COMPARATIVE STUDIES OF GESTURE-BASED AND SENSOR-BASED INPUT METHODS FOR MOBILE USER INTERFACES

SAURABH GARG

A THESIS SUBMITTED TO THE FACULTY OF GRADUATE STUDIES IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF MASTER OF SCIENCE

GRADUATE PROGRAM IN COMPUTER SCIENCE
YORK UNIVERSITY
TORONTO, ONTARIO

MAY 2021

Abstract

Three user studies were conducted to compare gesture-based and sensor-based interaction methods. The first study compared the efficiency and speed of three scroll navigation methods for touch-screen mobile devices: *Tap Scroll* (touch-based), *Kinetic Scroll* (gesture-based), and *Fingerprint Scroll* (our newly introduced sensor-based method). The second study compared the accuracy and speed of three zoom methods. One method was *GyroZoom* which uses the mobile phone's gyroscope sensor. The second one is *Pinch-to-Zoom* (Gesture-based) method. *VolumeZoom*, the third method, uses volume buttons that were reprogrammed to perform zoom operations. The third study on text entry compared a QWERTY-based onscreen keyboard with a novel 3D gesture-based *Write-in-Air* method. This method utilizes webcam sensors. Our key findings from the three experiments are that sensor-based interaction methods are intuitive and provide a better user experience than gesture-based interaction methods. The sensor-based methods were on par with the speed and accuracy of the gesture-based methods.

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 Motivation

The concept of human-computer interaction (HCI) is not at this point bound to gesture-based interactions. While activities such as pressing a button, touching a screen, or entering text using a soft keyboard are common, performing such tasks without physical touch is an interesting thought. This is fascinating both for non-disabled users and also for physically challenged users. This opens countless possibilities of interactions without physically pressing a button or touching a screen. Gregor et al. [10] discussed the digital separation that exists towards physically challenged people when it comes to modern technology. They note that developers often exclude disabled people from their target audience and argue that this digital divide need not exist.

Our work aims to diminish this gap as well as explore novel interaction methods for general use. Any user input that does not need direct actual touchscreen input is conceivably valuable for physically challenged users: The goal is sensor-based computing. We focus on minimizing touchscreen mobile interactions. We partition our methodology into three parts: scroll interaction, zoom interaction, and text entry. These three parts address essential user interaction in daily needs. Our goal is to find feasible sensor-based interaction alternatives for these tasks and to compare them with traditional gesture-based interaction and understand the performance gaps.

1.2 Input Styles for Mobile Devices

A significant challenge with mobile devices is providing appropriate input that is easy to use and can accurately work with common tasks. Numerous tasks such as target selection, text entry, or navigating user interfaces are challenging. Broadly speaking, input styles can be classified into three categories: software, hardware, and hybrid. Hardware-based input methods utilize physical properties on the device which are external to the actual application and might be add-ons to the gadgets themselves. Software-based input methods are built inside real applications such as touch-based software buttons or scrollbars. Hybrid input methods combine the approaches from software and hardware input techniques.

1.2.1 Hardware-based Input

Most mobile phones have hardware components and accessories to provide different kinds of inputs. For example, physical buttons can be programmed as a “home” button or a “mode” button suitable for different purposes depending on the current screen. Hence, the button is not fixed absolutely but changes based on the current perspective of the screen. Another example is a scroll wheel for navigating a document vertically. In 1996, Rekimoto introduced the idea of navigating using the device itself and added a sensor to detect device tilt and rotation when the user moved the device [24]. The utilization of tilt for navigation and selection tasks [1, 6, 7, 11, 13] and text input on mobile devices [21, 30] has been explored extensively. Researchers have combined tilt with other external inputs like buttons [21, 29], gestures, and other sensors [15]. A key advantage of hardware-based input interactions is that they can provide single-handed interaction. NaviPoint [15] and ScrollPad [8] are other examples of hardware-based inputs.

1.2.2 Software-based Input

Software-based input ordinarily utilizes a stylus or finger on a touch-sensitive display. For general navigation in smartphones, the most common strategy is scrollbars that are software-programmed. Even though this is familiar to most users, it still has limitations on the desktop [34]. Using the same technique on a mobile device is even more challenging. A study by Smith and schraefel [26] identified three user interaction methods with scrollbars and potential challenges with each. Users can drag the handle of the scrollbar by maintaining constant pressure on the mouse button. This might result in skipping important information if the user unintentionally let go of the thumb and skipped desired parts of the document. Due to smaller screen sizes, this problem is even more critical on mobile devices. Secondly, users can click on the arrow buttons at each end of the scrollbar to move the document. But, this is slow and tiring. Users can also click on the positions on the scrollbar to randomly jump to a particular segment of the document. But, this can be disorienting [6, 34] and the problem compounds on smaller screen sizes. Scrollbars require the user to draw their attention from the document to a software-designed scrollbar which may require additional cognitive effort and motor resources. Also, scrollbars restrict movement to a single direction (i.e., horizontal or vertical). Another software-based method for navigation on a mobile device is tap-and-drag. This technique requires the user to tap anywhere on the screen with a stylus and drag in the display area in any direction. This is a familiar interaction for desktop users. Johnson [14] compared different drag techniques with edge navigation on a touch screen and found that users were faster and had a preference for drag to navigate. One major disadvantage of using dragging is that it is not suitable for documents containing many selection targets. Also, dragging is limited by the size of the screen as users can only drag in increments of the screen size [15].

Research has also been conducted using touch gestures on mobile devices. For example, Harrison et al. [11] used finger gestures for flipping pages in a digital document. The advantage of software-based input is that it is built into the application itself and can be designed to use paradigms similar to a desktop application. The main drawback is that these interactions often require the use of both hands. Sometimes users prefer to hold a device with one hand while the other hand interacts with the device.

1.2.3 Hybrid-based Input

This combines the use of external hardware-based input with software-based input. Usually, one hand holds the mobile device to navigate the space (e.g., tilt) and the freehand makes selections. Peephole displays [33] utilize a spatially aware device that is moved to reveal different parts of an information space while making selections with a stylus. Eslambolchilar and Murray-Smith [6] and Eslambolchilar et al. [7] coupled an SDAZ (speed-dependent automatic zooming) system with a tilt to navigate and scroll through information while using a stylus for selection.

Interaction techniques for mobile devices all face the common challenge of needing to be used while the user is "on the move" (i.e., mobile). In general, it is hard for users to navigate and select items in mobile settings, regardless of the input method they are using. Research has shown that tactile [23] or audio feedback [22, 7] can improve users' accuracy and efficiency when performing navigation and selection tasks.

1.3 Our Approach

We conducted three experiments that involve custom-designed methods to perform sensor-based interactions for common tasks. The three experiments are summarized in Table 1.1. All the participants (six male and six female) were the same and consistent throughout all three experiments.

Table 1.1: Summary of experiments.

Exp	**Title**	**Focus**	**Hardware**	**Software**
1	Comparing Gesture-based and Sensor-based Scroll Methods	Scroll Methods	Google Pixel 3a	*ScrollExperiment*, *GoStats*.
2	Comparing Gesture-based and Sensor-based Zoom Methods	Zoom Methods	Google Pixel 3a	*ZoomExperiment*, *GoStats*.
3	Comparing Gesture-based and Sensor-based Text-Entry Methods	Text-Entry	Desktop/Laptop with webcam	*Write-in-Air* Experiment, *GoStats*.

The motivation for the first experiment comes from the fact that single-handed use of smartphones for scrolling is inefficient and does not deliver a good user experience. A typical scroll gesture occludes or interrupts the user's line of sight and affects the experience of using smartphones to read books or browse the web. We introduced a new scroll method that avoids occlusion, is simple to use with a single hand, and focuses on delivering a good user experience.

The second experiment derives similar motivation from the first experiment to tackle issues regarding single-hand use for zooming on a smartphone. Pinch-to-Zoom is a touch-based gestural method that is easy to use with both hands, but for single-hand use, it results in an awkward hand posture due to holding the phone and using the gesture through the same hand. We introduce two new methods that make it easier to use operations like zoom using a single hand.

The third experiment targets the potential applications of text entry without the dependency on a physical keyboard. We introduced a new method that enables text entry through air gestures. Some of the potential applications are writing in a car as a passenger, quick doodling while doing a presentation on stage or in a conference room, and assisting physically disabled people to enter text.

1.4 Research Contribution

1.4.1 Comparing Gesture-based and Sensor-based Scroll Navigation Methods

The first experiment examined scroll navigation. The performance of three methods of scrolling were compared in a user study. The three methods were *Kinetic Scroll* (a swipe gesture-based method), *Tap Scroll* (a touch-screen tap method), and *Fingerprint Scrolling* (a newly introduced scroll method). *Fingerprint Scroll* is a hybrid sensor-based method that utilizes a mobile phone hardware fingerprint sensor, normally used to authenticate users while unlocking the phone. This sensor is generally placed either on the front or back of the phone. We conducted a user study to compare the methods and analyze user performance and accuracy and derived meaningful insights.

1.4.2 Comparing Gesture-based and Sensor-based Zoom Interaction Methods

The second experiment examined zoom methods. The performance of three methods of zooming was compared in a user study. The three methods were *GyroZoom* (a newly introduced sensor-based zoom method), *VolumeZoom* (a newly introduced button-based method), and *Pinch-to-Zoom* (the traditional gesture-based method). VolumeZoom is a novel and alternative zoom method that uses volume keys in a mobile device to perform zoom interaction. The volume-up key is used to zoom in and the volume-down key is used for zooming out. This is potentially an effective way to utilize the volume keys for zooming as it is easily operated using a single hand. We conducted a user study to compare the methods and analyze the performance and accuracy and derive meaningful insights.

1.4.3 Comparing traditional and Sensor-based Text-Entry Methods

The third and final user study compared two text-entry methods. The first method is a traditional QWERTY method that is widely used and accepted in touch-based interactions. The second method is a novel *Write-in-Air* method. It is a custom-designed software method written in Python. *Write-in-Air* method recognizes real-time gestures in a 3D space through a webcam attached to a desktop or a laptop. We conducted a user study to compare both the methods and analyze user performance to derive meaningful insights.

1.5 Outline of Thesis

We describe our software systems, and implementations in Chapter 2 which is followed by a literature review in Chapter 3. After that, in Chapters 4, 5, and 6, we elaborate on the methodology and results of each experiment. Finally, we conclude and offer future opportunities for research in Chapter 7.

Chapter 2

Software Systems, Implementations and Theories

In this section, we examine the software implementations that were utilized for the three user studies. Some these frameworks previously existed, others were custom-developed for this research.

2.1 Scroll Experiment - Software

The experiment was conducted on Google Pixel 3a running the Android (11.0) operating system. The device has a 5.6-inch OLED display with a resolution of 2220 $\times$ 1080 pixels and a density of 441 ppi. The weight of the device is 147 grams. The software was developed in Java using the Android SDK in Android Studio. The experimental application was developed specifically for this research. Three scroll methods were implemented. It is a requirement for a device to have fingerprint sensor hardware behind the device for this software to function properly. The ap-

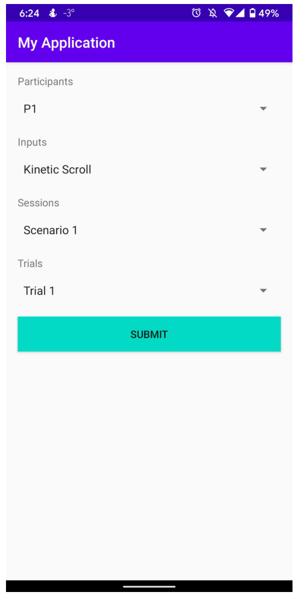


Figure 2.1: Scroll Experiment Configuration Activity.

plication begins with a configuration activity that prompts the user to select the participant code and other experimental parameters, such as input method, session code, and group code (for counterbalancing). See Figure 2.1.

Once configured, the user presses the SUBMIT button to initiate the testing process. The main activity contains a scrolling user interface where the user does the scroll test as written on the screen. After the scroll test is done, the user is expected to press the DONE TESTING button to indicate the end of the particular trial and advance to the next trial. Metrics like completion time and error percentage are calculated thereafter and the values are stored in a remote Google *firebase* database.

2.2 Zoom Experiment - Software

There are multiple ways to perform pinch interaction for zooming. In Figure 2.2a, the user holds the device in one hand and performs the gesture using the other hand. In Figure 2.2b, the user holds the device in both hands and uses both thumbs to perform gestures. In Figure 2.2c, the user single-handedly performs a pinch-to-zoom gesture and holds the device. However, this last procedure causes an abnormal and uncomfortable hand posture. Users are frequently compelled to play out the pinch-to-zoom motion using one hand. Most cell phones additionally utilize a double-tap gesture which takes into consideration zooming in or out by double-tapping a zone of interest. While this strategy functions admirably with one hand, it is limited to zooming by a discrete amount instead of offering a flexible interactive zoom.

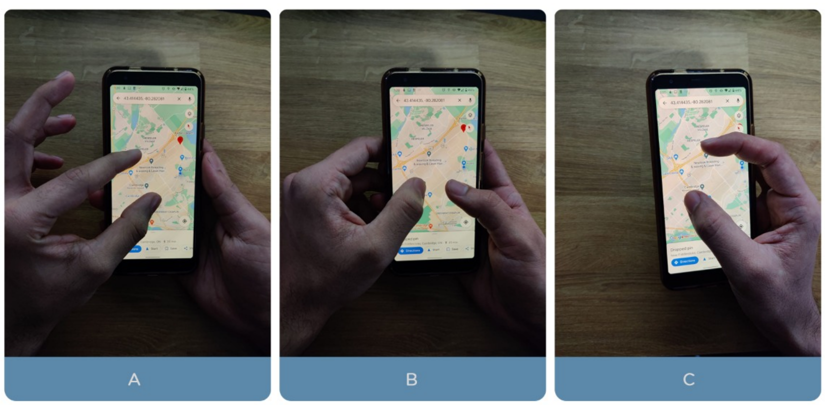


Figure 2.2: Three ways to perform pinch-to-zoom gestures.

In this thesis, we introduce *VolumeZoom* as an alternative zoom method. *VolumeZoom* uses the volume keys in a mobile device to perform zoom interaction. In Figure 2.3, the volume-up key is used to zoom in and the volume-down key is used to zoom out. This is potentially an effective way to utilize the volume keys for zooming, as the interaction is easily performed using a single hand. We introduce another new method called *GyroZoom* which does not involve any button press or touch gesture on the screen. *GyroZoom* utilizes the gyroscope sensor of the mobile device to calculate the rotation of the device in real-time. See Figure 2.3. Depending on the angle and direction of rotation, it zooms in or out by a corresponding amount. Clockwise rotation results in zooming in and the angle of rotation controls the amount of zoom. Similarly, anti-clockwise rotation results in zooming out and the angle of rotation controls the amount of zoom.

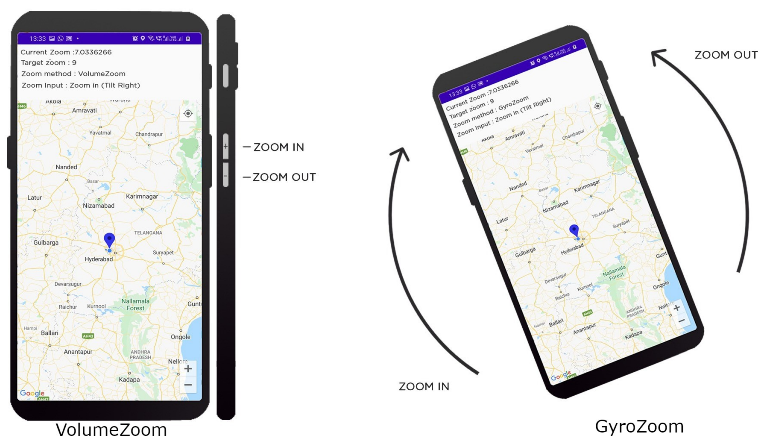


Figure 2.3: Demonstration of *VolumeZoom* and *GyroZoom*.

The experiment was conducted on Google Pixel 3a similar to the device used in Scroll Experiment. The software was developed in Java using the Android SDK in the software Android Studio. This experimental application was developed specifically for this research. Three zooming methods were implemented in this application. It is a requirement for a device to have a gyroscope sensor for this software to function properly. The application begins with a configuration activity that prompts the user to select the participant code and other experimental parameters, like the zoom method and group code (for counterbalancing). Once configured, the user presses the START button to initiate the testing process. The main activity contains a map zooming user interface.

The participant follows the zoom direction and target zoom mentioned on-screen stating exactly the direction of zooming and the amount of zoom required to complete the trial. The user interface also displays in real-time the current zoom level and whether the user has achieved the goal. When the current zoom matches the target zoom, the user presses the FINISH button to end the trial and advance to the next trial.

2.3 Text Entry - Software

Finger recognition is a significant element of numerous computer vision applications. In this application, a histogram-based methodology is utilized to isolate the hand from the background. Thresholding and filtering strategies are utilized for background cancellation to obtain optimal outcomes. One challenge in recognizing fingers was separating a hand from the background and distinguishing the tip of a finger. In an application that needs to track a user's hand movement, a skin shade histogram will be exceptionally helpful. This histogram is then used to subtract the background from a picture, just leaving portions of the picture that contain skin tone. An easier technique to recognize skin is to discover pixels that are in a specific RGB or HSV range. The challenge with this approach is that changing light conditions and skin tones can obscure the skin location. However, a histogram method will in general be more precise and can reasonably accommodate the current light conditions. The software was made using Python with a few dependencies, such as

Tensorflow, Keras, and NumPy.

Some constraints to this software, in its currenty form, are noted below.

- It only supports the text-entry of capital letters.
- It does not support special characters input.
- It takes input discretely, character by character. It cannot process multiple character gestures at once.
- It does not support punctuation marks.
- It supports *space* and *enter* key. The gestures for *space* and *enter* key were custom-designed. The gesture for *space* was a horizontal line from left to right, and for *enter* key was a horizontal line from right to left.

In operation, the hand is placed inside a blue square to start recording gestures, as shown in Figure 6.1. Further details are provided in Section 6.1.2. The user presses a key ('w') to bracket the recording of an input gesture. With this, the prediction of the gesture automatically follows the recording of the gesture. A user "types" (i.e., gestures) in a sentence, and the algorithm automatically predicts the gestures in real-time. The software also calculates various metrics like text-entry speed and accuracy which are exported in a CSV file for follow-up analyses.

Chapter 3

Literature Review

We now focus on previous research work on the three topics noted above, with interest in the potential for sensor-based and gesture-based interactions. The three topics are scroll navigation, zoom methods, and text entry. The literature review is organized into these topic areas.

3.1 Research on Scroll Navigation Methods

Table 3.1 summarizes results from four papers where user studies were performed utilizing scroll navigation methods.

Smith and schraefel [26] assessed a scroll navigation method called *Radial Scroll* which was found effective for variable speed document scrolling with touch screen devices. They also proposed improving radial scrolling by giving the user the ability to change the context of the scrolling in real-time, such as switching from per-line to per-page scrolling.

MacKay et al. [17] compared software-based navigation techniques with the new

Table 3.1: Summary of user studies on Scroll methods.

1st Author	**Method**	**N**[1]	**Notes**
Smith [26]	Touch	8	*Radial Scroll* performs better than traditional methods on short scroll target acquisition.
MacKay [17]	Touch	18	The techniques of *touch-n-go* and *tap-and-drag* outperformed the traditional scrollbar technique for the simple navigation tasks.
Moscovich [19]	Touch	10	The *Virtual Scroll Ring* is a tenable scrolling alternative. This is esp. true when most scrolling actions are expected to be longer than half a page.
Oakley [20]	Touchless	12	User performance using the position-based input was good and provided promise, and allowed us to select optimum parameters for position-based list navigation.

touch-n-go approach on a mobile device. In terms of preference, users found *touch-n-go* easier than the *scrollbar* and *tap-and-drag* method during multiple levels of mobility. Participants achieved better performance while sitting but were considerably slower while walking for all of the navigation techniques.

Moscovich and Hughes [19] assessed a *Virtual Scroll Ring* and found it a tenable scrolling alternative. This was especially true when scrolling is expected to be longer than half a page. In the case of smaller than half-page, extra care was required to ensure that enough data were collected for a robust estimate of the circle. Since the virtual scroll ring scrolls the view smoothly, in increments as small as one pixel, it allows users to read the text while they scroll.

[1] Number of Participants.

Oakley and O'Modhrain [20] offered that a position-based input mapping proposed in his paper had considerable potential. One credible reason for this is that users can learn to reach specific list items. This may lead to an open-loop interaction where an item can be point selected with certain confidence without requiring explicit feedback from the system. It is interesting to look into the implications of using this input technique with nested menu systems, where multiple selections are made to reach a single goal. In conclusion, the author believed that interfaces based around motion input and vibrotactile output may lead to a brighter future in mobile interaction.

3.2 Research on Zoom Methods

Table 3.2: Summary of user studies on Zoom methods.

1st Author	**Method**	**N**[1]	**Notes**
Ti [28]	Touchless	15	Flip Gesture performed significantly better than touch-based controls (1.83 s faster). Tilt and Hold was slower than touch-based conntrols.
Lai [16]	Touch	23	*ContextZoom* outperformed two-finger tap in both task completion time and number of discrete actions.
Farhad [9]	Touch	12	*Tap-and-drag* performed 17.9% better in terms of speed and 47.2% better in terms of efficiency as compared to *Pinch-to-Zoom*).
Boring [2]	Touch	24	*Fat Thumb* is fast, non-fatiguing, and the preferred technique, all while maintaining the offset rates of other techniques.
Harrison [12]	Touchless	10	Results from a study indicate that users believe *Lean and Zoom* interaction technique is intuitive, increases comfort, and improves performance.

Table 3.2 summarizes the results from five publications where user studies were performed utilizing zoom methods to interact with the device.

Farhad and MacKenzie [9] compared the performance of two zoom methods when performed with a single hand. The results from *Tap-and-drag* were generally good for single-handed usage. Also, tap-and-drag performed slightly worse in accuracy metric

[1]Number of Participants.

but over time, as participants become more familiar with tap-and-drag, performance improved.

Ti and Tjondronegoro [28] assessed a collection of tilt-based input methods for single-handed zooming. They analyzed the outcomes against traditional touch-based zooming and found that tilt-based methods are better than touch-based methods when performed with one hand. All participants found the traditional touch-based methods inferior due to the uncomfortable hand posture when performed with one hand. Nonetheless, they favored using pinch-to-zoom when two hands were accessible.

Lai et al. [16] assessed a single-handed partial zooming procedure. *ContextZoom* permits users to point to any spot on a display by long-pressing the location as the zooming focus (i.e., focal point). When the area is set, the user moves their thumb on the display to zoom in or out. Panning is disabled while zooming. The results were acceptable, with the completion time and the number of discrete activities, by and large, low. Participants likewise announced more significant levels of apparent adequacy and overall satisfaction.

Boring et al. [2] proposed *Fat Thumb*, a single-handed method that utilizes the thumb's contact size as a type of recreated pressure. The contact size takes into consideration toggling between panning and zooming relying upon the contact area.

Harrison et al. [12] presented a technique called *Lean and Zoom* that detects a user's proximity to the display using a camera and magnifies the on-screen content proportionally. Results from the user study indicate that users believe this interaction technique is intuitive, increases comfort, and improves performance.

3.3 Research on Text-Entry Methods

Table 3.3 summarizes the results obtained from five papers where user studies were performed utilizing some sensor-based or gesture-based text-entry methods.

Table 3.3: Summary of user studies on sensor-based Text-entry methods.

1st Author	**Method**	**N**[1]	**Notes**
Xu [31]	*RingText*	10	*RingText* compared with four other text entry mechanisms, is the most efficient technique with significantly higher text entry rate in the area of VR.
Darbar[4]	Hall Effect	5	This technique demands little cognitive load. It can well balance between typing speed and error rate.
Dobosz [5]	Eye Blink and EEG	12	Proposed touchless keyboard uses eye double blinks. In the area of such kind of text entry systems, this result of text-entry is good.
Rustagi [25]	Head Movement	25	The proposed interface uses a single camera and a QWERTY keypad displayed on a screen. which achieved an accuracy of 96.78%.
Darbar[3]	*MagiText*	3	*MagiText* is based on drawing character gesture in the space around the device using a magnet. This approach can be particularly suitable for very small mobile and tangible devices for taking short messages or notes quickly.

Xu et al. [31] introduced a technique called *RingText* that permits users to enter text by making small motions with their head and choose letters from a circular keyboard layout with two concentric circles: the outer circle contains letters housed

[1]Number of Participants

in distinct regions, while the inner one serves to reset selection and allows users to go looking for the following letter. They determined the acceptable size of the circle, the number of letters per region (LPR) within the areas of the outer circle, and also the alphabet starting position. The results show that a bigger center area can potentially decrease error rates, and users preferred the alphabet to start from the outer circle. A comparative study of hands-free text entry techniques in VR was conducted by comparing *RingText* with four other text entry mechanisms. Results showed that *RingText* is the best technique; it led users to realize a significantly higher text entry rate combined with a significantly lower total error rate.

Darbar et al. [4] presented hall effect sensors-based text entry mechanism that effectively uses the 3D space around the smartwatch for entering alphanumeric characters. This system doesn't consume any screen space and it doesn't need any visual search to seek out a personality. Also, it eliminates the fat finger problem. On average, they achieved and entry speed of 3.9 wpm ($SD = 0.36$) using the proposed text input method as compared to 5.78 wpm ($SD = 0.45$) using the typewriter keyboard. A t-test shows that the two techniques had a major effect on the entry speed ($p = 0.02$). The error rate using the QWERTY keyboard was 22.1% ($SD = 3.43$) and 6.4% ($SD = 2.62$) for our proposed technique.

Dobosz et al. [5] proposed a touchless keyboard that uses eye double blinks. Text entry with the blinks used EEG signals. The results of experiments show that the most effective mode used a meditation threshold of 80. It achieves an efficiency of 1.27 wpm, but we must note that the word prediction was used. When using meditation as a parameter for the mode of predicted words, the number of errors was lower compared to an attention-based method.

Rustagi et al. [25] presented a touchless typing interface that uses head movement based-gestures. The proposed interface used a webcam and an on-screen QWERTY keypad. Gestures captured by the camera were mapped to a sequence of clusters employing a GRU-based deep learning model trained on a rich set of head pose features. The presented interface achieved an accuracy of 96.8% and 86.8% under the intra-user and inter-user scenarios, respectively.

Darbar et al. [3] performed two separate experiments, one for Graffiti and another for *EdgeWrite*. It was observed that the *MagiText* system supporting *EdgeWrite* yielded 89.4% accuracy, whereas Graffiti can distinguish characters with an 81.2% recognition rate. As a result, *EdgeWrite* gesture input was less ambiguous than Graffiti.

Chapter 4

First Experiment: Comparison of Gesture-based and Sensor-based Scroll Navigation Methods

the first user study compared the efficiency and speed of three scroll navigation methods for touch-screen mobile devices: *Tap Scroll* (the traditional touch-based method), *Kinetic Scroll* (a touch-based gestural method), and *Fingerprint Scroll* (our newly introduced hybrid method). The study involved 12 participants and employed a Google *Pixel* device. We present the methodology, analysis of results, and discussions of this experiment below.

4.1 Methodology

A usability study was undertaken to evaluate the performance of the three scroll methods in terms of quantitative measures, user preference, and ease of use. We fully expected the *Kinetic Scroll* method to be better because of the user's familiarity but we wanted to relatively compare the performance and user experience of the newly introduced scroll methods with *Kinetic Scroll* method.

4.1.1 Participants

Twelve participants were recruited remotely from different universities across Canada. Six were male, six were female. Ages ranged from 22 to 26 years. All participants were comfortable with using smartphones. All participants were right-handed and were sitting down while performing the experiment. Participants were constrained to use only a single hand while testing. They had prior experience of using *Kinetic Scroll* method but no prior experience in using *Tap Scroll* or *Fingerprint Scroll* method. Participants were compensated $20 for their assistance.

4.1.2 Apparatus

The experiment was conducted on Google *Pixel 3a* running the Android 11.0 operating system. The specification of the device is mentioned in Section 2.1. The software was developed in Java using the Android SDK in the Android Studio. This experimental application was developed specifically for this research. Three scroll methods were implemented. The application begins with a configuration activity that prompts the user to select the participant code and other experimental parameters, like input method, session code, group code (for counterbalancing). Once configured, the user presses the SUBMIT button to initiate the testing process. The main activity contains a scrolling user interface.See Figure 4.1. On the main activity, the screen shows what exactly a participant needs to do. For example, a participant can be asked to go to page 10 and come back to page 1. Two new scroll methods

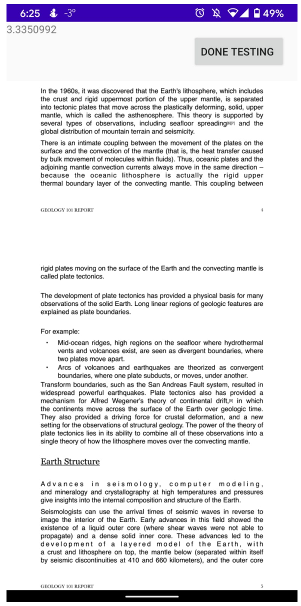


Figure 4.1: *MainActivity* user interface for the Scroll Experiment software.

were implemented. *Tap Scroll* requires discrete tap gestures on the arrow buttons shown on user interface in *MainActivity* to scroll up/down. *Fingerprint Scroll* requires fingerprint sensor hardware placed on the back of the phone which can be used to swipe up/down to scroll up/down respectively.

4.1.3 Procedure

Participants were informed and explained the purpose of the user study. They were requested to keep the Internet switched on during the entire experiment. Participants watched a video explaining the interaction methods and did some practice trials. Participants were asked to use just one hand for all interactions. Arm support was not allowed. With this brief introduction, testing began. Participants completed five trials per block and five blocks for each method. A trial is one instance of a participant scrolling from point A to point B. A participant took about 15-20 minutes to complete the experiment. The data were stored in a remote database in Google *Firebase*. The data were analyzed later for results and meaningful insights. After the testing was complete, the user was prompted with a questionnaire to gather feedback regarding their preference for scroll methods.

4.1.4 Design

The experiment used a 3×5 within-subjects design with the following independent variables and levels:

- Scroll Method (*Kinetic Scroll, Tap Scroll, Fingerprint Scroll*)
- Block Code (1, 2, 3, 4, 5)

Each participant completed five blocks of five trials for each scroll method. As such, the total number of trials for the experiment was 12 Participants $\times$ 3 Scroll Methods $\times$ 5 Blocks $\times$ 5 Trials = 900.

Participants were counterbalanced using a Latin square method. The application organized the participants into three groups with a group code assigned to every participant. Each group tested in a specific sequence of scroll methods. The dependent variables were completion time (s) and error rate (%). The duration of the experiment was roughly one hour per participant.

4.2 Results and Discussions

All trials were completed successfully. The data were later imported into a spreadsheet tool where summaries of various measures were calculated and charts were created. The analysis of variance test was performed using the GoStats[1] application.

4.2.1 Completion Time

Completion time is calculated when a user goes from point A to point B as asked in the test case. The software calculates the completion time on its own. The effect of group on completion time was not statistically significant ($F_{2,9} = 0.494$, ns), thus indicating that counterbalancing had the desired effect of offsetting order effects. The effect of scrolling method on completion time was statistically significant ($F_{2,18} = 1099.3, p < .0001$). The scrolling method × group interaction effect was not statistically significant ($F_{4,18} = 0.357$, ns), however. The effect of block on completion time was statistically significant ($F_{4,36} = 4243.7, p < .0001$). The block × group interaction effect was not statistically significant ($F_{8,36} = 0.241$, ns).The scrolling method × block interaction effect was statistically significant ($F_{8,72} = 375.2, p < .0001$).

The grand mean for completion time was 13.62 s. From Figure 4.2, we observe that *Tap Scroll* took 4.71% more time to complete a similar trial than *Kinetic Scroll*. The completion time of *Fingerprint Scroll* was 2.43 times longer than that of *Kinetic Scroll* and 2.32 times longer than *Tap Scroll*.

4.2.2 Error Percentage

Errors were logged when a user fails to do a task correctly. For example, the task can be "go to page 6 and come back to page 1". If the user goes to a different page than instructed, it counts as an error. The effect of group on error percentage was not statistically significant ($F_{2,9} = 0.150$, ns). However, the effect of scrolling

[1]http://www.yorku.ca/mack/GoStats/

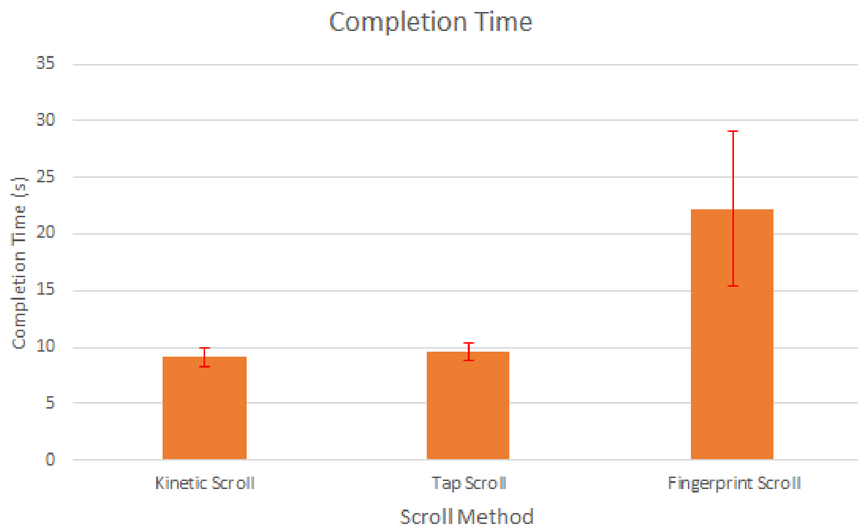


Figure 4.2: Completion time (s) by scroll method.

method on error percentage was statistically significant ($F_{2,18} = 73.500, p < .0001$). The scrolling method × group interaction effect was also not statistically significant ($F_{4,18} = 0.288$, ns).

The error percentage differences between different methods can also be viewed in terms of accuracy. More error percentage leads to lower accuracy.

As seen in Figure 4.3, the accuracy for *Fingerprint Scroll* was higher, with an average accuracy of 97.6% compared to 96.3% for *Tap Scroll* and 88.7% for *Kinetic Scroll*.

4.2.3 Participant Feedback

In the post-experiment questionnaire, participants were asked their preference of the scroll navigation method on a scale of 1 to 5. Their answers were recorded and analyzed using a Friedman non-parametric test. The *Fingerprint Scroll* method was

Figure 4.3: Error percentage (%) by scroll method.

preferred ($\chi^2 = 15.54$, $p = .0004$). Using a post hoc test, it was observed that all three pairwise comparisons were statistically significant.

Most participants stated that they found *Kinetic Scroll* frustrating to use while reading ebooks since it blocks the view and delivers a bad user experience. Most preferred using Fingerprint Scroll as it does not block the vision and they do not consider the speed that important for a majority of tasks like reading ebooks or web browsing. One participant noted the following:

Fingerprint Scroll is great as I read a lot of books on my phone during commuting. I don't want to see the finger coming in my way of vision all the time.

Overall, a majority of participants praised *Fingerprint scroll* and *Kinetic Scroll* various use scenarios. The average ratings on a 1-to-5 scale for *Fingerprint Scroll*, *Tap Scroll*, and *Kinetic Scroll* were 4.3, 3, and 4.1, respectively.

Chapter 5

Second Experiment: Comparison of Gesture-based and Sensor-based Zoom Control Methods

We conducted a user study comparing the accuracy and speed of three zoom methods for touch-screen devices. The comparison was between gesture-based and sensor-based zoom methods. One method is *GyroZoom* which uses the mobile phone's rotation to zoom in and zoom out. It is our newly introduced sensor-based approach. We constrained our research to single-handed interaction. The *Pinch-to-Zoom* method, also tested, is the standard touch-screen gesture method. *VolumeZoom* was the third method and does not require the user to touch the screen. Instead, the user presses the physical buttons that are normally used to increase or decrease device volume: they were reprogrammed to perform zoom operations. The user study engaged 12 participants and employed a Google *Pixel 3a* smartphone.

5.1 Methodology

A user study was conducted to compare the three user-interactive zooming methods for one-handed interaction. The goal was to compare the three zooming methods in terms of quantitative measures, user preference, and ease of use.

5.1.1 Participants

Twelve participants were recruited remotely from different universities across Canada. Six were male, six were female. Ages ranged from 22 to 26 years. All participants were comfortable using smartphones. All were right-handed and sat during testing. Participants were familiar with using *Pinch-to-Zoom* method. They weren't familiar with the other two zoom methods, however. Participants were compensated $20 for their assistance.

5.1.2 Apparatus

The experiment was conducted on Google Pixel 3a similar to the device used in the Scroll Experiment. The specifications of the device are noted in Section 2.1. The software was developed in Java using the Android SDK in Android Studio. The experimental application was developed specifically for this research. Three zooming methods were implemented in the application. The application begins with a configuration activity that prompts the user to select the participant code and other experimental parameters, like the zoom method and a group code (for counterbalancing).

Once configured, the user presses a START button to initiate the testing process. The main activity as shown in Figure 5.1 contains a map zooming user interface.

The software presented the participant with the desired zoom direction and target zoom and stated the real-time direction and amount of zoom required for completing the trial successfully. The user interface also displayed the real-time current zoom for the user to match with the target zoom.

After the current zoom matches the target zoom, the user pressed the FINISH button to end the trial and advance to the next trial. The FINISH button appears on UI only when the user matches the current zoom with target zoom. For the *VolumeZoom* method, the user pressed the volume up/down keys to zoom in/out respectively. For the *GyroZoom* method, the user just rotates the device clockwise/anticlockwise to zoom in/out respectively. A graphical depiction of these interactions was shown earlier in Figure 2.3. One trial refers to one instance of the user matching the current zoom displayed on the screen by target zoom.

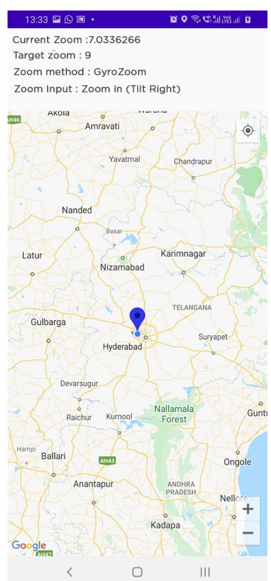


Figure 5.1: *MainActivity* user interface for Zoom Experiment software.

5.1.3 Procedure

Participants were informed and explained the purpose of the user study. They were requested to keep the Internet switched on during the entire experiment. They were shown videos explaining each zoom interaction method. They were also given a few practice trials. For testing, participants were seated and instructed to do the trials single-handedly. They were also asked to use whichever hand they feel comfortable with within trials. Arm support or any kind of armrest was not allowed. With this brief introduction, testing began. Participants completed five trials per zoom method. Each trial consisted of a randomized sequence of combinations of zoom direction (in or out) and zoom level (low, medium, high). Each participant took about 15-20 min to complete the experiment. The data were stored in a remote database in Google Firebase over the Internet. The data were analyzed later for meaningful insights. After the testing was complete, the user was prompted with a questionnaire to gather feedback on their preferences of the zoom methods.

5.1.4 Design

The user study employed a $3 \times 3 \times 2$ within-subjects design. The independent variables and levels were as follows:

- Zoom method (*GyroZoom, VolumeZoom, Pinch-to-Zoom*)
- Zoom Level (*low, medium, high*)
- Zoom Direction (*In, Out*)

Each participant completed 5 trials for each zoom method. The total number of trials was 12 participants × 3 Zoom Methods × 3 Zoom Levels × 2 Zoom Direction

$\times$ 5 Trials = 1080. To offset learning effects, participants were divided into three groups to counterbalance the order of testing the zoom methods. The zoom level and zoom direction conditions were chosen at random. The dependent variables were completion time per trial and efficiency. A detailed explanation of the calculation of efficiency is given in the results section.

5.2 Results and Discussion

All trials were completed successfully. The data were later imported into a spreadsheet tool where summaries of various measures were calculated and charts were created. The analysis of the variance test was performed using the GoStats[1] application.

5.2.1 Completion Time

The grand mean for completion time was 7.45 s. By zoom method, the means were 6.58 s for *GyroZoom*, 8.26s for *VolumeZoom*, and 7.51 s for *Pinch-to-Zoom*. As seen in Figure 5.2, trials using *GyroZoom* were 18.1% faster than the traditional *Pinch-to-Zoom* method. *VolumeZoom* was 18.9% slower than *Pinch-to-Zoom*. There was a clear pattern in the data showing that the zoom-out operation in *VolumeZoom* was always faster than zoom-in. This could be due to the ease of accessibility and reach to the volume-down button compared to the volume-up button. An analysis of variance was done to examine the mean completion times of the 12 participants while doing the zoom trials. The effect of zoom method on completion time was statistically significant ($F_{2,22} = 110.6, p < .0001$). The effect of zoom level on completion time

[1] http://www.yorku.ca/mack/GoStats/

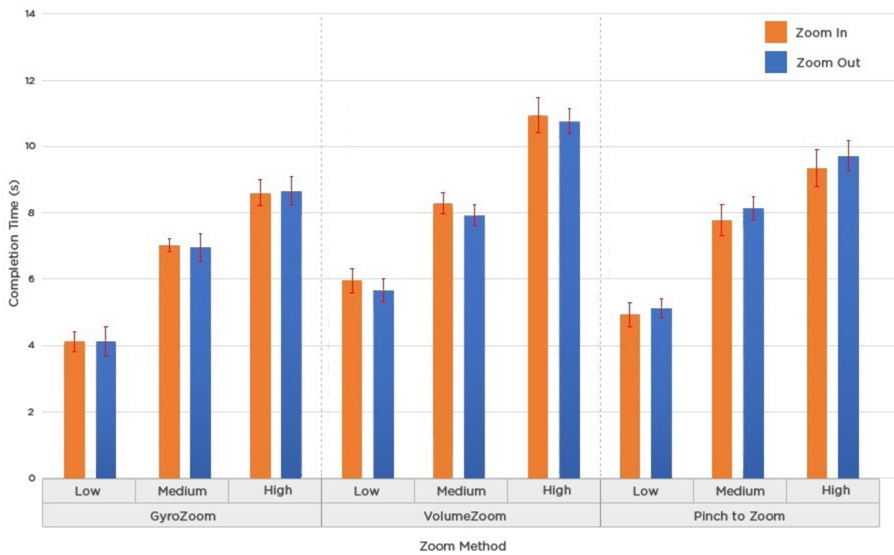


Figure 5.2: Completion time (s) by zoom method, with error bars shown in red.

was also statistically significant ($F_{2,22} = 888.0, p < .0001$). However, the effect of zoom direction on completion time was not statistically significant ($F_{1,11} = 1.27, p > .05$). The zoom method × zoom level interaction effect was statistically significant ($F_{4,44} = 9.46, p < .0001$), as was the zoom method × zoom direction interaction effect ($F_{2,22} = 5.71, p < .05$). The zoom level × zoom direction interaction effect was not statistically significant, however ($F_{2,22} = 1.002, p > .05$). Group effects were not statistically significant.

5.2.2 Efficiency

Efficiency is a measure of how usable or effective a zoom method is. It is defined as the number of zoom switches performed per trial. For example, if a user is required to zoom-in until the current zoom equals the target zoom, but, by mistake, the user zooms in more than required, then performs a zoom-out operation to finish the trial successfully, the extra zoom transition is counted as 1. Lower scores are better. In Figure 5.3, it is seen that *VolumeZoom* has higher efficiency than the *GyroZoom* and *Pinch-to-Zoom*. *Pinch-to-Zoom* is the least efficient method according to the user data. From the analysis of the variance, we found that the effect of the zoom method on efficiency was statistically significant ($F_{2,22} = 162.2, p < .0001$). The remaining effects were not statistically significant.

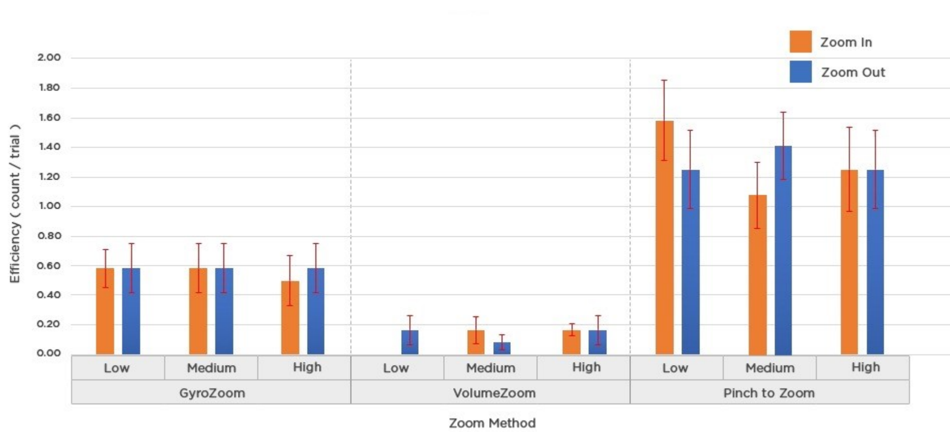


Figure 5.3: Efficiency by zoom method, with error bars shown in red. Lower scores are better.

5.2.3 Participant Feedback

In the post-experiment questionnaire, participants were asked their preference of the zoom method on a scale of 1 to 5. A Friedman test was performed using the GoStats [8] application. The differences by zoom method were significant ($\chi^2 = 15.395, p = .0005$). Using a post hoc test, it was observed that the pairwise comparison of *GyroZoom* with *Pinch-to-Zoom* and *VolumeZoom* with *Pinch-to-Zoom* were statistically significant. Most participants stated that they found single-handed *Pinch-to-Zoom* extremely frustrating to use as it results in an awkward hand posture. They found it easier using both hands. Some expressed concerns about dropping the phone while performing *Pinch-to-Zoom* by a single hand.

One participant noted, *GyroZoom is an innovative method while I am walking, and I need to single-handed perform zoom operations. I also find VolumeZoom to be easy to use.*

Overall, participants praised both *VolumeZoom* and *GyroZoom* in subjective feedback and indicated a preference for these over the traditional *Pinch-to-Zoom* method. The average ratings on a 1-5 scale were 4.3 and 4.2 for *GyroZoom* and *VolumeZoom*, respectively.

Chapter 6

Third Experiment: Comparison of Traditional and Sensor-based Text Entry Methods

We conducted a user study comparing two text-entry methods on a desktop or laptop computer.[1] The comparison was between a traditional on-screen QWERTY keyboard and our newly introduced and designed *Write-in-Air* method. The *Write-in-Air* method works by analyzing 3D gestures created in real-time in front of a webcam. The method converts the gesture to a character on the keyboard. Gesture recognition has endless possibilities by which we can avoid the effort of physical typing. The user study was done on a desktop or laptop computer with either Windows OS or macOS.

[1]For practical reasons, the apparatus for this user study was the participant's desktop or laptop computer with a built-in or add-on camera. The results, however, extend to mobile computing with the potential use of a mobile device's front-facing camera.

6.1 Methodology

6.1.1 Participants

Twelve participants were recruited remotely from different universities across Canada. Six were male, six were female. Ages ranged from 22 to 26 years. All participants were comfortable using desktop or laptop computers and have considerable experience using a QWERTY keyboard for text entry. All participants performed the experiment while sitting on a chair in front of their desktop or laptop computer. They maintained a distance of about 3 ft between the camera and themselves. Participants were compensated $20 for their assistance.

6.1.2 Apparatus

For hardware, we used a laptop or a desktop computer with a built-in webcam or an external webcam. Two experimental software applications were used to present trials and statistically capture and analyze the user data. The first app was *SoftKeyboardExperiment*[2]. This software presented trials and also captured the data for input using the on-screen QWERTY keyboard. The data captured were for typing speed, completion time, and accuracy. See Figure 6.1.

The second app, *WriteInAirExperiment*, was built in a similar design style to SoftKeyboardExperiment. The main constraint of this software is that even though the actual text entry is performed in 3D space, or air, it requires the use of a few keys on a physical keyboard to demarcate the beginning and ending of an in-air gesture. It has a few other constraints that were mentioned in Section 2.3. The app was built using Python. Some Python dependencies were included, such as Tensorflow,

[2]http://www.yorku.ca/mack/ExperimentSoftware/SoftKeyboardExperiment

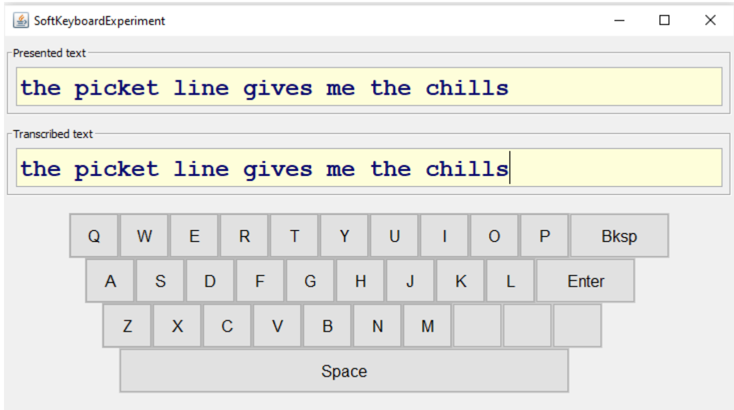


Figure 6.1: SoftkeyboardExperiment software layout for QWERTY method.

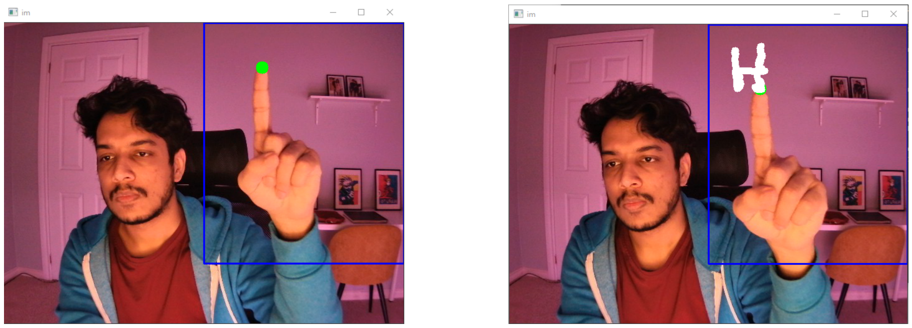
OpenCV-python, NumPy, and Keras. *GoStats*[3] – was used for statistical analysis.

To begin interaction with the *Write-in-Air* method, the user double clicks the software icon. This opens three windows. The first is the camera window which facilitates recording gestures and getting live feedback on where the gestures are drawn and what exactly is being drawn. See Figure 6.1. The second window shows the sentence that the user needs to enter via gestures, and the third window shows a real-time prediction of the gestures; that is, the sentence that is being written by the user. The process of entering text via the *Write-in-Air* method is as follows:

- After the software is launched, the user organizes their environment to have an empty stable background, then presses 'b' on the keyboard. The background must remain constant throughout the test process. This is a one-time activity. The result is a square in the camera window in which the user draws gestures. See Figure 6.1a.

- Second, the user presses 'w' on the keyboard and starts drawing a gesture in

[3]https://www.yorku.ca/mack/GoStats/



(a) (b)

Figure 6.2: The *Write-in-Air* text entry method records gestures in real-time. (a) Finger-tip detection (b) Recording of gesture.

the camera window. The resulting white-colored gesture is seen in real-time. See Figure 6.1b.

- Third, the user presses 'w' again to stop recording the gesture. This step triggers the prediction algorithm to analyse and predict the gesture. The resulting character is appended to the text shown in the third window.

- If the user wishes to input more characters, the second and third steps are repeated in a loop until the user decides to finish entering text for that trial.

- After the user is finished entering text, he/she gestures 'Enter' to exit. A dialog box appears indicating the end of testing. A CSV file is created and saved in the current directory. The file contains performance metrics such as speed and accuracy. The gestures for *enter* key and *space* key were custom-designed, as noted in section 2.3.

6.1.3 Procedure

Participants were welcomed and explained the purpose of the user study. They watched a video demonstrating the interaction methods. Participants were asked to use the hand of their preference. With this brief introduction, testing began. Participants completed five phrases per block and six blocks for each method. A participant took about 35-40 minutes to complete the experiment with adequate breaks between each block. The data were later imported in Microsoft *Excel* to analyze for results and meaningful insights. After the testing was complete, the user was prompted with a questionnaire to gather feedback regarding their experience with both methods.

6.1.4 Design

The user study was a 2×6 within-subjects design. The independent variables and levels were as follows:

- Entry method (QWERTY, *Write-in-Air*)
- Block (1, 2, 3, 4, 5, 6)

There were two dependent variables: text-entry speed (wpm) and error rate (%). The total number of trials was 720 ($2 \times 6 \times 5 \times 12$). The two input methods were counterbalanced with four participants in each group to offset learning effects. Each session lasted about 45 minutes and was divided into two 20 minute periods. One of the two layouts was assigned in each half-session period in alternating order from session to session. The order of the conditions was balanced between participants to reduce interactions. Each half-session contained six blocks of trials. A five-minute break was allowed between the two half-sessions. Each block contained five text phrases of about 45 characters each. These five phrases were randomly selected from a source file of 100 phrases. Phrases were not repeated within blocks but repeats were allowed from block to block. The phases were chosen to be representative of English. A sample phrase used in the experiment was *"the quick brown fox jumps over the lazy dog."*

Refer to Appendix for the entire phrase list of 100 phrases.

6.2 Results and Discussions

6.2.1 Text-Entry Speed (wpm)

Due to the speed-accuracy tradeoff, evaluations of human performance in text entry tasks must attend to both the speed of entry and the accompanying errors [18]. In the evaluation for text-entry speeds, the experimental software generates phrases of text for participants to enter. As a participant enters a phrase, the system records and logs keystrokes and their associated timestamps. Entry speed is the total number of characters typed divided by the time to enter them. The initial units are "characters per second". For example, a user enters 45 characters in 20 seconds. The entry speed is 45 / 20 = 2.25 characters per second (cps). This isn't exactly accurate.[4] It is essential to consider both the initial and termination points for the time estimation. If timing starts on the entry of the first character, the preparation time prompting the contribution of the first character is absent. For this reason, the total character count is decremented by one before computing the speed in characters per second. Hence, 44/20 = 2.2 cps. This is approximately 2.2% less than our initial figure. However, if the terminating time measurement is taken on a subsequent keystroke –

[4] http://www.yorku.ca/mack/RN-TextEntrySpeed.html

such as ENTER – then this keystroke is also included in the character count.

It is common to transform "characters per second" into "words per minute" (wpm). The definition of a "word" for this purpose is "five characters", including spaces or any other characters in the inputted text. The transformation requires multiplying the cps figure by 60 seconds/minute and dividing by 5 characters/word. The example mentioned above will be $2.2 \times (60/5) = 26.4$ wpm. Before about 1924, typing rates were reported using actual words per minute. Since then, rates have been reported using 5-stroke words per minute, or just "words per minute" [32]

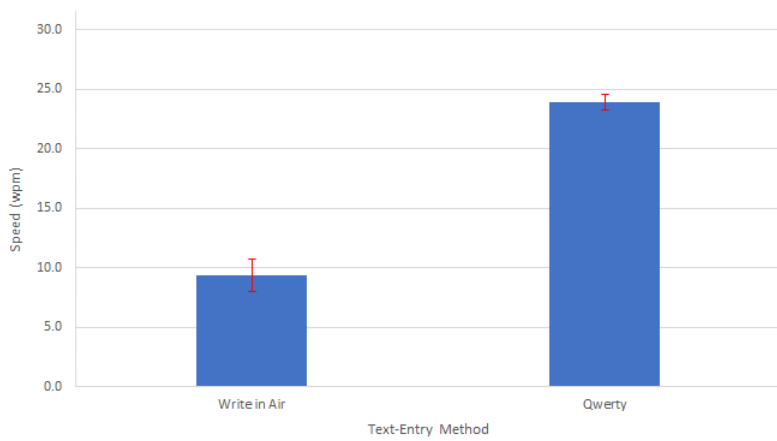


Figure 6.3: Text entry speed (wpm) by entry method . Error bars are shown in red.

The grand mean for text entry speed was 16.7 wpm. By entry method, the means were 23.9 wpm for the QWERTY soft keyboard and 9.4 wpm for the *Write-in-Air* gesture method. The effect of entry method on entry speed was statistically significant ($F_{1,11} = 811.6, p < .0001$). The effect of block on entry speed was statistically significant ($F_{5,55} = 33.17, p < .0001$). The entry speed $\times$ block interaction effect was statistically significant ($F_{5,55} = 2.70, p < .05$). Figure 6.3 shows the result for entry speed by entry method.

For each method, we derived standard regression models in the form of the power law of learning. The prediction equations and the squared correlation coefficients are illustrated in Figure 6.4. The high R^2 values infer that the fitted learning models give an excellent prediction of user behavior. In both cases over 90% of the variance is accounted for in the models. The somewhat lower R^2 value for the QWERTY method may be explained as follows. Since our participants were experienced computer users, they were familiar with the QWERTY method at the start of the study. By no means is the prediction model for the QWERTY layout capturing users' learning behavior from their "initial exposure" to the layout; subjects were "well along" the learning curve. For the *Write-in-Air* method, however, users had no prior experience with the layout, and, so, the learning model is more representative of the initial exposure and the learning thereafter.

Figure 6.4 shows the relationship between block and text-entry speed. The figure shows extrapolations to the 18th block. These lines were extended mathematically to anticipate performance with further practice.

6.2.2 Error Rate

Error rate is calculated by comparing the presented text and the transcribed text using the minimum-string distance (MSD) algorithm [27]. The grand mean for error rate was 3.7%. By entry method, the means were 1.7% for the QWERTY soft keyboard and 5.6% for the *Write-in-Air* gesture method. The effect of entry method on error rate was statistically significant ($F_{1,11} = 45.40, p < .0001$). The effect of block on error rate was also statistically significant ($F_{5,55} = 27.74, p < .0001$), as was the entry method × block interaction effect ($F_{5,55} = 16.80, p < .0001$).

Figure 6.5 shows the plot between error rate and text-entry method. Figure 6.6

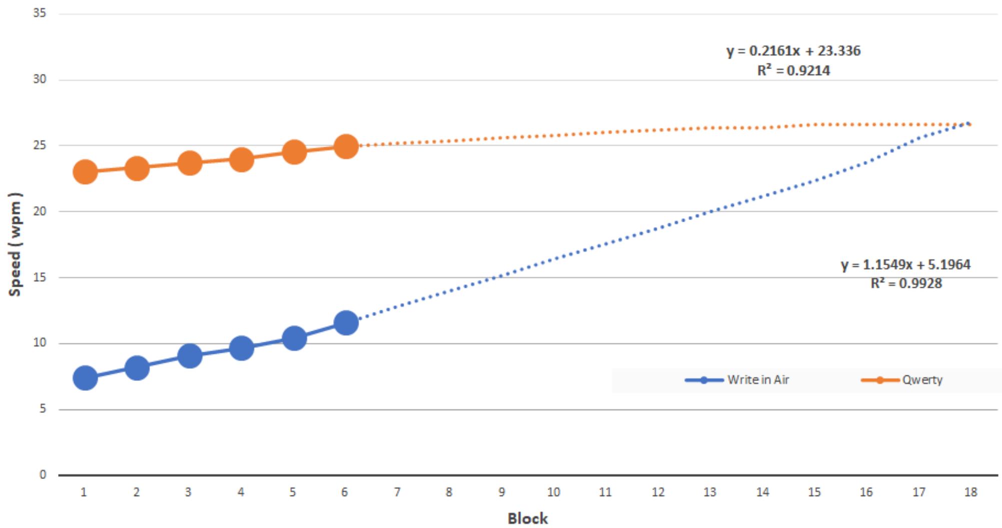


Figure 6.4: Text-entry speed (wpm) by Block. Dotted lines project forward to the 18th block.

shows the plot between block and error rate for both the text-entry methods.

6.2.3 Participant Feedback

In the post-experiment questionnaire, participants were asked their preference of the text-entry method on a scale of 1-10. A Wilcoxon Signed-Rank test was performed using *GoStats*. The results indicate that the difference in the responses was not significant ($z = -0.267, p > .05$). Participants appreciated the new *Write-in-Air* method for its innovation and touchless sensor-based capabilities. One participant noted, *Write-in-Air method is innovative. Once, I get the hang of it, I could avoid errors easily and intuitively. It makes text entry so futuristic and simple.*

Overall participants gave a rating of 7.5 out of 10 for *Write-in-Air* method and 7.3 out of 10 for the *QWERTY* method.

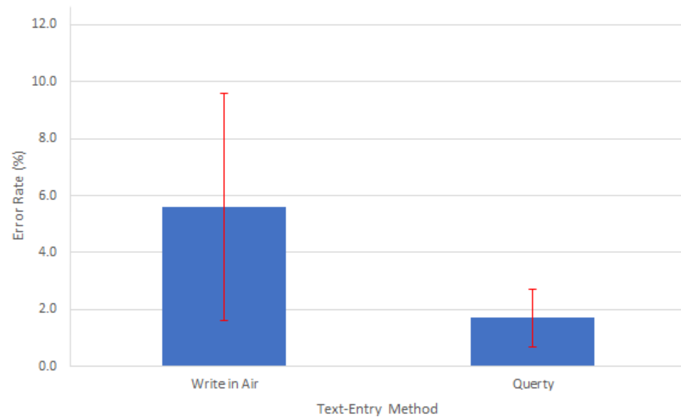


Figure 6.5: Error rate (%) by text-entry method. Error bars are shown in red.

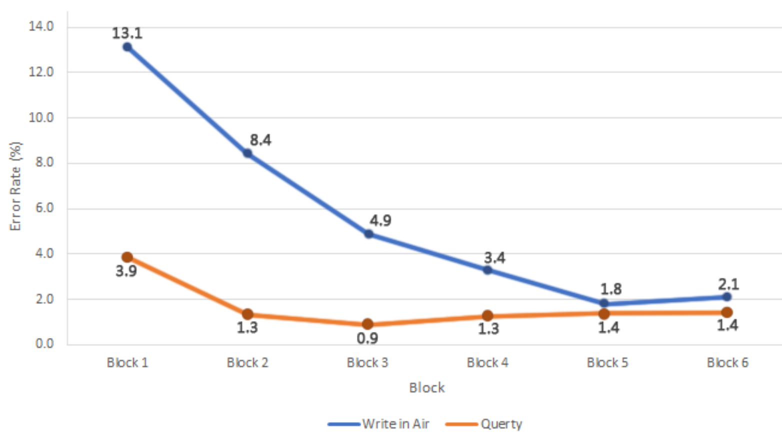


Figure 6.6: Error rate (%) by block and entry method.

Chapter 7

Conclusion

7.1 Findings from the First Experiment

A user study was conducted comparing the performance of three scroll navigation methods. Gesture-based and sensor-based scroll methods were compared and analyzed for performance and accuracy. The results were more accurate for our newly introduced method (*Fingerprint Scroll*) compared to two alternatives. The accuracy for fingerprint scroll was higher, with an average accuracy of 97.6% compared to 96.3% for *Tap Scroll* and 88.7% for *Kinetic Scroll*. The completion time was relatively long for *Fingerprint Scroll* compared to other traditional alternatives like *Kinetic Scroll*. Overall participants gave a favorable and preferential rating for *Fingerprint Scroll* for ebook reading and web browsing over mobile devices.

7.2 Findings from the Second Experiment

An experiment was conducted comparing the performance of three zooming methods when performed using one hand. The results for our newly introduced methods were significantly better compared to the traditional *Pinch-to-Zoom* method (performance using a single hand). *GyroZoom* performed 18.1% faster than the traditional *Pinch-to-Zoom* method and was more efficient. *VolumeZoom* was the most efficient method of all three but was 18.9% slower than the *Pinch-to-Zoom*. Overall, the participants gave a favorable and preferential rating for using *GyroZoom* and *VolumeZoom* over *Pinch-to-Zoom* for one-handed interaction.

7.3 Findings from the Third Experiment

In our third experiment, we compared *Write-in-Air* text-entry method with the QWERTY method. Since *Write-in-Air* method is a 3D gesture-based method we did not expect it to perform closely with the QWERTY method but by the power law of learning plot between speed and block, we learned that text-entry speed of *Write-in-Air* steadily increases with more practice of blocks. We performed testing with six blocks of five sentences each and projected the performance over the next 20 blocks and the performance of *Write-in-Air* method approaches the performance of the QWERTY method. Similarly, *Write-in-Air* method had a higher error rate initially because of lack of familiarity and practice but after six blocks of practice, it has been significantly reduced. Our results indicate that with four hours of practice, both methods will perform closely in terms of speed and accuracy. With 6 blocks of practice, the mean speed of the *Write-in-Air* method is 9.4 wpm as compared to QWERTY 23.9 wpm and the mean error rate of *Write-in-Air* method is 5.6% as compared to QWERTY 1.7%.

7.4 Future Work

We have conducted the experiments in a completely reproducible manner. There is always scope to take this research forward with improvements and bring new concepts into view. A few ideas for future work are listed below.

- Design a touchless scroll navigation method that tracks eye movement to scroll user interfaces or eye blinks to turn pages.

- Use the proximity of a finger from the camera can be used as a zoom method interaction for mobile phones.

- Implement a gesture-based text entry method that uses eye blinks to bracket camera-detected user hand or finger gestures.

- Implement a touchless way to input punctuation marks through gestures.

There are endless possibilities in sensor-based interaction. Virtual reality, augmented reality, voice control, and gesture control are also significant forms of hands-free interaction. Augmenting a reality on any other object doesn't necessarily mean you cannot interact with that object to be considered as a touchless interaction. There are all potential ideas for future empirical research.

Bibliography

[1] Bartlett, J. F. Rock'n'scroll is here to stay. *IEEE Computer Graphics and Applications 20*, 3 (2000), 40–45.

[2] Boring, S., Ledo, D., Chen, X., Marquardt, N., Tang, A., and Greenberg, S. The fat thumb: Using the thumb's contact size for single-handed mobile interaction. In *Proceedings of the 14th International Conference on Human-Computer Interaction with Mobile Devices and Services* (New York, 2012), ACM, pp. 39–48.

[3] Darbar, R., and Samanta, D. Magitext: Around device magnetic interaction for 3d space text entry in smartphone. In *2015 IEEE International Conference on Electronics, Computing and Communication Technologies (CONECCT)* (New York, 2015), IEEE, pp. 1–4.

[4] Darbar, R., Sen, P. K., Dash, P., and Samanta, D. Using hall effect sensors for 3d space text entry on smartwatches. *Procedia Computer Science 84* (2016), 79–85.

[5] Dobosz, K., and Stawski, K. Touchless virtual keyboard controlled by eye blinking and eeg signals. In *International Conference on Man–Machine Interactions – ICMMI 2017* (Berlin, 2017), Springer, pp. 52–61.

[6] Eslambolchilar, P., and Murray-Smith, R. Tilt-based automatic zooming and scaling in mobile devices–a state-space implementation. In *International Conference on Mobile Human-Computer Interaction* (Berlin, 2004), Springer, pp. 120–131.

[7] Eslambolchilar, P., Williamson, J., and Murray-Smith, R. Multimodal feedback for tilt controlled speed dependent automatic zooming. In *Proceedings of Annual ACM Symposium on User Interface Software and Technology – UIST 2004* (New York, 2004), ACM.

[8] Fallman, D., Lund, A., and Wiberg, M. Scrollpad: Tangible scrolling with mobile devices. In *Proceedings of the 37th Annual Hawaii International Conference on System Sciences* (New York, 2004), IEEE, p. 6.

[9] Farhad, M., and MacKenzie, I. S. Evaluating tap-and-drag: A single-handed zooming method. In *Proceedings of Human-Computer Interaction International – HCII 2018* (Berlin, 2018), (LNCS 10903), Springer, pp. 233–246.

[10] Gregor, P., Sloan, D., and Newell, A. F. Disability and technology: Building barriers or creating opportunities? *Advances in Computers 64* (2005), 283–346.

[11] Harrison, B. L., Fishkin, K. P., Gujar, A., Mochon, C., and Want, R. Squeeze me, hold me, tilt me! an exploration of manipulative user interfaces. In *Proceedings of the ACM SIGCHI Conference on Human Factors in Computing Systems – CHI '98* (New York, 1998), ACM, pp. 17–24.

[12] Harrison, C., and Dey, A. K. Lean and zoom: Proximity-aware user interface and content magnification. In *Proceedings of the ACM SIGCHI Conference*

on Human Factors in Computing Systems – CHI 2008 (New York, 2008), ACM, pp. 507–510.

[13] Igarashi, T., and Hinckley, K. Speed-dependent automatic zooming for browsing large documents. In *Proceedings of the 13th Annual ACM Symposium on User Interface Software and Technology – UIST 2000* (New York, 2000), ACM, pp. 139–148.

[14] Johnson, J. A. A comparison of user interfaces for panning on a touch-controlled display. In *Proceedings of the ACM SIGCHI Conference on Human Factors in Computing Systems – CHI '95* (New York, 1995), ACM, pp. 218–225.

[15] Kawachiya, K., and Ishikawa, H. Navipoint: An input device for mobile information browsing. In *Proceedings of the ACM SIGCHI Conference on Human Factors in Computing Systems – CHI '98* (New York, 1998), ACM, pp. 1–8.

[16] Lai, J., Zhang, D., and Wang, S. Contextzoom: A single-handed partial zooming technique for touch-screen mobile devices. *International Journal of Human–Computer Interaction 33*, 6 (2017), 475–485.

[17] MacKay, B., Dearman, D., Inkpen, K., and Watters, C. Walk'n scroll: A comparison of software-based navigation techniques for different levels of mobility. In *Proceedings of the 7th International Conference on Human-Computer Interaction with Mobile Devices & Services* (New York, 2005), ACM, pp. 183–190.

[18] MacKenzie, I. S., and Soukoreff, R. W. Text entry for mobile computing: Models and methods, theory and practice. *Human-Computer Interaction 17* (2002), 147–198.

[19] Moscovich, T., and Hughes, J. F. Navigating documents with the virtual scroll ring. In *Proceedings of the 17th Annual ACM Symposium on User Interface Software and Technology – UIST 2004* (New York, 2004), ACM, pp. 57–60.

[20] Oakley, I., and O'Modhrain, S. Tilt to scroll: Evaluating a motion based vibrotactile mobile interface. In *First Joint Eurohaptics Conference and Symposium on Haptic Interfaces for Virtual Environment and Teleoperator Systems. World Haptics Conference* (New York, 2005), IEEE, pp. 40–49.

[21] Partridge, K., Chatterjee, S., Sazawal, V., Borriello, G., and Want, R. Tilttype: Accelerometer-supported text entry for very small devices. In *Proceedings of the 15th Annual ACM Symposium on User Interface Software and Technology – UIST 2002* (New York, 2002), ACM, pp. 201–204.

[22] Pirhonen, A., Brewster, S., and Holguin, C. Gestural and audio metaphors as a means of control for mobile devices. In *Proceedings of the ACM SIGCHI Conference on Human Factors in Computing Systems – CHI 2002* (New York, 2002), ACM, pp. 291–298.

[23] Poupyrev, I., Rekimoto, J., and Maruyama, S. Touchengine: A tactile display for handheld devices. In *Proceedings of Extended Abstracts of the ACM SIGCHI Conference on Human Factors in Computing Systems - CHI EA 2002* (New York, 2002), ACM, pp. 644–645.

[24] Rekimoto, J. Tilting operations for small screen interfaces. In *Proceedings of the 9th Annual ACM Symposium on User Interface Software and Technology – UIST 1996* (New York, 1996), ACM, pp. 167–168.

[25] Rustagi, S., Garg, A., Anand, P. R., Kumar, R., Kumar, Y., and Shah, R. R. Touchless typing using head movement-based gestures. In *Proceedings of the IEEE Sixth International Conference on Multimedia Big Data* (New York, 2020), IEEE, pp. 112–119.

[26] Smith, G. M., and schraefel, m. The radial scroll tool: Scrolling support for stylus-or touch-based document navigation. In *Proceedings of the 17th Annual ACM Symposium on User Interface Software and Technology – UIST 2004* (2004), pp. 53–56.

[27] Soukoreff, R. W., and MacKenzie, I. S. Measuring errors in text entry tasks: An application of the levenshtein string distance statistic. In *Proceedings of the Extended Abstracts of the ACM SIGCHI Conference on Human Factors in Computing Systems – CHI EA 2001* (New York, 2001), ACM, pp. 319–320.

[28] Ti, J., and Tjondronegoro, D. Tiltzoom: Tilt-based zooming control for easy one-handed mobile interactions. In *Australian Computer-Human Interaction Conference (24th)* (2013), pp. 1–1.

[29] Weberg, L., Brange, T., and Hansson, Å. W. A piece of butter on the pda display. In *Proceedings of the Extended Abstracts of the ACM SIGCHI Converene on Human Factors in Computing Systems – CHI EA 2001* (New York, 2001), ACM, pp. 435–436.

[30] Wigdor, D., and Balakrishnan, R. Tilttext: Using tilt for text input to mobile phones. In *Proceedings of the 16th Annual ACM Symposium on User Interface Software and Technology – UIST 2003* (New York, 2003), ACM, pp. 81–90.

[31] Xu, W., Liang, H.-N., Zhao, Y., Zhang, T., Yu, D., and Monteiro, D. Ringtext: Dwell-free and hands-free text entry for mobile head-mounted displays using head motions. *IEEE Transactions on Visualization and Computer Graphics 25*, 5 (2019), 1991–2001.

[32] Yamada, H. A historical study of typewriters and typing methods, from the position of planning japanese parallels. *Journal of Information Processing 2* (1980), 175–202.

[33] Yee, K.-P. Peephole displays: Pen interaction on spatially aware handheld computers. In *Proceedings of the ACM SIGCHI Conference on Human Factors in Computing System – CHI 2003* (New York, 2003), ACM, pp. 1–8.

[34] Zhai, S., Smith, B. A., and Selker, T. Improving browsing performance: A study of four input devices for scrolling and pointing tasks. In *Proceedings of INTERACT '97* (Berlin, 1997), Springer, pp. 286–293.

Appendices

Appendix A : Phrase list used in third experiment

- sharp cheese keeps the mind sharp
- a steep learning curve in riding a unicycle
- rent is paid at the beginning of the month
- if you come home late the doors are locked
- employee recruitment takes a lot of effort
- nothing finer than discovering a treasure
- for murder you get a long prison sentence
- burglars never leave their business card
- knee bone is connected to the thigh bone
- irregular verbs are the hardest to learn
- an enlarged nose suggests you are a liar
- a good stimulus deserves a good response
- the accident scene is a shrine for fans
- safe to walk the streets in the evening
- pumping helps if the roads are slippery
- did you see that spectacular explosion
- construction makes traveling difficult
- handicapped persons need consideration
- salesmen must make their monthly quota
- saving that child was an heroic effort
- microscopes make small things look big
- insurance is important for bad drivers
- I cannot believe I ate the whole thing
- the biggest hamburger I have ever seen
- gamblers eventually loose their shirts
- they might find your comment offensive
- important news always seems to be late
- dormitory doors are locked at midnight
- that referendum asked a silly question
- the chamber makes important decisions
- the elevator door appears to be stuck
- please try to be home before midnight
- a thoroughly disgusting thing to say
- that agreement is rife with problems

– that land is owned by the government
– one never takes too many precautions
– dinosaurs have been extinct for ages
– the treasurer must balance her books
– a much higher risk of getting cancer
– gun powder must be handled with care
– weeping willows are found near water
– questioning the wisdom of the courts
– what a monkey sees a monkey will do
– bank transaction was not registered
– the presidential suite is very busy
– the punishment should fit the crime
– keep receipts for all your expenses
– where did you get such a silly idea
– our housekeeper does a thorough job
– granite is the hardest of all rocks
– every Saturday he folds the laundry
– suburbs are sprawling up everywhere
– dolphins leap high out of the water
– an injustice is committed every day
– look in the syllabus for the course
– rectangular objects have four sides
– a tumor is OK provided it is benign
– everybody looses in custody battles
– the picket line gives me the chills
– neither a borrower nor a lender be
– the music is better than it sounds
– my car always breaks in the winter
– universally understood to be wrong
– the protesters blocked all traffic
– stiff penalty for staying out late
– the pen is mightier than the sword
– beautiful paintings in the gallery
– this camera takes nice photographs
– a security force of eight thousand
– interactions between men and women
– would you like to come to my house
– a feeling of complete exasperation
– the cat has a pleasant temperament

– he underwent triple bypass surgery
– careless driving results in a fine
– that sticker needs to be validated
– the fire raged for an entire month
– labour unions know how to organize
– I like baroque and classical music
– an inefficient way to heat a house
– one hour is allotted for questions
– good jobs for those with education
– taking the train is usually faster
– tell a lie and your nose will grow
– staying up all night is a bad idea
– motivational seminars make me sick
– rejection letters are discouraging
– a good joke deserves a good laugh
– a dog is the best friend of a man
– vote according to your conscience
– the rationale behind the decision
– just like it says on the can good
– electric cars need big fuel cells
– shivering is one way to keep warm
– try to enjoy your maternity leave
– prescription drugs require a note
– our life expectancy has increased
– raindrops keep falling on my head
– do you get nervous when you speak
– parking tickets can be challenged
– everyone wants to win the lottery