Inclusive Text Entry Techniques for Mobile Devices

by

Gulnar Rakhmetulla

A dissertation submitted in partial satisfaction of the

requirements for the degree of

Doctor of Philosophy

in

Electrical Engineering and Computer Science

in the

Graduate Division

of the

University of California, Merced

Committee in charge:

Assistant Professor Ahmed Sabbir Arif, Chair
Associate Professor David Noelle
Assistant Professor Shijia Pan
Professor Marcello Kallmann

Summer 2022

Inclusive Text Entry Techniques for Mobile Devices

Inclusive Interaction Lab
https://www.theiilab.com

VITA

2014	Bachelor of Science in Information Systems *with distinction*, International Information Technology University, Almaty, Kazakhstan
2016	Master of Automation Control and Systems Engineering *with distinction*, The University of Sheffield, United Kingdom
2022	Ph. D. in Computer Science and Engineering, University of California, Merced

PUBLICATIONS

1. SwipeRing: Gesture Typing on Smartwatches Using a Segmented Qwerty Around the Bezel. In *Proc. of the 47th Graphics Interface Conference* (GI '21). Canadian Human-Computer Communications Society (CHCCS), Toronto, Ontario, Canada, 166–177. *Best paper award*. Gulnar Rakhmetulla and Ahmed Sabbir Arif.

2. Using Action-Level Metrics to Report the Performance of Multi-Step Keyboards. In *Proc. of the 47th Graphics Interface Conference* (GI '21). Canadian Human-Computer Communications Society (CHCCS), Toronto, Canada, 127–137. Gulnar Rakhmetulla, Ahmed Sabbir Arif, Steven Castellucci, I. Scott MacKenzie, Caitlyn Seim.

3. TapStr: A Tap and Stroke Reduced–Qwerty for Smartphones. In *Proc. of the 2020 ACM International Conference on Interactive Surfaces and Spaces* (ISS '20). ACM, New York, 47–50. Mohammad Akbor Sharif, Gulnar Rakhmetulla, Ahmed Sabbir Arif.

4. Senorita: A Chorded Keyboard for Sighted, Low Vision, and Blind Mobile Users. In *Proc. of the 2020 CHI Conference on Human Factors in Computing Systems* (CHI '20). ACM, New York, 1–13. Gulnar Rakhmetulla, Ahmed Sabbir Arif.

5. Put a Ring on It: Text Entry Performance on a Grip Ring Attached Smartphone. In MobileHCI 2018 *Workshop on Socio-Technical Aspects of Text Entry* ('18). Barcelona, Spain, 6–10. Monwen Shen, Gulnar Rakhmetulla, Ahmed Sabbir Arif.

6. Enabling Input on Tiny/Headless Systems Using Morse Code. Poster at Center for Cellular and Biomolecular Machines Open House ('18). University of California, Merced. Anna-Maria Gueorguieva, Gulnar Rakhmetulla, Ahmed Sabbir Arif.

Abstract

Inclusive Text Entry Techniques for Mobile Devices

by

Gulnar Rakhmetulla

Doctor of Philosophy in Electrical Engineering and Computer Science

University of California, Merced

Assistant Professor Ahmed Sabbir Arif, Chair

Text entry is one of the most commonly performed tasks on mobile devices, however, it still poses accessibility challenges. This dissertation investigates existing text entry techniques for tablet computers, smartphones, and smartwatches to identify accessibility issues with them, then designs, develops, and evaluates more inclusive text entry techniques to mitigate these problems. The first part of the dissertation presents Senorita — a novel two-thumb virtual chorded keyboard for sighted, low vision, and blind mobile users. The technique is design following the concept of inclusive design, using mainstream technology without any extramural devices. Results show that with Senorita blind users surpass their QWERTY entry speed by 32%. Besides, sighted users yield a comparative performance both on a smartphone and tablet. The second part of the dissertation explores a crown-based keyboard — a technique to enable text entry with one finger using the crown of a smartwatch for people with limited dexterity. It uses an iterative design process with multiple user studies. Results show that manual rotation is faster (4 vs. 6 wpm) and more accurate than automated for able-bodied people. A final study with ten people with limited dexterity reveal that both automated (2.3 wpm) and manual (2.6 wpm) the crown-based keyboards were faster and more accurate than the default QWERTY method on smartwatches. The third part of the dissertation presents SwipeRing — a novel keyboard that arranges the QWERTY layout around the bezel of a smartwatch divided into zones to enable gesture typing. These zones are optimized for usability and to maintain similarities between the gestures drawn on a mobile virtual QWERTY and a smartwatch to facilitate skill transfer. The comparison of SwipeRing with a similar method shows that SwipeRing yields a 33% faster entry speed (16.67 wpm) and a 56% lower error rate. Finally, the fourth part of the dissertation presents new action-level performance metrics: UnitER, UA, and UnitCX. They account for the error rate, accuracy, and complexity of multi-step chorded and constructive text entry methods. Results of a longitudinal study with two existing multi-step methods confirm that these metrics help to identify probable causes of slower text entry and input errors.

To my family

Contents

List of Figures

List of Tables

Acknowledgments

I would first like to thank my advisor Dr. Ahmed Sabbir Arif, for his valuable guidance, immense knowledge, and support during my Ph.D. journey. Your insightful feedback always made my work substantially better and expanded my knowledge. I would like to express my deepest appreciation to my committee members, Dr. David Noelle, Dr. Marcelo Kallmann, and Dr. Shijia Pan, for their time and valuable feedback. I would like to extend my thanks to Dr. I. Scott MacKenzie, Dr. Steven J. Castellucci, and Dr. Caitlyn Seim for their valuable contribution and helpful advice. It has been a great pleasure and honor working with you.

Many thanks to our lab members, Tafadzwa Joseph Dube, Yuan Ren, Laxmi Pandey, and Ghazal Zand, for your support, kindness, and encouragement. In addition, I would like to thank the Graduate Student Opportunity Program (GSOP) Fellowship at UC Merced for supporting me through their generous funding. I would like to acknowledge the help from the Center of Vision Enhancement (COVE) in Merced, San Francisco Mayor's Office on Disability, San Francisco Independent Living Resource Center and Golden Gate Regional Center, Modesto Healthy Aging Association, and Bay Area Outreach and Recreation Program for helping with the recruitment process for the user studies.

I am incredibly grateful to my parents, Batyrkhan Rakhmetulayev and Zaure Shandyrova, and my lovely sisters, Dr. Sabina Rakhmetulayeva and Dr. Elvira Rakhmetulayeva. Thank you for spoiling me with your love and unwavering support throughout my whole life. Thank you for believing in me and in all my ideas, especially in crazy ones, that brought me to where I am. I would also like to thank my extended family Rabiga Idelbayeva, for the help and support during my Ph.D.

I owe thanks to a very special person, my husband, Yerlan Idelbayev (Dr. Idelbayev now), for always being there for me through all my ups and downs. And I am so thankful to my baby boy, Russell, for being the major source of joy, happiness, and love! Words would never say how grateful I am to have you in my life!

Chapter 1

Introduction

Text entry is ubiquitous — we enter text everywhere on various devices: from composing emails on smartphones to replying to text messages directly from the wrist on smartwatches. Text entry is essential not only for work-related tasks but also to keep in touch with our loved ones. Although considered a standard feature today, text entry still poses significant challenges for people with vision and motor disabilities or those experiencing situational impairments[1] [86, 28, 133, 187]. Noted populations struggle to engage in these activities due to the absence of effective, enjoyable text entry techniques for mobile devices. We believe that faster and more accurate mobile text entry can make people with vision and motor disabilities more confident, productive, and elevate their economic and social standing. Therefore, it is essential to provide people with vision and motor impairments with accessible interaction on mobile devices. Since most people experience age-related declines in vision [191] and motor control [143, 36, 157], such techniques will eventually apply to a much larger population some point in life [47, 83, 191].

In pursuit of usability and enjoyability of interaction for non-disabled people, touchscreens pose significant challenges for people with vision and motor impairments. Numerous research has addressed inaccessibility of touchscreen mobile devices for both people with visual impairments [4, 28, 109, 128, 86] and people with motor impairments [7, 34, 43, 66, 67, 81, 129, 133, 176]. Researchers compel the need to make mainstream technology more accessible [86, 166], yet text entry on mobile devices remains challenging for people with vision and motor impairment.

Currently, the dominant text entry method for most mobile devices is the virtual Qwerty (see Figure 1.1). Virtual Qwerty is the software adaptation of the standard Qwerty — a physical keyboard for desktop settings. While both virtual and physical Qwerty keyboards serve their purpose, none of these methods were primarily designed for fast and accurate text entry [9]. Despite its popularity and spread, virtual Qwerty and mainstream text entry methods have many drawbacks. Here are a few fundamental challenges of mobile text entry:

[1] The state when mobile interaction is compromised due to the context in which people are using the device, e.g., during hot or cold weather, is called situational impairment

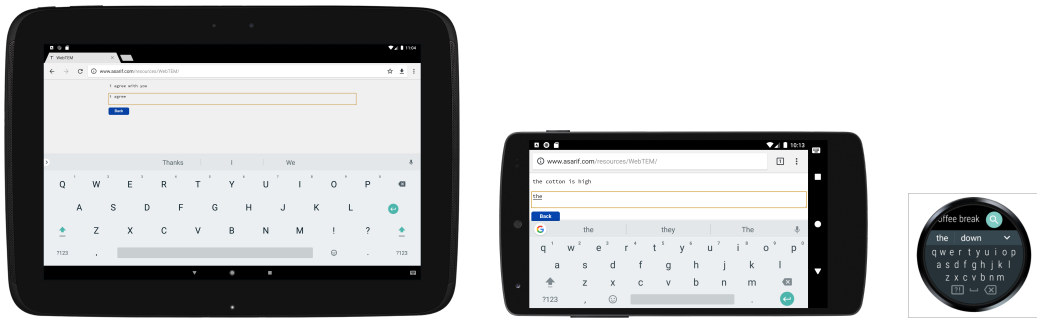


Figure 1.1: A virtual Qwerty on a tablet in landscape position (left), on a smartphone (middle), and on a smartwatch (right).

- Reliant on visuals. Existing text entry methods on mobile devices heavily rely on vision since it requires users to visually locate on-screen objects and select them precisely with the finger. This a priori excludes people with vision and motor impairments or at least deteriorates interaction experience.

- Tiny keys with no physical reference. The virtual keyboards[2] compete for the screen real-estate with the displayed content, which makes the keys on the keyboards too small, thus selecting keys is hard and error-prone due to the "fat-finger problem" [180]. The problem of tiny keys is deteriorated with the removal of physical and tactile feedback on touchscreens. For instance, with physical buttons, one can feel the actuation force on a keypress, but this concept is absent in virtual keys.

- Unavailable performance metrics. The multi-step text entry methods are commonly used on mobile devices as alternative input methods for accessibility. These methods require performing multiple actions simultaneously (chorded) or in a specific sequence (constructive) to produce input. However, evaluating these methods is difficult since traditional performance metrics were designed explicitly for non-ambiguous, uni-step methods like Qwerty.

- Expensive specialized technology. To address the listed problems, people with disabilities use external devices. Although some of these challenges could be addressed with additional dedicated assistive technology, it is expensive, bulky to carry, and might still need to be complemented with other devices. This forces people with impairments to choose inaccessible mainstream devices over specialized ones [128].

The aim of this work is to address the aforementioned challenges and bridge the gap between assistive and mainstream technology by designing inclusive text entry methods. To enable access for a diverse population we explore multi-device methods with unconventional

[2]Here, and throughout the dissertation by the virtual keyboard, we mean software keyboard

modes of text entry such as chording, constructive techniques, and physical buttons. We design text entry methods that benefit a wider range of people, which includes people with visual impairments, motor impairments, and non-disabled people.

One of the methods that we propose for low vision and blind people involves chording. Chorded techniques require the user to press multiple keys *simultaneously* to enter one character, like playing a chord on a piano. Examples of chorded methods include braille [5] and stenotype [97]. It is known that once learned, performing chords is much faster than tap sequence [183]. However, most of the existing chorded methods require a lot of time and effort to learn the technique and usually do not provide immediate usability. Chorded techniques also might not be appreciated by sighted people, because they can use visual feedback rather than learning chords. However, blind participants do not have such privilege thus might learn chords faster to perform better. We show that a carefully designed chorded keyboard can address all of these issues and surpass the QWERTY performance for blind people, and be an effective alternative in circumstances where sighted users are unable to see.

Next, we apply constructive techniques to enable text entry for people with motor impairments on smartwatches. Constructive techniques require the user to perform a combination of actions *sequentially* to enter one character. Examples of constructive methods include multi-tap [56] and Morse code [63, 168]. To make text input process stable for people with motor impairments, we augment the design with non-touch input that relies on one physical button — the digital crown located on the side of the smartwatch. The presence of tactile feedback, such as the crown, is crucial for people with limited dexterity: it serves as a physical reference for positioning the finger. We show that both people with limited dexterity and non-disabled people are able to enter text fast and accurately on smartwatches using only a single finger.

Next, we look into the possibility of skill transfer between various devices, which is crucial for a successful text entry. Herein, we explore gesture typing on smartwatches. Gesture typing requires the user to glide or swipe the finger over the letters on the keyboard to enter the word. Although gesture typing is much faster than tapping [92], it is not a dominant text entry method on mobile devices. We present an input method that facilitates skill transfer between various devices and input methods, is less visually intensive, and enables eyes-free typing. The keyboard arranges the standard QWERTY layout around the bezel of a smartwatch and is optimized for usability and to maintain similarities between the gestures drawn on a smartwatch and a virtual QWERTY. We compare our keyboard with a similar technique, and we show that our keyboard surpasses the performance of the competing method and results in a higher rate of skill transfer.

Finally, as we saw from the previous projects, there are methods that require performing multiple actions simultaneously (chorded) or in a specific sequence (constructive) to produce input. However, evaluating these methods is difficult since traditional performance metrics were designed explicitly for non-ambiguous, uni-step methods (e.g., QWERTY). As a result, they fail to capture the actual performance of a multi-step method and do not provide enough detail to aid in design improvements. We propose new metrics that account for the error

rate, accuracy, and complexity of multi-step chorded and constructive methods. We show that these metrics can help to identify probable causes of slower text entry and input errors with two existing multi-step methods.

1.1 Brief Outline

Chapter 2 presents Senorita — a chorded keyboard for sighted, low vision, and blind mobile users. It arranges the letters on eight keys in a single row by the bottom edge of the device based on letter frequencies and the anatomy of the thumbs. Unlike most chorded methods, it provides visual cues to perform the chording actions in sequence, instead of simultaneously, when the actions are unknown, facilitating "learning by doing."

Chapter 3 presents the crown-based keyboard — a technique to enable text entry with one finger using the crown of a smartwatch for people with limited dexterity. It uses a ring-shaped alphabetical layout divided into eight zones around the bezel. Users rotate the crown to switch between the zones and press the crown to select a zone.

Chapter 4 presents Swipering — a novel keyboard that arranges the QWERTY layout around the bezel of a smartwatch divided into seven zones to enable gesture typing. These zones are optimized for usability and to maintain similarities between the gestures drawn on a smartwatch and a virtual QWERTY to facilitate skill transfer.

Chapter 5 presents three new action-level performance metrics: UnitER, UA, and UnitCX. They account for the error rate, accuracy, and complexity of multi-step chorded and constructive methods.

Finally, Chapter 6 concludes the dissertation with the potential future research opportunities and extensions of the discussed works.

Chapter 2

Senorita: A Chorded Keyboard for Sighted, Low Vision, and Blind Mobile Users

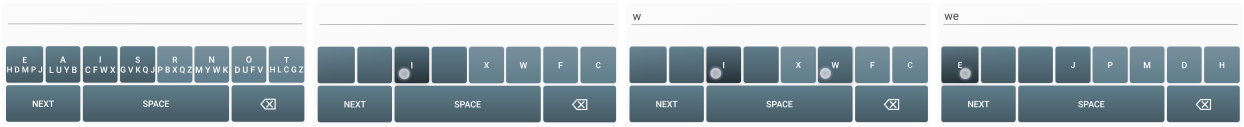


Figure 2.1: With Senorita, the left four keys are tapped with the left thumb and the right four keys are tapped with the right thumb. Tapping a key enters the letter in the top label. Tapping two keys simultaneously (chording) enters the common letter between the keys in the bottom label. The chord keys are arranged on opposite sides. Novices can tap a key to see the chords available for that key. This figure illustrates a novice user typing the word "we." She scans the keyboard from the left and finds 'w' on the 'I' key. She taps on it with her left thumb to see all chords for that key on the other side, from the edge: 'C', 'F', 'W', and 'X' (the same letters in the bottom label of the 'I' key). She taps on 'W' with her right thumb to form a chord to enter the letter. The next letter 'e' is one of the most frequent letters in English, thus has a dedicated key. She taps on it with her left thumb to complete the word.

Virtual Qwerty augmented with linguistic and behavioral models has become the dominant input method for mobile devices. It is evident that with enough practice, one can reach a reasonable entry speed with the state-of-the-art virtual Qwerty. However, using this method is difficult in some scenarios as it takes up more than half of the screen real-estate, especially when using a smartphone in landscape position, the keys are too small for precise target selection, causing frequent typing errors (the "fat-finger problem" [180]), and the

thumbs do not always reach the keys in the middle of the keyboard on larger devices, such as tablets. Although several alternatives have been proposed, Qwerty continues dominating mobile text entry since most alternatives rely on linguistic models, which make entering out-of-vocabulary words difficult, seldom impossible. Most of these also have a steep learning curve, thus require a substantial amount of time and effort in learning, encouraging users to stick to the method they are already familiar with [105]. Besides, users tend to discard a new solution if they cannot "learn by doing" [31] and are not immediately convinced that the performance and usability gain will worth the effort [32].

These make learning and using virtual Qwerty even more difficult for users with low vision and blindness[1]. Today, mobile devices are not a luxury, but essential for productivity, entertainment, and to keep in touch with loved ones. A large part of these activities requires text entry. Visually impaired users struggle to engage in these due to the absence of effective eyes-free text entry techniques for mobile devices. Most existing solutions rely on braille, when braille literacy in this population is merely 10% [26]. Other alternatives are time-consuming, error-prone, and have a high learning curve. Speech-to-text is becoming increasingly reliable, but not effective in loud environments [40] and compromises privacy and security [41].

Although the aforementioned scenarios may seem unrelated on the surface, a compact layout that does not occupy most of the screen, has larger keys to accommodate precise target selection, provides comfortable reach to all keys, facilitates learning, and enables the entry of out-of-vocabulary words is desired in all. Senorita is a two-thumb chorded keyboard[2] designed to meet these needs (Figure 2.1). It enables visually impaired people to enter text on mobile devices, and is effective in circumstances where sighted users are unable to see, such as when wearing surgical eye patches or have lost prescription eyeglasses. It is also a "comfortable" alternative to Qwerty in situations where comfort is more desired than speed, such as while commenting on a video or texting while walking. Its design is motivated by the "design for all" philosophy to accommodate the maximum possible group of users [156, 169].

The remainder of the Chapter is organized as follows. First, we discuss the design of Senorita and argue the benefits of chording. We summarize related work in the area. We then present results of a longitudinal study evaluating the keyboard on a smartphone. Then, we customize Senorita for larger devices and evaluate it on a tablet. Finally, we discuss the design modifications for low vision and blind users, and present results of a longitudinal study evaluating Senorita with the target user group. We conclude with speculations on future extensions.

[1]This work refers to people with *severe low vision* to *near-total blindness* with a visual acuity between 20/200 and 20/1,200 as "low vision" and people with *no light-perception* as "blind" [192]. Both groups require assistive technology to use mobile devices.

[2]With chorded keyboards, users press multiple keys simultaneously to form a chord (like playing a chord on piano) to enter a letter.

2.1 Designing Senorita

Senorita is a novel two-thumb chorded keyboard with eight keys laid out in a single row (Figure 2.1). The left four keys are tapped with the left thumb and the right four keys are tapped with the right thumb. It assigns dedicated keys for the most frequent eight letters in the English language: 'E', 'A', 'I', 'S', 'R', 'N', 'O', and 'T' [99]. The name "Senorita" is an anagram of these letters. To enter these letters, users tap once on the respective keys. The remaining eighteen letters are entered by performing chords. Each chord is composed of two keys from the opposite sides of the keyboard. However, the chord keys for the most infrequent letters 'J' and 'Z' are placed on the same side, the furthest from each other to ensure easier reach. Senorita dynamically changes size based on the screen width and the average thumb length (7.65 cm [142]) to ensure that these keys are never out of reach. In a pilot, users were able to enter these two letters on a smartphone in landscape position without any difficulty. The bottom label of each key displays the chords it can construct. Tapping two keys simultaneously enters the common letter between these keys. For example, tapping 'I' and 'N' together enters the letter 'w' (Figure 2.1).

The layout was designed keeping letter frequencies and the anatomy of the thumbs in mind. First, we used a linguistic table for the relative frequencies of all letter pairs in the English alphabet to sort all letters by frequency [99]. We then placed the most frequent letters closer to the side bezels to make them easier for the thumbs to reach [18, 138] and the infrequent letters further away to reduce thumb movement. The chords are composed of keys from the opposite sides to make them easier to enter with the thumbs. We only considered the horizontal reach of the thumbs to leave the upper area of the screen unoccluded and for the users to use the bottom bezel as a physical reference to facilitate eyes-free text entry. Using the side bezels as physical reference is inconvenient since reaching the topmost and the nethermost keys require extending the thumbs too far [138]. This convention resulted in two possible layouts: Senorita and flipped-Senorita that had the keys switch sides (left keys to the right and vice versa). In a pilot, both keyboards yielded comparable results, yet we selected Senorita since its key arrangement match the most with Qwerty, which may make the transition to Senorita easier for some users. Notice in Figure 2.1 that the letters in the bottom labels appear in reversed order than the order in which the keys appear on the opposite side. For example, the 'A' key shows the chords as 'LUYB', while the keys on the other side appear in the order of 'BYUL'. This is because we arranged the letters in the labels and the keys from the edges. While it may seem counterintuitive, a pilot revealed that users prefer this design as they naturally scan the letters from the edges. It also facilitates eyes-free text entry since visually impaired users tend to slide their thumbs from the edges. Currently, Senorita does not support the entry of upper-case letters, special symbols, numbers, or languages other than English. But support for these could be easily added by enabling the user to long-press, double-tap, or dwell over a key to switch back and forth between the cases, or change the layout to enter digits and symbols.

Senorita provides visual feedback to facilitate learning. The two sides are distinguished using two different shades of grey. The top and bottom labels in each key represent tap and

chord-based keys, respectively. When the user taps a key, Senorita displays only the possible chords on the other side (Figure 2.1). This enables novices to press two keys in sequence, instead of simultaneously. Hence, unlike most chorded methods, Senorita does not require users to learn the chords before using it, instead allows them to learn the chords as they type.

2.1.1 Benefits of Chording

Once learned, performing chords is much faster than tap sequence [183]. Stenotype is a chorded method still in use by court reporters as expert stenotyping speed is much faster than expert Qwerty [119]. The main challenge of existing chorded methods is that they require substantial time and effort to master. Thus, most existing chorded keyboards lack in immediate usability and require users to learn the chords before typing. Senorita attempts to mitigate this by letting users enter the most frequent letters with a single tap, using only two keys per chord (most methods use more), and letting user either perform the chording actions simultaneously or in sequences from either side of the keyboard to enter the chorded letters. The goal was to enable users to start typing immediately and learn the chords along the way. Using chords also enabled us to fit the English alphabet in eight large keys, which facilitates precise target selection and saves precious touchscreen real-estate. It also eliminated the need for using a decoder to disambiguate input, providing the support for out-of-vocabulary words.

2.2 Related Work

This section reviews reduced-size and split virtual keyboards, and text entry solutions for visually impaired people. It does not discuss speech-to-text and physical/non-touchscreen solutions since these are outside the scope of this work.

2.2.1 Reduced-size Keyboards

While numerous works have focused on developing linguistic and behavioral models [18, 51, 54, 103, 179] for faster and more accurate text entry with virtual Qwerty and designed novel keyboard layouts that are comparable to Qwerty in size [24, 120], not many have focused on reduced-size virtual keyboards. Romano et al. [150] designed a single-row tap-slide hybrid keyboard for mobile devices, but did not evaluate it empirically. Stick Keyboard [60] maps four rows of a standard Qwerty onto the home row. Although designed for smartphones, it was evaluated on a desktop using a physical prototype, where it yielded 10.4 wpm without the support of a linguistic model. 1Line Keyboard [104] is a similar virtual keyboard, designed for tablets. With the support of a statistical decoder, it yielded 30.7 wpm in a longitudinal study. Gueorguieva et al. [63] designed a reduced virtual keyboard to enable text entry through Morse code. In a user study, it reached 7 wpm by the seventh session without the

support of a linguistic model. In a different work, Zhu et al. [198] showed that with sufficient practice, expert virtual Qwerty users can reach 37.9 wpm with an invisible Qwerty keyboard augmented with a statistical decoder.

2.2.2 Split Keyboards

Some have explored split keyboards for tablets, which require both vertical and horizontal movement of the thumbs. Yazdi et al. [2] optimized splitting of a Qwerty keyboard for comfortable thumb movement. In a user study, it yielded 27.6 wpm. Bi et al. [22] enabled gesture typing with both thumbs on a split Qwerty. With the support of a statistical decoder, this method reached 26 wpm. KALQ [138] is a novel split keyboard optimized for thumb movements. It also uses a statistical decoder, and reached 37.1 wpm in a user study.

Method	**Speed**
BrailleTouch [172]	9.40–23.20 wpm
TypeInBraille [126]	6.30 wpm
Mobile Braille [29]	5.05 wpm
SingleTapBraille [3]	4.71 wpm
BrailleType [135]	1.49 wpm

Table 2.1: Entry speed of popular braille-based virtual keyboards.

2.2.3 Keyboards for Visually Impaired People

Most text entry solutions for visually impaired people rely on the knowledge of braille (Table 2.1), when braille literacy in this population is only 10% [26]. Alternative solutions include Escape-Keyboard [20] that enables entering letters by pressing the thumb on different areas of the screen, then performing directional strokes. It used a statistical decoder to improve its performance. In a longitudinal study, sighted participants yielded 14.7 wpm in an eyes-free condition. A similar method, ThumbStroke [95], yielded 10.5 wpm in a longitudinal study with sighted participants. No-Look Notes [27] arranges all letters in an eight-segment pie menu. Users perform a series of taps and stokes with both hands to enter a letter. In a study, it reached 1.67 wpm. EdgeWrite [188] enables entering letters by *"traversing the edges and diagonals of a square hole"* imposed over the screen. Although initially designed for people with motor impairments, it enables eyes-free text entry [94]. In a study, it yielded 6.6 wpm with able-bodied sighted participants. NavTouch [65] lets users navigate the alphabet by performing directional gestures using the vowels as anchors. In a study, blind users reached 1.72 wpm with this method on a smartphone. SpatialTouch [64] is a virtual Qwerty that exploits users' experience with physical Qwerty to enable eyes-free text entry through

multi-touch exploration and spatial, simultaneous auditory feedback. In a study, blind participants yielded 2–3 wpm with this method. AGTex [25] is a screen-reader supported virtual Qwerty that enables eyes-free gesture typing by switching between two modes. In a study, blind users reached 5.66 wpm. Some have also evaluated multi-tap with blind participants [135], where it yielded 2 wpm.

2.3 User Study 1: Smartphone

We conducted a longitudinal study to evaluate Senorita on a smartphone in landscape position.

2.3.1 Apparatus

We used a Motorola Moto G^5 Plus smartphone ($150.2 \times 74 \times 7.7$ mm, 155 g) at 1080×1920 pixels. A custom application was developed using the Android Studio 3.1, SDK 27 to record all performance metrics and interactions with timestamps.

2.3.2 Participants

Ten sighted volunteers aged from 20 to 31 years (M = 23.8, SD = 3.46) were recruited in the study (Figure 2.2) through online social communities, the local university, and by word of mouth. Three of them were female and seven were male. They all were experienced users of virtual Qwerty (M = 8.0 years of experience, SD = 3.44). Six of them rated themselves as native/bilingual-level, one rated herself as advanced-level, and three rated themselves as moderate-level English speakers. None of them had prior experience with chorded keyboards. Eight of them were right-handed and two were left-handed. They all received US $50 for participating in the study.

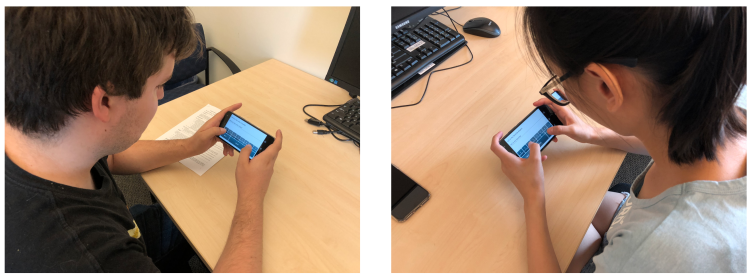


Figure 2.2: Two volunteers participating in the first study with a smartphone in landscape position.

2.3.3 Design

The study used a within-subjects design, where the independent variables were the session and method and the dependent variables were the performance metrics. We recorded the commonly used *words per minute (wpm)* and *error rate* [14, 171] metrics to measure speed and accuracy, respectively. We also recorded *chording rate*, which is the average percentage of chords performed per session. If there were 300 chords available in a session and the users performed 90, then the chording rate for the session is 30%. This metric was calculated only for Senorita as Qwerty does not have chords. In summary, the design was as follows.

Qwerty	**Senorita**
10 participants ×	10 participants ×
1 session (first day) ×	10 sessions (different days) ×
15 random phrases [118]	15 random phrases [118]
= 150 phrases.	= 1,500 phrases.

2.3.4 Procedure

The study was conducted in a quiet room. On the first day (Session 1), we explained the research to all participants and collected their consents and demographics. We then started the Qwerty condition, where participants transcribed 15 random English phrases from a set [118] using the default Android Qwerty. We changed all uppercase letters to lowercase, and all British spellings to American usage. We chose this corpus due to its high correlation with English language character frequencies and its wide use in text entry studies. All predictive features of the keyboard were disabled to eliminate a potential confound. The purpose of this condition was to record participants' speed and accuracy with Qwerty. We did not include this condition in the following sessions since they all were experienced virtual Qwerty users. Then, we demonstrated Senorita and enabled all participants to practice with it by transcribing two random phrases. These phrases were not repeated in the study. They then completed a System Usability Scale (SUS) [177] inspired pre-study questionnaire that asked them to rate various aspects of Senorita on a 7-point Likert scale. Its purpose was to record their immediate impression of the keyboard. The Senorita condition started after that, where participants transcribed 15 phrases using the keyboard.

Both conditions presented one phrase at a time. The phrase was shown at the top of the screen during the text entry. Participants were instructed to memorize the phrase, transcribe it *"as fast and accurate as possible"*, then tap the NEXT key to see the next phrase. Error correction was recommended but not forced. All participants held the device in a landscape position (Figure 2.2). They could take short breaks between the phrases when needed. Logging started from the first keystroke and ended when participants pressed NEXT. The sessions were scheduled on different days, with at most a two-day gap in between. All sessions followed the same procedure, except for the Qwerty condition and the practice block, which

were exclusive to Session 1. Upon completion of the study, participants completed a short post-study questionnaire that included the same questions as the pre-study questionnaire. The goal was to find out if practice influenced participants' impression of the keyboard.

2.3.5 Results

A Shapiro-Wilk test and a Mauchly's test indicated that the response variable (metrics) residuals are normally distributed and the variances of populations are equal, respectively. Hence, we used a repeated-measures ANOVA for all analysis. Only the performance of the last session (Session 10) was considered to compare Senorita with Qwerty.

2.3.5.1 Entry Speed

An ANOVA identified a significant effect of session on entry speed ($F_{9,9} = 47.9, p < .0001$). A Tukey-Kramer test revealed three distinct groups: 1–4, 5–7, and 8–10. An ANOVA also identified a significant effect of method ($F_{1,9} = 243.02, p < .0001$). Average entry speed with Qwerty and Senorita were 32.7 wpm (SD = 8.9) and 13.99 (SD = 3.3) wpm, respectively. Figure 2.3 shows average entry speed per session with Senorita.

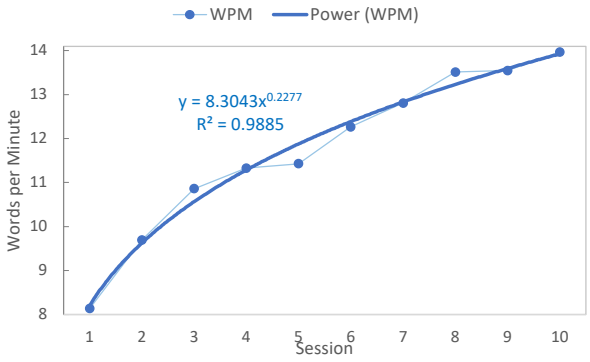


Figure 2.3: Average entry speed (wpm) per session with Senorita on a smartphone, fitted to a power trendline.

2.3.5.2 Error Rate

An ANOVA failed to identify a significant effect of session on error rate ($F_{9,9} = 0.95, p = .48$). There was also no significant effect of method ($F_{1,9} = 1.40, p = .27$). Average error rate with Qwerty and Senorita were 0.92% (SD = 1.67) and 0.89% (SD = 1.32), respectively. Figure 2.4 displays average error rate per session with Senorita.

2.3.5.3 Chording Rate

An ANOVA failed to identify a significant effect of session on chording rate ($F_{9,9} = 1.74, p = .09$). Average chording rate in all sessions was 9.71% (SD = 18.3). The most frequent chords in the study were 'h' (31.1%) and 'l' (28.6%). Figure 2.5 displays average chording rate per session.

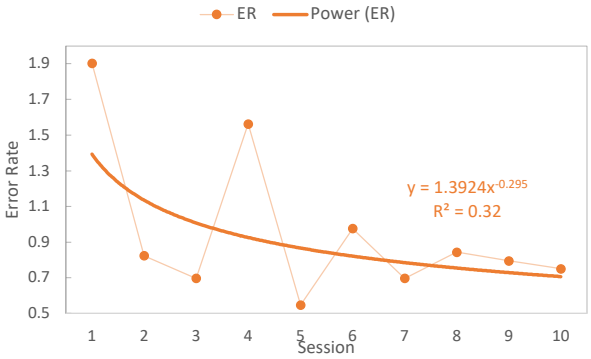


Figure 2.4: Average error rate per session with Senorita on a smartphone, fitted to a power trendline.

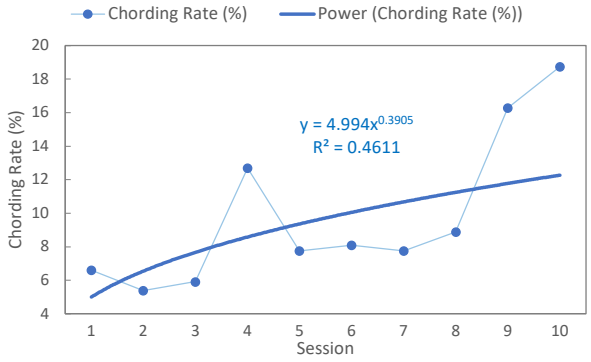


Figure 2.5: Average chording rate per session with Senorita on a smartphone, fitted to a power trendline.

2.3.6 User Feedback

A Wilcoxon Signed-Rank test revealed that user opinion about Senorita changed significantly in regard to willingness to use ($z = -2.39, p < .05$) and learnability ($z = -2.5, p < .05$) after

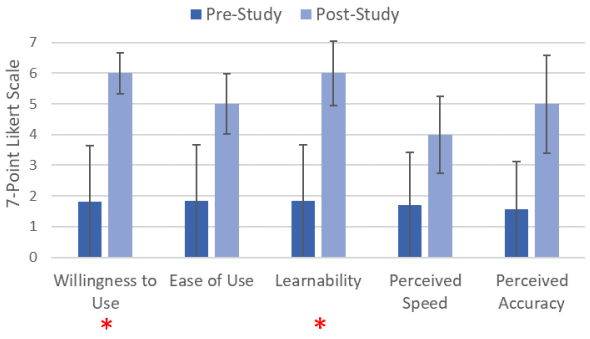


Figure 2.6: Median user ratings of willingness to use, ease of use, learnability, perceived speed, and perceived accuracy of Senorita on a 7-point Likert scale, where '1' to '7' signify 'strongly disagree' to 'strongly agree'. The error bars represent ±1 standard deviation (SD) and the red asterisks signify statistical significance.

using it. But no significant effects were identified on ease of use ($z = -6.8, p = .09$), perceived speed ($z = -0.7, p = .4$), and perceived accuracy ($z = -1.6, p = .1$). Figure 2.6 illustrates median user ratings of all investigated aspects of the keyboard. The following questions were used in the pre and post-study questionnaire: "1. I think that I would like to use the chorded keyboard frequently; 2. I thought the chorded keyboard was easy to use; 3. I would imagine that most people would learn to use the chorded keyboard very quickly; 4. The chorded keyboard will improve my input speed; 5. The chorded keyboard will improve my input accuracy."

2.3.7 Discussion

Senorita yielded a competitive entry speed. Similar virtual keyboards for smartphones yielded a maximum of 11 wpm with sighted participants [95] compared to Senorita's 14 wpm. Senorita was significantly slower (60%) than Qwerty. But this speed was achieved with only a 19% chording rate, thus likely to improve with sufficient practice. The average entry speed over session correlated well ($R^2 = 0.9885$) with the power law of practice [167]. The fact that learning occurred even in the last session indicates that Senorita did not reach its highest possible speed in the study. Learning of chords occurred at a slower rate, presumably because participants had the choice of performing the chording actions in sequence, instead of simultaneously. Prior studies showed that users learn a new method faster in the absence of a reliable alternative [16]. While forcing users to use chords could have improved entry speed, compromised the keyboard's immediate usability. Senorita's error rate was comparable to Qwerty ($< 1\%$). There was no significant difference in error rate between the sessions, which suggests that Senorita was moderately accurate from the start. User feedback revealed that initially most participants felt that learning and using Senorita

will be difficult, slower, and more error-prone, hence did not show much interest in using it on mobile devices. However, their responses were much positive after practice (Figure 2.6). Particularly, their opinion about the learnability of the keyboard and willingness to use it on mobile devices were significantly more positive. Most of them felt that Senorita could be an effective alternative to Qwerty in special scenarios.

2.4 User Study 2: Tablet

We conducted a study to evaluate Senorita on a tablet. We made two design adjustments for this. The two sides of the keyboard split based on the average thumb length (7.65 cm [142]) and the 'J' and 'Z' keys switch sides when the first key of a chord is tapped. These were to ensure that the thumbs can comfortably reach all keys (Figure 2.7) and most of the touchscreen is available for the users to engage in other activities.

Figure 2.7: Senorita dynamically resizes, splits, and the 'J' and 'Z' keys jump sides when chording (bottom) to assure a comfortable reach to all keys on larger devices that are usually held with two hands.

2.4.1 Apparatus

We used a Samsung Galaxy Tab A (212.09×124.206×8.89 mm, 690 g) at 1280×800 pixels. A custom application developed with the Android Studio 3.1, SDK 27 recorded all metrics and interactions with timestamps (Figure 2.8).

2.4.2 Participants

Ten new sighted participants aged from 24 to 30 years (M = 26.8, SD = 1.99) took part in the study (Figure 2.8). Two of them were female and eight were male. They all were experienced users of mobile Qwerty (M = 7.5 years of experience, SD = 2.22). Four of them rated themselves as advanced-level and six rated themselves as moderate-level English speakers. None of them had prior experience with chorded keyboards. Nine of them were right-handed and one was left-handed. They all received US $10 for volunteering.

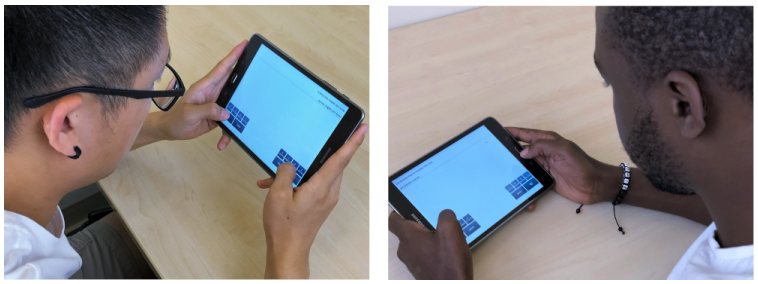


Figure 2.8: Two participants taking part in the second study with a tablet.

2.4.3 Design

The study used a within-subjects design, where the independent variable was block and method, and the dependent variables were the performance metrics. We recorded the same metrics as the first user study. In summary, the design was:

Qwerty	**Senorita**
10 participants ×	10 participants ×
1 block ×	5 blocks ×
10 random phrases [118]	10 random phrases [118]
= 100 phrases.	= 500 phrases.

2.4.4 Procedure

This user study was conducted in a quiet room. We started by explaining the procedure to all participants and collecting their consents and demographics. We then started the Qwerty condition, where participants transcribed 10 random English phrases from a set [118] using the default Android Qwerty. The WebTEM [12] application was used to record all performance metrics. All predictive features of the keyboard were disabled to eliminate a potential confound. We then introduced Senorita and enabled all to practice with it by transcribing two random phrases. These phrases were not repeated in the study. Participants then completed a System Usability Scale (SUS) [177] inspired short pre-study questionnaire that asked them to rate various aspects of Senorita on a 7-point Likert scale. Its purpose was to record the participants' initial impression of the keyboard. The Senorita condition started after that, where we instructed participants to transcribe fifty random phrases in five blocks (10×5) on the same day. Both conditions presented one phrase at a time. Participants were asked to memorize the phrase, transcribe it "as *fast and accurate as possible,*" then tap the NEXT key to see the next phrase. Error correction was recommended but not forced. Participants could take short breaks between phrases, when needed. Logging started from the first keystroke and ended when users NEXT. Once done, they all completed a short

post-study questionnaire that included the same questions as the pre-study questionnaire to investigate if practice influenced participants' impression of the keyboard.

2.4.5 Results

A Shapiro-Wilk test and a Mauchly's test indicated that the response variable residuals are normally distributed and the variances of populations are equal, respectively. Hence, we used a repeated-measures ANOVA for all analysis. Only the performance of the last block was used for the statistical tests between methods.

2.4.5.1 Entry Speed

An ANOVA identified a significant effect of block on entry speed ($F_{4,9} = 7.57, p < .0005$). There was also a significant effect of method ($F_{1,9} = 16.55, p < .005$). Average speed with Qwerty and Senorita were 27.18 wpm (SD = 8.23) and 9.27 wpm (SD = 2.96), respectively. Figure 2.9 displays average entry speed per block with Senorita.

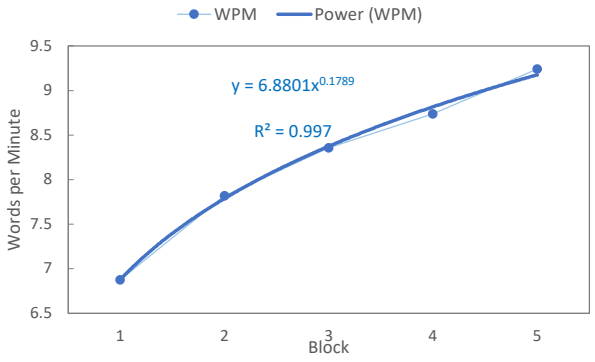


Figure 2.9: Average entry speed (wpm) per block with Senorita on a tablet, fitted to a power trendline.

2.4.5.2 Error Rate

An ANOVA failed to identify a significant effect of block on error rate ($F_{4,9} = 0.70, p = .6$). There was also no significant effect of method ($F_{1,9} = 0.04, p = .85$). Average error rate with Qwerty and Senorita were 2.27% (SD = 0.68) and 2.16% (SD = 4.56), respectively. Figure 2.10 displays average error rate per block with Senorita.

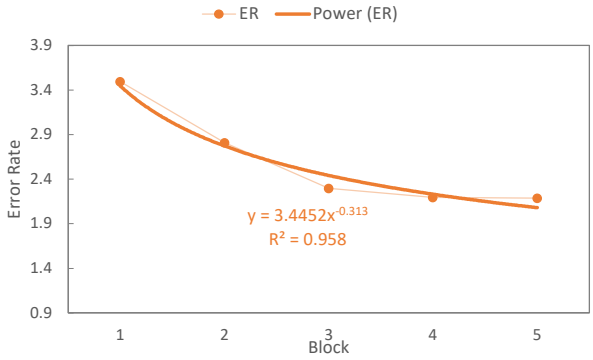


Figure 2.10: Average error rate per block with Senorita on a tablet, fitted to a power trendline.

2.4.5.3 Chording Rate

An ANOVA failed to identify a significant effect of block on chording rate ($F_{4,9} = 0.43, p = .78$). Average chording rate in all sessions was 9.4% (SD = 16.9). As the previous study, the most frequent chords were 'h' (35.6%) and 'l' (12.4%). Figure 2.11 displays average chording rate per session.

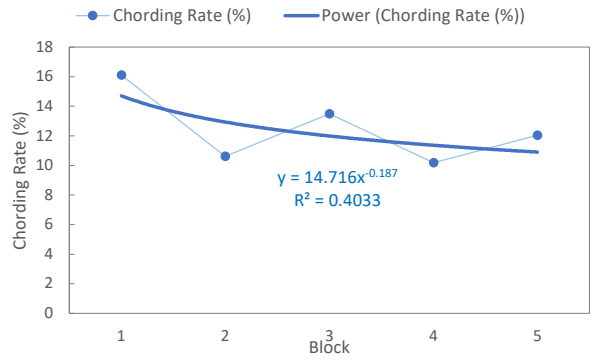


Figure 2.11: Average chording rate per block with Senorita on a tablet, fitted to a power trendline. No clear trend is visible here, which is not surprising considering the short duration of the study.

2.4.6 User Feedback

A Wilcoxon Signed-Rank test failed to identify any significant change in user opinion about Senorita in regard to willingness to use ($z = -0.877, p = .4$), ease of use ($z = -0.857, p = .4$), learnability ($z = -0.33, p = .7$), perceived speed ($z = -1.56, p = .1$), or perceived accuracy ($z = -0.877, p = .4$). Figure 2.12 displays median user ratings of all explored aspects of Senorita.

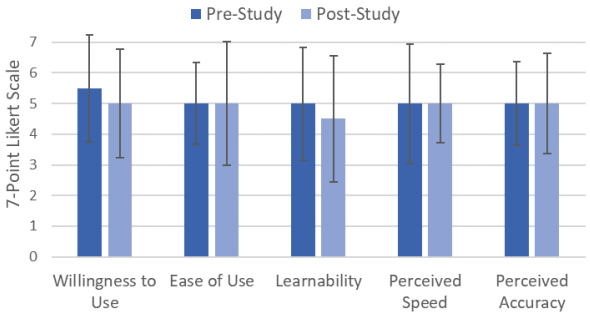


Figure 2.12: Median User ratings of Senorita's willingness to use, ease of use, learnability, perceived speed, and perceived accuracy on a 7-point Likert scale, where '1' to '7' signify 'strongly disagree' to 'strongly agree'. The error bars represent ±1 standard deviation (SD).

2.4.7 Discussion

Participants reached a 9.3 wpm entry speed and a 2.2% error rate in a single session (10 phrases), which is comparable to Senorita's 8.23 wpm and 2% error rate in Session 1 of the first study (15 phrases). We find this inspiring since smartphone keyboards tend to reduce in speed and accuracy on tablets [2, 22]. It is also promising that learning occurred in this short-term study. Compared to the first block, entry speed with Senorita increased by 23% in the last block. Like the first study, the average entry speed over block correlated well ($R^2 = 0.997$) with the power law of practice [167]. Learning occurred even in the last block, which suggest that Senorita's entry speed on tablet is likely to increase substantially with practice. As expected, Senorita was significantly slower (66%) than Qwerty. But this speed was achieved with only a 9.4% chording rate. Hence, we speculate that with practice both chording rate and chording time will increase, resulting in a higher entry speed.

The 1Line Keyboard [104] that also maps all letters to a single row performed much better on a tablet than Senorita. However, its 30.7 wpm entry speed was achieved with the support of a linguistic model. This keyboard relies on a decoder to disambiguate the input, which makes entering out-of-vocabulary words difficult. An almost identical method [60] yielded a 10.4 wpm without the support of a linguistic model, suggesting that the former's

performance gain was due to its predictive system. Senorita's performance could also improve substantially when augmented with word prediction and auto-correction.

Interestingly, participants were on board with Senorita from the start. Most felt that Senorita will be easy to learn and use, improve their entry speed and accuracy, thus wanted to use it on tablets even before trying it. Their impression of Senorita did not deviate much after practice. This finding highlights users' frustration with Qwerty on larger devices like tablets. Accordingly, most participants felt that it can be an effective alternative to Qwerty is special circumstances.

2.5 Eyes-Free Senorita

We used an iterative design process to fine-tune Senorita for visually impaired people. First, we customized the design based on the existing literature and design guidelines (e.g., [69, 100, 140]), then evaluated it in pilot studies involving blindfolded sighted participants[3]. We refined the design based on the findings. This process was repeated until the design reached a satisfactory level. We then tested Senorita with a representative user group, where five low vision and blind participants (2 female, 3 male) used Senorita on a smartphone, then took part in a focus group discussing the challenges in mobile text entry, and the design of the keyboard. All participants were supportive of Senorita from the start, but suggested some design modifications. They all appreciated that Senorita has only eight keys, thus does not require scanning their thumbs through an array of keys. They also found it ergonomic as it does not require stretching their thumbs vertically. They all expressed their frustration with the existing solutions and were willing to invest time and effort in learning a new keyboard if it is effective and user-friendly. We went through another design iteration to address the feedback from the focus group. The final design (Figure 2.13) made the following modifications.

2.5.1 Hybrid and Adaptive Design

We added a high-contrast black-and-white theme to provide visual aids to low vision users with some light-perception [100]. We removed all inactive areas between the keys (known as "gutter") to enable smooth sliding between the keys without lifting the thumbs. We then removed the letters 'J' and 'Z' from the 'S' and 'R' keys, and kept them only on the 'E' and 'T' keys, as visually impaired users found their placement in both the edge and center keys confusing. This is interesting since sighted users did not complain about it. Finally, we replaced the SPACE and BACKSPACE keys with directional strokes, enabling users to enter a space by performing a *right swipe* and delete a letter by performing a *left swipe* anywhere on the screen. While we acknowledge that these gestures could interfere with underlying apps, this can be addressed by using a simple behavioral model. Participants could also *left swipe and hold* for repeated backspaces (like press-holding the BACKSPACE key). Not only this

[3]Blindfolded is defined as a person whose eyes are covered so the person cannot see

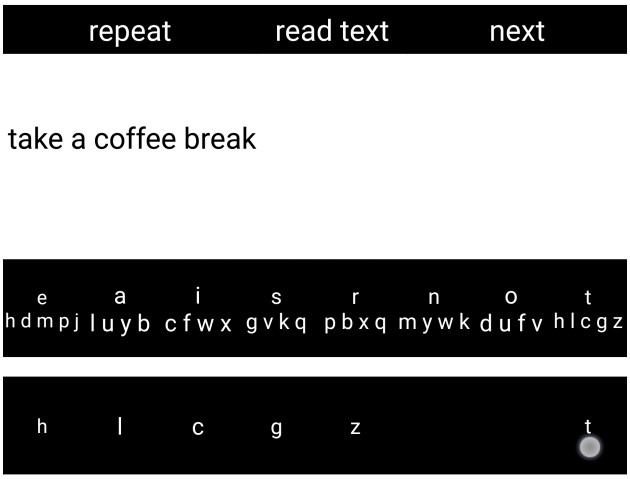


Figure 2.13: Senorita dynamically resizes the layout, splits the two sides (when needed), and switches positions of the 'J' and 'Z' keys when the first key of a chord is tapped (bottom) to make sure that the thumbs can comfortably reach all keys when holding a device with two hands.

made the design more intuitive but brought the keys closer to the bottom bezel, enabling users to use it as a physical reference (slide their thumbs along the edge).

2.5.2 Screen Reader

We augmented a screen reader to Senorita to provide auditory feedback on each input and interaction. This feature was implemented using the Android SDK 27's `TextToSpeech` class [6]. It uses a female voice (`Voice I`) from seven available options in default pitch and 2x speed, based on the feedback from the focus group. When a thumb slides over the keys, it reads the letters on the keys as the thumb touches them. For example, when the thumb touches the 'T' key (Figure 2.13, bottom), it reads 'T', pauses for a brief moment, then reads 'HLCGZ'. Lifting the thumb enters the first letter 'T'. When the other thumb touches the keys from the other side, Senorita reads the chorded letters on the respective keys, specifically 'H', 'L', 'C', 'G', and 'Z'. Lifting the finger on one of these keys enters the respective letter. When the thumb leaves a key, Senorita starts reading the next key immediately, without keeping the user waiting. This enables novices to slowly slide over the keys to find the target letter, while experts could swiftly slide to the intended key. Senorita confirms all input, such as "*A entered*" or "*B deleted*." When space is entered, it reads the last entered word for the user to verify the input. The user could also press the READ TEXT key or *swipe down* anywhere on the screen to hear what has been typed so far. In pilots, we did not receive any complaints from the participants about not being able to comprehend the spoken letters or an increased cognitive load due to information overload.

2.6 User Study 3: Low Vision and Blind Users

We evaluated the refined eyes-free design with a representative user group in a longitudinal user study.

2.6.1 Apparatus

We used a Motorola Moto G^5 Plus smartphone ($150.2 \times 74 \times 7.7$ mm, 155 g) at 1080×1920 pixels. A custom application was developed using the Android Studio 3.1, SDK 27 to record all metrics and interactions with timestamps.

2.6.2 Participants

Initially, we recruited thirteen participants through the Center of Vision Enhancement (COVE) in Merced, CA. But two participants withdrew from the study due to personal reasons. Eleven participants, three blind people (2 female, 1 male, M = 52 years, SD = 7) and eight low vision[1] people (6 female, 2 male, M = 38.13 years, SD = 11.24), took part in the study. Their age ranged from 27 to 62 years (M = 41.91, SD = 11.84). Eight of them were female and three were male. Nine of them rated themselves as native/bilingual and two rated themselves as advanced-level English speakers. They all were frequent users of virtual Qwerty with a screen reader (M = 6.93 years of experience, SD = 3.15). They all expressed their frustration with Qwerty (P5, female, 57 years, "*I hate using Qwerty, it is slow and frustrating*"). Some of them also used Bluetooth keyboards (N = 3) and speech-to-text (N = 5) to enter text on mobile devices. Using these methods, they composed on average 12.9 text messages per day (SD = 7.6). Nine participants were right-handed, one left-handed, and one was ambidextrous. Five of them were Braille literate. COVE arranged transportation to the study for those who needed it. They all received a US $60 gift card for volunteering.

Figure 2.14: A low vision person (left) and a blind person (right) participating in the third user study.

2.6.3 Design

We used a within-subjects design, where the independent variable were the session and method and the dependent variables were the performance metrics. We recorded the same metrics as the previous studies. In summary, the design was as follows.

Qwerty	**Senorita**
11 participants ×	11 participants ×
1 session ×	6 sessions × as many random
10 random phrases [118]	phrases [118] as possible in 20m
= 100 phrases.	= 887 phrases.

2.6.4 Procedure

The study was conducted in a quiet room at COVE. We shared the informed consent form with potential volunteers ahead of time for them to learn about the research. Upon arrival, we explained the procedure again, responded to any questions they had, then asked them to verbally consent to participate in the study. A sighted employee of COVE was present to witness this process and sign the consent form on their behalf. Participants then responded to a demographics and mobile usage questionnaire, where we read the questions and the options to them, and recorded all responses. The first condition started after that, where we asked participants to pick any virtual keyboard of their choice to transcribe 15 random phrases from a set [118]. They all chose Qwerty with a screen reader. We disabled all predictive features of this keyboard to eliminate a potential confound. We used the WebTEM [12] app to record all metrics. We did not include this condition in the following sessions since all participants were experienced in virtual Qwerty. Then, we demonstrated Senorita and enabled participants to practice with it by transcribing two random phrases, which were not repeated in the study. The Senorita condition started after that, where participants were instructed to transcribe as many phrases as possible with Senorita in 20 minutes. Both conditions read one phrase at a time. Participants were instructed to memorize the phrase, transcribe it "*as fast and accurate as possible*," then tap the NEXT key to see the next phrase. Participants could hear the phrase again by either pressing the REPEAT key (Figure 2.13) or *swiping up* anywhere on the screen, and SWIPE DOWN to hear what they have entered so far. Error correction was recommended but not forced. Participants held the device in landscape position (Figure 2.14). They could take short breaks between the phrases, when needed. Logging started from the first keystroke and ended when participants pressed NEXT. The sessions were scheduled from 90 minutes to 24 hours apart, with a maximum of three sessions per day. All sessions followed the same procedure, except for the Qwerty condition and the practice block, which were exclusive to Session 1. All sessions were video-recorded for further analysis. When done, participants took part in a brief interview session where they were asked to comment on the performance and usability of Senorita.

2.6.5 Results

A Shapiro-Wilk test and a Mauchly's test indicated that the response variable residuals are normally distributed and the variances of populations are equal, respectively. Hence, we used a repeated-measures ANOVA for all analysis. Only the performance of the last session was considered for the statistical tests comparing methods.

2.6.5.1 Entry Speed

An ANOVA identified a significant effect of session on entry speed ($F_{5,10} = 40.79, p < .0001$). A Tukey-Kramer test revealed two distinct groups: 1–4 and 5–6. An ANOVA also identified a significant effect of method ($F_{5,10} = 6.62, p < .05$). Average entry speed with Qwerty and Senorita were 11.4 wpm (SD = 9.1) and 5.5 wpm (SD = 1.64), respectively. Further, a One-way ANOVA identified a significant effect of level-of-sight on entry speed for both Qwerty ($F_{1,108} = 61.06, p < .0001$) and Senorita ($F_{1,885} = 219.2, p < .0001$). Low vision and blind participants yielded on average 14.73 wpm (SD = 8.52) and 2.52 wpm (SD = 0.8) with Qwerty, respectively, and 5.8 wpm (SD = 1.55) and 3.69 wpm (SD = 0.85) with Senorita, respectively. Figure 2.15 displays average entry speed per session with Senorita.

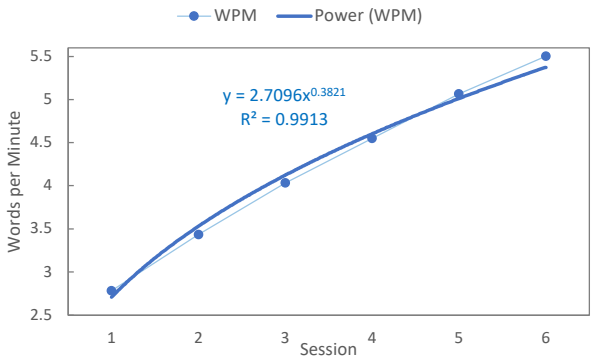


Figure 2.15: Average eyes-free text entry speed (wpm) per session with Senorita, fitted to a power trendline.

2.6.5.2 Error Rate

An ANOVA identified a significant effect of session on error rate ($F_{5,10} = 2.58, p < .05$). A Tukey-Kramer revealed two distinct groups: 1–4 and 5–6. An ANOVA also identified a significant effect of method ($F_{1,9} = 243.02, p < .0001$). Average error rate with Qwerty and Senorita were 8.5% (SD = 25.2) and 5.46% (SD = 8.4), respectively. A One-way ANOVA failed to identify a significant effect of level-of-sight on error rate for Qwerty ($F_{1,108} = 0.02, p = .88$), but identified a significant effect for Senorita ($F_{1,885} = 54.35, p < .0001$). Low

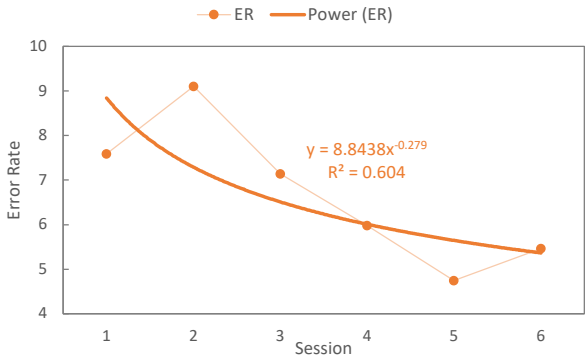


Figure 2.16: Average eyes-free error rate per session with Senorita, fitted to a power trendline.

vision and blind participants yielded on average 8.26% (SD = 28.53) and 9.06% (SD = 12.91) error rate with Qwerty, respectively, and 4.2% (SD = 6.52) and 13.24% (SD = 13.36) error rate with Senorita, respectively. Figure 2.16 displays average error rate per session with Senorita.

2.6.5.3 Chording Rate

An ANOVA identified a significant effect of session on chording rate ($F_{5,10} = 18.30, p < .0001$). Average chording rate in all sessions was 25.01% (SD = 21.45). A One-way ANOVA identified a significant effect of level-of-sight on chording rate ($F_{1,64} = 4.95, p < .05$). Average chording rate for low vision and blind participants were 23.74% (SD = 4.16) and 43.52% (SD = 22.89), respectively. As the previous studies, the most frequent chords were 'h' (82.2%) and 'l' (61.2%). Figure 2.17 displays average chording rate per session with Senorita.

2.6.6 Discussion

Session had a significant effect on entry speed and chording rate. Besides, average entry speed and chording rate over session correlated well ($R^2 = 0.9913$ and 0.9355, respectively) with the power law of practice [167]. These and the fact that learning occurred even in the last session (Figure 2.15, 2.17) suggest that Senorita did not reach its highest possible speed in the study, and likely to get much faster with practice. Note that we did not observe a significant effect of session or block on chording rate in the prior studies. Visually impaired users yielded a 62% higher chording rate than the previous studies although they were not forced to use chords. They received auditory feedback for sequential input when they pressed on one key and ran their thumb across the other keys. But this process was time-consuming since as novices they had to hear all letters. Sighted users, in contrast, could tap one key, then visually scan through the other keys for the intended letter, which was relatively faster.

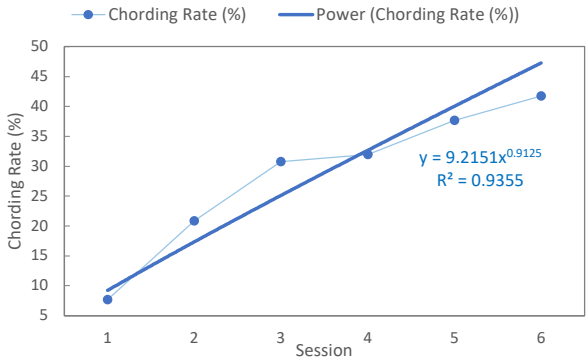


Figure 2.17: Average eyes-free chording rate per session with Senorita, fitted to a power trendline.

Hence, the absence of sight made the sequential approach much slower for the blind users than the sighted users, incentivizing blind users to learn the chords faster since it was the only viable alternative for them to improve entry speed. There was a significant effect of level-of-sight. Blind users performed 45% more chords than low vision users. There was also a significant effect of session on error rate, while there was none with sighted participants. Error rate reduced by 28% in the last session, compared to the first. This is most likely due to the difficulty in locating the letters in the beginning. It is possible that over time visually impaired users will reach an error rate comparable to the sighted users.

Entry speed with Qwerty was significantly different for low vision and blind users, 14.73 wpm and 2.52 wpm, respectively. We speculate, this is because low vision users had some light-perception that aided them in typing to some extent. In Figure 2.18, one can see that low vision participants yielded a wide range of speed: 5.4–27.8 wpm. We tried to find relationships between age, expertise, and speed but failed to find a clear pattern. It is possible that low vision users had different levels of light-perception, which influenced speed. However, we could not explore this as participants were unable to articulate their levels of light-perception. It is troubling that most yielded < 10 wpm with Qwerty after years of practice. The three blind users struggled even more: they yielded < 3 wpm, two had years of practice and the other was a novice (Figure 2.18). This implies that it is unlikely that their Qwerty performance will improve over time. The fact that they all surpassed their Qwerty speed with Senorita (32% faster) suggests that it can be an effective method for them to input text on mobile devices.

Senorita performed well compared to most braille-based methods (Table 2.1). BrailleTouch [172] reported a 9.4–23.2 wpm, but with users who were already familiar with its physical counterpart. It is unknown how it will perform with users who had never used the layout before. TypeInBraille[126] reported a 6.3 wpm, but is difficult to learn. A user who practiced it almost daily for two months reached only 10 wpm [126]. One benefit of Senorita

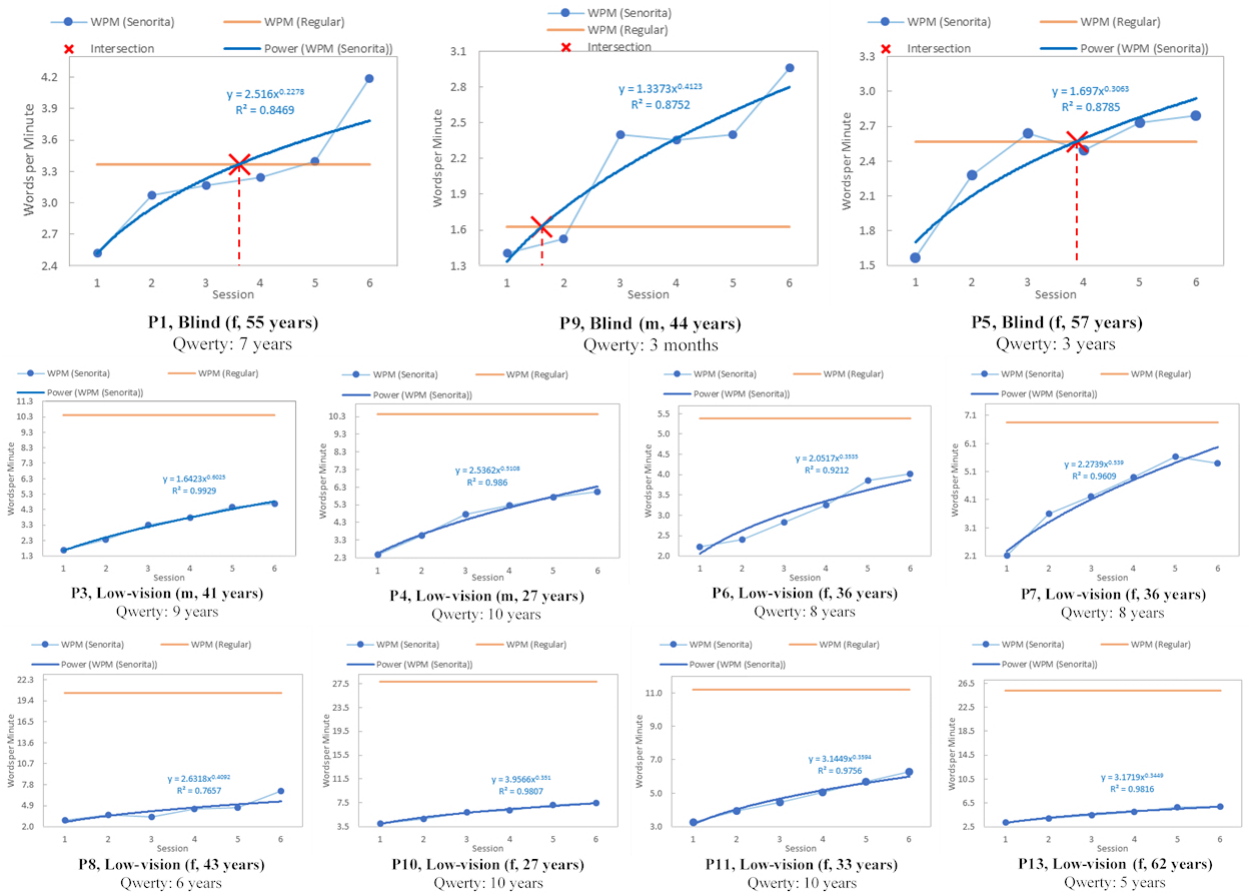


Figure 2.18: Average entry speed (wpm) with Senorita for all participants in all sessions compared to Qwerty.

is that it does not rely on the knowledge of braille, thus accessible to a larger audience. Other methods reported less than 2 wpm [65, 94, 135]. Methods that reported 10-15 wpm [20, 95, 94, 188] were evaluated with sighted people, thus not reliable, and/or rely on linguistic models to decode input, making the entry out-of-vocabulary words difficult. Senorita's speed could also be improved by using a predictive system.

Participant responses were very positive in the post-study interview. They all found Senorita easy to learn and use, and most (N = 10) wanted to use it on mobile devices frequently. Blind users praised the faster speed of Senorita. P1: "*Speed is faster especially when I know where the letter.*" P5: "*It's better [than Qwerty], over time I will be more accurate and faster.*" P5: "*With that keyboard, I will text all the time, but with iPhone's keyboard I don't like texting.*" Although, low vision users did not perform as well, they were thrilled by the speed and accuracy achieved in such a short period. P11: "*I really liked*

it, it's innovative. I look forward to use it every day." P3: "*It's very accurate, because of the fat finger it is difficult with regular keyboard, usually I have to delete a lot.*" Some even felt that they were faster with Senorita, while in reality they were not. P11: "*I am faster with chorded keyboard than with Qwerty. Accuracy is the same.*" P13: "*I like chorded keyboard better in terms of speed.*" Some found the keyboard playful and fun. P6: "*It's fun, like a memory game*!" P13: "*It's playful.*" P5, P6: "*It was fun*!" All participants said they would recommend Senorita to their friends and family, particularly to those who are visually impaired. There were also some suggestions for improving Senorita, such as using a predictive system (P11) and providing haptic feedback (P13).

2.7 Conclusion

In this Chapter, we presented Senorita, a novel two-thumb chorded keyboard aimed at both sighted and visually impaired mobile users. It yielded a 14 wpm on a smartphone by the tenth session and 9.3 wpm on a tablet in a single session. In the final longitudinal study, blind users surpassed their Qwerty performance with Senorita (32% faster), while low vision participants yielded a 5.8 wpm. Besides, visually impaired users found it effective, playful, and wanted to keep using it on mobile devices. In the next Chapter, we discuss a novel text entry method that enables users with limited dexterity to enter text on smartwatches with only one finger using the crown of a smartwatch.

Chapter 3

Crown-Based Keyboard: A One-Finger Crown-Based Smartwatch Keyboard for Users with Limited Motor Skills

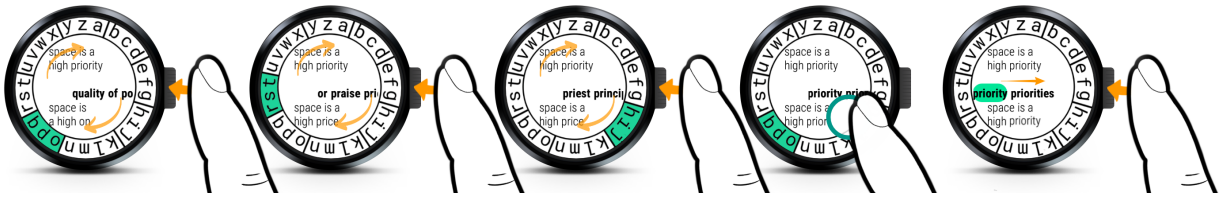


Figure 3.1: The crown-based keyboard arranges eight keys in alphabetical order at the edge of a smartwatch. The keys are automatically highlighted one by one in a clockwise direction. Once the key with the desired letter is highlighted, the user presses on the crown to enter the word. A statistical decoder is employed to present the most probable words corresponding to the entered key sequence; the most probable word among those is automatically added into the transcribed text area. The left-to-right swipe anywhere on the screen confirms the most probable prediction. While a suggestion bar represents the remaining words. To select a word from the suggestion bar, the user needs to tap anywhere on the screen, which starts to highlight the predictions one after another in sequential order. Pressing on the digital crown replaces the last entered word with the word from the suggestion bar. This figure depicts the process of entering the word "*priority*."

3.1 Introduction

There are many examples of limited dexterity causes that, in turn, need the development of unique technologies to assist such people in entering text on mobile devices. For example, stroke survivors often suffer from the upper body motor impairments such as reduced sensation, dexterity loss, and muscle weakness. Patients with cerebral palsy, Parkinson's disease, multiple sclerosis, and essential tremor also find it hard to manipulate their hands precisely and suffer from severe dexterity loss. Thus, it is essential to design a text entry technique that would fulfill and satisfy their needs, given the motor control loss. Such different needs can be stable means of interaction; avoidance of gestures, touch, or multi-touch that require high precision; and exploiting edges or bezel of the mobile devices [122]. In order to fully understand the needs, it is essential to gather the feedback from the target audience. For example, many might fall into a misguided belief that smartphones are efficient for people with limited dexterity as for able-bodied people. However, one of routine tasks such as pulling a smartphone from the pocket, can be difficult for people with upper body motor impairments [133]. Most of them agree that holding still a smartphone in their palm is the most challenging task.

Smartwatches can be an intermediary between people with limited dexterity and their smartphones, as smartwatches provide easy access to most of the smartphone's functionality. Smartwatches resolve the smartphones' accessibility problem as it is affixed on the user's hand and thus do not require to be held in the palm. However, the absence of an accessible text entry method on the smartwatches hinders the advantages that these devices present for a wider range of people. This limits smartwatch interaction to using them as fitness trackers to record daily physical activity or as an extra screen to read various notifications. Notably, the existing research for people with motor impairments primarily focuses on the health-tracking capabilities of the smartwatches [108, 78, 145] and leaves out the text entry aspect of smartwatch interactions.

A specific challenge for smartwatches is the lack of screen space, which makes interaction difficult or even impossible for people with upper body motor impairments. Most of the existing text entry methods on the smartwatches rely on precise tapping [178, 58, 137, 74, 35] or drawing gestures on small interactive space [39, 58], which are not compliant with the guidelines for accessible smartwatch interactions [122, 50]. Furthermore, the prior research has demonstrated the inaccessibility of touch interaction for people with motor impairments [34, 50, 81, 131, 132]. The possible solution for non-touch interaction on smartwatches is voice input (e.g., [130, 79]). However, text correction with speech dictation is difficult, error-prone [133], and generally inefficient in public places due to social and privacy concerns [46]. In the case of people with motor impairments, some of them might not be able to communicate using speech [115].

To address these issues, we created a new crown-based keyboard keyboard, which allows people with limited dexterity or range of motion and non-disabled people to experience text entry on the smartwatches without additional tools. The crown-based keyboard is a circular keyboard where text entry is mostly performed using only one physical button —

the digital crown located on the side of the smartwatch. The presence of tactile feedback, such as the crown, is crucial for people with upper body motor impairments: it serves as a physical reference for positioning the finger. Text entry on crown-based keyboard is primarily non-touch, and only the mode switch and space are assigned to touch inputs. This has a significant advantage as users can tap or swipe anywhere on the screen, meaning that touch target size is big enough for the interaction [50, 176]. Finally, a circular layout saves screen space, which is crucial for small devices, this space can be used for extra information or other tasks.

In this Chapter, we first review the existing works in the field. We then present the optimized new keyboard and discuss its design decisions and the optimization process. We then present the results of the series of user studies evaluating the keyboard. Finally, we conclude with potential future extensions of the crown-based keyboard.

3.2 Related Work

Due to a limited amount of literature on the intersection of the fields of text entry for smartwatches and text entry for people with upper body motor impairments, we give a comprehensive literature review on both of these fields separately. This section first reviews the most common input and interaction techniques for smartwatches and then reviews the text entry methods for people with motor impairments on desktops.

3.2.1 Input and Interaction on Smartwatches

There has been rigorous research in accessibility of touchscreens for people with motor impairments [34, 43, 66, 67, 81, 176, 7, 129, 133] and in improving touch accuracy [131, 132] on smartphones and interactive tables. However, only a few works addressed this issue on smartwatches. A notable exception is the work of Malu et al. [122], where authors evaluated different interactions on the smartwatches for people with upper body motor impairments; and the follow-up work from the same group [123] where authors compared the touch input and bezel input on the smartwatches for people with upper body motor impairments. We want to note that all methods presented in this subsection were designed for and evaluated with non-disabled people.

3.2.1.1 Miniature Qwerty Layout

Many smartwatch keyboards directly employ the standard virtual QWERTY keyboard that is resized to fit into the smartwatch's small screen. To make the text entry on the minaturized keyboard easier, one approach is to repetitively zoom, magnify, swipe over, or drag regions of the keyboard to allow precise key selection [137, 74, 35, 98, 165]. To overcome the "fat-finger problem" [180] exacerbated by multi-step interactions on miniature QWERTY, other methods propose to reduce number of keys by grouping the letters and use disambiguation

techniques [180, 75, 148]. Additional external hardware to identify fingers [68] or built-in pressure sensors [75] can be used to enhance the techniques. Another category of methods directly uses miniature QWERTY but rely on powerful language models to correct the errors [178, 58, 195].

3.2.1.2 Ring-Shaped Layouts

Apart from conventional keyboards that adopt miniature QWERTY layout, there are keyboards that use non-traditional circular layouts. These keyboards arrange the letters around the bezel in alphabetical [147, 59, 57] or QWERTY-inspired way [39]. The text entry is performed by tapping/swiping (sometimes several times) over each key belonging to an individual letter [39, 174] or multiple letters (employing disambiguation techniques) (see Chapter 4), or by other means like the rotation of the watch's bezel [196] or reading the wrist movements [57].

3.2.1.3 Summary

All methods presented here were not designed for or evaluated with people with motor impairments. The methods, that use miniature QWERTY layout, occupy a large area of the screen real-estate due to the form-factor of the QWERTY and, at the same time, do not ease the key selection due to reliance on high precision or multiple action sequences to enter a single letter. Thus, all these make typing on smartwatches difficult or even impossible for people with upper body motor impairments [122]. While ring-shaped techniques mentioned above require multiple actions, dragging finger all over the screen, using multiple fingers simultaneously, or precisely selecting targets on a small touchscreen area, which poses significant accessibility challenges as well [122].

3.2.2 Desktop and Phone Keyboards for People with Motor Impairments

Here we give a short overview of the text entry techniques for people with motor impairments, which we can group as ambiguous keyboards and scanning keyboards. All keyboards in this section, except for HandyGlyph [21], were developed for desktop computers and cannot be easily transferred onto smartwatches, nor have they been evaluated on them. The HandyGlyph was developed for mobile usage but requires additional external hardware (a push button) which complicates its adoption on smartwatches. Most importantly, although all methods discussed in this subsection were designed for people with motor impairments, only a few of them have been evaluated with the representative group. For a detailed review of the existing text entry methods for motor-impaired people, we refer to Polacek et al. [144].

3.2.2.1 Ambiguous Keyboards

Ambiguous techniques assign multiple characters per key, thus need a user- or software-level disambiguation process. Ambiguous keyboards reduce the number of keys, thus require fewer actions to enter a word which is a desirable feature for people with limited dexterity. Harbusch and Kühn [70] presented an ambiguous keyboard where letters are grouped into keys based on frequency information. Tanaka-Ishii et al. [173] presented a keyboard designed with the alphabetical order of the letters.

3.2.2.2 Scanning Keyboards

In the scanning keyboards, the keys are highlighted in sequential order, and the desired key is selected with a single switch. These methods were primarily designed for linear scanning, and most of them support ambiguous input as well. Other techniques employ group scanning, where the first scan happens over larger groups of keys, and once the group is selected the scanning is repeated to select the exact key within a group. Due to requiring only a single switch, scanning keyboards are the most common text entry technique for people with motor impairments [144].

Linear scanning Several ambiguous keyboards use linear scanning. HandyGlyph [21] has three ambiguous keys where each key contains a set of characters composed of a similar shape. SAK [114, 115] is an automatic scanning keyboard with three ambiguous keys in alphabetical order. Qanti [48], and BlinkWrite2 [17] use the keyboard similar to SAK: Qanti uses intentional muscle contractions as input signal, while BlinkWrite2 uses eye blinks.

Group scanning Lin et al. [106, 107] proposed the scanning keyboard, divided into nine groups, with inside characters arranged based on their frequency. 3DScan [49] is a group-row-column scanning keyboard, where users first need to select the group, which invokes row-column scanning within that group, and then select the desired character. Prabhu and Prasad [146] proposed the circular scanning keyboard, where multiple letters were grouped into eight sub circles.

3.2.2.3 Summary

People with severe motor impairments are usually constrained to only single switch interaction. Scanning allows to highlight the options in a sequential order where the selection can be performed using an only single switch. The discussed methods rely on different scanning techniques, such as linear and group scanning. Based on the literature, scanning ambiguous text entry methods with static layout perform better for people with motor impairments [144]. However, most of the keyboards presented above were designed for desktop computers, thus it is unclear how they could be adapted to smartwatches. Besides, only a few of them evaluated the techniques with the representative users.

Table 3.1: Average text entry speed of ambiguous smartwatch keyboards that map multiple letters to each key.

Method	Letters per Key	Entry Speed
Yi et al. [196]	3	9–13 wpm
Jiang and Weng [82]	4–5	9.6–11 wpm
Gong et al. [57]	4–5	10 wpm
Dunlop et al.[45], Komninos and Dunlop [91]	3–6	8 wpm

3.3 The Crown-Based Keyboard: Design and Optimization

The design goal of the crown-based keyboard was to strike the right balance between usability, learnability, and performance such as entry speed and accuracy. Assigning all letters to one key makes interaction easy and fast but hurts the disambiguation ability of predictive models [114]. On the other hand, assigning dedicated keys to each letter eliminates the need for disambiguation but increases the scanning time and affects entry speed and accuracy. Therefore, we adapted a systemic approach to decide how many letters each key must contain, which letters to group into each key, and the most effective rotation interval.

3.3.1 Key Size

To identify the optimal number(s) of letters per key, we conducted a literature review of ambiguous smartwatch keyboards that use linguistic models to disambiguate the input. We observed a correlation between the number of letters per key and entry speed. There seems to be an inverse relationship between them—the fewer the number of letters per key the better the speed (Table 3.1). Considering this, and that fewer number of keys makes scanning keyboards more accessible to people with motor impairment [102], we explored 3–6 letters per each key in the following steps.

3.3.2 Layout Optimization

We explored all possible layouts with 3 to 6 letters per key for the circular string {`abcdefghijklmnopqrstuvwxyz`} to find the least ambiguous alphabetical layout. One way to approach the problem is to redefine it as a search for a layout that has the minimal number of clashing bigrams for any given key within it, in other words, reduce frequent letter pairs in the keys. For example, "th" is the most frequent bigram, thus 't' and 'h' must be on different keys. Formally, let us denote the set of possible keys as L, set of all bigrams as B, first and second letter within a particular bigram as b_1 and b_2, respectively, and the frequency of a bigram

b_1b_2 as $f(b_1b_2)$. Then, the optimization goal corresponds to the following problem:

$$\min_{l \in L} \sum_{b_1b_2 \in B} f(b_1b_2)\, I(\text{key}(b_1, l) = \text{key}(b_2, l)), \tag{3.1}$$

where I is the indicator function with a value of 1 if the evaluated condition is true, and $\text{key}(x, l)$ is the key to which letter x is assigned within the layout l.

This minimization problem can be solved by a simple enumeration (brute-force search) over all configurations of the problem, with the total runtime of $O(|L||B|)$. To generate the set of possible layouts L, we split the circular string into key sizes ranging from 3 to 6 letters, which gave us a total of 27,560 different layouts. The set of bigrams B contains all 676 possible bigrams of the English language [134], we used bigrams.json file [110] as a reference for our frequencies $f(b_1b_2)$. To prevent numeric overflow, we scaled down all frequencies by multiplying them with 1.0×10^{-11}, and report the scaled down frequencies as the final scores. The layout with minimal ambiguity score (Eq. 3.1) was: {yza}{bcd}{efg}{hij}{klmn}{opq}{rst}{uvwx} with the value of 1.53, which we use in the crown-based keyboard. For reference, the layout with the worst score was: {xyzab}{uvw}{opqrst}{ijklmn}{cdefgh} with the value of 6.19.

3.3.3 The Rotation/Scanning Interval

A rotation or scanning interval is commonly used in text entry techniques for people with motor impairment [106, 107, 115] to automatically go over the keys by highlighting them until the desired key is selected by the user. The maximum possible entry speed of a scanning keyboard depends on the pace of its linear rotation (scanning interval). Slower scanning intervals can make users impatient and affect entry speed [146], while too fast scanning intervals can cause too many errors, and even prevent users from using the technique [48]. To determine the most effective rotation/scanning interval, we conducted a literature review of scanning keyboards aimed at people with motor impairments. Table 3.2 presents an excerpt of the review, where the scanning speed varies from 700 ms (fastest) to 2,100 ms (slowest). Based on the review and the findings of a focus group (discussed in Section 3.4), we decided to use a rotation/scanning interval (time to rotate from one key to another) of 1,000 ms. However, the system allows to adjust the rotation/scanning interval if necessary.

3.3.4 The Disambiguation Process

The crown-based keyboard disambiguates the input (sequences of keys) into words. When there are multiple possible words for a sequence of keys, it automatically selects the most probable one and enables users to pick a different possible word from a suggestion bar. Formally, given a key sequence s, the crown-based keyboard predicts the most probable word w from a vocabulary of N words $w_1, \ldots, w_N$ using the following equation:

$$w = \underset{w_i \in w_1, \ldots, w_N}{\arg\max}\ P(w_i|s), \tag{3.2}$$

Table 3.2: Rotation/scanning interval of scanning keyboards aimed at users with motor impairments, along with the reported entry speed. "R" signifies studies conducted with representative users (people with motor impairment), while "NR" represent non-representative users.

Method	Participant	Scanning interval	Entry Speed
Ashtiani and MacKenzie [17]	NR	700, 850, 1,000 ms	4.3, 5.3, 4.6 wpm
MacKenzie et al. [114]	NR	1,100–700 ms	4–5 wpm
Felzer et al. [48]	NR	1,000–500 ms	2–7 wpm
Belatar and Poirier [21]	R_{expert}	754–118 ms	2–3 wpm
Baljko and Tam [19]	NR	750, 1,250 ms	2.6, 1.8 wpm
Prabhu and Prasad [146]	NR	2,100, 1,800, 1,500, 1,200 ms	NA

where $P(w_i|s)$ is the conditional probability of getting word w_i given a sequence s. To compute these probabilities we apply Bayes rule, then replace probabilities by counts of the occurrences of the words/sequences in the training corpus:

$$P(w_i|s) = \frac{P(w_i, s)}{P(s)} = \frac{\text{count}(\text{prefix}(w_i) = s)}{\text{count}(s)}. \tag{3.3}$$

To efficiently compute and store a conditional probability table of $P(w_i|s)$ for all possible words w_i and sequences s, we use a binary prefix tree, also known as Trie data structure. Once the tree is constructed, we trim it to contain at most $K = 10$ most probable words for each sequence s and save it to run on the smartwatch. It is a unigram word model, thus does not account for previously typed words. To address this, we accompany the model with a bigram model that predicts the most probable word w_i, given a sequence of the keys s and the previously typed word w_k. This requires computing probabilities $P(w_i|s, w_k)$ by a straightforward extension of Eq. (3.3). Despite extending the model for bigram probabilities, it remains simplistic in nature, which is necessary to account for the limited processing power of smartwatches. Strengthening the decoder by conditioning on more than one previously typed words can potentially improve the performance of the keyboard. Employing recent machine learning models that can natively handle sequence-level information (such as, LSTMs and Transformers) and training them on a substantial amount of data could help. However, designing and training such models is challenging and outside the scope of this work.

3.3.5 Error Correction and Special Characters

The crown-based keyboard appends a *space* when a word is selected by the keyboard or by the user from the suggestion bar. Selection from the suggestion bar is confirmed using a crown press. On the other hand, to confirm the selection of the most probable word, which

appears in the transcribed text area (and not in the suggestion bar) users need to swipe from left to right anywhere on the screen. To delete the last entered word, users need to long press anywhere on the screen; such an interaction was used previously in some keyboards as well [114].

Current prototype of the crown-based keyboard is in English and does not provide an access for uppercase letters, numbers, special symbols. However, these features could be implemented by adding double tap, or swiping gestures to switch between different layouts for digits and symbols. Besides, it is a common practice to evaluate novel text entry methods without enabling numeric and special characters since it eliminates a potential confound [118].

3.4 User Study 1: Focus Group Discussion

We conducted a focus group with people with limited fine motor skills to find out whether there is a need for text entry on smartwatches in the community. We also discussed the challenges they face in entering text on mobile devices, and gathered their opinion about three existing text entry techniques for smartwatches. Before getting started with the design and development of the smartwatch keyboard, we collected feedback and expectations from target audience.

3.4.1 Participants

Initially, we recruited six participants through local disability centers, however, three of them were unable to participate due to health related issues, resulting in three participants in the focus group. Table 3.3 presents their demographic information. They all used various accessibility tools to access mobile and other computer devices. One of them owned a smartwatch. Each participant received a US $20 Amazon gift card for volunteering.

Table 3.3: Demographics of the focus group participants.

Participant ID	Gender	Age	Highest Degree	English Proficiency	Level of Impairment	Cause
P1	Woman	31	University	Native	Moderate	Cerebral palsy
P2	Man	48	University	Native	Severe	Delayed development
P3	Woman	37	University	Native	Severe	Multiple sclerosis

3.4.2 Procedure

Due to the spread of COVID-19 virus, the focus group discussion was conducted via a teleconference application. We shared the digital informed consent form and the demographic questionnaire with potential volunteers ahead of time for them to learn about the research. Participants completed and signed all forms electronically. During the discussion, all participants were addressed with pseudo names (e.g., Mr. A) to protect their anonymity. Once the teleconference session started, a researcher explained the procedure again and answered all questions participants had. The discussion started after that. First, the researcher asked general questions about smartphone and smartwatch usage, followed by a discussion on the challenges participants face when using these devices. Then the researcher demonstrated three existing text entry techniques for smartwatches (Fig. 3.2). The techniques were default QWERTY method called Gboard, WatchWriter [58], and SwipeRing (see Chapter 4). With default QWERTY Gboard in order to enter text users either tap or gesture type. WatchWriter [58] is a smartwatch keyboard that enables touch and gesture typing, as well. While with SwipeRing (see Chapter 4) users need to connect the zones or keys that contain the target letters by drawing gesture on the screen. After, the researcher demonstrated proposed techniques and discussed the design and interaction features with participants. During the focus group session, the researcher demonstrated all methods and their interactions step by step using videos, images, and by showing live demonstrations. The researcher recorded a complete study session.

Figure 3.2: Existing text entry methods for smartwatches. The user enters the text using miniature versions of the standard QWERTY such as (a) default QWERTY method, where to enter text, the user needs to tap on the desired key or perform gesture to enter word, (b) WatchWriter, where the user can enter text either by touch or gesture typing, and a novel keyboard called (c) SwipeRing, which arranges the QWERTY layout around the bezel where the user enters text by gesture typing.

3.4.3 Findings

3.4.3.1 Daily Activities on Mobile Devices

Participants used their smartphones for a variety of tasks, but primarily for writing and sending text messages, checking emails, surfing the internet, and making phone calls.

Participant (P1), who owned an Apple smartwatch, uses it as a substitution to a smartphone, she commented , "*I had been using my smartwatch basically as my cell phone because the way my phone is set up. In a public place, it's hard for me to answer my regular phone because it's on a lanyard [. . .].*" It is easier for the participant to answer with the smartwatch because the smartwatch is always placed on the wrist and helps prevent situational impairments, such as pulling a smartphone from the pocket or detaching it from the lanyard, which may increase the possibility of dropping the smartphone. She has unsatisfactory experience in typing on the smartwatch using speech to text feature.

Other participants showed interest and tried using the smartwatch at the store but wanted to explore it more, especially the features, before purchasing it, since they were not sure if the smartwatches would be useful for them. If there will be an efficient keyboard on the smartwatch, given the physical configuration and constraints of the smartwatch, they were willing to try it.

3.4.3.2 Accessibility Challenges Existing on Mobile Devices

Participant (P2) mentioned that some tasks especially involving twisting, and twirling fingers are hard to complete on a smartphone since these gestures are impossible to perform; this forces him to use a laptop to complete some particular tasks. All participants agreed that it is impossible for them to use the voiceover rotor function since it requires twirling both fingers like twisting a doorknob.

Participants (P1, P3) said that enlarging the text and pinching is hard on the smartphone since it requires two fingers and it is hard to control them, furthermore, they can hit the wrong item, which is frustrating. The aforementioned issues prevent Participant (P3) from performing some tasks on the phone, she commented "*I usually use my phone, but like if I need to fill out forms, I tend to use a computer [. . .] because computer is less responsive.*"

3.4.3.3 Perceived Usability and Design Preference

During the demonstration of the existing keyboards on the smartwatch (Fig. 3.2), none of the participants were aware of such keyboards, even Participant (P1), who owns the smartwatch.

Alphabetical layout vs. Qwerty All participants preferred the alphabetical layout over the QWERTY. Participant (P2) commented, "*I think the* QWERTY *keyboard shape fits with the idea of [. . .] the computer keyboard [. . .], but we're talking about a watch and I think the idea of the letters going around alphabetically might be more intuitive.*" Participant (P3)

mentioned that it might be hard to learn QWERTY keyboard due to the learning curve and preferred alphabetical layout since it is easier to locate the letters.

Grouping vs. Separate Letters All participants preferred the idea of grouping letters since it is faster. Participant (P3) mentioned that prediction makes typing easier as well. Participant (P2) commented "*I think grouping letters would actually make the process faster.*"

Physical Button vs. Touch All participants preferred the physical button since once it is pressed, one can feel push back and the actuation force on a keypress. During the interaction, the finger covers the screen for the touch action, while the physical button does not. Participant (P3) mentioned that the button is the more stable mean of interaction, "*I like the physical button because if I'm having shaky hands that day, I don't have to try to force my finger to stay on one specific spot versus pressing a button with thumb.*"

Crown vs. Bezel All participants preferred the crown over the bezel. Participant (P1) commented, "*I couldn't do the bezel, I could use only buttons.*" and that she cannot use two fingers reliably. Participants (P3) mentioned that although the bezel is larger than the crown, the crown seems simple and straightforward and could be easily rotated with one finger, she commented "*I mean the Crown seems the simplest since I don't have to like get my whole hand involved [. . .] [Bezel] requires too much coordination.*"

Automated vs. Manual Participants were indecisive about automated and manual rotation but were willing to try both. Participant (P2) commented "*I might be able to rotate it [crown] with one finger, but I haven't tried it, just depends . . .*" For the automated crown-based keyboard participants wanted to make the rotational speed adjustable, so people can change the speed as they want. Since each person has her own preferences and medical condition, even one person might perform one task at a different rate throughout the day.

Clockwise vs. Counterclockwise vs. Shortest Path (most probable direction) In terms of rotation direction, participants were hesitant about it as well, they showed a willingness to try possible options. Participant (P2) liked the idea of the shortest path but was not sure since it mostly depends on how it was implemented, *"I like the idea of the most probable letter, or I could hate it. [. . .] [Shortest Path] could end up being a disaster if your prediction of what the most likely letter is wrong"*. Participant (P3) wanted to try clockwise and the shortest path, "*[. . .] clockwise is more natural for me, because you know that there actually is going to be always the same, and we can anticipate [the direction]. But I mean I would love to have the option to choose.*"

Other suggestions All participants wanted to be able to dictate OOV (out-of-vocabulary) words. All participants wanted to have a dictation option since they use it a lot. However,

Participant (P2) said that it is hard to dictate the text using any Bluetooth headset, such as air pods, because they do not work well.

Participants also mentioned that they would not enter private information in public places; Participant (P2) will wait until he goes home to complete some private or complex tasks and chooses not to do the task now. Participant (P3) mentioned that she does not want to use dictation at work since she does not want others to hear the conversation. To sum up, participants felt fine using dictation for small talk conversations, but not for private information such as dictating an address, credit card information and etc.

3.5 Discussion

3.5.1 Rotation Possibilities

The findings of the focus group study (sec. 3.4.3) indicate that majority of the representative group preferred automated rotation due to a less physical demand such selection mechanism requiring. However, one focus group participant mentioned that he might be able to rotate the digital crown with one finger. Since all rotation versions were viable, we decided to explore both manual and automated rotations. Respectively, we developed two versions of the crown-based keyboard: Automated and Manual. The Automated crown-based keyboard uses a single physical button and does not require any rotation by hand. In contrast, the Manual crown-based keyboard provides more freedom, since users may decide the rotation direction. Except for the rotation part, both versions have the same layout and interaction design. We compare the performances of Automated and Manual crown-based keyboards in section 3.6, and in the remainder of this section, we discuss the interactions arising in these keyboards in more detail.

3.5.1.1 Automated Crown-Based Keyboard

Automated crown-based keyboard supports single button interaction, where selectable keys are highlighted (scanned) in a circular sequential order in a clockwise direction. The rotation/scanning time is 1000 ms, which was chosen based on literature review (sec. 3.3.3) and focus group discussion (sec. 3.4.3). To enter a word sequence, the users press on the digital crown once the key with a desired letter is highlighted. The statistical decoder disambiguates the current sequence of inputs and 1) shows the most probable word in the transcribed text area and 2) shows the rest of the predictions in the suggestion bar. If the desired word is the one already shown in the transcribed text area (i.e., the most probable prediction), users enter *space* with swiping gesture from left to right anywhere on the screen and finish the entry. Otherwise, users might continue the entry of the desired word or might want to select a word from the suggestion bar. To switch to the suggestion bar, the users need to tap anywhere on the screen, which starts highlighting/scanning the predictions in the suggestion bar (instead of the keys in the keyboard) in sequential order. Pressing on the digital crown replaces the last entered word with the highlighted word from the suggestion bar and finishes

Figure 3.3: The process of entering the word “drink” with the manual crown-based keyboard. The user rotates the crown upward to rotate over the zones in clockwise direction, then pushes on the crown to select the letter ‘d’. Then, the user rotates the crown downward to enable counterclockwise rotation and select the letter ‘r’. The user then rotates towards the letter ‘i’ by rotating the crown downward. The users taps anywhere on the screen to switch to the predictions, then rotates the crown upward to highlight the desired word. Once it is highlighted, the user selects the desired word “drink” by pushing the crown.

the entry. If no suitable word were found in the suggestion bar, users tap anywhere on the screen to return back to continue word entry by selecting keys (Fig. 3.1).

3.5.1.2 Manual Crown-Based Keyboard

With Manual crown-based keyboard users have to rotate the digital crown on the side of the smartwatch using the finger; the upward rotation of the crown enables clockwise rotation for key selection, and downward rotation of the crown enables counterclockwise rotation. The interaction design is almost the same as in Automated crown-based keyboard. To enter a word sequence, users need to reach the desired key (highlighted with color) by rotating the crown and then press on the crown to select this key. A statistical decoder disambiguates the current sequence of inputs and enters the most probable word into the transcribed text area and the rest of the predictions into the suggestion bar. The most probable word can be confirmed with a left-to-right swipe; otherwise, users might continue selecting keys or switch to the suggestion bar by tapping anywhere on the screen. Once in the suggestion bar, words can be highlighted by manually rotating the crown; the highlighted word can be confirmed by pressing the crown (which will finish the entry). If no suitable word were found in the suggestion bar, users tap anywhere on the screen to return back to continue word entry by selecting keys (Fig. 3.3).

3.6 User Study 2: Manual vs. Automated Crown-Based Keyboard

We compared manual and automated crown-based keyboards in a between-subjects user study involving people without motor impairments.

3.6.1 Participants

Sixteen participants voluntarily took part in the study. We divided them randomly into two groups: manual and automated. Table 3.4 presents demographic information of these groups. None of them reported to have a condition limiting their fine motor skills. Each participant received US $15 for volunteering.

Table 3.4: Demographics of the two user groups (User Study 2).

	Manual	Automated
Age	M = 27.5 years (SD = 3.7)	M = 28.9 years (SD = 2.7)
Gender	1 female, 7 male	3 female, 5 male
Handedness	7 right, 1 left	7 right, 1 ambidextrous
Experience with mobile devices	M = 10.3 years (SD = 1.2)	M = 10.3 years (SD = 3.1)
Experience with smartwatches	M = 1.30 years (SD = 1.7)	M = 0.8 years (SD = 1.1)

3.6.2 Apparatus

We used an LG Watch Style smartwatch, 42.3×45.7×10.8 mm, 9.3 cm^2 circular display, 46 grams, running on the Wear OS at 360×360 pixels in the study. We decided to use a circular watch in the study since it is the most popular shape for (smart)watches [84, 89]. We developed both versions of crown-based keyboard with the Android Studio 4.0, SDK 26. Both versions calculated all performance metrics directly and logged all interactions with timestamps.

3.6.3 Design

The study used a mixed-design with one between-subjects independent variable: method (two conditions: manual, automated) and one within-subjects independent variable: block (five blocks). We decided to use a mixed-design to avoid interference between the conditions. Since both methods use the same layout, the skills acquired in one condition could have affected the performance of the other condition in a within-subjects design [112]. We divided the participants into two separate groups: manual and automated, with eight participants each. The groups used the technique assigned to them to enter short English phrases from the MacKenzie et al. [118] set in five blocks. Each block contained eight random unique phrases from the set. In summary, the design was:

2 groups (manual, automated) ×

8 participants ×

5 blocks ×

8 random phrases = 640 phrases in total.

The dependent variables were the following commonly used performance metrics:

- **Words per minute** (wpm) signify the total number of words entered in one minute, where a "word" is defined as five characters including letters, spaces, and other printable characters [14].
- **Error rate** (%) is the average percentage of erroneous characters remained in the final transcribed text. In other words, it is the ratio of the total number of incorrect characters in the transcribed text to the length of the transcribed text.

3.6.4 Procedure

We conducted study with only one participant at a time in the lab. First, we explained the procedure and answered to all questions participants had, after that we collected the consent forms. We then asked them to answer questions about demographic and mobile usage experience. After that, we introduced the keyboard assigned to them, instructed them to sit in front of a desk, and wear the smartwatch on the hand they prefer. However, to increase the external validity of the study, we enforced the use of only one finger for interaction with the device. All participants wore the smartwatch on their left hand, rested the arm on the table, and performed the actions using the index finger (Fig. 3.4).

We asked participants to practice with the keyboard assigned to them by transcribing 2–3 phrases from the MacKenzie et al. [118] set. These phrases were not repeated in the study. The main study started after this short practice session. There were five blocks per condition, with eight random unique phrases per block. We enforced a 2–3 minutes gap between the blocks to reduce any potential effects of fatigue. During the study for both methods, the phrases were presented one by one at the top part of the smartwatch (Fig. 3.4). Participants were asked to read a presented phrase carefully, transcribe it "*as fast and accurate as possible*," then swipe from top to bottom anywhere on the screen to see the

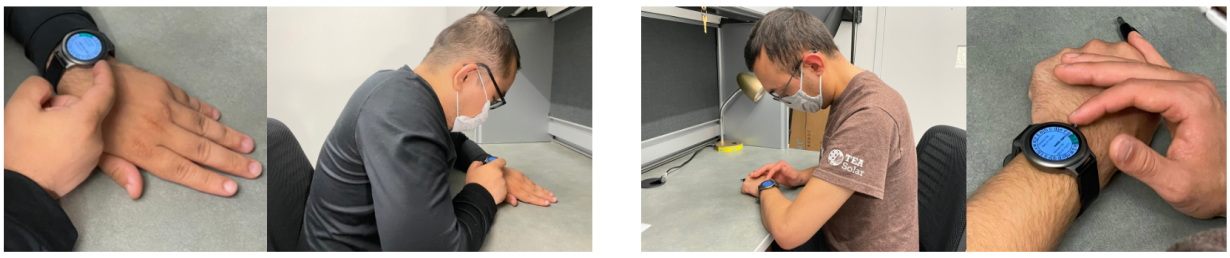


Figure 3.4: Participants entering text with the manual (left) and the automated crown-based keyboard (right) in the second user study.

next phrase. The transcribed phrase was displayed at the bottom part of the smartwatch screen. Error correction was recommended but not forced.

Upon completion, participants answered to a short post-study questionnaire about the assigned method's speed, accuracy, learnability, ease-of-use, and their willingness-to-use the method on smartwatches on a 5-point Likert scale. They also took part in a debrief session where they were asked about any potential strategies used to enhance the performance of the method assigned to them.

3.6.5 Safety Measures for COVID-19

All researchers involved in this study were fully vaccinated for COVID-19. All participants were pre-screened for COVID-19 symptoms during the recruitment process by a researcher, and on the day of the experiment by the host institute. Both the researcher and the participants wore face coverings and sanitized their hands before a study session. The researcher also maintained a 3 inches distance from the participants at all times. All study devices and furniture were disinfected before and after each study session. This protocol was reviewed and approved by the Institutional Review Board (IRB).

3.6.6 Results

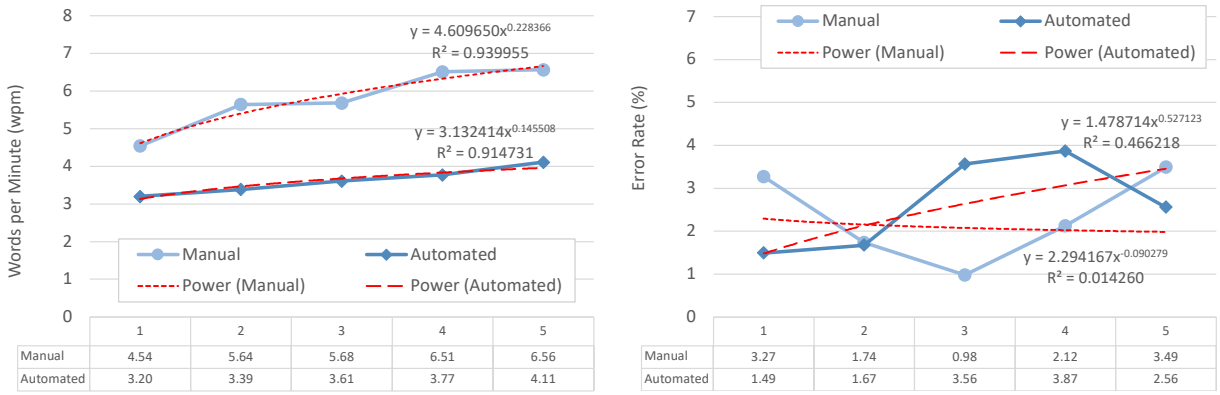
A complete study session took about 60 minutes to complete, including demonstration, questionnaires, and breaks. A Shapiro-Wilk test revealed that the response variable residuals were normally distributed. A Mauchly's test indicated that the variances of populations were equal. Hence, we used a mixed-design ANOVA for the quantitative factors. We used a Mann-Whitney U test on the between-subjects questionnaire data.

3.6.6.1 Entry Speed

An ANOVA identified a significant effect of method on entry speed ($F_{1,14} = 20.03, p < .001$). Average entry speed with manual and automated were 5.8 wpm (SD = 2.2) and 3.9 wpm (SD = 1.5), respectively. There was also a significant effect of block ($F_{4,4} = 18.55, p < .0001$). The method $\times$ block interaction effect was also statistically significant ($F_{4,56} = 3.90, p < .01$). Fig. 3.5a illustrates average entry speed per block for both methods, fitted to power trendlines.

3.6.6.2 Error Rate

An ANOVA failed to identify a significant effect of method on error rate ($F_{1,14} = 0.05, p = .82$). Average error rates with manual and automated were 2.3% (SD = 7.1) and 2.8 (SD = 9.0), respectively. There was also no significant effect of block ($F_{4,4} = 0.61, p = .65$). Fig. 3.5b illustrates average error rate per block for both methods, fitted to power trendlines.



	1	2	3	4	5
Manual	4.54	5.64	5.68	6.51	6.56
Automated	3.20	3.39	3.61	3.77	4.11

(a) Entry speed (wpm)

	1	2	3	4	5
Manual	3.27	1.74	0.98	2.12	3.49
Automated	1.49	1.67	3.56	3.87	2.56

(b) Error rate (%)

Figure 3.5: (a) Average entry speed (wpm) and (b) error rate (%) per block, with both manual and automated crown-based keyboards, fitted to power trendlines.

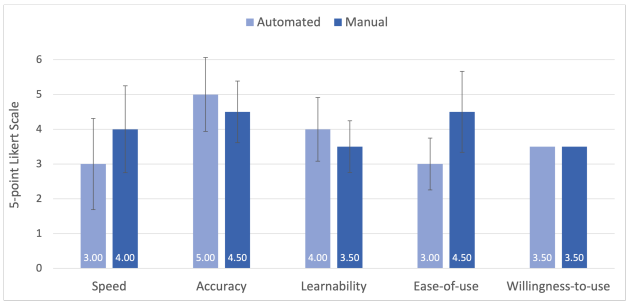


Figure 3.6: Median user ratings of the two methods on a 5-point Likert scale, where 1–5 represented disagree–agree. Error bars represent ±1 standard deviation (SD).

3.6.6.3 User Feedback

A Mann-Whitney U test failed to identify a significant effect of method on perceived speed ($U = 20.0, Z = -1.30, p = .19$), accuracy ($U = 28.0, Z = -0.47, p = .72$), learnability ($U = 28.0, Z = -0.49, p = .72$), ease-of-use ($U = 17.0, Z = -1.68, p = .13$), or willingness-to-use ($U = 26.0, Z = -0.66, p = .57$). Fig. 3.6 illustrates median user ratings of the two methods.

3.6.7 Discussion

The manual crown-based keyboard was significantly faster than the automated crown-based keyboard (60% faster). This is not surprising as the 1,000 ms rotation interval adds to

the total time needed to select a zone [144]. Since the participants did not have a motor impairment, they could easily rotate the crown at a desired speed and strategize rotation direction to improve entry speed. In the post-study debrief session, six out of eight participants of the manual group reported that they intentionally switched the crown's rotation direction to reach the desired zone faster. If the intended zone was closer from the left (i.e., fewer number of zones between the current and the intended zone), they rotated the crown counterclockwise, and vice versa. There was a significant effect of block on entry speed. Learning occurred with both methods. Average entry speed over block correlated well with the power law of practice [167] for both manual ($R^2 = 0.94$) and automated ($R^2 = 0.91$). However, participants' entry speed improved at a much faster rate with manual than automated (Fig. 3.5a). Entry speed with manual improved by 45% from the first block to the last, while the same with automated improved by 28%. A post hoc Tukey-Kramer Multiple-Comparison test identified three significantly different groups of blocks in manual: {1}, {2, 3}, {4, 5} and in automated: {1}, {2, 3, 4}, {5}. The fact that entry speed with automated improved by 10% from the fourth block to the last (Fig. 3.5a) suggests that the performance of this method is likely to improve further with practice.

The methods were comparable in terms of accuracy. We did not observe learning between the blocks, instead, error rates were rather erratic (Fig. 3.5b), which is not unusual for word-based text entry methods [33]. When transcription errors did occur, participants misspelled entire words due to the selection of an incorrect zone or an incorrect word from the suggestion bar, or omitted them entirely.

Subjective data revealed that almost all participants found the method assigned to them fast, accurate, easy-to-learn, easy-to-use, and wanted to use it on their smartwatches. The remaining participants were neutral. This is surprising the slower entry speed of the method compared to state-of-the-art text entry techniques for smartwatches [11].

3.7 The Shortest Path Crown-Based Keyboard

Results of the second user study revealed that non-disabled people perform much better with the manual crown-based keyboard than the automated version. Yet, we intended to improve the performance of the automated version based on the focus group discussion with people with limited dexterity (Section 3.4.3) that revealed a desire for an automated version since rotating the crown could be difficult at times.

Based on the findings that users tend to strategize the crown rotation direction (that they intentionally switch the crown's rotation direction to reach the desired zone faster), we implemented a shortest path version of the automated crown-based keyboard (Fig. 3.7). We give the full implementation details of this version of the crown-based keyboard in Algorithm 1 and discuss it next.

When users select the current zone z_p by pushing the crown, Alg. 1 guesses in several steps the rotation direction in which it must highlight (clockwise or counterclockwise) to enable faster zone selection for the user. First, we find the next probable letters (l_1 and l_2 in

Algorithm 1: Shortest Path Crown-Based Keyboard rotation direction

Input: Previous typed word w, currently typed sequence s, last used zone z_p
Function `ShortestPathCrownBasedKeyboard`(w, p, z_p)**:**
  $l_1 = \text{argmax}_{l \in \text{letters}} P(\text{letter} = l|w, s)$
  $l_2 = \text{argmax}_{l \in \text{letters}} P(\text{letter} = l|w, s)$ s.t. $P(\text{letter} = l|w, s) < P(\text{letter} = l_1|w, s)$
  if $P(letter = l_1|w, s) > P(letter = l_2|w, s) + 0.1$ *or* $zone(l_1) = zone(l_2)$ **then**
    return direction(zone(l_1), z_p)
  else
    $P(\text{zone} = z|w, s) := \sum_{l \in z} P(\text{letter} = l|w, s)$
    $z_1 = \text{argmax}_{z \in \text{zones}} P(\text{zone} = z|w, s)$
    $z_2 = \text{argmax}_{z \in \text{zones}} P(\text{zone} = z|w, s)$ s.t. $P(\text{zone} = z|w, s) < P(\text{zone} = z_1|w, s)$
    if $P(zone = z_1|w, s) > P(zone = z_2|w, s) + 0.1$ *or* $z_1 = z_2$ **then**
      return direction(z_1, z_p)
    else
      $P_{cc} = \sum_{z \in \text{left of } z_p} P(\text{zone} = z|w, s)$
      if $P_{left} > 0.5$ **then**
        return `counterclockwise`
      else
        return `clockwise`
      end
    end
  end

Algorithm 2: Direction indicating shortest path between two zones

Input: End zone z, starting zone z_p
Function `direction`(z, z_p)**:**
  if $z = z_p$ *or* $|z - z_p| = 4$ **then**
    return `no preference`, keep current direction
  else if $z_p > z$ *and* $z_p - z \leq 3$ *or* $z_p < z$ *and* $z - z_p > 3$ **then**
    return `counterclockwise`
  else
    return `clockwise`
  end

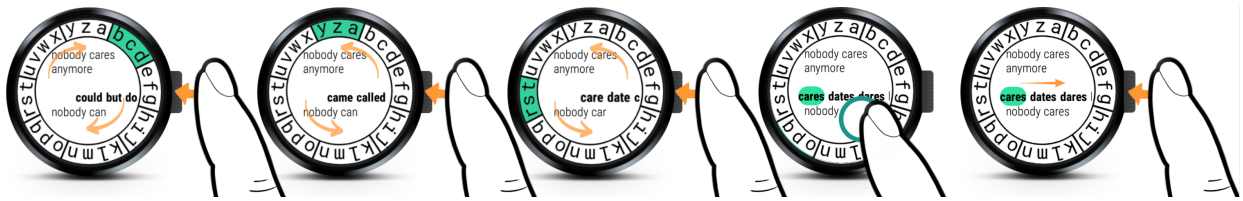


Figure 3.7: The process of entering the word "cares" with the shortest path crown-based keyboard. Upon start, the keyboard highlights the zones clockwise. When 'c' is selected by pushing the crown, the keyboard switches rotation to counterclockwise since the most probable zones are on that side (combined probability: 0.9432). As the zone containing 'a' is selected, the keyboard continues highlighting the zones counterclockwise since the next most probable letter 'n' (probability: 0.9432) is closest when rotating in that direction. This process continues until a word is confirmed by the user.

Alg. 1) which are calculated using the bigram model described in Section 3.3.4. If the most probable letter l_1 has significant probability (greater than P(l_2) by at least 0.1 score), we choose the shortest direction towards the zone that contains the letter l_1. This is denoted as direction(zone(l_1), z_p). Otherwise, we make the decision based on the probability of zones, rather than letters. We find most probable zones z_1 and z_2 where probability of zone is defined as the combined probability of letters in the zone. If the most probable zone z_1 has significant probability (greater than P(z_2) by at least 0.1 score) we then choose the shortest direction towards the zone z_1. Otherwise, we choose the side that has the most probability (total sum of probabilities of zones in that side). We denote the probability of the left side as P_{left}, and if $P_{\text{left}} > 0.5$ we choose counterclockwise rotation.

Finally, we discuss the Alg. 2. It computes the shortest path direction between current zone z_p and target zone z. It is achieved by counting the total number of zones between z and z_p in both directions, and choosing the direction that has the least amount of rotation steps.

3.8 User Study 3: Clockwise vs. Shortest Path Crown-Based Keyboard

The purpose of this study was to investigate whether the shortest path version of the crown-based keyboard improves performance in terms of entry speed and accuracy.

3.8.1 Participants

We had twelve participants in the study. None of them participated in the previous studies. Table 3.5 presents their demographic information. None of them reported to have a condition

the limited their fine motor skills. Each participant received US $15 for volunteering.

3.8.2 Design

We used a within-subjects design for this user study, where the independent variables were: method (two conditions: clockwise, shortest path) and block (5 blocks). We counterbalanced the conditions to reduce any potential effects of order. Each participant used both methods and entered short English phrases from the MacKenzie et al. [118] set in five blocks. Each block contained five random unique phrases from the set. In summary, the design was:

12 participants ×

2 methods (clockwise, shortest path) counterbalanced ×

5 blocks ×

5 random phrases = 600 phrases in total.

The dependent variables were the same performance metrics used in the previous study (Section 3.6.3).

Table 3.5: Demographics of the participants (User Study 3).

Age	M = 28.7 years (SD = 7.2)
Gender	6 female, 5 male and 1 non-binary
Handedness	12 right
Experience with mobile devices	M = 12.1 years (SD = 5.6)
Experience with smartwatches	M = 1.0 year (SD = 2.1)

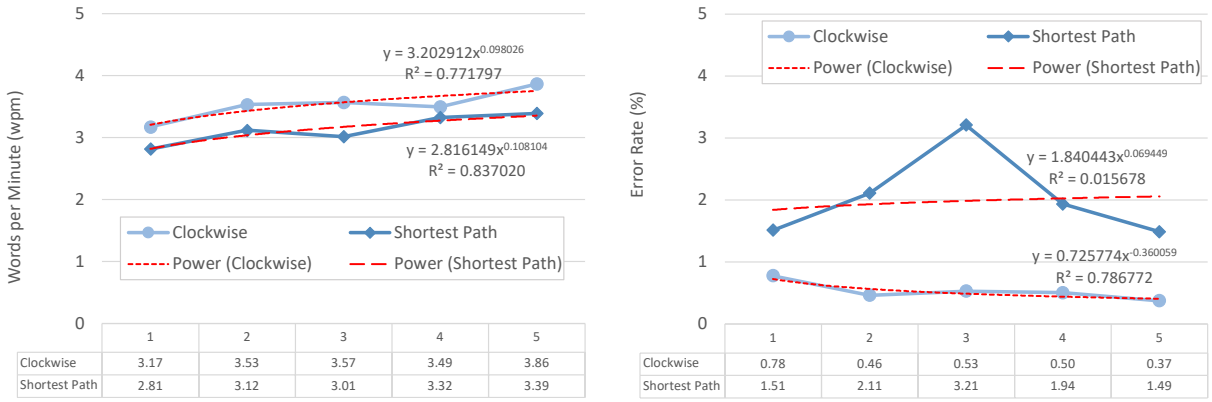
3.8.3 Procedure

We used the same procedure (Section 3.6.4) and adopted the same safety measures (Section 3.6.5) as the previous study. However, unlike the previous study, each participant practiced with both methods and used them in the main study in a counterbalanced order (Fig. 3.8).

3.8.4 Results

A complete study session took about 60 minutes to complete, including demonstration, questionnaires, and breaks. There were no significant effects of the order of conditions on the dependent variables ($p > .05$), which suggests that counterbalancing worked [112, pp. 177–180]. A Shapiro-Wilk test revealed that the response variable residuals were normally distributed. A Mauchly's test indicated that the variances of populations were equal. Hence,

Figure 3.8: Participants entering text with the clockwise (left) and the shortest path crown-based keyboard (right) in the third user study.



	1	2	3	4	5
Clockwise	3.17	3.53	3.57	3.49	3.86
Shortest Path	2.81	3.12	3.01	3.32	3.39

(a) Entry speed (wpm)

	1	2	3	4	5
Clockwise	0.78	0.46	0.53	0.50	0.37
Shortest Path	1.51	2.11	3.21	1.94	1.49

(b) Error rate (%)

Figure 3.9: (a) Average entry speed (wpm) and (b) error rate (%) per block, with both the clockwise and shortest path crown-based keyboard, fitted to power trendlines.

we used a repeated-measures ANOVA for the quantitative factors. We used a Wilcoxon Signed-Rank test on the within-subjects questionnaire data.

3.8.4.1 Entry Speed

An ANOVA identified a significant effect of method on entry speed ($F_{1,11} = 6.11, p < .05$). Average entry speed with clockwise and shortest path were 3.5 wpm (SD = 1.2) and 3.1 wpm (SD = 1.1), respectively. There was also a significant effect of block ($F_{4,44} = 6.05, p < .001$). However, method × block interaction effect was not statistically significant ($F_{4,44} = 0.49, p = .74$). Fig. 3.9a illustrates average entry speed per block for both methods, fitted to power trendlines.

3.8.4.2 Error Rate

An ANOVA identified a significant effect of method on error rate ($F_{1,11} = 10.18, p < .01$). Average error rates with clockwise and shortest path were 0.53% (SD =2.6) and 2.1% (SD = 7.4), respectively. However, there was no significant effect of block ($F_{4,44} = 0.42, p = .79$) or method × block ($F_{4,44} = 0.41, p = .80$). Fig. 3.9b illustrates average error rate per block for both methods, fitted to power trendlines.

3.8.4.3 User Feedback

A Wilcoxon Signed-Rank test identified a significant effect of method on learnability ($Z = -2.12, p < .05$). However, no significant effect was identified on perceived speed ($Z = -0.99, p = .32$), accuracy ($Z = -1.40, p = .16$), ease-of-use ($Z = -1.73, p = .08$), or willingness-to-use ($Z = -0.56, p = .58$). Fig. 3.6 illustrates median user ratings of the two methods.

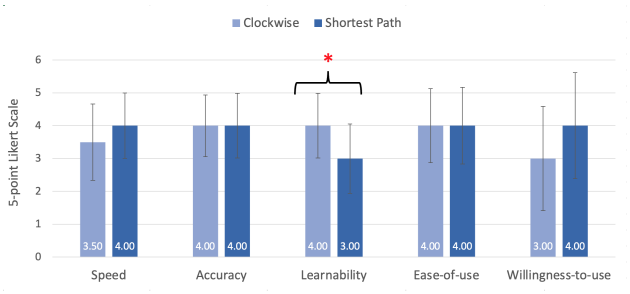


Figure 3.10: Median user ratings of the two methods on a 5-point Likert scale, where 1–5 represented disagree–agree. Error bars represent ±1 standard deviation (SD) and the red asterisks signify statistical significance.

3.8.5 Discussion

The average entry speed of the clockwise crown-based keyboard is comparable to the previous study (3.5 vs. 3.9 wpm). It was significantly faster than the shortest path crown-based keyboard (13% faster). The post-study debrief session revealed that participants struggled with shortest path since they could not anticipate the direction of the rotation, which prevented them from learning the method. This is also reflected in the post-study questionnaire, where shortest path was rated significantly more difficult to learn than clockwise (Fig. 3.10). There was a significant effect of block on entry speed. Learning occurred to some extent with both methods. Average entry speed over block correlated moderately well with the power law of practice [167] for both clockwise ($R^2 = 0.8$) and shortest path ($R^2 = 0.8$). Participants' entry speed improved at a relatively faster rate with automated than shortest path

(Fig. 3.9a). Entry speed with automated improved by 22% from the first block to the last, while the same with shortest path improved by 20%. Relevantly, a post hoc Tukey-Kramer Multiple-Comparison test identified three significantly different groups of blocks in clockwise: {1}, {2,3,4}, {5}, but no such difference was identified in shortest path. The fact that entry speed with clockwise improved by 11% from the fourth block to the last (Fig. 3.9a) suggests that the performance of this method is likely to improve further with practice.

Interestingly, there was a significant effect of method on error rate. Clockwise yielded a 74% lower error rate than shortest path. This also supports the claim that participants had difficulties with shortest path due to the unpredictable nature of its rotation. There was no significant effect of block or method × block, yet average error rate over block correlated moderately well with the power law of practice [167] for clockwise ($R^2 = 0.8$). Hence, there is a chance that the effect of block on accuracy will reach statistical significance with a larger sample size.

Subjective data revealed that almost all participants found the examined methods fast, accurate, and easy-to-use. Participants found shortest path the most difficult to learn, for the reasons discussed earlier. Besides, unlike the previous study, most participants were neutral about using the methods on their smartwatches.

3.9 User Study 4: Comparative Study with Representative Users

We compared the manual and the automated (clockwise) crown-based keyboards with the default virtual keyboard on Wear OS (QWERTY with gesture typing and the predictive system enabled) in a user study involving people with limited motor skills. Based on the findings of the previous study, we excluded the shortest path version from this study (since users had difficulties in learning the method).

3.9.1 Participants

We struggled to recruit participants for this study due to the spread of the delta and omicron variants of COVID-19 virus. Many of our potential volunteers lived in assisted living facilities that were in lockdown. Many others had limited access to transportation. The study required the researcher to travel long distances, since participants were recruited through multiple organizations in different cities. Finally, we recruited ten participants for the study. Table 3.6 presents their demographics information. P3 also took part in the focus group. Each participant received US $40 for volunteering in the study.

3.9.2 Design

The study used a within-subjects design. The independent variable was: method (three conditions: default QWERTY, manual, and automated crown-based keyboards). The dependent

variables were the performance metrics used in the previous studies (Section 3.6.3). Each participant used the three methods to enter short English phrases from the MacKenzie et al. [118] set in four blocks. Each block contained two random unique phrases from the set. In summary:

10 participants ×
3 methods (default, [(manual, automated) counterbalanced]) ×
4 blocks ×
2 random phrases = 165 in total (160 phrases for manual and automated, and 5 phrases for default QWERTY).

Table 3.6: Demographics of the participants (User Study 4).

Participant ID	Gender	Age	Highest Degree	English Proficiency	Cause of Limited Dexterity
P01	Woman	72	University	Native	Age related limited dexterity
P02	Woman	48	Secondary	Native	Quadriplegic paralyzed from the shoulders down, uses pinky finger to type
P03	Woman	37	University	Native	Multiple sclerosis
P04	Man	65	Secondary	Advanced	Dwarfism, arthritis
P05	Woman	65	University	Native	Age related limited dexterity
P06	Man	34	University	Native	Limited dexterity due to hands structure
P07	Woman	30	Secondary	Advanced	Brain injury related limited dexterity
P08	Woman	62	Secondary	Native	Right carpal tunnel surgery, right shoulder surgery. Impingement to jaw, elbow, shoulder. Pain-nerve illness
P09	Woman	62	University	Native	C5–6 quadriplegic, a complete spinal cord injury
P10	Woman	65	University	Native	Quadriplegic, C5–6 spinal cord injury

3.9.3 Procedure

We used the same procedure (Section 3.6.4) and adopted the same safety measures (Section 3.6.5) as the previous studies, except for a few minor differences. First, due to the

unavailability of adequate means of transportation to the participants, the study was conducted in locations convenient to them (Fig. 3.11). The researcher conducted study with one participant at a time in a quiet place. Second, the default condition was always introduced first considering that participants might not be able to use it at all, while the other two conditions were counterbalanced. Third, we excluded the practice session to reduce the duration of the study. Finally, in addition to the usability questionnaire, participants completed the NASA-TLX questionnaire [72] to rate the examined methods' perceived workload.

Figure 3.11: P06 and P10 entering text with the manual (left) and the automated (right) crown-based keyboard, respectively, in the final study.

3.9.4 Results

A complete study session took about 60 minutes to complete, including demonstration, questionnaires, and interviews. In the study, none of the participants were able to complete all sessions with the default QWERTY. As a matter of fact, 70% of them (N = 7) were unable to transcribe even a single phrase with the method. The remaining participants (N = 3) could only enter 1–3 phrases, while committing numerous errors in the process. Their entry speed with manual (M = 2.67 wpm) and automated (M = 2.82 wpm) were 19% and 23% faster than QWERTY (M = 2.17 wpm). Likewise, their error rates with manual (M = 1.5%) and automated (M = 0%) were 83% and 100% lower than QWERTY (M = 8.9%). We, therefore, exclude QWERTY from all quantitative analyses.

A Shapiro-Wilk test revealed that the response variable residuals were normally distributed. A Mauchly's test indicated that the variances of populations were equal. Hence, we used a repeated-measures ANOVA for all quantitative within-subjects factors. We used a Friedman test for the questionnaire data.

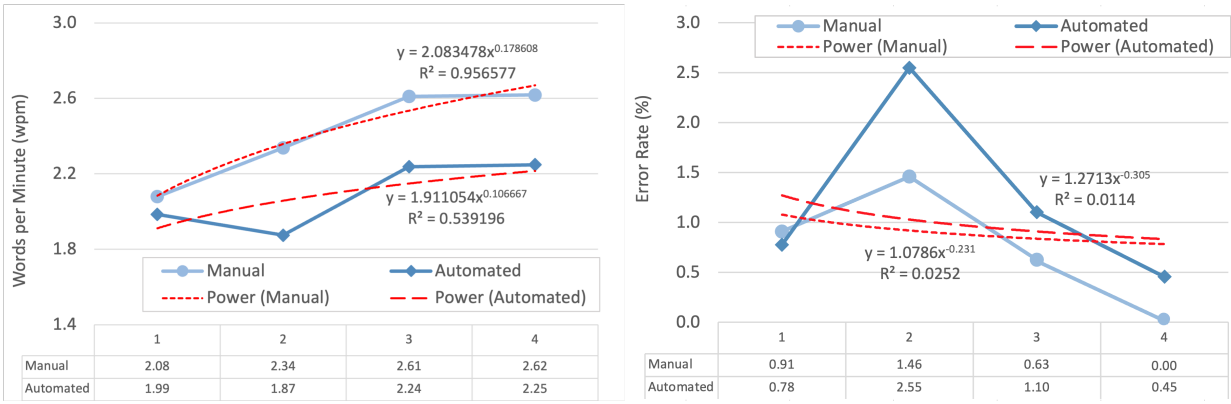
3.9.4.1 Entry Speed

An ANOVA failed to identify a significant effect of method (manual, automated) on entry speed ($F_{1,9} = 1.57, p = .24$). There was also no significant effect of block ($F_{3,27} = 2.34, p =$

.09). Average entry speed with manual and automated were 2.41 wpm (SD = 1.13) and 2.09 wpm (SD = 0.78), respectively. Fig. 3.12a illustrates average entry speed per block for the methods fitted to power trendlines.

3.9.4.2 Error Rate

An ANOVA failed to identify a significant effect of method (manual, automated) on error rate ($F_{1,9} = 0.40, p = .54$). There was also no significant effect of block ($F_{3,27} = 1.12, p = .36$). Average error rates for manual and automated crown-based keyboards were 0.75% (SD = 3.85) and 1.22% (SD = 4.15), respectively. Fig. 3.12b illustrates average error rate per block for both methods fitted to power trendlines.



	1	2	3	4
Manual	2.08	2.34	2.61	2.62
Automated	1.99	1.87	2.24	2.25

(a) Entry speed (wpm)

	1	2	3	4
Manual	0.91	1.46	0.63	0.00
Automated	0.78	2.55	1.10	0.45

(b) Error rate (%)

Figure 3.12: (a) Average words per minute (wpm) and (b) error rate (%) per block for the user study with representative users, with both manual and automated crown-based keyboards, fitted to power trendlines.

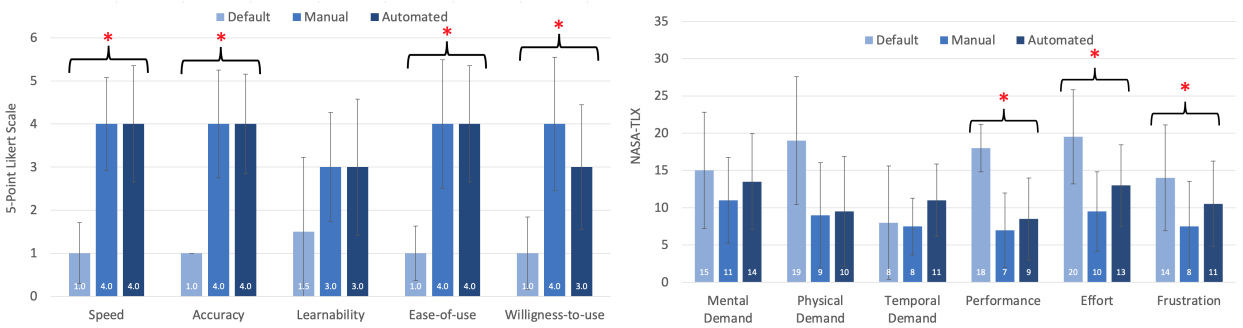
3.9.4.3 User Feedback

A Friedman test identified significant effects of method (QWERTY, manual, automated) on perceived speed ($\chi^2 = 14.29, df = 2, p < .001$), accuracy ($\chi^2 = 15.68, df = 2, p < .0001$), ease of use ($\chi^2 = 14.0, df = 2, p < .0001$), and willingness to use ($\chi^2 = 9.25, df = 2, p < .01$). However, no significant effect was identified on learnability ($\chi^2 = 0.25, df = 2, p = .88$). Fig. 3.13a illustrates median user ratings of the three methods.

3.9.4.4 Task Load Index

For analysis, we considered raw NASA-TLX scores by individual sub-scales, a common practice in the literature [71]. A Friedman test identified significant effects of method

(QWERTY, manual, automated) on performance ($\chi^2 = 11.7, df = 2, p < .01$), effort ($\chi^2 = 6.06, df = 2, p < .05$), and frustration ($\chi^2 = 6.73, df = 2, p < .05$). However, there was no significant effects on mental demand ($\chi^2 = 0.24, df = 2, p = .89$), physical demand ($\chi^2 = 1.52, df = 2, p = .47$), or temporal demand ($\chi^2 = 3.71, df = 2, p = .16$). Fig. 3.13b illustrates median NASA-TLX ratings of all methods.



(a) Questionnaire

(b) NASA-TLX

Figure 3.13: (a) Median user ratings of the three methods on a 5-point Likert scale, where 1–5 represented disagree–agree and (b) raw NASA-TLX scores of the examined methods on a 20-point scale. Error bars represent ±1 standard deviation (SD) and the red asterisks signify statistical significance.

3.9.5 Discussion

Only three participants (P01, P03, P05) were able to enter text with the default keyboard. P05 entered three phrases with a high error rate of 27% before giving up, while P01 and P03 could complete only one phrase each. These participants had mild to moderate motor impairments compared to the other participants. P01 and P05 had limited dexterity due to old age (mostly hand tremor), P03 showed early symptoms of multiple sclerosis, while the others had severe motor impairments (Table 3.6). Nevertheless, all participants were able to complete all blocks with the proposed methods, and those who could use the default method were 19–23% faster and 83–100% more accurate with the new methods. These results suggest that the proposed methods are more accessible to people with various levels of motor impairments. Subjective evaluation also supports this, where participants found the new methods significantly faster, more accurate, and easier to use than the default method (Fig. 3.13a). Although not statistically significant, we also find it encouraging that many participants found the proposed methods more (40%, N = 4) or as learnable (20%, N = 2) as the default method, especially because they all were users of QWERTY on their desktop and laptop platforms. Naturally, almost all of them (90%, N = 9) preferred to use these

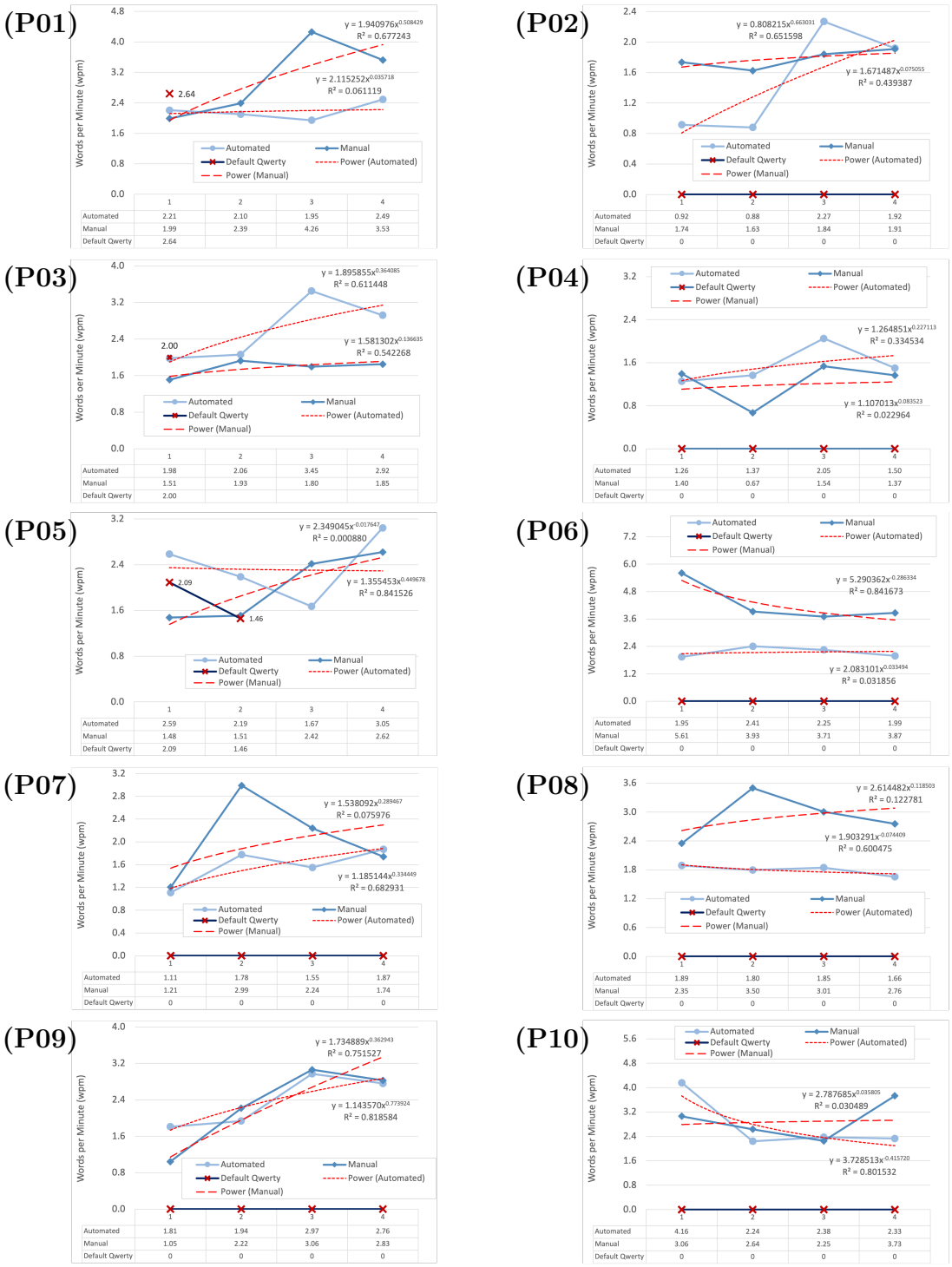


Figure 3.14: Average words per minute (wpm) per block for each of ten participants (P01–P10) for the user study with representative users, with both manual and automated crown-based keyboards, fitted to power trendlines and default QWERTY.

methods on their smartwatches than the default method (statistically significant), while one participant was neutral about it.

Interestingly, there were no statistically significant differences between the methods in terms of mental, physical, and temporal demands (Fig. 3.13). When enquired about this in the post-study interview, participants responded that they rated the demands of tapping on the display and rotating the crown, rather than techniques under investigation. This confusion was caused by the phrasing of the questions, which asked, "*How mentally demanding was the task*?", "*How physically demanding was the task*?", and "*How hurried or rushed was the pace of the task*?", where "the task" was interpreted as tapping and rotating. Note that we used the exact questions from the original questionnaire, without modifications of any kind. Participants felt that tapping on the screen and rotating the crown are somewhat comparable in mental, physical, and temporal demands, but the techniques associated with them make the difference in performance and effort. This is reflected in their responses to the subsequent questions, "*How successful were you in accomplishing what you were asked to do?*", "*How hard did you have to work to accomplish your level of performance*?", and "*How insecure, discouraged, irritated, stressed, and annoyed were you*?" Participants found the new methods significantly better performed than the default method. They also felt that the default method required significantly more effort than the new ones, thus, were significantly more frustrated with it than the proposed ones (Fig. 3.13).

Manual crown-based keyboard was about 13% faster and 37% more accurate than automated crown-based keyboard, but this differences were not statistically significant (Fig. 3.12). This is because there was not a general trend in performance with these methods. Some participants performed much better with one method and some with the other (see Fig. 3.14). We speculate this due to personal preferences as we failed to identify any effects of age and severity of the impairment on performance with the methods. In the post-study interview, some participants praised the automated method, while others criticized it. P02 liked the automated method as it does not require rotating the crown, while P04 criticized it, saying, "*It takes longer [with the method] because if you miss the letter you need to wait when it circles around.*" There was no significant effect of block on entry speed. This is, presumably, due to the brief exposure to the methods. Participants entered only two phrases in four blocks. Yet, average entry speed over block correlated well with the power law of practice [167] for manual crown-based keyboard ($R^2 = 0.96$) and moderately for automated crown-based keyboard ($R^2 = 0.54$). This suggests that participants are likely to get much faster with both methods with practice. The manual and automated crown-based keyboards were comparable in terms of accuracy (0.0% vs. 0.45% ER). We did not find any significant differences between the methods in subjective analyses. As discussed earlier, participants liked both methods significantly better than the default method. These results indicate that both automated and manual crown-based keyboards can be effective in enabling people with limited dexterity to enter text on smartwatches, but people with different levels of motor impairments are likely to prefer different versions of the keyboard.

3.10 Conclusion

In this Chapter, we presented a series of studies to evaluate different versions of crown-based keyboard. In the first study, we assessed manual and automated rotations and found that manual rotation is faster and more accurate. In the second study, we compared automated clockwise rotation with the shortest path rotation, where we found that with the latter one participants could not learn the method since they could not anticipate the direction of the rotation. As a result, automated clockwise rotation was faster and more accurate. Lastly, we conducted the final study with ten people with limited dexterity, where we compared default QWERTY with both manual and automated crown-based keyboards. In the study, most of the participants could not enter text with default QWERTY, while with manual and automated crown-based keyboards they were able to reach 2.62 and 2.25 wpm, respectively, with almost 0% error rate for both. In the next Chapter, we present the keyboard that arranges the QWERTY layout around the bezel of a smartwatch. The keyboard is optimized for usability and to maintain similarities between the gestures drawn on a smartwatch and a virtual QWERTY to facilitate skill transfer.

Chapter 4

SwipeRing: Gesture Typing on Smartwatches Using a Segmented Qwerty Around the Bezel

Figure 4.1: SwipeRing arranges the standard QWERTY layout around the edge of a smartwatch in seven zones. To enter a word, the user connects the zones containing the target letters by drawing gestures on the screen, like gesture typing on a virtual QWERTY. A statistical decoder interprets the input and enters the most probable word. A suggestion bar appears to display other possible words. The user could stroke right or left on the suggestion bar to see additional suggestions. Tapping on a suggestion replaces the last entered word. One-letter and out-of-vocabulary words are entered by repeated strokes from/to the zones containing the target letters, in which case the keyboard first enters the two one-letter words in the English language (see the second last image from the left), then the other letters in the sequence in which they appear in the zones (like multi-tap). Users could also repeatedly tap (instead of stroke) on the zones to enter the letters. The keyboard highlights the zones when the finger enters them and traces all finger movements. This figure illustrates the process of entering the phrase "*the world is a stage*" on the SwipeRing keyboard (upper sequence) and on a smartphone keyboard (bottom sequence). We can clearly see the resemblance of the gestures.

Smartwatches are becoming increasingly popular among mobile users [96]. However, the absence of an efficient text entry technique for these devices limits smartwatch interaction to mostly controlling applications running on smartphones (e.g., pausing a song on a media player or rejecting a phone call), checking notifications on incoming text messages and social media posts, and using them as fitness trackers to record daily physical activity. Text entry on smartwatches is difficult due to several reasons. First, the smaller key sizes of miniature keyboards make it difficult to tap on the target keys (the "fat-finger problem" [180]), resulting in frequent input errors even when augmented with a predictive system. Correcting these errors is also difficult, and often results in additional errors. To address this, many existing keyboards use a multi-action approach to text entry, where the user performs multiple actions to enter one letter (e.g., multiple taps [63], chords (see Chapter 2)). This increases not only learning time but also physical and mental demands. Besides, most existing keyboards cover much of the smartwatch touchscreen (50–85%), reducing the real estate available to view or interact with the elements in the background. Many keyboards for smartwatches that use novel layouts [11] do not facilitate skill transfer. That is, the skills acquired in learning new keyboards are usually not usable on other devices. This discourages users from learning a new technique. Further, most of these keyboards were designed for square watch-faces, thus do not always work on round screens. Finally, some techniques rely on external hardware, which is impractical for wearable devices.

To address these issues, we present SwipeRing, a ring-shaped keyboard that sits around the smartwatch bezel to enable gesture typing with the support of a statistical decoder. Its ring-shaped layout keeps most of the touchscreen available to view additional information and perform other tasks. It uses a QWERTY-like layout divided into seven zones that are optimized to provide comfortable areas to initiate and release gestures, and to maintain similarities between the gestures drawn on a virtual QWERTY and SwipeRing to facilitate skill transfer.

The remainder of the Chapter is organized as follows. First, we discuss the motivation for the work, followed by a review of the literature in the area. We then introduce the new keyboard and discuss its rigorous optimization process. We present the results of a user study that compared the performance of the proposed keyboard with the C-QWERTY keyboard that uses a similar layout but does not divide the keys into zones or optimize for skill transfer and target selection. Finally, we conclude the Chapter with potential future extensions of the work.

4.1 Motivation

The design of SwipeRing is motivated by the following considerations.

4.1.1 Free-up Touchscreen Real-Estate

On a 30.5 mm circular watch, a standard QWERTY layout without and with the suggestion bar occupy about 66% (480 mm^2) and 85% (621 mm^2) of the screen, respectively (Fig. 4.2). On the same device, our technique, SwipeRing, occupies only about 36% (254.34 mm^2) of the screen, almost half the QWERTY layout. Saving screen space is important since the extra space could be used to display additional information and to make sure that the interface is not cluttered, which affects performance [85]. For example, the extra space could be used to display the email users are responding to or more of what they have composed. The former can keep users aware of the context. The latter improves writing quality [185, 189] and performance [73]. Numerous studies have verified this in various settings, contexts, and devices [153, 154, 152, 160, 161, 162, 164].

Figure 4.2: When entering the phrase "the work is done take a coffee break" with a smartwatch QWERTY, only the last 10 characters are visible (left), while the whole phrase is visible (37 characters) with SwipeRing (right). Besides, there is extra space available below the floating suggestion bar to display additional information.

4.1.2 Facilitate Skill Transfer

The skill acquired in using a new smartwatch keyboard is usually not transferable to other devices. This discourages users from learning a new technique. The keyboards that attempt to facilitate skill transfer are miniature versions of QWERTY that are difficult to use due to the small key sizes. To mitigate this, most of these keyboards rely on statistical decoders, making the entry of out-of-vocabulary words difficult, or impossible. SwipeRing uses a different approach. Although gesture typing is much faster than tapping [92], it is not a dominant text entry method on mobile devices. SwipeRing strategically arranges the letters in the zones to maintain gesture similarities between the gestures drawn on a virtual QWERTY and SwipeRing to enable performing the same (or very similar) gesture to enter the same word on various devices. The idea is that it will encourage gesture typing by facilitating skill transfer from smartphones to smartwatches and vice versa.

4.1.3 Increase Usability

As a result of an optimization process, the layout is strategically divided into seven zones, accounting for the mean contact area in index finger touch (between 28.5 and 33.5 mm^2 [181]), to facilitate comfortable and precise target selection during text entry [139]. The zones range between 29.34 and 58.68 mm^2 (lengths between 9.0 and 18.0 mm), which are within the recommended range for target selection on both smartphones [88, 93, 141] and smartwatches (7.0 mm) [45]. SwipeRing employs the whole screen for drawing gestures, which is more comfortable than drawing gestures on a miniature QWERTY. Unlike most virtual keyboards, SwipeRing requires users to slide their fingers from/to the zones instead of tapping, which also makes target selection easier. Existing work on eyes-free bezel-initiated swipe for the ring-shaped layouts revealed that the most accurate layouts have 6–8 segments [190]. SwipeRing enables the entry of out-of-vocabulary words through a multi-tap like an approach [44], where users repeatedly slide their fingers from/to the zone that contains the target letter until the letter is entered (see Section 4.4.2 for further details). Besides, research showed that radial interfaces on circular devices visually appear to take less space even when they occupy the same area as rectangular interfaces, which not only increases clarity but also makes the interface more pleasant and attractive [159].

4.1.4 Face Agnostic

Since SwipeRing arranges the keys around the edge of a smartwatch, it works on both round and square/rectangular smartwatches. To validate this, we investigated whether the gestures drawn on a square smartwatch and a circular smartwatch are comparable to the ones drawn on a virtual QWERTY. A Procrustes analysis [42] on the 10,000 most frequent words drawn on these devices, SwipeRing yielded a score of 114.81 with the square smartwatch and 118.48 with the circular smartwatch. This suggests that the gestures drawn on these devices are very similar. In fact, the square smartwatch yielded a slightly better score than the circular smartwatch (the smaller the score, the better the similarity, Section 4.3.2), most likely because the shapes of these devices are similar (Fig. 4.3).

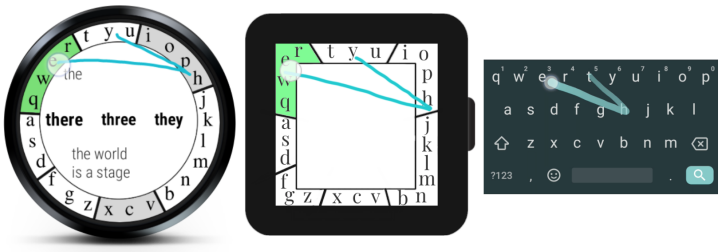


Figure 4.3: Gestures for the most common word “the” on a circular SwipeRing, a square SwipeRing, and a virtual QWERTY.

4.2 Related Work

This section covers the most common text entry techniques for smartwatches. Tables 4.1 and 4.2 summarize the performance of some of these techniques. For a comprehensive review of existing text entry techniques for smartwatches, we refer to Arif et al. [11].

4.2.1 Qwerty Layout

Most text entry techniques for smartwatches are miniature versions of the standard QWERTY that use multi-step approaches to increase the target area. ZoomBoard [137] displays a miniature QWERTY that enables iterative zooming to enlarge regions of the keyboard for comfortable tapping. SplitBoard [74] displays half of a QWERTY keyboard so that the keys are large enough for tapping. Users flick left and right to see the other half of the keyboard. SwipeBoard [35] requires two swipes to enter one letter, the first to select the target letter region and the second towards the letter to enter it. DriftBoard [165] is a movable miniature QWERTY with a fixed cursor point. To enter text, users drag the keyboard to position the intended key within the cursor point. Some miniature QWERTY keyboards use powerful statistical decoders to account for the"fat-finger problem" [180]. WatchWriter [58] appropriates a smartphone QWERTY for smartwatches. It supports both predictive tap and gesture typing. Yi et al. [195] uses a similar approach with even smaller keyboards (30 and 35 mm). VelociWatch [178] also uses a statistical decoder, but enables users to lock in particular letters of their input to disable potential auto-corrections. Some techniques use variants of the standard QWERTY layout. ForceBoard [75] maps QWERTY to a 3×5 grid by assigning two letters to each key. Applying different levels of force on the keys enters the respective letters. DualKey [68] uses a similar layout, but requires users to tap with different fingers to disambiguate the input. It uses external hardware to differentiate between the fingers. DiaQWERTY [90] uses diamond-shaped keys to fit QWERTY in a round smartwatch at 10:7 aspect ratio. Optimal-T9 [148] maps QWERTY to a 3×3 grid, then disambiguates input using a statistical decoder. These techniques, however, occupy a substantial area of the screen real-estate, require multiple actions to enter one letter, or use prediction models that make the entry of out-of-vocabulary words difficult.

4.2.2 Other Layouts

There are a few techniques that use different layouts. Dunlop et al. [45] and Komninos and Dunlop [91] map an alphabetical layout to six ambiguous keys, then uses a statistical decoder to disambiguate input. It enables contextual word suggestions and word completion. QLKP [74] (initially designed for smartphones [77]) maps a QWERTY-like layout to a 3×3 grid. Similar to multi-tap [44], users tap on a key repeatedly until they get the intended letter. These techniques also occupy a substantial area of touchscreen real-estate and require multiple actions to enter one letter.

Table 4.1: Average entry speed (WPM) of popular keyboards for smartwatches from the literature (only the highest reported speed in the last block or session are presented, when applicable) along with the estimated percentage of touchscreen area they occupy.

Method	Used Device	Screen Occupancy	Entry Speed (WPM)
Yi et al.[195]	Watch 1.56”	50%	33.6
WatchWriter [58]	Watch 1.30”	85%	24.0
DualKey [68]	Watch 1.65”	80%	21.6
SwipeBoard [35]	Tablet	45%	19.6
SplitBoard [74]	Watch 1.63”	75%	15.9
ForceBoard [75]	Phone	67%	12.5
ZoomBoard [137]	Tablet	50%	9.3

4.2.3 Ring-Shaped Layouts

There are some ring-shaped keyboards available for smartwatches. InclineType [59] places an alphabetical layout around the edge of the devices. To enter a letter, users first select the letter by moving the wrist, then tap on the screen. COMPASS [196] also uses an alphabetical layout, but does not use touch interaction. To enter text, users rotate the bezel to place one of the three available cursors on the desired letter, then press a button on the side of the watch. WrisText [57] is a one-handed technique, with which users enter text by whirling the wrist of the hand towards six directions, each representing a key in a ring-shaped keyboard with the letters in alphabetical order. BubbleFlick [174] is a ring-shaped keyboard for Japanese text entry. It enables text entry through two actions. Users first touch a key, which partitions the ring-shaped area inside the layout into four radial areas for the four kana letters on the key. Users then stroke towards the intended letter to enter it. A commercial product, TouchOne Keyboard Wear [175] divides an alphabetical layout into eight zones to let users enter text using a T9-like [62] approach. It enables the entry of out-of-vocabulary words using an approach similar to BubbleFlick. Go et al. [53] designed an eyes-free text entry technique that enables users to EdgeWrite [188] on a smartwatch with the support of auditory feedback. Most of these techniques use a sequence of actions or a statistical decoder to disambiguate the input. Besides, most of these techniques are standalone, hence the skills acquired in using these keyboards are usually not usable on other devices.

Cirrin [124] is a pen-based word-level text entry technique for PDAs. It uses a novel ring-shaped layout with dedicated keys for each letter. To enter a word, users pass their pen through the intended letters. This approach has also been used on other devices [87]. C-Qwerty is a similar technique [39], but differs in letter arrangement, which is based on the Qwerty layout. To enter a word with C-Qwerty, users either tap on the letters in the word individually or drag the finger over them in sequence. While there are some

Table 4.2: Average entry speed (WPM) of popular ring-shaped keyboards for smartwatches from the literature (only the highest reported speed in the last block or session are presented, when applicable) along with the estimated percentage of touchscreen area they occupy.

Method	Used Device	Screen Occupancy	Entry Speed (WPM)
WrisText [57]	Watch 1.4”	42.99%	15.2
HiPad [82]	VR	39.70%	13.6
COMPASS [196]	Watch 1.2”	36.07%	12.5
BubbleFlick [174]	Watch 1.37”	52.06%	8.0
C-Qwerty$_{\text{gesture}}$ [39]	Watch 1.39”	43.16%	7.7
Cirrin [87]	PC	60.57%	6.4
InclineType [59]	Watch 1.6”	38.64%	5.9

similarities between Cirrin, C-Qwerty, and SwipeRing, the approach employed in the latter technique is fundamentally different. First, SwipeRing divides the layout into seven zones, thus does not require precise selection of the letters, but much larger zones. Both Cirrin and C-Qwerty, in contrast, use individual keys for each letter, thus require a precise selection of the keys. Second, like gesture typing on a smartphone, SwipeRing does not require users to go over the same letter (or the letters on the same zone) repeatedly if they appear in a word multiple times in sequence (such as, “oo” in “book”). But both Cirrin and C-Qwerty require users to go over the same letter repeatedly in such cases by sliding the finger out of the keyboard then sliding back to the key. We found out users use this strategy also for entering letters that have the respective keys placed side-by-side, as they are difficult to select consecutively due to the smaller size. Finally, SwipeRing is optimized to maintain gesture similarities between SwipeRing and a virtual Qwerty to facilitate skill transfer between devices.

4.3 Layout Optimization

SwipeRing maps the standard Qwerty layout to a ring around the edge of a smartwatch (Fig. 4.1). It places the left and the right-hand keys of Qwerty [127] to the left and right sides of the layout, respectively. Likewise, the top, home, and bottom row keys of Qwerty are placed at the top, middle, and bottom parts of the layout, respectively. Each letter is positioned at multiple of $360°/26$, resulting in an angular step of $13.80°$, starting with the letter ‘q’ at $180°$. This design was adapted to maintain a likeness to Qwerty to exploit the widespread familiarity with the keyboard to facilitate learning [77, 148]. We then grouped the letters into zones to improve the usability of the keyboard by facilitating precise target selection, further discussed in Section 4.3.3. In practice, the letters can be grouped in

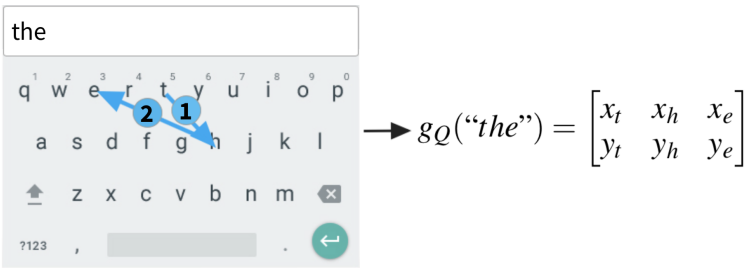


Figure 4.4: The gesture for the most common word “the” on a virtual QWERTY and the respective 2×3 dimensional matrix.

numerous different ways, resulting in a set of possible layouts L. However, the purpose here was to identify a particular layout $l \in L$ that ensures that the gestures drawn on the layout l are similar to the ones drawn on a virtual QWERTY. This requires searching for an optimal letter grouping that maximizes gesture similarity. We introduce the following notation to formally define the optimization procedure. Let $g_Q(w)$ be the gesture used to enter a word w on the virtual QWERTY and $g_{\text{SwipeRing}}(w;\, l)$ be the gesture used to enter the word w on the layout l of SwipeRing. Instead of maximizing the similarity between the gestures, we can equivalently minimize the discrepancy between the gestures, which we measure using a function ψ (see Section 4.3.2 for further details). Then, our problem is to find the layout l that minimizes the following loss function $\mathcal{L}$:

$$\min_{\text{layout } l \in L} \mathcal{L}(l) = \sum_{w \in W} p(w)\, \psi\Big(g_Q(w), g_{\text{SwipeRing}}(w;\, l)\Big). \tag{4.1}$$

Here, the dissimilarity between the gestures is weighted by the probability of the occurrence of the word to assure that the gestures for the most frequent words are the most similar. We made several modeling assumptions and simplifications to efficiently optimize this problem, which are discussed in the following sections.

4.3.1 Gesture Modelling

We model each gesture as a piece-wise linear curve connecting the letters on a virtual QWERTY or the zones on SwipeRing. Therefore, the gesture for a word composed of n letters can be seen as a $2 \times n$ dimensional matrix (Fig. 4.4), where each column contains coordinates (x, y) of the corresponding letter. To simulate the drawn gesture on a virtual QWERTY for the word w, denoted as $g_Q(w)$, we connect the centers of the corresponding keys of the default Android QWERTY on a Motorola G^5 smartphone (24 cm^2 keyboard area)[1], producing unique gestures for each word. With SwipeRing, however, we account for the fact that a word can

[1]Since most virtual QWERTY layouts maintain comparable aspect ratios and the gestures only loosely connect the keys, the gestures on different sized phones, keyboards, keys are comparable when the recognizer

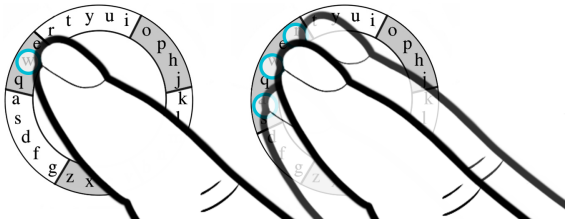


Figure 4.5: Gestures on the three letter zones are likely to be initiated from the center, while gesture on the wider zones (such as, a six letter zone) can be initiated from either the center or the two sides.

have multiple gestures forming a set $G_{\text{SwipeRing}}(w;\, l)$. The zones containing 4–6 letters are wide enough to enable initiating a gesture either at the center, left, or right side of the zone (Fig. 4.5), resulting in multiple possibilities. Therefore, we set the gesture $g_{\text{SwipeRing}}(w;\, l)$ to be the one that has minimal difference when compared to the gesture drawn on the virtual QWERTY measured by discrepancy function ψ:

$$g_{\text{SwipeRing}}(w;\, l) := \underset{g \in G_{\text{SwipeRing}}(w;\, l)}{\arg\min} \psi(g_Q(w), g). \tag{4.2}$$

4.3.2 Discrepancy Function

For the discrepancy function $\psi(g_1, g_2)$ between gestures g_1 and g_2, our requirement is to have a rotation and scale agnostic measure that attains a value of 0 if and only if g_2 is a rotated and re-scaled version of g_1. One possible form of such ψ function can be defined as yet another optimization problem of:

$$\psi(g_1, g_2) = \min_{R,\, \alpha} \|g_2 - \alpha R g_1\|_F^2. \tag{4.3}$$

Where α and R are the rescaling factor and the rotation matrix applied to a gesture, respectively, while $\|.\|_F$ is the Frobenius norm[2]. We recognize Equation 4.3 as an Ordinary Procrustes analysis problem, the solution of which is given in closed-form by Singular Value Decomposition [42]. Note that the value of $\psi(g_1, g_2)$ is within the range of $[0, \infty]$. Additionally, we restrict the rotation within the range of $\pm 45°$ since SwipeRing gestures that are rotated more than 45° in either direction are unlikely to look similar to their virtual QWERTY counterparts (Fig. 4.6).

is size agnostic. A Procrustes analysis of the gestures drawn on five different sized phones with different keyboards and key sizes yielded results between 6 and 19, suggesting they are almost identical.

[2]Frobenius norm is a generalization of Euclidean norm to the matrices, such as if A is a matrix then $\|A\|_F^2 = \sum_{i,j} a_{ij}^2$

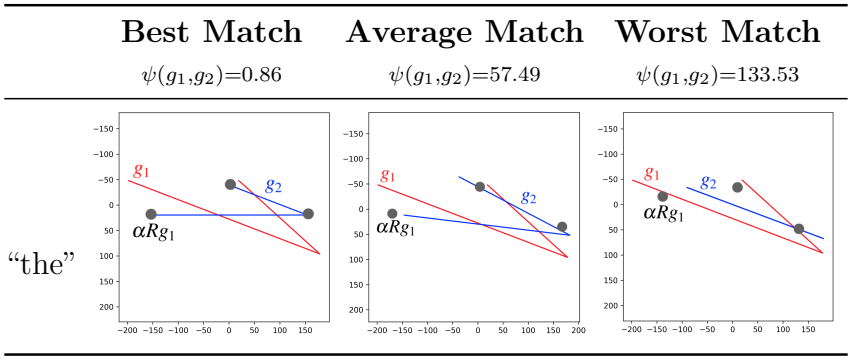


Figure 4.6: Gesture dissimilarity measures using the Procrustes loss function. The red curve represents the gesture for the word "the" on a virtual QWERTY (g_1), the blue curves represent gestures for the same word on a SwipeRing layout (g_2), and the gray dots show the optimal rotation and rescaling of the gesture (g_1) represented as ($\alpha R g_1$) to match (g_2).

4.3.3 Enumeration of All Possible Letter Groupings

The total number of possible letter groupings, and thus layouts, depends on how large we allow the groups to be. To determine this, we conducted a literature review of ambiguous keyboards that use linguistic models for decoding the input to find out whether the number of letters assigned per key or zone (the level of ambiguity) affects the performance of a keyboard. Table 4.3 displays an excerpt of our review, where one can see "somewhat" inverse relationship between the level of ambiguity and entry speed. Keyboards that assign fewer letters per key or zone yield a relatively better entry speed than those with more letters per key or zone. Based on this, we decided to assign 3–6 letters per zone. Although, this alone cannot determine the appropriate number of letters in each key since the performance of a keyboard depends on other factors, such as the layout and the reliability of the decoder, it gives a rough estimate.

Next, we discovered all possible shatterings of the ring-shaped string {`qwertyuiophjklmn-`

Table 4.3: Average entry speed of several ambiguous keyboards that map multiple letters to each key or zone.

Method	Letters per Key	Entry Speed (WPM)
COMPASS [196]	3	9–13
HiPad [82]	4–5	9.6–11
WrisText [57]	4–5	10
Komninos, Dunlop [45, 91]	3–6	8

bvcxzgfdsa} into substrings of length 3–6 letters each, resulting in 4,355 different layouts in total, which constitute our search set L. Each possible shattering, such as {qwer}{tyu}{ioph-jk}{lmnbv}{cxzg}{fdsa}, represents one possible layout. We tested several of these layouts on a small smartwatch (9.3 cm^2 circular display) to investigate if the zones containing three letters are wide enough for precise target selection. Results showed that the zones range between 29.0 and 57.5 mm^2 (lengths between 9.0 and 18.0 mm), which are within the length recommended for target selection on both smartphones [88, 93, 141] and smartwatches (7.0 mm) [45]. Fig. 4.7 illustrate some of these layouts.

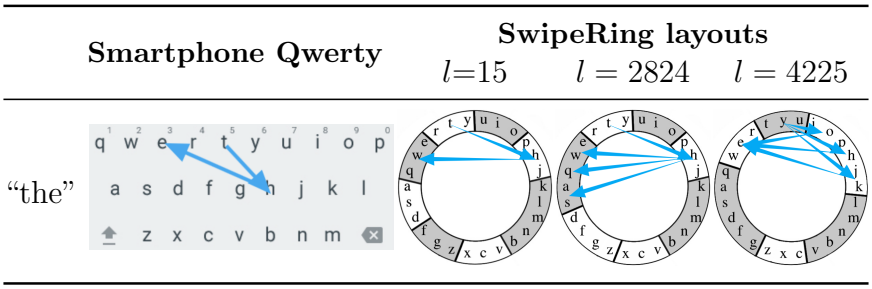


Figure 4.7: Gesture typing the word "the" on a virtual QWERTY and three possible SwipeRing layouts. For the virtual QWERTY, the figure shows $g_Q(\text{“the”})$. For the SwipeRing layouts, the figure shows all possible gestures for "the": $G_{\text{SwipeRing}}(\text{“the”}, l)$. Notice how the gestures for the same word are different on different SwipeRing layouts.

4.3.4 Algorithm

To find the optimal layout, we simulated billions of gestures for the 10,000 most frequent words in the English language [184] on the 4,355 possible segmented SwipeRing layouts[3]. We then matched the gestures produced for each word on each layout with the gestures produced on a virtual QWERTY using the Procrustes analysis to pick the layout that yielded the best match score (118.48). The final layout (Fig. 4.1) scored, on average, 1.27 times better Procrustes value compared to the other possible layouts.

4.4 Keyboard Features

This section describes some key features of the proposed keyboard.

[3]We only used words that had more than one letter, there were 9,828 such words in the corpus.

Algorithm 3: Search for an optimal layout l.

Input: Possible grouping layouts $L = \{l_1, l_2, \dots\}$, word corpus $W = \{w_1, w_2, \dots\}$
Function `OptLayout`(L, W)**:**
  $\mathcal{L}_{\min} \leftarrow \infty, l_{\min} \leftarrow \infty$
  for *layout* $l \in L$ **do**
    $\mathcal{L} \leftarrow 0$
    for *word* $w \in W$ **do**
      $\mathcal{L} \leftarrow \mathcal{L} + p(w)\,\psi\Big(g_Q(w), g_{\text{SwipeRing}}(w;\, l)\Big)$
    end
    if $\mathcal{L} \leq \mathcal{L}_{\min}$ **then**
      $\mathcal{L}_{\min} \leftarrow \mathcal{L}, l_{\min} \leftarrow l$
    end
  end

4.4.1 Decoder

We developed a simple decoder to suggest corrections and display the most probable words in a suggestion bar. For this, we used a combination of a character-level language model and a word-level bigram model for the next word prediction. To this end, we calculate the conditional probability of the user typing the word w given that the previous word was w_{n-1} and the current zone sequence is s:

$$\begin{aligned} P(w_n = w |\, s, w_{n-1}) &= \frac{P(w_n = w, s, w_{n-1})}{P(s, w_{n-1})} \\ &= \frac{\text{count}(w_n = w, w_{n-1}) \times \text{match}(M(w), s)}{\sum_{w'} \text{count}(w_n = w', w_{n-1}) \times \text{match}(M(w'), s))}. \end{aligned} \tag{4.4}$$

Here, $M(w)$ is the sequence of zones that the user must gesture over to enter the word w with SwipeRing, $\text{match}(s_1, s_2)$ is the indicator function that returns 1 if s_2 is a prefix of s_1 or 0 otherwise, and $\text{count}(w_n, w_{n-1})$ is the number of occurrences of a bigram (w_n, w_{n-1}) in the training corpus.

To predict the most probable word for a given zone sequence s and previous word w_{n-1}, we compute $\arg\max_w P(w_n = w |\, s, w_{n-1})$ using the prefix tree (Trie) data structure. This implementation can output k highest probable words, which we display in the suggestion bar. When no word has been typed yet, we use a unigram reduction of the model, otherwise, we use the bigram model trained on the COCA corpus [38]. Due to the limited memory capacity of the smartwatch, the Trie uses the 1,300 most probable bigrams: bigram models scale as the square of the number of words, thus quickly outrun the available memory on the device. If the Trie does not have a bigram containing the user's previous word w_{n-1}, we revert to the unigram predictions. Our language model is fairly simple, and more advanced models (involving neural nets, for instance) can be created. However, devising efficient language

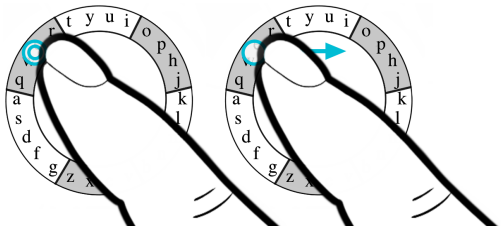


Figure 4.8: SwipeRing enables users to enter one-letter and out-of-vocabulary words by repeated strokes from/to the zones containing the target letters, like multi-tap (right). Users could also repeatedly tap on the zones (instead of strokes) to enter the letters (left).

models for new keyboards is a research problem on its own and beyond the scope of this work.

After obtaining the list of the most probable words, SwipeRing places up to 10 most probable words in the suggestion bar, automatically positioned in close proximity to the input area (Fig. 4.1). The suggestion bar automatically updates as the user continues gesturing. Once done, the most probable word from the list is entered. The user could select a different word from the suggestion bar by tapping on it, which replaces the last entered word. Although the user can only see 2–4 words in the suggestion bar due to the smaller screen, she can swipe left and right on the bar to see the remaining words.

4.4.2 One-Letter and Out-of-Vocabulary (OOV) Words

SwipeRing enables the entry of one-letter and out-of-vocabulary words through repeated taps or strokes from/to the zones containing the target letters. The keyboard first enters the two one-letter words in the English language "a" and "I," then the other letters in the sequence in which they appear in the zones, like multi-tap [44]. For instance, to enter the letter 'e', which is in the top-right zone containing the letters: 'q', 'w', 'e', and 'r' (Fig. 4.1), the user taps or slides the finger three times from the middle area to the zone or from the edge to the middle area (Fig. 4.8).

4.4.3 Error Correction and Special Characters

SwipeRing automatically enters a *space* when a word is predicted or manually selected from the suggestion bar. During character-level text entry (to enter out-of-vocabulary words), users enter *space* by performing a right stroke inside the empty area of the keyboard. Tapping on the transcribed text deletes the last entry, either a word or a letter. The keyboard performs a *carriage return* or an *enter* action when the user double-taps on the screen. Currently, SwipeRing does not support uppercase letters, special symbols, numbers, and languages other than English. However, these could be easily added by enabling the user to long-press

or dwell on the screen or the zones to switch back and forth between the cases and change the layout for digits and symbols. Note that the evaluation of novel text entry techniques without the support for numeric and special characters is common practice since it eliminates a potential confound [118].

4.5 User Study

We conducted a user study to compare SwipeRing with C-Qwerty. C-Qwerty uses almost the same layout as SwipeRing but places 'g' at the NE corner, while SwipeRing places it at the SW corner (left side of the layout since 'g' on Qwerty is usually pressed with the left hand). Both layouts share the design goal of maintaining similarity to Qwerty by using the touch-typing metaphor of physical keyboards (keys assigned to different hands). This likely resulted in similar (but nonidentical) layouts. Studies showed that using a physical analogy/metaphor like this enables novices to learn a method faster by skill transfer [112, pp. 255–263]. Besides, C-Qwerty does not divide the keys into zones, optimize them for gesture typing and skill transfer, and uses a slightly different mechanism for gesture drawing approaches for the two are also different (Section 4.2.3). Hence, a comparison between the two will highlight the performance difference due to the contributions of this work.

4.5.1 Apparatus

We used an LG Watch Style smartwatch, 42.3×45.7×10.8 mm, 9.3 cm^2 circular display, 46 grams, running on the Wear OS at 360×360 pixels in the study (Fig. 4.9). We decided to use a circular watch in the study since it is the most popular shape for (smart)watches [84, 89]. We developed SwipeRing with the Android Studio 3.4.2, SDK 28. We collected the original source code of C-Qwerty from Costagliola et al. [39], which was also developed for the Wear OS. Both applications calculated all performance metrics directly and logged all interactions with timestamps.

4.5.2 Design

We used a between-subjects design to avoid interference between the conditions. Since both techniques use similar layouts, the skill acquired while learning one technique would have affected performance with the other technique [112]. There were separate groups of twelve participants for C-Qwerty and SwipeRing. Each group used the respective technique to enter short English phrases in eight blocks. Each block contained 10 random phrases from a set [118]. Hence, the design was as follows.

2 groups: C-Qwerty and SwipeRing ×
12 participants ×
8 blocks ×
10 random phrases = 1,920 phrases in total.

4.5.3 Participants

Twenty-four participants took part in the user study. They were divided into two groups. Table 4.4 present the demographics of these groups. Almost all participants chose to wear the smartwatch on their left hand and perform the gestures using the index finger of the right hand (Fig. 4.9). All participants were proficient in the English language. In both groups, three participants identified themselves as experienced gesture typists. However, none of them used the method dominantly, instead frequently switched between tap typing and gesture typing for text entry. The remaining participants never or very rarely used gesture typing on their devices. Initially, we wanted to recruit more experienced gesture typists to compare the performance of inexperienced and experienced users to investigate whether the gesture typing skill acquired on mobile devices transferred to SwipeRing. But we were unable to recruit experienced gesture typists after months of trying. This strengthens our argument that gesture typing is still not a dominant method of text entry, regardless of being much faster than tap typing [92], and using SwipeRing may encourage some users to apply the acquired skill on mobile devices. All participants received a small compensation for participating in the study.

Table 4.4: Demographics of the two groups. YoE stands for years of experience.

	C-Qwerty	SwipeRing
Age	21–34 years (M = 25.8, SD = 3.9)	21–28 years (M = 24.8, SD = 2.3)
Gender	3 female, 9 male	4 female, 8 male
Handedness	11 right, 1 left	10 right, 1 ambidextrous, 1 left
Smartwatch owners	5 (M = 0.8 YoE, SD = 1.4)	6 (M = 1.2 YoE, SD = 0.9)
Experienced gesture typists	3 (M = 4.7 YoE, SD = 2.5)	3 (M = 2.7 YoE, SD = 0.9)

Figure 4.9: The device with C-Qwerty and a participant volunteering in the study over Zoom (left). The device with SwipeRing and a volunteer participating in the study (right).

4.5.4 Performance Metrics

We calculated the conventional *words per minute* (WPM) [14] and *total error rate* (TER) performance metrics to measure the speed and accuracy of the keyboard, respectively. TER [171] is a commonly used error metric in text entry research that measures the ratio of the total number of *incorrect* characters and *corrected* characters to the total number of *correct*, *incorrect*, and *corrected* characters in the transcribed text. We also calculated the *actions per word* metric that signifies the average number of actions performed to enter one word. An action could be a gesture performed to enter a word, a tap on the suggestion bar, or a gesture to delete an unwanted word or letter.

4.5.5 Procedure

The study was conducted in a quiet room, one participant at a time. First, we introduced the keyboards to all participants, explained the study procedure, and collected their consents. We then asked them to complete a short demographics and mobile usage questionnaire. We instructed participants to sit on a chair, wear the smartwatch on their preferred hand, and practice with the keyboard they were assigned to by transcribing two short phrases. These practice phrases were not included in the main study. Interestingly, all participants decided to wear the smartwatch on their left hand and perform the gestures using the index finger of the other hand. The actual study started after that. There were eight blocks in each condition, with at least 5-10 minutes gap between the blocks. In each block, participants transcribed ten random short English phrases from a set [118] using either C-Qwerty$_{\text{Gesture}}$ or SwipeRing. Both applications presented one phrase at a time at the bottom of the smartwatch (Fig. 4.9). Participants were instructed to read, understand, and memorize the phrase, transcribe it "*as fast and accurate as possible*," then double-tap on the touchscreen to see the next phrase. The transcribed text was displayed on the top of the smartwatch. Error correction was recommended but not forced. After the study, all participants completed a short post-study questionnaire that asked them to rate various aspects of the keyboard on a 7-point Likert scale. It also enabled participants to comment and give feedback on the examined keyboards.

Due to the spread of COVID-19, the C-Qwerty group participated in the study via Zoom, a teleconference application. We personally delivered the smartwatch to each participant's mailbox and scheduled individual online sessions with them. They were instructed to join the session from a quiet room. All forms were completed and signed electronically. Apart from that, an online session followed the same structure as a physical session. A researcher observed and recorded a complete study session. We picked up the devices after the study. The device, the charger, and the container were disinfected before delivery and after pickup.

4.5.6 Results

A Shapiro-Wilk test revealed that the response variable residuals were normally distributed. A Mauchly's test indicated that the variances of populations were equal. Hence, we used a Mixed-design ANOVA for one between-subjects and one within-subjects factors (technique and block, respectively). We used a Mann-Whitney U test to compare user ratings of various aspects of the two techniques.

4.5.6.1 Entry Speed

An ANOVA identified a significant effect of technique on entry speed ($F_{1,22} = 25.05, p < .0001$). There was also a significant effect of block ($F_{7,22} = 63.65, p < .0001$). The technique $\times$ block interaction effect was also statistically significant ($F_{7,154} = 4.02, p < .0005$). Fig. 4.10 (top) illustrates average entry speed for both techniques in each block, fitted to a function to model the power law of practice [30]. In the last block, the average entry speed with C-QWERTY and SwipeRing were 11.20 WPM (SD = 3.0) and 16.67 WPM (SD = 5.36), respectively. Nine users of SwipeRing yielded a much higher entry speed than the maximum entry speed reached with C-QWERTY, illustrated in Fig. 4.10 (bottom). The highest average entry speed in the last block was 21.53 WPM (P23, inexperienced gesture typist).

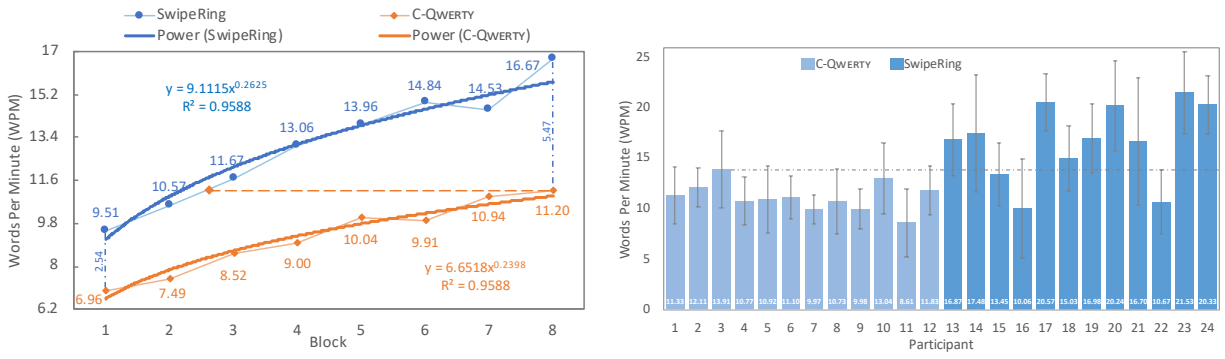


Figure 4.10: Average entry speed (WPM) per block fitted to a power trendline (left). The SwipeRing group surpassed the C-QWERTY group's maximum entry speed by the third block. Note the scale on the vertical axis. Average entry speed (WPM) with the two techniques for each participant in the final block (right). Error bars represent ± 1 standard deviation (SD).

4.5.6.2 Error Rate

An ANOVA identified a significant effect of technique on error rate ($F_{1,22} = 24.61, p < .0001$). There was also a significant effect of block ($F_{7,22} = 2.89, p < .01$). However, the technique $\times$ block interaction effect was not significant ($F_{7,154} = 1.01, p > .05$). Fig. 4.11 (top) illustrates average error rate for both techniques in each block, fitted to a function to model the power

law of practice [30]. In the last block, the average error rates with C-QWERTY and SwipeRing were 12.52% (SD = 13.91) and 5.56% (SD = 8.53), respectively.

4.5.6.3 Actions per Word

An ANOVA identified a significant effect of technique on actions per word ($F_{1,22} = 10.31, p < .005$). There was also a significant effect of block ($F_{7,22} = 3.14, p < .005$). However, the technique × block interaction effect was not significant ($F_{7,154} = 0.61, p > .05$). Fig. 4.11 (bottom) illustrates average actions per word for both techniques in each block, fitted to a function to model the power law of practice [30]. In the last block, the average actions per word with C-QWERTY and SwipeRing were 2.45 (SD = 1.64) and 1.59 (SD = 0.72), respectively.

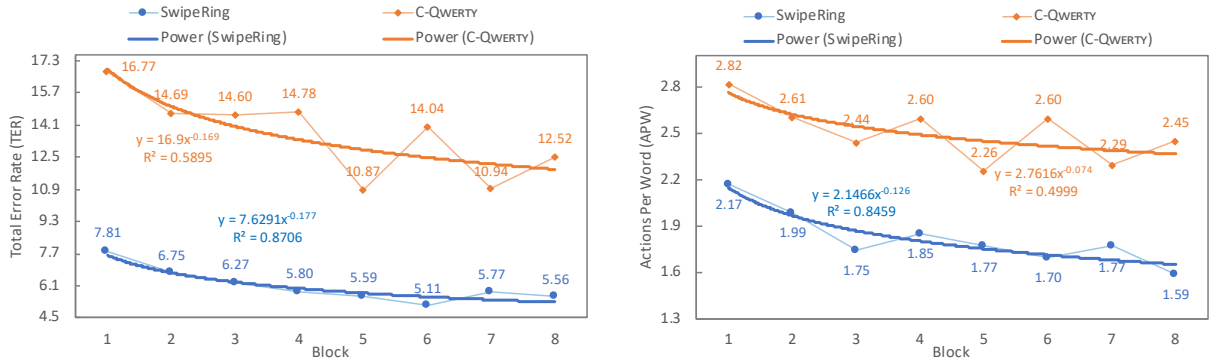


Figure 4.11: Average total error rate (TER) (left) and actions per word (APW) (right) in each block fitted to a power trendline. Note the scale on the vertical axis.

4.5.7 Inexperienced vs. Experienced Gesture Typists

Although there were not enough data points to run statistical tests to compare the two user groups, average performance over blocks suggests that learning occurred with both experienced and inexperienced gesture typists with both techniques. The average entry speed over block correlated well with the power law of practice for C-QWERTY with both user groups (experienced: $R^2 = 0.8142$, inexperienced: $R^2 = 0.9547$), also for SwipeRing (experienced: $R^2 = 0.8732$, inexperienced: $R^2 = 0.9702$), illustrated in Fig. 4.12. The average error rate over block for both techniques, in contrast, correlated well with the power law of practice [30] for inexperienced participants (C-QWERTY: $R^2 = 0.86$, SwipeRing: $R^2 = 0.7019$), but not for experienced participants (C-QWERTY: $R^2 = 0.0231$, SwipeRing: $R^2 = 0.3329$). The average actions per word over block yielded a similar trend for C-QWERTY, where learning was observed with inexperienced participants ($R^2 = 0.8978$), but not with experienced par-

ticipants ($R^2 = 0.1426$). However, with SwipeRing, both user groups continued improving with practice (experienced: $R^2 = 0.7171$, inexperienced: $R^2 = 0.8012$), see Fig. 4.13.

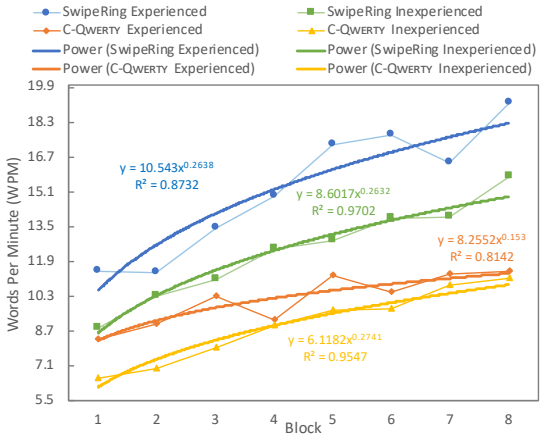


Figure 4.12: Average entry speed (WPM) per block for the two user groups with the two techniques fitted to power trendlines. Note the scale on the vertical axis.

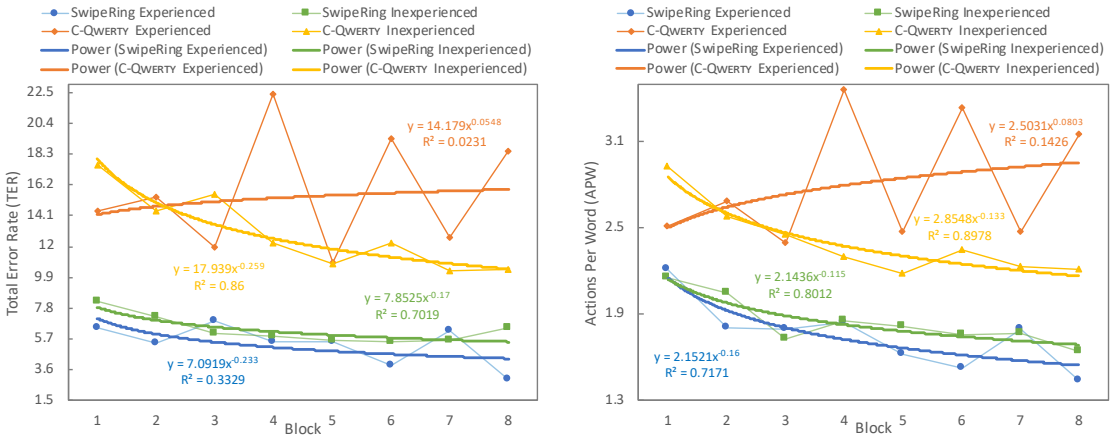


Figure 4.13: Average error rate (TER) (left) and average actions per word (APW) (right) per block for the two user groups with the two techniques fitted to power trendlines. Note the scale on the vertical axis.

4.5.8 User Feedback

A Mann-Whitney U test identified a significant effect of technique on willingness to use ($U = 21.0, Z = -3.1, p < .005$), perceived speed ($U = 22.5, Z = -3.07, p < .005$), and

perceived accuracy ($U = 27.0, Z = -2.72, p < .01$). However, there was no significant effect on ease of use ($U = 48.0, Z = -1.5, p > .05$) or learnability ($U = 66.0, Z = -0.37, p > .05$). Fig. 4.14 illustrates median user ratings of all investigated aspects of the two keyboards on a 7-point Likert scale.

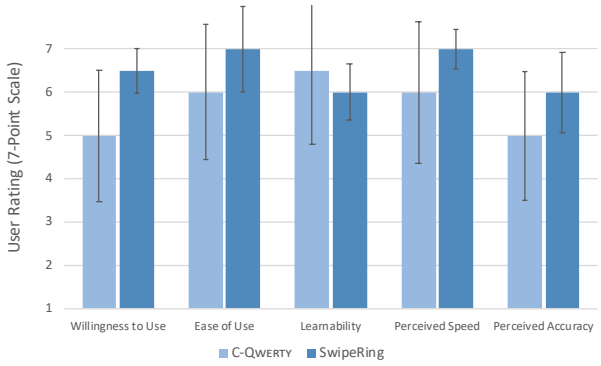


Figure 4.14: Median user ratings of the willingness to use, ease of use, learnability, perceived speed, and perceived accuracy of SwipeRing and C-Qwerty on a 7-point Likert scale, where "1" to "7" represented "Strongly Disagree" to "Strongly Agree." The error bars signify ±1 standard deviations (SD).

4.6 Discussion

SwipeRing reached a competitive entry speed in only eight short blocks. It was 33% faster than C-Qwerty. The average entry speed with C-Qwerty and SwipeRing were 11.20 WPM and 16.67 WPM, respectively. Four participants reached over 20 WPM with SwipeRing (Fig. 4.10, bottom). Further, the SwipeRing group surpassed the C-Qwerty group's maximum entry speed by the third block (Fig. 4.10, top). It also performed better than all popular ring-shaped text entry techniques (Table 4.2) and some Qwerty-based techniques (Table 4.1) for smartwatches. Yi et al. [195] and WatchWriter [58] reported much higher entry speed than SwipeRing. Both techniques use aggressive statistical models with a miniature Qwerty to account for frequent incorrect target selection due to the smaller key sizes (the "fat-finger problem" [180]). This makes entering out-of-vocabulary words difficult with these techniques. In fact, the former technique does not include a mechanism for entering out-of-vocabulary words [195, p. 58]. DualKey [68] and SwipeBoard [35] also reported higher entry speed than SwipeRing. However, DualKey depends on external hardware to distinguish between different fingers and has a steep learning curve (the reported entry speed was achieved in the 15th session). SwipeBoard, on the other hand, was evaluated on a tablet computer, hence unclear whether the reported entry speed can be maintained on an actual

smartwatch. Besides, all of these keyboards occupy about 45–85% of the screen real-estate, leaving a little room for displaying the entered text, let alone additional information. There was a significant effect of block and technique × block on entry speed. Entry speed increased by 38% with C-QWERTY and 43% with and SwipeRing in the last block compared to the first. The average entry speed over block for both techniques correlated well with the power law of practice [30] ($R^2 = 0.9588$). However, the learning curve for C-QWERTY was flattening out by the last block, while SwipeRing was going strong. An analysis revealed that entry speed improved by 2% with C-QWERTY and 13% with SwipeRing in the last block compared to the second-last. This suggests that SwipeRing did not reach its highest possible speed in the study. Relevantly, the highest entry speed recorded in the study was 33.18 WPM (P23, Block 6).

There was a significant effect of technique on error rate. SwipeRing was significantly more accurate than C-QWERTY (Fig. 4.11, top). The average error rate with C-QWERTY and SwipeRing were 12.52% and 5.56%, respectively, in the last block (56% fewer errors with SwipeRing). This is unsurprising since the designers of C-QWERTY also reported a high error rate with the technique (20.6%) using the same TER metric [39] that accounts for both corrected and uncorrected errors in the transcribed text [171]. Most text entry techniques for smartwatches report character error rate (CER) that only accounts for the uncorrected errors in the transcribed text [14]. Most errors with C-QWERTY were committed due to incorrect target selection since the keys were too small. SwipeRing yielded a lower error rate due to the larger zones that were designed to accommodate precise target selection. There was a significant effect of block on error rate. Participants committed 13% fewer errors with C-QWERTY and 29% fewer errors with SwipeRing in the last block compared to the first. The average error rate over block correlated moderately for C-QWERTY ($R^2 = 0.5895$) but well for SwipeRing ($R^2 = 0.8706$) with the power law of practice [30]. Hence, it is likely that SwipeRing will become much more accurate with practice. Actions per word yielded a similar pattern as error rate. SwipeRing consistently required fewer actions to enter words than C-QWERTY (Fig. 4.11, bottom). C-QWERTY and SwipeRing required on average 2.45 and 1.59 actions per word in the last block, respectively (35% fewer actions with SwipeRing). This is mainly because participants performed fewer corrective actions with SwipeRing than C-QWERTY. There was also a significant effect of block. The average actions per word over block correlated well for SwipeRing ($R^2 = 0.8459$) but not for C-QWERTY ($R^2 = 0.4999$) with the power law of practice [30]. This suggests that actions per word with SwipeRing is likely to further improve with practice.

Qualitative results revealed that the SwipeRing group found the examined technique faster and more accurate than the C-QWERTY group (Fig. 4.14). These differences were statistically significant. Consequently, the SwipeRing group was significantly more interested in using the technique on their devices than the C-QWERTY group. However, both techniques were rated comparably on ease of use and learnability, which is unsurprising since both techniques used similar layouts.

We compared the performance of C-QWERTY in our study with the results from the literature to find out whether conducting the study remotely affected its performance. Costagli-

ola et al. [39] reported a 7.7 WPM entry speed with a 20.6% error rate on a slightly larger smartwatch using the same phrase set in a single block containing 6 phrases. In our study, C-Qwerty yielded a comparable 7 WPM and 16.8% error rate in the first block containing 10 phrases.

4.6.1 Skill Transfer from Virtual Qwerty

Although there were not enough data points to run statistical tests, average performance over blocks suggests that experienced participants were performing much better with both techniques from the start. Fig. 4.12 shows that experienced participants consistently performed better than inexperienced participants. This suggests that experienced participants were able to transfer their smartphone gesture typing skills to both techniques. However, with C-Qwerty, inexperienced participants almost caught up with the experienced participants by the last block. While with SwipeRing, both user groups were learning at comparable rates in all blocks. Besides, both experienced and inexperienced participants constantly performed better with SwipeRing than C-Qwerty. These indicate towards the possibility that optimizing the zones for gesture similarities facilitated a higher rate of skill transfer. As blocks progressed, experienced participants were most probably more confident, consciously or subconsciously, in applying their gesture typing skills to SwipeRing.

Interestingly, error rate and actions per word patterns were quite different from the patterns observed in entry speed. With SwipeRing, experienced participants were consistently better than inexperienced users, while inexperienced participants were learning to be more accurate. We speculate this is because experienced participants made fewer errors than inexperienced participants, which required performing fewer corrective actions (a phenomenon reported in the literature [16]). In contrast, with C-Qwerty, experienced participants committed more errors, requiring more corrective actions. In Fig. 4.13, one can see that experienced participants' error rates and actions per word went up and down in alternating blocks. We do not have a definite explanation for this, but based on user comments we speculate that this is because experienced participants were trying to apply their gesture typing skills to C-Qwerty, only committing more errors due to the smaller target size, then reduced speed in the following block to increase accuracy (i.e., the speed-accuracy trade-off). This process continued till the end of the study. Interestingly, with SwipeRing, both inexperienced and experienced gesture typists were improving their average actions per word with practice. It could be because participants gradually learned how to exploit the wider zones of the layout to reduce the number of incorrect actions. This suggests that optimizing the zones for precise target selection facilitated a higher rate of skill acquisition.

4.7 Limitations

We discussed several limitations of the work. To summarize: first, we had different numbers of experienced and inexperienced participants in the study. The sample size was also

small. Hence, a definitive conclusion cannot be drawn about the transference of skill between devices. Second, due to the spread of COVID-19, we had to switch to an online format mid-study, as in-person studies were unsafe. This resulted in one condition being studied in-person, while the other online. However, based on the performance reported in prior work, we speculate that this did not impact the findings of this research. Finally, we did not investigate the proposed method in mobile settings, such as while walking or commuting. However, we anticipate our method to perform much better in such scenarios compared to the conventional methods since it does not always require precise target selection.

4.8 Conclusion

In this Chapter, we presented SwipeRing, a ring-shaped keyboard arranged around the bezel of a smartwatch to enable gesture typing with the support of a statistical decoder. It also enables character-based text entry by using a multi-tap like approach. It divides the layout into seven zones and maintains a resemblance to the standard QWERTY layout. Unlike most existing solutions, it does not occupy most of the touchscreen real-estate or require repeated actions to enter most words. Yet, it employs the whole screen for drawing gestures, which is more comfortable than drawing gestures on a miniature QWERTY. The keyboard is optimized for target selection and to maintain similarities between the gestures drawn on a smartwatch and a virtual QWERTY to facilitate skill transfer between devices. We compared the technique with C-QWERTY, a similar technique that uses almost the same layout to enable gesture typing, but does not divide the keyboard into zones or optimize the zones for target selection and gesture similarity. In the study, SwipeRing yielded a 33% higher entry speed, 56% lower error rate, and 35% lower actions per word than C-QWERTY in the last block. The average entry speed with SwipeRing was 16.67 WPM, faster than all popular ring-shaped and most QWERTY-based text entry techniques for smartwatches. Results indicate towards the possibility that skilled gesture typists were able to transfer their skills to SwipeRing. Besides, participants found the keyboard easy to learn, easy to use, fast, and accurate, thus wanted to continue using it on smartwatches. In the next Chapter, we present three new action-level performance metrics: UnitER, UA, and UnitCX. They account for the error rate, accuracy, and complexity of multi-step chorded and constructive methods.

Chapter 5

Using Action-Level Metrics to Report the Performance of Multi-Step Keyboards

Most existing text entry performance metrics were designed to characterize uni-step methods that map one action (a key press or a tap) to one input. However, there are also multi-step constructive and chorded methods. These require the user to perform a predetermined set of actions either in a specific sequence (pressing multiple keys in a particular order) or simultaneously (pressing multiple keys at the same time). Examples of constructive methods include multi-tap [56] and Morse code [63, 168]. Chorded methods include braille [5] and stenotype [97]. Many text entry methods used for accessibility and to enter text in languages with large alphabets or complex writing systems are either chorded or constructive. However, current metrics for analyzing the performance of text entry techniques were designed for uni-step methods, such as the standard desktop keyboard. Due to the fundamental difference in their input process, these metrics often fail to accurately illustrate participants' actions in user studies evaluating the performance of multi-step methods.

To address this, we present a set of revised and novel performance metrics. They account for the multi-step nature of chorded and constructive text entry techniques by analyzing the actions required to enter a character, rather than the final output. We posit that while conventional metrics are effective in reporting the overall speed and accuracy, a set of action-level metrics can provide extra details about the user's input actions. For designers of multi-step methods, this additional insight is crucial for evaluating the method, identifying its pain points, and facilitating improvements. More specifically, conventional error rates fail to capture learning within chords and sequences. For instance, if entering a letter requires four actions in a specific order, with practice, users may learn some of these actions and the corresponding sub-sequences. Conventional metrics ignore partially correct content by counting each incorrect letter as one error, giving the impression that no learning has occurred. UnitER, in contrast, accounts for this and provides designers with an understanding

Table 5.1: Conventional error rate (ER) and action-level unit error rate (UnitER) for a constructive method (Morse code). This table illustrates the phenomenon of using Morse code to enter "*quickly*" with one character error ("*l*") in each attempt. ER is 14.28% for each attempt. One of our proposed action-level metrics, UnitER, gives a deeper insight by accounting for the entered input sequence. It yields an error rate of 7.14% for the first attempt (with two erroneous dashes), and an improved error rate of 3.57% for the second attempt (with only one erroneous dash). The action-level metric shows that learning has occurred with a minor improvement in error rate, but this phenomenon is omitted in the ER metric, which is the same for both attempts.

								ER (%)	UnitER (%)
Presented Text	q	u	i	c	k	**l**	y	14.28	7.14
	--.-	..-	..	-.-.	-.-	.-..	-.--		
Transcribed Text	q	u	i	c	k	**j**	y		
	--.-	..-	..	-.-.	-.-	.---	-.--		
Presented Text	q	u	i	c	k	**l**	y	14.28	3.57
	--.-	..-	..	-.-.	-.-	.-..	-.--		
Transcribed Text	q	u	i	c	k	**p**	y		
	--.-	..-	..	-.-.	-.-	.--.	-.--		

of (1) whether users learned some actions or not and (2) which actions were difficult to learn, thus can be replaced (Table 5.1).

The remainder of this Chapter is organized as follows. We start with a discussion on the existing, commonly used text entry performance metrics, and then elaborate on our motivations. This is followed with a set of revised and new performance metrics targeted at multi-step input methods. We proceed to validate the metrics in a longitudinal study evaluating one constructive and one chorded input methods. The results demonstrate how the new metrics provide a deeper insight into action-level interactions. Researchers identify factors compromising the performance and learning of a multi-step keyboard, and suggest changes to address the issues.

5.1 Related Work

Conventional metrics for text entry speed include characters per second (CPS) and words per minute (WPM). These metrics represent the number of resulting characters entered, divided by the time spent performing input. Text entry researchers usually transform CPS to WPM by multiplying by a fixed constant (60 seconds, divided by a word length of 5 characters for English text entry) rather than recalculating the metrics [14].

A common error metric, keystrokes per character (KSPC), is the ratio of user actions, such as keystrokes, to the number of characters in the final output [113, 171]. This metric was designed primarily to measure the number of attempts at typing each character accurately [113]. However, many researchers have used it to represent a method's potential entry speed as well [173, 116], since techniques that require fewer actions are usually faster. Some researchers have also customized this metric to fit the need of their user studies. The two most common variants of this metric are gesture per character (GPS) [16, 186, 188] and actions per character (APC) [186], which extend keystrokes to include gestures and other actions, respectively. Error rate (ER) and minimum string distance (MSD) are metrics that measure errors based on the number of incorrect characters in the final output [14]. Another metric, erroneous keystrokes (EKS) [14], considers the number of incorrect actions performed in an input episode. None of these metrics considers error correction efforts, thus Soukoreff and MacKenzie [171] proposed the total error rate (TER) metric that combines two constituent errors metrics: corrected error rate (CER) and not-corrected error rate (NER). The former measures the number of corrected errors in an input episode and the latter measures the number of errors left in the output.

Accessibility text entry systems mainly use constructive or chorded techniques. Both techniques require the user to perform multiple actions for one input, but the difference is in the order of actions. Constructive techniques require the user to perform a combination of actions *sequentially* to enter one character. The chorded techniques require the user to press multiple keys *simultaneously* to enter one character, like playing a chord on a piano. Morse code is a constructive text entry system that was named one of the "Top 10 Accessibility Technologies" by RESNA [149]. In 2018, Android phones integrated this entry system as an accessibility feature for users with motor impairments [182]. Morse code users can tap on one or two keys to enter short sequences of keystrokes representing each letter of the alphabet. This method reduces the dexterity required to enter text, compared to the ubiquitous Qwerty keyboard. Braille is a tactile representation of language used by individuals with visual impairments. Braille keyboards [172, 136, 125] contain just six keys so that users do not need to search for buttons; instead, users can keep their hands resting across the keys. To produce each letter using this system, users must press several of these keys simultaneously.

Word	কান্ড	
Conjunct	কা	ন্ড
Primary Character	ক + আ	ন + ড

Figure 5.1: Text entry methods for languages that have a large alphabet or a complex writing system usually use constructive or chorded techniques. This figure breaks down a Bangla word to its primary characters.

Table 5.2: Performance metrics used to evaluate chorded and constructive keyboards in recent user studies. "ALM" represents action-level metric.

	Technique	Speed	Accuracy	ALM	R^2
Chorded	Twiddler [111]	WPM	ER	*NA*	Yes
	BrailleType [136]	WPM	MSD ER	*NA*	*NA*
	ChordTap [183]	WPM	ER	*NA*	Yes
	Chording Glove [151]	WPM	MSD ER	*NA*	*NA*
	Two-handed [193]	WPM	ER	*NA*	*NA*
Construct.	Mutitap [55]	WPM	ER	*NA*	*NA*
	Reduced Qwerty [60]	WPM	*NA*	*NA*	*NA*
	JustType [121]	WPM	*NA*	*NA*	*NA*
	UOIT [1]	WPM	TER	*NA*	*NA*

Text entry methods for languages that have a large alphabet or a complex writing system also use constructive or chorded techniques. The Bangla language, for example, has approximately 11 vowels, 36 consonants, 10 inflexions, 4 signs, and 10 numeric characters. Additionally, Bangla supports combinations between consonants and consonant and diacritic forms of the vowels. In Fig. 5.1, the top row presents one Bangla word, pronounced kăndō, meaning "stem", composed of two conjoined letters. The first is a combination of a consonant and a vowel, while the second is a combination of two consonants. Due to the large alphabet, Bangla writing systems map many characters to each key of the Qwerty layout, thus requiring the user to disambiguate the input using either the constructive or the chorded techniques. Combining two or more characters also requires performing a combination of actions sequentially or simultaneously. Sarcar et al. [155] proposed a convention for calculating the most common performance metrics for such techniques that considers both the input and the output streams.

Little work has focused on performance metrics for such multi-step input techniques. Grossman et al. [61] defined "soft" and "hard" errors for two-level input techniques, where errors at the first level are considered "soft" and errors at the second level are considered "hard".[1] Seim et al. [158] used a dynamic time warping (DTW) algorithm [76] to measure the accuracy of chorded keystrokes on a piano keyboard. Similar to the Levenshtein distance [101], it measures the similarity between two sequences, but accounts for variants in time or speed. Arif and Stuerzlinger [15] proposed a model for predicting the cost of error correction with uni-step techniques by breaking down each action into low-level motor and cognitive responses. In a follow-up work, Arif [8] discussed how the model can be extended to multi-step techniques and to construct new metrics, however did not develop the metrics

[1]With two-level techniques, users usually perform two actions to select the desired character, for example, the first action to specify the region and the second to choose the character.

or validated them in user studies.

Based on Shannon's information theory [163] and the observation that the speed-accuracy trade-off arises as a predictable feature of communication within humans [170], a method-independent throughput metric was proposed, where the amount of information transmitted via a text entry method per unit time reflects the input efficiency of the method [197].

Some have also used custom conventions for keyboard optimization. Bi et al. [23] optimized a keyboard layout to reduce stylus travel distance for multiple languages. Oulasvirta et al. [138] minimized thumb travel distance for two-thumb text entry on mobile devices. We optimized a smartwatch keyboard (see Chapter 4) to facilitate the transference of gesture typing skill between devices by maintaining similarities between the gestures drawn on them. Some have also proposed new metrics to capture the effectiveness of predictive features, such as auto-correction and word prediction [12], and developed tools to enable action-level analysis of input[2]. Table 5.2 shows the performance metrics used to evaluate recent chorded and constructive text entry techniques.

5.2 Motivation

5.2.1 Partial Errors and Predictions

Users can make partial errors in the process of performing a chord or a sequence of actions. To enter "*x*", a Twiddler [111] user could incorrectly perform the chord "*M0R0*" instead of "*MR00*", and a Morse code user could incorrectly perform the sequence "-..." instead of "-..-" (Table 5.3). In both cases, user actions would be partially correct. However, typical error metrics ignore this detail by considering the complete sequence as one incorrect input. Hence, they yield the same value as when users enter no correct information. In reality, users may have learned, and made fewer mistakes within a chord or a sequence. Not only does this show an incomplete picture of user performance, it also fails to provide the means to fully explore learning of the text entry method. More detailed metrics of multi-step keyboard interaction can facilitate improved input method design through a better understanding of the user experience. These data can also train algorithms to predict and compensate for the most common types of errors.

5.2.2 Correction Effort

Prior research [10, 15] shows that correction effort impacts both performance and user experience, but most current error metrics do not represent the effort required to fix errors with constructive techniques. With uni-step character-based techniques, one error traditionally requires two corrective actions: a backspace to delete the erroneous character, and a keystroke to enter the correct one [15]. Suppose two users want to input "-..-" for the letter "*x*". One user enters "-...", the other enters "...-". Existing error metrics consider both as

[2]A web application to record text entry metrics, https://www.asarif.com/resources/WebTEM

Table 5.3: Performance metrics used to evaluate "x" for chorded (Twiddler) and constructive (Morse) methods.

Technique	Correct input for "x"	Actual input for "x"	ER	UnitER
Twiddler [111]	M**R0**0	M**0R**0	100	50
Morse code [168]	-..-	-...	100	25

one erroneous input. However, correcting these may require different numbers of actions. If a technique allowed users to change the direction of a gesture mid-way [13], the errors would require one and five corrective actions, respectively. If the system only allowed the correction of one action at a time, then the former would require two and the latter would require five corrective actions. In contrast, if the system does not allow the correction of individual actions within a sequence, both would require five corrective actions. Hence, error correction relies on the correction mechanism of a technique, the length of a sequence, and the position and type of error (insertion, deletion, or substitution). Existing metrics fail to capture this vital detail in both chorded and constructive text entry techniques.

5.2.3 Deeper Insights into Limitations

The proposed metrics aim to give quantitative insights into the learning and input of multi-step text entry techniques. Insights, such as tapping locations affecting speed, might seem like common sense, but action-level metrics can also provide similar insights for less straightforward interactions, such as breath puffs, tilts, swipes, etc. Although some findings might seem unsurprising for experts, the proposed metrics will benefit the designers of new techniques aimed at multi-step text entry and complement their efforts and insights.

The new metrics can facilitate design improvements by quantifying the design choices. Conventional metrics identify difficult and erroneous letters, while our UnitER, UA, and UnitCX indicate what is contributing towards it. The action-level metrics can help us to identify difficult-to-enter sequences, so they can be assigned to infrequent characters or avoided altogether. For example, UnitER and UA can reveal target users having difficulty performing the long-press of a three-action input sequence for some character. Designers then can replace the long-press with an easier action (e.g., a physical button press) for a particular group of users to reduce that letter's error rate.

5.3 Notations

In the next section, we propose three new action-level metrics for evaluating multi-step text entry techniques. For this, we use the following notations.

- Presented text (PT) is the text presented to the user for input, $|PT|$ is the length of PT, PT_i is the ith character in PT, pt_i is the sequence of actions required to enter the ith character in PT, and $|pt_i|$ is the length of pt_i.
- Transcribed text (TT) is the text transcribed by the user, $|TT|$ is the length of TT, TT_i is the ith character in TT, tt_i is the sequence of actions performed by the user to enter the ith character in TT, and $|tt_i|$ is the length of tt_i.
- Minimum string distance (MSD) measures the similarity between two sequences using the Levenshtein distance [101]. The "distance" is defined as the minimum number of primitive operations (*insertion, deletion, and substitution*) needed to transform a given sequence (TT) to the target sequence (PT) [113].
- Input time (t) is the time, in seconds, the user took to enter a phrase (TT).
- An action is a user action, including a keystroke, gesture, tap, finger position or posture, etc. Action sequence (AS) is the sequence of all actions required to enter the presented text. |AS| is the length of AS. We consider all sub-actions within a chord as individual actions. For example, if a chord requires pressing three keys simultaneously, then it is composed of three actions.
- We denote a substring (suffix) of a string s starting at the kth character as $s[k:]$. For example, if $s =$ "quickly", substring $s[1:]$ is string "uickly".

5.4 Speed and Accuracy Metrics

5.4.1 Inputs per Second (IPS)

We present IPS, a variant (for convenience of notation) of the commonly used CPS metric [188] to measure the entry speeds of multi-step techniques.

$$\text{IPS} = \frac{|\text{AS}|}{t} \tag{5.1}$$

IPS uses the length of the action sequence $|AS|$ instead of the length of transcribed text $|TT|$ to account for all multi-step input actions. It is particularly useful to find out if the total number of actions needed for input is affecting performance or not.

5.4.2 Actions per Character (APC)

For convenience of notation, we present Actions per Character (APC), a variant of the Keystrokes per Character (KSPC) [14, 113], and the Gesture per Character (GPS) [16, 171] metrics. It measures the average number of actions required to enter one input unit, such as a character or a symbol [188]. One can measure accuracy by comparing the number of

actions required to enter presented text to the number of actions actually performed by participants.

$$\text{APC} = \frac{|\text{AS}|}{|TT|} \tag{5.2}$$

5.5 Proposed Action-Level Metrics

5.5.1 Unit Error Rate (UnitER)

The first of our novel metrics, unit error rate (UnitER) represents the average number of errors committed by the user when entering one input unit, such as a character or a symbol. This metric is calculated in the following three steps:

5.5.1.1 *Step 1: Optimal Alignment*

First, obtain an optimal alignment between the presented (PT) and transcribed text (TT) using a variant of the MSD algorithm [117]. This addresses all instances where lengths of presented ($|PT|$) and transcribed text ($|TT|$) were different.

$$\text{MSD}(a,b) = \begin{cases} |b|, & \text{if } a = \text{“”} \\ |a|, & \text{if } b = \text{“”} \\ 0, & \text{if } a = b, \\ S \end{cases} \tag{5.3}$$

where S is defined as

$$S = \min \begin{cases} \text{MSD}(a[1:\,], b[1:\,] & \text{if } a[0] = b[0], \\ \text{MSD}(a[1:\,], b) + 1, \\ \text{MSD}(a, b[1:\,]) + 1, \\ \text{MSD}(a[1:\,], b[1:\,]) + 1. \end{cases} \tag{5.4}$$

If multiple alignments are possible for a given MSD between two strings, select the one with the least number of insertions and deletions. If there are multiple alignments with the same number of insertions and deletions, then select the first such alignment. For example, if the user enters “*qucehkly*” (TT) instead of the target word “*quickly*” (PT), then MSD(PT,TT) = 3 and the following alignments are possible:

```
quic--kly  quic-kly  qui-ckly  qu-ickly
qu-cehkly  qucehkly  qucehkly  qucehkly
```

Here, a dash in the top sequence represents an *insertion*, a dash in the bottom represents a *deletion*, and different letters in the top and bottom represents a *substitution*. Our algorithm selects the highlighted alignment.

5.5.1.2 *Step 2: Constructive vs. Chorded*

Because the sequence of performed actions is inconsequential in chorded methods, sort both the required (pt_i) and the performed actions (tt_i) using any sorting algorithm to obtain consistent MSD scores. Action sequences are not sorted for constructive methods since the order in which they are performed is vital for such methods.

5.5.1.3 *Step 3: Minimum String Distance*

Finally, apply the MSD algorithm [117] to measure the minimum number of actions needed to correct an incorrect sequence.

$$\text{UnitER} = \frac{\sum_{i=1}^{|\overline{TT}|} \frac{\text{MSD}(pt_i, tt_i)}{\max(|pt_i|, |tt_i|)}}{|\overline{TT}|} \times 100\% \tag{5.5}$$

Here, $|\overline{TT}|$ is the length of the aligned transcribe text (same as $|\overline{PT}|$), and $\overline{pt_i}$ and $\overline{tt_i}$ are the sequence of actions (sorted for chorded techniques in *Step 2*) required and performed, respectively, to enter the ith character in TT.

This step requires certain considerations about the presence of *insertion* and *deletion* in the optimal alignment. If the $i-$th character of the aligned presented text $|\overline{TT}|$ has a deletion, then MSD of the corresponding $i-$th character is 100%. But when $|\overline{PT}|$ has an *insertion*, a misstroke error is assumed, as it is the most common cause of *insertions* [52]. A misstroke errors occur when the user mistakenly strokes (or taps) an incorrect key. However, the question remains: To which character do we attribute the insertion? For this, we propose comparing the MSD-s of current TT_i to the neighboring letters of PT_i (which are different for different layouts), and attributing it to the one with the lowest MSD. If there is a tie, attributed it to the right neighbor PT_i+1. We propose this simplification, since it is difficult to determine the exact cause of an *insertion* in such a scenario.

This metric can be used to measure error rate of a specific character, in which case, Equation 5.6 considers only the character under investigation, where c is the character under investigation and *Total(c)* is the total occurrence of c in the transcribed text.

$$\text{UnitER}(c) = \frac{\sum_{i=1}^{|\overline{TT}|} \frac{\text{MSD}(pt_i, tt_i)}{\max(|pt_i|, |tt_i|)} (\text{if } PT_i{=}c)}{Total(c)} \tag{5.6}$$

5.5.2 Unit Accuracy (UA)

Unit accuracy (UA) is the opposite and simply a convenient re-framing of UnitER. Instead of error rate, UA represents the accuracy rate of a unit. Also, unlike UnitER, the values of UA range between 0 and 1 inclusive to reflect the action-level nature of the metric (i.e., 0%–100%). UA can be used for a specific character c as well, using Equation 5.8.

$$\text{UA} = \frac{100-\text{UnitER}}{100} \tag{5.7}$$

$$\mathrm{UA}(c) = \frac{100 - \mathrm{UnitER}(c)}{100} \tag{5.8}$$

5.5.3 Unit Complexity (UnitCX)

Apart from speed and accuracy, we also propose the following novel metric to measure an input unit's complexity. For this, each action in a unit (such as a character or a symbol) is categorized into different difficulty levels, represented by the continuous values from 0 to 1.

$$\mathrm{UnitCX} = \frac{\left(\sum_{n=1}^{|tt_i|} \frac{d(a_n)}{|TT|}\right) - d_{\min}}{d_{\max} - d_{\min}} \tag{5.9}$$

Here, $d(a_n)$ signifies the difficulty level of the $n-$th action in tt_i, and $d_{\min}$ and $d_{\max}$ are the minimum and maximum difficulty levels of all actions within or between text entry techniques. This yields a normalized unit complexity value, ranging from 0 to 1. The difficulty level of an action is based on a custom convention, considering the posture and ergonomics, memorability, and the frequency of the letters [37]. However, more sophisticated methods are available in the literature [23, 194].

5.6 Experiment: Validation

We validated the effectiveness of our proposed metrics by applying them to data collected from a longitudinal user study. This study evaluated one constructive and one chorded text entry technique. Although we conducted a comparative study, our intent was to demonstrate how the proposed metrics can provide deeper quantitative insights into the multi-step techniques' performance and limitations, specifically with respect to learning, rather than comparing the performance of the two techniques.

5.7 Apparatus

We used a Motorola Moto G^5 Plus smartphone (150.2 × 74 × 7.7 mm, 155 g) at 1080 × 1920 pixels in the study (Fig. 5.4). The virtual multi-step keyboards used in the study were developed using the default Android Studio 3.1, SDK 27. The keyboards logged all user actions with timestamps and calculated all performance metrics directly.

5.8 Constructive Method: Morse Code

We received the original code from the authors of a Morse code keyboard [63] to investigate the performance of a constructive keyboard. It enables users to enter characters using sequences of dots (.) and dashes (-) [168]. The keyboard has dedicated keys for dot, dash, backspace, and space (Fig. 5.2). To enter the letter "R", represented by ".-." in Morse code,

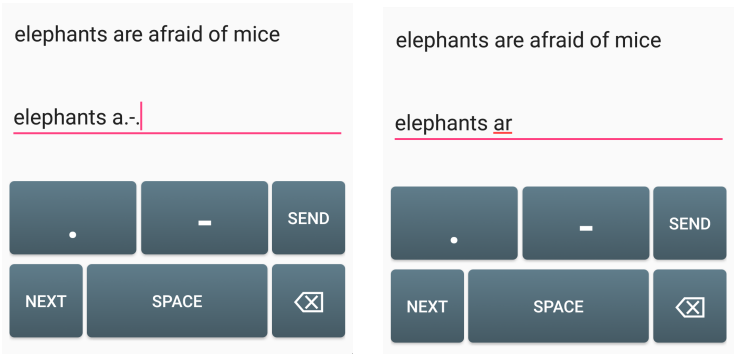


Figure 5.2: The Morse code inspired constructive keyboard used in the experiment.

the user presses the respective keys in that exact sequence, followed by the SEND key, which terminates the input sequence. The user presses the NEXT key to terminate the current phrase and progress to the next one.

5.9 Chorded Method: Senorita

We used code from the chorded keyboard Senorita which is discussed in Chapter 2. It enables users to enter characters using eight keys (Fig. 5.3). The most frequent eight letters in English appear on the top of the key labels, and are entered with only one tap of their respective key. All other letters require simultaneous taps on two keys (with two thumbs). For example, the user taps on the "E" and "T" keys together to enter the letter "H". The keyboard provides visual feedback to facilitate learning. Pressing a key with one thumb highlights all available chords corresponding to that key, and right-thumb keys have a lighter shade than left-thumb keys (Fig. 5.3b).

(a) (b)

Figure 5.3: The Senorita chorded keyboard used in the experiment. It enables users to enter text by pressing two keys simultaneously using the thumbs. (b) Pressing a key highlights all available chords for the key.

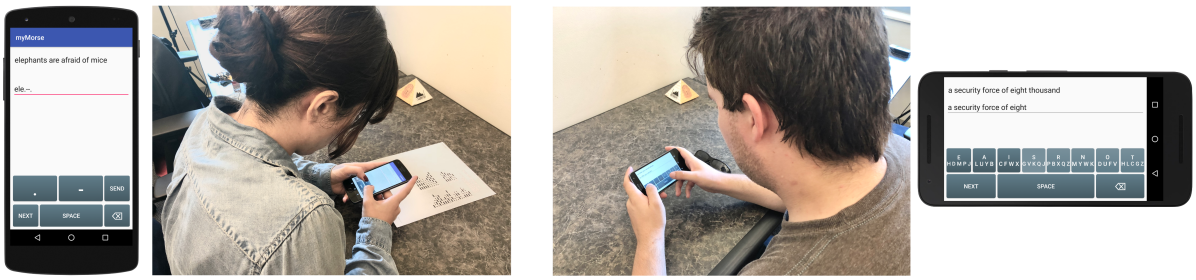
5.10 Participants

Ten participants, aged from 18 to 37 years ($M = 26.1$, $SD = 5.6$), took part in the experiment. Six identified as male, four as female. None identified as non-binary. Eight were right-handed, one was left-handed, and the other was ambidextrous. They all used both hands to hold the device and their thumbs to type. All participants were proficient in English. Six rated themselves as *Level 5: Functionally Native* and four as *Level 4: Advanced Professional* on the Interagency Language Roundtable (ILR) scale [80]. All participants were experienced smartphone users, with an average of 9.3 years' experience ($SD = 1.8$). None of them had prior experience with any chorded methods, but eight had used a constructive method in the past (either multi-tap or pinyin). But none of the participants had experience with the constructive or chorded methods used in the study. They all received US $50 for volunteering.

5.11 Design

We used a within-subjects design, where the independent variables were input method and session. The dependent variables were the IPS, APC, ER, and UnitER metrics. In summary, the design was as follows.

10 participants ×
5 sessions (different days) ×
2 input methods (constructive vs. chorded, counterbalanced) ×
10 English pangrams
= 1,000 phrases, in total.



(a) (b)

Figure 5.4: Two volunteers entering text using: (a) the Morse code constructive keyboard with the assistance of a cheat- sheet, and (b) the Senorita chorded keyboard.

5.12 Procedure

To study learning all letters of the English alphabet, we used the following five pangrams during the experiment, all in lowercase.

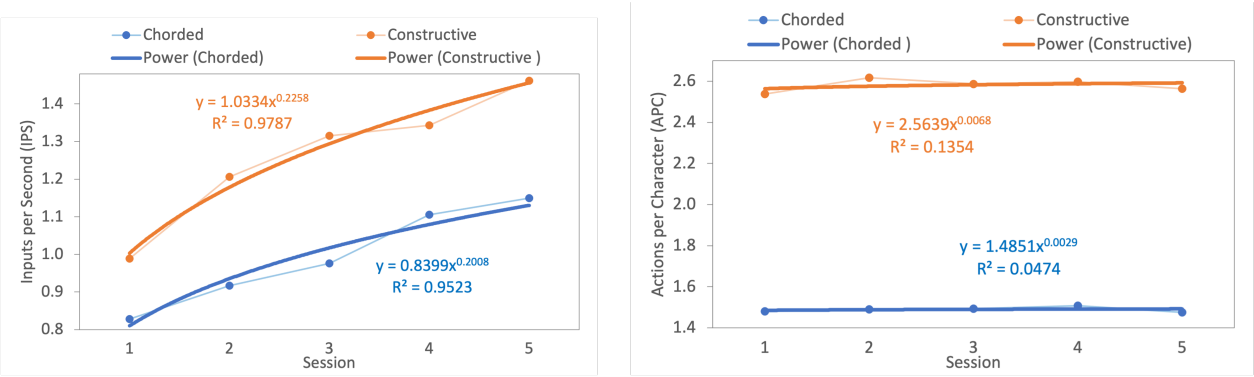
> quick brown fox jumps over the lazy dog
> the five boxing wizards jump quickly
> fix problem quickly with galvanized jets
> pack my red box with five dozen quality jugs
> two driven jocks help fax my big quiz

Participants were divided into two groups, one started with the constructive method and the other started with the chorded method. This order was switched on each subsequent session. Sessions were scheduled on different days, with at most a two-day gap in between. In each session, participants entered one pangram ten times with one method, and then a different pangram ten times with the other method. Pangrams were never repeated for a method. There was a mandatory 30–60 minutes break between the conditions to mitigate the effect of fatigue.

During the first session, we demonstrated both methods to participants and collected their consent forms. We asked them to complete a demographics and mobile usage questionnaire. We allowed them to practice with the methods, where they entered all letters (A–Z) until they felt comfortable with the methods. Participants were provided with a cheat-sheet for Morse code that included all combinations (Fig. 5.4a) as Morse code relies on memorization of those. For Senorita, we did not provide a cheat-sheet as the keyboard provides visual feedback by displaying the characters (i.e., it has a "built-in cheat sheet"). The experiment started shortly after that, where a custom app presented one pangram at a time, and asked participants to enter it "as fast and accurately as possible". Once done, they pressed the NEXT key to re-enter the phrase. Logging started from the first tap and ended with the last. Error correction was intentionally disabled to exclude correction efforts from the data to observe the UnitER metric. Subsequent sessions used the same procedure, excluding practice and questionnaire.

5.13 Results

A Shapiro-Wilk test and a Mauchly's test revealed that the assumption of normality and sphericity were not violated for the data, respectively. Hence, we used a repeated-measures ANOVA for all analysis.



(a) (b)

Figure 5.5: Average IPS for the two methods in all sessions (a) and average APC for the two methods in all sessions (b). Note the scale on the vertical axis.

5.13.1 Inputs per Second (IPS)

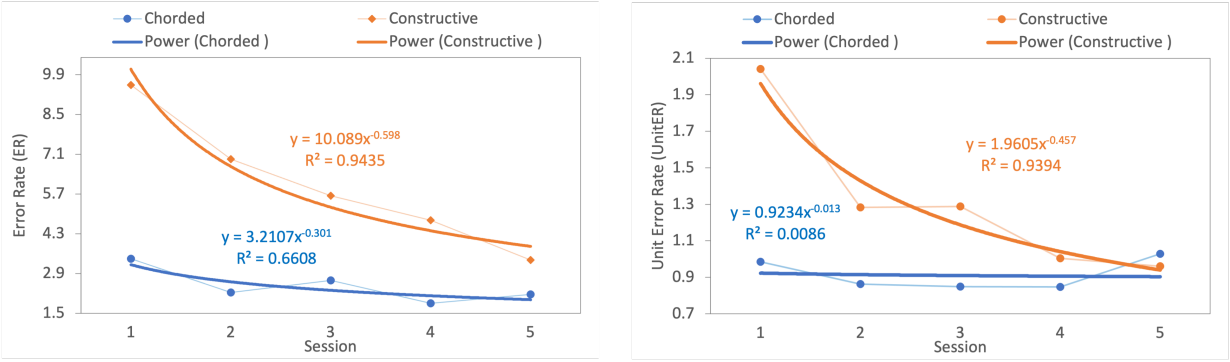
The average IPS for the constructive method was 1.29 ($SD = 0.38$), and for the chorded it was 1.02 ($SD = 0.29$). For the constructive method, IPS increased from 1.02 in session one to 1.49 in session five. This effect was statistically significant ($F_{4,36} = 17.16, p < .0001$). For the chorded method, IPS increased from 0.86 in session one to 1.18 in session five. This effect was also statistically significant ($F_{4,36} = 28.26, p < .0001$). Fig. 5.5a displays IPS for both methods in all sessions. WPM metric also increased throughout all five sessions, which was statistically significant for both the constructive method ($F_{4,36} = 17.32, p < .0001$) and the chorded method ($F_{4,36} = 27.69, p < .0001$).

5.13.2 Actions per Character (APC)

On average constructive and chorded methods yielded 2.58 ($SD = 0.13$) and 1.49 ($SD = 0.02$) APC, respectively. For the constructive method, APC started with 2.54 in session one and ended with 2.56 in session five. This effect was statistically significant ($F_{4,36} = 2.84, p < .05$). For the chorded method, APC started with 1.48 in session one and ended with 1.47 in session five. This result was also statistically significant ($F_{4,36} = 6.52, p < .0005$). Fig. 5.5b displays APC for both methods in all sessions.

5.13.3 Error Rate (ER)

Error rate (ER) is a commonly used error metric, which is traditionally calculated as the ratio of the total number of incorrect characters in the transcribed text to the length of the transcribed text [14]. On average, constructive and chorded methods yielded 6.05% ($SD = 10.11$) and 2.46% ($SD = 4.65$) ER, respectively. As expected, ER improved over the five



(a) (b)

Figure 5.6: Average ER for the two methods in all sessions (a) and average UnitER for the two methods in all sessions (b). Note the scale on the vertical axis.

sessions of testing for both methods. For the constructive method, ER dropped from 9.54% in session one to 3.37% in session five. This effect was statistically significant ($F_{4,36} = 2.86, p < .05$). For the chorded method, ER dropped from 3.42% in session one to 2.16% in session five. However, an ANOVA failed to identify a significant effect of session on ER for the chorded method ($F_{4,36} = 1.79, p = .15$). Fig. 5.6a displays ER for both methods in all sessions. Another widely used error metric, TER [171], yielded comparable result for the constructive method ($F_{4,36} = 3.33, p < .05$) and the chorded method ($F_{4,36} = 1.51, p = .22$).

5.13.4 Unit Error Rate (UnitER)

On average, UnitER for constructive and chorded methods were 1.32 ($SD = 2.2$) and 0.91 ($SD = 2.1$). For the constructive method, UnitER decreased from 2.04 in session one to 0.96 in session five. An ANOVA did not identify a significant effect of session on UnitER for constructive method ($F_{4,36} = 2.14, p = .09$). For the chorded method, UnitER started from 0.98 in session one and ended with 1.03 in session five. An ANOVA did not identify a significant effect of session on UnitER for chorded method as well ($F_{4,36} = 0.12, ns$). Fig. 5.6b displays UnitER for both methods in all sessions.

5.14 Discussion

The IPS metric reflected a statistically significant increase in text entry speed across the five sessions for both input techniques. These results suggest that entry speed improved for both techniques with practice. The standard WPM [14] metric yielded similar statistical results. IPS per session also correlated well with the power law of practice [30] for both constructive ($R^2 = 0.98$) and chorded ($R^2 = 0.95$) methods (Fig. 5.5a).

There was a significant effect of session on APC for the two techniques. Similar trends were observed for UnitCX, and the commonly used KSPC metric. Though average values remain largely consistent (Fig. 5.5b), a Tukey-Kramer Multiple-Comparison test revealed that sessions 1 and 2 were significantly different for constructive, while sessions 1, 4, and 5 were different for chorded. This finding may be due to a difference in group skewness or variance between sessions.

There was a significant reduction in ER from baseline to session five for the constructive technique (Fig. 5.6a). Accordingly, ER per session correlated well with the power law of practice [30] for the constructive method ($R^2 = 0.94$). Similarly, there was a reduction in UnitER from baseline to session five for the constructive method (Fig. 5.6b), though this difference was not statistically significant. UnitER per session also correlated well with the power law of practice [30] for the constructive method ($R^2 = 0.94$). The UnitER metric also provides useful additional details articulated in the following section. Neither ER or UnitER reflected a significant change across sessions for the chorded method. This finding, and the fact that IPS improved over each session for the chorded method, suggest that participants may have learned to type faster without a significant reduction in error rate; perhaps due to the physical nature of a chorded input technique. Additional data is needed to fully investigate this finding.

5.15 Action-Level Analysis

The above discussion demonstrates that the proposed metrics complement conventional metrics by facilitating a discussion of the methods' speed and accuracy in general. In this section, we demonstrate how the proposed metrics shed further light on these methods' performance and limitations.

To investigate which letters may have hampered text entry speed and accuracy, we first calculated UnitER for each letter. Because the letters do not appear the same number of times, we normalized the values based on each letter's total appearance in the study. We then identified the letters that were both difficult to enter and learn to input accurately (henceforth "Not Learned"), and the letters that were difficult to enter but relatively easier to learn to input accurately (henceforth "Learned"). For this, we compared the average UnitER of each letter from the first three sessions to the last two sessions. The letters designated as "Learned" included the letters that demonstrated a significant improvement in UnitER from the first to the last sessions. In contrast, the "Not Learned" included the letters that consistently yielded higher UnitER in all sessions.

5.15.1 Constructive Method

Table 5.4 displays the top four Not Learned and the Learned letters from the Morse code keyboard. For better visualization, we calculated Unit Accuracy (UA) for these letters using

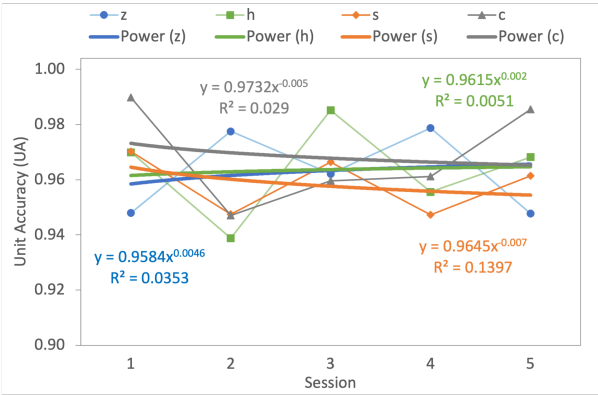
Table 5.4: Action-Level representation of the difficult to enter and learn (Not Learned) and difficult to enter but easier to learn letters (Learned) for the constructive method.

Not Learned	Sequence	Learned	Sequence
z	--..	g	--.
h	....	p	.--.
s	...	x	-..-
c	-.-.	q	--.-

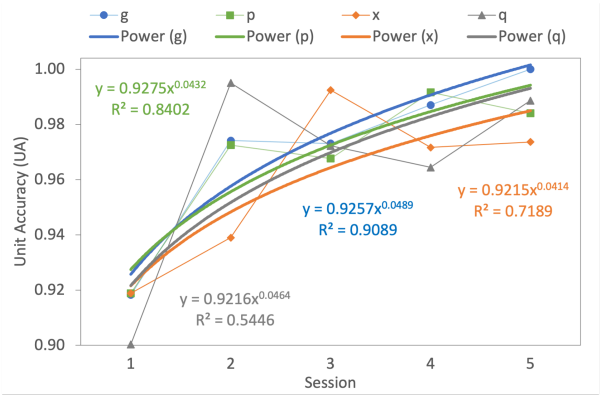
simple transformation given by Eq. 5.8 and fitted them to power law of practice [30] to identify any trends (Fig. 5.7).

Fig. 5.7a illustrates the high inconsistency in UA across the Not Learned letters. However, for the Learned group (Fig. 5.7b), there is a prominent upward trend. Evidently, participants were learning these letters even in the last session, suggesting that performance with these letters would have continued to increase if the study lasted longer. We performed multiple comparisons between the letters to find the cause of entry difficulty, as well as the factors that facilitated learning. We identified the following trends that may have contributed towards the aforementioned trends.

The analysis revealed that participants were having difficulty differentiating between the letters that required similar actions to enter. For instance, it was difficult for them to



(a)



(b)

Figure 5.7: (a) Average Unit Accuracy (UA) of the letters “z”, “h”, “s”, “c” per session with power trendlines. Trends for the letters show decreasing UA across sessions, indicating that learning was not occuring, (b) average UA of the letters “g”, “p”, “x”, “q” per session with power trendlines. Trends for these letters show increasing UA as the sessions progressed, indicating learning.

differentiate between "h" ("....") and "s" ("..."), and "k" ("-.-") and "c" ("-.-."), presumably, because their corresponding actions are very similar. A deeper analysis of the action-level errors revealed that participants frequently entered an extra dot when typing "k", resulting in the letter "c", and vice versa. This error totaled 23% of all UnitER for "c". Participants also made similar mistakes for other Not Learned letters. For example, they entered "s" ("...") instead of "h" ("....") and vice versa, resulting in 50% of all UnitER for "h" and 30% of all UnitER for "s". This information is vital. It helps the designer of a constructive method assign sufficiently different actions to frequent letters to avoid confusion, likely resulting in increased speed and accuracy.

Interestingly, participants tend to enter an extra dot or dash when these actions are repeated. For example, participants often entered "---.", "---.-" or similar patterns for z ("--.."), resulting in 17% of all UnitER in "z". Likewise, participants entered additional dots when entering g ("--."), such as z ("--.."), which resulted in 20% of all UnitER for "g". Similar errors were committed for other letters as well. These details are useful, since the designer can compensate by reducing the number of repeated actions for frequent letters.

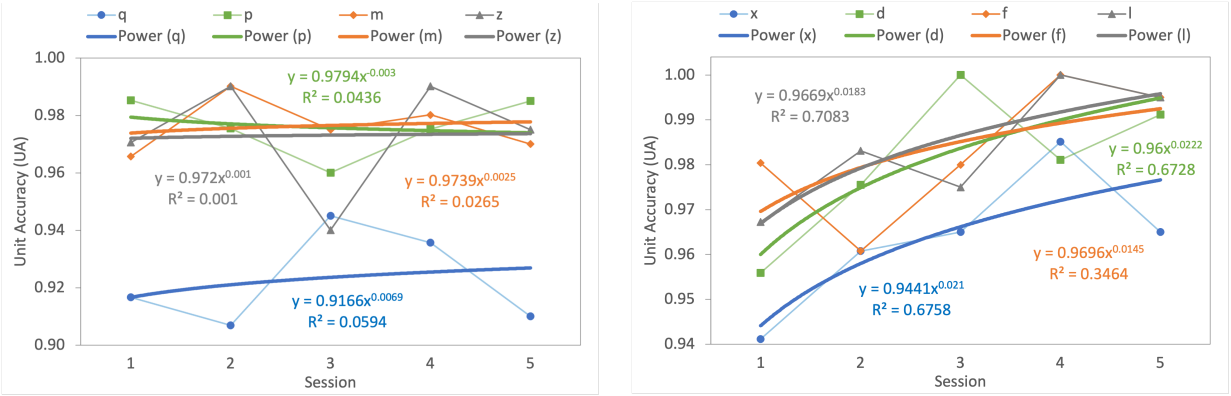
5.15.2 Chorded Method

Table 5.5 displays the top four Not Learned and the Learned letters for the Senorita keyboard. Like the constructive method, we calculated UA for these letters using Eq. 5.8 and fitted them to power law of practice [30] to identify any trends, see Fig. 5.8. We observed that, like the constructive method, trends for all Learned letters were increasing, suggesting the occurrence of learning across the techniques. Participants were learning these letters even in the last session. This might indicate that the performance of these letters would have been much better if the study lasted longer. Multiple comparisons between action-level actions per letter identified the following trends that may have contributed towards the aforementioned trends.

Letters "q", "p", and "z" have the "r" key in their chords ("r", "er", and "rt", respectively), which is the furthest key from the right side. We speculate that it is difficult to learn, and be fast and accurate if letter has "r" in a chord pair, because "r" and "s" are

Table 5.5: Action-Level representation of the difficult to enter and learn (Not Learned) and difficult to enter but easier to learn letters (Learned) for the chorded method.

Not Learned letter	Chord	Learned letter	Chord
q	*rs*	x	*ir*
p	*er*	d	*eo*
m	*en*	f	*io*
z	*rt*	l	*at*



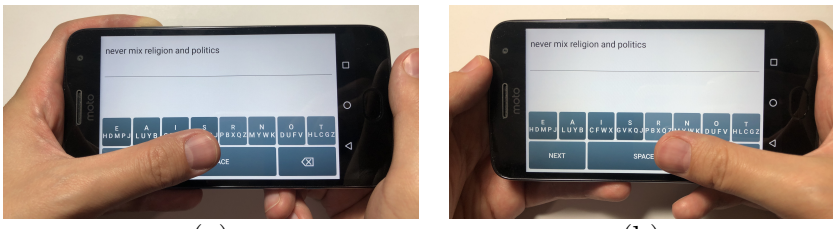
(a) (b)

Figure 5.8: (a) Average Unit Accuracy (UA) of the letters "q", "p", "m", "z" per session with power trendlines. Trends for the letters show decreasing UA across sessions, indicating that learning was not occurring, (b) average UA of the letters "x", "d", "f", "l" per session with power trendlines. Trends for these letters show increasing UA as the sessions progressed, indicating learning.

the furthest letters from the edges (Fig. 5.9). Keys near the center of the screen are more difficult to reach than those at the edge. Relevantly, four letters with improving trendlines ("x", "d", "f", and "l") have chord pairs that are close to the screen edge. This detail may encourage designers to place the most frequent letters toward the edge of the device.

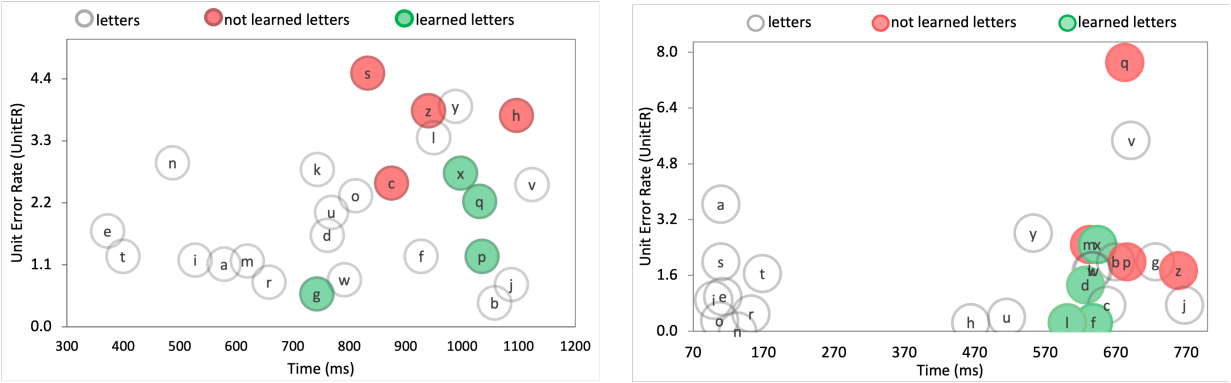
5.15.3 Time Spent per Letter

We compared the above analysis with the time spent to perform the sequences and chords for each letter in the last two sessions. The purpose was to further validate the metrics by studying if the letters that took additional time to enter correspond well to the Not Learned letters. Fig. 5.10a illustrates all difficult letters, highlighted in red (Not Learned) and green



(a) (b)

Figure 5.9: The user stretching her thumb to reach the letter "s" (a) and the letter "r" (b).



(a) (b)

Figure 5.10: (a) Average UnitER of all letters vs. the time spent to perform the sequence for those letters for constructive method, (b) average UnitER of all letters vs. the time spent to perform the chords for those letters with the chorded method.

(Learned), identified for the constructive method. One can see the correlation between UnitER and the time spent to enter the sequence of these letters. This suggests that these letters require higher cognitive and motor effort to enter. Similarly, Fig. 5.10b illustrates all difficult letters, highlighted in red (Not Learned) and green (Learned), identified for the chorded method. One can see that the chords ("q, "v", "p", "g", "z", "j") required more time than taps ("s", "e", "n", "o", "r", "i", "t", "a"). However, the chords that are composed of the keys closer to the boundaries were much faster (e.g., "h", "u", "l", etc.). This further strengthens the argument of the previous section.

5.16 Conclusion

In this Chapter, we proposed a trio of action-level performance metrics (UnitER, UA, and UnitCX) aimed at multi-step text entry techniques that account and compensate for the constructive and chorded input process. We validated these metrics in a longitudinal study involving one constructive and one chorded technique. In our presentation, we used existing techniques such as Morse code [63, 168] and Senorita (see Chapter 2), not to change their mapping or design, but to amply demonstrate how can the newly proposed metrics be applied for different multi-step techniques and to give a deeper insight into possible limitations of the techniques with an emphasis on learning. None of the metrics would give the conclusions automatically, but they could point towards limitations. Our UnitER helped us to investigate specific erroneous characters, while conventional metrics failed to identify them. The results of this study demonstrate how the proposed metrics provide a deeper understanding of action-level error behaviors, particularly the difficulties in performing and learning the sequences

and chords of the letters, facilitating the design of faster and more accurate multi-step keyboards. Although there was previously no formal method to analyze action-level actions of the multi-step method, it is likely that veteran text entry researchers perform similar analysis on user study data. This work provides a formal method that will enable researchers new to the area to perform these analyses, and facilitate better comparison between methods from the literature. In the final Chapter, we summarize the dissertation and discuss possible future directions of the projects.

Chapter 6

Conclusion and Future Work

In this dissertation, we designed and evaluated unconventional modes of interaction for people with vision and motor impairments. Our main goal was to design inclusive text entry methods using mainstream devices: to reduce the burden of expenses that people with disabilities experience. The methods that we used relied on chording, constructive techniques, and physical buttons. The design of our techniques is motivated by the "design for all" philosophy [156, 169] to accommodate the maximum possible group of users: people with vision impairments, motor impairments, and non-disabled people. We first presented a novel chorded virtual keyboard that enables sighted, low vision, and blind users to enter text on tablets and smartphones. The letters on the keyboard layout are arranged on eight keys in a single row. Its compact design leaves most of the screen available, and its position near the edge accommodates eyes-free text entry by using the bezel as a physical reference. In the longitudinal study, it yielded 3.7 words per minute with blind users, surpassing their Qwerty performance by 32%. Low vision users yielded 5.8 words per minute. We then discussed uni-finger text entry on smartwatches for people with limited dexterity that aims to reduce the physical load on the motor function. Since people with limited dexterity are unable to use multi-key approaches like Qwerty to enter text, we proposed a keyboard that supports single-button interaction, where keys are selected using the digital crown of the smartwatch. A final study with ten representative users revealed that both automated and manual rotations were faster and more accurate than the default Qwerty method on smartwatches. With manual rotation being slightly faster than automatic clockwise rotation (2.6 vs 2.3 wpm) and more accurate (0.0% vs. 0.45% er). Next, we showed a novel virtual keyboard that arranges the Qwerty layout around the bezel of a smartwatch and segments it into seven zones to enable gesture typing on smartwatches. The zones are rigorously optimized for usability and for similarities between the gestures drawn on smartphones and smartwatches to facilitate skill transfer between these devices. In an empirical study, the proposed solution reached a competitive entry speed of 17 words per minute. It was 33% faster than the state-of-the-art. Besides, users who had experience with gesture typing on smartphones performed much better with it, indicating skill transfer. Finally, we presented

a project that focused on the theory of text entry. Conventional metrics provide a high-level perspective of a method's performance, namely its speed (words per minute) and accuracy (error rate). However, conventional metrics fail to capture the action-level error committed by users and the difficult-to-learn and difficult-to-perform actions. To address this, we developed novel action-level performance metrics, and demonstrated in two longitudinal studies that they can measure and provide insights into action-level input errors and complexity.

There are multiple possible extensions for our works. For instance, one of the possible future directions would be combining existing accessibility methods (e.g., Braille) with our novel text-entry methods. It would be interesting to employ braille principles in our chorded keyboard to see if novices can learn braille. This work also could be augmented with Passive Haptic Learning (PHL) to enable people with visual impairments to learn braille without distracting them from typing flow. A Braille keyboard augmented with haptic feedback, particularly vibration, might stimulate people with vision impairments to learn braille passively. Another potential direction would be exploring text editing on mobile devices. Text editing on touchscreens is difficult due to the challenges with direct cursor positioning and cut/copy-paste actions. Besides, the trend of increasing the display size makes reaching smaller targets more difficult and sometimes impossible since the reachability of our fingers remains the same. Thus, it is crucial to design user-friendly and effective text editing methods that can be used on various touchscreen devices, including smartphones, tablets, and large displays. Finally, many works designed interaction systems to assist people with motor impairments on computers and smartphones, however, only a few works addressed this issue on smartwatches. Due to compactness, closeness (they are right on the wrist), and being lightweight, smartwatches can be used as a replacement for smartphones in situations where using smartphones is either difficult or impractical. They can also provide people with upper body motor impairments access to mobile systems as tasks like pulling a phone from the pocket can be challenging to them. The absence of an efficient text entry technique for these devices aimed at such situations and population severely limits smartwatch usage. Thus, it is essential to provide people with upper body motor impairments with user-friendly, comfortable text entry methods on smartwatches. Particularly, one possible direction is to explore alternative smartwatch interactions, e.g., mid-air gestures, on-body gestures, and gestures performed on non-touchscreen locations.

Bibliography

[1] T. AbuHmed, K. Lee, and D. Nyang. "UOIT Keyboard: A Constructive Keyboard for Small Touchscreen Devices." In: *IEEE Transactions on Human-Machine Systems* 45.6 (Dec. 2015), pp. 782–789. ISSN: 2168-2305. DOI: 10.1109/THMS.2015.2449309.

[2] M. A. Alamdar Yazdi, A. Negahban, L. Cavuoto, and F. M. Megahed. "Optimization of Split Keyboard Design for Touchscreen Devices." In: *International Journal of Human-Computer Interaction* 35.6 (2019), pp. 468–477. DOI: 10.1080/10447318.2018.1464255.

[3] M. Alnfiai and S. Sampalli. "An Evaluation of SingleTapBraille Keyboard: A Text Entry Method that Utilizes Braille Patterns on Touchscreen Devices." In: *Proceedings of the 18th International ACM SIGACCESS Conference on Computers and Accessibility*. ASSETS '16. Reno, Nevada, USA: ACM, 2016, pp. 161–169. ISBN: 978-1-4503-4124-0. DOI: 10.1145/2982142.2982161.

[4] M. Alnfiai and S. Sampalli. "BrailleEnter: A Touch Screen Braille Text Entry Method for the Blind." In: *Procedia Computer Science* 109 (Dec. 2017), pp. 257–264. DOI: 10.1016/j.procs.2017.05.349.

[5] American Foundation for the Blind. *What Is Braille?* 2020. URL: https://web.archive.org/web/20220628060733/https://www.afb.org/blindness-and-low-vision/braille/what-braille.

[6] Android Developers. *TextToSpeech*. 2017. URL: https://developer.android.com/reference/android/speech/tts/TextToSpeech.

[7] L. Anthony, Y. Kim, and L. Findlater. "Analyzing User-Generated Youtube Videos to Understand Touchscreen Use by People with Motor Impairments." In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. CHI '13. Paris, France: Association for Computing Machinery, 2013, pp. 1223–1232. ISBN: 9781450318990. DOI: 10.1145/2470654.2466158.

[8] A. S. Arif. "Metrics for Multi-Touch Input Technologies." In: (2010). arXiv: 2009.13219. URL: http://arxiv.org/abs/2009.13219.

[9] A. S. Arif. "Predicting and Reducing the Impact of Errors in Character-Based Text Entry." PhD thesis. 2013.

[10] A. S. Arif, S. Kim, W. Stuerzlinger, G. Lee, and A. Mazalek. "Evaluation of a Smart-Restorable Backspace Technique to Facilitate Text Entry Error Correction." In: *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems*. CHI '16. San Jose, California, USA: Association for Computing Machinery, 2016, pp. 5151–5162. ISBN: 9781450333627. DOI: 10.1145/2858036.2858407.

[11] A. S. Arif and A. Mazalek. "A Survey of Text Entry Techniques for Smartwatches." en. In: *Human-Computer Interaction. Interaction Platforms and Techniques*. Ed. by M. Kurosu. Lecture Notes in Computer Science. Cham: Springer International Publishing, 2016, pp. 255–267. ISBN: 978-3-319-39516-6. DOI: 10.1007/978-3-319-39516-6_24.

[12] A. S. Arif and A. Mazalek. "WebTEM: A Web Application to Record Text Entry Metrics." In: *Proceedings of the 2016 ACM International Conference on Interactive Surfaces and Spaces*. ISS '16. Niagara Falls, Ontario, Canada: ACM, 2016, pp. 415–420. ISBN: 978-1-4503-4248-3. DOI: 10.1145/2992154.2996791.

[13] A. S. Arif, M. Pahud, K. Hinckley, and B. Buxton. "Experimental Study of Stroke Shortcuts for a Touchscreen Keyboard with Gesture-Redundant Keys Removed." In: *GI '14 Proceedings of Graphics Interface 2014*. Canadian Information Processing Society, May 2014, pp. 43–50. ISBN: 978-1-4822-6003-8. URL: https://www.microsoft.com/en-us/research/publication/experimental-study-stroke-shortcuts-touchscreen-keyboard-gesture-redundant-keys-removed/.

[14] A. S. Arif and W. Stuerzlinger. "Analysis of Text Entry Performance Metrics." In: *2009 IEEE Toronto International Conference Science and Technology for Humanity (TIC-STH)*. Sept. 2009, pp. 100–105. DOI: 10.1109/TIC-STH.2009.5444533.

[15] A. S. Arif and W. Stuerzlinger. "Predicting the Cost of Error Correction in Character-Based Text Entry Technologies." In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. CHI '10. Atlanta, Georgia, USA: Association for Computing Machinery, 2010, pp. 5–14. ISBN: 9781605589299. DOI: 10.1145/1753326.1753329.

[16] A. S. Arif and W. Stuerzlinger. "User Adaptation to a Faulty Unistroke-Based Text Entry Technique by Switching to an Alternative Gesture Set." In: *Proceedings of Graphics Interface 2014*. GI '14. Toronto, Ont., Canada: Canadian Information Processing Society, 2014, pp. 183–192. ISBN: 978-1-4822-6003-8. URL: http://dl.acm.org/citation.cfm?id=2619648.2619679.

[17] B. Ashtiani and I. S. MacKenzie. "BlinkWrite2: An Improved Text Entry Method Using Eye Blinks." In: *Proceedings of the 2010 Symposium on Eye-Tracking Research; Applications*. ETRA '10. Austin, Texas: Association for Computing Machinery, 2010, pp. 339–345. ISBN: 9781605589947. DOI: 10.1145/1743666.1743742.

[18] S. Azenkot and S. Zhai. "Touch Behavior with Different Postures on Soft Smartphone Keyboards." In: *Proceedings of the 14th International Conference on Human-computer Interaction with Mobile Devices and Services*. MobileHCI '12. San Francisco, California, USA: ACM, 2012, pp. 251–260. ISBN: 978-1-4503-1105-2. DOI: 10.1145/2371574.2371612.

[19] M. Baljko and A. Tam. "Indirect Text Entry Using One or Two Keys." In: *Proceedings of the 8th International ACM SIGACCESS Conference on Computers and Accessibility*. Assets '06. Portland, Oregon, USA: Association for Computing Machinery, 2006, pp. 18–25. ISBN: 1595932909. DOI: 10.1145/1168987.1168992.

[20] N. Banovic, K. Yatani, and K. N. Truong. "Escape-Keyboard: A Sight-Free One-Handed Text Entry Method for Mobile Touch-Screen Devices." In: *International Journal of Mobile Human Computer Interaction (IJMHCI)* 5.3 (2013), pp. 42–61. DOI: 10.4018/jmhci.2013070103.

[21] M. Belatar and F. Poirier. "Text Entry for Mobile Devices and Users with Severe Motor Impairments: Handiglyph, a Primitive Shapes Based Onscreen Keyboard." In: *Proceedings of the 10th International ACM SIGACCESS Conference on Computers and Accessibility*. Assets '08. Halifax, Nova Scotia, Canada: Association for Computing Machinery, 2008, pp. 209–216. ISBN: 9781595939760. DOI: 10.1145/1414471.1414510.

[22] X. Bi, C. Chelba, T. Ouyang, K. Partridge, and S. Zhai. "Bimanual Gesture Keyboard." In: *Proceedings of the 25th Annual ACM Symposium on User Interface Software and Technology*. UIST '12. Cambridge, Massachusetts, USA: ACM, 2012, pp. 137–146. ISBN: 978-1-4503-1580-7. DOI: 10.1145/2380116.2380136.

[23] X. Bi, B. A. Smith, and S. Zhai. "Multilingual Touchscreen Keyboard Design and Optimization." In: *Human–Computer Interaction* 27.4 (2012), pp. 352–382. DOI: 10.1080/07370024.2012.678241.

[24] X. Bi and S. Zhai. "IJQwerty: What Difference Does One Key Change Make? Gesture Typing Keyboard Optimization Bounded by One Key Position Change from Qwerty." In: *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems*. CHI '16. San Jose, California, USA: ACM, 2016, pp. 49–58. ISBN: 978-1-4503-3362-7. DOI: 10.1145/2858036.2858421.

[25] S. M. Billah, Y.-J. Ko, V. Ashok, X. Bi, and I. Ramakrishnan. "Accessible Gesture Typing for Non-Visual Text Entry on Smartphones." In: *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*. CHI '19. New York, NY, USA: ACM, 2019, 376:1–376:12. ISBN: 978-1-4503-5970-2. DOI: 10.1145/3290605.3300606.

[26] N. F. of the Blind (NFB). *The Braille Literacy Crisis in America*. Tech. rep. Baltimore, MD, USA: National Federation of the Blind (NFB), 2009. URL: https://web.archive.org/web/20220711024415/https://nfb.org/images/nfb/documents/pdf/braille_literacy_report_web.pdf.

[27] M. N. Bonner, J. T. Brudvik, G. D. Abowd, and W. K. Edwards. "No-Look Notes: Accessible Eyes-Free Multi-Touch Text Entry." In: *International Conference on Pervasive Computing*. Springer. 2010, pp. 409–426. DOI: 10.1007/978-3-642-12654-3_24.

[28] M. C. Buzzi, M. Buzzi, B. Leporini, and A. Trujillo. "Designing a Text Entry Multimodal Keypad for Blind Users of Touchscreen Mobile Phones." In: *Proceedings of the 16th International ACM SIGACCESS Conference on Computers; Accessibility*. ASSETS '14. Rochester, New York, USA, 2014, pp. 131–136. ISBN: 9781450327206. DOI: 10.1145/2661334.2661354.

[29] R. A. D. Cahya, A. N. Handayani, and A. P. Wibawa. "Mobile Braille Touch Application for Visually Impaired People using Double Diamond Approach." In: *MATEC Web of Conferences*. Vol. 197. EDP Sciences. 2018, p. 15007. DOI: 10.1051/matecconf/201819715007.

[30] S. K. Card, T. P. Moran, and A. Newell. *The Psychology of Human-Computer Interaction*. en-US. CRC Press, 1983. ISBN: 0-89859-859-1.

[31] J. M. Carroll and C. Carrithers. "Training Wheels in a User Interface." In: *Commun. ACM* 27.8 (Aug. 1984), pp. 800–806. ISSN: 0001-0782. DOI: 10.1145/358198.358218.

[32] J. M. Carroll and M. B. Rosson. "Paradox of the Active User." In: *Interfacing Thought: Cognitive Aspects of Human-Computer interaction*. Ed. by J. M. Carroll. 1st ed. Cambridge, MA, USA: MIT Press, 1986. Chap. 5, pp. 80–111.

[33] S. J. Castellucci, I. S. MacKenzie, M. Misra, L. Pandey, and A. S. Arif. "TiltWriter: Design and Evaluation of a No-Touch Tilt-Based Text Entry Method for Handheld Devices." In: *Proceedings of the 18th International Conference on Mobile and Ubiquitous Multimedia*. MUM '19. Article 7. ACM, Nov. 2019, pp. 1–8. ISBN: 978-1-4503-7624-2. DOI: 10.1145/3365610.3365629.

[34] K. B. Chen, A. B. Savage, A. O. Chourasia, D. A. Wiegmann, and M. E. Sesto. "Touch Screen Performance by Individuals With and Without Motor Control Disabilities." In: *Applied Ergonomics* 44.2 (2013), pp. 297–302. ISSN: 0003-6870. DOI: 10.1016/j.apergo.2012.08.004.

[35] X. A. Chen, T. Grossman, and G. Fitzmaurice. "Swipeboard: A Text Entry Technique for Ultra-Small Interfaces That Supports Novice to Expert Transitions." In: *Proceedings of the 27th Annual Acm Symposium on User Interface Software and Technology*. UIST '14. ACM, Oct. 2014, pp. 615–620. ISBN: 978-1-4503-3069-5. DOI: 10.1145/2642918.2647354.

[36] B. C. Clark, A. J. Woods, L. A. Clark, C. R. Criss, R. Shadmehr, and D. R. Grooms. "The Aging Brain and The Dorsal Basal Ganglia: Implications for Age-Related Limitations of Mobility." In: *Advances in geriatric medicine and research* 1 (2019). DOI: 10.20900/agmr20190008.

[37] T. M. E. Club. *Digraph Frequency (based on a sample of 40,000 words)*. 2020. URL: https://web.archive.org/web/20220401024310/http://pi.math.cornell.edu/~mec/2003-2004/cryptography/subs/digraphs.html.

[38] COCA. *N-Grams Based on 520 Million Word COCA Corpus*. Dec. 2019. URL: https://web.archive.org/web/20220727011121/https://www.ngrams.info/.

[39] G. Costagliola, M. Rosa, R. D'Arco, S. Gregorio, V. Fuccella, and D. Lupo. "C-QWERTY: A Text Entry Method for Circular Smartwatches (S)." In: *DMSVIVA*. July 2019, pp. 51–57. DOI: 10.18293/DMSVIVA2019-014.

[40] L. Deng and X. Huang. "Challenges in Adopting Speech Recognition." In: *Commun. ACM* 47.1 (Jan. 2004), pp. 69–75. ISSN: 0001-0782. DOI: 10.1145/962081.962108.

[41] W. Diao, X. Liu, Z. Zhou, and K. Zhang. "Your Voice Assistant is Mine: How to Abuse Speakers to Steal Information and Control Your Phone." In: SPSM '14. Scottsdale, Arizona, USA: Association for Computing Machinery, 2014, pp. 63–74. ISBN: 9781450331555. DOI: 10.1145/2666620.2666623.

[42] I. L. Dryden and K. V. Mardia. *Statistical Shape Analysis, with Applications in R. Second Edition*. Chichester: John Wiley and Sons, 2016.

[43] S. N. Duff, C. B. Irwin, J. L. Skye, M. E. Sesto, and D. A. Wiegmann. "The Effect of Disability and Approach on Touch Screen Performance during a Number Entry Task." In: *Proceedings of the Human Factors and Ergonomics Society Annual Meeting* 54.6 (2010), pp. 566–570. DOI: 10.1177/154193121005400605.

[44] M. D. Dunlop and A. Crossan. "Predictive Text Entry Methods for Mobile Phones." en. In: *Personal Technologies* 4.2 (June 2000), pp. 134–143. ISSN: 1617-4917. DOI: 10.1007/BF01324120.

[45] M. D. Dunlop, A. Komninos, and N. Durga. "Towards High Quality Text Entry on Smartwatches." In: *CHI '14 Extended Abstracts on Human Factors in Computing Systems*. CHI EA '14. Toronto, Ontario, Canada: Association for Computing Machinery, Apr. 2014, pp. 2365–2370. ISBN: 978-1-4503-2474-8. DOI: 10.1145/2559206.2581319.

[46] A. Easwara Moorthy and K.-P. L. Vu. "Privacy Concerns for Use of Voice Activated Personal Assistant in the Public Space." In: *International Journal of Human-Computer Interaction* 31.4 (2015), pp. 307–335. DOI: 10.1080/10447318.2014.986642.

[47] A. Edwards et al. *Extraordinary Human-Computer Interaction: Interfaces for Users with Disabilities*. Vol. 7. CUP Archive, 1995.

[48] T. Felzer, I. S. MacKenzie, P. Beckerle, and S. Rinderknecht. "Qanti: A Software Tool for Quick Ambiguous Non-standard Text Input." In: *Computers Helping People with Special Needs*. Ed. by K. Miesenberger, J. Klaus, W. Zagler, and A. Karshmer. Berlin, Heidelberg: Springer Berlin Heidelberg, 2010, pp. 128–135. ISBN: 978-3-642-14100-3. DOI: 10.1007/978-3-642-14100-3_20.

[49] T. Felzer and S. Rinderknecht. "3DScan: An Environment Control System Supporting Persons with Severe Motor Impairments." In: *Proceedings of the 11th International ACM SIGACCESS Conference on Computers and Accessibility*. Assets '09. Pittsburgh, Pennsylvania, USA: Association for Computing Machinery, 2009, pp. 213–214. ISBN: 9781605585581. DOI: 10.1145/1639642.1639681.

[50] L. Findlater, K. Moffatt, J. E. Froehlich, M. Malu, and J. Zhang. "Comparing Touchscreen and Mouse Input Performance by People With and Without Upper Body Motor Impairments." In: *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems*. CHI '17. Denver, Colorado, USA: Association for Computing Machinery, 2017, pp. 6056–6061. ISBN: 9781450346559. DOI: 10.1145/3025453.3025603.

[51] A. Fowler, K. Partridge, C. Chelba, X. Bi, T. Ouyang, and S. Zhai. "Effects of Language Modeling and Its Personalization on Touchscreen Typing Performance." In: *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems*. CHI '15. Seoul, Republic of Korea: ACM, 2015, pp. 649–658. ISBN: 978-1-4503-3145-6. DOI: 10.1145/2702123.2702503.

[52] D. R. Gentner, J. T. Grudin, S. Larochelle, D. A. Norman, and D. E. Rumelhart. "A Glossary of Terms Including a Classification of Typing Errors." In: *Cognitive Aspects of Skilled Typewriting*. Ed. by W. E. Cooper. New York, NY: Springer New York, 1983, pp. 39–43. ISBN: 978-1-4612-5470-6. DOI: 10.1007/978-1-4612-5470-6_2.

[53] K. Go, M. Kikawa, Y. Kinoshita, and X. Mao. "Eyes-Free Text Entry with EdgeWrite Alphabets for Round-Face Smartwatches." In: *2019 International Conference on Cyberworlds (CW)*. Oct. 2019, pp. 183–186. DOI: 10.1109/CW.2019.00037.

[54] M. Goel, A. Jansen, T. Mandel, S. N. Patel, and J. O. Wobbrock. "ContextType: Using Hand Posture Information to Improve Mobile Touch Screen Text Entry." In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. CHI '13. Paris, France: ACM, 2013, pp. 2795–2798. ISBN: 978-1-4503-1899-0. DOI: 10.1145/2470654.2481386.

[55] J. Gong, B. Haggerty, and P. Tarasewich. "An Enhanced Multitap Text Entry Method with Predictive Next-Letter Highlighting." In: *CHI '05 Extended Abstracts on Human Factors in Computing Systems*. CHI EA '05. Portland, OR, USA: Association for Computing Machinery, 2005, pp. 1399–1402. ISBN: 1595930027. DOI: 10.1145/1056808.1056926. URL: https://doi.org/10.1145/1056808.1056926.

[56] J. Gong and P. Tarasewich. "Alphabetically Constrained Keypad Designs for Text Entry on Mobile Devices." In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. CHI '05. Portland, Oregon, USA: Association for Computing Machinery, 2005, pp. 211–220. ISBN: 1581139985. DOI: 10.1145/1054972.1055002.

[57] J. Gong, Z. Xu, Q. Guo, T. Seyed, X. '. Chen, X. Bi, and X.-D. Yang. "WrisText: One-Handed Text Entry on Smartwatch Using Wrist Gestures." In: *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*. CHI '18. New York, NY, USA: ACM, 2018, 181:1–181:14. ISBN: 978-1-4503-5620-6. DOI: 10.1145/3173574.3173755.

[58] M. Gordon, T. Ouyang, and S. Zhai. "WatchWriter: Tap and Gesture Typing on a Smartwatch Miniature Keyboard with Statistical Decoding." In: *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems*. CHI '16. New York, NY, USA: ACM, 2016, pp. 3817–3821. ISBN: 978-1-4503-3362-7. DOI: 10.1145/2858036.2858242.

[59] T. Götzelmann and P.-P. Vázquez. "InclineType: An Accelerometer-Based Typing Approach for Smartwatches." In: *Proceedings of the XVI International Conference on Human Computer Interaction*. Interacción '15. Vilanova i la Geltru, Spain: Association for Computing Machinery, Sept. 2015, pp. 1–4. ISBN: 978-1-4503-3463-1. DOI: 10.1145/2829875.2829929.

[60] N. Green, J. Kruger, C. Faldu, and R. St. Amant. "A Reduced QWERTY Keyboard for Mobile Text Entry." In: *CHI '04 Extended Abstracts on Human Factors in Computing Systems*. CHI EA '04. Vienna, Austria: ACM, 2004, pp. 1429–1432. ISBN: 1-58113-703-6. DOI: 10.1145/985921.986082.

[61] T. Grossman, X. A. Chen, and G. Fitzmaurice. "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*. MobileHCI '15. Copenhagen, Denmark: Association for Computing Machinery, 2015, pp. 144–152. ISBN: 9781450336529. DOI: 10.1145/2785830.2785867.

[62] D. L. Grover, M. T. King, and C. A. Kushler. "Reduced Keyboard Disambiguating Computer." US5818437A. Oct. 1998. URL: https://patents.google.com/patent/US5818437A/en.

[63] A.-M. Gueorguieva, G. Rakhmetulla, and A. S. Arif. *Enabling Input on Tiny/Headless Systems Using Morse Code*. Tech. rep. arXiv: 2012.06708. Dec. 2020. URL: http://arxiv.org/abs/2012.06708.

[64] J. Guerreiro, A. Rodrigues, K. Montague, T. Guerreiro, H. Nicolau, and D. Gonçalves. "TabLETS Get Physical: Non-Visual Text Entry on Tablet Devices." en. In: *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems - CHI '15*. Seoul, Republic of Korea: ACM Press, 2015, pp. 39–42. ISBN: 978-1-4503-3145-6. DOI: 10.1145/2702123.2702373.

[65] T. Guerreiro, P. Lagoá, H. Nicolau, D. Gonçalves, and J. A. Jorge. "From Tapping to Touching: Making Touch Screens Accessible to Blind Users." In: *IEEE MultiMedia* 15.4 (2008), pp. 48–50. DOI: 10.1109/MMUL.2008.88.

[66] T. Guerreiro, H. Nicolau, J. Jorge, and D. Gonçalves. "Towards Accessible Touch Interfaces." In: *Proceedings of the 12th International ACM SIGACCESS Conference on Computers and Accessibility*. ASSETS '10. Orlando, Florida, USA: Association for Computing Machinery, 2010, pp. 19–26. DOI: 10.1145/1878803.1878809.

[67] T. J. V. Guerreiro, H. Nicolau, J. Jorge, and D. Gonçalves. "Assessing Mobile Touch Interfaces for Tetraplegics." In: *Proceedings of the 12th International Conference on Human Computer Interaction with Mobile Devices and Services*. MobileHCI '10. Lisbon, Portugal: Association for Computing Machinery, 2010, pp. 31–34. DOI: 10.1145/1851600.1851608.

[68] A. Gupta and R. Balakrishnan. "DualKey: Miniature Screen Text Entry via Finger Identification." In: *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems*. CHI '16. San Jose, California, USA: Association for Computing Machinery, May 2016, pp. 59–70. ISBN: 978-1-4503-3362-7. DOI: 10.1145/2858036.2858052.

[69] L. Hakobyan, J. Lumsden, D. O'Sullivan, and H. Bartlett. "Mobile Assistive Technologies for the Visually Impaired." In: *Survey of ophthalmology* 58.6 (2013), pp. 513–528. DOI: 10.1016/j.survophthal.2012.10.004.

[70] K. Harbusch and M. Kühn. "Towards an Adaptive Communication Aid with Text Input from Ambiguous Keyboards." In: *Proceedings of the Tenth Conference on European Chapter of the Association for Computational Linguistics - Volume 2*. EACL '03. Budapest, Hungary: Association for Computational Linguistics, 2003, pp. 207–210. ISBN: 1111567890. DOI: 10.3115/1067737.1067786.

[71] S. G. Hart. "Nasa-Task Load Index (NASA-TLX); 20 Years Later." en. In: *Proceedings of the Human Factors and Ergonomics Society Annual Meeting* 50.9 (Oct. 2006). Publisher: SAGE Publications Inc, pp. 904–908. ISSN: 2169-5067. DOI: 10.1177/154193120605000909.

[72] S. G. Hart and L. E. Staveland. "Development of NASA-TLX (Task Load Index): Results of Empirical and Theoretical Research." en. In: *Advances in Psychology*. Vol. 52. Amsterdam, The Netherlands: Elsevier, 1988, pp. 139–183. ISBN: 978-0-444-70388-0. DOI: 10.1016/S0166-4115(08)62386-9.

[73] R. L. Hershman and W. A. Hillix. "Data Processing in Typing: Typing Rate as a Function of Kind of Material and Amount Exposed." en. In: *Human Factors* 7.5 (Oct. 1965), pp. 483–492. ISSN: 0018-7208. DOI: 10.1177/001872086500700507.

[74] J. Hong, S. Heo, P. Isokoski, and G. Lee. "SplitBoard: A Simple Split Soft Keyboard for Wristwatch-Sized Touch Screens." In: *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems*. CHI '15. New York, NY, USA: ACM, 2015, pp. 1233–1236. ISBN: 978-1-4503-3145-6. DOI: 10.1145/2702123.2702273.

[75] M.-C. Hsiu, D.-Y. Huang, C. A. Chen, Y.-C. Lin, Y.-p. Hung, D.-N. Yang, and M. Chen. "Forceboard: Using Force as Input Technique on Size-Limited Soft Keyboard." In: *Proceedings of the 18th International Conference on Human-Computer Interaction with Mobile Devices and Services Adjunct*. MobileHCI '16. Florence, Italy: Association for Computing Machinery, Sept. 2016, pp. 599–604. ISBN: 978-1-4503-4413-5. DOI: 10.1145/2957265.2961827.

[76] K. Huang, T. Starner, E. Do, G. Weinberg, D. Kohlsdorf, C. Ahlrichs, and R. Leibrandt. "Mobile Music Touch: Mobile Tactile Stimulation for Passive Learning." In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. CHI '10. Atlanta, Georgia, USA: Association for Computing Machinery, 2010, pp. 791–800. ISBN: 9781605589299. DOI: 10.1145/1753326.1753443.

[77] S. Hwang and G. Lee. "Qwerty-Like 3×4 Keypad Layouts for Mobile Phone." In: *CHI '05 Extended Abstracts on Human Factors in Computing Systems*. CHI EA '05. Portland, OR, USA: Association for Computing Machinery, Apr. 2005, pp. 1479–1482. ISBN: 978-1-59593-002-6. DOI: 10.1145/1056808.1056946.

[78] D. Iakovakis, R. E. Mastoras, S. Hadjidimitriou, V. Charisis, S. Bostanjopoulou, Z. Katsarou, L. Klingelhoefer, H. Reichmann, D. Trivedi, R. K. Chaudhuri, L. J. Hadjileontiadis, S. D, and J. D. "Smartwatch-based Activity Analysis During Sleep for Early Parkinson's Disease Detection." In: *2020 42nd Annual International Conference of the IEEE Engineering in Medicine Biology Society (EMBC)*. 2020, pp. 4326–4329. DOI: 10.1109/EMBC44109.2020.9176412.

[79] A. Inc. *Use Siri on your Apple Watch*. 2019. URL: https://support.apple.com/en-us/HT205184.

[80] *IRL, Interagency Language Roundtable Language Skill Level Descriptions*. 2020. URL: https://web.archive.org/web/20220621012650/https://www.govtilr.org/Skills/ILRscale5.htm.

[81] C. B. Irwin and M. E. Sesto. "Performance and Touch Characteristics of Disabled and Non-Disabled Participants During a Reciprocal Tapping Task Using Touch Screen Technology." In: *Applied Ergonomics* 43.6 (2012), pp. 1038–1043. ISSN: 0003-6870. DOI: https://doi.org/10.1016/j.apergo.2012.03.003.

[82] H. Jiang and D. Weng. "HiPad: Text entry for Head-Mounted Displays Using Circular Touchpad." en. In: *2020 IEEE Conference on Virtual Reality and 3D User Interfaces (VR)*. Atlanta, GA, USA: IEEE, Mar. 2020, pp. 692–703. ISBN: 978-1-72815-608-8. DOI: 10.1109/VR46266.2020.1581236395562.

[83] J. A. Johnson. "Designing Technology for an Aging Population." In: *Extended Abstracts of the 2018 CHI Conference on Human Factors in Computing Systems*. CHI EA '18. Montreal QC, Canada: Association for Computing Machinery, 2018, pp. 1–2. ISBN: 9781450356213. DOI: 10.1145/3170427.3170641.

[84] Y. Jung, S. Kim, and B. Choi. "Consumer Valuation of the Wearables: The Case of Smartwatches." In: *Computers in Human Behavior* 63 (2016), pp. 899–905. ISSN: 0747-5632. DOI: https://doi.org/10.1016/j.chb.2016.06.040.

[85] T. Kamba, S. A. Elson, T. Harpold, T. Stamper, and P. Sukaviriya. "Using Small Screen Space More Efficiently." en. In: *Proceedings of the SIGCHI conference on Human factors in computing systems common ground - CHI '96*. Vancouver, British Columbia, Canada: ACM Press, 1996, pp. 383–390. ISBN: 978-0-89791-777-3. DOI: 10.1145/238386.238582.

[86] S. K. Kane, C. Jayant, J. O. Wobbrock, and R. E. Ladner. "Freedom to Roam: A Study of Mobile Device Adoption and Accessibility for People with Visual and Motor Disabilities." In: *Proceedings of the 11th International ACM SIGACCESS Conference on Computers and Accessibility*. Assets '09. Pittsburgh, Pennsylvania, USA: Association for Computing Machinery, 2009, pp. 115–122. ISBN: 9781605585581. DOI: 10.1145/1639642.1639663.

[87] K. Katsuragawa, J. R. Wallace, and E. Lank. "Gestural Text Input Using a Smartwatch." In: *Proceedings of the International Working Conference on Advanced Visual Interfaces*. AVI '16. Bari, Italy: Association for Computing Machinery, June 2016, pp. 220–223. ISBN: 978-1-4503-4131-8. DOI: 10.1145/2909132.2909273.

[88] J. H. Kim, L. S. Aulck, O. Thamsuwan, M. C. Bartha, C. A. Harper, and P. W. Johnson. "The Effects of Touch Screen Virtual Keyboard Key Sizes on Typing Performance, Typing Biomechanics and Muscle Activity." en. In: *Digital Human Modeling and Applications in Health, Safety, Ergonomics, and Risk Management. Human Body Modeling and Ergonomics*. Ed. by V. G. Duffy. Lecture Notes in Computer Science. Berlin, Heidelberg: Springer, 2013, pp. 239–244. ISBN: 978-3-642-39182-8. DOI: 10.1007/978-3-642-39182-8_28.

[89] K. J. Kim. "Shape and Size Matter for Smartwatches: Effects of Screen Shape, Screen Size, and Presentation Mode in Wearable Communication." In: *Journal of Computer-Mediated Communication* 22.3 (2017), pp. 124–140. DOI: 10.1111/jcc4.12186.

[90] S. Kim, S. Ahn, and G. Lee. "DiaQwerty: QWERTY Variants to Better Utilize the Screen Area of a Round or Square Smartwatch." en. In: *Proceedings of the 2018 ACM International Conference on Interactive Surfaces and Spaces - ISS '18*. Tokyo, Japan: ACM Press, 2018, pp. 147–153. ISBN: 978-1-4503-5694-7. DOI: 10.1145/3279778.3279792.

[91] A. Komninos and M. Dunlop. "Text Input on a Smart Watch." In: *IEEE Pervasive Computing* 13.4 (Oct. 2014), pp. 50–58. ISSN: 1558-2590. DOI: 10.1109/MPRV.2014.77.

[92] P. O. Kristensson and K. Vertanen. "The Inviscid Text Entry Rate and Its Application as a Grand Goal for Mobile Text Entry." In: *Proceedings of the 16th international conference on Human-computer interaction with mobile devices & services*. MobileHCI '14. Toronto, ON, Canada: Association for Computing Machinery, Sept. 2014, pp. 335–338. ISBN: 978-1-4503-3004-6. DOI: 10.1145/2628363.2628405.

[93] S. Kwon, D. Lee, and M. K. Chung. "Effect of Key Size and Activation Area on the Performance of a Regional Error Correction Method in a Touch-Screen QWERTY Keyboard." en. In: *International Journal of Industrial Ergonomics* 39.5 (Sept. 2009), pp. 888–893. ISSN: 0169-8141. DOI: 10.1016/j.ergon.2009.02.013.

[94] J. Lai, D. Zhang, S. Wang, I. D. Y. Kilic, and L. Zhou. "ThumbStroke: A Virtual Keyboard in Support of Sight-Free and One-Handed Text Entry on Touchscreen Mobile Devices." In: *ACM Transactions on Management Information Systems (TMIS)* 10.3 (2019), p. 11. DOI: 10.1145/3343858.

[95] J. Lai, D. Zhang, S. Wang, I. Yakut Kilic, and L. Zhou. "A Thumb Stroke-Based Virtual Keyboard for Sight-Free Text Entry on Touch-Screen Mobile Phones." In: *Proceedings of the 51st Hawaii International Conference on System Sciences*. 2018.

[96] P. Lamkin. *Smart Wearables Market to Double by 2022: $27 Billion Industry Forecast*. Oct. 2018. URL: https://web.archive.org/web/20201202174401/https://www.forbes.com/sites/paullamkin/2018/10/23/smart-wearables-market-to-double-by-2022-27-billion-industry-forecast/ (visited on 12/17/2020).

[97] X. Lee. *Stenotype Machine*. 2019. URL: https://web.archive.org/web/http://xahlee.info/kbd/stenotype_machine.html.

[98] L. A. Leiva, A. Sahami, A. Catala, N. Henze, and A. Schmidt. "Text Entry on Tiny QWERTY Soft Keyboards." In: *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems*. CHI '15. Seoul, Republic of Korea: Association for Computing Machinery, 2015, pp. 669–678. ISBN: 9781450331456. DOI: 10.1145/2702123.2702388.

[99] J. S. Leon. *Frequency of Character Pairs in English Language Text, Codes and Cryptography*. 2008. URL: https://web.archive.org/web/20220113092124/http://homepages.math.uic.edu/~leon/mcs425-s08/handouts/char_freq2.pdf.

[100] S. Leuthold, J. A. Bargas-Avila, and K. Opwis. "Beyond Web Content Accessibility Guidelines: Design of Enhanced Text User Interfaces for Blind Internet Users." In: *International Journal of Human-Computer Studies* 66.4 (2008), pp. 257–270. DOI: 10.1016/j.ijhcs.2007.10.006.

[101] V. I. Levenshtein. "Binary Codes Capable of Correcting Deletions, Insertions, and Reversals." In: *Soviet physics doklady*. Vol. 10. 8. 1966, pp. 707–710.

[102] S. H. Levine and C. Goodenough-Trepagnier. "Customised Text Entry Devices for Motor-Impaired Users." In: *Applied Ergonomics* 21.1 (Mar. 1990), pp. 55–62. ISSN: 0003-6870. DOI: 10.1016/0003-6870(90)90074-8.

[103] F. C. Y. Li, L. Findlater, and K. N. Truong. "Effects of Hand Drift While Typing on Touchscreens." In: *Proceedings of Graphics Interface 2013*. GI '13. Regina, Sascatchewan, Canada: Canadian Information Processing Society, 2013, pp. 95–98. ISBN: 978-1-4822-1680-6. URL: http://dl.acm.org/citation.cfm?id=2532129.2532146.

[104] F. C. Y. Li, R. T. Guy, K. Yatani, and K. N. Truong. "The 1Line Keyboard: A Qwerty Layout In a Single Line." In: *Proceedings of the 24th Annual ACM Symposium on User Interface Software and Technology*. UIST '11. Santa Barbara, California, USA: ACM, 2011, pp. 461–470. ISBN: 978-1-4503-0716-1. DOI: 10.1145/2047196.2047257.

[105] S. J. Liebowitz and S. E. Margolis. "The Fable of the Keys." In: *The Journal of Law and Economics* 33.1 (1990), pp. 1–25. DOI: 10.1086/467198.

[106] Y.-L. Lin, M.-C. Chen, Y.-P. Wu, Y.-M. Yeh, and H.-P. Wang. "A Flexible On-Screen Keyboard: Dynamically Adapting for Individuals' Needs." In: *Universal Access in Human-Computer Interaction. Applications and Services*. Ed. by C. Stephanidis. Lecture Notes in Computer Science. Berlin, Heidelberg: Springer, 2007, pp. 371–379. ISBN: 978-3-540-73283-9. DOI: 10.1007/978-3-540-73283-9_42.

[107] Y.-L. Lin, T.-F. Wu, M.-C. Chen, Y.-M. Yeh, and H.-P. Wang. "Designing a Scanning On-Screen Keyboard for People with Severe Motor Disabilities." en. In: *Computers Helping People with Special Needs*. Ed. by K. Miesenberger, J. Klaus, W. Zagler, and A. Karshmer. Lecture Notes in Computer Science. Berlin, Heidelberg: Springer, 2008, pp. 1184–1187. ISBN: 978-3-540-70540-6. DOI: 10.1007/978-3-540-70540-6_178.

[108] R. López-Blanco, M. A. Velasco, A. Méndez-Guerrero, J. P. Romero, M. D. del Castillo, J. I. Serrano, E. Rocon, and J. Benito-León. "Smartwatch For the Analysis of Rest Tremor In Patients With Parkinson's disease." In: *Journal of the Neurological Sciences* 401 (2019), pp. 37–42. ISSN: 0022-510X. DOI: https://doi.org/10.1016/j.jns.2019.04.011.

[109] V. Luthra and S. Ghosh. "Understanding, Evaluating and Analyzing Touch Screen Gestures for Visually Impaired Users in Mobile Environment." In: *Universal Access in Human-Computer Interaction. Access to Interaction*. Ed. by M. Antona and C. Stephanidis. Cham: Springer International Publishing, 2015, pp. 25–36. ISBN: 978-3-319-20681-3. DOI: 10.1007/978-3-319-20681-3_3.

[110] S. Lydell. *English Bigram and Letter Pair Frequencies from the Google Corpus Data in JSON Format*. en. Aug. 2015. URL: https://web.archive.org/web/20211009124111/https://gist.github.com/lydell/c439049abac2c9226e53.

[111] K. Lyons, T. Starner, D. Plaisted, J. Fusia, A. Lyons, A. Drew, and E. W. Looney. "Twiddler Typing: One-Handed Chording Text Entry for Mobile Phones." In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. CHI '04. Vienna, Austria: Association for Computing Machinery, 2004, pp. 671–678. ISBN: 1581137028. DOI: 10.1145/985692.985777.

[112] I. S. MacKenzie. *Human-Computer Interaction: An Empirical Research Perspective*. First edition. Amsterdam: Morgan Kaufmann, 2013. ISBN: 978-0-12-405865-1. DOI: 10.1016/C2012-0-02819-0.

[113] I. S. MacKenzie. "KSPC (Keystrokes per Character) as a Characteristic of Text Entry Techniques." In: *Human Computer Interaction with Mobile Devices*. Ed. by F. Paternò. Berlin, Heidelberg: Springer Berlin Heidelberg, 2002, pp. 195–210. ISBN: 978-3-540-45756-5. DOI: 10.1007/3-540-45756-9_16.

[114] I. S. MacKenzie. "The One-Key Challenge: Searching for a Fast One-Key Text Entry Method." In: *Proceedings of the 11th International ACM SIGACCESS Conference on Computers and Accessibility*. Assets '09. Pittsburgh, Pennsylvania, USA: Association for Computing Machinery, 2009, pp. 91–98. ISBN: 9781605585581. DOI: 10.1145/1639642.1639660.

[115] I. S. Mackenzie and T. Felzer. "SAK: Scanning Ambiguous Keyboard for Efficient One-Key Text Entry." In: *ACM Trans. Comput.-Hum. Interact.* 17.3 (July 2010), 11:1–11:39. ISSN: 1073-0516. DOI: 10.1145/1806923.1806925.

[116] I. S. MacKenzie, H. Kober, D. Smith, T. Jones, and E. Skepner. "LetterWise: Prefix-Based Disambiguation for Mobile Text Input." In: *Proceedings of the 14th annual ACM symposium on User interface software and technology*. UIST '01. Orlando, Florida: Association for Computing Machinery, Nov. 2001, pp. 111–120. ISBN: 978-1-58113-438-4. DOI: 10.1145/502348.502365.

[117] I. S. MacKenzie and R. W. Soukoreff. "A Character-Level Error Analysis Technique for Evaluating Text Entry Methods." In: *Proceedings of the Second Nordic Conference on Human-Computer Interaction*. NordiCHI '02. Aarhus, Denmark: Association for Computing Machinery, 2002, pp. 243–246. ISBN: 1581136161. DOI: 10.1145/572020.572056.

[118] I. S. MacKenzie and R. W. Soukoreff. "Phrase Sets for Evaluating Text Entry Techniques." In: *CHI '03 Extended Abstracts on Human Factors in Computing Systems*. CHI EA '03. New York, NY, USA: ACM, 2003, pp. 754–755. ISBN: 978-1-58113-637-1. DOI: 10.1145/765891.765971.

[119] I. S. MacKenzie and K. Tanaka-Ishii. *Text Entry Systems: Mobility, Accessibility, Universality*. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 2007. ISBN: 0123735912, 9780080489797.

[120] I. S. MacKenzie and S. X. Zhang. "The Design and Evaluation of a High-Performance Soft Keyboard." In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. CHI '99. Pittsburgh, Pennsylvania, USA: ACM, 1999, pp. 25–31. ISBN: 0-201-48559-1. DOI: 10.1145/302979.302983.

[121] I. S. Mackenzie, S. X. Zhang, and R. W. Soukoreff. "Text Entry Using Soft Keyboards." In: *Behaviour & Information Technology* 18.4 (1999), pp. 235–244. DOI: 10.1080/014492999118995.

[122] M. Malu, P. Chundury, and L. Findlater. "Exploring Accessible Smartwatch Interactions for People with Upper Body Motor Impairments." In: *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*. CHI '18. Montreal QC, Canada: Association for Computing Machinery, 2018, pp. 1–12. ISBN: 9781450356206. DOI: 10.1145/3173574.3174062.

[123] M. Malu, P. Chundury, and L. Findlater. "Motor Accessibility of Smartwatch Touch and Bezel Input." In: *The 21st International ACM SIGACCESS Conference on Computers and Accessibility*. ASSETS '19. Pittsburgh, PA, USA: Association for Computing Machinery, 2019, pp. 563–565. ISBN: 9781450366762. DOI: 10.1145/3308561.3354638.

[124] J. Mankoff and G. D. Abowd. "Cirrin: A Word-Level Unistroke Keyboard for Pen Input." In: *Proceedings of the 11th annual ACM symposium on User interface software and technology — UIST '98*. San Francisco, California, United States: ACM Press, 1998, pp. 213–214. ISBN: 978-1-58113-034-8. DOI: 10.1145/288392.288611.

[125] S. Mascetti, C. Bernareggi, and M. Belotti. "TypeInBraille: A Braille-Based Typing Application for Touchscreen Devices." In: *The Proceedings of the 13th International ACM SIGACCESS Conference on Computers and Accessibility*. ASSETS '11. Dundee, Scotland, UK: Association for Computing Machinery, 2011, pp. 295–296. ISBN: 9781450309202. DOI: 10.1145/2049536.2049614.

[126] S. Mascetti, C. Bernareggi, and M. Belotti. "TypeInBraille: Quick Eyes-Free Typing on Smartphones." In: *International Conference on Computers for Handicapped Persons*. Springer. 2012, pp. 615–622. DOI: 10.1007/978-3-642-31534-3_90.

[127] E. Matias, I. S. MacKenzie, and W. Buxton. "Half-QWERTY: Typing with One Hand Using Your Two-Handed Skills." en. In: *Conference companion on Human factors in computing systems - CHI '94*. Boston, Massachusetts, United States: ACM Press, 1994, pp. 51–52. ISBN: 978-0-89791-651-6. DOI: 10.1145/259963.260024.

[128] D. McGookin, S. Brewster, and W. Jiang. "Investigating Touchscreen Accessibility for People with Visual Impairments." In: *Proceedings of the 5th Nordic Conference on Human-Computer Interaction: Building Bridges*. NordiCHI '08. Lund, Sweden: Association for Computing Machinery, 2008, pp. 298–307. ISBN: 9781595937049. DOI: 10.1145/1463160.1463193.

[129] K. Montague, H. Nicolau, and V. L. Hanson. "Motor-Impaired Touchscreen Interactions in the Wild." In: *Proceedings of the 16th International ACM SIGACCESS Conference on Computers; Accessibility*. ASSETS '14. Rochester, New York, USA: Association for Computing Machinery, 2014, pp. 123–130. ISBN: 9781450327206. DOI: 10.1145/2661334.2661362.

[130] Motorola Mobility LLC. *Voice commands — Moto 360 2nd Generation*. 2020. URL: https://web.archive.org/web/20220812081355/https://motorola-global-portal.custhelp.com/app/answers/detail/a_id/106996/p/2690.

[131] M. E. Mott, R.-D. Vatavu, S. K. Kane, and J. O. Wobbrock. "Smart Touch: Improving Touch Accuracy for People with Motor Impairments with Template Matching." In: *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems*. New York, NY, USA: Association for Computing Machinery, 2016, pp. 1934–1946. ISBN: 9781450333627. DOI: 10.1145/2858036.2858390.

[132] M. E. Mott and J. O. Wobbrock. "Cluster Touch: Improving Touch Accuracy on Smartphones for People with Motor and Situational Impairments." In: *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*. New York, NY, USA: Association for Computing Machinery, 2019, pp. 1–14. ISBN: 9781450359702. DOI: 10.1145/3290605.3300257.

[133] M. Naftali and L. Findlater. "Accessibility in Context: Understanding the Truly Mobile Experience of Smartphone Users with Motor Impairments." In: *Proceedings of the 16th International ACM SIGACCESS Conference on Computers and Accessibility*. ASSETS '14. Rochester, New York, USA: Association for Computing Machinery, 2014, pp. 209–216. ISBN: 9781450327206. DOI: 10.1145/2661334.2661372.

[134] P. Norvig. *English Letter Frequency Counts: Mayzner Revisited or ETAOIN SRHLDCU*. 2012. URL: https://web.archive.org/web/20210817234739/http://norvig.com/mayzner.html.

[135] J. Oliveira, T. Guerreiro, H. Nicolau, J. Jorge, and D. Gonçalves. "Blind People and Mobile Touch-Based Text-Entry: Acknowledging the Need for Different Flavors." In: *The Proceedings of the 13th International ACM SIGACCESS Conference on Computers and Accessibility*. ASSETS '11. Dundee, Scotland, UK: ACM, 2011, pp. 179–186. ISBN: 978-1-4503-0920-2. DOI: 10.1145/2049536.2049569.

[136] J. Oliveira, T. Guerreiro, H. Nicolau, J. Jorge, and D. Gonçalves. "BrailleType: Unleashing Braille over Touch Screen Mobile Phones." In: *Human-Computer Interaction — INTERACT 2011*. Ed. by P. Campos, N. Graham, J. Jorge, N. Nunes, P. Palanque, and M. Winckler. Berlin, Heidelberg: Springer Berlin Heidelberg, 2011, pp. 100–107. ISBN: 978-3-642-23774-4. DOI: 10.1007/978-3-642-23774-4_10.

[137] S. Oney, C. Harrison, A. Ogan, and J. Wiese. "ZoomBoard: A Diminutive Qwerty Soft Keyboard Using Iterative Zooming for Ultra-Small Devices." In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. CHI '13. New York, NY, USA: ACM, 2013, pp. 2799–2802. ISBN: 978-1-4503-1899-0. DOI: 10.1145/2470654.2481387.

[138] A. Oulasvirta, A. Reichel, W. Li, Y. Zhang, M. Bachynskyi, K. Vertanen, and P. O. Kristensson. "Improving Two-Thumb Text Entry on Touchscreen Devices." In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. CHI '13. Paris, France: Association for Computing Machinery, 2013, pp. 2765–2774. ISBN: 9781450318990. DOI: 10.1145/2470654.2481383.

[139] P. Parhi, A. K. Karlson, and B. B. Bederson. "Target Size Study for One-Handed Thumb Use on Small Touchscreen Devices." In: *Proceedings of the 8th Conference on Human-Computer Interaction with Mobile Devices and Services*. MobileHCI '06. Helsinki, Finland: Association for Computing Machinery, 2006, pp. 203–210. ISBN: 1595933905. DOI: 10.1145/1152215.1152260.

[140] K. Park, T. Goh, and H.-J. So. "Toward Accessible Mobile Application Design: Developing Mobile Application Accessibility Guidelines for People with Visual Impairment." In: *Proceedings of HCI Korea*. HCIK '15. Seoul, Republic of Korea: Hanbit Media, Inc., 2014, pp. 31–38. ISBN: 978-89-6848-752-1. URL: https://dl.acm.org/doi/10.5555/2729485.2729491.

[141] Y. S. Park, S. H. Han, J. Park, and Y. Cho. "Touch Key Design for Target Selection on a Mobile Phone." In: *Proceedings of the 10th international conference on Human computer interaction with mobile devices and services*. MobileHCI '08. Amsterdam, The Netherlands: Association for Computing Machinery, Sept. 2008, pp. 423–426. ISBN: 978-1-59593-952-4. DOI: 10.1145/1409240.1409304.

[142] K. B. Perry and J. P. Hourcade. "Evaluating One Handed Thumb Tapping on Mobile Touchscreen Devices." In: *Proceedings of Graphics Interface 2008*. GI '08. Windsor, Ontario, Canada: Canadian Information Processing Society, 2008, pp. 57–64. ISBN: 978-1-56881-423-0. URL: https://dl.acm.org/doi/10.5555/1375714.1375725.

[143] G. Poirier, A. Ohayon, A. Juranville, F. Mourey, and J. Gaveau. "Deterioration, Compensation and Motor Control Processes in Healthy Aging, Mild Cognitive Impairment and Alzheimer's Disease." In: *Geriatrics* 6.1 (2021). ISSN: 2308-3417. DOI: 10.3390/geriatrics6010033.

[144] O. Polacek, A. J. Sporka, and P. Slavik. "Text Input for Motor-Impaired People." In: *Univers. Access Inf. Soc.* 16.1 (Mar. 2017), pp. 51–72. ISSN: 1615-5289. DOI: 10.1007/s10209-015-0433-0.

[145] R. Powers, M. Etezadi-Amoli, E. M. Arnold, S. Kianian, I. Mance, M. Gibiansky, D. Trietsch, A. S. Alvarado, J. D. Kretlow, T. M. Herrington, et al. "Smartwatch Inertial Sensors Continuously Monitor Real-World Motor Fluctuations in Parkinson's Disease." In: *Science Translational Medicine* 13.579 (2021). DOI: 10.1126/scitranslmed.abd7865.

[146] V. Prabhu and G. Prasad. "Designing a Virtual Keyboard with Multi-Modal Access for People with Disabilities." In: *2011 World Congress on Information and Communication Technologies*. 2011, pp. 1133–1138. DOI: 10.1109/WICT.2011.6141407.

[147] M. Proschowsky, N. Schultz, and N. E. Jacobsen. "An Intuitive Text Input Method for Touch Wheels." In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. CHI '06. Montréal, Québec, Canada: Association for Computing Machinery, 2006, pp. 467–470. ISBN: 1595933727. DOI: 10.1145/1124772.1124842.

[148] R. Qin, S. Zhu, Y.-H. Lin, Y.-J. Ko, and X. Bi. "Optimal-T9: An Optimized T9-like Keyboard for Small Touchscreen Devices." In: *Proceedings of the 2018 ACM International Conference on Interactive Surfaces and Spaces*. ISS '18. Tokyo, Japan: Association for Computing Machinery, Nov. 2018, pp. 137–146. ISBN: 978-1-4503-5694-7. DOI: 10.1145/3279778.3279786.

[149] RESNA. *RESNA*. https://resna.stanford.edu. 2020. URL: https://resna.stanford.edu.

[150] M. Romano, L. Paolino, G. Tortora, and G. Vitiello. "The Tap and Slide Keyboard: A New Interaction Method for Mobile Device Text Entry." In: *International Journal of Human-Computer Interaction* 30.12 (2014), pp. 935–945. DOI: 10.1080/10447318.2014.924349.

[151] R. Rosenberg and M. Slater. "The Chording Glove: A Glove-Based Text Input Device." In: *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)* 29.2 (May 1999), pp. 186–191. ISSN: 1558-2442. DOI: 10.1109/5326.760563.

[152] T. Salthouse. "Anticipatory Processing in Transcription Typing." In: *Journal of Applied Psychology* 70.2 (1985), pp. 264–271. ISSN: 1939-1854(Electronic),0021-9010(Print). DOI: 10.1037/0021-9010.70.2.264.

[153] T. Salthouse and J. Saults. "Multiple Spans in Transcription Typing." In: *Journal of Applied Psychology* 72.2 (May 1987), pp. 187–196. ISSN: 0021-9010.

[154] T. A. Salthouse. "Effects of Age and Skill in Typing." In: *Journal of Experimental Psychology: General* 113.3 (1984), pp. 345–371. ISSN: 1939-2222(Electronic),0096-3445(Print). DOI: 10.1037/0096-3445.113.3.345.

[155] S. Sarcar, A. S. Arif, and A. Mazalek. "Metrics for Bengali Text Entry Research." In: *arXiv: 1706.08205* (2017).

[156] A. Savidis and C. Stephanidis. "Unified User Interface Development: The Software Engineering of Universally Accessible Interactions." In: *Univers. Access Inf. Soc.* 3.3–4 (Oct. 2004), pp. 165–193. ISSN: 1615-5289. DOI: 10.1007/s10209-004-0096-8.

[157] R. D. Seidler, J. A. Bernard, T. B. Burutolu, B. W. Fling, M. T. Gordon, J. T. Gwin, Y. Kwak, and D. B. Lipps. "Motor Control and Aging: Links to Age-Related Brain Structural, Functional, and Biochemical Effects." In: *Neuroscience & Biobehavioral Reviews* 34.5 (2010). Special Section: Dopaminergic Modulation of Lifespan Cognition, pp. 721–733. ISSN: 0149-7634. DOI: 10.1016/j.neubiorev.2009.10.005.

[158] C. Seim, T. Estes, and T. Starner. "Towards Passive Haptic Learning of Piano Songs." In: *2015 IEEE World Haptics Conference (WHC)*. June 2015, pp. 445–450. DOI: 10.1109/WHC.2015.7177752.

[159] M. Serrano, A. Roudaut, and P. Irani. "Visual Composition of Graphical Elements on Non-Rectangular Displays." In: *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems*. CHI '17. New York, NY, USA: ACM, 2017, pp. 4405–4416. ISBN: 978-1-4503-4655-9. DOI: 10.1145/3025453.3025677.

[160] L. H. Shaffer. "Latency Mechanisms in Transcription." In: *Attention and Performance* IV (1973), pp. 435–446.

[161] L. H. Shaffer and A. French. "Coding Factors in Transcription." In: *Quarterly Journal of Experimental Psychology* 23.3 (Aug. 1971), pp. 268–274. ISSN: 0033-555X. DOI: 10.1080/14640746908401821.

[162] L. H. Shaffer and J. Hardwick. "The Basis of Transcription Skill." In: *Journal of Experimental Psychology* 84.3 (1970), pp. 424–440. ISSN: 0022-1015(Print). DOI: 10.1037/h0029287.

[163] C. E. Shannon. "A Mathematical Theory of Communication." In: *The Bell System Technical Journal* 27.3 (July 1948), pp. 379–423. ISSN: 0005-8580. DOI: 10.1002/j.1538-7305.1948.tb01338.x.

[164] M. A. Sharif, G. Rakhmetulla, and A. S. Arif. "TapStr: A Tap and Stroke Reduced-Qwerty for Smartphones." In: *Companion Proceedings of the 2020 Conference on Interactive Surfaces and Spaces*. New York, NY, USA: Association for Computing Machinery, 2020, pp. 47–50. ISBN: 9781450375269. DOI: 10.1145/3380867.3426208.

[165] T. Shibata, D. Afergan, D. Kong, B. F. Yuksel, I. S. MacKenzie, and R. J. Jacob. "DriftBoard: A Panning-Based Text Entry Technique for Ultra-Small Touchscreens." In: *Proceedings of the 29th Annual Symposium on User Interface Software and Technology*. UIST '16. New York, NY, USA: ACM, 2016, pp. 575–582. ISBN: 978-1-4503-4189-9. DOI: 10.1145/2984511.2984591.

[166] K. Shinohara and J. O. Wobbrock. "In the Shadow of Misperception: Assistive Technology Use and Social Interactions." In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. CHI '11. Vancouver, BC, Canada: Association for Computing Machinery, 2011, pp. 705–714. ISBN: 9781450302289. DOI: 10.1145/1978942.1979044.

[167] G. S. Snoddy. "Learning and Stability: A Psychophysiological Analysis of a Case of Motor Learning with Clinical Applications." In: *Journal of Applied Psychology* 10.1 (1926). Place: US Publisher: American Psychological Association, pp. 1–36. ISSN: 1939-1854(Electronic),0021-9010(Print). DOI: 10.1037/h0075814.

[168] R. T. Snodgrass and V. F. Camp. *Radio Receiving for Beginners*. New York: MacMillan, 1922.

[169] M. Soegaard. *Accessibility: Usability for all. Interaction Design Foundation*. 2019. URL: https://web.archive.org/web/20220118210410/https://www.interaction-design.org/literature/article/accessibility-usability-for-all.

[170] R. W. Soukoreff and I. S. MacKenzie. "An Informatic Rationale for the Speed-Accuracy Trade-Off." In: *2009 IEEE International Conference on Systems, Man and Cybernetics*. Oct. 2009, pp. 2890–2896. DOI: 10.1109/ICSMC.2009.5346580.

[171] R. W. Soukoreff and I. S. MacKenzie. "Metrics for Text Entry Research: An Evaluation of MSD and KSPC, and a New Unified Error Metric." In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. CHI '03. New York, NY, USA: ACM, 2003, pp. 113–120. ISBN: 978-1-58113-630-2. DOI: 10.1145/642611.642632.

[172] C. Southern, J. Clawson, B. Frey, G. Abowd, and M. Romero. "An Evaluation of BrailleTouch: Mobile Touchscreen Text Entry for the Visually Impaired." In: *Proceedings of the 14th International Conference on Human-Computer Interaction with Mobile Devices and Services*. MobileHCI '12. San Francisco, California, USA: Association for Computing Machinery, 2012, pp. 317–326. ISBN: 9781450311052. DOI: 10.1145/2371574.2371623.

[173] K. Tanaka-Ishii, Y. Inutsuka, and M. Takeichi. "Entering Text with a Four-Button Device." In: *Proceedings of the 19th International Conference on Computational Linguistics — Volume 1*. COLING '02. Taipei, Taiwan: Association for Computational Linguistics, 2002, pp. 1–7. DOI: 10.3115/1072228.1072377.

[174] T. Tojo, T. Kato, and S. Yamamoto. "BubbleFlick: Investigating Effective Interface for Japanese Text Entry on Smartwatches." In: *Proceedings of the 20th International Conference on Human-Computer Interaction with Mobile Devices and Services*. MobileHCI '18. Barcelona, Spain: Association for Computing Machinery, Sept. 2018, pp. 1–12. ISBN: 978-1-4503-5898-9. DOI: 10.1145/3229434.3229455.

[175] TouchOne. *The First Dedicated Smartwatch Keyboard*. May 2016. URL: https://web.archive.org/web/20220527030402/https://www.kickstarter.com/projects/790443497/touchone-keyboard-the-first-dedicated-smartwatch-k/.

[176] S. Trewin, C. Swart, and D. Pettick. "Physical Accessibility of Touchscreen Smartphones." In: *Proceedings of the 15th International ACM SIGACCESS Conference on Computers and Accessibility*. 2013, pp. 1–8. DOI: 10.1145/2513383.2513446.

[177] T. S. Tullis and J. N. Stetson. "A Comparison of Questionnaires for Assessing Website Wsability." In: *Usability professional association conference*. Vol. 1. Minneapolis, USA. 2004. DOI: 10.1.1.396.3677.

[178] K. Vertanen, D. Gaines, C. Fletcher, A. M. Stanage, R. Watling, and P. O. Kristensson. "VelociWatch: Designing and Evaluating a Virtual Keyboard for the Input of Challenging Text." In: *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*. CHI '19. ACM, May 2019, pp. 1–14. ISBN: 978-1-4503-5970-2. DOI: 10.1145/3290605.3300821.

[179] K. Vertanen, H. Memmi, J. Emge, S. Reyal, and P. O. Kristensson. "VelociTap: Investigating Fast Mobile Text Entry Using Sentence-Based Decoding of Touchscreen Keyboard Input." In: *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems*. CHI '15. Seoul, Republic of Korea: ACM, 2015, pp. 659–668. ISBN: 978-1-4503-3145-6. DOI: 10.1145/2702123.2702135.

[180] D. Vogel and P. Baudisch. "Shift: A Technique for Operating Pen-Based Interfaces Using Touch." In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. CHI '07. New York, NY, USA: ACM, 2007, pp. 657–666. ISBN: 978-1-59593-593-9. DOI: 10.1145/1240624.1240727.

[181] F. Wang, X. Cao, X. Ren, and P. Irani. "Detecting and Leveraging Finger Orientation for Interaction with Direct-Touch Surfaces." In: *Proceedings of the 22nd annual ACM symposium on User interface software and technology*. UIST '09. Victoria, BC, Canada: Association for Computing Machinery, Oct. 2009, pp. 23–32. ISBN: 978-1-60558-745-5. DOI: 10.1145/1622176.1622182.

[182] C. Welch. *Google's Gboard Keyboard Now Lets You Communicate through Morse Code on both Android and iOS*. 2018. URL: https://web.archive.org/web/20200523040248/https://www.theverge.com/2018/7/11/17561958/google-gboard-morse-code-communication-feature.

[183] D. Wigdor and R. Balakrishnan. "A Comparison of Consecutive and Concurrent Input Text Entry Techniques for Mobile Phones." In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. CHI '04. Vienna, Austria: Association for Computing Machinery, 2004, pp. 81–88. ISBN: 1581137028. DOI: 10.1145/985692.985703.

[184] Wiktionary. *Frequency lists/PG/2006/04/1-10000 - Wiktionary*. Nov. 2019. URL: https://en.wiktionary.org/wiki/Wiktionary:Frequency_lists/PG/2006/04/1-10000.

[185] S. P. Witte and R. D. Cherry. "Writing Processes and Written Products in Composition Research." In: *Studying Writing: Linguistic Approaches*. 1st ed. Vol. 1. Beverly Hills, CA, USA: Sage Publications, 1986, pp. 112–153. ISBN: 0-8039-2372-4.

[186] J. O. Wobbrock. "Measures of Text Entry Performance." In: *Text entry systems: Mobility, accessibility, universality* (2007), pp. 47–74.

[187] J. O. Wobbrock. "Situationally Aware Mobile Devices for Overcoming Situational Impairments." In: *Proceedings of the ACM SIGCHI Symposium on Engineering Interactive Computing Systems*. EICS '19. Valencia, Spain: Association for Computing Machinery, 2019. ISBN: 9781450367455. DOI: 10.1145/3319499.3330292.

[188] J. O. Wobbrock, B. A. Myers, and J. A. Kembel. "EdgeWrite: A Stylus-Based Text Entry Method Designed for High Accuracy and Stability of Motion." In: *Proceedings of the 16th Annual ACM Symposium on User Interface Software and Technology*. UIST '03. New York, NY, USA: ACM, 2003, pp. 61–70. ISBN: 978-1-58113-636-4. DOI: 10.1145/964696.964703.

[189] M. Wolfersberger. "L1 to L2 Writing Process and Strategy Transfer: A Look at Lower Proficiency Writers." en. In: *TESL-EJ* 7.2 (2003). ISSN: 1072-4303. URL: https://www.tesl-ej.org/issues/volume7/ej26/ej26a6/.

[190] P. C. Wong, K. Zhu, X.-D. Yang, and H. Fu. "Exploring Eyes-Free Bezel-Initiated Swipe on Round Smartwatches." In: *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems*. CHI '20. Honolulu, HI, USA: Association for Computing Machinery, 2020, pp. 1–11. ISBN: 9781450367080. DOI: 10.1145/3313831.3376393.

[191] World Health Organization. *Blindness and vision impairment*. URL: https://web.archive.org/web/20180430040736/https://www.who.int/news-room/fact-sheets/detail/blindness-and-visual-impairment.

[192] World Health Organization. *Change the Definition of Blindness*. 2018. URL: https://web.archive.org/web/20210321121325/https://www.who.int/blindness/Change%20the%20Definition%20of%20Blindness.pdf?ua=1.

[193] K. Yatani and K. N. Truong. "An Evaluation of Stylus-Based Text entry Methods on Handheld Devices Studied in Different User Mobility States." In: *Pervasive and Mobile Computing* 5.5 (2009), pp. 496–508. ISSN: 1574-1192. DOI: 10.1016/j.pmcj.2009.04.002.

[194] X. Yi, C. Yu, W. Shi, X. Bi, and Y. Shi. "Word Clarity as a Metric in Sampling Keyboard Test Sets." In: *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems*. CHI '17. Denver, Colorado, USA: Association for Computing Machinery, 2017, pp. 4216–4228. ISBN: 9781450346559. DOI: 10.1145/3025453.3025701.

[195] X. Yi, C. Yu, W. Shi, and Y. Shi. "Is It Too Small?: Investigating the Performances and Preferences of Users When Typing on Tiny Qwerty Keyboards." en. In: *International Journal of Human-Computer Studies* 106 (Oct. 2017), pp. 44–62. ISSN: 1071-5819. DOI: 10.1016/j.ijhcs.2017.05.001.

[196] X. Yi, C. Yu, W. Xu, X. Bi, and Y. Shi. "COMPASS: Rotational Keyboard on Non-Touch Smartwatches." In: *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems*. CHI '17. Denver, Colorado, USA: Association for Computing Machinery, May 2017, pp. 705–715. ISBN: 978-1-4503-4655-9. DOI: 10.1145/3025453.3025454.

[197] M. R. Zhang, S. Zhai, and J. O. Wobbrock. "Text Entry Throughput: Towards Unifying Speed and Accuracy in a Single Performance Metric." In: *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*. CHI '19. Glasgow, Scotland UK: Association for Computing Machinery, 2019, pp. 1–13. ISBN: 9781450359702. DOI: 10.1145/3290605.3300866.

[198] S. Zhu, T. Luo, X. Bi, and S. Zhai. "Typing on an Invisible Keyboard." In: *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*. CHI '18. Montreal QC, Canada: ACM, 2018, 439:1–439:13. ISBN: 978-1-4503-5620-6. DOI: 10.1145/3173574.3174013.