. In *Proceedings of graphics interface 2012*. 121–129.

[7] Xiaojun Bi, Tom Ouyang, and Shumin Zhai. 2014. Both complete and correct? multi-objective optimization of touchscreen keyboard. In *Proceedings of the SIGCHI conference on human factors in computing systems*. 2297–2306.

[8] Brian Kemler. [n. d.]. https://blog.google/products/android/all-new-talkback/

[9] Maria Claudia Buzzi, Marina Buzzi, Barbara Leporini, and Amaury Trujillo. 2014. Designing a text entry multimodal keypad for blind users of touchscreen mobile phones. In *Proceedings of the 16th international ACM SIGACCESS conference on Computers & accessibility*. 131–136.

[10] 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*. 487–499.

[11] Nem Khan Dim, Chaklam Silpasuwanchai, Sayan Sarcar, and Xiangshi Ren. 2016. Designing mid-air TV gestures for blind people using user-and choice-based elicitation approaches. In *Proceedings of the 2016 ACM conference on designing interactive systems*. 204–214.

[12] Matthew Ernst, Travis Swan, Victor Cheung, and Audrey Girouard. 2017. Typhlex: Exploring deformable input for blind users controlling a mobile screen reader. *IEEE Pervasive Computing* 16, 4 (2017), 28–35.

[13] 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*. 910–921.

[14] Shirin Feiz and IV Ramakrishnan. 2019. Exploring feasibility of wrist gestures for non-visual interactions with wearables. In *Proceedings of the 16th international web for all conference*. 1–4.

[15] Vittorio Fuccella, Poika Isokoski, and Benoit Martin. 2013. Gestures and widgets: performance in text editing on multi-touch capable mobile devices. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. 2785–2794.

[16] Vittorio Fuccella and Benoît Martin. 2017. Touchtap: A gestural technique to edit text on multi-touch capable mobile devices. In *Proceedings of the 12th biannual conference on Italian SIGCHI chapter*. 1–6.

[17] Debjyoti Ghosh, Pin Sym Foong, Shengdong Zhao, Di Chen, and Morten Fjeld. 2018. EDITalk: Towards designing eyes-free interactions for mobile word processing. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*. 1–10.

[18] Debjyoti Ghosh, Pin Sym Foong, Shengdong Zhao, Can Liu, Nuwan Janaka, and Vinitha Erusu. 2020. Eyeditor: Towards on-the-go heads-up text editing using voice and manual input. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems*. 1–13.

[19] Debjyoti Ghosh, Can Liu, Shengdong Zhao, and Kotaro Hara. 2020. Commanding and re-dictation: Developing eyes-free voice-based interaction for editing dictated text. *ACM Transactions on Computer-Human Interaction (TOCHI)* 27, 4 (2020), 1–31.

[20] Joshua Goodman, Gina Venolia, Keith Steury, and Chauncey Parker. 2002. Language modeling for soft keyboards. In *Proceedings of the 7th international conference on Intelligent user interfaces*. 194–195.

[21] Google TalkBack. [n. d.]. https://support.google.com/accessibility/android/answer/6283677?hl=en

[22] Jonggi Hong, Christine Vaing, Hernisa Kacorri, and Leah Findlater. 2020. Reviewing speech input with audio: differences between blind and sighted users. *ACM Transactions on Accessible Computing (TACCESS)* 13, 1 (2020), 1–28.

[23] Shashank Mohan Jain. 2022. Hugging face. In *Introduction to transformers for NLP: With the hugging face library and models to solve problems*. Springer, 51–67.

[24] Clare-Marie Karat, Christine Halverson, Daniel Horn, and John Karat. 1999. Patterns of entry and correction in large vocabulary continuous speech recognition systems. In *Proceedings of the SIGCHI conference on Human Factors in Computing Systems*. 568–575.

[25] Pegah Karimi, Erin Brady, Aqueasha Martin-Hammond, and Davide Bolchini. 2023. "I stepped into a puddle": Non-Visual Texting in Nomadic Contexts. In *Proceedings of the 20th International Web for All Conference*. 32–43.

[26] Akif Khan and Shah Khusro. 2021. An insight into smartphone-based assistive solutions for visually impaired and blind people: issues, challenges and opportunities. *Universal Access in the Information Society* 20, 2 (2021), 265–298.

[27] Md Nafiz Hasan Khan, Md Amit Hasan Arovi, Hasan Mahmud, Md Kamrul Hasan, and Husne Ara Rubaiyeat. 2015. Speech based text correction tool for the visually impaired. In *2015 18th International Conference on Computer and Information Technology (ICCIT)*. IEEE, 150–155.

[28] Prerna Khanna, Shirin Feiz, Jian Xu, IV Ramakrishnan, Shubham Jain, Xiaojun Bi, and Aruna Balasubramanian. 2023. AccessWear: Making Smartphone Applications Accessible to Blind Users. In *Proceedings of the 29th Annual International Conference on Mobile Computing and Networking*. 1–16.

[29] Prerna Khanna, IV Ramakrishnan, Shubham Jain, Xiaojun Bi, and Aruna Balasubramanian. 2024. Hand Gesture Recognition for Blind Users by Tracking 3D Gesture Trajectory. In *Proceedings of the CHI Conference on Human Factors in Computing Systems*. 1–15.

[30] Kevin Larson and David Mowatt. 2003. Speech error correction: the story of the alternates list. *International Journal of Speech Technology* 6 (2003), 183–194.

[31] Huy Viet Le, Sven Mayer, Maximilian Weiß, Jonas Vogelsang, Henrike Weingärtner, and Niels Henze. 2020. Shortcut gestures for mobile text editing on fully touch sensitive smartphones. *ACM Transactions on Computer-Human Interaction (TOCHI)* 27, 5 (2020), 1–38.

[32] Barbara Leporini, Maria Claudia Buzzi, and Marina Buzzi. 2012. Interacting with mobile devices via VoiceOver: usability and accessibility issues. In *Proceedings of the 24th Australian computer-human interaction conference*. 339–348.

[33] Mingzhe Li, Mingming Fan, and Khai N Truong. 2017. BrailleSketch: A gesture-based text input method for people with visual impairments. In *Proceedings of the 19th International ACM SIGACCESS Conference on Computers and Accessibility*. 12–21.

[34] Guanhong Liu, Yizheng Gu, Yiwen Yin, Chun Yu, Yuntao Wang, Haipeng Mi, and Yuanchun Shi. 2020. Keep the phone in your pocket: Enabling smartphone operation with an imu ring for visually impaired people. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 4, 2 (2020), 1–23.

[35] Meethu Malu, Pramod Chundury, and Leah Findlater. 2018. Exploring accessible smartwatch interactions for people with upper body motor impairments. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*. 1–12.

[36] Arthur E McNair and Alex Waibel. 1994. Improving recognizer acceptance through robust, natural speech repair. In *Proc. ICSLP 1994*. 1299–1302.

[37] National Federation of the Blind. [n. d.]. https://nfb.org/images/nfb/documents/pdf/braille_literacy_report_web.pdf Accessed: 2024-09-10.

[38] Sharon Oviatt and Robert VanGent. 1996. Error resolution during multimodal human-computer interaction. In *Proceeding of Fourth International Conference on Spoken Language Processing. ICSLP'96*, Vol. 1. IEEE, 204–207.

[39] 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.

[40] Matheus Vieira Portela and David Rozado. 2014. Gaze enhanced speech recognition for truly hands-free and efficient text input during hci. In *Proceedings of the 26th Australian Computer-Human Interaction Conference on Designing Futures: the Future of Design*. 426–429.

[41] Lorenzo Porzi, Stefano Messelodi, Carla Mara Modena, and Elisa Ricci. 2013. A smart watch-based gesture recognition system for assisting people with visual impairments. In *Proceedings of the 3rd ACM international workshop on Interactive multimedia on mobile & portable devices*. 19–24.

[42] Gulnar Rakhmetulla, Yuan Ren, and Ahmed Sabbir Arif. 2023. GeShort: One-Handed Mobile Text Editing and Formatting with Gestural Shortcuts and a Floating Clipboard. *Proceedings of the ACM on Human-Computer Interaction* 7, MHCI (2023), 1–23.

[43] André Rodrigues, Hugo Nicolau, Kyle Montague, João Guerreiro, and Tiago Guerreiro. 2020. Open challenges of blind people using smartphones. *International Journal of Human–Computer Interaction* 36, 17 (2020), 1605–1622.

[44] Stan Salvador and Philip Chan. 2007. Toward accurate dynamic time warping in linear time and space. *Intelligent Data Analysis* 11, 5 (2007), 561–580.

[45] Johan Schalkwyk, Doug Beeferman, Françoise Beaufays, Bill Byrne, Ciprian Chelba, Mike Cohen, Maryam Kamvar, and Brian Strope. 2010. "Your word is my command": Google search by voice: A case study. In *Advances in Speech Recognition: Mobile Environments, Call Centers and Clinics*. Springer, 61–90.

[46] Korok Sengupta, Sabin Bhattarai, Sayan Sarcar, I Scott MacKenzie, and Steffen Staab. 2020. Leveraging error correction in voice-based text entry by Talk-and-Gaze. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems*. 1–11.

[47] Caleb Southern, James Clawson, Brian Frey, Gregory Abowd, and Mario Romero. 2012. An evaluation of BrailleTouch: mobile touchscreen text entry for the visually impaired. In *Proceedings of the 14th international conference on Human-computer interaction with mobile devices and services*. 317–326.

[48] Daniel Trindade, André Rodrigues, Tiago Guerreiro, and Hugo Nicolau. 2018. Hybrid-brailler: Combining physical and gestural interaction for mobile braille input and editing. In *Proceedings of the 2018 CHI conference on human factors in computing systems*. 1–12.

[49] Keith Vertanen and Per Ola Kristensson. 2009. Automatic selection of recognition errors by respeaking the intended text. In *2009 IEEE Workshop on Automatic Speech Recognition & Understanding*. IEEE, 130–135.

[50] Keith Vertanen and Per Ola Kristensson. 2010. Getting it right the second time: Recognition of spoken corrections. In *2010 IEEE Spoken Language Technology Workshop*. IEEE, 289–294.

[51] Keith Vertanen, Haythem Memmi, Justin Emge, Shyam Reyal, and Per Ola Kristensson. 2015. VelociTap: Investigating fast mobile text entry using sentence-based decoding of touchscreen keyboard input. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems*. 659–668.

[52] James R Williams. 1998. Guidelines for the use of multimedia in instruction. In *Proceedings of the Human Factors and Ergonomics Society Annual Meeting*, Vol. 42. SAGE Publications Sage CA: Los Angeles, CA, 1447–1451.

[53] Jian Xu, Syed Masum Billah, Roy Shilkrot, and Aruna Balasubramanian. 2019. DarkReader: bridging the gap between perception and reality of power consumption in smartphones for blind users. In *Proceedings of the 21st International ACM SIGACCESS Conference on Computers and Accessibility*. 96–104.

[54] Hanlu Ye, Meethu Malu, Uran Oh, and Leah Findlater. 2014. Current and future mobile and wearable device use by people with visual impairments. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. 3123–3132.

[55] Mingrui Zhang and Jacob O Wobbrock. 2020. Gedit: Keyboard gestures for mobile text editing. In *Graphics Interface 2020*.

[56] 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*. 843–855.

[57] Maozheng Zhao. 2023. *Intelligent Multimodal Input Technologies on Mobile Devices*. Ph. D. Dissertation. State University of New York at Stony Brook.

[58] 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*. 162–178.

[59] Maozheng Zhao, Henry Huang, Zhi Li, Rui Liu, Wenzhe Cui, Kajal Toshniwal, Ananya Goel, Andrew Wang, Xia Zhao, Sina Rashidian, et al. 2022. Eyesaycorrect: Eye gaze and voice based hands-free text correction for mobile devices. In *Proceedings of the 27th International Conference on Intelligent User Interfaces*. 470–482.

[60] 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*. 1–17.