TouchEditor: Interaction Design and Evaluation of a Flexible Touchpad for Text Editing of Head-Mounted Displays in Speech-unfriendly Environments

LISHUANG ZHAN, Xiamen University, China
TIANYANG XIONG, Xiamen University, China
HONGWEI ZHANG, Xiamen University, China
SHIHUI GUO[*], Xiamen University, China
XIAOWEI CHEN, Xiamen University, China
JIANGTAO GONG, Institute for AI Industry Research (AIR), Tsinghua University, China
JUNCONG LIN, Xiamen University, China
YIPENG QIN, Cardiff University, United Kingdom

A text editing solution that adapts to speech-unfriendly (inconvenient to speak or difficult to recognize speech) environments is essential for head-mounted displays (HMDs) to work universally. For existing schemes, *e.g.*, touch bar, virtual keyboard and physical keyboard, there are shortcomings such as insufficient speed, uncomfortable experience or restrictions on user location and posture. To mitigate these restrictions, we propose TouchEditor, a novel text editing system for HMDs based on a flexible piezoresistive film sensor, supporting cursor positioning, text selection, text retyping and editing commands (*i.e.*, *Copy*, *Paste*, *Delete*, etc.). Through literature overview and heuristic study, we design a pressure-controlled menu and a shortcut gesture set for entering editing commands, and propose an area-and-pressure-based method for cursor positioning and text selection that skillfully maps gestures in different areas and with different strengths to cursor movements with different directions and granularities. The evaluation results show that TouchEditor i) adapts to various contents and scenes well with a stable correction speed of 0.075 corrections per second; ii) achieves 95.4% gesture recognition accuracy; iii) reaches a considerable level with a mobile phone in text selection tasks. The comparison results with the speech-dependent EYEditor and the built-in touch bar further prove the flexibility and robustness of TouchEditor in speech-unfriendly environments.

CCS Concepts: • **Human-centered computing → Gestural input**.

Additional Key Words and Phrases: gesture design, text editing, flexible touchpad, head-mounted displays

ACM Reference Format:
Lishuang Zhan, Tianyang Xiong, Hongwei Zhang, Shihui Guo, Xiaowei Chen, Jiangtao Gong, Juncong Lin, and Yipeng Qin. 2023. TouchEditor: Interaction Design and Evaluation of a Flexible Touchpad for Text Editing of Head-Mounted Displays in Speech-unfriendly Environments. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.* 7, 4, Article 198 (December 2023), 29 pages. https://doi.org/10.1145/3631454

[*]Corresponding author

Authors' addresses: Lishuang Zhan, zzzlis@stu.xmu.edu.cn, Xiamen University, Xiamen, China; Tianyang Xiong, Xiamen University, Xiamen, China; Hongwei Zhang, Xiamen University, Xiamen, China; Shihui Guo, guoshihui@xmu.edu.cn, Xiamen University, Xiamen, China; Xiaowei Chen, Xiamen University, Xiamen, China; Jiangtao Gong, Institute for AI Industry Research (AIR), Tsinghua University, Beijing, China; Juncong Lin, Xiamen University, Xiamen, China; Yipeng Qin, Cardiff University, Cardiff, United Kingdom.

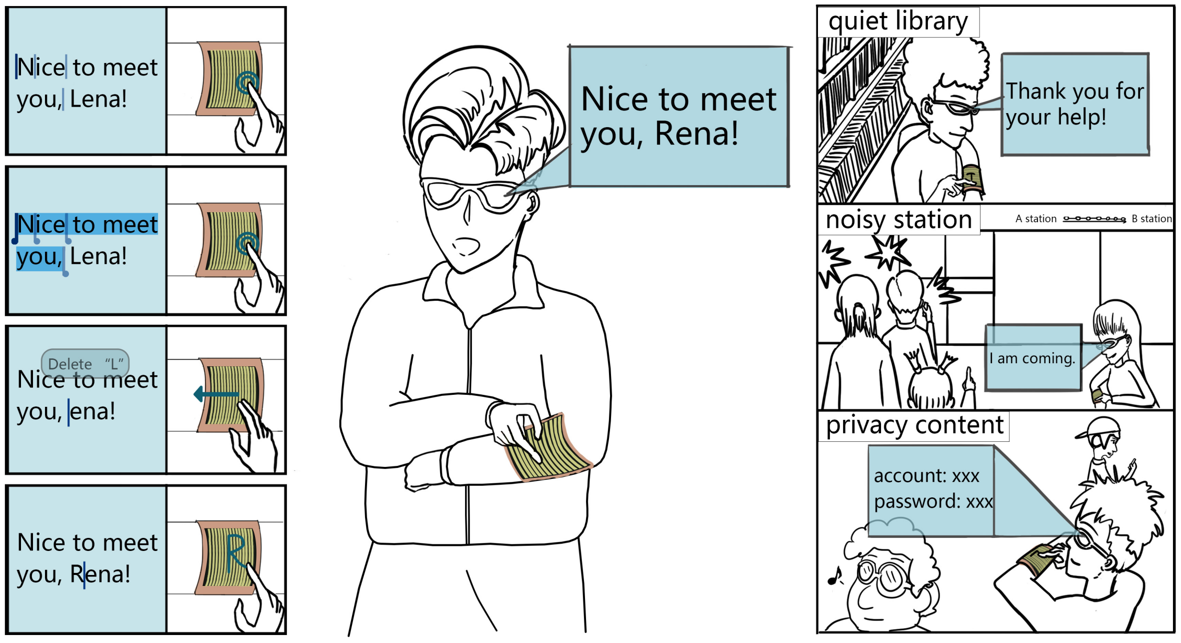


Fig. 1. The user employs voice input for text and utilizes TouchEditor to correct errors in speech recognition. TouchEditor is a text editing system for head-mounted displays based on a flexible touchpad, which supports cursor positioning, text selection, editing command entering and text retyping. The cursor moves with different granularities at letters, words and sentences, positively related to the pressure on the sensor. Editing commands such as *Copy*, *Paste* and *Delete* can also be entered by custom gestures on the touchpad. Besides, users perform text retyping using speech or handwriting. Our system shows strong adaptability to speech-unfriendly environments. Users are allowed to input various contents including private and sensitive information in quiet libraries, noisy stations and other places.

1 INTRODUCTION

Head-mounted displays (HMDs), encompassing both Augmented Reality (AR) and Virtual Reality (VR) devices, are considered an integral part of the future computing platform. Due to their visual characteristics, HMDs provide great potential in interactive teaching [56, 84], surgical simulation [75], navigation auxiliary interface [35], etc. With the increasing popularity of AR and VR applications, HMDs are poised to function as an independent platform. However, the most fundamental function, text editing, requires optimization. Operations with the built-in touch bar of HMDs are seriously limited by the small interaction area, while handheld controllers are inconvenient to carry around.

Exploring text input solutions for HMDs has long been an active research topic but researchers pay less attention to text editing tasks. Some previous studies addressed text input tasks by exploring specific gesture designs on the touch bar of HMDs [95] or by using a combination of gestures with eye tracking [4], achieving an input speed of 19.6 words per minute (WPM) and 11.04 WPM respectively. When text editing is taken into account, the task becomes more complex. Mid-air gestures [2, 18] are promising solutions but long-time use may easily lead to fatigue and seem strange on social occasions. Some researchers also explore physical keyboards [46, 86], which are more familiar to users. However, the sight occlusion caused by wearing HMDs makes it difficult to use a physical keyboard. In addition, the restrictions of physical keyboards on user location and posture are more serious problems. Speech is popular due to its fast and intimate characteristics when the sensitivity of information is low or users pay less attention to privacy. However, in speech-unfriendly environments, users prefer gestures and touch-based interactions [11], as voice input i) is not a preferred option in public due to the risk of privacy and security problems, ii) relies on accurate speech recognition, which can be difficult in a noisy environment [78], and iii) people may feel awkward talking to themselves in front of others [61] and may not be allowed to speak loudly in some occasions, such as libraries. In contrast, wearable devices are considered a potential and efficient alternative in speech-unfriendly environments. Wearable devices based on peripheral pressure sensors can provide a large interaction space for various tactile gestures. The flexibility and wearability can also bring a comfortable user experience and break through the limitations of user location and posture.

In this paper, we propose TouchEditor, a text editing system for head-mounted displays based on a flexible piezoresistive film sensor (see Fig. 1). We affixed the pressure sensor to the upper forearm position of the sleeve using Velcro, creating a freely detachable touchpad. Through literature overview and heuristic study, we design a pressure-controlled menu and a shortcut gesture set on the touchpad to enter text editing commands such as *Copy*, *Paste* and *Delete*, etc. These touchpad-like operations can easily migrate users' interaction habits on other devices to TouchEditor, ensuring that users can effectively complete text editing tasks. Besides, making full use of the direction information of the two-dimension sensor plane and the pressure property of the third dimension, we develop an area-and-pressure-based interaction mean to realize cursor positioning and text selection. Different touch areas correspond to different moving directions of the cursor. Different pressure ranges from low to high are mapped to letters, words and sentences, and linked with the hierarchical movement of the cursor and the hierarchical selection of the text. As a result, users execute cursor positioning, text selection, text retyping and editing commands by performing intuitive gestures and handwriting on the flexible pressure sensor. The peripheral pressure sensor provides users with enough interaction space, and its flexibility also brings users a comfortable wearing experience. In conclusion, we encourage users to utilize TouchEditor as a text editing tool for error correction and supplementation of voice-input text across various environments.

Our contributions are summarized as follows:

- We propose a text editing system for head-mounted displays based on a flexible touchpad, providing cursor positioning, text selection, text retyping and editing command entering (*Copy*, *Paste*, *Delete*, etc). It is helpful to achieve fast editing, adapt to speech-unfriendly environments, and bring users a portable and comfortable wearing experience.
- We conduct a heuristic study and design a pressure-controlled menu and a shortcut gesture set on the touchpad to efficiently perform text editing commands on head-mounted displays. An area-and-pressure-based method is also designed for cursor movement in different directions and granularities, to achieve fast and accurate cursor positioning and text selection.
- We conduct evaluation experiments for error correction performance (0.075 corrections per second), gesture recognition accuracy (95.4%), and text selection speed of our TouchEditor, and conduct comparison experiments with EYEditor and touch bar, which proves our TouchEditor' superiority in text editing tasks under speech-unfriendly environments.

2 RELATED WORK

2.1 Text Input on Head-Mounted Displays

Text input on head-mounted displays (HMDs) has been studied for years due to the longstanding challenge of effective interaction in a limited space, but remains an active research topic. Existing works either design more complex interaction methods or extend the interaction space.

Touch Bar. Without extending the interaction space, existing works focus on designing better text input methods using the native touch bar of HMDs. Although effective (*e.g.* the input speed can reach 8.73 words per minute (WPM) [95] or even 19.6 WPM [26]), this approach has a relatively poor user experience due to the inherent dissonance between one-dimension gestures and two-dimension text input tasks. To achieve better user experiences, the community turned to the extension of the interaction space.

Physical Keyboard. Using a physical keyboard [68] allows for a large interaction space like a PC and can achieve an input speed of 24 to 67 WPM [92]. For these methods, the challenge resides mainly in tracking the relative position of the fingers and the keyboard in the presence of occlusion. Finger tracking utilizing an external camera [46] and visualization with a virtual keyboard assistant [86] are proposed to solve this problem. However, for physical keyboards, the improved interaction space and user experience comes at the cost of flexibility, *i.e.* they are restricted to a fixed location or an indoor environment.

Virtual Keyboard. A virtual keyboard is also an available solution with an input speed of 8 to 25 WPM, which has no additional physical entities and is thus more flexible than physical keyboards. Common techniques to control the virtual keyboard include mid-air gestures [2, 18, 51], eye tracking [34, 55, 71, 98], and auxiliary devices like handheld controllers [9, 40, 92, 96]. However, the use of mid-air gestures is likely to cause fatigue [36] and social problems. People are generally less willing to use mid-air gestures publicly [39], and the challenge of translating ten-finger gestures into text is exacerbated by noise in complex interaction environments [20]. Similarly, the use of eye tracking can lead to visual fatigue, and sustained focus on the screen can exacerbate eye strain. The negative impact of sunlight on eye-tracking device accuracy would also limit its outdoor usability [42]. Additionally, handheld controllers are inconvenient to carry around.

Voice Input. Voice input underpinned by speech recognition techniques has been widely used in our daily lives (*e.g.* virtual assistants). Despite its fast speed, there are many occasions where voice input is not applicable due to social and privacy concerns [11, 22]. Moreover, the accuracy of speech recognition is also a challenge [1] in noisy environments. Therefore, voice input is often used in a multi-modal context combined with other techniques to minimize its errors [79].

Wearable Device. As a novel technique to expand the interactive space, wearable devices remove the restrictions on the environment and human posture, achieving an input speed of 6 to 31 WPM and bringing users a comfortable wearing and using experience. Existing wearable typing devices usually appear in the form of gloves [6, 50, 89], wristbands [7, 60], rings [30, 44, 63], etc. And they often employ a thumb-to-finger touch interaction mechanism for one-handed typing, where discrete regions such as finger segments [88], touch-sensitive nails [49] and fingertips [93] are mapped to the split keyboard layouts. Although promising, the implementation of text editing with wearable devices have rarely been explored in existing works.

2.2 Text Editing on Head-Mounted Displays

An effective text editing technique is essential for complete, accurate and safe text input applications. However, text editing involves complex operations such as cursor positioning, text selection and text retyping, whose implementation usually involves new techniques or a combination of existing techniques [80].

Keyboards. There are usually backspace and direction keys on the keyboard to assist in editing tasks. In addition, some works have explored typing techniques with an automatic correction mechanism. Walker et al. [86] proposed a physical keyboard with a virtual keyboard assistant, providing automatic correction of typing errors by extending a state-of-the-art touchscreen decoder. Their evaluation results show that users wearing a head-mounted display could type at over 40 WPM while obtaining an error rate of less than 5%. Dudley et al. [13] presented a mid-air single-hand virtual keyboard, which uses a statistical decoder to infer users' intended text and to provide error-tolerant predictions. With word predictions and two hours of practice, their system achieves a mean input speed of 18 WPM with a mean error rate of 1%. The performance of state-of-the-art auto-correct methods is usually relatively high, but any errors that occur can be difficult to fix [73]. Moreover, as described in Sec.2.1, physical keyboards are restricted to a fixed position or an indoor environment, while mid-air gestures are likely to lead to fatigue and social problems. Besides, the challenge of finger tracking and gesture recognition is exacerbated in complex interaction environments.

Speech Recognition. SpeeG [37] is a multi-modal text input and editing system based on speech recognition and mid-air gestures. Specifically, users use mid-air gestures to zoom in and correct the characters parsed by the speech recognizer in a Dasher [87] interface, resulting in a mean input speed of 6 WPM with no errors. SpeeG2 [38] improves SpeeG by segmenting a speech sentence into words and recommending word alternatives, allowing users to select correct word combination with a sliding gesture. As a result, SpeeG2 achieves an input speed of 21.04 WPM with no errors. Both SpeeG and SpeeG2 detect gestures via a Kinect.

Adhikary and Vertanen [1] proposed a text input and editing system supporting both voice and mid-air keyboard inputs. Their evaluation results show that i) when deleting and retyping using only the keyboard with a backspace key, users can achieve an input speed of 11 WPM and an error rate of 1.2%; ii) when using both voice and keyboard to input sentences consisting of in-vocabulary words, users can achieve a faster input speed of 36 WPM and a lower error rate of 0.3%. The rationale for their high performance stems from the high-quality in-vocabulary word alternatives generated by their system that facilitate corrections.

Nevertheless, the performances of the above systems are dependent on the size of the vocabulary and the quality of recommendation, making it coarse-grained and less flexible. In contrast, our TouchEditor is fine-grained and more flexible as it allows users to insert arbitrary text anywhere with cursor movement.

Wearable Devices. EYEditor [24] is a text editing solution using speech and a ring-mouse. The ring-mouse is used for cursor positioning and text selection, and speech is used for re-speaking. For a sentence containing errors, users can correct them by re-speaking the entire sentence or only the selected content. In contrast, TouchEditor supports retyping not only by speech but also by handwritten characters. As a result, TouchEditor behaves more flexibly and robustly in dealing with different contents and scenes, especially speech-unfriendly environments. In addition, the natural gestures on the flexible pressure sensor are more understandable and learnable compared with physical keys. In Sec.6, we conducted an experiment to compare the text editing performance of TouchEditor and EYEditor in speech-unfriendly environments. Experimental results demonstrate the superiority of the proposed TouchEditor.

2.3 Application of Flexible Pressure Sensors

Flexible pressure sensors have excellent flexibility, high ductility and flexible structure. Compared with rigid sensors, they demonstrate greater potentials in wearable intelligent devices [91] and have been widely used in human-computer interaction [72], health care [3], robot touch [43], etc. Specifically, flexible sensors are designed as wearable devices placed on various human body parts, such as hands [47], wrists [23], feet [76], knees [54], etc. They are usually used to detect human physiological signals such as pulse [16] and respiration [41], and to track human motion [14, 48, 59]. It has also been recognized that flexible pressure sensors have great potentials in the robot industry, *e.g.* tactile gloves [81] or intelligent prosthetics [52].

Flexible pressure sensors are also considered a new interface of interaction commands input for mobile devices. For example, SmartSleeve [70] is a deformable textile sensor, which can sense both surface and deformation gestures such as twirl, twist, fold, push and stretch in real-time. zPatch [77] is an eTextile patch for hover, touch, and pressure input, using both resistive and capacitive sensing, which can be used for controlling a music player, text entry and gaming input. I/O Braid [66, 67] senses proximity, touch, and twist through a spiraling, repeating braiding topology of touch matrices, which can be used for continuous real-time control, discrete actions and mode switching. Li et al. [53] presented a wrapping forming method provided to fabricate MXene textile strain (MTS). Weaving MTS yarns into the fabric as a glove can recognize sign language. Some works also place pressure sensors inside pant pockets for eyeless input [90].

In addition to interaction commands, typing [88, 93] can also be recognized through surface touch on flexible pressure sensors. For example, TEXTile [7] is a forearm interaction prototype integrated into clothing that encodes characters through the contact and release of one to four fingers with the fabric surface. Lee et al. [49] proposed a split keyboard layout on touch-sensitive nails to explore single-handed wearable typing using intra-hand touches between the thumb and fingers. As a result, users entered text at a speed of more than 30 WPM, with an error rate of about 4%. Nevertheless, the implementation of text editing and related functions with wearable devices have rarely been explored in existing works. In contrast, TouchEditor supports fine-grained text editing and retyping by making full use of the interaction space of the pressure sensor to recognize gestures and handwritten characters by detecting successive pressure points in the touch area.

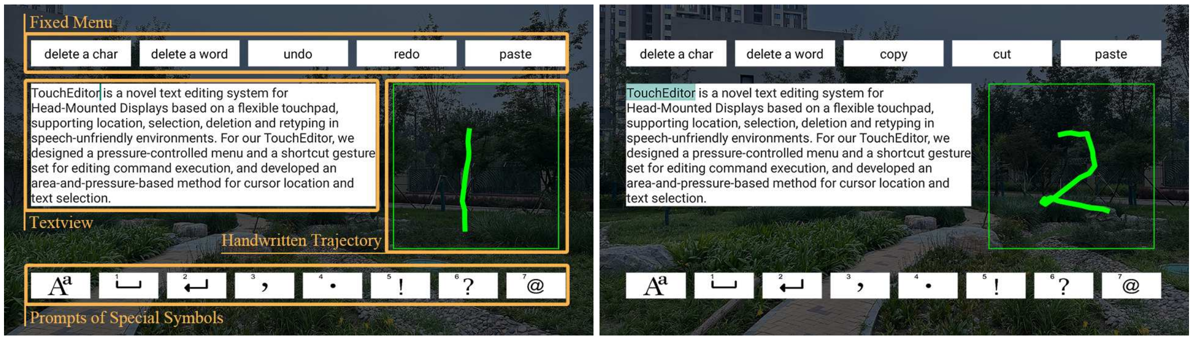
3 INTERACTION DESIGN

We aim to realize a complete text editing system based on a flexible touchpad, including cursor positioning, text selection, editing command entering, and text retyping. To this end, we carried out a series of interaction design activities. Generally speaking, i) through literature overview, we propose an area-and-pressure-based method for cursor positioning and text selection; ii) Through a heuristic study, we design a pressure-controlled menu and a shortcut gesture set to enter editing commands; iii) For text retyping, we provide two methods, handwriting on the touchpad following Handwriting Velcro [17] and speech recognition based on Iflytek SDK. As an extension to handwriting, we introduced support for letter case conversion, as well as commonly used special symbols. Specifically, a tap gesture quickly following a handwritten character is used to achieve character formatting conversion. Users switch between uppercase and lowercase letters by tapping after handwriting a letter; and they input corresponding special symbols by tapping after handwriting a number from 1 to 7 based on prompts at the bottom of the user interface shown in Fig. 2. Note that unlike Handwriting Velcro, we focus on text editing tasks of head-mounted displays (HMDs).

3.1 Cursor Positioning and Text Selection

3.1.1 Design Inspiration. Cursor positioning and text selection are necessary steps before executing many text editing operations such as *Delete*, *Copy*, etc [82]. A good cursor operation solution can significantly improve the efficiency of text editing. Handle-based techniques are the predominant form of cursor positioning and text selection on touchscreen mobile devices, which draw on the familiar metaphor of desktop computers and provide users with low learning costs. Different from dragging the handle to a specified position with mouse or finger on computers and mobile phones, the more prominent extensions for touch interaction on a touchpad using more information of the finger are finger identification [28, 29], finger contact size [10], finger force [8, 25], finger orientation [85], finger touch area [65], etc. Inspired by these, we correspond different touch areas of the finger to different moving directions of the handle. In addition, following Goguey et al. [25], we relate the finger's touch force to the handle's moving granularity. Finally, intuitive and accurate handle control in different directions and granularities is realized.

3.1.2 Area-and-Pressure-Based Cursor Positioning and Text Selection. *Tap* and *Long Press* are two of the most commonly used tactile interactive gestures [15, 74]. Tapping and long pressing in different areas and long pressing with different strengths can cover the movement requirements of the handle in different directions and granularities. All gestures in cursor operation are single-fingering. Furthermore, this interactive scheme is suitable for both cursor positioning and text selection tasks without conflict, as they would not occur simultaneously.



TouchEditor is a novel text editing system for Head-Mounted Displays based on a flexible touchpad, supporting location, selection, deletion and retyping in speech-unfriendly environments. For our TouchEditor, we designed a pressure-controlled menu and a shortcut gesture set for editing command execution, and developed an area-and-pressure-based method for cursor location and text selection.

(a) Cursor State (b) Selection State

Fig. 2. User interfaces when no text is selected (a) and when there is text selected (b). The user interface consists of four main panels: a fixed menu at the top, a text box and a trace box at the center, and prompts for special symbols at the bottom.

When in the cursor state (Fig. 2a), users are allowed to select the word closest to the cursor by *Single-finger Double Tap* to switch to the selection state (Fig. 2b). The detailed interaction scheme is described as follows.

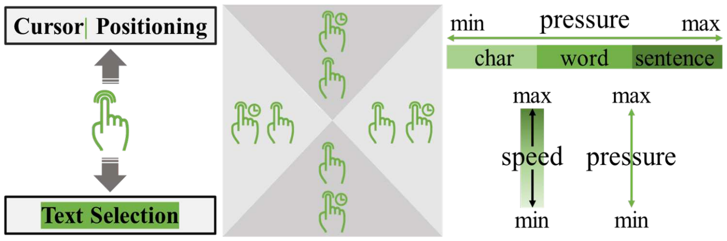


Fig. 3. The area-and-pressure-based gesture design for cursor positioning and text selection.

Cursor Positioning. The gestures for cursor operation include *Tap* and *Long Press* in four areas (see Fig.3), *i.e.*, 4 areas × 2 gestures = 8 gestures. The relationship between gestures and commands is as follows.

- Tap: The cursor moves one unit in the direction of the corresponding area, *i.e.*, the cursor moves one letter at a time horizontally or one line vertically.
- Long Press: The cursor moves continuously in the direction of the corresponding area. When the target direction is horizontal, the pressure is positively correlated with the granularity of movement. As a result, the cursor moves horizontally over the letters, words and sentences according to different pressures. When the target direction is vertical, there is a positive correlation between pressure and speed of movement. Hence, the cursor moves vertically in rows and its speed varies with the pressure.

Text Selection. Different from the cursor positioning task, there are two cursors (start and end cursors) to be controlled in the text selection task. Unlike a mobile phone where users change the selection range by moving the cursors at both ends, we propose a different text selection method for our TouchEditor. Specifically, we mimic the text selection method of a computer and control the two cursors one by one: once the initial selection is confirmed, the start cursor is fixed and only the end cursor moves. In this case, the text selection task is regarded as the positioning task of the end cursor.

3.2 Editing Command Entering

3.2.1 Heuristic Study.

Research Questions. A common way to enter commands in interactive applications is using a menu. Bailly et al. [5] characterized the design space of menus. The command entering process of a menu system includes menu activation, item selection, item activation and command execution (menu releasing). For faster interactions that allow users to focus on their task, menu items can also be activated by either keyboard shortcuts or gestures. However, using a pressure sensor for menu operations could be troublesome and time-consuming as it could be difficult to find the corresponding location of a menu (item) on the non-display sensor. On the other hand, although shortcut gestures are fast and fit well to pressure sensors, memorizing too many gestures can be exhausting and costly to learn. We consider that maybe using a menu as the basic entering solution and providing shortcut gestures for frequently used commands will be the best solution, which need to be verified by the following user study. Therefore, we conduct a heuristic study to collect participants' insights through their free exploration of the touchpad, and design our editing command entering solution accordingly.

Participants and Apparatus. We invited 27 participants (15M, 12F, mean age = 21.4 years, SD = 2.54) to participate in this experiment. Among them, 13 are undergraduate students and 14 are graduate students, all majoring in computer science. Ten of them have experience in interactive system design. The experimental device used is a flexible touchpad (see Fig.8A) stuck on the upper forearm of the sleeve. To fully enhance users' imagination, we switch off the sensing system so that it neither works nor provides any feedback to users. Our research was IRB approved by the local institution of the university, and all participants signed the informed consent form before the experiments and were paid accordingly after the experiments.

Task and Procedure. Participants were asked to choose one of the three methods to enter editing commands: 1) use a menu only; 2) use gestures only; 3) combine a menu and gestures. With option 1), participants need to describe how to activate and release the menu, and how to select and activate a menu item. With option 2), participants need to design gestures for these seven common editing commands one by one: *Delete a Char*, *Delete a Word*, *Copy*, *Cut*, *Paste* [21], *Undo*, and *Redo* [94]. With option 3), in addition to describing how to operate the menu, participants also need to design gestures for some or all of the above seven commands (depending on the wishes of participants). Apart from these, participants were asked to explain their design ideas and reasons. Design activities were carried out independently among participants without communication.

Data Collection and Analysis. Two researchers participated in the independent experiment of each participant, one responsible for the interview and the other responsible for recording. After the experiment, we analyzed the experimental data and discussed the final interaction scheme with the 10 participants who have experience in interactive system design. Besides, the combined approval voting (CAV) [19] method was adopted to decide the scheme with the highest recognition for differences of opinion.

3.2.2 Pressure-Controlled Menu and Shortcut Gesture Set. As shown in Fig. 4a, 18 (66.7%) participants chose to enter editing commands by combining a menu and shortcut gestures, 6 (22.2%) participants chose to use a menu only, and only 3 (11.1%) participants chose to use gestures only. "*Gestures are really easier and faster to use, so they should be kept*," said P4 explaining his design philosophy. "*But for gestures, a novel interactive technology, I am concerned about its stability. So it is also necessary to keep the menu as an alternative.*" P12 said: "*The reason is so simple, more choices are always better.*" Others said: "*For commands that are frequently used, gestures are best used. But for those used less frequently, setting a gesture for them only adds a memory burden. Just put them on the menu.*" Based on the experimental results, we choose to combine a menu and gestures to enter editing commands, as anticipated.

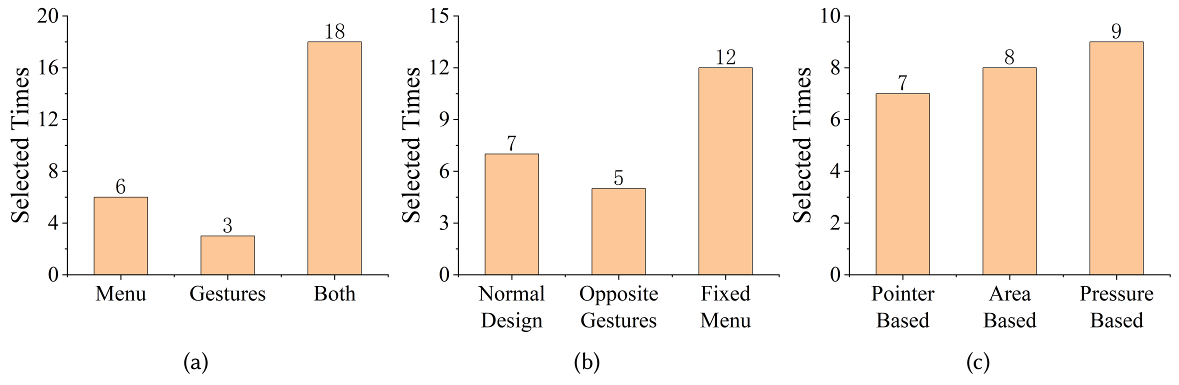


Fig. 4. (a) Preferences over the methods for editing commands. (b) Preferences over the methods of activating and releasing the menu. (c) Preferences over the methods of selecting and activating a menu item.

Pressure-Controlled Menu. Participants expressed different ideas about the method of activating and releasing the menu. Through discussion with the 10 participants, we summarized participants' designs of menu operations into three categories: *Normal Design*, *Opposite Gestures* and *Fixed Menu*.

- Normal Design: Long press to activate a context menu like on a mobile phone, and release it after the selection is completed or by tapping the surrounding blank space;
- Opposite Gestures: Activate and release a context menu with two opposite gestures respectively, such as *Slide Left and Right*;
- Fixed Menu: Use a fixed menu without activating and releasing it.

As shown in Fig. 4b, 12 of the 24 participants (50%) who chose to use a menu thought it was best to fix the menu on the user interface without additional operations to activate and release it. The remaining twelve felt they were more comfortable with a context menu. Respectively, 7 (29.2%) and 5 (20.8%) participants proposed to use *Normal Design* and *Opposite Gestures*. P11 said: "*The context menu is too cumbersome to use. I can complete a command at the time of activating it.*" Some participants also mentioned the problem of text occlusion caused by context menus. As P20 explained: "*I think the system interface should be as clear as possible for an AR environment, without confusing my vision.*"

For selecting and activating a menu item, we also summarized participants' designs into three categories: *Pointer Based*, *Area Based* and *Pressure Based*.

- Pointer Based: Drag a pointer to the corresponding item position to select, and tap to confirm;
- Area Based: Divide the sensing area into three parts: left, middle and right. Tap the left and right areas to switch items, and tap the middle area to confirm;
- Pressure Based: Long press to start switching items. Start from the first item, switch to the next at regular intervals, and release the engaged finger to confirm.

Fig. 4c shows the number of participants who preferred *Pointer Based* (7, 29.2%), *Area Based* (8, 33.3%) and *Pressure Based* (9, 37.5%) methods among the 24 participants who chose to use a menu. It can be seen that the proportion of *Pressure Based* method is slightly higher than those of the other two methods. Among them, the pointer- and area-based designs are relatively traditional and inherent more of users' habits on other devices. In contrast, the pressure-based method is relatively new but can make full use of the pressure characteristics of the touch sensor. As P15 mentioned: '*'I prefer the long press gesture because it is the way that uses the least* *gestures.*" P26 said "*I think it is a novel and efficient method. Moreover, I propose it to minimize the conflict with other operations.*" when explaining her motivation for the pressure-based design.

We finally decided to design a fixed menu and use the pressure-based gesture to select and activate a menu item. In order to avoid the conflict with the cursor operation and to expand the design space of touch interaction, the long press here is a double-finger operation. Following Corsten et al. [12], we associate the pressure with the switching speed, *i.e.*, a greater pressure leads to a faster switching speed.

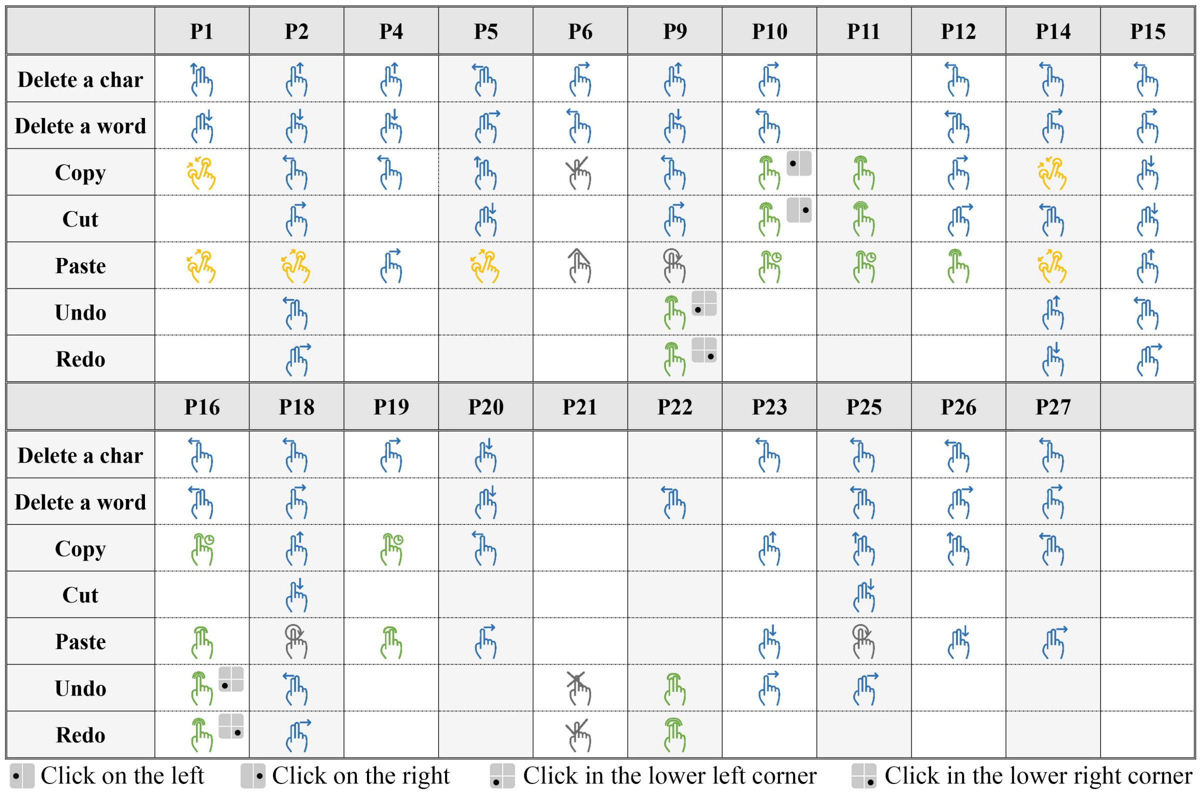

P1
P2
P4
P5
P6
P9
P10
P11
P12
P14
P15
Delete a char
Delete a word
Copy
Cut
Paste
Undo
Redo
P16
P18
P19
P20
P21
P22
P23
P25
P26
P27
Click on the left
Click on the right
Click in the lower left corner
Click in the lower right corner

Fig. 5. Gestures designed for the seven editing commands (*Delete a Char*, *Delete a Word*, *Copy*, *Cut*, *Paste*, *Undo*, and *Redo*) by 21 participants who choose to use shortcut gestures.

Shortcut Gesture Set. Fig. 5 shows the touch gestures designed for the seven editing commands of 21 participants who chose to use shortcut gestures. Most of them are sliding gestures in different directions, and gestures based on a single position without displacement, such as tapping and long pressing. There are also a small number of pattern gestures, such as drawing a circle or a tick. Unfortunately, we observe no obvious patterns or preferences in gesture design for the same command across participants. In summary, we group them into four different classes: *Slide*, *Non-Displacement*, *Pinch* and *Pattern*, described below and shown in Fig. 6.

- Slide (70.29% of participants): Slides consist of eight gestures in four directions with single and double fingers, where single-finger gestures accounted for 41.58% and double-finger ones accounted for 28.71%. However, due to the conflict between single-finger sliding and single-finger handwriting, we only retain the double-finger sliding gestures in four directions in our final design.

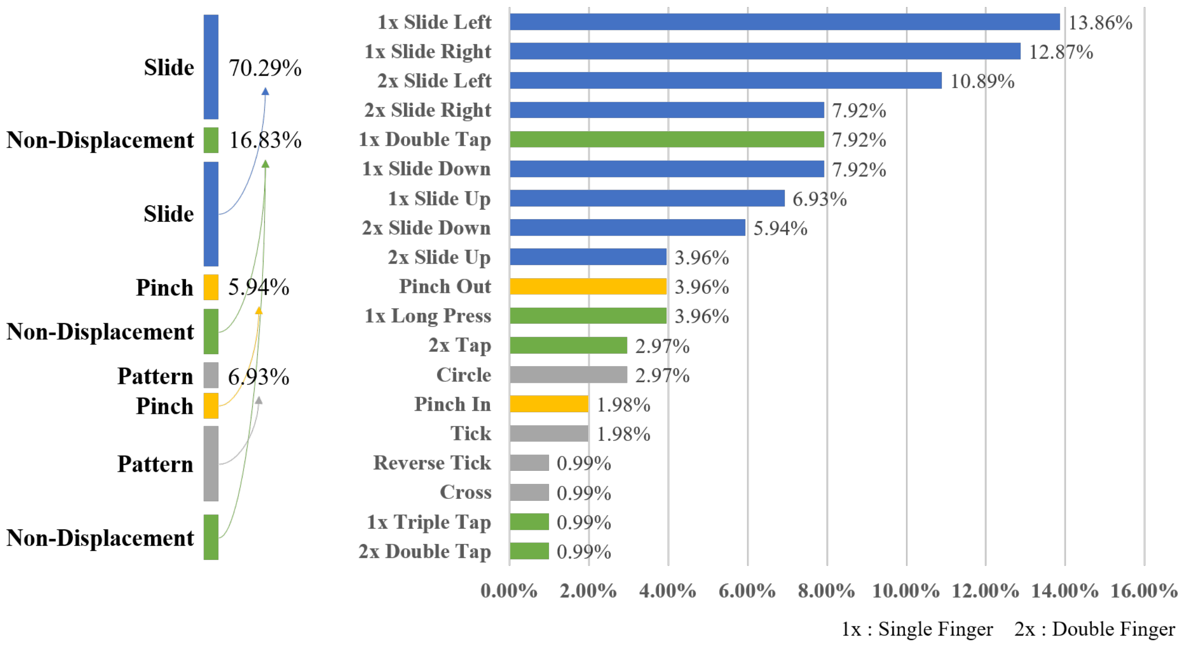


Fig. 6. A summary of all gestures in Fig. 5, including four categories: Slides, Non-displacements, Pinchs, and Patterns.

- Non-Displacement (16.83% of participants): Non-displacements include five gestures: *Double Tap*, *Triple Tap* and *Long Press* with a single finger, and *Tap* and *Double Tap* with double fingers. Two tapping gestures, *Double-finger Tap* and *Double-finger Double Tap*, are finally retained. Note that we abandon i) *Single-finger Double Tap* to unify the editing commands into double-finger gestures to facilitate users' memorization; ii) *Triple Tap* as it is not commonly used (0.99% of participants); and iii) *Long Press* to avoid conflicts with menu events.
- Pinch (5.94% of participants) & Pattern (6.93% of participants): Pinchs and Patterns are not very popular, and the above two types of gestures are enough for editing tasks. Following the *Occam's razor* philosophy, we minimize the types of gestures used and do not include them in our final gesture set.

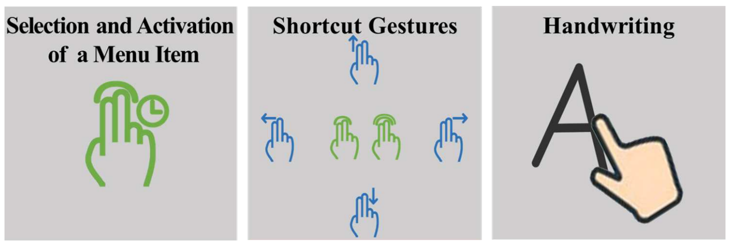


Fig. 7. The gesture to control the menu and the shortcut gesture set for entering editing commands.

Finally, we designed six double-finger gestures for editing commands: *Slide* in four directions, *Tap* and *Double Tap* (see Fig 7). However, as shown in Fig. 5, the relationship between commands and gestures mostly depends on users' personal preferences and there is no common pattern. Taking deletion as an example, different participants have different interpretations of their gesture design, *e.g.* "*Sliding right means crossing out and excluding.*", "*Sliding down has the meaning of 'negative', and deletion is a 'negative' operation.*", "*Sliding left means backspacing, and the backspace on the keyboard is used to perform deletion.*" We thus introduce the customization function. Specifically, users can assign corresponding shortcut gestures to some or all editing commands according to their preferences. Note that all the above design decisions were discussed and determined by two researchers and the 10 participants with experience in interactive system design together.

4 SYSTEM IMPLEMENTATION

4.1 Hardware

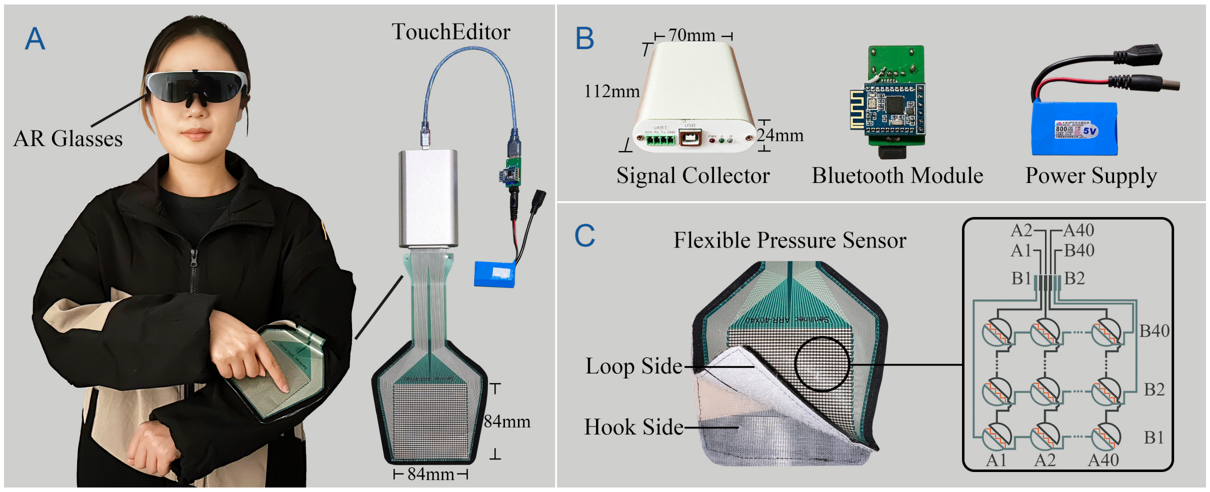


Fig. 8. TouchEditor in use (A). Three components of the sensor reading module: signal collector, Bluetooth module and power supply (B). The structure of the touchpad and the layout of sensing points on the pressure sensor (C).

Touchpad. As an interaction interface of head-mounted displays (HMDs), the proposed TouchEditor (see Fig.8A) consists of a touchpad and a sensor reading module. The primary component of the touchpad is a flexible pressure sensor, which is off-shelf provided by *Legact*[1]. The sensor is composed of flexible polyester films, highly-conductive materials and nano-scale piezoresistive materials. It is divided into the top and bottom layers of flexible pressure-sensitive films, which are bonded by double-sided adhesive to isolate the sensing areas of the upper and lower layers. When the sensing area is under pressure, the two pressure-sensitive layers contact each other, and the output resistance of the channel at the corresponding position changes. The resistance value of each channel is represented with 12 bits and digitized into a value within [0, 4095]. The sensor weighs 30 grams and its thickness is 0.2 mm. The size of sensing area is $84 \times 84\ mm^2$, and the resolution is 40×40. This implies that the size of each sensing point is $1.8 \times 1.8\ mm^2$, and the distance between two sensing points is 0.3 mm. The sensing threshold is 10 grams and the response time is <10 μs, which are both sufficiently small for convenience and interaction. The flexible pressure sensor is attached to the loop side of a Velcro patch (see Fig.8C). The hook side of the Velcro is sewn on the sleeve at the position of the upper forearm. This freely detachable

[1]https://film-sensor.com/product/rpps-1600/

design based on Velcro is flexible and convenient for clothes washing. In addition to bringing a comfortable wearing experience, it can also be carried everywhere with no restrictions on user postures. Users are allowed to use TouchEditor sitting, standing or even lying down.

Sensor Reading Module. The touchpad is connected to an 80-pin self-designed circuit for signal collection, *i.e.*, the signal collector (Fig.8B). This signal collector can be purchased from the sensor vendor upon request. Forty pins are connected to the horizontal channels (denoted as A1, A2, A3$\cdots$), and another 40 pins are connected to the vertical ones (denoted as B1, B2, B3$\cdots$). The operating voltage of the signal collector ranges from 4.5V to 5.5V, with an average operating current of approximately 80mA. For the sake of protection and portability, the circuit is covered by an aluminum shell with a size of $112{\times}70{\times}24mm^3$, with a total weight of 150 grams. The sampling channel for all sensing points is selected by a multiplexing micro-processing unit, and the frequency of data collection is 60 fps. The collected serial sensor data is transmitted to the head-mounted display via an external Bluetooth transmission module and processed by a dedicated program in real-time. The whole sensor reading module is powered by a 5V rechargeable lithium battery. For the rapid realization and verification of the prototype, we adopt a simple method to combine the separate modules directly to build the sensor reading module. In the future, we will optimize the circuit design and realize the integration and miniaturization of the whole module.

AR Glasses. The head-mounted display used in this work is Rokid Air Glass[2], which has two 1920×1080 resolution screens, two high performance denois microphones, and an Android operating system supporting WIFI5 and Bluetooth5.0 connections. It weighs approximately 83 grams. It is equipped with a 64-bit quad-core ARM processor, and configured with a 2GB memory and a 32GB flash storage. The sensor data is transmitted to the AR glasses from the sensor reading module via Bluetooth, and then the data processing and recognition program runs on the AR glasses to visualize the processed information in real-time on its screens.

4.2 Interaction Techniques

4.2.1 Data Collection and Preprocessing. We use the signal collector to capture continuous data streams frame by frame from the flexible pressure sensor. Data for each frame is recorded as a 40×40 pressure matrix with a frame rate of 60 fps. We retain data points for the [400, 4095] pressure range in the matrix and discard other points to eliminate noise and outliers. The matrix containing the position and pressure information of the reserved points is wirelessly transmitted to the AR glasses for further computation. The point with the maximum pressure value in each frame is recorded as an effective point, and all the effective points from each input are connected to form a real-time handwritten trajectory.

4.2.2 Gesture Recognition. Through a series of interactive design activities, there are a total of 16 gestures used by our TouchEditor, *i.e.*, Tap and Long Press in four areas, Slide in four directions, Double Tap, Double-finger Tap, Double-finger Double Tap and Double-finger Press. These 16 gestures can be distinguished according to the number of fingers in contact, touch type, and touch area.

Firstly, a one-scan connected-component analysis (CCA) algorithm [33] is adopted to implement multi-finger recognition, which is well-suited for real-time applications as it completes the analysis process within a single scan, thereby reducing computational costs and latency. Specifically, the seed point is set to the data point in the first row and first column of the pressure matrix, and the eight points in its neighborhood are screened. If the pressure at an adjacent point is greater than or equal to half of the seed point, they are marked as the same class. Neighborhood analysis is also performed for adjacent points until all points in the pressure matrix are classified. If the number of data points in a class is not less than five, it is recorded as a valid class. The number of valid classes indicates the number of fingers identified. Following that, we categorized these gestures into four distinct

[2]https://air.rokid.com

types, namely *Tap*, *Double Tap*, *Long Press*, and *Slide*. The categorization criteria, as presented in Table 1, are based on four conditions: trajectory length, duration, the type of the previous gesture, and the time interval with the previous gesture. Finally, the angle between a vector α and the horizontal axis is employed to signify two aspects: i) the area of tapping and long-pressing, and ii) the direction of sliding. More precisely, for tapping and long-pressing, α corresponds to the vector extending from the matrix center to the action point; whereas, for sliding, α denotes the vector spanning from the starting to the ending position of its trajectory.

Table 1. The categorization criteria for the four distinct types of gestures.

	Trajectory Length	Duration	Previous Gesture	Interval Time
Slide	⩾30	–	–	–
Long Press	<30	⩾30 frames	–	–
Tap	<30	<30 frames	Tap	⩾30 frames
			Not Tap	–
Double Tap	<30	<30 frames	Tap	<30 frames

4.2.3 Character Recognition. Inputs that do not conform to any of the aforementioned criteria are identified as handwritten characters, including both 26 letters and 10 numbers. The character recognition task is considered an image classification task of handwritten trajectories. Image classification is a significant task in the field of computer vision, and there are various implementation methods available, such as traditional machine learning approaches like SVM [52] and random forests [97], as well as deep learning methods based on CNN and RNN. Among these, Convolutional Neural Networks (CNN) [27, 57, 99] have emerged as the mainstream method for image classification. Specifically, ResNet [32], as a popular CNN model, has been widely applied in numerous visual tasks, including image recognition, image segmentation, and object detection. Therefore, we build a variant of ResNet18 by adjusting its input layer to match our input image size and its output layer to match our output dimension of 36, achieving an average test accuracy of 97.21% [17].

4.2.4 Pressure Adaptation. Due to individual differences (*e.g.* habits and strength differences), users exert different forces on the pressure sensor, which poses a challenge to the recognition of pressure-based gestures of different users. Addressing this challenge, we define the maximum pressure range that can be detected by the pressure sensor as the "global pressure range", and the actual pressure range that a user exerts practically as the "effective pressure range". When the "effective pressure range" is less than the "global pressure range", scope loss will occur. Therefore, we adaptively process the pressure value and map the "effective pressure range" to the "global pressure range":

$$\text{Adaptive Pressure} = \begin{cases} \frac{\text{pressure}-\text{top}}{\text{max}-\text{top}} * \text{max}, & \text{pressure} \geq \text{top} \\ \text{pressure}, & \text{pressure} < \text{top} \end{cases} \tag{1}$$

where *max* is the maximum value of the "global pressure range" (*i.e.*, 4095), *pressure* is the actual pressure, and *top* is the pressure peak within the first 30 frames of recognition. Since users tend to keep the same or greater strength on the pressure sensor after 30 frames, we take *top* as the baseline and perform mapping when the pressure is greater than or equal to *top*. Otherwise, the original value is retained and no mapping is performed.

5 SYSTEM EVALUATION

In this section, we present a comprehensive evaluation of our TouchEditor. We commence by providing the program performance analysis on Rokid Air Glass (Sec. 5.1). Subsequently, we conduct a series of user experiments to evaluate the error correction performance (Sec. 5.2), gesture recognition accuracy (Sec. 5.3) and text selection speed (Sec. 5.4) of our text editing system. Finally, we report the text input performance of TouchEditor (Sec. 5.5).

5.1 Program Performance Analysis

The Android program running on Rokid Air Glass receives data input from the sensor reading module and processes it for gesture recognition or character recognition. To integrate the character recognition algorithm into the Android program, we converted the trained ResNet18 model into the TensorFlow Lite format, harmoniously compatible with the Android platform. Subsequently, we employed the Android Profiler to analyze the program's performance. The analysis revealed that the program's average CPU utilization during gesture recognition tasks was approximately 4.46%, and it escalated to an average of approximately 12.85% during character recognition tasks. The character recognition process exhibited an average latency of 93.16ms; whereas the gesture recognition, founded on logical decisions, exhibited negligible latency in the microsecond range. Furthermore, the overall memory consumption of the program amounted to approximately 120MB, and the power consumption remained at a "Light" level.

5.2 Error Correction Performance

5.2.1 Participants and Apparatus. Twelve participants (6M, 6F, mean age = 22.8 years, SD = 1.82) voluntarily participated in the error correction tasks. During the experiment, participants wore the Rokid Air Glass and the jacket integrated with our TouchEditor. The equipment remained in operation throughout the whole study. To reduce the impact of proficiency, participants practiced for 20 minutes before the experiment.

5.2.2 Task and Procedure. We conducted the evaluation from two aspects: *content complexity* and *scene noise.*

Content Complexity. We leveraged three different texts with Flesch reading ease scores [45] of 79.55 (fair easy), 65.89 (standard) and 54.26 (fair difficult) to explore the impact of content complexity on user performance. We insert 3 errors (1 insertion, 1 deletion and 1 modification), 4 errors (1 insertion, 1 deletion and 2 modification), and 5 errors (2 insertion, 1 deletion and 2 modification) into the above three texts respectively and ask the participants to correct them. We recorded the completion times of these three corrections.

Scene Noise. Noise often affects the choice of input mode. Therefore, participants were asked to complete correction tasks in environments with varying levels of noise: 20-40db (quiet), 40-60db (ordinary indoor conversation) and 60-80db (noisy). The text (43 words, 198 characters) used in these correction tasks was formed with eight phrases which were randomly selected from MacKenzie & Soukoreff phrase set [58]. Besides, four errors (2 insertion and 2 modification) had been inserted to the text. The completion times for the correction tasks in the above three cases were recorded.

Among all the above tasks, participants were allowed to locate and delete the errors using gestures on the touchpad, and to retype the correct content using either speech or handwriting during the correction process. All the above tasks are repeated three times by users, and the final scores are the average of the three trials. For each case, participants were asked to fill in a satisfaction rating questionnaire and a NASA TLX questionnaire [31] respectively according to their feelings. Finally, we collected a total of 72 satisfaction rating questionnaires and 72 NASA TLX questionnaires ((3 content complexity + 3 scene noise) $\times$ 12 participants).

5.2.3 Metrics. The error correction performance of TouchEditor was evaluated from the following three metrics: *correction speed*, *satisfaction*, and *task load.*

Correction Speed (CPS, corrections per second). Correction speed is measured by the number of errors corrected per second:

$$\text{Correction Speed} = \frac{N_{error}}{T_{correction\ task}}, \tag{2}$$

where N_{error} is the total number of errors to be corrected, $T_{correction\ task}$ is the correction task completion time (seconds), *i.e.*, the time spent correcting all errors.

Satisfaction. Satisfaction is a subjective evaluation by participants. It is measured by the score of the satisfaction rating questionnaire (range 0-9), which is designed by retaining the appropriate options of the QUIS (Questionnaire For User Interaction Satisfaction) [64] and adding questions about wearability and environmental adaptability of the text editing system. A high score indicates an increased user satisfaction.

Task Load. Task load is another subjective evaluation by participants and is measured by the score of the NASA TLX questionnaire [31] (range 0-20). A high score indicates an increased difficulty in using the system.

Table 2. Statistical results of the evaluation experiment of error correction performance.

Content Complexity	Correction Speed	Satisfaction*	Task Load*
Fairly Easy	0.0754±0.0074	$7.64\pm0.64_a$	$4.74\pm2.50_a$
Standard	0.0765±0.0070	$7.38\pm0.46_{ab}$	$7.06\pm2.71_b$
Fairly Difficult	0.0750±0.0037	$6.98\pm0.60_b$	$9.38\pm4.10_b$
Scene Noise	**Correction Speed**	**Satisfaction**	**Task Load***
20-40 db	0.0771±0.0071	7.38±0.62	$4.29\pm2.52_a$
40-60 db	0.0750±0.0064	7.26±0.74	$6.52\pm3.09_{ab}$
60-80 db	0.0709±0.0050	7.26±1.00	$8.15\pm4.50_b$

*: Results of RM-ANOVA, indicating that the mean difference is significant at the 0.05 level.
ab: APA style reporting post-hoc tests [83]. There is no significant difference between groups with the same letters, *i.e.* only those with completely different letters have significant differences.

5.2.4 Results and Discussion.

Content Complexity. As shown in Table 2, as the content complexity increases, user satisfaction tend to decrease, and the task load tend to increase. We attribute this to the fact that for more complex texts, participants usually need to concentrate more on locating errors, leading to decreased user satisfaction and increased task load. Nevertheless, the performance of our system is quite robust against content complexity as the correction speed of these three cases remains largely unchanged at 0.075 corrections per second (CPS). The RM-ANOVA result shows that content complexity has no significant effect on correction speed (F = 0.180, p = 0.836). This indicates that although content complexity does have an impact on participants' scanning and locating of errors, it can be compensated for by the excellent performance of participants in editing tasks. In addition, from the results of post-hoc tests (Table 2), there are significant differences in task load (F = 4.735, p = 0.041) between fairly easy and standard texts, while there is no significant difference in task load (F = 2.684, p = 0.116) between standard and fairly difficult texts. Hence, it can be inferred that when the complexity reaches a certain degree, the task load tend to stabilize rather than continue to deteriorate. This is due to participants' gradual adaptation to increasing content complexity. Besides, post-hoc results show that there is no significant difference in satisfaction between texts of similar complexity – fairly easy vs. standard (F = 1.215, p = 0.282) and standard vs. fairly difficult (F = 3.616, p = 0.070), which indicates that our TouchEditor works well with varying levels of content complexity.

Scene Noise. As the noise increases, speech recognition is gradually invalidated and replaced by handwritten characters. As shown in Table 2, there is no significant difference in correction speed (F = 3.056, p = 0.061) and satisfaction (F = 0.115, p = 0.892) among these three noise levels, despite a slight decrease in correction speed. Participants explained: "*Although noise makes input difficult, I can still complete the task successfully because there is an optional retyping method.*" "*In the noisy environment, I realized the importance of different optional retyping methods.*" Moreover, some participants mentioned: "*When retyping a small amount of text, handwriting input is as efficient as voice input that requires an activation gesture and a 1s wait.*" These indicate that our TouchEditor, which supports editing operations such as cursor positioning, text deletion and text retyping independent of speech, is an excellent scheme for text editing in speech-unfriendly environments. In addition, the increase of noise placed additional psychological pressure on participants and increases their perceived task load. Nevertheless, post-hoc experimental results show that there is no significant difference in task load between successive noise levels. In general, our TouchEditor, a multi-modal system, has strong adaptability to various scenes without losing system performance.

5.3 Gesture Classification Accuracy

As shown in Fig.9a, there are a total of 16 gestures used by our TouchEditor, *i.e.*, Tap and Long Press in four areas, Slide in four directions, Double Tap, Double-finger Tap, Double-finger Double Tap and Double-finger Press. We conducted an experiment to evaluate the accuracy of gesture recognition. Participants were asked to input specific gestures according to the gesture prompts randomly generated. We invited a total of 15 participants, and each participant repeated the above random task 160 times. We finally collected 2,400 pieces of data (15 participants × 160 repetitions).

Fig.9b shows the confusion matrix of the experimental results. The average recognition accuracy of all gestures is 95.4%. However, the accuracy of *Long Press* is relatively low due to incorrectly identified areas or being mistaken for *Tap* or *Slide*. For *Tap* and *Double Tap*, the incorrect recognition mainly comes from the breakpoints in continuous pressure. Such breakpoints stem from the gaps between the touchpad and the forearms of thin users, despite our efforts to position the touchpad as close to the upper forearm as possible to obtain a larger base. These gaps may also result in the displacement of pressure points, which accounts for the the incorrect recognition of *Slide*. In summary, although the gap between the touchpad and small forearms is a hidden factor that may affect the recognition performance, gesture recognition still achieves high accuracy. We will address this issue in future work.

5.4 Text Selection Speed

To quantify the effectiveness of the area-and-pressure-based method in text selection tasks, we conducted an experiment. As shown in Fig.10a, six text selection tasks [25] were chosen: 1) Sub-word: 4 letters inside a 10-letter word; 2) Word and char: 2 words and 4 letters of an 8-letter word; 3) Four words; 4) Sentence: one sentence spanning 5 lines; 5) Char to paragraph: from a letter inside a word to the end of the current paragraph, spanning 5 lines; 6) Two sentences: two sentences spanning 7 lines; which covers the different types of selections that people usually use in a paragraph. In each trial, these six tasks were performed in a random order. Ten participants participated in the experiment, and each participant repeated the trial 15 times. In total, 900 text selection tasks (6 tasks × 15 repetitions × 10 participants) were recorded in this experiment.

The result of the experiment shows the mean time costs of these six text selection tasks (see Fig.10b): *Sub-word*: $\bar{x}$ = 5.17s; *Word and char*: $\bar{x}$ = 8.64s; *Four words*: $\bar{x}$ = 6.38s; *Sentence*: $\bar{x}$ = 5.34s; *Char to paragraph*: $\bar{x}$ = 3.74s; *Two sentences*: $\bar{x}$ = 5.89s. Compared with the FORCESELECT design for mobile devices proposed by Goguey et al. [25],

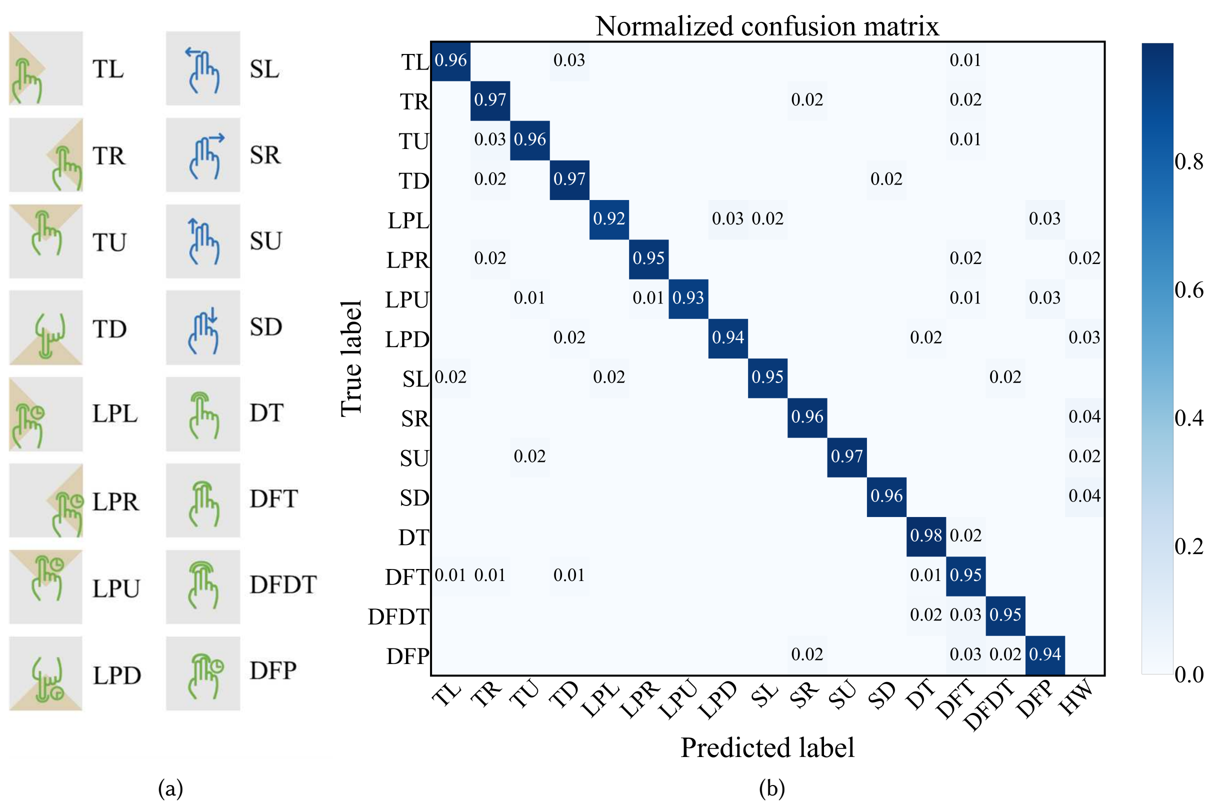
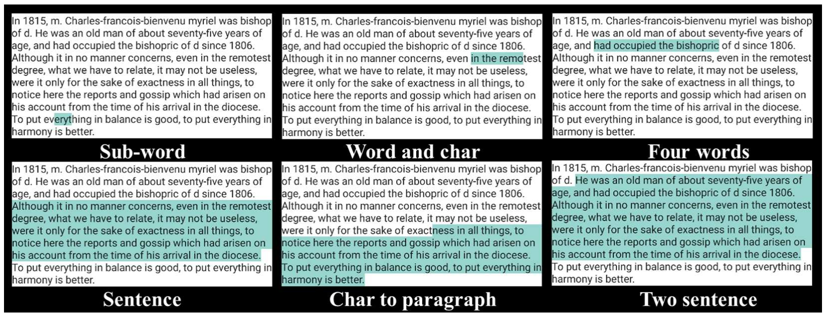


Fig. 9. (a) All 16 gestures used by our TouchEditor. (b) The confusion matrix for the recognition of the 16 gestures in (a). "HW" refers to **h**and**w**ritten characters, denoting cases where gesture input is misclassified as handwritten characters.

TouchEditor performs better on four tasks (*Sub-word*, *Four words*, *Char to paragraph*, *Two sentences*) and almost the same on the other two tasks (*Word and char*, *Sentence*). In particular, the mean time costs of FORCESELECT when conducting the *Char to paragraph* task and the *Two sentences* task are more than 10s, which is much slower than our TouchEditor. This demonstrates TouchEditor's superiority in text selection tasks spanning multiple lines and moving to the end of paragraphs, which is largely due to the fast vertical movement at the granularity of rows. Besides, thanks to the precise positioning at the granularity of characters achieved by *Tapping*, TouchEditor works well when selecting a small number of characters. Hence, with the pressure-based method, continuous movement can be achieved in time to improve speed, and different granularities can be achieved in space to ensure accuracy; with the area-based method, flexible movement in different directions is allowed. In summary, with the area-and-pressure-based method, the speed of text selection of TouchEditor can reach a considerable level with that on mobile devices.

5.5 Text Input Performance

TouchEditor is a text editing system that facilitates correcting speech recognition errors and supplementing sensitive information in specific contexts, with voice input as the primary method and handwritten input as a secondary method. Our system evaluation results demonstrate that i) the average input speed of handwriting


In 1815, m. Charles-francois-bienvenu myriel was bishop of d. He was an old man of about seventy-five years of age, and had occupied the bishopric of d since 1806. Although it in no manner concerns, even in the remotest degree, what we have to relate, it may not be useless, were it only for the sake of exactness in all things, to notice here the reports and gossip which had arisen on his account from the time of his arrival in the diocese. To put everything in balance is good, to put everything in harmony is better.

Sub-word

In 1815, m. Charles-francois-bienvenu myriel was bishop of d. He was an old man of about seventy-five years of age, and had occupied the bishopric of d since 1806. Although it in no manner concerns, even in the remotest degree, what we have to relate, it may not be useless, were it only for the sake of exactness in all things, to notice here the reports and gossip which had arisen on his account from the time of his arrival in the diocese. To put everything in balance is good, to put everything in harmony is better.

Word and char

In 1815, m. Charles-francois-bienvenu myriel was bishop of d. He was an old man of about seventy-five years of age, and had occupied the bishopric of d since 1806. Although it in no manner concerns, even in the remotest degree, what we have to relate, it may not be useless, were it only for the sake of exactness in all things, to notice here the reports and gossip which had arisen on his account from the time of his arrival in the diocese. To put everything in balance is good, to put everything in harmony is better.

Four words

In 1815, m. Charles-francois-bienvenu myriel was bishop of d. He was an old man of about seventy-five years of age, and had occupied the bishopric of d since 1806. Although it in no manner concerns, even in the remotest degree, what we have to relate, it may not be useless, were it only for the sake of exactness in all things, to notice here the reports and gossip which had arisen on his account from the time of his arrival in the diocese. To put everything in balance is good, to put everything in harmony is better.

Sentence

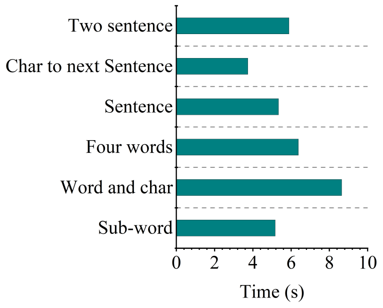
In 1815, m. Charles-francois-bienvenu myriel was bishop of d. He was an old man of about seventy-five years of age, and had occupied the bishopric of d since 1806. Although it in no manner concerns, even in the remotest degree, what we have to relate, it may not be useless, were it only for the sake of exactness in all things, to notice here the reports and gossip which had arisen on his account from the time of his arrival in the diocese. To put everything in balance is good, to put everything in harmony is better.

Char to paragraph

In 1815, m. Charles-francois-bienvenu myriel was bishop of d. He was an old man of about seventy-five years of age, and had occupied the bishopric of d since 1806. Although it in no manner concerns, even in the remotest degree, what we have to relate, it may not be useless, were it only for the sake of exactness in all things, to notice here the reports and gossip which had arisen on his account from the time of his arrival in the diocese. To put everything in balance is good, to put everything in harmony is better.

Two sentence

(a)

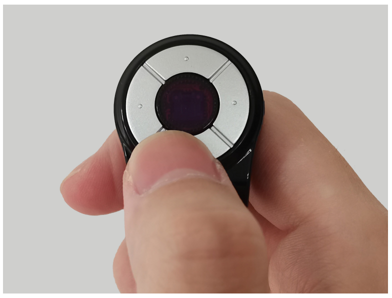


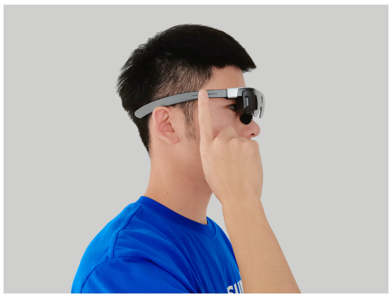
(b)

Fig. 10. (a) Six text selection tasks covering different types of selections that people usually use in a paragraph. (b) Average completion time for the six selection tasks in (a).

is 6.60 words per minute (WPM) with an error rate of 0.04; ii) Within a high-noise environment, resulting in a speech recognition error rate of approximately 0.15, the utilization of voice input followed by error correction through TouchEditor achieves an input speed of 25.72 WPM with zero errors.

6 PERFORMANCE COMPARISON


(a) EYEditor


(b) Touch Bar

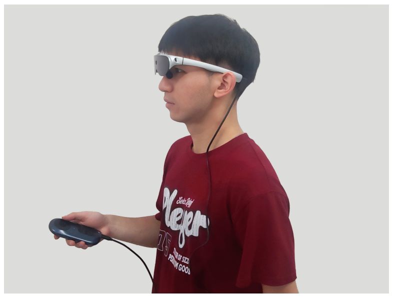

(c) Air Station

Fig. 11. Three devices in our performance comparison experiments. We compare the text editing performance of our TouchEditor with EYEditor and the touch bar, and the text input performance with the radial keyboard using Air Station.

6.1 Performance Comparison of Text Editing

In this section, we compared the comprehensive text editing performance across different devices in the task involving insertion, deletion, and modification. The comparison results demonstrated the superiority of our TouchEditor in speech-unfriendly environments over EYEditor [24] and the built-in touch bar of Rokid Glass 2[3] in terms of correction speed, user satisfaction and task load in text editing tasks.

[3]https://rokid.ai/products/rokid-glass-2/

6.1.1 Participants and Apparatus. 12 participants (8M, 4F, mean age = 23.92 years, SD = 1.26) were recruited in this experiment. All participants in comparison experiments had no experience using AR glasses and had not participated in the above-mentioned heuristic study and evaluation experiments. In order to minimize the impact of different levels of spoken English across different users on the accuracy of speech recognition, an equal number of participants who had and had not obtained a National College English Test (CET) speaking certificate were recruited. Specifically, we compare TouchEditor with the following two devices in this experiment.

- EYEditor: EYEditor is a text editing solution combining mouse and speech. Participants performed cursor operation and deletion through the touchpad in the center of the ring-mouse and buttons in four directions (Fig.11a), and overwrote the wrong content by re-speaking.
- Touch Bar: Head tracking was applied to move the pointer and tap on the touch bar (Fig.11b) was used to locate the editing positions. Then, participants performed deletion and retyping with a virtual keyboard controlled by one-dimension sliding gestures on the touch bar.

6.1.2 Task and Procedure. The experiments were carried out in an ideal environment where speech recognition could work freely without noise or other interference, and speech-unfriendly environments, *i.e.*, a noisy dining hall with an average environmental noise of 80 decibels and a public classroom, respectively. We randomly selected eight phrases from MacKenzie & Soukoreff phrase set to form a text (38 words, 188 characters) and inserted four errors (1 insertion, 1 deletion and 2 modification). In these three scenarios, participants were asked to use TouchEditor, EYEditor and the touch bar to complete the above correction task respectively, and repeated three times for each task. Participants had 20 minutes of practice for each device before the experiment. After the experiment, they filled in a satisfaction rating questionnaire and a NASA TLX questionnaire for each device.

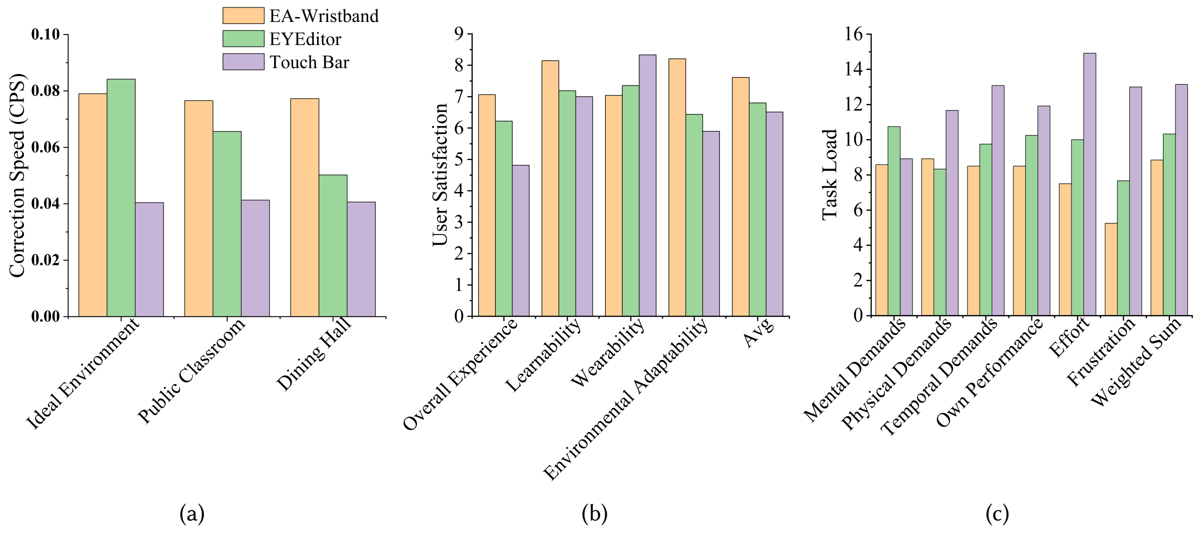


Fig. 12. The experimental results of text editing performance comparison: (a) Correction speed (CPS, corrections per second); (b) Satisfaction (range 0-9); (c) Task load (range 0-20).

6.1.3 Results and Discussion.

Correction Speed. Fig.12a shows the correction speeds of the three devices in different environments. Limited by the narrow interaction area, the touch bar exhibited the lowest correction speed of 0.040 corrections per second (CPS). In the ideal environment, the correction speed of EYEditor (0.084 CPS) was slightly higher than that of TouchEditor (0.079 CPS) without significant difference (F=0.852, 0.366). It is inseparable from the inherent advantages of physical keys of the ring-mouse, which are more stable and provide physical feedback. Among various input technologies, devices based on physical keys usually have the best performance in speed [92]. This provides us with a new avenue for future research, where we can explore the integration of physical feedback on TouchEditor through methods such as incorporating flexible actuators. Furthermore, the navigation functionality based on physical keys was not significantly affected in different environments, while the decrease in EYEditor's correction speed was primarily attributed to the difficulties in voice input during text retyping. In the noisy dining hall, the error rate of speech recognition was approximately 0.15, which significantly affected the speed of text retyping and consequently led to a decrease in correction speed (0.050 CPS). Some participants said that the noise in the dining hall was so loud that the system could not recognize the speech accurately, and they had to try many times to succeed. On the other hand, failed attempts often have a detrimental effect on participants' mindset, leading to frustration and resulting in a vicious cycle of decreased operational speed and accuracy. For the public classroom, participants became hesitant because they were embarrassed to speak publicly. Some of them attempted to speak quietly for voice input, which similarly led to an increase in speech recognition error rate (around 0.12) and a decrease in correction speed (0.066 CPS). In contrast, TouchEditor based on tactile gestures and handwritten characters maintains a high correction speed (0.078 CPS) in different environments. Nevertheless, TouchEditor allows flexible cursor movements in different directions and granularities, giving users more possibilities to obtain the best path to a specified position.

Satisfaction. The four indicators in Fig.12b are used to evaluate the satisfaction. In terms of average satisfaction score, TouchEditor (7.61) > EYEditor (6.80) > touch bar (6.51). In the detailed evaluation, i) TouchEditor scores significantly higher than EYEditor and touch bar for environmental adaptability. According to participants' descriptions, voice input is exceptionally convenient and fast, yet speech recognition is indeed vulnerable to scene noise. To compensate for the errors of voice input, they had to undo the input frequently, which led to poor user experiences. Some participants said that they were not used to voice input, especially when talking about sensitive content, and it seems silly to read digital strings in public. Some others also suggested that for more letters to retype, using voice input proved to be quicker, while for fewer letters, handwritten input was more convenient. As described by Oviatt [69]: "*A multi-modal architecture can function more robustly than any individual recognition technology that is inherently error prone, including spoken-language systems.*" ii) Regarding the overall experience and learnability, TouchEditor similarly achieved higher scores compared to EYEditor and the touch bar. Participants generally believe that our gesture design based on area and pressure is easy to understand and remember. The narrow interactive area of the touch bar and the fatigue caused by eye tracking greatly affect users' interactive experience. As for EYEditor, the interaction design implemented on a innovative mouse for intricate text editing tasks necessitated users to invest more time in learning and adapting. Furthermore, users perceived that TouchEditor offers more comprehensive functionalities compared to the other two options. iii) In terms of wearing experience, the touch bar scored the highest as it is not reliant on external devices. Despite the compact size of the ring-mouse, EYEditor only achieved a slightly higher score than TouchEditor, as not everyone is accustomed to wearing a ring. On the other hand, many daily activities involve finger movements, and wearing such a ring-mouse could lead to frequent donning and doffing. Even though TouchEditor has a larger size, it seamlessly integrates onto clothing in a natural manner.

Task Load. Fig.12c shows the scores of the six dimensions of the NASA TLX questionnaire and their weighted sum, that is, the task load score. Overall, TouchEditor has a lower task load (8.85) than that of EYEditor (10.33). The touch bar is the most difficult to use, with a task load of 13.14. Among them, the high task load associated with the touch bar primarily originates from two aspects: i) Constrained by the narrow interaction area, employing simplistic one-dimensional gestures to accomplish intricate editing tasks proves excessively challenging; ii)

Prolonged focus on eye tracking demands heightened participant fatigue and physical exertion. In the six dimensions, TouchEditor outperforms EYEditor in five of them, except for physical demands. This is mainly due to the sensitive response and tactile feedback of physical buttons. As for other metrics, TouchEditor's highly intuitive and strongly explanatory tactile gesture design facilitates quick familiarization and proficient usage in editing tasks. Conversely, EYEditor's limited interaction controls for complex editing tasks demand a greater memory load from users. Moreover, EYEditor's dependence on speech recognition makes its performance plummet in speech-unfriendly environments; While TouchEditor supports a multi-modal text retyping method, which is more robust to environmental noise and more flexible to different contents. Since text editing with the touch bar performs worse than TouchEditor and EYEditor in correction speed, satisfaction and task load, we recommend considering it as an alternative option only when other devices are unavailable.

6.2 Performance Comparison of Specific Functionalities

In this section, we conducted additional comparative experiments focusing on evaluating two specific functionalities of TouchEditor: text editing functionality and handwritten input functionality, without considering the influence of the environment. Notably, the mentioned text editing functionality encompasses cursor operations and editing commands, excluding text retyping.

6.2.1 Participants and Apparatus. We invited 12 participants (6M, 4F, mean age = 23.17 years, SD = 1.62) to take part in this experiment under the same recruitment conditions as outlined in Sec. 6.1. We compared TouchEditor with two devices based on physical keys, respectively: EYEditor for the text editing functionality and a radial layout keyboard using the hand-held controller matched with Rokid Air Glass (namely Air Station shown in Fig.11c) for the text input functionality. Referring to the "T9 Layout" proposed by Nguyen et al. [62], the different sectors of the radial keyboard group the 26 letters of the alphabet into eight groups of two or three characters each in clockwise order. The Backspace, Space and Shift keys were placed along the radial keyboard's mid-line. Participants navigate through the various sectors on the radial keyboard by using the directional keys on the Air Station, and utilize the central confirmation key to open a subview or confirm the current selection.

6.2.2 Task and Procedure. Details about the above two tasks are as follows.

- Task 1 (Text Editing Task): We give a text with a Flesch reading ease score of 65.89 (standard), and insert six deletion errors into it, which are located at the beginning, middle and end of words and sentences respectively. Participants correct errors by performing cursor operations and deletions.
- Task 2 (Text Input Task): Participants were asked to input the phrase "The quick brown fox jumps over the lazy dog", which contains all English letters within a moderately sized sentence.

While using TouchEditor, participants exclusively employed handwritten input instead of voice input, aiming to compare the performance between handwriting and radial keyboards. Participants had 20 minutes of practice for each device before the experiment. Following the experiment, we engaged in discussions with the participants and gathered their subjective evaluations.

Table 3. The comparison results of TouchEditor with EYEditor and the radial keyboard.

	Correction Speed (CPS)	Text Input Speed (WPM)	Error Rate
EYEditor	0.1161±0.0200	–	–
Radial Keyborad	–	5.35±0.21	0.0302±0.0155
TouchEditor	0.1096±0.0043	6.60±0.22	0.0413±0.0098

6.2.3 Results and Discussion.

Correction Speed of Text Editing Task. Devices based on physical keys often exhibit the fastest speed among various input technologies [92]. Physical keys typically offer tactile feedback, which assists users in confirming key presses and facilitating quicker successive key operations. The comparative experiment results with EYEditor in Task 1 reaffirm this observation; the EYEditor solution achieved a higher correction speed compared to the approach using our TouchEditor (see Table 3). Nonetheless, there was no significant difference in error correction speed between the two devices (F = 1.118, p = 0.302), thereby validating the feasibility of our text editing solution proposed based on a flexible pressure sensor. Furthermore, participants provided more positive feedback regarding our solution. In summary, the feedback can be categorized into the following three points: i) The integration of the text editing system into everyday clothing design is remarkably natural and unobtrusive, evoking a sense of intelligence and futurism. This affirmation underscores its strong potential for development and practical applications. ii) Flexible pressure sensors offer a wide design space for text editing interactions, enabling support for a rich and comprehensive set of functionalities. iii) Our logically robust and user-friendly gesture design allows users to quickly familiarize themselves with the extensive functionalities of the text editing system.

Input Speed and Error Rate of Text Input Task. Compared to traditional QWERTY keyboards, the compact layout of a radial keyboard minimizes the distance traveled while selecting letters. Nevertheless, the comparative results of Task 2 shown in Table 3 indicate that our TouchEditor exhibited significantly higher input speed than the radial keyboard using Air Station (F = 185.341, $p < 0.001$), while there was no significant difference in error rates between the two (F = 4.053, p = 0.056). Despite its compact radial layout, entering a single letter on a radial keyboard requires a minimum of two presses (one to select the sector, another to select the specific letter), and when the desired letter is at a slightly distant position from the current location, additional presses are needed. In contrast, with our approach, participants directly write the letters on the touchpad. Certainly, due to its current capability of only supporting single-letter input, TouchEditor's input speed remains relatively lower compared to specialized text input solutions. This is acceptable in our current text editing system, where handwritten input serves as a corrective measure for voice recognition errors in specific contexts or as a supplementary tool for sensitive information, rather than being the primary input method. However, in our future work, we will actively explore solutions to enhance text input speed.

7 CONCLUSION AND FUTURE WORK

We presented TouchEditor, a novel text editing system for head-mounted displays based on a flexible piezoresistive film sensor, supporting cursor positioning, text selection, text retyping and editing command entering. First, we proposed an area-and-pressure-based method to achieve cursor movement in different directions and different granularities to implement fast and accurate cursor positioning and text selection. Furthermore, we conducted a heuristic study and designed a pressure-controlled menu and a shortcut gesture set on the pressure sensor to enter text editing commands such as *Copy*, *Paste* and *Delete* efficiently. Finally, we conducted a series of experiments to evaluate the gesture recognition accuracy and text selection speed of our system, and to explore the impact of content complexity and scene noise on the error correction performance. We also conducted comparison experiments to compare the performance of different devices in text editing tasks under speech-unfriendly environments. The comparison results confirmed the strong robustness of our TouchEditor with correction speeds stable at 0.078 corrections per second, and users have positively evaluated its strong environmental adaptability.

In the future, we expect to optimize the hardware system towards a more compact form factor, and explore a method of continuous handwriting with more interaction positions to further improve the flexibility, stability, and handwriting speed of our system:

Miniaturizing the sensor reading module. For rapid prototyping and validation of our concept, we utilized an off-shelf signal collector and wired with a self-made Bluetooth module and power source to construct the sensor reading module, resulting in a relatively larger form factor. Following assessment by hardware professionals, with further optimization of circuit layout, there is potential to reduce the signal collector's size by half in future efforts and incorporate Bluetooth transmission capabilities into the same circuit as well.

Improving the input speed of handwriting. At present, users are only allowed to handwrite words on the touchpad letter by letter, which results in a relatively slow text retyping speed. Even though it is enough for the error correction task that requires only a small amount of text retyping, we hope our system can reach faster text editing speed and perform well in tasks requiring more handwritten text in the future. We hope to achieve this by designing a continuous handwriting method.

Exploring more interaction positions. Currently, we have only explored integrating the touchpad onto the forearm position of clothing. There are more interaction positions worth exploring, such as the upper arm, the thigh and so on. In further exploration, we aspire to integrate the touchpad into various segments of clothing, affording users the freedom to choose their preferred interaction positions.

Enhancing the robustness of signal recognition. In this work, we positioned the flexible touchpad on the upper forearm to obtain increased interaction space and reduce sensor bending, enabling simple threshold-based noise reduction to yield satisfactory results. However, the potential noise introduced by excessive sensor bending when exploring additional interaction positions and applications in the future remains a concern. Exploring suitable denoising algorithms and sensor materials with enhanced flexibility (*e.g.*, fabric-based pressure sensors) are both feasible directions for further research.

ACKNOWLEDGMENTS

This work is supported by National Natural Science Foundation of China (62072383, 62077039), the Fundamental Research Funds for the Central Universities (20720210044), and Royal Society (IEC\NSFC\211022).

REFERENCES

[1] 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 (2021), 2648–2658. https://doi.org/10.1109/TVCG.2021.3067776

[2] Jiban Adhikary and Keith Vertanen. 2021. Typing on Midair Virtual Keyboards: Exploring Visual Designs and Interaction Styles. In *Human-Computer Interaction – INTERACT 2021*, Carmelo Ardito, Rosa Lanzilotti, Alessio Malizia, Helen Petrie, Antonio Piccinno, Giuseppe Desolda, and Kori Inkpen (Eds.). Springer International Publishing, Cham, 132–151.

[3] Deepti Aggarwal, Weiyi Zhang, Thuong Hoang, Bernd Ploderer, Frank Vetere, and Mark Bradford. 2017. SoPhy: A Wearable Technology for Lower Limb Assessment in Video Consultations of Physiotherapy. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems* (Denver, Colorado, USA) *(CHI '17)*. Association for Computing Machinery, New York, NY, USA, 3916–3928. https://doi.org/10.1145/3025453.3025489

[4] Sunggeun Ahn and Geehyuk Lee. 2019. Gaze-Assisted Typing for Smart Glasses. In *Proceedings of the 32nd Annual ACM Symposium on User Interface Software and Technology* (New Orleans, LA, USA) *(UIST '19)*. Association for Computing Machinery, New York, NY, USA, 857–869. https://doi.org/10.1145/3332165.3347883

[5] Gilles Bailly, Eric Lecolinet, and Laurence Nigay. 2016. Visual Menu Techniques. *ACM Comput. Surv.* 49, 4, Article 60 (dec 2016), 41 pages. https://doi.org/10.1145/3002171

[6] Bartosz Bajer, I. MacKenzie, and Melanie Baljko. 2012. Huffman base-4 text entry glove (H4 TEG). *Proceedings - International Symposium on Wearable Computers, ISWC*, 41–47. https://doi.org/10.1109/ISWC.2012.28

[7] Ilyasse Belkacem, Isabelle Pecci, Benoît Martin, and Anthony Faiola. 2019. TEXTile: Eyes-Free Text Input on Smart Glasses Using Touch Enabled Textile on the Forearm. In *Human-Computer Interaction – INTERACT 2019*, David Lamas, Fernando Loizides, Lennart Nacke, Helen Petrie, Marco Winckler, and Panayiotis Zaphiris (Eds.). Springer International Publishing, Cham, 351–371.

[8] Tobias Boceck, Sascha Sprott, Huy Viet Le, and Sven Mayer. 2019. Force Touch Detection on Capacitive Sensors Using Deep Neural Networks. In *Proceedings of the 21st International Conference on Human-Computer Interaction with Mobile Devices and Services* (Taipei,

Taiwan) *(MobileHCI '19)*. Association for Computing Machinery, New York, NY, USA, Article 42, 6 pages. https://doi.org/10.1145/3338286.3344389

[9] Costas Boletsis and Stian Kongsvik. 2019. Text Input in Virtual Reality: A Preliminary Evaluation of the Drum-Like VR Keyboard. *Technologies* 7, 2 (2019). https://doi.org/10.3390/technologies7020031

[10] Sebastian Boring, David Ledo, Xiang 'Anthony' Chen, Nicolai Marquardt, Anthony Tang, and Saul Greenberg. 2012. The Fat Thumb: Using the Thumb's Contact Size for Single-Handed Mobile Interaction *(MobileHCI '12)*. Association for Computing Machinery, New York, NY, USA, 207–208. https://doi.org/10.1145/2371664.2371711

[11] Eugene Cho. 2019. Hey Google, Can I Ask You Something in Private?. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems* (Glasgow, Scotland Uk) *(CHI '19)*. Association for Computing Machinery, New York, NY, USA, 1–9. https://doi.org/10.1145/3290605.3300488

[12] Christian Corsten, Simon Voelker, Andreas Link, and Jan Borchers. 2018. Use the Force Picker, Luke: Space-Efficient Value Input on Force-Sensitive Mobile Touchscreens. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems* (Montreal QC, Canada) *(CHI '18)*. Association for Computing Machinery, New York, NY, USA, 1–12. https://doi.org/10.1145/3173574.3174235

[13] John J. Dudley, Keith Vertanen, and Per Ola Kristensson. 2018. Fast and Precise Touch-Based Text Entry for Head-Mounted Augmented Reality with Variable Occlusion. *ACM Trans. Comput.-Hum. Interact.* 25, 6, Article 30 (dec 2018), 40 pages. https://doi.org/10.1145/3232163

[14] Don Samitha Elvitigala, Denys J.C. Matthies, Löic David, Chamod Weerasinghe, and Suranga Nanayakkara. 2019. GymSoles: Improving Squats and Dead-Lifts by Visualizing the User's Center of Pressure. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems* (Glasgow, Scotland Uk) *(CHI '19)*. Association for Computing Machinery, New York, NY, USA, 1–12. https://doi.org/10.1145/3290605.3300404

[15] Lee Englestone and Lee Englestone. 2021. Touch Gestures and Interaction. *. NET Developer's Guide to Augmented Reality in iOS: Building Immersive Apps Using Xamarin, ARKit, and C#* (2021), 107–115.

[16] Wenjing Fan, Qiang He, Keyu Meng, Xulong Tan, Zhihao Zhou, Gaoqiang Zhang, Jin Yang, and Zhong Lin Wang. 2020. Machine-knitted washable sensor array textile for precise epidermal physiological signal monitoring. *Science Advances* 6, 11 (2020), eaay2840. https://doi.org/10.1126/sciadv.aay2840 arXiv:https://www.science.org/doi/pdf/10.1126/sciadv.aay2840

[17] Fengyi Fang, Hongwei Zhang, Lishuang Zhan, Shihui Guo, Minying Zhang, Juncong Lin, Yipeng Qin, and Hongbo Fu. 2023. Handwriting Velcro: Endowing AR Glasses with Personalized and Posture-Adaptive Text Input Using Flexible Touch Sensor. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.* 6, 4, Article 163 (jan 2023), 31 pages. https://doi.org/10.1145/3569461

[18] Jacqui Fashimpaur, Kenrick Kin, and Matt Longest. 2020. PinchType: Text Entry for Virtual and Augmented Reality Using Comfortable Thumb to Fingertip Pinches. In *Extended Abstracts of the 2020 CHI Conference on Human Factors in Computing Systems* (Honolulu, HI, USA) *(CHI EA '20)*. Association for Computing Machinery, New York, NY, USA, 1–7. https://doi.org/10.1145/3334480.3382888

[19] Dan S. Felsenthal. 1989. On combining approval with disapproval voting. *Behavioral Science* 34, 1 (1989), 53–60. https://doi.org/10.1002/bs.3830340105 arXiv:https://onlinelibrary.wiley.com/doi/pdf/10.1002/bs.3830340105

[20] Conor R. Foy, John J. Dudley, Aakar Gupta, Hrvoje Benko, and Per Ola Kristensson. 2021. Understanding, Detecting and Mitigating the Effects of Coactivations in Ten-Finger Mid-Air Typing in Virtual Reality. In *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems* (Yokohama, Japan) *(CHI '21)*. Association for Computing Machinery, New York, NY, USA, Article 287, 11 pages. https://doi.org/10.1145/3411764.3445671

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

[22] Masaaki Fukumoto. 2018. SilentVoice: Unnoticeable Voice Input by Ingressive Speech. In *Proceedings of the 31st Annual ACM Symposium on User Interface Software and Technology* (Berlin, Germany) *(UIST '18)*. Association for Computing Machinery, New York, NY, USA, 237–246. https://doi.org/10.1145/3242587.3242603

[23] Yuji Gao, Hiroki Ota, Ethan W. Schaler, Kevin Chen, Allan Zhao, Wei Gao, Hossain M. Fahad, Yonggang Leng, Anzong Zheng, Furui Xiong, Chuchu Zhang, Li-Chia Tai, Peida Zhao, Ronald S. Fearing, and Ali Javey. 2017. Wearable Microfluidic Diaphragm Pressure Sensor for Health and Tactile Touch Monitoring. *Advanced Materials* 29, 39 (2017), 1701985. https://doi.org/10.1002/adma.201701985 arXiv:https://onlinelibrary.wiley.com/doi/pdf/10.1002/adma.201701985

[24] 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*. Association for Computing Machinery, New York, NY, USA, 1–13. https://doi.org/10.1145/3313831.3376173

[25] Alix Goguey, Sylvain Malacria, and Carl Gutwin. 2018. Improving Discoverability and Expert Performance in Force-Sensitive Text Selection for Touch Devices with Mode Gauges. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems* (Montreal QC, Canada) *(CHI '18)*. Association for Computing Machinery, New York, NY, USA, 1–12. https://doi.org/10.1145/3173574.3174051

[26] Tovi Grossman, Xiang Anthony Chen, and George Fitzmaurice. 2015. Typing on Glasses: Adapting Text Entry to Smart Eyewear. In *Proceedings of the 17th International Conference on Human-Computer Interaction with Mobile Devices and Services* (Copenhagen, Denmark)

(MobileHCI '15). Association for Computing Machinery, New York, NY, USA, 144–152. https://doi.org/10.1145/2785830.2785867

[27] Shihui Guo, Lishuang Zhan, Yancheng Cao, Chen Zheng, Guyue Zhou, and Jiangtao Gong. 2023. Touch-and-Heal: Data-driven Affective Computing in Tactile Interaction with Robotic Dog. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 7, 2 (2023), 1–33.

[28] Aakar Gupta, Muhammed Anwar, and Ravin Balakrishnan. 2016. Porous Interfaces for Small Screen Multitasking Using Finger Identification. In *Proceedings of the 29th Annual Symposium on User Interface Software and Technology* (Tokyo, Japan) *(UIST '16)*. Association for Computing Machinery, New York, NY, USA, 145–156. https://doi.org/10.1145/2984511.2984557

[29] Aakar Gupta and Ravin Balakrishnan. 2016. DualKey: Miniature Screen Text Entry via Finger Identification. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems* (San Jose, California, USA) *(CHI '16)*. Association for Computing Machinery, New York, NY, USA, 59–70. https://doi.org/10.1145/2858036.2858052

[30] Aakar Gupta, Cheng Ji, Hui-Shyong Yeo, Aaron Quigley, and Daniel Vogel. 2019. RotoSwype: Word-Gesture Typing Using a Ring. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems* (Glasgow, Scotland Uk) *(CHI '19)*. Association for Computing Machinery, New York, NY, USA, 1–12. https://doi.org/10.1145/3290605.3300244

[31] Sandra G. Hart and Lowell E. Staveland. 1988. Development of NASA-TLX (Task Load Index): Results of Empirical and Theoretical Research. In *Human Mental Workload*, Peter A. Hancock and Najmedin Meshkati (Eds.). Advances in Psychology, Vol. 52. North-Holland, 139–183. https://doi.org/10.1016/S0166-4115(08)62386-9

[32] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 770–778.

[33] Lifeng He, Xiwei Ren, Qihang Gao, Xiao Zhao, Bin Yao, and Yuyan Chao. 2017. The connected-component labeling problem: A review of state-of-the-art algorithms. *Pattern Recognition* 70 (2017), 25–43. https://doi.org/10.1016/j.patcog.2017.04.018

[34] Ramin Hedeshy, Chandan Kumar, Raphael Menges, and Steffen Staab. 2021. Hummer: Text Entry by Gaze and Hum. In *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems* (Yokohama, Japan) *(CHI '21)*. Association for Computing Machinery, New York, NY, USA, Article 741, 11 pages. https://doi.org/10.1145/3411764.3445501

[35] Julia Hertel and Frank Steinicke. 2021. Augmented Reality for Maritime Navigation Assistance - Egocentric Depth Perception in Large Distance Outdoor Environments. In *2021 IEEE Virtual Reality and 3D User Interfaces (VR)*. 122–130. https://doi.org/10.1109/VR50410.2021.00033

[36] Juan David Hincapié-Ramos, Xiang Guo, Paymahn Moghadasian, and Pourang Irani. 2014. Consumed Endurance: A Metric to Quantify Arm Fatigue of Mid-Air Interactions. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (Toronto, Ontario, Canada) *(CHI '14)*. Association for Computing Machinery, New York, NY, USA, 1063–1072. https://doi.org/10.1145/2556288.2557130

[37] Lode Hoste, Bruno Dumas, and Beat Signer. 2012. SpeeG: A Multimodal Speech- and Gesture-based Text Input Solution. *Proceedings of the Workshop on Advanced Visual Interfaces AVI*, 156–163. https://doi.org/10.1145/2254556.2254585

[38] Lode Hoste and Beat Signer. 2013. SpeeG2: A Speech- and Gesture-Based Interface for Efficient Controller-Free Text Input. In *Proceedings of the 15th ACM on International Conference on Multimodal Interaction* (Sydney, Australia) *(ICMI '13)*. Association for Computing Machinery, New York, NY, USA, 213–220. https://doi.org/10.1145/2522848.2522861

[39] Yi-Ta Hsieh, Antti Jylhä, Valeria Orso, Luciano Gamberini, and Giulio Jacucci. 2016. Designing a Willing-to-Use-in-Public Hand Gestural Interaction Technique for Smart Glasses. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems* (San Jose, California, USA) *(CHI '16)*. Association for Computing Machinery, New York, NY, USA, 4203–4215. https://doi.org/10.1145/2858036.2858436

[40] Haiyan Jiang and Dongdong Weng. 2020. HiPad: Text entry for Head-Mounted Displays Using Circular Touchpad. In *2020 IEEE Conference on Virtual Reality and 3D User Interfaces (VR)*. 692–703. https://doi.org/10.1109/VR46266.2020.00092

[41] Annkatrin Jung, Miquel Alfaras, Pavel Karpashevich, William Primett, and Kristina Höök. 2021. Exploring Awareness of Breathing through Deep Touch Pressure. In *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems* (Yokohama, Japan) *(CHI '21)*. Association for Computing Machinery, New York, NY, USA, Article 263, 15 pages. https://doi.org/10.1145/3411764.3445533

[42] Pawel Kasprowski. 2022. Eye Tracking Hardware: Past to Present, and Beyond. In *Eye Tracking: Background, Methods, and Applications*. Springer, 31–48.

[43] Farnaz Khoshmanesh, Peter Thurgood, Elena Pirogova, Saeid Nahavandi, and Sara Baratchi. 2021. Wearable sensors: At the frontier of personalised health monitoring, smart prosthetics and assistive technologies. *Biosensors and Bioelectronics* 176 (2021), 112946. https://doi.org/10.1016/j.bios.2020.112946

[44] Junhyeok Kim, William Delamare, and Pourang Irani. 2018. ThumbText: Text Entry for Wearable Devices Using a Miniature Ring. In *Proceedings of the 44th Graphics Interface Conference* (Toronto, Canada) *(GI '18)*. Canadian Human-Computer Communications Society, Waterloo, CAN, 18–25. https://doi.org/10.20380/GI2018.04

[45] J. Peter Kincaid, Robert P. Fishburne, R L Rogers, and Brad S. Chissom. 1975. Derivation of New Readability Formulas (Automated Readability Index, Fog Count and Flesch Reading Ease Formula) for Navy Enlisted Personnel.

[46] Pascal Knierim, Valentin Schwind, Anna Maria Feit, Florian Nieuwenhuizen, and Niels Henze. 2018. Physical Keyboards in Virtual Reality: Analysis of Typing Performance and Effects of Avatar Hands. In *Proceedings of the 2018 CHI Conference on Human Factors*

in Computing Systems (Montreal QC, Canada) *(CHI '18)*. Association for Computing Machinery, New York, NY, USA, 1–9. https://doi.org/10.1145/3173574.3173919

[47] Lingyi Lan, Fengnian Zhao, Yao Yao, Jianfeng Ping, and Yibin Ying. 2020. One-Step and Spontaneous in Situ Growth of Popcorn-like Nanostructures on Stretchable Double-Twisted Fiber for Ultrasensitive Textile Pressure Sensor. *ACS Applied Materials & Interfaces* 12, 9 (2020), 10689–10696. https://doi.org/10.1021/acsami.0c00079 arXiv:https://doi.org/10.1021/acsami.0c00079 PMID: 32028766.

[48] Andreas Leber, Alexis Gérald Page, Dong Yan, Yunpeng Qu, Shahrzad Shadman, Pedro Reis, and Fabien Sorin. 2020. Compressible and Electrically Conducting Fibers for Large-Area Sensing of Pressures. *Advanced Functional Materials* 30, 1 (2020), 1904274. https://doi.org/10.1002/adfm.201904274 arXiv:https://onlinelibrary.wiley.com/doi/pdf/10.1002/adfm.201904274

[49] DoYoung Lee, Jiwan Kim, and Ian Oakley. 2021. FingerText: Exploring and Optimizing Performance for Wearable, Mobile and One-Handed Typing. In *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems* (Yokohama, Japan) *(CHI '21)*. Association for Computing Machinery, New York, NY, USA, Article 283, 15 pages. https://doi.org/10.1145/3411764.3445106

[50] Lik Hang Lee, Kit Yung Lam, Tong Li, Tristan Braud, Xiang Su, and Pan Hui. 2019. Quadmetric Optimized Thumb-to-Finger Interaction for Force Assisted One-Handed Text Entry on Mobile Headsets. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.* 3, 3, Article 94 (sep 2019), 27 pages. https://doi.org/10.1145/3351252

[51] Lik Hang Lee, Kit Yung Lam, Yui Pan Yau, Tristan Braud, and Pan Hui. 2019. HIBEY: Hide the Keyboard in Augmented Reality. In *2019 IEEE International Conference on Pervasive Computing and Communications (PerCom*. 1–10. https://doi.org/10.1109/PERCOM.2019.8767420

[52] Joanne Leong, Patrick Parzer, Florian Perteneder, Teo Babic, Christian Rendl, Anita Vogl, Hubert Egger, Alex Olwal, and Michael Haller. 2016. ProCover: Sensory Augmentation of Prosthetic Limbs Using Smart Textile Covers. In *Proceedings of the 29th Annual Symposium on User Interface Software and Technology* (Tokyo, Japan) *(UIST '16)*. Association for Computing Machinery, New York, NY, USA, 335–346. https://doi.org/10.1145/2984511.2984572

[53] Han Li, Jiqiang Cao, Junli Chen, Xiao Liu, Yawen Shao, and Zhaoqun Du. 2022. Highly Sensitive MXene Helical Yarn/Fabric Tactile Sensors Enabling Full Scale Movement Detection of Human Motions. *Advanced Electronic Materials* 8, 4 (2022), 2100890. https://doi.org/10.1002/aelm.202100890 arXiv:https://onlinelibrary.wiley.com/doi/pdf/10.1002/aelm.202100890

[54] Yudong Li, Yanan Li, Meng Su, Wenbo Li, Yifan Li, Huizeng Li, Xin Qian, Xingye Zhang, Fengyu Li, and Yanlin Song. 2017. Electronic Textile by Dyeing Method for Multiresolution Physical Kineses Monitoring. *Advanced Electronic Materials* 3, 10 (2017), 1700253. https://doi.org/10.1002/aelm.201700253 arXiv:https://onlinelibrary.wiley.com/doi/pdf/10.1002/aelm.201700253

[55] Xueshi Lu, Difeng Yu, Hai-Ning Liang, and Jorge Goncalves. 2021. IText: Hands-Free Text Entry on an Imaginary Keyboard for Augmented Reality Systems. In *The 34th Annual ACM Symposium on User Interface Software and Technology* (Virtual Event, USA) *(UIST '21)*. Association for Computing Machinery, New York, NY, USA, 815–825. https://doi.org/10.1145/3472749.3474788

[56] Yi Lu, Xiaoye Wang, Jiangtao Gong, Yun Liang, and Kalidoss Rajakani. 2022. ChordAR: An Educational AR Game Design for Children's Music Theory Learning. *Wirel. Commun. Mob. Comput.* 2022 (jan 2022), 9 pages. https://doi.org/10.1155/2022/5268586

[57] Yiyue Luo, Yunzhu Li, Michael Foshey, Wan Shou, Pratyusha Sharma, Tomás Palacios, Antonio Torralba, and Wojciech Matusik. 2021. Intelligent carpet: Inferring 3d human pose from tactile signals. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 11255–11265.

[58] I. Scott MacKenzie and R. William Soukoreff. 2003. Phrase Sets for Evaluating Text Entry Techniques. In *CHI '03 Extended Abstracts on Human Factors in Computing Systems* (Ft. Lauderdale, Florida, USA) *(CHI EA '03)*. Association for Computing Machinery, New York, NY, USA, 754–755. https://doi.org/10.1145/765891.765971

[59] Denys J. C. Matthies, Thijs Roumen, Arjan Kuijper, and Bodo Urban. 2017. CapSoles: Who is Walking on What Kind of Floor?. In *Proceedings of the 19th International Conference on Human-Computer Interaction with Mobile Devices and Services* (Vienna, Austria) *(MobileHCI '17)*. Association for Computing Machinery, New York, NY, USA, Article 9, 14 pages. https://doi.org/10.1145/3098279.3098545

[60] Manuel Meier, Paul Streli, Andreas Fender, and Christian Holz. 2021. TapID: Rapid Touch Interaction in Virtual Reality using Wearable Sensing. In *2021 IEEE Virtual Reality and 3D User Interfaces (VR)*. 519–528. https://doi.org/10.1109/VR50410.2021.00076

[61] Carolina Milanesi. 2016. Voice Assistant Anyone? Yes please, but not in public. In *Creative Strategies*.

[62] Anh Nguyen, Samuel Bittman, and Markus Zank. 2020. Text Input Methods in Virtual Reality Using Radial Layouts. In *Proceedings of the 26th ACM Symposium on Virtual Reality Software and Technology*. 1–3.

[63] Shahriar Nirjon, Jeremy Gummeson, Dan Gelb, and Kyu-Han Kim. 2015. TypingRing: A Wearable Ring Platform for Text Input. In *Proceedings of the 13th Annual International Conference on Mobile Systems, Applications, and Services* (Florence, Italy) *(MobiSys '15)*. Association for Computing Machinery, New York, NY, USA, 227–239. https://doi.org/10.1145/2742647.2742665

[64] Kent L Norman, Ben Shneiderman, B Harper, and L Slaughter. 1998. Questionnaire for user interaction satisfaction. *University of Maryland (Norman, 1989) Disponível em* (1998).

[65] Ian Oakley, Carina Lindahl, Khanh Le, DoYoung Lee, and MD. Rasel Islam. 2016. The Flat Finger: Exploring Area Touches on Smartwatches. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems* (San Jose, California, USA) *(CHI '16)*. Association for Computing Machinery, New York, NY, USA, 4238–4249. https://doi.org/10.1145/2858036.2858179

[66] Alex Olwal, Jon Moeller, Greg Priest-Dorman, Thad Starner, and Ben Carroll. 2018. I/O Braid: Scalable Touch-Sensitive Lighted Cords Using Spiraling, Repeating Sensing Textiles and Fiber Optics. In *Proceedings of the 31st Annual ACM Symposium on User*

Interface Software and Technology (Berlin, Germany) *(UIST '18)*. Association for Computing Machinery, New York, NY, USA, 485–497. https://doi.org/10.1145/3242587.3242638

[67] Alex Olwal, Thad Starner, and Gowa Mainini. 2020. E-Textile Microinteractions: Augmenting Twist with Flick, Slide and Grasp Gestures for Soft Electronics. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems*. Association for Computing Machinery, New York, NY, USA, 1–13. https://doi.org/10.1145/3313831.3376236

[68] Alexander Otte, Daniel Schneider, Tim Menzner, Travis Gesslein, Philipp Gagel, and Jens Grubert. 2019. Evaluating Text Entry in Virtual Reality using a Touch-sensitive Physical Keyboard. In *2019 IEEE International Symposium on Mixed and Augmented Reality Adjunct (ISMAR-Adjunct)*. 387–392. https://doi.org/10.1109/ISMAR-Adjunct.2019.000-4

[69] Sharon Oviatt. 2000. Taming Recognition Errors with a Multimodal Interface. *Commun. ACM* 43, 9 (sep 2000), 45–51. https://doi.org/10.1145/348941.348979

[70] Patrick Parzer, Adwait Sharma, Anita Vogl, Jürgen Steimle, Alex Olwal, and Michael Haller. 2017. SmartSleeve: Real-Time Sensing of Surface and Deformation Gestures on Flexible, Interactive Textiles, Using a Hybrid Gesture Detection Pipeline. In *Proceedings of the 30th Annual ACM Symposium on User Interface Software and Technology* (Québec City, QC, Canada) *(UIST '17)*. Association for Computing Machinery, New York, NY, USA, 565–577. https://doi.org/10.1145/3126594.3126652

[71] Alexander Plopski, Teresa Hirzle, Nahal Norouzi, Long Qian, Gerd Bruder, and Tobias Langlotz. 2022. The eye in extended reality: A survey on gaze interaction and eye tracking in head-worn extended reality. *ACM Computing Surveys (CSUR)* 55, 3 (2022), 1–39.

[72] Narjes Pourjafarian, Anusha Withana, Joseph A. Paradiso, and Jürgen Steimle. 2019. Multi-Touch Kit: A Do-It-Yourself Technique for Capacitive Multi-Touch Sensing Using a Commodity Microcontroller. In *Proceedings of the 32nd Annual ACM Symposium on User Interface Software and Technology* (New Orleans, LA, USA) *(UIST '19)*. Association for Computing Machinery, New York, NY, USA, 1071–1083. https://doi.org/10.1145/3332165.3347895

[73] Felix Putze, Tilman Ihrig, Tanja Schultz, and Wolfgang Stuerzlinger. 2020. Platform for Studying Self-Repairing Auto-Corrections in Mobile Text Entry Based on Brain Activity, Gaze, and Context. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems*. Association for Computing Machinery, New York, NY, USA, 1–13. https://doi.org/10.1145/3313831.3376815

[74] Philip Quinn, Wenxin Feng, and Shumin Zhai. 2021. Deep Touch: Sensing Press Gestures from Touch Image Sequences. *Artificial Intelligence for Human Computer Interaction: A Modern Approach* (2021), 169–192.

[75] Danny Schott, Patrick Saalfeld, Gerd Schmidt, Fabian Joeres, Christian Boedecker, Florentine Huettl, Hauke Lang, Tobias Huber, Bernhard Preim, and Christian Hansen. 2021. A VR/AR Environment for Multi-User Liver Anatomy Education. In *2021 IEEE Virtual Reality and 3D User Interfaces (VR)*. 296–305. https://doi.org/10.1109/VR50410.2021.00052

[76] Xinyu Shi, Junjun Pan, Zeyong Hu, Juncong Lin, Shihui Guo, Minghong Liao, Ye Pan, and Ligang Liu. 2019. Accurate and Fast Classification of Foot Gestures for Virtual Locomotion. In *2019 IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*. 178–189. https://doi.org/10.1109/ISMAR.2019.000-6

[77] Paul Strohmeier, Jarrod Knibbe, Sebastian Boring, and Kasper Hornbæk. 2018. ZPatch: Hybrid Resistive/Capacitive ETextile Input. In *Proceedings of the Twelfth International Conference on Tangible, Embedded, and Embodied Interaction* (Stockholm, Sweden) *(TEI '18)*. Association for Computing Machinery, New York, NY, USA, 188–198. https://doi.org/10.1145/3173225.3173242

[78] Zixiong Su, Shitao Fang, and Jun Rekimoto. 2023. LipLearner: Customizable Silent Speech Interactions on Mobile Devices. In *Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems* (Hamburg, Germany) *(CHI '23)*. Association for Computing Machinery, New York, NY, USA, Article 696, 21 pages. https://doi.org/10.1145/3544548.3581465

[79] Bernhard Suhm, Brad Myers, and Alex Waibel. 2001. Multimodal Error Correction for Speech User Interfaces. *ACM Trans. Comput.-Hum. Interact.* 8, 1 (mar 2001), 60–98. https://doi.org/10.1145/371127.371166

[80] Bernhard Suhm, Brad Myers, and Alex Waibel. 2001. Multimodal Error Correction for Speech User Interfaces. *ACM Trans. Comput.-Hum. Interact.* 8, 1 (mar 2001), 60–98. https://doi.org/10.1145/371127.371166

[81] Subramanian Sundaram, Petr Kellnhofer, Yunzhu Li, Jun-Yan Zhu, Antonio Torralba, and Wojciech Matusik. 2019. Learning the signatures of the human grasp using a scalable tactile glove. *Nature* 569 (2019), 698–702. https://doi.org/10.1038/s41586-019-1234-z

[82] Huawei Tu, Boyu Gao, Huiyue Wu, and Fei Lyu. 2023. Text Pin: Improving text selection with mode-augmented handles on touchscreen mobile devices. *International Journal of Human-Computer Studies* 175 (2023), 103028. https://doi.org/10.1016/j.ijhcs.2023.103028

[83] Ruben Geert van den Berg. 2022. SPSS ANOVA with Post Hoc Tests. Website. https://www.spss-tutorials.com/spss-one-way-anova-with-post-hoc-tests-example.

[84] Ana Villanueva, Zhengzhe Zhu, Ziyi Liu, Kylie Peppler, Thomas Redick, and Karthik Ramani. 2020. Meta-AR-App: An Authoring Platform for Collaborative Augmented Reality in STEM Classrooms. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems*. Association for Computing Machinery, New York, NY, USA, 1–14. https://doi.org/10.1145/3313831.3376146

[85] Jonas Vogelsang, Francisco Kiss, and Sven Mayer. 2021. A Design Space for User Interface Elements Using Finger Orientation Input. In *Proceedings of Mensch Und Computer 2021* (Ingolstadt, Germany) *(MuC '21)*. Association for Computing Machinery, New York, NY, USA, 1–10. https://doi.org/10.1145/3473856.3473862

[86] James Walker, Bochao Li, Keith Vertanen, and Scott Kuhl. 2017. Efficient Typing on a Visually Occluded Physical Keyboard. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems* (Denver, Colorado, USA) *(CHI '17)*. Association for Computing

Machinery, New York, NY, USA, 5457–5461. https://doi.org/10.1145/3025453.3025783

[87] David J. Ward, Alan F. Blackwell, and David J. C. MacKay. 2000. Dasher—a Data Entry Interface Using Continuous Gestures and Language Models. In *Proceedings of the 13th Annual ACM Symposium on User Interface Software and Technology* (San Diego, California, USA) *(UIST '00)*. Association for Computing Machinery, New York, NY, USA, 129–137. https://doi.org/10.1145/354401.354427

[88] Eric Whitmire, Mohit Jain, Divye Jain, Greg Nelson, Ravi Karkar, Shwetak Patel, and Mayank Goel. 2017. DigiTouch: Reconfigurable Thumb-to-Finger Input and Text Entry on Head-Mounted Displays. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.* 1, 3, Article 113 (sep 2017), 21 pages. https://doi.org/10.1145/3130978

[89] Chien-Min Wu, Chih-Wen Hsu, Tzu-Kuei Lee, and Shana Smith. 2017. A Virtual Reality Keyboard with Realistic Haptic Feedback in a Fully Immersive Virtual Environment. *Virtual Real.* 21, 1 (mar 2017), 19–29. https://doi.org/10.1007/s10055-016-0296-6

[90] Te-Yen Wu, Zheer Xu, Xing-Dong Yang, Steve Hodges, and Teddy Seyed. 2021. Project Tasca: Enabling Touch and Contextual Interactions with a Pocket-Based Textile Sensor. In *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems* (Yokohama, Japan) *(CHI '21)*. Association for Computing Machinery, New York, NY, USA, Article 4, 13 pages. https://doi.org/10.1145/3411764.3445712

[91] Hou Xing-Yu and Guo Chuan-Fei. 2020. Sensing mechanisms and applications of flexible pressure sensors. *Acta Phys. Sin.* 69, 17 (2020). https://doi.org/10.7498/aps.69.20200987

[92] Wenge Xu, Hai-Ning Liang, Anqi He, and Zifan Wang. 2019. Pointing and Selection Methods for Text Entry in Augmented Reality Head Mounted Displays. In *2019 IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*. 279–288. https://doi.org/10.1109/ISMAR.2019.00026

[93] Zheer Xu, Pui Chung Wong, Jun Gong, Te-Yen Wu, Aditya Shekhar Nittala, Xiaojun Bi, Jürgen Steimle, Hongbo Fu, Kening Zhu, and Xing-Dong Yang. 2019. TipText: Eyes-Free Text Entry on a Fingertip Keyboard. In *Proceedings of the 32nd Annual ACM Symposium on User Interface Software and Technology* (New Orleans, LA, USA) *(UIST '19)*. Association for Computing Machinery, New York, NY, USA, 883–899. https://doi.org/10.1145/3332165.3347865

[94] Yiya Yang. 1988. Undo support models. *International Journal of Man-Machine Studies* 28, 5 (1988), 457–481. https://doi.org/10.1016/S0020-7373(88)80056-7

[95] Chun Yu, Ke Sun, Mingyuan Zhong, Xincheng Li, Peijun Zhao, and Yuanchun Shi. 2016. One-Dimensional Handwriting: Inputting Letters and Words on Smart Glasses. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems*. Association for Computing Machinery, New York, NY, USA, 71–82. https://doi.org/10.1145/2858036.2858542

[96] Difeng Yu, Kaixuan Fan, Heng Zhang, Diego Monteiro, Wenge Xu, and Hai-Ning Liang. 2018. PizzaText: Text Entry for Virtual Reality Systems Using Dual Thumbsticks. *IEEE Transactions on Visualization and Computer Graphics* 24, 11 (2018), 2927–2935. https://doi.org/10.1109/TVCG.2018.2868581

[97] Yingwei Zhang, Yiqiang Chen, Hanchao Yu, Xiaodong Yang, Ruizhe Sun, and Bixiao Zeng. 2021. A feature adaptive learning method for high-density semg-based gesture recognition. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 5, 1 (2021), 1–26.

[98] Maozheng Zhao, Alec M Pierce, Ran Tan, Ting Zhang, Tianyi Wang, Tanya R Jonker, Hrvoje Benko, and Aakar Gupta. 2023. Gaze Speedup: Eye Gaze Assisted Gesture Typing in Virtual Reality. In *Proceedings of the 28th International Conference on Intelligent User Interfaces*. 595–606.

[99] Bo Zhou, Sungho Suh, Vitor Fortes Rey, Carlos Andres Velez Altamirano, and Paul Lukowicz. 2022. Quali-mat: Evaluating the quality of execution in body-weight exercises with a pressure sensitive sports mat. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 6, 2 (2022), 1–45.