UNIVERSITY OF CALIFORNIA
SANTA CRUZ

LEARNING VISUAL-BASED HEAD-BASED POINTING

A dissertation submitted in partial satisfaction of the
requirements for the degree of

DOCTOR OF PHILOSOPHY

in

COMPUTER ENGINEERING

by

Muratcan Cicek

June 2024

The Dissertation of Muratcan Cicek
is approved:

Professor Roberto Manduchi, Chair

Professor Sri Kurniawan

Professor Yang Liu

Peter F. Biehl
Vice Provost and Dean of Graduate Studies

Table of Contents

List of Figures

List of Tables

Abstract

Learning Visual-based Head-based Pointing

by

Muratcan Cicek

Devices such as computers, smartphones, tablets, and other smart gadgets have become integral to daily life, enhancing productivity and connectivity. A fundamental aspect of Human-Computer Interaction is pointing, which involves selecting and manipulating objects on display using a pointing device such as a mouse, touchpad, or stylus. However, such devices require fine control of the wrist and fingers. Those with poor upper limb mobility or control can use technology such as eye gaze tracking or head tracking for pointing control. In particular, head tracking does not require a dedicated device (since it can be accomplished by analyzing images of the user taken by a screen camera). It was shown to be more effective and acceptable than eye gaze tracking.

This dissertation first studies visual-based head-based pointing in different settings. It starts by proposing a mobile head-based pointing in the context of online shopping. Then, the study is extended by proposing a more general mobile head-based pointing and conducting Fitts' Law studies with people with motor impairments. User studies demonstrated the method's robustness across different environments. These initial evaluations also suggest that head-based pointing solutions with predefined control mechanisms can be limited. For example, the pointer motion may not accurately reflect the user's intent, forcing them to move their head in "non-natural" ways to accomplish specific pointing tasks.

The second phase of the dissertation proposes a user-centric approach to create a flexible pointing algorithm that adapts to the user's intent. This approach involves collecting data by asking participants to follow with their heads the motion of a target on the screen, moving through pre-defined trajectories, while images of their heads are taken by a screen camera. The videos thus recorded were analyzed using computer vision algorithms. This analysis considered several types of features, such as facial landmarks and head poses. An affine transformation was computed using least squares regression to map these features to pointer locations on a screen. The analysis revealed unique head movement patterns among individuals performing similar pointing tasks, indicating that the relationship between head position and desired pointer motion is complex and conditional, with biases based on the location and direction of pointing. To improve pointing precision, I also implemented fully connected neural networks (FCNs) and recurrent neural networks (RNNs) based on the features extracted from video frames. In my evaluation, I considered different visual feature sets and personalization techniques. The results of this analysis show the potential of using advanced transformation models and carefully selected feature sets to enhance the accuracy of head-based pointing systems. The findings underscore the importance of personalization and selecting appropriate mapping types and feature sets in designing efficient and user-friendly head-based pointing. As technology advances, these insights could pave the way for more intuitive and accessible human-computer interaction mechanisms.

To those who cannot utilize the default computer interfaces...

Acknowledgments

I would like to begin by expressing my deepest gratitude to my professor, Roberto Manduchi. I am immensely thankful for the opportunity he provided, which allowed me to embark on a fascinating journey that reshaped my identity through unique and beneficial experiences. Over the past seven years, his generous and insightful mentorship has not only honed my problem-solving skills but also taught me the essentials of research methodology. He supported me through the challenging years the world recently faced, and his unwavering belief in me, even more than my own, kept my faith intact. I am deeply appreciative of his help, without which my current achievements would not have been possible. His impact on my life is so profound that it cannot be adequately summarized in words.

My thanks also extend to Prof Sri Kurniawan and Prof Yang Liu for their constructive feedback on my proposal and for motivating me to delve deeper into my thesis details.

I am thankful to all members of the Computer Vision Lab for their invaluable discussions and brainstorming sessions. I would like to express my heartfelt appreciation to Robinson Piramuthu, my mentor during my internship at eBay, whose guidance was pivotal to my career. Similarly, I am grateful to Jeffry Nichols, my mentor at Google, for the enriching internship experience under his guidance. I also thank Ability Now Bay Area (Oakland, CA) for their collaboration in my field study, as they allowed me to recruit their visitors and host me for two days through the study. I would like to thank Google for the Ph.D. Fellowship, which not only supported my research but also encouraged me to investigate various methodologies more thoroughly before drawing definitive conclusions.

I am indebted to my mother for her unconditional support. She was my first teacher, teaching me to read when I was initially denied access to elementary school due to my disability. She never gave up on me and took many risks to ensure I could pursue higher education. My father passed away when I was eight, leaving me nothing but big dreams. Despite having no resources, he showed me the world through TV and inspired me to explore it. His problem-solving skills and adaptability to technology led me to become an engineer. Even his absence shaped my personality, as I often bore the responsibility of progress while my family struggled to survive.

My gratitude goes out to the Turkish community at UC Santa Cruz and the Bay Area for their unwavering support and kindness.

Lastly, I thank Cesar Mauri Loba (Universitat Rovira i Virgili) who built and open-sourced Enable Viacam that provides head-based pointing for millions for years. After all, this dissertation is written with this software.

Chapter 1

Introduction

In the past few decades, the use of electronic devices has significantly expanded across various domains, including personal, professional, and industrial applications. Devices such as computers, smartphones, tablets, and other smart gadgets have become integral to daily life, enhancing productivity and connectivity[28, 48]. The proliferation of these devices has led to the development of various interaction methods to improve user experience and accessibility. Human-Computer Interaction (HCI) is a field of study focused on designing and using computer technology, particularly the interfaces between users and computers. The primary goal of HCI is to create intuitive and efficient interactions between humans and machines. Common interaction methods include the traditional mouse and keyboard, the most widely used input devices, allowing precise control and text input. Touchscreens, found in smartphones, tablets, and some laptops, enable direct interaction with the display. Voice control uses speech recognition technology to interpret and execute user commands. Gestural interfaces employ cameras and sensors to detect and interpret body movements, while eye tracking monitors eye movements to control the cursor and other interface elements.

A fundamental aspect of HCI is the pointing task, which involves selecting and manipulating objects on a display using a pointing device such as a mouse, touchpad, or stylus. The efficiency of pointing tasks is crucial for overall user performance and satisfaction[13, 61, 93]. The effectiveness of these tasks can be measured by speed, accuracy, and ease of use. Evaluating the performance of different HCI methods is essential to determine their suitability for various applications. Evaluation typically involves user studies, where participants perform specific tasks using the interface. Metrics such as task completion time, error rate, and user satisfaction are recorded and analyzed.

Several methods are currently available for accomplishing pointing tasks. The traditional mouse provides precise control but requires a flat surface, which can lead to repetitive strain injuries. Trackpads and touchpads, integrated into laptops, offer a compact alternative to the mouse. Stylus and pen input used primarily with tablets and graphic design tools, allow for the detailed input. Head-based pointing uses head movements tracked by a camera to control the cursor, which benefits users with limited hand mobility. Eye-tracking devices allow users to control the cursor with their gaze, which is useful in accessibility applications.

User studies are critical for understanding the practical implications of different pointing methods[61]. These studies typically involve participants completing predefined tasks while researchers measure various performance metrics. Such studies help identify the strengths and weaknesses of each method and guide the development of more effective interfaces. Fitts' Law is a predictive model of human movement primarily used in HCI to quantify the difficulty of pointing tasks. The law states that the time required to move to a target area is a function of the distance to the target and the size of the target. Mathematically, it is expressed as:

$$T = a + b \log_2 \left(\frac{D}{W} + 1 \right) \tag{1.1}$$

where T is the average time to complete the movement, a and b are empirically determined constants, D is the distance to the target, and W is the width of the target. Fitts' Law is widely used to evaluate the performance of pointing devices and to design user interfaces that facilitate faster and more accurate pointing[3, 61, 96].

For individuals with motor impairments, performing pointing tasks can be particularly challenging due to the necessity for fine control of the wrist and fingers. Fine motor control is essential for precise cursor movements and accurate selection of small interface elements. Many people with motor impairments lack this fine control of the upper limbs due to various conditions, such as cerebral palsy, muscular dystrophy, multiple sclerosis, or spinal cord injuries[78, 79, 80]. These conditions can result in muscle weakness, spasticity, tremors, and involuntary movements, all of which impede the ability to use traditional pointing devices effectively.

The inability to control upper limbs precisely affects computer interaction significantly, as it limits the use of conventional input devices like the mouse and keyboard. These users often face difficulties performing simple tasks such as clicking icons, dragging items, or navigating menus. This not only hampers their productivity but also affects their overall user experience and accessibility to technology. On the other hand, access to technology is crucial for people with motor impairments, as it can significantly improve their quality of life by providing greater independence and the ability to participate more fully in society. Ensuring these users have access to effective assistive technologies and alternative pointing methods is a critical step towards achieving digital inclusivity and empowering individuals with disabilities to use technology to its fullest potential[41, 55, 73].

Assistive technologies and alternative pointing methods have been developed to address the physical challenges of this user group. Assistive technologies are designed to enhance the functional capabilities of individuals with disabilities, enabling them to interact with computers and other electronic devices more effectively[21, 36, 39]. Some alternative pointing methods include voice recognition systems, which allow users to control the computer via spoken commands, and eye-tracking systems, which enable cursor control through eye movements. Additionally, head-based pointing systems, which use head movements tracked by a camera to move the cursor, provide a viable option for users with limited hand mobility[8, 15, 18, 69].

User studies evaluating the pointing abilities of people with motor impairments are essential to developing and refining these assistive technologies. Such studies typically involve participants with various motor impairments performing tasks using different pointing methods while researchers collect data on performance metrics such as speed, accuracy, and user satisfaction[11, 31, 72, 87, 104]. These studies help identify the most effective solutions and inform the design of more accessible and user-friendly interfaces.

I am a person with motor impairments, Cerebral Palsy specifically. I have relied on visual-based head-based pointing by Enable Viacam[68] since 2010. The interaction method allowed me to become a computer scientist and build a solid career, exponentially improving my living standards. As my appreciation, I dedicated my doctoral research to studying, evaluating, and improving visual-based head-based pointing. The dissertation documents this research and covers my collaborations with several parties. First, in Chapter 2, I provide a background on the literature.

Chapter 3 presents my research collaboration with eBay. During my summer internship in 2018 with eBay's computer vision team, I led a project under the supervision of Robinson Piramuthu, with significant support from colleagues Jinrong Xie and Qiaosong Wang. Our research, culminating in a peer-reviewed publication, focused on developing head-based pointing interactions for mobile applications, particularly benefiting people with motor impairments. Motivated by the difficulties these individuals face with in-store and online shopping, I proposed a research plan leveraging modern smartphones' advanced sensors and computing power, such as the iPhone X, to implement head-tracking technology. This technology was designed to facilitate hands-free interaction by enabling a button to activate via head movements, making basic activities like online shopping more accessible. Our open-source project, built on the ARKit 2 toolkit, demonstrated how augmented reality tools could be employed as assistive technology, offering significant advantages over existing solutions by enhancing mobility and ease of use. The project not only aimed to provide an accessible shopping experience but also showcased potential applications in gaming and other hands-free scenarios. The key contributions of our work include an open-source mobile user interface solution for hands-free interaction featuring accurate 3D head pose mapping, a dwelling function for touch events, a practice interface, and a demo application, thus paving the way for further development and integration of hands-free applications. The project also involved a lab-based user study following Fitts' Law to evaluate our pointing mechanism. However, Chapter 3 focused on demonstrating the applicability of head-based pointing (HBP) across various contexts without evaluating it with the target group—individuals with motor impairments.

Chapter 4 extends that research by conducting evaluations with this target group during my summer internship at Google in 2019, supervised by Jeffrey Nichols. The research highlights the effectiveness of a custom head-based pointing (HBP) as an input method for virtual reality, gaming, and accessibility. Computer vision advancements have made vision-based HBP feasible on mobile devices using built-in hardware, offering a portable and user-friendly solution for people with motor impairments. However, most existing HBP research focuses on stationary settings, with few studies including participants with motor impairments or investigating HBP on mobile devices. To address this gap, a Fitts' Law study was conducted to assess HBP's practicality on smartphones for users with motor impairments. The research identified three requirements for mobile head trackers: ease of use without cumbersome settings, precise pointing for typical user interfaces, and configurability to individual needs. The proposed technique uses neural network-based face detection for precision and supports multiple selection methods. User studies demonstrated the method's robustness across different environments and effectiveness compared to the Eva Facial Mouse[69]. The research resulted in a calibration-free, device-independent pointing mechanism, comprehensive evaluation, and open-sourced implementation, recommending HBP as a viable interaction method for mobile devices for users with motor impairments.

Based on the previous evaluations, I concluded that the head-based pointing solutions with predefined control mechanisms can be limited; for example, head tracking may not accurately reflect user intent, leading to non-natural movements. After a detailed discussion with my advisor, Prof. Roberto Manduchi, we proposed a user-centric approach to create a flexible pointing algorithm that adapts to user intent. This approach involves collecting data by asking participants to mimic marker movements on a screen with their heads, which is tracked using video frames. The collected data is then used to develop a more intuitive head-tracking system. Chapter 5 details the remote data collection strategy and analyzes the variance in head movement data across participants.

Chapter 6 analyzes the collected data, focusing on several types of features, such as facial landmarks and head poses. An affine transformation was computed using least squares regression to map these features to pointer locations on a screen. The study compared these mapped pointer trajectories with the actual trajectories of a marker followed by participants' heads. Significant discrepancies were observed, which were expected due to the lack of feedback provided to participants regarding their head movements. Furthermore, the analysis revealed unique head movement patterns among individuals performing similar pointing tasks, necessitating tailored mappings from head to pointer motion using affine transformation as the mapping function. It was found that the relationship between head position and desired pointer motion is complex and conditional, with biases based on the location and direction of pointing.

In Chapter 7, neural networks capable of learning non-linear correlations and biases through conditional neuron activation were proposed to improve precision. Evaluations involved fully connected neural networks (FCNs) and recurrent neural networks (RNNs). FCNs offered straightforward data formation adjustments, and RNNs learned sequence-to-sequence mappings to account for previous steps in estimating current pointing coordinates. Different training approaches, including stateful and stateless training, were explored. Using cross-validation, feature comparison studies with FCNs and RNNs involved training individual models for each feature set and participant. Personal models trained for specific participants were compared to generic models tested on unseen participants. Normalization effects were also studied, revealing that generic models underperformed compared to personal models. To address this, personalized models with limited end-user data were introduced, demonstrating the fine-tuning capacity of neural networks. Various fine-tuning strategies, such as adjusting learning rates and weight freezing, were tested, and the benefits of standard normalization in fine-tuning were explored to enhance the overall contribution of the work.

Chapter 2

Related Work

My research focused on proposing novel head-tracking mechanisms that provide visual-based head-based pointing on different devices as assistive technology to make those devices accessible to everyone, including people with motor impairments. Different devices come with different assistive technology options, providing alternative *selection* and *pointing* methods. Alternative text inputs or voice-based solutions can also be employed as an alternative pointing tool. This chapter presents a review of the related literature to provide a greater context for my research. The related literature includes several assistive technologies that provide or support pointing functionality.

2.1 Assistive Technology

WebAIM [103] lists several *selection* and *pointing* methods as the mainstream assistive technologies, as discussed in the following sections.

2.1.1 Selection Methods

Adaptive switches of any kind replace a necessary button pressing for mouse clicking, tapping, or for a more specific function like using a screen scanning solution. By design, adaptive switches are larger buttons that can be attached anywhere from floor to wheelchair headrests and can be hit by appropriate body parts. Tecla [100] lists the 7 most common switches as adaptive switches, including several types of buttons, joysticks, Sip-and-Puff tools, and blink recognizers. Besides these regular switches, other interesting studies propose sensitive switch [30], vision-based switches [57, 59] triggered by a smile, a tongue or tooth-click detecting device, and voice-based switches [94]. The mobile platform, iOS by Apple [46], where I developed an application during my early research, has built-in features like touch screen tapping as a switch and a front camera-based switch that detects very simple head movements as a trigger and supports external switches.

2.1.1.1 Dwelling function

The dwelling function is a selection mechanism rather than an assistive tool. Most solutions tend to have a dwelling function as a replacement for the switches I mentioned above because these solutions are designed to provide a completely hands-free interaction and greater mobility by not relying on any external switch. Instead, they require the user to hover over a target in graphical user interfaces for an activation time to select the target. As several comparative studies [64, 109] show, the dwelling method is quite slower than the clicking method for selection. It also has the Midas

Touch problem, unintentional selection due to ‘pointing-hold’ on a random target. This issue limits the design graphical user interface since bigger targets are necessary [82]. Using an external device increases the complexity of the task by requiring two separate physical abilities, pointing and button clicking simultaneously. So, with a careful choice of appropriate activation times and a specialized design, a dwelling function may still be preferable to an external switch.

2.1.2 Pointing Methods

Adaptive pointing devices are several adaptive mouse solutions with different shapes and sizes to provide easier use in the market in addition to larger touchpads and touchscreens that recognize customized gestures like Mott et al. [74] suggest. While these solutions provide greater accessibility than standard input devices, they still require fine physical ability that most people with motor impairments do not have. Also, having an external device or an extra large touchscreen counteracts the logic of complete mobility that my approach brings.

2.1.2.1 Screen Scanning mechanisms

Screen scanning mechanisms scan along the horizontal and vertical axes of the screen in a loop until the user selects the desired target by a switch. Although this is the only practical interaction technique when the user has no physical ability, it only functions as a single switch in the worst cases. This is the least efficient pointing method in terms of time since the system needs to scan two-dimensional targets in one dimension. The missing item selection cost during the scan is twice the looping time, which would be very long when the number of items is considerably large. Like most of today's popular operating systems, iOS also has a built-in scanning mechanism, Switch Control, [46] that supports both item and point scanning.

2.1.2.2 Gaze-tracking

The gaze-based pointing methods calculate an approximate gaze point by tracking the movement of eye components where the user has no other physical ability. In these unfortunate cases, gaze-based techniques [53, 66, 107] connect the user to the outside world by building direct communication. Other gaze-based solutions with more sophisticated interaction [27, 46, 71, 75] allow users to play games and browse the web. However, these solutions require a well-calibrated external device under fixed lighting conditions. Also, comparisons between head-based and gaze-based interactions [7, 54] conclude that head-based techniques are more voluntary, stable, and have greater accuracy, while gaze-based techniques would be faster for some specific tasks like typing [34]. Despite experimental studies [43, 52, 86] that have the potential to increase their accuracy on mobile platforms, the practical mobile usage of gaze-based interactions is still limited.

2.1.2.3 Head-based Pointing

The head-mounted stylus has a long history. It is used as a writing tool by attaching a regular pencil at the edge of the stylus. The first commercially available device used an ultrasound transmitter attached to a computer's screen and receivers attached to a headset to determine the location of the user's head [84]. Today, we still have physical head-mounted styluses [83] for touchscreens and sophisticated products like Quha Zono [81] and Glassouse [35]. Besides physical devices, many of today's alternative pointing methods employ visual-based interactions [67] that detect and track a voluntary movement of a body part [91] for two-dimensional pointing. Betke et al. [8] show that visual tracking of body features, especially facial [e.g., face, nose, eyebrow], can be a successful pointing tool for people with motor impairments. Mauri et al. [70] also review assistive technologies for the same user group and conclude that visual-based systems would be the only way of computer interaction for some users. Advantages include flexibility and lower cost over other traditional assistive technologies. There are many successful applications of head-based pointing in assistive technology (e.g., Camera Mouse [77], Smyle Mouse [59], Enable Viacam [68], HeadMouse Nano [22]). MacOS also embedded head pointing[47] as an internal assistive feature one year after this research. In addition to assistive technology, I would also like to highlight the huge potential of head-based tracking in other areas, including desktop GUIs [9], wearable computing [12], and VR 3D user interface [10, 20, 54].

Today, there are also ready-to-use mobile head-tracking solutions such as EVA Facial Mouse [69] and Essential Accessibility [2] available on mobile platforms that rely on head-tracking and provide considerably free control of the mobile environment without any external devices, nor requiring a sensitive calibration. But most of these products are only available on Android. On the other hand, I find that recent studies

[1, 89, 90, 91] that evaluate head-tracking on mobile devices tend to use iOS as their experimental environments. In their first study, Roig-Maimó et al. [91] propose two similar tasks to evaluate head-tracking: a picture-revealing puzzle game for pointing and an item selection task for different-sized items. In their most recent work, Roig-Maimó et al. [90] applies a user performance evaluation through Fitts' law for a mobile head-tracking interface by following the multidirectional tapping test described in the ISO standard [49] after his non-ISO study [89].

Most visual-based head-based pointing solutions rely on off-the-shelf face-tracking algorithms to capture user feedback (i.e., head movement) and convert this input into pointing coordinates on the screen. Enable Viacam [68] benefits from the Haar Cascade algorithm [101] for face detection by evaluating its source code. Camera Mouse [77] also allows users to choose the input mechanism for pointing. Besides face tracking, it includes point tracking based on simple optical flow calculation. In this setting, users can determine a small patch on their face and the software tries to keep track of this patch on the following frames. On the other hand, HeadGazeLib [19] utilizes the depth sensors of the device to locate the user's face with respect to the camera. While other methods [15, 59] use advanced deep learning algorithms to detect and track the facial features from RGB images, their conversion functions are again tailored by the developers and involve no machine learning. To the best of my knowledge, no visual-based head-based pointing solution aims to directly learn to point from the user's appearance.

2.2 Pointing Ability of People with Motor Impairments

Fitts [32] designed and ran the first experimental study evaluating human performance in target acquisition tasks. His seminal work became known as Fitts' Law and has adapted countless times to many related input tasks. The ISO 9241 standard [49] includes variations of his work for testing the performance of non-keyboard input devices. Fitts' Law techniques have also been used to evaluate pointing performance tasks for specific user groups, including individuals with motor impairments. One of the first Fitts' Law studies in this latter category was conducted by Bravo et al. [11] to compare the reaction and movement times between able-bodied and cerebral palsied groups and concluded that the cerebral palsied groups required more time to respond. Riviere and Thakor [87] also showed that movement disorders make mouse use quite inaccurate and nonlinear. Montague et al. [72] found similar performance and interaction behaviors for motor-impaired users using touchscreens on mobile devices. Similarly, Findlater et al. [31] compared touchscreen and mouse input performance by people with and without upper body motor impairments. These studies and Wobbrock [104] show that pointing at targets in graphical user interfaces, whether with a mouse, a stylus, or a touch screen, is still a serious access barrier for people with motor impairments because of the required fine motor skills. This has led researchers to work on improving the efficiency of pointing tasks through alternate methods.

Chapter 3

Mobile Head Tracking for eCommerce and Beyond

This chapter explains my research during my summer internship in the computer vision team at eBay 2018. The project was led by me and supervised by Robinson Piramuthu, the head of the team at the time. Our colleagues, Jinrong Xie, and Qiaosong Wang, also supported the project, especially in terms of running the user studies and validating the implementation. The published work [18] was written by me and peer-reviewed by my colleagues before the final submission.

I was always interested in introducing head-based pointing into different applications to show its practicality and necessity for people with motor impairments. At eBay, I was asked to propose a research plan that brings our interests together. After a literature survey, I found out that 80% of Americans shop online, according to Pew Research Center in 2016 [95]. Shopping at stores can be cumbersome as consumers do not prefer to be present at stores physically for several reasons [51]. In addition, there are also specific barriers [97] that make in-store shopping harder for people with motor impairments. I believe online shopping would address this problem since it requires considerably less effort to interact with digital devices to buy an item than to travel to the nearest store. But even this task is hard to complete for people with motor impairments. They are required to operate a pointer with a mouse-like device or tapping a certain point on the touch screen [31, 72, 87, 104]. According to The Centers for Disease Control and Prevention, there are 39.5 million adult Americans with some sort of physical functioning difficulty [33], and they potentially have difficulties during shopping, both in-store and online.

Proprietary software can emulate mouse and keyboard via head tracking. However, such a solution is not common on smartphones. Modern smartphones like the iPhone X have various sensors and abundant computing power. Unlike desktop and laptop computers, they are much easier to carry around. Therefore, it is natural to make head-tracking-based interaction available on smartphones. For example, a button, typically sensitive to touch, could capture head-tracking motion and activate when the tracking region falls within its extent. It is difficult for developers to achieve this since it requires special expertise. The work in this chapter is proposed to make it easy for application developers to use buttons that capture head-tracking events. This will make basic activities such as shopping easily accessible to people with motor impairments. Also, this could be extended to other situations demanding hand-free browsing.

I implemented such a button, and eBay open-sourced the project so developers could easily integrate it into their projects. Using the software, users could interact with the application only by head movement. This requires significantly less effort compared to using a touchscreen. I built the application on top of the state-of-the-art ARKit 2 [44] toolkit. The work shows that Augmented Reality tools can be used for assistive technology and have several advantages over the existing assistive solutions as they rely on the built-in feature and provide complete mobility.

This work focuses on its application as an assistive tool, mainly for people with motor impairments. To illustrate the use of such a modality, it provides an online shopping experience for everyone through a mobile application. Although the initial motivation was developing an accessible online shopping application, the scope of possible use cases includes other applications such as gaming and hands-free situations such as during cooking and auto-repair. A great variety of mobile applications can be empowered with a touch-free design by employing our ready-to-use interface module.

The contributions of this work are:

- An open-source solution for mobile user interfaces that provides hands-free interaction with smartphones. The code is completely open-sourced[1]. It features an accurate mapping from a 3D head pose to a virtual pointer, a dwelling function to generate hands-free touch events, a sample practice interface, and a demo application for the online shopping experience. Anyone should be able to extend this work and/or develop hands-free applications by integrating their design with its UI components.

[1] `https://github.com/eBay/HeadGazeLib`

- Design of a mobile user interface for online shopping that can benefit everyone, including those with motor impairments. To the best of my knowledge, this research is the first one that highlights the defined problem and proposes a practical solution to developing a working product.

3.1 Design Process

In this section, I discuss the design process that the team also contributed. I list my *initial observations* that I specify the problems, the *design requirements* I state how to address these problems, and the *design decisions* we make with the team to meet these requirements.

3.1.1 Initial Observations

These were my observations prior to the start of this work. Firstly, people with motor impairments have several difficulties during shopping. I overview the literature and discuss my findings in the previous section. The current interface of online shopping applications has its own barriers for people with motor impairments since it heavily relies on standard interaction methods. I highlighted the limitation of standard methods in Chapter 2. Besides, standard interaction methods (i.e., selection, pointing, and typing) need to be carefully adapted for people with motor impairments. Although sophisticated interactions introduced to Virtual and Augmented Reality inspired this work, they are nothing short of limitations in the mobile environment.

I surveyed relevant interaction methods individually in corresponding subsections under Chapter 2 and tried to choose the most optimal solution. Among alternative interaction methods for people with motor impairments, only a few provide true mobility. This is especially true on the iOS platform, where open-source tools for hands-free pointing are sparse. Although Apple ARKit is mainly intended for entertainment, e.g., Pokemon Go and Animoji, I saw the potential of taking advantage of its head/face tracking capability to empower mobile interface with a hands-free option.

3.1.2 Design Requirements

The aforementioned observations and the urgent need to enable persons with motor impairments to use mobile devices with minimum effort inspired us to create a mobile hands-free control without any auxiliary hardware that can be easily integrated into existing applications. To be specific:

- *Hands-free* is a must-have as we want to empower conventional mobile applications to be accessible to everyone, including persons with motor impairments, while reducing the friction in target-pointing and -selection.
- *Auxiliary-free* Any external device or sensor brings additional cost and undermines mobility in most cases.
- *Mobile-friendly* Nowadays, smartphones are ubiquitous and have become the main source of information we consume every day. Being mobile-friendly means maximizing the benefits of hands-free control on the most common platform and being influential in numerous existing and future mobile applications.

3.1.3 Design Decisions

Next, I explain the following design decisions to meet each requirement in the most compact way. The decisions were made after a discussion process with the team.

3.1.3.1 Interaction Methods

I propose a *Completely mobile* solution to the defined problem. After a comprehensive literature search, I concluded that people with motor impairments have serious difficulty using regular mobile devices and that the solutions rely on external devices, thus limiting mobility significantly. To make the solution *hands-free* and *auxiliary-free*, I evaluate all possible methods for human-computer interaction in the 2 section. These comparisons led me to pick head-based *pointing* with dwelling function for *selection*. With this combination, I make sure the solution relies only on the embedded features of the device - specifically the front camera since the methods I prefer are visual-based interaction methods.

3.1.3.2 Application for Online Shopping

The team wanted to enable people with motor impairments to be able to shop online on a smartphone and feel independent. With this purpose, I designed an online shopping application requiring minimum interaction effort. To keep the design even simpler, I built only a browse-based shopping application that allows users to browse products across categories. This also eliminated other complex tasks, such as searching and ranking, required for a regular online shopping application. While a browse-only shopping application did not provide a complete shopping experience, it was still important since the approach was one of the very first assistive eCommerce applications. Also, since eBay open-sourced this project, anyone could extend its functionality by adding searching logic on top of my work.

3.1.3.3 Development Environment

With the intention of developing mobile online shopping that only depends on visual-based interaction, head-tracking specifically, I searched for the most appropriate environment with the necessary functionalities. Instant head-tracking was still a challenging task on mobile devices at the time since it relied on power-hungry video processing and the ability to capture high-quality video through a front-facing camera. This solution was not practical on many mobile devices with limited resources and led me to focus on the most recent product releases only. iPhone X had True-Depth Camera [45], an impressive feature. It was originally designed to support advanced AR applications on iOS 11 and is accessible through Apple's ARKit 2 [44], an AR development SDK. ARKit 2 implements efficient on-device head-tracking and exposes precise measurements of several facial landmarks to the developer. The biggest advantage of this setup is that it only relies on the device's built-in components without requiring extra accessories. This is power efficient and portable. The open-source head-based interface components on iOS were built on top of ARKit 2. Even though the tools I built existed only on that iPhone model, which may have limited its usability at the time of this research, according to the report of Strategy Analytics [76], the iPhone X was the world's best-selling smartphone model in that year, shipping an impressive 16 million units during the first quarter of 2018. These numbers, together with the promising future of AR applications, led me to believe that most of the smartphones on the market would intend to adopt similar features in the near future. Overall, a smartphone-based solution is more affordable, portable, and beneficial than many of the common external solutions that heavily rely on separate gaze or head-tracking gadgets.

3.2 System Design and Implementation

3.2.1 Head-based Pointing

To meet the design requirements, I first implement a *hands-free*, *auxiliary-free*, *mobile-friendly* interaction system. The system consists of two main modules. A module for head-based pointing and a customized UI module for the interaction with on-screen pointer and feedback visualization.

3.2.1.1 Head-tracking

ARKit 2 comes with stable head-tracking capability where the 3D location and orientation of the tracked head can be queried at each AR session through *ARFaceAnchor* object. Abbaszadegan et al. [1] used the same technique to highlight the potential of using the True-Depth front-facing camera of an iPhone X for tracking.

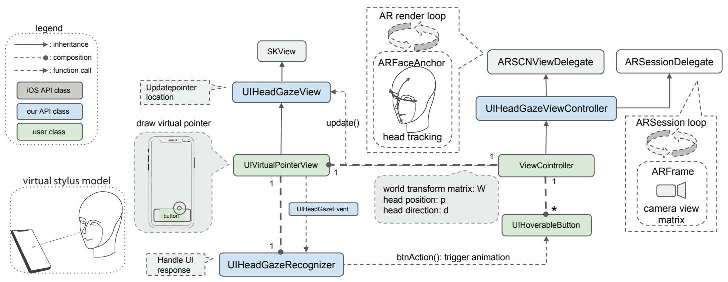


Figure 3.1: Mobile head tracking architecture to control button using head "pointing". I created a set of modules to extend native UIKit to handle head tracking, pointer visualization, head gaze event detection, triggering, and forwarding.

3.2.1.2 Head-to-pointer Mapping

However, the limitation of ARKit head-tracking is that it does not provide additional information regarding the location on the screen at which the head or nose is pointing. To bridge this gap between 3D head posture and pointing location on the phone screen, I designed an intuitive virtual stylus model to give users stable control of their pointing direction and instant feedback on the pointing location on the screen. Figure 3.1 illustrates how the model works. To aid the calculation of pointing location, I model the phone screen as a 2D plane perpendicular to the z-axis at position z=0. A proxy ray is spawned from the head center and passes through the nose. The ray-plane intersection is then calculated and is considered as the pointing location. Another possible solution is to project the nose position directly on the screen and take its screen space coordinates as the pointing location. I found that, in practice, the virtual stylus model takes into consideration the distance between the head and the phone, which allows distance-based sensitivity adjustment. Such a feature is adorable as it adapts to a UI interface with widgets/controls that vary in size, shape, and gap. To compute the intersection, let $\mathbf{p}_0 \in \mathbb{R}^4$ denote the homogeneous coordinates of the original head position in the object space, and $\mathbf{W} \in \mathbb{R}^{4\times4}$ denote the world transformation matrix from *ARFaceAnchor* object updated at each frame. The initial virtual stylus direction is denoted by $\mathbf{d}_0 \in \mathbb{R}^3$ and points to the negative z direction perpendicular to the screen. At each frame, the head center is updated to $\mathbf{p} = \mathbf{W} \cdot \mathbf{p}_0$ and the pointing direction is updated to $\mathbf{d} = \mathbf{W} \cdot \mathbf{d}_0$. The resulting ray-plane intersection $\mathbf{b}$ in Normalized Device Coordinate (NDC) space $[0, 1] \times [0, 1]$ is then calculated by $\mathbf{b} = \mathbf{p} + \mathbf{d} \times t$, where $t = -\mathbf{p}.\mathbf{z}/\mathbf{d}.\mathbf{z}$. I further apply viewport transformation to map the NDC coordinates to the screen space that can be used for the UI widget intersection test.

3.2.1.3 Pointer Visualization

To give users instant visual feedback on head-pointing, I visualize the location as an on-screen cross-hair using Apple SpriteKit API. Specifically, to cover the full screen and to provide precise tracking feedback, I add to the application a *UIHeadGazeView* layer that inherits from *SKView* to make *UIHeadGazeView* a Spritekit Scene object that serves as a canvas on rendering 2D geometry on the screen to represent pointing location.

3.2.2 UI Module

3.2.2.1 Selection Method

Native UI widgets implemented by the Apple UIKit package only react to user physical touch events such as touch down, touch up, pinch, etc. Unfortunately, there is nothing to support interacting with the UI from mid-air. This requires a new definition of interaction for head-based pointer and native UI components. To bridge this gap between the pointing location and interaction with existing UI, I defined several interactions and implemented them through customized UI widgets on top of Apple UIKit so that they can be seamlessly integrated into new iOS applications or extend the existing ones with minimum effort. For example, I created a *UIHoverableButton* class in Swift by extending iOS' default *UIButton* class to respond to the pointer interaction. Such a customized button senses the pointer's location change and triggers hovering events whenever the pointer enters the button. It will further trigger a selection event if the pointer hovers beyond a user-specified time interval, hence dwelling. During pointer hovering, a user-customized animation effect can be added to the *UIHoverableButton* to serve as a visual cue on the elapsed hovering duration. For instance, gradually increasing the button size or gradually filling the background with different colors. See Figure 3.2 for illustration.

Besides *UIHoverableButton*, other customized classes are implemented to extend native UIKit to support head tracking, pointer visualization, head gaze event detection, triggering, and forwarding. The inter-module connection is illustrated in Figure 3.1. At a high level, *UIHeadGazeViewController* keeps track of head motion and passes the head's world transformation matrix to *UIHeadGazeView*. *UIHeadGazeView* then computes the head-to-pointer mapping, updates the current pointer location, and notifies the registered event recognizer *UIHeadGazeRecognizer* about the updates. These updates are encapsulated in a *UIHeadGazeEvent* object. The design follows iOS UIKit

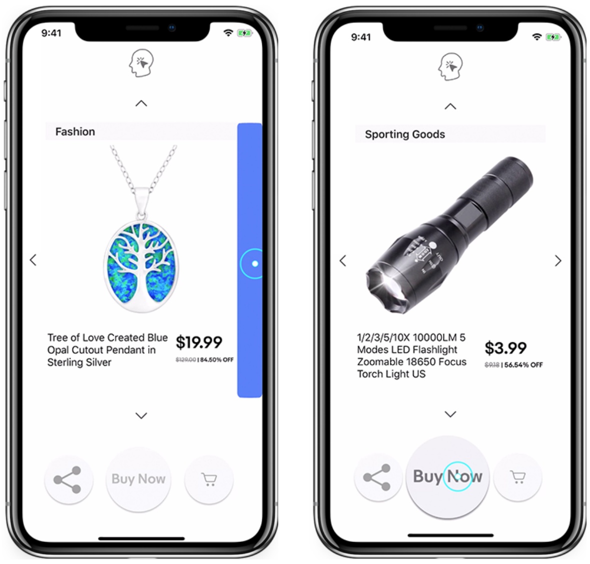

9:41
Fashion
Tree of Love Created Blue Opal Cutout Pendant in Sterling Silver
$19.99
Buy Now
9:41
Sporting Goods
1/2/3/5/10X 10000LM 5 Modes LED Flashlight Zoomable 18650 Focus Torch Light US
$3.99
Buy Now

Figure 3.2: Illustration of a sample eCommerce application that we designed and open sourced. Cursor is shown as a cyan circle with dot at its center. Navigation buttons are filled with blue when they are fully activated (left image). Action buttons expand fully for activation (right image).

protocol and blends into the regular event handling scheme. Such a design requires minimum effort to modify the existing code base to support the head-based pointing feature.

Users can select different types of *UIHeadGazeEvent* depending on how long the pointer needs to hover over a button before triggering the corresponding type of *UIHeadGazeEvent*. I currently have predefined two types based on the length of the dwelling duration: glance (1 sec) and gaze (2 secs). The trade-off has to be made on setting the duration, as a longer duration reduces the chance of accidental clicking but increases latency. By default, the duration is 1 second (glance). This configuration reduces the accidental clicking of unintended buttons while maintaining the interaction at high throughput. Furthermore, to avoid unintentional repeated clicks on the same buttons, the button is configured not to emit another *UIHeadGazeEvent* until the pointer reenters that button. With this limitation, the user has to move the pointer away from the button to emulate touch up before they can re-click the button if needed. While this is an extra effort for intentional re-clicking, the effort is actually negligible compared to the one incurred by undoing the unintentional re-clicking, which normally requires clicking different buttons to reverse it.

3.2.3 Simple Online Shopping Interface Design

Once I implemented the head-based interaction system, I integrated this system into a simple shopping application to allow people with motor impairments to shop online.

3.2.3.1 Product Browsing Page

The shopping application starts with a product browsing page, which takes up the application's main screen (figure 3.2). It presents one product at a time on the screen with a reasonably large photo along with title and price information. At the bottom of the screen are the three action buttons that allow users to share items on social media, check out items, or add items to shopping carts. Each of the three action buttons brings up a separate hands-free page for the subsequent process. For the interest of space, we focus our discussion on the main product view page.

Items are organized in a 2D grid and grouped in each row by their category. The item view is surrounded by four direction buttons, allowing users to browse items (horizontal swipe) and categories (vertical swipe). Users are presented with one cell of the grid at a time while they can swipe the item by "clicking" the direction buttons.

3.2.3.2 Open Source

eBay open-sourced the head-based pointing solution with advanced mobile UI components compatible with head-based pointing. Such a generic library can benefit researchers and developers in many different use cases. Through a sample online shopping application, we addressed several design concerns and showed the practicality of the proposed solution.

3.3 Experiments

Beyond the online shopping application I implement, I believe that the primary contribution of this work is the open-source tools for head-based pointing on iOS which is one of the most common mobile platforms all around the World. To see the end-user experience and understand these tools' relative performance, usability, and user preference, eBay supported me in conducting a field experiment with 75 able-bodied participants during an internal company showcase. Although the research states the target user group is people with motor impairments, anyone would benefit from a hands-free smartphone interaction. Providing a baseline evaluation of this system with able-bodied participants would be an important contribution since the results would inspire the community for future work. In addition to the field study, the company recruited 27 able-bodied participants for a lab-based user study to test my head-based interface for possible future work not limited by online shopping. Finally, I participated in a simpler experiment as a person (24, male) with motor impairments to try the same interface and provide feedback for comparison.

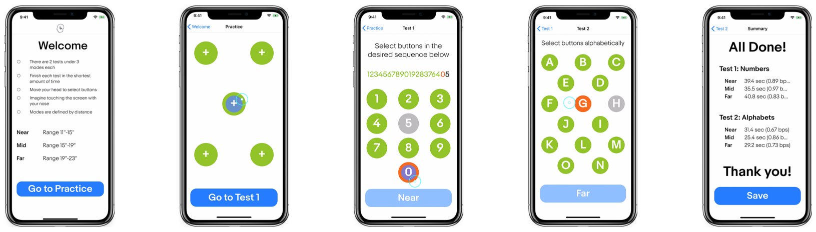
3.3.1 Participants

With the assistance of my colleagues, we conducted 2 separate experiments. In the field study, we recruited 75 able-bodied unpaid participants (35 female, 40 male) to experience the online shopping app. We randomly picked the participants during a crowded event and solicited their initial feedback on my approach right after they played with the app. In the lab-based study, we recruited 27 able-bodied unpaid participants from the company and involved them in an instructed user study. Among them, 16 were male and 11 were female. The participants had no prior experience with head-based pointing methods or the dwelling function in both experiments. They were of different ages and various races. Some of them wore glasses during the experiment. With this diverse group of participants, we intended to get a realistic insight into the learnability and usability of the system from novice adopters of the technology since this methodology is uncommon, especially on mobile. These studies are insightful on the practicality of our approach to a real-world application such as an online shopping solution.

3.3.2 Design and Procedure

I implemented a separate UI design for the lab-based study as a single mobile application. We conducted the experiment by running this App on an Apple iPhone X running iOS 11.2 with a resolution of 1125×2436 px and a pixel density of 458 ppi. This corresponds to a resolution of 375×812 Apple points (pt), an abstract unit covering two pixels on retina devices. The camera was also the iPhone X's embedded 7-megapixel front-facing True-Depth Camera [45]. We fixed the phone's position by placing it on a holder in portrait mode. The participants were required to sit against the phone at 3 different distances for each test. We placed the phone on a holder attached to an adjustable table. Before the experiment, participants were also informed of the purpose and instructed to complete the tasks. The experiment lasted about 15 minutes per participant.

The user study App for this experiment had 5 different screens (figure 3.3) to show in order during each session. The user study App opens with a welcome screen with the instructions for the overall experiment. Then, it has an unlimited practice session that familiarizes the participants with head-based pointing. It includes two consecutive tests with different layouts on separate pages to get different feedback on participants' performance. During the experiment, the app also collected the timestamps of participants' each action and the cursor's position for analysis. At the end of this experiment, we also asked each participant to fill a questionnaire for recording their user experience. The experiment is carried out on two layouts with three distances:


Welcome

- There are 2 tests under 3 modes each
- Finish each test in the shortest amount of time
- Move your head to select buttons
- Imagine touching the screen with your nose
- Modes are defined by distance

Near	Range 11"-15"
Mid	Range 15"-19"
Far	Range 19"-23"

Select buttons in the desired sequence below

12345678901928376405

Select buttons alphabetically

All Done!

Test 1: Numbers

Near	39.4 sec (0.89 bp...
Mid	35.5 sec (0.97 b...
Far	40.8 sec (0.83 b...

Test 2: Alphabets

Near	31.4 sec (0.67 bps)
Mid	25.4 sec (0.86 b...
Far	29.2 sec (0.73 bps)

Thank you!

(a) (b) (c) (d) (e)

Figure 3.3: UI design for user study via 2 tests. Each test is performed at 3 different distances between face and screen (near, mid, far). A test is complete when the desired sequence is complete. The next button in the sequence to be selected is shown in orange. To reduce response time, I highlight the following button in the sequence in gray. Buttons fill up blue during dwell time, after which the next button is highlighted in orange and the following button in gray. (a) welcome screen with instructions (b) unlimited practice session for warm-up in different distance modes (c) first test for densely packed buttons around the center of the screen (d) second test with buttons uniformly distributed across the screen (e) summary is shown in the final screen.

Near (11 to 15 inches), Mid (15 to 19 inches), and Far (19 to 23 inches).

Test 1 (Numbers) was designed for densely packed targets around the center of the screen (figure 3.3c). The first test screen contains 10 targets labeled from zero to nine by digits, similar to the numeric keypads. In addition to highlighting the next target in orange by the predefined order, I also displayed the whole desired sequence on the upper part of the screen and colored the sequence accordingly. The desired order was [12345678901928376405]; we especially selected this sequence to catch participants' performance on vertical, horizontal, and diagonal trials with different lengths.

Test 2 (Alphabets) was with targets uniformly distributed across the screen (figure 3.3d). I aimed to see the reachability of targets positioned on different parts of the screen. Test 2 had 15 same-sized targets which were labeled by the letters, from A to N. The targets had almost the same distances with their neighbors and were required to be selected in the alphabetic order. Here, I expected that the participants could select the targets around the center of the screen easily, but it could be hard to select the targets far from the center, especially the targets at the bottom.

In total, there were 27 participants $\times$ 3 distances $\times$ (20 + 15 targets) = 2835 trials in the user study.

3.3.3 Results

The lab-based study tended to evaluate the practicality of the proposed interaction precisely. Therefore, I applied a quite similar study with Fitts' Law [32] since it is the standard way to evaluate this kind of approach and derive the dependent measure throughput (Fitts' *index of performance*) as part of the comparison and evaluation [89].

3.3.3.1 Objective Evaluation

I calculated throughput (TP) as follows:

$$TP = \frac{Effective\ index\ of\ difficulty}{Movement\ time} = \frac{ID_e}{MT}, \tag{3.1}$$

where ID_e is derived from the movement amplitude A, and effective target width W_e and MT is averaged movement time per trial over a sequence. They have units "bits" and "seconds" respectively; the units of TP are "bits per second (bps)". The effective index of difficulty is also a measurement of the difficulty and user precision in completing a task:

$$ID_e = \log_2(\frac{A}{W_e} + 1), \tag{3.2}$$

where A is the movement amplitude, the distance between the centers of two consecutive targets, and W_e is the effective target width, calculated from the width distribution of selection coordinates made by a participant over a sequence of trials, which is calculated as below:

$$W_e = 4.133 \cdot S_x, \tag{3.3}$$

Note that I used the standard deviation method to calculate throughput [63]. This is because my specific design utilizes the dwelling function and has no *error rate* [108]. Also, I followed Roig-Maimó's [89] equations and applied our own Non-ISO tests. A detailed description of the calculation of throughput can be found in [89, 108]. I show the summary result of dwell time and throughput in Fig. 3.4. The top row shows the average dwell time of all users at the 3 distances. The box and whisker plot goes from the lower to upper quartile values of the dwell time data collected from all users at all 3 distances, with a line at the median. The whiskers extend from the box to indicate the data range. The median value of the data range is plotted in black, while mean values per distance are shown in the legend. I also follow the same fashion to show the throughput per motion sequence in the middle row. Finally, at the bottom row we show throughput per user averaged over all motion sequences.

For the *Numbers* test, it is worth noting that it is easier for the participants to navigate in the *Mid* range compared to the *Near* range, and most participants completed the test in the shortest time under the *Far* range setting. This is because the spacing between the buttons is relatively large in this test. If the users are operating at a

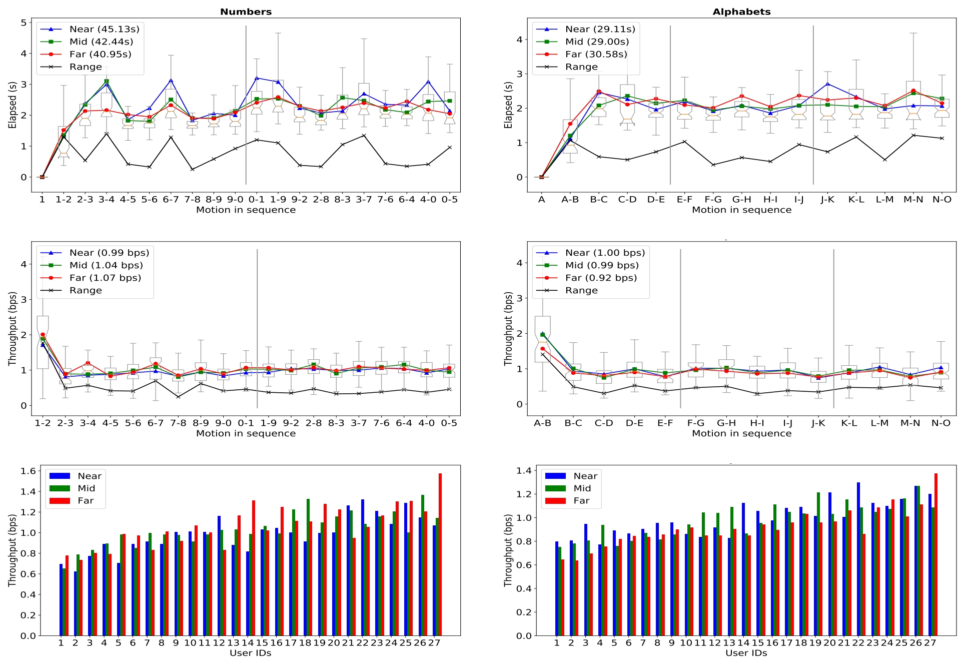


Figure 3.4: Analysis of dwell time and throughput. Top: elapsed time per motion sequence averaged over all participants. Middle: Throughput per motion sequence averaged for all participants. I also show box and whisker plots for data collected from all participants at each sequence. The range plot shows the median of the range of a set of data collected from all participants across all 3 distances. Bottom: throughput per participant averaged over all motion sequences, sorted by throughput averaged across all distances. Note that for the numbers experiment, the throughput is generally higher at far distances, while for the alphabet experiment, throughputs are usually higher at near distances. See Sec. 6 for details

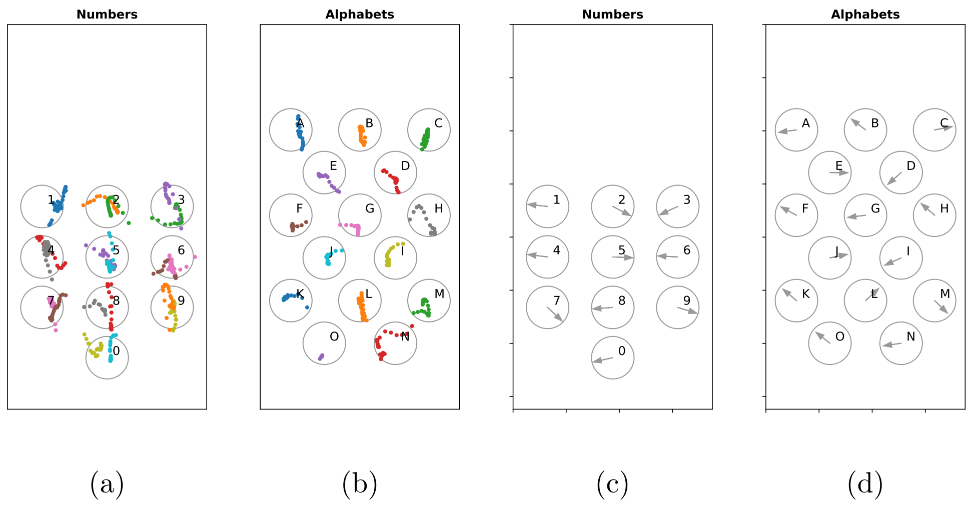


Figure 3.5: Analysis of dwell points. (a-b) Randomly selected dwell points from a participant. (c-d) Directions of eigenvectors derived from dwell point coordinates.

closer distance to the screen, it requires more head movements to control the cursor to move across larger distances. This is especially true when the participants perform large diagonal movements from number 0 to 1 or from 1 to 9. On the contrary, for the *Alphabets* test, participants complete under the *Near* setting in the shortest amount of time with the highest throughput. As the button size and spacing become smaller (See Fig. 3.3), it is easier to navigate in the *Near* range as it offers more precise control. The above observations can also be verified at the bottom plot from Fig. 3.4.

For the *Numbers* experiment, throughputs are generally higher at far distances, while for the *Alphabets* experiment, throughputs are usually higher at near distances. In addition to throughput calculation, we also studied the direction of selection points with respect to buttons at different locations (see Fig. 3.5). I calculated the eigenvectors of selection point coordinates on the covariance matrix to see if there is a pattern. I found that for the first column of buttons, statistics show that most users point to the left side. This may indicate that the participants are trying to position the cursor more toward the left side of the screen, and most of the cursors overshoot the center position of the buttons. For the center column of the buttons, the eigenvectors show a pattern of a path placed more toward the right side. For the third and final column, most participants tend to reverse the path more toward the left side in order to hit the buttons sequentially. On the third row, the pattern changed compared to the previous two rows, indicating participants getting more familiar with the app and thus exhibiting better control.

3.3.3.2 Subjective Evaluation:

We applied a questionnaire to the participants of the lab-based study right after their individual sessions. 96.3% of participants considered moving the cursor by head was easy and normal. The majority of them also reported that the selection method was normal. 48.1% reported no scientific difference between the difficulty of selection based on the target regions (upper, lower, and middle portions of the screen in portrait mode). Only 18.5% thought the lower region was the hardest to reach, while 29.6% disagreed and said the targets at the upper region were the hardest to select. The majority of the participants claimed that practicing the tests by keeping the distance at mid from the phone was easier to use the system, while 29.6% said the far distance was easier.

As a person with motor impairments, I also completed both tests and experienced the shopping app. I completed the first test slower than other participants since I had difficulty keeping my head stable for dwelling. I showed average performance on the second test as I got more familiar with dwelling and the software interface. Ultimately, I found the system quite practical for several use cases, including typing. I think the system's pointing accuracy is quite better than the software I use on a daily basis for communication, which runs on Windows via a webcam.

We also applied a questionnaire to the participants who experienced my online shopping interface during the field study. We sought their initial thoughts about the proposed hands-free online shopping App. Among 75 participants, 51 of them thought this was a useful idea. 41.3% reported the app was easy to use even though they were unfamiliar with this interaction method. 31 participants also found it enjoyable, while 5 of the participants could not find a use case and said this solution was not for them. Only 9.3% claimed moving the cursor by head was hard for them, while 52% thought it was easy, and 38.7% thought it was normal to practice. Furthermore, 50 participants reported that this solution would fit somebody's needs they know.

3.4 Discussion

I pointed out that online shopping has several barriers for people with motor impairments. The proposed design led me to address this problem with a *hands-free*, *auxiliary-free*, *completely mobile online shopping sample application.* My application allows users to browse several deals from different categories and select/share the desired products. Users do not need any external device or a fine calibration step to use my approach. This eliminates the need for a touch screen, which requires fine motor control. The initial user feedback also shows that this interaction with an online shopping application is practical and enjoyable. Beyond this application, this work actually proposed *hands-free* interface components as an open-source solution so that developers may incorporate head tracking into their mobile applications, not limited to eCommerce. The open-source solution can be used in several applications, from communication tools for people with motor impairments to a hands-free recipe or DIY application.

The lab-based study suggests keeping the phone at mid-distance (15 to 19 inches) from the face increases the interaction quality and is more comfortable since it provides an accurate and robust interaction. There is a trade-off between cursor speed and stableness. To be specific, small phone-to-head distances require bigger head movements that can easily cause fatigue but deliver fine control with higher stableness, while larger distances require minor head movements but with higher precision and stable head movements, which is potentially stressful as well. Future work on head-based interaction may consider this fact and design its interface accordingly. Another fact is that, while the majority of the participants reported that there was no significant difference between the difficulty of selection based on the target regions, I concluded that the targets around the center of the screen are more reachable than the targets on the lower portion of the screen since there are higher throughput values for the center sequence of *Alphabets* test (see Fig. 3.4). This is natural since pitching the head up or down is a relatively more uncomfortable task than yawing the head left or right. Placing the frequently visited targets around the screen center while reserving the lower portion for information display or less commonly used targets like the ones in the provided design (Fig. 3.2) would provide a more ergonomic interaction via head-based pointing on mobile and keeps the amount of uncomfortable pitching tasks to the minimum.

This research highlights the potential of the proposed approach as an important number of people who experienced our approach considered it useful for them in several use cases such as baby feeding or fixing automobiles with dirty hands. In addition to the able-bodied user group, I was also able to practice precise target selection on such a small screen for the very first time as a person With motor impairments. I can also add that hands-free smartphone interaction is especially a serious need for wheelchair riders, even if they have fine hand control since they also want to keep interacting with their phones while moving. In so many cases like this, the open-sourced, hands-free UI components would be really helpful for developers to build new applications in consideration of these needs. Therefore, future work would be developing any kind of mobile applications on top of the published framework or adding new head-pointing sensitive UI components to the open-sourced tool set.

Chapter 4

Designing and evaluating head-based pointing on smartphones for people with motor impairments

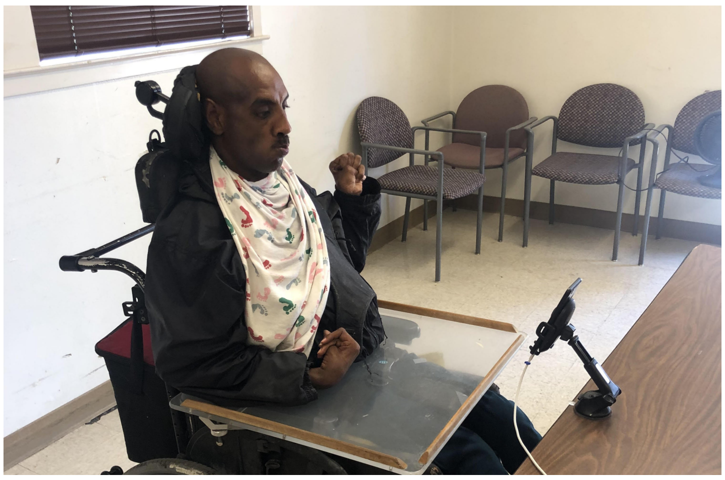


Figure 4.1: A user is using our mobile head tracker.

I was mainly focused on showing the applicability of head-based pointing (HBP) in different concepts in the previous chapter. Also, during that research, I had no chance to evaluate it with the target group, people with motor impairments. In this chapter, I extended the first research and completed the evaluation by conducting a user with the target group. The chapter explains my research during my summer internship at Google 2019. The project was led by me and supervised by Jeffrey Nichols, the team manager of a user experience research team at the time. Several colleagues in the team also supported the project, especially in terms of running the user studies and validating the implementation. The published work [15] was written by me and peer-reviewed by my colleagues before the final submission.

HBP is an effective input method for Virtual Reality and Gaming [10, 20, 40, 54], as well as for accessibility [8, 64, 65, 67, 70, 91]. As computer vision techniques advance, vision-based HBP has become feasible for interaction on mobile devices using their built-in hardware. HBP on smartphones can leverage the front-facing camera and thus can be especially useful for users with hand motor impairments since it is portable, easy to install, and requires no specialized equipment. Despite its great potential as an assistive technology, most existing HBP research has focused on desktop/stationary settings. Few studies have included participants with motor impairments [91, 109], fewer have investigated HBP on mobile devices [2, 69], and none I was aware of have explored both. To fill this research gap, Google employed me to conduct the first HBP Fitts' Law study on smartphones with users with motor impairments and quantitatively studied its practicality with throughput (Fitts' index of performance [32]). Through these standardized outputs, one may compare any pointing methodology with the results and claim its practicality for people with motor impairments.

The research team asked me to identify three major requirements for the design of a mobile head tracker based on my lifelong experience using HBP. It should 1) be easy to use and require no cumbersome settings or calibration before or during use; 2) support precise pointing to match the requirements expected by typical user interface designs, and 3) be configurable and adaptive to personal needs. The research then proposes a low-cost, device-independent, calibration-free pointing technique based on head tracking using the front-facing camera on mobile devices. It uses neural-network-based face detection to provide pixel-level precision and allows for multiple selection methods, including dwell time, smile, and blink. To further ensure its customizability and truly enable personalization at scale, the Android implementation was open-sourced[1].

The user studies of the mobile HBP not only show its effectiveness compared with the state-of-the-art Eva Facial Mouse (EFM) [69] but also shed light on future directions for mobile HBP for assistive technology in general. The design of the studies followed previous practice [90] of Fitts' Law studies on mobile devices. We conducted two user studies with the research team to evaluate Fitts' Law performance across users with and without motor impairments (see Figure 4.1). The first was a between-subjects user study with 42 participants without motor impairments, to compare the proposed technique with an existing product (Eva Facial Mouse) and provide a baseline to understand the performance difference across user groups. The second study was performed

[1]`https://github.com/muratcancicek/mobile_head_based_pointing`

with 15 participants with motor impairments. It revealed a number of invaluable lessons and edge cases. Three participants in the second study participated remotely in uncontrolled environments, demonstrating the method's robustness to various device settings and lighting conditions. The experimental results show that the mobile HBP can work across environments and devices for users with different degrees of motor impairments, and the throughput of the proposed technique compares favorably with previous work [24, 64, 69] under similar conditions. Therefore, the research recommends HBP for users with motor impairments as an alternative interaction method on mobile devices.

The contributions in this chapter are three-fold:

1. I designed and refined a calibration-free mobile head-based pointing mechanism based on the in-depth lessons learned from motor impairment experience;

2. We conducted the first Fitts' Law study on participants with motor impairments on small-screen mobile devices, performed a comprehensive evaluation and analyzed with quantitative results, compared the proposed method with the state of the art [69] and suggest a set of guidelines for future accessibility studies of mobile HBP;

3. Google open-sourced the proposed HBP[2] framework to allow for personalized mobile HBP at scale.

[2]Implementation at `https://github.com/muratcancicek/mobile_head_based_pointing`

4.1 Designing a Mobile Head-based Pointing System

This section describes the design principle for the mobile HBP. As a person with motor impairments, I have more than 15 years of personal experience using similar technologies. This background provided inspiration for the following list of design requirements.

4.1.1 Motivation

I, a person with motor impairments, have an immediate need for a head-tracking mechanism that provides pointing functionality on mobile. While I have benefited from HBP in stationary desktop settings for years, the number of available solutions on mobile is limited. With a few exceptions, existing mobile HBP solutions either lack an objective evaluation [2, 69] or are not publicly available [2, 88, 91] for further development. I both properly evaluated and open-sourced its implementation in the previous chapter. However, head-tracking in that approach relied on specialized hardware. Here, I develop an HBP technique that can function on a wider range of devices that feature a front-facing camera, such as the vast majority of smartphones.

4.1.2 Personal User Experience

Despite living with significant motor impairments, I have pursued a career requiring heavy computer usage. This has led to over ten years of experience with head-based pointing on a number of computing devices and, thus, insight into the essential design requirements for an HBP method. All of my work on this dissertation, including the implementation of the methods, designing the pointing tasks, and writing the papers, was done through the use of HBP methods. Ultimately, this work presents an interaction method for people with motor impairments designed and developed by a person with motor impairments.

4.1.3 Base Requirements for Mobile Head-based Pointing

Based on personal experiences with head-based pointing, I summarize a list of requirements that I consider critical for HBP on mobile devices with small screens:

- Hardware-free: the proposed mobile HBP should exploit existing built-in sensors and be standalone. Requiring additional non-standard assistive hardware would undermine its availability and usability. As such, I employ a completely vision-based method that relies on the standard front-facing cameras available on today's smartphones.
- Customizable: I understand that users' physical abilities vary across individuals. A usable HBP should be able to adapt to basic personal needs. Therefore, the proposed HBP is designed to provide a set of selection options, including dwell, blink, and smile, to provide a degree of customizability to the user's ability level.
- Calibration-free: calibration is generally intrusive to user experience and perhaps even tedious and time-consuming if frequent re-calibration is required. To improve in-situ usability and mitigate the dependency on external supervision from a person without motor impairments, the proposed HBP is designed to be

calibration-free.

- Precise: on-screen visual targets can be very close, particularly on mobile devices, sometimes making target selection challenging even for people without motor impairments. To address this, our HBP must be able to provide precise pointing while requiring minimal physical effort by the user.
- Available and extendable: besides making the mobile device accessible, the solution itself also should be available and extendable by third parties for other potential users. To meet these two requirements, we believe the solution must be open-sourced.

These requirements help me to refine the existing technology on mobile and carefully choose the implementation tools to build on.

To avoid biasing by my personal experience alone, the research team at Google assisted me in conducting multi-person evaluations of our HBP mechanism, which will be discussed in the following sections.

4.2 Implementing Mobile Head-based Pointing

According to the above requirements, we developed an HBP algorithm for mobile devices. Our system first detects the head and key facial information from the front-facing camera on a smartphone. It then utilizes the coordinate of the nose tip and maps its movement onto the on-screen cursor location.

4.2.1 Platform and Libraries

I implemented the proposed HBP algorithm on Android so it can be available for a large population of potential users in both the developed and developing world. I used the Flutter (flutter.dev) UI toolkit for the user interface and ML Kit for Firebase to track facial features using the front-facing camera on mobile devices. ML Kit for Firebase is a mobile SDK with a set of ready-to-use machine-learning APIs. Its face detection API allows me to detect faces in an image, identify key facial features, and retrieve the detected face contours. It also provides high-level facial information, such as eye openness and smiling, which I use to implement the selection methods.

Given these off-the-shelf toolkits, my technical challenge becomes two-fold:

1. To map the movement of the identified facial features into the movement of an on-screen pointer; and
2. To implement selection methods that translate facial gestures or lack of head movement into a click on the screen.

4.2.2 Mapping Head Movement to On-Screen Cursor Location

This section describes my mapping function from the physical head movement to the on-screen pixel coordinate. There are two types of descriptors: one is with respect to the screen coordinate, such as the pixel location of eye corners; the other is with respect to the camera coordinate, such as the estimated head pose given by the vision tracking algorithm. The proposed HBP opts for the former, as it is unaffected by head pose estimation error and the camera intrinsic parameters across devices.

4.2.2.1 Mapping onto Cursor Relative Change v.s. Absolute Location

As I aim to achieve calibration-free HBP, I map the head movement onto the relative change of cursor location, rather than the absolute location. This is because, in my experience, the one-to-one mapping from head location to cursor absolute location is hard to use. In order to point at a certain on-screen location, the user has to make a specific head pose w.r.t. the camera. This inevitably requires head calibration in practice. In case of pose change from either the user or the device, a re-calibration is needed. Such a design violates the base design requirements defined in Section 4.1.3. Instead, my mapping function incrementally changes the cursor location according to head movements. In this case, the ways to point the cursor to an exact location are non-deterministic, meaning that users have the flexibility to start from an arbitrary head pose, move their heads in a natural fashion, and gradually approach the intentional target.

4.2.2.2 Clipping to the Edge of the Screen

To provide truly calibration-free interaction, the proposed HBP introduces a clipping mechanism. It clips the cursor location when the cursor reaches the screen edge, and the user continues moving in the same direction. Figure 4.2 shows an example in which, through some occurrence, the cursor is located at the center of the screen when the user's nose is pointed $15°$ off center. At this point (a) the user may likely feel the

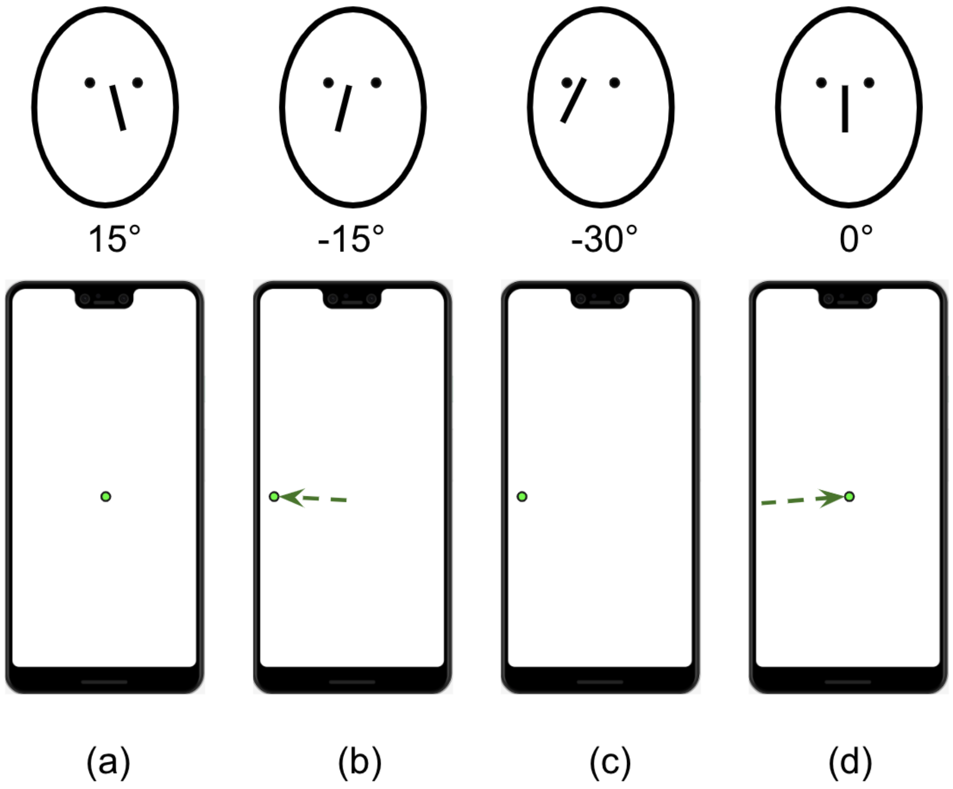


Figure 4.2: A demonstration of clipping the cursor at the screen edge when head movements overshoot. This mechanism allows for the intuitive adjustment of the head-to-cursor mapping.

need to realign the face angle with the cursor location (e.g. 0° head yaw onto screen center). To accomplish this, the user can turn left (-15°) until the cursor touches the screen edge and (c) keep turning along the original movement (-30°). Once the cursor reaches the left edge, the proposed HBP will discard the leftward control signal from head movements. I refer to this phase as overshoot. When the user (d) starts to move back toward the screen (0°), the cursor will start to follow from its position on the screen edge. By enabling this head movement overshoot, I believe users will be able to adjust the head-to-cursor mapping on the fly easily. I later found this to be true in the user studies.

4.2.2.3 Pointing by Nose

A key choice in my algorithm is how to convert the extracted facial features into a two-dimensional space to guide the on-screen pointer. The main challenge is detection reliability in practice; as more components are involved, the more intrusive detection noise becomes. To alleviate this issue, I exploit just one facial feature: the nose tip. Specifically, I define an input velocity function V_{input} by the nose tip pixel coordinates n_t on the captured image in the frame t as follows:

$$V_{input}(n_t) = S_{input}(n_t) - S_{input}(n_{t-1}), \tag{4.1}$$

The smoothing function on the input S_{input} is defined as the mean filter with a constant window length s_{input} where

$$S_{input}(n_t) = \frac{1}{s_{input}} \sum_{k=0}^{s_{input}-1} n_{t-k} \tag{4.2}$$

Empirically, I found that $s_{input} = 3$ gives a good consistency between head and cursor when the system runs at around 40 fps. The input velocity $v_{input} = V_{input}(n_t)$ presents the input change as the user changes head position. A mapping function ϕ translates v_{input} to the output velocity v_{output} as $v_{output} = \phi(v_{input})$ where

$$\phi(v_{input}) = gainFactor * (\frac{v_{input}}{R_{image}} * R_{screen}), \tag{4.3}$$

R_{image} is the resolution of the input image in units of physical pixels, while R_{screen} is the resolution of the mobile screen in units of logical pixels. A gain factor, empirically set to $gainFactor = (6, 8)$, transforms the head motion to the cursor motion linearly, and it yields consistent user perception across different device sizes. The output velocity v_{output} in this formulation can be too sensitive to input changes since the scaling factors also amplify the high-frequency noise while reducing the physical head movement. Therefore, I introduce a motion threshold function $\Lambda(v_{output})$. It ignores the subtle noise and returns values only greater than the predefined thresholds, i.e., $\Lambda(v_x, v_y) = (\lambda(v_x, m_x), \lambda(v_y, m_y))$ where

$$\lambda(v, m) = \begin{cases} 0, & \text{if } |v| < m \\ v, & \text{otherwise} \end{cases}, \tag{4.4}$$

I observed that $m_x = 5$ and $m_y = 5$ resulted in a good trade-off between the physical efforts to start driving the cursor and to maintain the head pose to keep the cursor stationary. I apply thresholds on each axis independently to allow the user to travel linearly along one axis while excluding the noise from the other axis. Applying Λ gives me an intermediate on-screen cursor location as c'_t where $c'_t = c_{t-1} - \Lambda(v_{output})$, where c_{t-1} denotes the cursor location in the last frame.

In my design, as is common in other approaches, the cursor is required to stay within the screen when the user input would otherwise push the cursor beyond the edge. I confine the pointing space within the screen by encapsulating the proposed location with a boundary check function β as $\beta(c'_t)$. This function is used to achieve the re-center procedure described above. In addition to smoothing the input, the final pointing coordinates c_t also include additional smoothing on $\beta(c'_t)$ to improve stability, and it is defined as

$$c_t = S_{output}(\beta(c'_t)) \tag{4.5}$$

where S_{output} has the same behavior with S_{input} in the equation 4.2 and applies a mean filter with the same window length as $s_o = 3$. In conclusion, the end-to-end head to cursor mapping function Φ that calculates the final pointing coordinates c_t from the given nose tip coordinates n_t is defined as:

$$\Phi(n_t) = S_{output}(\beta(\Lambda(gainFactor * (\frac{V_{input}(n_t)}{R_{image}} * R_{screen})))) \tag{4.6}$$

Notice that Φ is designed to be device-independent by introducing scaling into logical pixels and tends to be calibration-free by only relying on relative input change.

However, it also includes a set of constants $\{s_{input}, gainFactor, (m_x, m_y), s_{output}\}$ which are predefined based on our initial exploration and kept fixed through the study. This, on the other hand, offers flexibility for those who want to customize the mapping function Φ. It is straightforward to fine-tune these parameters in an optional calibration according to personal needs.

4.2.2.4 Selection Methods

To provide a full mechanism that completes point-and-select tasks, I introduce multiple selection methods alongside the HBP implementation. Dwell-based selection is preferable when the available input channels are limited since dwelling depends on users' behavior with the pointing mechanism as a selection is fired when the pointer is kept within a constrained region (dwelling circle) for a certain time period. I found that a dwelling diameter of $d = 20$ logical pixels and a dwelling time of $p = 0.8$ seconds work well with my algorithm. I also implemented smiling and blinking as visual selection methods and also included them in the user study.

4.3 Evaluation Method

To evaluate the pointing performance of my implementation objectively, I employ a Fitts' Law [32] pointing task since it is the standard way to evaluate pointing methods and derive the dependent measure of throughput (Fitts' index of performance) as part of the comparison and evaluation [89]. This section describes the common elements of the evaluation across the two experiments. Details where the procedure or apparatus varied will be discussed in the next section.

4.3.1 Multi-directional Corner Task

The standard multi-directional pointing task (MD) described in the ISO 9241-9 standard [49] employs a circular arrangement of targets that does not fully utilize the long rectangular shape of most mobile device screens. Using this arrangement is possible, but the range of possible indices of difficulty would not be representative of real-world mobile device tasks because the longer dimension of the device is not explored. Roig-Maimó et al. [90] noted this challenge and developed a new Multi-Directional Corner task which takes better advantage of the space available on a mobile screen while maintaining some consistency with the standard task. I replicate this method in the studies, though with a wider range of indices of difficulty.

The Multi-directional Corner Task [90] consists of four "subspaces," each of which starts with a target in one corner of the device screen, then requires the user to choose one of three targets on an arc a fixed radius from the initial target, and then return to select the initial target. This results in six pointing tasks per subspace. After completing one subspace, the user must select the initial target for the next subspace and then targets within that subspace using the same approach. This gives a total of 24 pointing tasks per "block." Note that only targets selected within a subspace are used for evaluation, and movements between targets in different subspaces were not included in the analysis. Please refer to the Figure 4.3.

For most trials, users completed three blocks at each difficulty level. I reduced

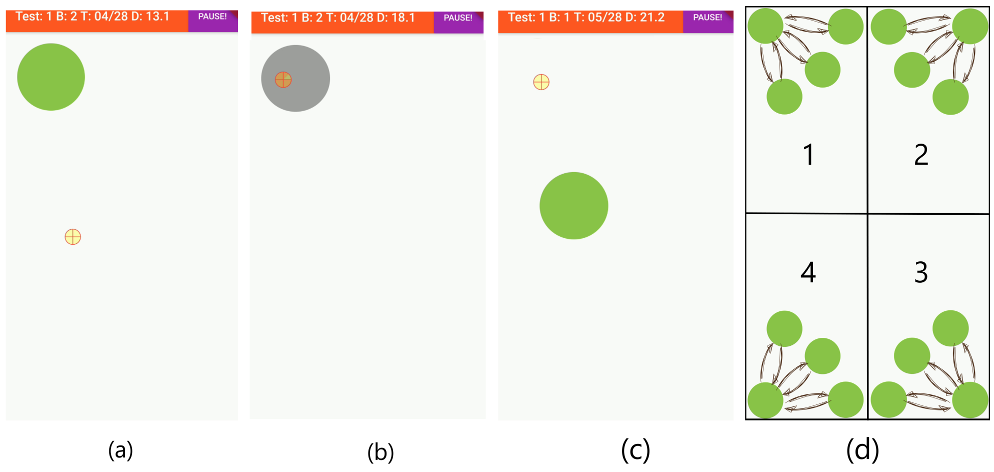

Test: 1 B: 2 T: 04/28 D: 13.1
PAUSE!
Test: 1 B: 2 T: 04/28 D: 18.1
PAUSE!
Test: 1 B: 1 T: 05/28 D: 21.2
PAUSE!
1
2
4
3
(a)
(b)
(c)
(d)

Figure 4.3: Illustration of the Multi-Directional Task. Amplitude = 125 dp, Target Width = 60, Index of Difficulty = 1.65 bits, corresponding to Test 1 in our experiment design. (a) The target appears on a blank screen with the current test info above. The pointer is an orange crosshair. (b) In dwell selection mode, hovering over the target turns it gray, and the crosshair fills, indicating the time until selection. (c) The next target appears immediately after selection. (d) Shows all targets for one block of a test, requiring 24 consecutive pointing tasks across 4 subspaces. The movements from the last target in one subspace to the first target in the next are not recorded.

the number of blocks in some conditions to address fatigue or time constraints, as discussed later in the chapter. The vast majority of trials with the implemented software, and all of the trials with Eva Facial Mouse, made use of the dwelling selection method to select targets. Some trials conducted with the implemented software made use of my blinking and smiling selection techniques, but often just to get users' impressions of the techniques rather than conduct a formal evaluation.

4.3.2 Apparatus and Procedure

All of the experiments were conducted on Android devices, with the vast majority of subjects using a Google Pixel 2 device with a 5-inch display with 1920x1080 resolution (441 ppi). The phone was placed on a mount, which was stuck to the table immediately in front of the subject's chair at a distance of two feet. The only variation from this was for the small number of remote participants with motor impairments, who each used their personal Android device and placed it as was comfortable for their environment. I implemented a custom Head Pointing Test App, which implemented the Multi-directional Corner Task [90] mentioned above and was installed on the devices in advance of the experiment. The entire testing procedure took place within the app.

Participants were recruited through the recruitment service of Turkey. Every participant signed a consent form before beginning the study and also provided responses to a pre-study questionnaire, which collected details such as previous experiences with related technology, whether they wear glasses and basic demographic information. Afterward, participants were briefed about the goal of the study, were shown the running software, and were allowed to play with a practice mode of the testing application for as long as they desired. Most users practiced for less than one minute. During the practice, my colleagues instructed participants on the self-calibration technique and asked them to find a comfortable position so that they could point at all corners of the screen. We also informed them that they were free to take breaks or stop at any point.

Once they were ready to begin, we started the first block of the study. After each block, the user had the opportunity to rest, and then either the subject or the experimenter could press an on-screen button to start the next block. In the able-bodied experiments, subjects generally continued to the next block on their own, whereas the experimenter often helped the subjects in the motor impairment experiments.

Following the study, participants were asked to fill out a post study questionnaire, which allowed participants to comment on the interaction methods and any fatigue they may have been feeling. For most experiments, participants needed 30 minutes to complete the study.

4.3.3 Throughput Calculation

For each sequence of trials that the subject completed consecutive pointing tasks, I calculated throughput (TP) as follows:

$$TP = \frac{\mathit{Effective\ index\ of\ difficulty}}{\mathit{Movement\ time}} = \frac{ID_e}{MT}, \tag{4.7}$$

where MT is averaged movement time per trial and ID_e is derived as:

$$ID_e = \log_2(\frac{A_e}{W_e} + 1), \tag{4.8}$$

where A_e is the mean of the actual movement amplitudes and W_e is the effective target width, calculated as $W_e = 4.133 \cdot SD_x$, where SD_x is the standard deviation in the selection coordinates as well-defined by MacKenzie [62]. The units of TP are "bits per second (bps)" since ID_e and MT have units of "bits" and "seconds" respectively.

Note that I used the standard-deviation method to calculate throughput [63] following the same strategy as Cuaresma and MacKenzie [23]. This is because my specific design utilizes the dwelling function and has no *error rate* [108]. One may find the details of the throughput calculation in the previous work [23, 62, 90, 108].

4.4 Experiments

We conducted two evaluation experiments of our head-based pointing system with my colleagues at Google.

1. **Comparison with the state-of-the-art.** We compared my algorithm against Eva Facial Mouse [69], a freely available head-based pointing method for Android, to show statistically how the state-of-the-art HBP performs on mobile and how comparable our performance is against this. Able-bodied participants were recruited for this comprehensive Fitts' Law study, where numerous participants completed various pointing tasks, as we inferred this study could not be so practical with participants with motor impairments.

2. **Exploration with participants with motor impairments.** We evaluated my algorithm with participants with motor impairments in several settings. We generally used the same procedure for these experiments as for the comparison study, though it was necessary to make modifications to suit the abilities of some participants. In this study, our goal was not so much to understand the Fitts' Law performance characteristics but to understand users' varied experiences.

4.4.1 Comparison with Eva Facial Mouse

Our first evaluation compared my technique to Eva Facial Mouse (EFM) [69], which. is a refined piece of commercial software with a lot of bells and whistles (e.g., a drag and drop mode, long tap, etc.) that is freely available for Android. We used a between-subjects design in this experiment, with each participant experiencing just one of the two pointing methods. The main reason for this choice was time, as it is much easier at the company to recruit for a 30-minute study than a 60-minute study, and we found 30 minutes to be the minimum time needed to evaluate a single technique. Learning effects and fatigue were other concerns. Participants reported minor issues with fatigue in our study, such as dry eyes, but none dropped out. We suspect fatigue would have become a greater concern, and some participants would have dropped out with a 60-minute study, which would have complicated our analysis.

4.4.1.1 Participants

Forty-two able-bodied participants (16 females) were recruited from Google using the company's user study recruitment service. The ages of participants ranged from 18 to 52, with a mean of 31.79 years (SD = 7.14). It was a prerequisite of participation that participants did not identify as a person with motor impairments and were not suffering from any injuries that might impact their ability to use a head-based pointer (e.g., a neck injury). Participants were compensated the equivalent of $15 US in internal corporate credits.

4.4.1.2 Procedure

All studies were conducted in the interior conference rooms of the company. Target widths and distances are listed in Table 4.1 for all trials. 20 participants used my pointing method, and 22 used Eva Facial Mouse. In both cases, participants were told that they were testing pre-release software and not informed of the actual provenance of the software until after completing the post-study questionnaire.

Besides the pointing software, two additional differences existed between the study conditions. Due to a mistake in software configuration, the smaller target width differed: 30dp in the Eva Facial Mouse condition and 15dp in the condition with the implemented software. The discrepancy was noticed too late to correct it without throwing away the work of nearly all participants in one of the conditions. Throughput calculations take into account differences in target width, so I believe the results can still be taken up with confidence. Participants using the implemented software completed

Table 4.1: Fitts' Law Performance of Participants Comparison

			A	**W**	**ID**	**MT**	**MT**	**TP**	**TP**
	Test	**SM**	**(dp)**	**(dp)**	**(bit)**	**(s)**	**(stdev)**	**(bps)**	**(stdev)**
EFM	**1**	D	125	60	1.62	**2.96**	0.22	**0.62**	0.10
	2	D	535	60	3.31	**3.57**	0.52	0.88	0.13
	3	D	125	30	2.37	3.05	0.20	**0.76**	0.08
	4	D	535	30	4.24	**3.69**	0.35	**1.06**	0.13
				Grand Mean		**3.32**	0.28	0.83	0.09
HBP	**1**	D	125	60	1.62	**1.89**	0.30	**0.84**	0.12
	2	D	535	60	3.31	**3.06***	0.37	0.88*	0.11
	3	D	125	15	3.22	3.02	0.40	**0.92**	0.13
	4	D	535	15	5.20	**4.57**	0.68	**0.95**	0.15
				Grand Mean		**3.14**	0.34	0.90	0.10
	5	B	535	60	3.31	3.43*	0.80	0.80*	0.20
	6	S	535	60	3.31	3.09*	0.58	0.83	0.14

3 Selection Methods (SM): Dwelling (D), Blinking (B), and Smiling (S). In the cells, the asterisks (*) indicate that the difference between those values and the corresponding values in Test 2 was statistically significant ($p < 0.05$) within the HBP design. Also, the corresponding Bold values in two designs indicate their differences from each other were significant.

two extra tests, one each using the blinking and smiling selection methods in order to collect feedback on these other options. These tests consisted of one block of tasks at a single index of difficulty. While I calculated throughput for these tasks for completeness, these values should be considered with caution. All other tests in both conditions used the dwelling selection method.

4.4.1.3 Results

I report the Fitts' Law performance of participants in Table 4.1 for each test in addition to a grand mean of participants calculated from participants' individual performances. The grand means for movement time and throughput were 3.20 s and 0.87 bps, respectively, for my method. The standard deviation of throughput at 0.05 across all dwelling trials indicates my method performs consistently across different Index of Difficulties without requiring calibration.

In the comparison of the two methods, I observe that the movement times and throughput values for the participants who used my method were strongly correlated to the test's Index of Difficulty (ID), while the participants who used Eva Facial Mouse seem to demonstrate non-linear performance as difficulty increases. Figure 4.4 shows that the two methods reach the same efficiency around the ID of 4 while HBP outperforms at lower IDs, whose values are actually more common in mobile environments. I ran a two-tailed t-test with type 2 (between subjects) to evaluate whether there was a difference between the conditions. In Tests 1 & 4, the easiest and hardest tests in both experiment designs, I measured that the performance differences between HBP and

EFM were statistically significant (p ¡ 0.05) in terms of the moment times and throughput values. However, the difference in throughput values in Test 2 and the difference in movement times in Test 3 were not statistically significant. There was no significant difference in overall throughput, but there was a significant difference in overall movement time (p ¡ 0.05). The insignificance in the other tests and overall throughput could be the result of close performances around the intermediate IDs, as HBP and EFM seem to behave similarly around ID = 4 bits, while HBP may be superior at the edge cases.

Tests 2, 5, and 6 (Table 4.1) in HBP design, all at the same index of difficulty, can be compared to get a sense of the differences between selection methods. Performance seems remarkably similar across these tasks, though they performed slightly better with dwelling and achieved a throughput of 0.88 bps, while they only achieved 0.80 bps with blinking and 0.83 bps with smiling. The differences between dwelling and the other two methods respectively were statistically significant (p ¡ 0.05 in both cases), although the performance difference between the blinking and smiling methods was not statistically significant. The p values were calculated based on two-tailed t-tests with type 1. More trials would need to be conducted with the alternate selection methods to reach a conclusive result.

4.4.2 Exploration with participants with motor impairments

After conducting our comparison study, we were interested to see how my method might fare with the target user population. We found that reaching such participants was challenging, and thus, we carried out two sub-experiments with two different sub-populations:

A field study at Ability Now Bay Area, a non-governmental organization that offers adults with developmental and physical disabilities a variety of programs, including education, wellness, and community integration.

A remote study with participants with motor impairments recruited by the user study recruitment service of the company. These participants installed my software on their own phones and participated in a video conference while they completed the pointing tasks.

4.4.2.1 Participants

Sixteen participants (8 females) were recruited in total from two different sources. Ages ranged from 25 to 65, with a mean of 38.2 years (SD = 16.05). There were no requirements for prior experience to participate in the study except for identifying as a person with motor impairments. Field study participants were compensated the equivalent of \$100 US in gift cards; remote participants were compensated the equivalent of \$75 also in gift cards. The difference in compensation is due to the shorter anticipated duration of the remote study.

We recruited 13 field study participants. Due to their severe physical limitations, 5 participants were not able to complete any tasks, while 8 participants completed at least one of the given tasks. In all cases, the participants were able to provide valuable feedback on the usefulness of the technique.

The three remote participants were much more capable on average than the field study participants. Each had a different type of motor impairment. Each was able to install the software on their own device and complete all trials.

4.4.2.2 Procedure

The procedure generally followed that for the comparison study, though I had to make modifications for the setting and participants' ability levels.

The remote participants' experience was the closest to the comparison study. The order of trials, blocks, and the indices of difficulty can be found in Table 4.2. The other difference with these trials is that we could not ensure consistent placement of the device compared to the comparison and field study experiments. Participants were told not to hold the device and asked to place it on a solid surface at which it was comfortable to view the full screen.

We originally planned for a much more comprehensive set of trials during the field study, but found that our plan was too ambitious given the ability level of most of the field study participants. To address this challenge, we modified our plan in two phases:

- *Phase 1:* We found that participants could only complete a few tasks, and we were uncertain of the validity of a Fitts' Law study. We revised our goal to simply have participants complete trials at one, and at most two, difficulty levels.

- *Phase 2:* After some experience with the first phase design, we revised our design to have fewer iterations of the same task and more task variety.

Table 4.2: Fitts' Law Performance of Participants with Motor Impairments Within-Test and Within-Experiment Design

					A	W	ID	MT	TP
	Test	**SM**	**P**	**Bs**	**(dp)**	**(dp)**	**(bit)**	**(s)**	**(bps)**
FP1	**1**	D	4	3	250	60	2.37	5.34	0.44
	2	D	1	3	250	40	2.86	2.81	0.94
FP2	**1**	D	5	2	550	40	3.88	7.92	0.78
	2	B	2	2	550	40	3.88	3.16	0.95
	3	D	2	2	550	20	4.88	3.81	1.09
R	**1**	D	3	3	125	60	1.62	2.38	0.80
	2	D	3	3	535	60	3.31	2.91	1.15
	3	B	3	1	535	60	3.31	4.53	0.78
	4	D	3	3	125	30	2.37	2.45	0.95
	5	D	3	3	535	30	4.24	3.81	1.00
	6	S	3	1	535	60	3.31	2.26	1.11

Study includes three experiment designs as Field Phase 1 (FP1), Field Phase 2 (FP2), and Remote (R). These designs include three Selection Methods (SM): Dwelling (D), Blinking (B), and Smiling (S). Participation (P) and number of blocks (Bs) in each test also vary.

Table 4.2 shows the details of each design, such as the block counts for each test and Fitts' Law parameters. As its *Participants* column states, participation in the tests was not equal since some participants had to quit the experiment after completing just a few blocks. Two participants in phase one and three participants in phase two quit the experiment without completing any blocks.

We were particularly careful during the field study to ensure that participants were aware that they could rest or quit at any time. We monitored participants for frustration and fatigue, checked in on them as necessary, and reminded them of their options if we felt it was warranted.

4.4.2.3 Results

Based on overall ability, we split the 16 participants with motor impairments into 3 subgroups.

1. *The group with mild motor impairments* includes 6 participants who have fine head control ability and were able to complete the given tasks in similar amounts of time. They reported that their physical impairments limit or completely block their interaction with smartphones via standard methods but did not affect their head-based interaction.

2. *The group with moderate motor impairments* includes 5 participants that were able to complete at least one block of the given tasks. But, the completion times were inconsistent among the participants and we observed that their overall pointing

performance was low compared to the first group.

3. *The group with severe motor impairments* includes the remaining 5 participants who were unable to complete any block of the given tasks due to ability-related reasons while they spent approximately the same amount of time in the sessions with the other participants.

We choose to report the means based on participants' overall performances calculated across the tests they individually completed. We believe this is still informative, given that a participant's overall performance suggests how head-based pointing likely works for that individual. Table 4.3 shows that the grand mean for throughput was 0.61 bps. however, we noticed that the participants with mild impairments performed noticeably better than the group with moderate impairments and thus reported the mean for each group separately as 0.96 bps and 0.20 bps.

Participation in the tests that explored the blinking and smiling selection methods was low (Table 4.2) and only included the participants in the group with mild impairments. Analyzing the results of Tests 2, 3, and 6 in the Design for Remote study would be a fair comparison for the selection methods as they introduced the same Index of Difficulty and were performed by the same 3 participants. The participants appeared to perform slightly better with smiling than as they completed the trails slightly faster, 2.26 s and 2.91 s respectively. Blinking appears to be the slowest method with an average movement time of 4.53 s. However, the differences were not statistically significant.

4.5 Discussion

I designed and evaluated head-based pointing (HBP) on smartphones for people with motor impairments. For the evaluation, With the help of my colleagues, we conducted two separate user experiments that included participants with and without motor impairments; the first also included a comparison with Eva Facial Mouse (EFM).

4.5.1 Evaluation Against the State of the Art

In the comparison experiment with participants without motor impairments, the grand means for movement time and throughput were 3.14 s and 0.90 bps, respectively, for the proposed HBP, while they were 3.32 s and 0.83 bps For EFM (Table 4.1). These values in Figure 4.4 indicate that the proposed HBP approach slightly outperforms EFM at the IDs lower than 4 bit (target widths ¿ 30 dp). Considering Android's Material Design principles that recommend touch target width should be at least 48 dp, HBP has superiority in the most common point tasks on mobile. For reference, HBP also outperforms FittsFace [23], another head-based pointing implementation, and reports a movement time of 7.14 s and throughput of 0.47 bps. However, my measurements stayed short compared to the original Multi-Directional Corner Task Experiment [90] where the grand means for movement time and throughput per trial were 1.41 s and 1.54 bps, respectively. However, they utilized screen touch as a selection method, which was not practical for my target group as their fine hand control was so limited.

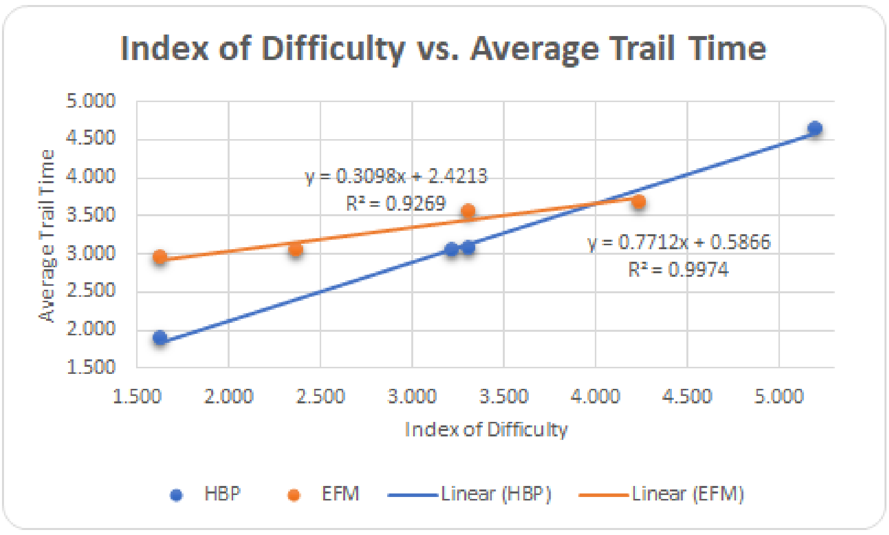


Figure 4.4: Comparison of Average Trail Durations (Target-to-Target Movement Time) for HBP (n=20) and EFM (n=22) at different Indices of Difficulty.

4.5.1.1 Takeaways

HBP's superiority over EFM is likely due to EFM's earlier development and my focus on evaluating pointing performance. I recommend my algorithm as a reference point to build from for future researchers and developers of HBP techniques. However, I recommend EFM to everyday users with an immediate need as it is a much more polished product than my current software.

4.5.2 Evaluation with Target User Group

In the second experiment with participants with motor impairments, the grand mean for throughput was 0.61 bps (Table 4.3), though the results also show that the performance of participants with different levels of impairments can vary substantially.

Table 4.3: Fitts' Law Performance of Participants with Motor Impairments.

		MT (s)	**MT (stdev)**	**TP (bps)**	**TP (stdev)**
Physical	**Mild**	3.29	0.73	0.96	0.19
Impairment	**Moderate**	12.86	8.28	0.20	0.09
Group	**Severe**	N/A	N/A	N/A	N/A
Experiment	**Field Phase 1**	6.30	3.33	0.53	0.31
Design	**Field Phase 2**	6.47	5.17	0.87	0.27
	Remote	3.19	0.89	0.97	0.21
Grand Mean		7.70	7.22	0.61	0.42

In this case, I suggest narrowing the target user group when developing real applications with HBP and providing different functionalities for users from different ability groups. For example, one may provide multiple selection methods as optional to different groups. I also believe that developers tend to build their applications for the worst-case scenarios and target the group with severe impairments. However, this trend leaves out the intermediate groups out of the market. This group is not willing to be limited with very basic tools since they have greater abilities, but on the other hand they cannot fully utilize the off-the-shelf products due to their impairments. While this is just an observation and needs further investigation, this work showed that the intermediate user groups can complete complex pointing tasks at higher IDs through head-based pointing.

4.5.2.1 Limitations

During the evaluation with the target group, I noted several limitations of the study design that affected the overall results. I would like to share these limitations here as an observation and to guide future studies with similar settings.

Accessing and working with people with motor impairments had several limitations, including the physical ones like distance and time. In the field, we had to recruit all the candidates with motor impairments we could access based on self reports of capabilities such as head control. However, these reports were not well calibrated, and as a result, the group had a much wider range of physical abilities and more significant impairments than I expected. Although I tried to adjust my test design accordingly, 5 participants with severe impairments could not complete the tests even though they were able to use the system. Despite the difficulty, a few of these participants enjoyed using the system so much that they wanted and were allowed to continue using the system long after their scheduled time, even though they were only able to complete only a couple of the easiest trials. It seemed that these participants likely could have completed many tasks via HBP if I could have further reduced the index of difficulty or had additional technology to filter spasmodic movements while the participants were trying to dwell. But, in that case, the difficulty level would not be realistic for the participants with mild and moderate impairments.

I further observed that some participants with motor impairments had conditions that made head-based pointing difficult, such as a lack of stability in their chairs.

While most of the participants had fine control of their heads, their conditions would cause involuntary movements of their limbs, which affected their head position, and this dramatically affected their pointing performance in the Fitts' Law Study. This was especially true during the dwelling time when they were trying to stay still. Indeed, for some participants, it seemed that trying to stay still would make involuntary spasms more likely. Interestingly, one of the highest performing participants was nearly completely restrained in his chair with his arms strapped tightly to his body, and this seemed to greatly assist him in maintaining stability. Many other participants did not have this benefit, possibly because many seemed to be in loaner chairs because their main chair was being repaired.

4.5.2.2 Takeaways

In future work, the recruitment criteria for participants with motor impairments should be narrowed down to sub-groups based on ability levels, and separate test designs should be adapted for each impairment group. This helps to show the true benefit of HBP when appropriate pointing tasks are given to each group. Additionally, the potential external constraints of head-based pointing need to be well-considered. While I was able to gather statistical results from this Fitts' Law study, they do not necessarily show the effectiveness of HBP for the target group that has such limited options. HBP also needs to be evaluated as a communication tool for the same group by different measurements rather than Fitts' Law.

It was also interesting that few of the participants used mobile phones and most were very dependent on visiting the rehabilitation facility to be able to use a computer, which enabled them to communicate with friends and even conduct business. If I could make it possible for them to use a mobile device, then that could lead to a very big change in their independence and communication ability. Many were unaware that a free head-based pointing system (i.e., EFM [69]) was available for everyday usage, except for one participant who used a sophisticated head-based pointing system on a daily basis. I believe head-based pointing techniques would help many more users if they were made aware of their availability.

To further explore head-based pointing on mobile and improve its practicality for people with motor impairments, I was considering several possibilities for future work. First, it seems that more advanced HBP techniques are needed that will work for difficult cases and address issues such as the involuntary movements I mentioned above. Filtering out such movements might be possible using machine learning that detects spasms based either on visual input or pointer behavior. Another possibility might be to employ an interaction technique that does not require the on-screen pointer to be tightly connected with the head position at all times. For example, a "clutched" method that would somehow allow participants to disconnect the pointer from their head when a spasm occurs might be very helpful. Second, the small size of many targets on mobile user interfaces makes them difficult to use. If a system was able to understand the constructions of these user interfaces, then it might be possible to modify the user interface itself to be more usable, such as with larger buttons, fewer targets, etc. Such techniques would be beneficial not only for users with motor impairments but also for those with cognitive impairments and likely other conditions.

4.5.3 Meeting the Base Requirements

In Section 4.1.3, I stated four base requirements for a robust head-based pointing solution. Here, I discuss how my proposed HBP meets these requirements. It is a *hardware-free* solution as it does not require any external hardware. It employs a completely vision-based method that relies on the standard front-facing cameras available on today's smartphones. It is *customizable* with different selection methods and performs consistently. The first experiment supports this as the participants achieved a throughput of 0.88 bps with dwelling, 0.80 with blinking, and 0.83 with smiling at the same ID (3.31 bits). It is *calibration-free* and performs consistently across different Index of Difficulties and different user groups without requiring an initial calibration per condition. I found the performance of HBP stayed stable while the Index of Difficulties (IDs) increased within the range we tested. Movement time per trial linearly increased, as shown in Figure 4.4, while the standard deviation of throughput was 0.05 across all IDs. I furthermore observed this behavior with HBP in both experiments with participants with and without impairment (see Tables 4.1 and 4.2). It stays *precise* even beyond the limits of Android's Material Design principles, which recommend that touch target width should be at least 48 dp. My HBP method achieved a throughput of 0.95 bps (above the grand mean) in an extreme setting with the ID of 5.20 bits where the target width was 15 dp. As Google open-sourced my implementation, it became *available* and *extendable.*

Chapter 5

Data Collection For Personalized Head-Tracking Pointing

In the previous chapters, I proposed two different head-based pointing mechanisms and provided their evaluations with several user studies. These systems use a camera (e.g., embedded in a computer screen) to track the user's head motion using computer vision algorithms. Typically, measurements are taken in terms of a "face box" or of a specific facial figure (e.g., the nose tip [15]). These measurements are then mapped to the pointer location in the screen using a pre-defined algorithm.

A main drawback of this approach is that the mapping from head motion to pointer location is not necessarily representative of the user's intent. For example, moving one's head to the right may lead to a rightward motion of the pointer faster than the user intended. This may result in an overshoot, which must be corrected by a leftward head motion. In practice, the user must learn to use the system with head patterns that may not feel “natural”. While these algorithms typically afford some parameter tuning, the general mapping mechanism remains unchanged.

In this dissertation, I propose a user-centric approach to designing a pointing algorithm based on head tracking. Rather than imposing a pre-defined algorithm mapping head position to pointer position, I would like to learn a flexible mechanism that adapts to the user’s intent. With Prof. Manduchi’s greater efforts, we created a data set where the measured data (head position from video frames) is associated with the desired location of the pointer. To build such a data set, we resorted to the following strategy. We showed a well-visible marker (a white disk) moving on the screen in specific patterns. While watching the marker moving, participants were asked to move their heads “as if” they were controlling the marker themselves. A screen-embedded camera collected images and time-registered with the marker’s location on the screen each time. We believe that the videos thus recorded are representative examples of the way participants would move their heads if asked to move the cursor to replicate the same trajectories traversed by the pattern they saw moving on the screen.

This chapter describes the data collection strategy accomplished remotely due to COVID-19 social distancing constraints, which work[16] we already published with Prof. Manduchi. I also present examples of the dynamic of a specific facial feature (the nose tip) while participants were following different trajectories of the pattern in the screen and provide a simple analysis of the variance of the location measured for this feature across participants.

5.1 Data Collection

With Prof. Manduchi, we recruited 8 participants (3 female) from our university. One participant has a motor impairment due to cerebral palsy and is a regular user of head-based pointing technology. Although this is a relatively small sample size, it is adequate for a proof-of-concept.

This study aimed to collect videos of the participants as they moved their heads, following the path of a small white disk shown on their computer screen. The participants were instructed to pretend that they were controlling the white disk with their head motion. They were asked to not just follow the disk with their eye gaze, but by moving their head. No other instructions (e.g., how much to move their head, whether to rotate it vs. move it, etc.) were given. Hence, we can assume that the head motion of each participant was as "natural" and spontaneous as it could be.

I first generated a number of "trajectory videos" with a small white disk moving along a predetermined trajectory against a black background. Some of these trajectories were repeated at a slower velocity. Some trajectories included "pause" points, where the disk would stop for one second. Participants were able to see the future path of the disk (shown with dimmed brightness) so that they would know in advance where the disk would move next. Examples of disk trajectories are shown in Fig. 5.1. Note that in all trajectories, the disk started and ended at the center of the screen. I uploaded these trajectory videos (17 in total) on YouTube and created separate playlists for each participant, with the order of the video randomly permuted for each playlist.

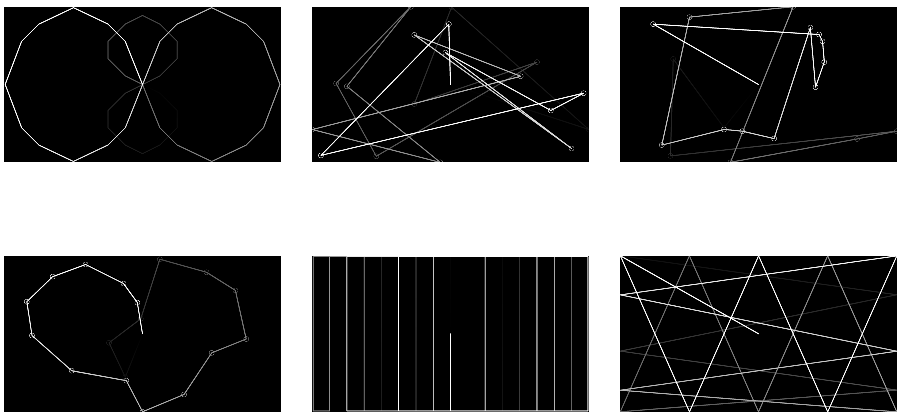


Figure 5.1: Samples of trajectory videos. The whole trajectory of the white disk is visible, with a lighter color indicating earlier locations in the trajectory. Small circles correspond to the location where the white disk stopped for one second.

The study was conducted remotely due to the social distancing requirements imposed by the COVID-19 pandemic. We utilized the Zoom platform to run the data collection sessions, including recording the participants' visual input during the pointing tasks.

For each participant, I scheduled a one-hour online meeting via Zoom. I collected information about the computer they would use for the test, whether they would use the embedded camera in the screen or an external camera, and the screen size and resolution. In the teleconference, Prof. Manduchi explained to each participant how the test would be conducted, then asked them to go to the YouTube site using the playlist assigned to them and to expand the browser window to full screen. In this way, participants would only interact with the moving disk in the trajectory videos, while images of their heads were taken by the camera and recorded in the cloud via Zoom.

Consecutive trajectory videos within the playlist were separated by 10-second intervals. Participants could use these time intervals to briefly rest and were allowed to pause the playlist between trajectory videos. An acoustic signal was played at the beginning and the end of each trajectory video in the playlist. This was used to synchronize the video displayed to the user with the video of the user recorded via Zoom ("user video"). These user videos were recorded at a resolution of 1280×720 pixels and at a rate of 25 frames per second. The whole session for each participant, as recorded by Zoom, was exported as a single video for simplicity. I then cropped individual user videos, using as a reference the acoustic signal recorded at the beginning and the end of each trajectory video. This way, I obtained pairs of synchronized trajectory-user videos for my analysis. 17 such video pairs were recorded for each user. I had to discard only 2 such video pairs, one due to noticeable latency caused by Zoom and one because I mistakenly interrupted the video. In total, I obtained 136 synchronized video pairs from 8 participants, with the length of the user videos varying between 536 and 2267 frames.

5.2 Head Motion Computation

One of the goals of this study is to explore whether the motion of the white disk on the screen could be predicted from the user video. For this purpose, I first extracted a number of visual "features" that can be used to describe the user's head's motion. These features can then be mapped, using suitable machine learning mechanisms, to the position of the disk on the screen.

A very simple, though perhaps not very informative, feature is the location

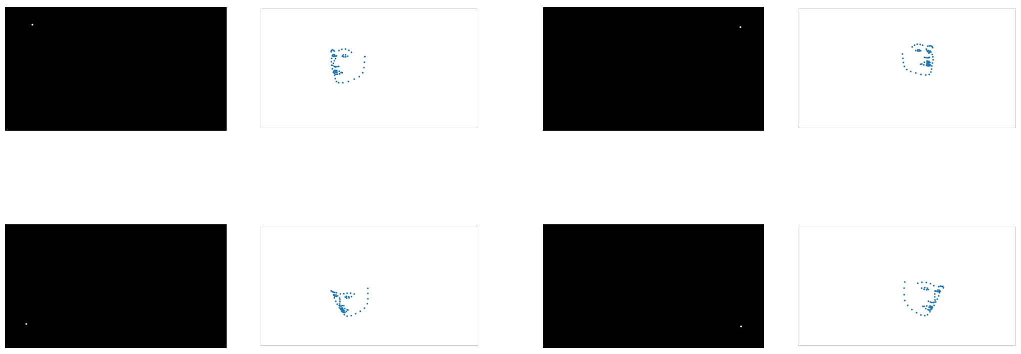


Figure 5.2: Facial landmarks produced by the PFLD algorithm [38] for one of the participants, taken at the time the white disk appeared in the location shown in the left half of the figure.

of the "face box", defined as a rectangle encompassing the whole face image [25, 58, 98]. A richer description can be obtained by identifying specific facial landmarks. I experimented with three state-of-the-art facial landmark detection models [38, 102, 105]. For example, Fig. 5.2 shows the location of the facial landmarks produced by the PFLD algorithm [38] for one of the participants at the times when the white disk being followed was situated in the vicinity of the four corners of the screen.

A higher-level feature that I will consider in the following chapters is the pose (3-D location + orientation) of the user's head, which can be computed using 3-D deformable models (e.g., [29, 92, 106]).

5.3 Trajectories Analysis

It is instructive to compare the trajectory of the tracked visual features against that of the white disk on the screen. This can provide some intuition about how a user would move their head in relation to the desired pointer location. In Fig. 5.3, I show the trajectory of a specific facial feature, the user's nose tip, for two participants (P2 and P6), viz-a-viz the trajectory of the white disk. Note that the nose tip location was successfully used for head-based pointing control in prior work [15]. While the trajectories of the nose tips may vaguely resemble the trajectory of the white disk on the screen, it is clear that precise one-to-one positional mapping would be hard, if not impossible.

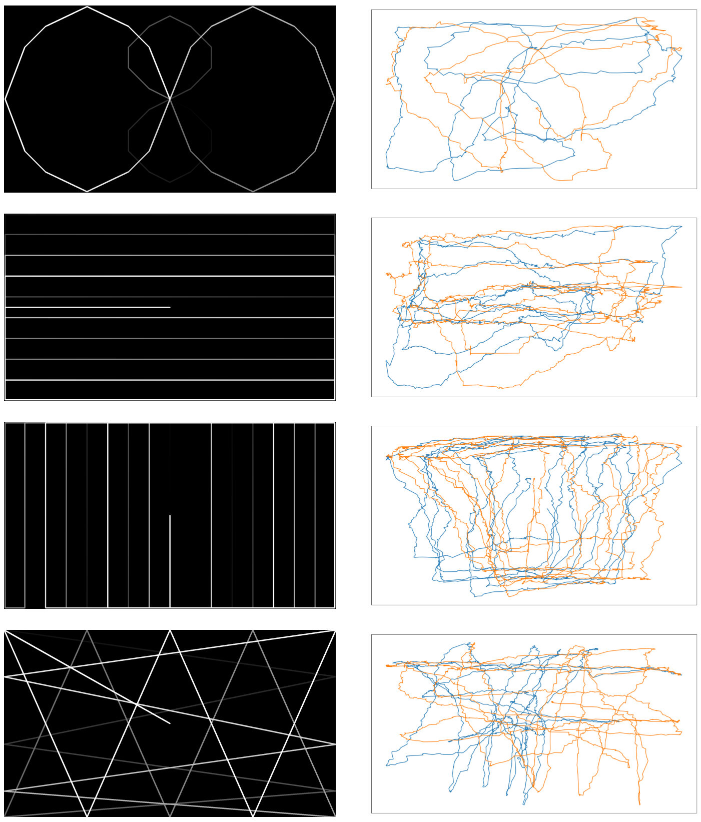


Figure 5.3: Trajectories of the nose tip features for two participants (P2 and P6) associated with the white disk trajectories shown in the left half of each row.

The trajectories of the nose tip feature shown in Fig. 5.3 are clearly different across the two considered participants. This is to be expected since the dynamic of head motion associated with tracking the white disk on the screen is completely subjective (remember that participants were not given instructions about how to move their heads). In some cases (see, e.g., the last case of Fig. 5.3), a positional bias is visible (possibly because the users positioned themselves at different locations in front of the camera). In these cases, the bias could be easily recovered and compensated for.

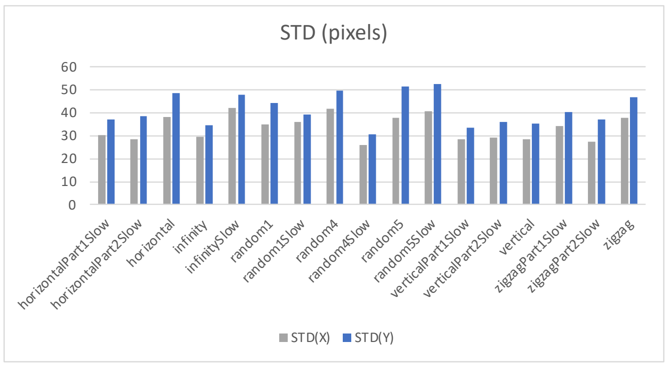


Figure 5.4: Average standard deviation of the X and Y coordinate of the nose tip across participants for each trajectory video.

To quantify the difference between trajectories across participants, I computed a measure of variance as follows. For each trajectory video, I measured, at each time, the variance in the X and Y coordinate of the nose tip location across all participants. (I excluded P5 in this analysis, as facial feature detection was unreliable for this participant.) Then, I computed the average of these variances over the whole trajectory. The squared root of the average variance (i.e., the standard deviation) for the X and Y coordinates of the nose tip are plotted for each trajectory video in Fig. 5.4. These values vary between 28 and 42 pixels for X and between 31 and 53 pixels for Y (remember that the recorded images have a resolution of 1280×720 pixels).

5.4 Conclusions

This chapter presents a unique data set collected for the purpose of understanding the different head motion dynamics adopted by different participants while imagining controlling a moving disk on a screen. We are currently using this data set to train a machine learning system that can predict the desired location of the cursor based on the user's head motion. Our hope is that, by learning from videos collected in response to a stimulus on the screen, this system can do a better job of mapping image features to cursor locations than current, hand-tailored algorithms.

Our initial analysis of the collected data shows that there is a fairly large variance in the location of facial features (e.g., the nose tip) across participants while following the same disk trajectory. This suggests that a certain degree of personalization may be necessary, in order to adjust the algorithm to the specific head dynamics of each user.

Chapter 6

Towards Personalized Head-Tracking Pointing

Chapter 5 explains how I created ”trajectory videos” showing a target moving on a screen, which participants watched while mimicking the motion with their heads as if controlling the target. The study aimed to capture natural head movements without the participants directly controlling the cursor. Participants were informed about the target’s future trajectory but received no other feedback. Their head movements, recorded by a camera, were proposed to indicate natural, active head motion patterns that could be used in designing a pointer control system where the cursor moves as the user intends.

This chapter analyzes the collected data in depth. I considered several types of features (different facial landmarks and geometric head pose). For each feature type, I computed a simple affine transformation (via least squares regression) mapping the

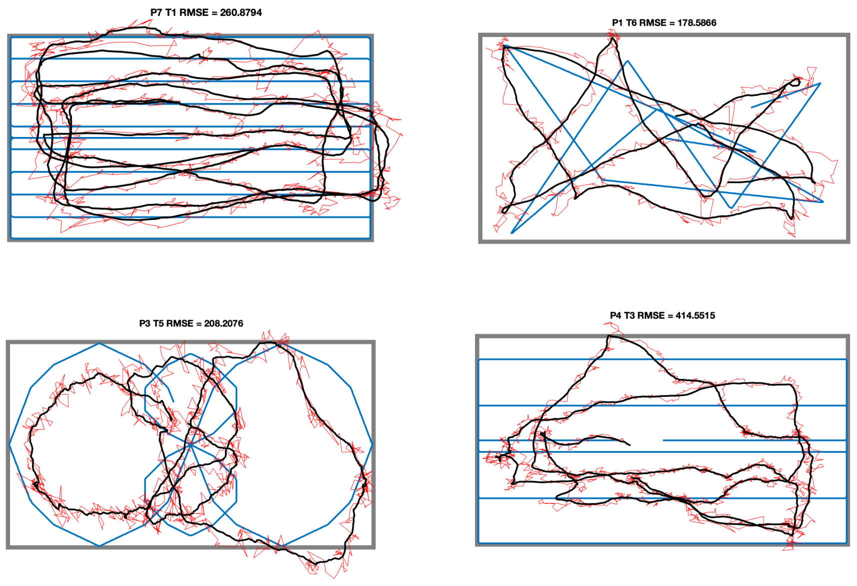


Figure 6.1: Examples of mapped trajectories (red), shown along with their smoothed version (black) and target trajectory (blue), for the `6_landmarks_xyz` feature. Each plot displays the participant ID (e.g., P7), trajectory ID (e.g., T1), and resulting RMSE in screen pixel units. The gray frame represents the screen viewport.

chosen features to pointer locations on the screen. I then compared the "mapped" pointer trajectories with the trajectories of the marker the participants were following with their heads. Ideally, the mapping would reflect the user's intent, and the two trajectories would coincide. In practice, I observed large discrepancies. This should not come as a surprise: the participants did not receive any feedback about whether their head movement would map to the desired movement of the pointer. In a real application, users rely on visual feedback from the mapped pointer motion on the screen to control their own head motion.

Still, the discrepancy between mapped and desired trajectories in this "feed-forward" system, which operates by predicting and acting without receiving real-time feedback, may reveal the extent to which the considered mapping enables "natural" head motions to accomplish desired pointer movements.

With Prof. Manduchi's heavy contributions, we already published The work[17] presented in this chapter. It is structured into three main tasks:

Task 1: Feature Selection. I measured the root square mean error (RSME) of trajectory discrepancy for two families of features: facial landmarks and geometric pose (3-D head rotation along with 3-D head location).

Task 2: Trajectory Bias. I computed the spatial distribution (over 25 regions on the screen) of bias and standard deviation of discrepancy (difference between mapped and target location). Bias reveals consistent errors that could potentially be mitigated by an appropriate mapping design. Standard deviation is an inverse measure of *consistency*: whether or not a participant moved his/her head in the same way when the target trajectory was going through the same region of the screen.

Task 3: Orientation. Professor Manduchi measured the angular discrepancy between the velocity of the mapped and target trajectories. This analysis is especially insightful for head-pointing systems based on velocity mapping. Such mapping also enables simple resting position reset strategies and is detailed in Section 2. In addition, to account for angular discrepancies that can result from the activation of different muscle groups when moving one's head in different directions, we computed the distribution of angular bias and standard deviation over the 8 octants of the plane.

6.1 Method

6.1.1 Video Data Set

As described in Section 5, this data set contains videos of 9 participants (3 female, 6 male) taken by a screen camera as they moved their heads while imagining following a moving dot visible on the screen (*target trajectory*). All participants except for P9 had no mobility impairment. P9, who has cerebral palsy, has limited control of his limbs but regularly uses a head-pointing system (Enable Viacam) as an interface device.

Participants in the study were shown 17 short target trajectory videos with a small target (a white disk) moving along a predetermined trajectory against a black background. They were able to see the future path of the target (shown with dimmed brightness) so that they would know in advance where the target would move next. Participants were instructed to move their head while watching each video "as if" they were controlling the target with their head motion. No other instructions (e.g., how much to move their head, whether to rotate it vs. displace it, etc.) were given. Some of the target trajectories in the videos were repeated at a slower velocity. Some trajectories included "pause" points, where the target would stop for one second. Examples of trajectories are shown in Fig. 6.1.

Target trajectory videos were shown on a screen with a resolution of 1920 by 1080 pixels. Videos of the participants were taken at 1280 by 720 pixels resolution and at a frame rate of 25 Hz.

Feature type	**RMSE (pixels)**
nose_tip_xy	333.24
nose_tip_xyz	332.40
6_landmarks_xyz	296.71
Euler_angs	324.88
Euler_angs_loc	307.66

Figure 6.2: The table shows the RMSE values (averaged over all participants) for the features considered.

6.1.2 Features

6.1.2.1 Facial Landmarks

I used mediaPipe [60] for face and hand landmark detection. I found it more practical during the implementation. Moreover, the MediaPipe framework detects 468 landmarks as opposed to the alternatives like OpenFace[6], which detects only 68 landmarks. MediaPipe uses weak perspective (scaled orthography) to compute each landmark's (x, y) coordinates. In addition, it computes the relative depth (z coordinate) of each feature. The (x, y, z) coordinates are then rescaled by a common factor.

I consider the following features based on facial landmarks:

nose_tip_xy : This is formed by the (x, y) coordinates of MediaPipe landmark #4. The location of the nose tip in the image has been considered in prior systems described in the literature [15, 37] as well as in [16].

nose_tip_xyz : While the prior work cited above only used the 2-D nose tip location in the image, for this feature, I considered all three (x, y, z) coordinates of MediaPipe landmark #4. I aimed to verify whether the inclusion of depth (the z coordinated) could prove beneficial.

6_landmarks_xyz : This feature contains the (x, y, z) coordinates of 6 selected MediaPipe landmarks (#4, #61, #152, #159, #291, #386). These landmarks represent the nose tip, the left mouth corner, the chin tip, the left outer eye corner, the right mouth corner, and the right outer eye corner, respectively. I selected these landmarks as they form a minimal face shape with richer information than the nose tip. They implicitly indicate the head location and rotation, which are more effective in determining the orientation and focus of the head.

6.1.2.2 Head Pose Landmarks

Since one's head can be approximately modeled as a rigid object, its pose (location + orientation) can be considered as a feature. I computed the full pose of the participant's head at each frame using the Perspective-n-Point (PnP) algorithm [56], which aligns 3D head model points to 2D facial landmarks detected in the image, providing accurate pose estimation with minimal computational overhead. Note that this computation requires camera calibration, and the calibration accuracy may affect the quality of pose estimation. I defined the following features:

Euler_angs : This feature contains the three Euler angles representing the head rotation with respect to a fixed reference frame.

Euler_angs_loc : In addition to the three Euler angles, I included here the location, with respect to a fixed frame, of a frame attached to the head, forming a 6-dimensional feature vector.

6.1.3 Affine Mapping

Given a feature vector $\mathbf{f}(\mathbf{t})$ at time t, I map it to a pixel location $\mathbf{p}(t) = (p_x(t), p_y(t))$ using an affine transformation: $\mathbf{p} = A\mathbf{f}(t) + \mathbf{b}$. In this equation, $\mathbf{b}$ is a 2-D vector, while A is a matrix whose dimensions vary from 2×4 for `nose_tip_xy` to 2×12 for `6_landmarks_xyz`. The parameters A and $\mathbf{b}$ are computed via least squares regression. Specifically, given the target trajectory $\hat{\mathbf{p}}(t)$, I minimize the average squared residual $\|A\mathbf{f}(t) + \mathbf{b} - \hat{\mathbf{p}}(t)\|^2$.

Each feature type's parameters were computed individually for each participant (representing a sort of "personalization"). To minimize the risk of overfitting, I employed the leave-one-out policy: for each participant, when evaluating the mapping for a certain trajectory, the parameters were computed on the 16 remaining trajectories.

In addressing the jitter observed in the mapped locations, attributed to fluctuations in landmark localization, I employed an exponential smoothing technique. This method integrates the current location data with previously smoothed values to mitigate rapid changes, using the formula $\mathbf{p}_\text{S}(t) = (1-\alpha)\mathbf{p}_\text{S}(t-1) + \alpha\mathbf{p}(t)$, where α, the smoothing constant, was chosen as 0.1. This selection of α reflects a deliberate balance, prioritizing the stability of historical data over the volatility of new measurements, thereby ensuring a more consistent trajectory by dampening the effects of noise.

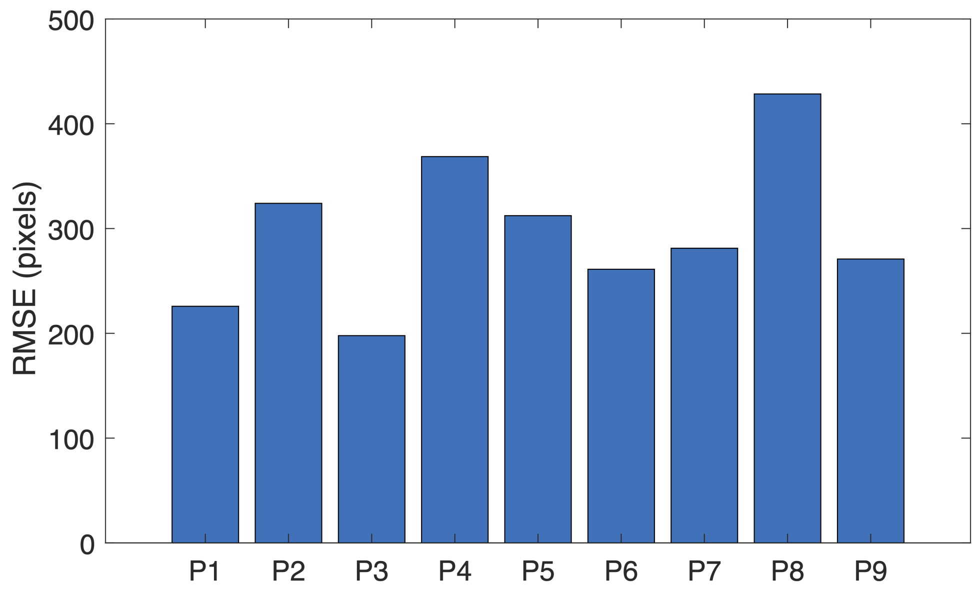


Figure 6.3: The bar graph at the top RMSE for all participants using feature 6_landmarks_xyz. The table at the bottom shows the RMSE values (averaged over all participants) for the features considered.

6.1.4 Quantitative Feature Comparison

For each feature and each participant, I considered each target trajectory in turn, computed the affine parameters based on the remaining trajectories, and computed the residual error $\mathbf{p}_S(t) - \hat{\mathbf{p}}(t)$. I then averaged the squared norm of this error over all trajectories and took the square root of the result, obtaining one RMSE value per feature and per participant.

At this point, Prof. Manduchi tested the null hypothesis that neither participant nor feature had an effect on the RMSE values. 2-way ANOVA rejected this null hypothesis and found a significant effect of both feature and participant (in this and other tests in this dissertation, statistical significance was set at $p \leq 0.05$). Tukey's multiple comparison test revealed a significant difference between the mean MSE obtained with 6_landmarks_xyz and that obtained with any other feature except for Euler_angs_loc. Fig. 6.2 shows the RMSE values for each feature, averaged over all participants. Based on this result, we conducted the rest of our analysis using the 6_landmarks_xyz feature, which produced the smallest average RMSE value (the distribution of RMSE across participants is shown in Fig. 6.3). Examples of mapped trajectories for different target trajectories and participants are shown in Fig. 6.1.

6.1.5 Spatial Distribution of Location Discrepancies

I divided the screen area into 5×5 regions uniformly and computed the bias (average) and the total standard deviation (square root of the trace of the covariance matrix) of the error $e(t) = \mathbf{p}_\mathrm{S}(t) - \hat{\mathbf{p}}(t)$. For each participant and each region, the error was averaged over all t for which any target trajectory was located in that region. Note that while the bias is a 2-D vector, the total standard deviation is a scalar. For each participant and each screen region, we thus have two coordinates of bias and one value of total standard deviation. The x (y) coordinate of the screen region was found to have a significant effect on the x (y) coordinate of bias. No significant effect of region location was found on the total standard deviation. A significant effect of participants was found on the total standard deviation only. Fig. 6.4, left, shows the distribution of location bias and total standard deviation across regions.

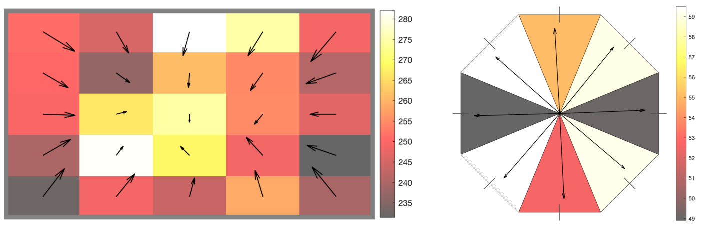


Figure 6.4: Top: The spatial distribution of location discrepancies. Location bias vectors are shown as arrows centered at each region. The region's color indicates the total standard deviation at a region. Units in the accompanying color bar are screen pixels. Bottom: The distribution of angular discrepancies across octants. The bisector of each octant is shown rotated by an amount equal to the angular bias in that octant. The color of the octant reflects the angular standard deviation in that octant. Units in the accompanying color bar are in degrees.

6.1.6 Angular Distribution of Velocity Discrepancies

In the previous section, I considered localization errors. Here, Prof. Manduchi looks at the angular discrepancy between the velocity of the target and that of the mapped trajectory, which is an important consideration for controllers based on velocity mapping. More precisely, he considered the angular difference at each time between the tangent to the mapped and the target trajectories distributed across octants. Specifically, for each octant ($[k \cdot 45° - 22.5°, k \cdot 45° + 22.5°]$, he considered the times t in which any target trajectory $\hat{\mathbf{p}}(t)$ had tangent with a slope that was within that octant. For these time intervals, he computed the angular difference between the slope of the tangent to the target trajectories and that to the mapped trajectories. He then computed the bias (mean) and standard deviation of these differences. This results in one value of angular bias and one value of angular standard deviation per participant and per octant. Neither participant nor octant were shown to have a significant effect on bias. Averaged across participants, the angular bias was positive for all octants (total average: 3.14°). This means that participants always move their heads in a direction slightly more counterclockwise compared to the direction of the target.

Both participant and octant affected angular standard deviation. Multiple comparison analyses did not reveal a significant pairwise difference in angular standard deviation across different octants. As an additional test, He averaged together the angular standard deviation values for even-ordered and odd-ordered octants. He found a significant effect of the octant group (even- or odd-ordered) on this statistic. Fig. 6.4, right, shows the distribution of angular bias and standard deviation across octants.

6.2 Discussion

6.2.1 Analysis of Results

My analysis (Fig. 6.2) compared multiple feature types to find which of these features could be mapped (through an affine transformation) to screen points forming a trajectory that best resembles the target trajectory. A set of 6 facial landmarks (6_landmarks_xyz) was shown to give significantly better results (in terms of RMSE) than just the nose tip or the vector of Euler rotation angles. This feature type can be computed robustly using modern software packages such as MediaPipe.

As shown in Fig. 6.4, analysis of the residual location bias (discrepancy between mapped and target trajectories) revealed a distinct spatial pattern. (Note that, although I only show results with 6_landmarks_xyz, a similar pattern was also observed when using other features.) This clearly indicates that the simple affine transformation used to map features to screen points may not be sufficient to reproduce the intended pointer motion. Different mapping models (e.g., polynomials) could reduce or eliminate these localized biases. Importantly, the total standard deviation of the error (discrepancy) was generally very large (in excess of 200 pixels, or about 10% of the screen width). This indicates that the participants were not consistent in their head motion in response to the same target trajectory; each participant had slightly different motions. While a careful mapping function could potentially remove localized bias, this will not help with poor consistency.

Spatial inconsistency could be due to multiple reasons. For example, users may not be able to exactly replicate a certain head pose due to proprioceptive bias [4, 5, 99]. It is also conceivable that trying to reach a certain head position/orientation starting from different points may result in slightly different trajectories of head movement. If, for example, one is trying to move the pointer to the upper left corner of the screen, different neck muscle groups would be activated depending on whether the pointer is currently in the lower left area (extension), in the top right area (axial rotation), or the center of the screen (motion around an oblique axis). Activation of different muscle groups may affect the velocity and precision of coordinated motion [85]. I conjecture that more complex sequence-to-sequence maps (e.g., recurrent neural networks) may help mitigate the lack of consistency by modeling different trajectories of head motion, thus producing a closer approximation to the intended pointer location at each time.

Prof. Manduchi's analysis of the distribution across octants of the angular error revealed that its standard deviation is larger for “diagonal” octants (Fig. 6.4, right). Moving the pointer along the “horizontal” or “vertical” octants requires moving one’s head left/right (rotation) or up/down (extension/flexion). In each of these movements, only one group of muscles is generally activated. However, for the “diagonal” octants (diagonal head circumduction), multiple neck muscles need to work in coordination, which may be the cause of the observed reduced consistency in these movements.

6.3 Conclusions

I presented an analysis of the discrepancy between the intended target trajectories and mapped trajectories using a simple affine transformation of selected features from images of participants’ heads. This analysis was based on an existing data set with videos of users moving their heads while following a moving target on the screen. Our analysis has shown that a set of 6 facial landmarks is superior, in terms of mapping error, to other commonly used features (nose tip, head rotation). I also reported mapping errors (in terms of bias and standard deviation) for both mapped locations and mapped velocities (angular error). In future work, I will consider more complex mapping using machine learning in an effort to ensure that the mapped trajectory faithfully reflects the user intent.

Chapter 7

Comparison of Personal, Generic, and Fine-tuned Head-Tracking Pointing

I analyzed the collected video data and explored that individuals have unique head movement patterns for similar pointing tasks, necessitating tailored mappings from head to pointer motion. I utilized affine transformation as the mapping function in the chapter 6. As the input for pointing, I relied on the feature sets, including facial landmark estimations and head pose calculation as described in Section 6.1.2. However, the correlation between the participant's head pointing and the desired pointer motion would be a conditional and possibly complex relationship rather than a straightforward proportional one. I also showed in Chapter 6 that the individuals have different biases based on the location and the direction of the pointing. Affine transformation cannot capture such biases as it only provides a straightforward proportional mapping. On the other hand, neural networks may also learn such non-linear correlations. They can learn several biases in the input patterns and generate a non-linear transformation by activating different neurons conditionally through the network. It would improve the overall pointing precision and may lead to training a generic model that can work for multiple users with little fine-tuning.

I evaluated fully connected neural networks (FCNs) and recurrent neural networks (RNNs) to transform input features to the desired pointing coordinates. Utilizing an FCN instead of affine transformation was straightforward, as the data formation was the same for both algorithms. I only have to batch data samples for FCN as it is trained better by iterating over small batches, while affine transformation is calculated once over the entire data set. One can also adjust the capacity of an FCN by choosing different hyperparameters for the number of neurons at each layer and the number of hidden layers. A wider FCN can capture local patterns and several biases in the input features. It may structure a nonlinear transform through its neurons that can be conditionally activated. This is also true for RNNs. However, different data formations are available for RNNs that allow me to learn sequence-to-sequence mapping. I can feed RNNs by sequences of input features from current and previous steps rather than considering input features at the current step alone. This introduces the utilization of previous steps in estimating the current pointing coordinates and allows the model to react to velocity or direction changes. I consider shaping the window of these input steps in different fashions, including step skipping and overlapping. I also explored the effects of stateful and stateless training of RNNs.

I started by extending the feature comparison study with FCNs and RNNs and trained individual models for each feature set. For the feature comparison study, I trained *personal* models for each participant separately. I split one participant's data into four approximately equal folds and applied cross-validation by leaving one fold out each time as the test set. This study also reports the performance of personal head-tracking pointing, where the models are trained for the specific participant with 75% of their existing data. I called such models *personal* as they were built for only one person. I reported the performance and discussed the practicality of such methods due to the necessity of data collection in significant amounts.

To compare with *personal* models, I aimed to learn a *generic* head-tracking pointing, tested on unseen participants not included in the training. I collected the data from 9 participants as described in Chapter 5. In this case, I trained the model with 8 participants and tested it with data from the 9th participant. I reported the results by 9-fold validation and highlighted the drawbacks of *generic* models. I also explored the effects of normalization with *generic* models by pointing out each individual has unique ranges for input features. I applied standard normalization for most experiments to bring the participants' input in the same range, as I only focused on end-to-end pointing.

Generic models underperformed compared to *personal* models, as expected. I then introduced *fine_tuned* models where I assumed the data from the target user exists but is limited. I examined the fine-tuning capacity of neural networks, which is specifically beneficial in this case. I compared their performance against the simple affine mapping with fine-tuning data alone and with the extended data containing the training and fine-tuning data. I had several options to fine-tune neural networks with new data, such as freezing the weights in the first layers and lowering the learning rate. I ran different experiments to investigate each option and only reported the ones fine-tuned with lower learning rates and no weight freezing.

7.1 Method

7.1.1 Video Data Set

Each participant in the data set was recorded through 17 distinct trajectories described in Chapter 5. For *personal* models, I applied 4-fold cross-validation and trained the models on approximately 75% of the participant data. I grouped 17 trajectories based on their length and ensured that the folds contained a similar number of frames in total. Each fold contains 6318, 6344, 6178, and 6689 frames, respectively. They consist of four complete trajectory videos except for the fourth one, which consists of five complete videos. Participant 4 missed a single video, and Participant 6 missed two videos due to errors in the recording process. However, those were some of the shortest videos, and all went into the fourth fold. In conclusion, I maintained the balance of the folds in size.

7.1.2 Features

7.1.2.1 Feature Sets

I used the future set already described in Section 6.1.2. To revisit each, `nose_tip_xy` and `nose_tip_xyz` represent the facial landmark estimated by MediaPipe[60].

The only difference is `nose_tip_xyz` also includes the landmark's Z coordinate estimation by MediaPipe. While the nose-tip is a strong feature for head-tracking pointing, `6_landmarks_xyz` includes six individual facial landmarks that represent different concerns of the face shape. Together, they represent a rigid shape in the future shape rather than a single point. On the other hand, `Euler_angs` and `Euler_angs_loc` was calculated via the Perspective-n-Point (PnP) algorithm [56], which aligns 3D head model points to 2D facial landmarks detected in the image, providing accurate pose estimation with minimal computational overhead. Note that this computation requires camera calibration, and the calibration accuracy may affect the quality of pose estimation.

7.1.2.2 Standard Normalization

I applied standard normalization to balance the impact of individual dimensions in the selected feature sets. Standard normalization is a preprocessing step used to rescale features to have the properties of a standard normal distribution with a mean of zero and a standard deviation of one. This is particularly useful when features have different units or scales, ensuring each feature contributes equally to the analysis.

The standard normalization formula is given by:

$$X' = \frac{X - \mu}{\sigma}$$

where:

- X is the original feature value,

- μ is the mean of the feature values,
- σ is the standard deviation of the feature values,
- X' is the normalized feature value.

By applying standard normalization, we ensure that the features are on a common scale, which can improve the performance of many machine learning algorithms, particularly those that rely on distance measurements such as k-nearest neighbors and support vector machines.

7.1.2.3 Mapping Types: Velocity Against Exact Coordinates

Pointing on a screen is the task of identifying the coordinates that the user intends to interact with. In the settings where the screen itself is interactable, like a touchscreen, pointing happens instantly and only at the moment of interaction. For example, physical touch on a touchscreen indicates the exact interaction point, and an additional cursor is not necessary. On the other hand, some settings require external pointing mechanisms since the screen is not interactable. Such mechanisms rely on a virtual cursor on the screen and relocate it based on momentary input change captured by an external mechanism. Head-tracking pointing is not so different; it updates the cursor location based on the input change on the camera. Relativity is preferred for these pointing mechanisms with static mapping function as discussed in Section 4.2.2.1 and the previous work [15]. Relative pointing also introduces edge clipping, allowing users to re-calibrate their position as explained in Section 4.2.2.2.

I collected a data set where participants were allowed to express head-pointing as naturally as possible. I only asked them to align their head with the center of the screen before each session. Also, no feedback was provided on their alignment with the target during the pointing sessions so they were not aware if they needed re-calibration during the session. Under these conditions, participants had no presumptions regarding the correlation between the target and their movements when following the target. They would assume an exact correlation where they should have a specific head position to point to a specific coordinate on the screen and revisit the exact head position every time the target visits the same coordinate. Alternatively, they would assume a relative correlation where they change their head position in correlation with the target movement. In this assumption, participants ignore the exact positions and only focus on maintaining their velocity with respect to the target. To investigate the dominant assumption, I evaluated two different transformations. One transformation was from `exact_values` of features to the exact target coordinates that represent one-to-one mapping and the other from `delta_values`, which were the feature change in 200 milliseconds (or between 5 frames) to the target relocation in the same period. When generating the final outputs, I still applied edge clipping described in Section 4.2.2.2 for the transformation with `delta_values`. I also limited the output space of the transformation with `exact_values` with the screen size, which was 1920X1080. I moved the estimations that fell out of the screen to the corresponding edges.

7.1.3 Transformations

I defined three transformation functions that map the input at time t to a pixel location $\mathbf{p}(t) = (p_x(t), p_y(t))$. The first two take a feature vector $\mathbf{f}(\mathbf{t})$ as the input. The last one takes a feature matrix consisting of consecutive feature vectors from a time interval.

For comparison, I included the affine transformation I defined in Section 6.1.3 in the experiments explained in this chapter. I also explored the potential of neural networks to learn tailored mappings from head to pointer motion. Since the affine transformation provided a strong benchmark, I tended to build simple architectures with a few hidden layers. I implemented a custom fully connected neural network (FCN) with three hidden layers, each consisting of 4096 neurons. I applied dropout with the rate of 20%. The learning rate was set to 0.0001, with a batch size of 16. The network was optimized using Adam [50].

Additionally, I implemented a custom recurrent neural network. I evaluated two of the popular architectures, namely, Long Short-Term Memory[42] (LSTM) and a network with Gated Recurrent Units[14] (GRU). I found that GRU trained faster than LSTM while reaching lower error rates. Since the difference between their performances was slight but consistent, I excluded LSTM from the study. I referred to the network with GRUs as the recurrent neural network to highlight its recurrent functionality, which was its key difference against the FCN and the affine transformation. The architecture started a recurrent layer with 4096 gated units, followed by a fully connected layer with 1024 neurons, and ended with an output layer that generates 2D estimations for pointing coordinates. I did not apply dropout for RNN. The learning rate was set to 0.0001, with a batch size of 16. The network was optimized using Adam [50]. I prepared the input windows with a length of 3 while skipping four frames between each step. I also benefited from overlapping with one step. Therefore, I did not utilize *stateful* where the windows must be consecutive as the hidden state kept across the windows until the actual sequence ends. Instead, I trained it in a stateless fashion so that I was able to generate a higher number of windows by overlapping. I could also randomize the order of windows at each epoch. I found the stateless training preferable as it improved the overall performance.

7.1.4 Personalization Models

I evaluated different personalization models for head-tracking pointing to investigate the amount of data required for a personalized mechanism. I introduced three main models as described below.

Personal Head-Tracking Pointing : I regressed or trained models specific to the participant. *Personal* models were optimized with 75% of the data available from that participant. I also proposed *personal_quick* models that were optimized with only 25% of the same data. In both cases, I ran 4-fold cross-validation to generalize the results.

Generic Head-Tracking Pointing : I proposed *generic* models that were tested on participants who were left out of the training set. The data set with nine participants allowed me to run a 9-fold cross-validation, leaving out each participant once.

Fine-tuned Head-Tracking Pointing Note that collecting data from each individual user would not be practical. My previous observations show that the end users' time investment in data collection would not be sufficient. Additionally, we cannot guarantee how strictly the user followed the instructions for data acquisition without supervision. Therefore, models trained solely on the data from the end user would be unreliable for pointing. Instead, a potential end product would employ the models that are pretrained on reliable data and only fine-tuned with the limited data from the end user. I also investigated this case by fine-tuning the *generic* models by 25% of the data available from the test participant to obtain *fine_tuned* models and evaluating their performance on the rest.

7.1.5 Experiment Space

This study has four independent variables that generate 120 results in total, as given below.

- 5 Feature Sets X 2 Mapping Types X 3 Transformations X 4 Personalization Models = 120 Separate results

As the validation folds varied between 4 and 9, 1080 individual experiments were conducted.

		Transformations		
Mapping Type	**Feature Set**	**RNN**	**FCN**	**Affine**
Exact Values	`nose_tip_xy`	203.71	229.67	223.42
	`nose_tip_xyz`	202.50	228.76	222.25
	`6_landmarks_xyz`	200.42	204.41	202.63
	`Euler_angs`	216.43	236.07	218.47
	`Euler_angs_loc`	202.89	219.56	207.85
Delta Values	`nose_tip_xy`	258.80	272.94	268.47
	`nose_tip_xyz`	257.95	270.24	267.35
	`6_landmarks_xyz`	250.91	263.43	259.72
	`Euler_angs`	292.54	325.53	295.07
	`Euler_angs_loc`	253.17	281.44	261.33

Table 7.1: The RMSE values measured for the *personal* pointing. The errors for the *delta_values* are computed over the reconstructed trajectory by summing together the delta values at each step.

7.2 Results

7.2.1 Quantitative Feature Comparison

For each result, I considered each target trajectory in turn, computed the residual error $\mathbf{p}_{\mathrm{S}}(t) - \hat{\mathbf{p}}(t)$. I then averaged the squared norm of this error over all test points and took the square root of the result, obtaining one RMSE value per result.

		Transformations		
Mapping Type	**Feature Set**	**RNN**	**FCN**	**Affine**
Exact Values	nose_tip_xy	202.33	286.51	335.07
	nose_tip_xyz	202.52	287.78	338.83
	6_landmarks_xyz	199.22	297.57	331.28
	Euler_angs	216.01	294.09	311.33
	Euler_angs_loc	200.06	284.58	310.53
Delta Values	nose_tip_xy	257.87	274.29	310.07
	nose_tip_xyz	257.71	276.58	309.63
	6_landmarks_xyz	253.27	282.79	306.99
	Euler_angs	296.51	287.11	325.16
	Euler_angs_loc	253.50	274.38	308.97

Table 7.2: The RMSE values measured for the *generic* pointing. The errors for the *delta_values* are computed over the reconstructed trajectory by summing together the delta values at each step.

For *personal* models, I confirmed my previous finding in section 6.1.4 that suggested a significant effect of both feature and participant (in this and other tests in this article, statistical significance was set at $p \leq 0.05$). The results are provided in Table 7.1. As Table 7.2 shows the details, I also tested the null hypothesis that neither participant nor feature affected the RMSE values for the *generic* models. Again, 2-way Analysis of Variance (ANOVA) rejected this null hypothesis and found a significant effect of both feature and participant. Lastly, I repeated the same test with the three different transformations and provided detailed results in 7.3, 7.4 and 7.5. 6_landmarks_xyz outperformed other feature sets in each transformation. However, I noticed that the performance difference between the feature sets gets quite lower with RNN transformation, except Euler_angs is still high compared to the remaining features. Based on this conclusion, I only investigated the results with 6_landmarks_xyz in the next sections.

7.2.2 Mapping Type Comparison

I ran every experiment with two different mapping types and provided the results in Table 7.3, 7.4 and 7.5. I also provided Fig. 7.1 that shows the effect of mapping types for each test participant. I applied the paired t-test to the null hypothesis that suggests there is no significant difference between the RMSE values of the (`exact_values` and the `delta_values`. Again, I repeated the same test with the three different transformations. The null hypothesis is rejected for each transformation. This indicates that the difference in RMSE between `exact_values` mappings and `delta_values` map-

		Models			
Mapping Type	**Feature Set**	**Personal Quick**	**Personal**	**Fine-tuned**	**Generic**
Exact Values	nose_tip_xy	203.71	200.66	198.88	202.33
	nose_tip_xyz	202.50	199.04	199.07	202.52
	6_landmarks_xyz	200.42	184.33	193.41	199.22
	Euler_angs	216.43	211.54	212.45	216.01
	Euler_angs_loc	202.89	196.16	195.57	200.06
Delta Values	nose_tip_xy	258.80	256.13	257.87	251.12
	nose_tip_xyz	257.95	255.25	257.71	250.93
	6_landmarks_xyz	250.91	247.28	253.27	246.20
	Euler_angs	292.54	291.04	296.51	291.69
	Euler_angs_loc	253.17	250.34	253.50	246.88

Table 7.3: The RMSE values measured for the pointing with RNN transformation

		Models			
Mapping Type	**Feature Set**	**Personal Quick**	**Personal**	**Fine-tuned**	**Generic**
Exact Values	nosetip_xy	235.00	229.67	280.11	286.51
	nose_tip_xyz	234.23	228.76	280.83	287.78
	6_landmarks_xyz	235.50	204.41	276.43	297.57
	Euler_angs	243.47	236.07	289.80	294.09
	Euler_angs_loc	229.51	219.56	275.66	284.58
Delta Values	nose_tip_xy	276.48	272.94	271.65	274.29
	nose_tip_xyz	278.37	270.24	271.40	276.58
	6_landmarks_xyz	280.16	263.43	281.03	282.79
	Euler_angs	312.98	325.53	299.74	287.11
	Euler_angs_loc	290.37	281.44	279.21	274.38

Table 7.4: The RMSE values measured for the pointing with FCN transformation

		Models			
Mapping Type	**Feature Set**	**Personal Quick**	**Personal**	**Fine-tuned**	**Generic**
Exact Values	nose_tip_xy	227.95	223.42	327.02	335.07
	nose_tip_xyz	227.07	222.25	327.61	338.83
	6_landmarks_xyz	220.49	202.63	296.11	331.28
	Euler_angs	225.39	218.47	303.65	311.33
	Euler_angs_loc	221.58	207.85	292.70	310.53
Delta Values	nose_tip_xy	270.67	268.47	310.07	308.77
	nose_tip_xyz	269.64	267.35	309.63	308.14
	6_landmarks_xyz	262.04	259.72	306.99	304.32
	Euler_angs	297.20	295.07	325.16	324.48
	Euler_angs_loc	263.89	261.33	308.97	303.53

Table 7.5: The RMSE values measured for the pointing with affine transformation

pings is statistically significant across all the settings. Based on these statistics, I only evaluated the experiments with `exact_values` mappings in the following sections.

7.2.3 Transformation Comparison

I compared three transformation models. Differences in model performance were statistically examined using one-way ANOVA. I concluded that there is a significant difference between the transformations. My exploration also delved into model personalization, as shown in Fig. 7.2. It shows the RNN's superior performance over the affine transformation, with RNN achieving the lowest RMSE values (184.3266 for personal and 199.2223 for generic models). While the FCN transformation's performance closely mirrored the affine transformation in personal models, it improved in generic models but did not surpass the RNN transformation.

7.2.4 Personalization Comparison

The *personal* models reached lower RMSE values across all transformations, highlighting the effectiveness of personalized models over *generic* ones, as visualized in Fig. 7.2. The ***personal_quick*** models, even with limited training data, demonstrated considerable efficacy, especially within the RNN transformation, suggesting the feasibility of quick personalization.

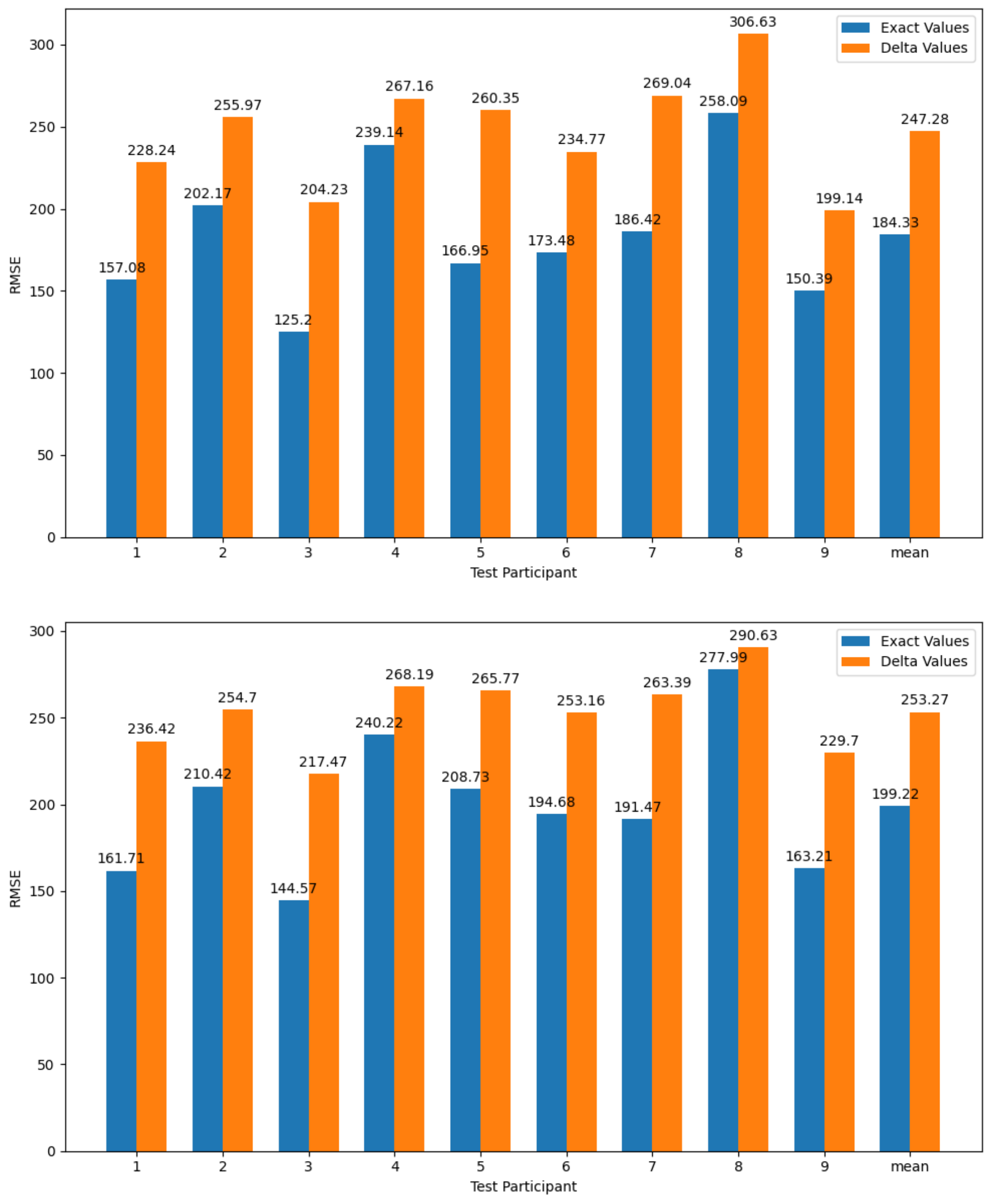


Figure 7.1: The RMSE (Root Mean Square Error) values for different mapping types (`exact_values` and `delta_values` across both *personal* (top) and *generic* (bottom) models

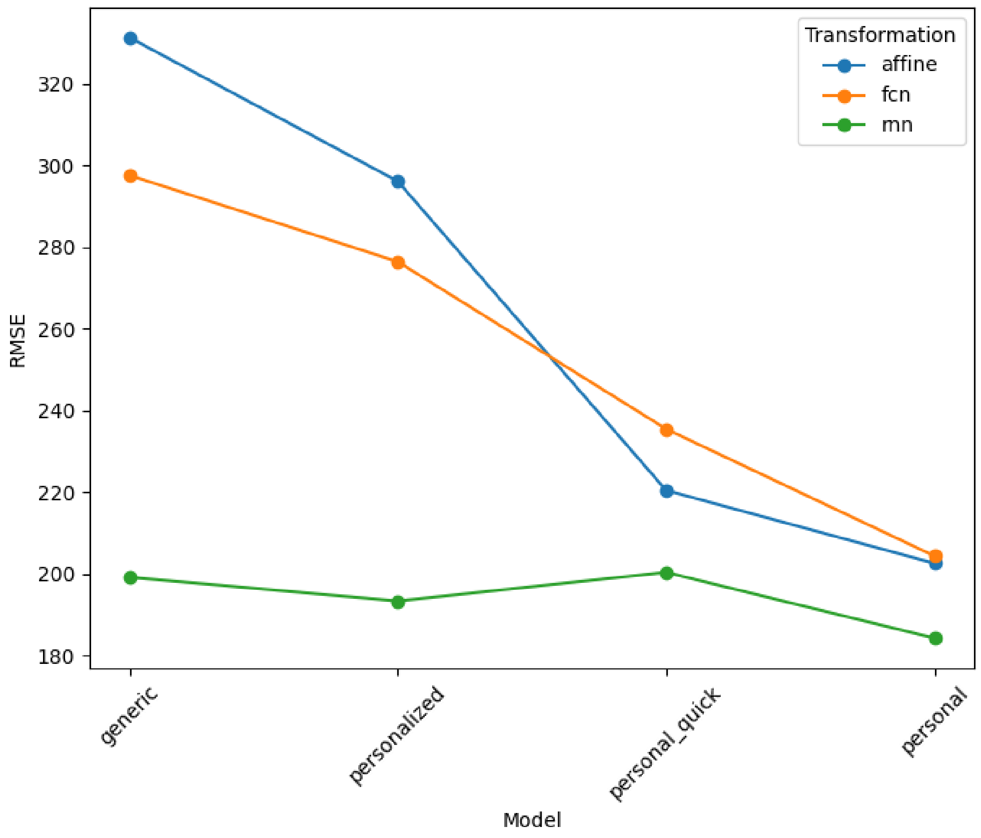


Figure 7.2: The RMSE (Root Mean Square Error) value by Transformation and Model for `exact_values`

7.3 Discussion

This study explored the efficacy of different transformation models and feature sets in improving head-tracking pointing accuracy. The results highlighted the superiority of the `6_landmarks_xyz` feature set across various transformations and mapping types, underpinning the importance of selecting appropriate features for head-tracking technologies.

7.3.1 Feature Set Efficacy

The `6_landmarks_xyz` feature set consistently outperformed other feature sets in terms of RMSE, suggesting its robustness in capturing essential facial landmarks for accurate head-tracking. This finding corroborates the initial hypothesis that a combination of landmarks representing a rigid face shape significantly enhances tracking accuracy. Future research must further investigate the potential of composite features in head-tracking applications, especially in scenarios requiring precise control.

7.3.2 Impact of Mapping Types on Accuracy

The comparison between `exact_values` and `delta_values` mapping types revealed a significant impact on pointing accuracy. `Exact_values` mappings led to better accuracy, suggesting that one-to-one head positions to screen coordinates are more effective for head-tracking pointing mechanisms. This insight could be instrumental in designing head-tracking interfaces, particularly in applications where precision is paramount.

7.3.3 Transformations

Among the transformations, RNN demonstrated the lowest RMSE, indicating its capability to effectively learn complex mappings between head movements and screen coordinates. This finding suggests the potential of RNNs in developing adaptive and efficient head-tracking systems. However, the FCN's performance closely mirrors the affine transformation, especially in *fine_tuned* models, indicating that simpler models can also achieve considerable accuracy with proper training and personalization.

7.3.4 Personalization and Model Efficiency

The study further emphasizes the significance of personalization in head-tracking technologies. Transformations with personalized models, including those optimized with limited data (*personal_quick* models), showed remarkable performance, highlighting the feasibility of developing efficient personalization strategies with minimal user data. This has profound implications for user-centric designs of head-tracking systems, where personalization can significantly enhance user experience and accuracy.

7.3.5 Limitations and Future Directions

While this study provides valuable insights, it also has limitations. The data set was relatively small, and experiments were conducted under controlled conditions. It is important to note that the data being analyzed was collected without any feedback to the participants, who were moving their heads as if controlling the target with their head motion. In practice, the user of a head-pointing system relies on visual feedback to make the necessary adjustments to the pointer location. For example, if a user rotates their head to the right to move the pointer to a checkbox and overshoots the desired location, the user would then slightly move their head to the left to compensate for it. Still, it is crucial that the controller should reflect the user's intention as much as possible to avoid the need for continuous adjustments, which can make the whole experience of computer interaction slow and frustrating.

A limitation of this data set is that the data only included "smooth" pointing tasks, with participants pretending to move a target with constant velocity along specific trajectories. This type of motion could be representative of tasks such as drawing or repositioning a window on the screen. A much more common interaction task is point-select [26], whereby the user moves the pointer from a certain location to reach a target and then selects it (normally, via a mouse click). The video trajectories considered are poorly representative of point-select tasks, and new data collection would be necessary to study head motion in these cases.

An obvious limitation of this data set is that all participants except for one (P9) had no mobility impairments. Since the intended users of head-pointing systems are people with mobility limitations, it will be important to acquire similar data for this community of users.

Future research should aim to validate my findings in more diverse and realistic settings, including different lighting conditions, user demographics, and applications. Additionally, exploring the integration of the proposed transformations and feature sets into real-world applications could validate their effectiveness and usability further.

Investigating user feedback on the usability and comfort of these head-tracking systems would also be beneficial.

7.3.6 Conclusion

In conclusion, my dissertation demonstrates the potential of using advanced transformation models and carefully selected feature sets to enhance the accuracy of head-tracking pointing systems. The findings underscore the importance of personalization and the selection of appropriate mapping types and feature sets in designing efficient and user-friendly head-tracking interfaces. As technology advances, these insights could pave the way for more intuitive and accessible human-computer interaction mechanisms.

Bibliography

[1] Mahdieh Abbaszadegan, Sohrab Yaghoubi, and I Scott MacKenzie. Trackmaze: A comparison of head-tracking, eye-tracking, and tilt as input methods for mobile games. In *International Conference on Human-Computer Interaction*, pages 393–405. Springer, 2018.

[2] Essential Accessibility. Essential accessibility, 2018. Retrieved August 6, 2018 from `https://www.essentialaccessibility.com/assistive-technology-for-android/`.

[3] Johnny Accot and Shumin Zhai. Beyond fitts' law: models for trajectory-based hci tasks. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 295–302. ACM, 1997.

[4] Bridget Armstrong, Peter McNair, and Denise Taylor. Head and neck position sense. *Sports medicine*, 38:101–117, 2008.

[5] Neil J Artz, Michael A Adams, and Patricia Dolan. Sensorimotor function of the cervical spine in healthy volunteers. *Clinical biomechanics*, 30(3):260–268, 2015.

[6] Tadas Baltrušaitis, Amir Zadeh, Yao Chong Lim, and Louis-Philippe Morency. Openface 2.0: Facial behavior analysis toolkit. In *2018 13th IEEE International Conference on Automatic Face & Gesture Recognition (FG 2018)*, Piscataway, NJ, United States, 2018. IEEE, Institute of Electrical and Electronics Engineers.

[7] Richard Bates and Howell O Istance. Why are eye mice unpopular? a detailed comparison of head and eye controlled assistive technology pointing devices. *Universal Access in the Information Society*, 2:280–290, 2003.

[8] Margrit Betke, James Gips, and Peter Fleming. The camera mouse: visual tracking of body features to provide computer access for people with severe disabilities. *IEEE Transactions on neural systems and Rehabilitation Engineering*, 10(1):1–10, 2002.

[9] Martin Bichsel and Alex Pentland. Automatic interpretation of human head movements. In *Proceedings of the 13th International Joint Conference on Artificial Intelligence, Workshop on Looking At People*, 1993.

[10] Doug Bowman, Ernst Kruijff, Joseph J LaViola Jr, and Ivan P Poupyrev. *3D User interfaces: theory and practice.* Addison Wesley, 2004.

[11] Pedro E Bravo, Miriam LeGare, Albert M Cook, and Susan Hussey. A study of the application of fitts' law to selected cerebral palsied adults. *Perceptual and motor skills*, 77(3_suppl):1107–1117, 1993.

[12] Stephen Brewster, Joanna Lumsden, Marek Bell, Malcolm Hall, and Stuart

Tasker. Multimodal 'eyes-free' interaction techniques for wearable devices. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 473–480. ACM, 2003.

[13] Stuart K Card, Thomas P Moran, and Allen Newell. *The psychology of human-computer interaction.* Hillsdale, NJ: Erlbaum, 1983.

[14] Kyunghyun Cho, Bart van Merrienboer, Çaglar Gülçehre, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using RNN encoder-decoder for statistical machine translation. *CoRR*, abs/1406.1078, 2014.

[15] Muratcan Cicek, Ankit Dave, Wenxin Feng, Michael Xuelin Huang, Julia Katherine Haines, and Jeffry Nichols. Designing and evaluating head-based pointing on smartphones for people with motor impairments. In *Proceedings of the 22nd International ACM SIGACCESS Conference on Computers and Accessibility*, pages 1–12, New York, NY, United States, 2020. Association for Computing Machinery.

[16] Muratcan Cicek and Roberto Manduchi. Learning a head-tracking pointing interface. In *International Conference on Computers Helping People with Special Needs*, pages 399–406, Cham, 2022. Springer, Springer International Publishing.

[17] Muratcan Cicek and Roberto Manduchi. Towards personalized head-tracking pointing. In *Extended Abstracts of the 2024 CHI Conference on Human Factors in Computing Systems*, CHI EA '24, New York, NY, USA, 2024. Association for Computing Machinery.

[18] Muratcan Cicek, Jinrong Xie, Qiaosong Wang, and Robinson Piramuthu. Mobile head tracking for ecommerce and beyond. *arXiv preprint arXiv:1812.07143*, 2018.

[19] Muratcan Cicek, Jinrong Xie, Qiaosong Wang, and Robinson Piramuthu. Mobile head tracking for ecommerce and beyond. In *IS&T International Symposium on Electronic Imaging 2020: Mobile Devices and Multimedia: Enabling Technologies, Algorithms, and Applications proceedings*, pages 303–1–303–11, San Francisco, CA, USA, 2020. Society for Imaging Science and Technology, Society for Imaging Science and Technology.

[20] Rory MS Clifford, Nikita Mae B Tuanquin, and Robert W Lindeman. Jedi force-extension: Telekinesis as a virtual reality interaction metaphor. In *3D User Interfaces (3DUI), 2017 IEEE Symposium on*, pages 239–240. IEEE, 2017.

[21] Albert M. Cook and Janice M. Polgar. *Assistive Technologies: Principles and Practice.* Elsevier Health Sciences, 2014.

[22] Origin Instruments Corporation. Headmouse nano, 2017. Retrieved January 31, 2022 from http://www.orin.com/access/headmouse/.

[23] Justin Cuaresma and I Scott MacKenzie. Fittsface: Exploring navigation and selection methods for facial tracking. In *Universal Access in Human–Computer Interaction. Designing Novel Interactions: 11th International Conference, UAHCI 2017, Held as Part of HCI International 2017, Vancouver, BC, Canada, July 9–*

14, 2017, Proceedings, Part II 11, pages 403–416, Cham, 2017. Springer, Springer International Publishing.

[24] Gamhewage C De Silva, Michael J Lyons, Shinjiro Kawato, and Nobuji Tetsutani. Human factors evaluation of a vision-based facial gesture interface. In *2003 Conference on Computer Vision and Pattern Recognition Workshop*, volume 5, pages 52–52. IEEE, 2003.

[25] Jiankang Deng, Jia Guo, Yuxiang Zhou, Jinke Yu, Irene Kotsia, and Stefanos Zafeiriou. Retinaface: Single-stage dense face localisation in the wild. *CoRR*, 2019.

[26] Sarah A Douglas, Arthur E Kirkpatrick, and I Scott MacKenzie. Testing pointing device performance and user assessment with the iso 9241, part 9 standard. In *Proceedings of the SIGCHI conference on Human Factors in Computing Systems*, pages 215–222, New York, NY, USA, 1999. Association for Computing Machinery.

[27] Tobii Dynavox. Microsoft & tobii dynavox, 2018. Retrieved July 16, 2018 from `https://www.tobiidynavox.com/en-US/landing-pages/td_and_microsoft/`.

[28] Electropages. 2024 electronics industry trends: Ai, evs & iot lead the way, 2024.

[29] Gabriele Fanelli, Juergen Gall, and Luc Van Gool. Real time head pose estimation with random regression forests. In *CVPR 2011*, pages 617–624. IEEE, 2011.

[30] Torsten Felzer and Stephan Rinderknecht. Clickeraid: a tool for efficient clicking using intentional muscle contractions. In *Proceedings of the 14th international*

ACM SIGACCESS conference on Computers and accessibility, pages 257–258. ACM, 2012.

[31] Leah Findlater, Karyn Moffatt, Jon E Froehlich, Meethu Malu, and Joan 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*, pages 6056–6061. ACM, 2017.

[32] Paul M Fitts. The information capacity of the human motor system in controlling the amplitude of movement. *Journal of experimental psychology*, 47(6):381, 1954.

[33] National Center for Health Statistics. Disability and functioning (noninstitutionalized adults aged 18 and over), 2017. Retrieved July 9, 2018 from `https://www.cdc.gov/nchs/fastats/disability.htm`.

[34] Yulia Gizatdinova, Oleg Špakov, and Veikko Surakka. Comparison of video-based pointing and selection techniques for hands-free text entry. In *Proceedings of the ACM international working conference on advanced visual interfaces*, pages 132–139. ACM, 2012.

[35] Glassouse. Glassouse assistive device, 2018. Retrieved July 17, 2018 from `http://glassouse.com/`.

[36] Monica Gori, Giulia Cappagli, Alessia Tonelli, Gabriel Baud-Bovy, and Sara Finocchietti. Towards assisting visually impaired individuals: A review on current assistive technologies. *Neuroscience & Biobehavioral Reviews*, 69:109–123, 2016.

[37] Dmitry O Gorodnichy and Gerhard Roth. Nouse ‘use your nose as a mouse’perceptual vision technology for hands-free games and interfaces. *Image and Vision Computing*, 22(12):931–942, 2004.

[38] Xiaojie Guo, Siyuan Li, Jiawan Zhang, Jiayi Ma, Lin Ma, Wei Liu, and Haibin Ling. PFLD: A practical facial landmark detector. *CoRR*, 2019.

[39] Bence Halmosi, Cecilia Sik-Lanyi, and Tibor Guzsvinecz. Technologies designed to assist individuals with cognitive impairments. *Sustainability*, 15(18):13490, 2023.

[40] John Paulin Hansen, Vijay Rajanna, I Scott MacKenzie, and Per Bækgaard. A fitts’ law study of click and dwell interaction by gaze, head and mouse with a head-mounted display. In *Proceedings of the Workshop on Communication by Gaze Interaction*, page 7. ACM, 2018.

[41] Salwa H. Henni, Sissel Maurud, Kristin Skeide Fuglerud, and Anne Moen. The experiences, needs and barriers of people with impairments related to usability and accessibility of digital health solutions, levels of involvement in the design process and strategies for participatory and universal design: a scoping review. *BMC Public Health*, 22(1):35, 2022.

[42] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural Computation*, 9(8):1735–1780, 1997.

[43] Qiong Huang, Ashok Veeraraghavan, and Ashutosh Sabharwal. Tabletgaze: un-

constrained appearance-based gaze estimation in mobile tablets. *arXiv preprint arXiv:1508.01244*, 2015.

[44] Apple Inc. Get ready for arkit 2, 2018. Retrieved July 17, 2018 from `https://developer.apple.com/arkit/`.

[45] Apple Inc. Truedepth camera, 2018. Retrieved July 17, 2018 from `https://www.apple.com/iphone-x/\#truedepth-camera`.

[46] Apple Inc. Use switch control to navigate your iphone, ipad, or ipod touch, 2018. Retrieved July 15, 2018 from `https://support.apple.com/en-us/ht201370`.

[47] Apple Inc. Move the pointer using head pointer on mac, 2024. Retrieved May 22, 2024 from `https://support.apple.com/en-us/guide/mac-help/mchlb2d4782b/mac`.

[48] Global Market Insights Inc. Consumer electronics market size - global market insights inc., 2021.

[49] ISO ISO. 9241-9 ergonomic requirements for office work with visual display terminals (vdts)-part 9: Requirements for non-keyboard input devices (fdis-final draft international standard), 2000. *International Organization for Standardization*, 2000.

[50] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[51] KPMG. The truth about online consumers. 2017 global online consumer report, Klynveld Peat Marwick Goerdeler, 2017.

[52] Kyle Krafka, Aditya Khosla, Petr Kellnhofer, Harini Kannan, Suchendra Bhandarkar, Wojciech Matusik, and Antonio Torralba. Eye tracking for everyone. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016.

[53] Andrew Kurauchi, Wenxin Feng, Ajjen Joshi, Carlos Morimoto, and Margrit Betke. Eyeswipe: Dwell-free text entry using gaze paths. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems*, pages 1952–1956. ACM, 2016.

[54] Mikko Kytö, Barrett Ens, Thammathip Piumsomboon, Gun A Lee, and Mark Billinghurst. Pinpointing: Precise head-and eye-based target selection for augmented reality. In *Proceedings of the 2018 ACM CHI Conference on Human Factors in Computing Systems*, page 81. ACM, 2018.

[55] Ioannis Lazarou, Stavros Nikolopoulos, Panagiotis C. Petrantonakis, Ioannis Kompatsiaris, and Magda Tsolaki. Eeg-based brain-computer interfaces for communication and rehabilitation of people with motor impairment: A novel approach of the 21st century. *Frontiers in Human Neuroscience*, 12:14, 2018.

[56] Vincent Lepetit, Francesc Moreno-Noguer, and Pascal Fua. Epnp: An accurate

o(n) solution to the pnp problem. *International Journal of Computer Vision*, 81(2):155, 2009.

[57] Brian Leung and Tom Chau. A multiple camera tongue switch for a child with severe spastic quadriplegic cerebral palsy. *Disability and Rehabilitation: Assistive Technology*, 5(1):58–68, 2010.

[58] Jian Li, Yabiao Wang, Changan Wang, Ying Tai, Jianjun Qian, Jian Yang, Chengjie Wang, Ji-Lin Li, and Feiyue Huang. DSFD: dual shot face detector. *CoRR*, 2018.

[59] Perceptive Devices LLC. Smylemouse, 2016. Retrieved January 31, 2022 from https://smylemouse.com/.

[60] Camillo Lugaresi, Jiuqiang Tang, Hadon Nash, Chris McClanahan, Esha Uboweja, Michael Hays, Fan Zhang, Chuo-Ling Chang, Ming Guang Yong, Juhyun Lee, et al. Mediapipe: A framework for building perception pipelines. *arXiv preprint arXiv:1906.08172*, abs/1906.08172, 2019.

[61] I. Scott MacKenzie. Fitts' law as a research and design tool in human-computer interaction. *Human-Computer Interaction*, 7(1):91–139, 1992.

[62] I Scott MacKenzie. Fitts' throughput and the remarkable case of touch-based target selection. In *International Conference on Human-Computer Interaction*, pages 238–249. Springer, 2015.

[63] I Scott MacKenzie. Fitts' law. *The Wiley Handbook of Human Computer Interaction*, 1:347–370, 2018.

[64] John Magee, Torsten Felzer, and I Scott MacKenzie. Camera mouse+clickeraid: Dwell vs. single-muscle click actuation in mouse-replacement interfaces. In *International Conference on Universal Access in Human-Computer Interaction*, pages 74–84. Springer, 2015.

[65] John J Magee, Samuel Epstein, Eric S Missimer, Christopher Kwan, and Margrit Betke. Adaptive mouse-replacement interface control functions for users with disabilities. In *International Conference on Universal Access in Human-Computer Interaction*, pages 332–341. Springer, 2011.

[66] Päivi Majaranta. *Gaze Interaction and Applications of Eye Tracking: Advances in Assistive Technologies: Advances in Assistive Technologies.* IGI Global, 2011.

[67] Cristina Manresa-Yee, Pere Ponsa, Javier Varona, and Francisco J Perales. User experience to improve the usability of a vision-based interface. *Interacting with Computers*, 22(6):594–605, 2010.

[68] Cesar Mauri. Enable viacam, 2017. Retrieved January 31, 2022 from http://eviacam.crea-si.com/index.php.

[69] Cesar Mauri. Eva facial mouse, 2018. Retrieved January 31, 2022 from https://github.com/cmauri/eva_facial_mouse.

[70] César Mauri, Toni Granollers i Saltiveri, Jesús Lorés Vidal, and Mabel García.

Computer vision interaction for people with severe movement restrictions. *Human Technology: An Interdisciplinary Journal on Humans in ICT Environments, vol. 2, núm. 1, p. 38-54*, 2006.

[71] Microsoft. Eye control for windows 10, 2018. Retrieved July 16, 2018 from `https://www.microsoft.com/en-us/garage/wall-of-fame/eye-control-windows-10/`.

[72] Kyle Montague, Hugo Nicolau, and Vicki L Hanson. Motor-impaired touchscreen interactions in the wild. In *Proceedings of the 16th international ACM SIGACCESS conference on Computers & accessibility*, pages 123–130. ACM, 2014.

[73] Rahim Moradi, Esmaeil Zaraii Zavaraki, Parviz Sharifi-Daramadi, Mohammad Reza Nili-Ahmadabadi, and Ali Delavar. The impact of an instructional model with assistive technology on achievement satisfaction of people with physical-motor impairments. *Research and Development in Medical Education*, 7(2):95–101, 2018.

[74] Martez E Mott, Radu-Daniel Vatavu, Shaun K Kane, and Jacob 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*, pages 1934–1946, New York, NY, USA, 2016. Association for Computing Machinery.

[75] MyGaze. Mygaze assistive, 2018. Retrieved July 16, 2018 from `http://www.mygaze.com/products/mygaze-assistive/`.

[76] Neil Mawston. Apple iphone x becomes world's no.1 smartphone model in q1 2018. Technical report, Strategy Analytics, 2018. Retrieved August 13, 2018 from `https://www.strategyanalytics.com/access-services/devices/mobile-phones/smartphone/smartphone-model-tracker/reports/report-detail/apple-iphone-x-becomes-world-s-best-selling-smartphone-model-in-q1-2018`.

[77] Trustees of Boston College. Cameramouse, 2018. Retrieved January 31, 2022 from http://www.cameramouse.org/.

[78] University of New Orleans. Mobility impairments. *Office of Disability Services*, 2023.

[79] University of Washington. Mobility impairments. *DO-IT*, 2023.

[80] SUNY Oswego. Mobility and health impairments. *Accessibility Resources*, 2023.

[81] Quha oy. Quha zono, 2018. Retrieved July 15, 2018 from `http://www.quha.com/products-2/zono/`.

[82] Abdul Moiz Penkar, Christof Lutteroth, and Gerald Weber. Designing for the eye: design parameters for dwell in gaze interaction. In *Proceedings of the 24th Australian Computer-Human Interaction Conference*, pages 479–488. ACM, 2012.

[83] Ondrej Polacek, Thomas Grill, and Manfred Tscheligi. Nosetapping: what else can you do with your nose? In *Proceedings of the 12th International Conference on Mobile and Ubiquitous Multimedia*, pages 1–9, New York, NY, USA, 2013. Association for Computing Machinery.

[84] Ronald E Prior. Mobility aids for the severely handicapped a status report" charles m. scott, bsme president. *Bulletin of Prosthetics Research*, 26:192, 1976. PMID: 143300.

[85] Robert G Radwin, Gregg C Vanderheiden, and Mei-Li Lin. A method for evaluating head-controlled computer input devices using fitts' law. *Human factors*, 32(4):423–438, 1990.

[86] Rajeev Ranjan, Shalini De Mello, and Jan Kautz. Light-weight head pose invariant gaze tracking. *arXiv preprint arXiv:1804.08572*, 2018.

[87] Cameron N Riviere and Nitish V Thakor. Effects of age and disability on tracking tasks with a computer mouse: Accuracy and linearity. *Journal of Rehabilitation Research and Development*, 33(1):6–15, February 1996. PMID: 8868412.

[88] Maria Roig-Maimó, Cristina Manresa-Yee, and Javier Varona. A robust camera-based interface for mobile entertainment. *Sensors*, 16(2):254, 2016.

[89] Maria Francesca Roig-Maimó, I Scott MacKenzie, Cristina Manresa-Yee, and Javier Varona. Evaluating fitts' law performance with a non-iso task. In *Pro-*

ceedings of the XVIII International Conference on Human Computer Interaction, page 5. ACM, 2017.

[90] Maria Francesca Roig-Maimó, I Scott MacKenzie, Cristina Manresa-Yee, and Javier Varona. Head-tracking interfaces on mobile devices: Evaluation using fitts' law and a new multi-directional corner task for small displays. *International Journal of Human-Computer Studies*, 112:1–15, 2018.

[91] Maria Francesca Roig-Maimó, Cristina Manresa-Yee, Javier Varona, and I Scott MacKenzie. Evaluation of a mobile head-tracker interface for accessibility. In *Computers Helping People with Special Needs: 15th International Conference, ICCHP 2016, Linz, Austria, July 13-15, 2016, Proceedings, Part II 15*, pages 449–456, Cham, 2016. Springer, Springer International Publishing.

[92] Nataniel Ruiz, Eunji Chong, and James M Rehg. Fine-grained head pose estimation without keypoints. In *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, pages 2074–2083, Piscataway, NJ, United States, 2018. Institute of Electrical and Electronics Engineers.

[93] Sergio Ruiz, José Martín Molina-Espinosa, Alejandra J Magana, and Bedrich Benes. Systematic review of multimodal human–computer interaction. *Informatics*, 9(1):13, 2022.

[94] Tyler Simpson, Michel Gauthier, and Arthur Prochazka. Evaluation of tooth-

click triggering and speech recognition in assistive technology for computer access. *Neurorehabilitation and neural repair*, 24(2):188–194, 2010.

[95] Aaron Smith and Monica Anderson. Online shopping and purchasing preferences. Technical report, Pew Research Center, 2016.

[96] R. William Soukoreff and I. Scott MacKenzie. Towards a standard for pointing device evaluation, perspectives on 27 years of fitts' law research in hci. *International Journal of Human-Computer Studies*, 61(6):751–789, 2004.

[97] Bonnie Swaine, Delphine Labbé, Tiiu Poldma, Maria Barile, Catherine Fichten, Alice Havel, Eva Kehayia, Barbara Mazer, Patricia McKinley, and Annie Rochette. Exploring the facilitators and barriers to shopping mall use by persons with disabilities and strategies for improvements: Perspectives from persons with disabilities, rehabilitation professionals and shopkeepers. *ALTER-European Journal of Disability Research/Revue Européenne de Recherche sur le Handicap*, 8(3):217–229, 2014.

[98] Xu Tang, Daniel K. Du, Zeqiang He, and Jingtuo Liu. Pyramidbox: A context-assisted single shot face detector. *CoRR*, 2018.

[99] JL Taylor and DI McCloskey. Proprioception in the neck. *Experimental Brain Research*, 70:351–360, 1988.

[100] Tecla. Introduction to 7 common adaptive switches, 2018. Re-

trieved July 16, 2018 from `https://gettecla.com/blogs/news/introduction-to-assistive-switches`.

[101] Paul Viola and Michael Jones. Rapid object detection using a boosted cascade of simple features. In *Proceedings of the 2001 IEEE computer society conference on computer vision and pattern recognition. CVPR 2001*, volume 1, pages I–I. Ieee, 2001.

[102] Xinyao Wang, Liefeng Bo, and Fuxin Li. Adaptive wing loss for robust face alignment via heatmap regression. *CoRR*, 2019.

[103] WebAIM. Assistive technologies, 2018. Retrieved July 12, 2018 from `https://webaim.org/articles/motor/assistive`.

[104] Jacob O Wobbrock. Improving pointing in graphical user interfaces for people with motor impairments through ability-based design. In *Assistive Technologies and Computer Access for Motor Disabilities*, pages 206–253. IGI Global, 2014.

[105] Wayne Wu, Chen Qian, Shuo Yang, Quan Wang, Yici Cai, and Qiang Zhou. Look at boundary: A boundary-aware face alignment algorithm. *CoRR*, 2018.

[106] Tsun-Yi Yang, Yi-Ting Chen, Yen-Yu Lin, and Yung-Yu Chuang. Fsa-net: Learning fine-grained structure aggregation for head pose estimation from a single image. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 1087–1096, Piscataway, NJ, United States, 2019. Institute of Electrical and Electronics Engineers.

[107] Xiaoyi Zhang, Harish Kulkarni, and Meredith Ringel Morris. Smartphone-based gaze gesture communication for people with motor disabilities. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems*, pages 2878–2889. ACM, 2017.

[108] Xuan Zhang and I Scott MacKenzie. Evaluating eye tracking with iso 9241-part 9. In *International Conference on Human-Computer Intrction*, pages 779–788. Springer, 2007.

[109] Rafael Zuniga and John Magee. Camera mouse: dwell vs. computer vision-based intentional click activation. In *Universal Access in Human–Computer Interaction. Designing Novel Interactions: 11th International Conference, UAHCI 2017, Held as Part of HCI International 2017, Vancouver, BC, Canada, July 9–14, 2017, Proceedings, Part II 11*, pages 455–464, Cham, 2017. Springer, Springer International Publishing.

Chapter 8

Appendix

.1 Target Trajectories

The following figures visualize the target trajectories. They are the first frame of each video that the participants viewed on the full screen. Note that the target is represented by a white disk and follows the path from the brighter end to the darker one. The participants were asked to indicate the target's motion by their heads.

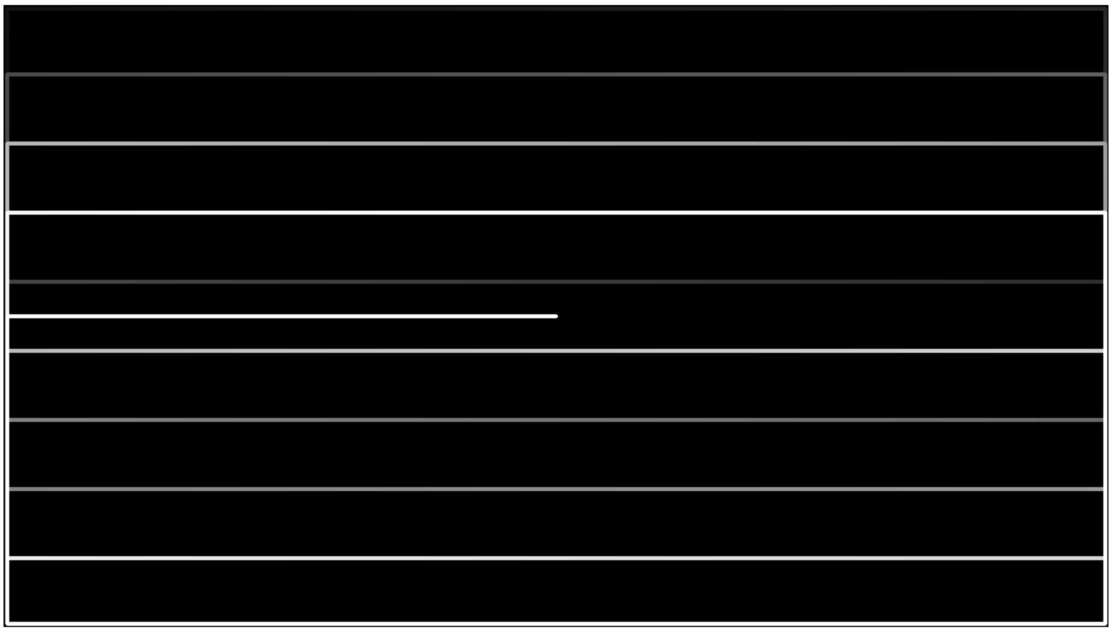


Figure .1: the trajectory labeled as *horizontal*

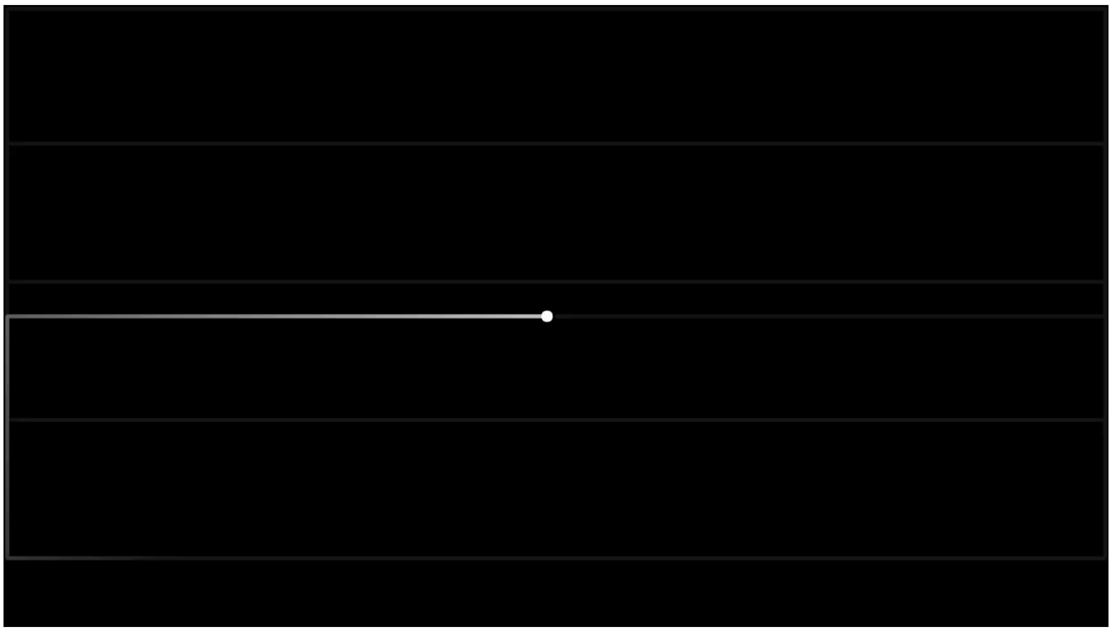


Figure .2: the trajectory labeled as *horizontal_part1_slow*

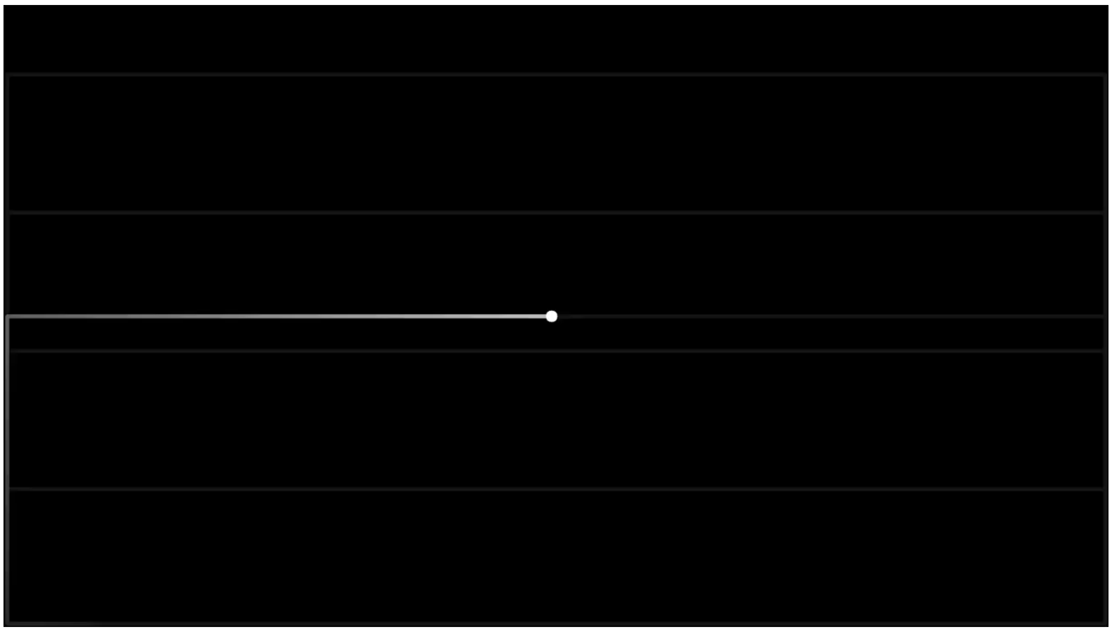


Figure .3: the trajectory labeled as *horizontal_part2_slow*

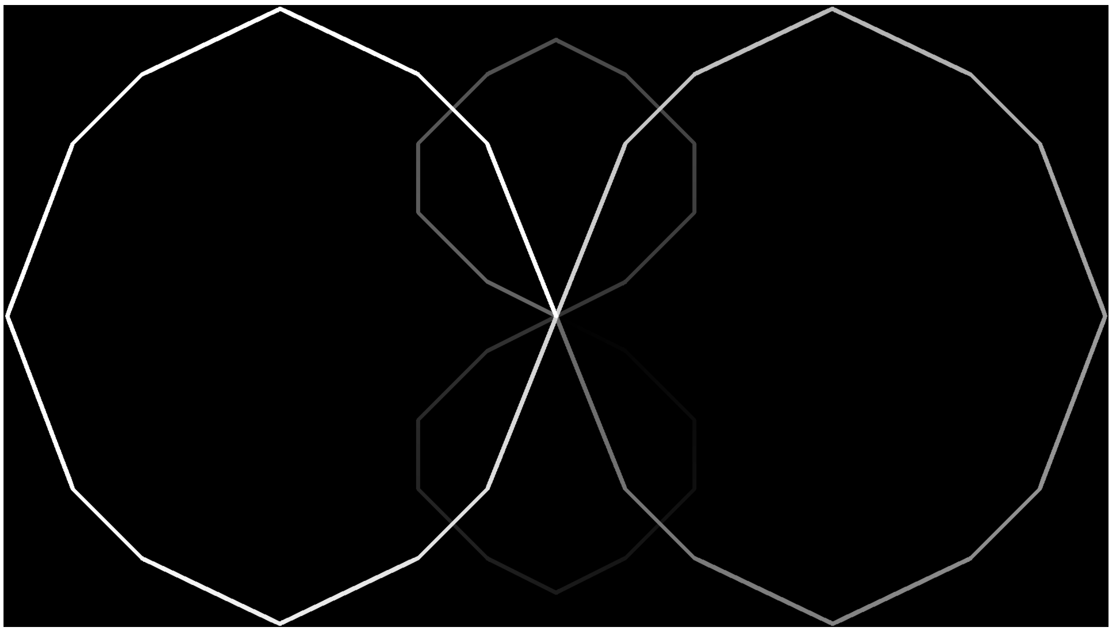


Figure .4: the trajectory labeled as *infinity*

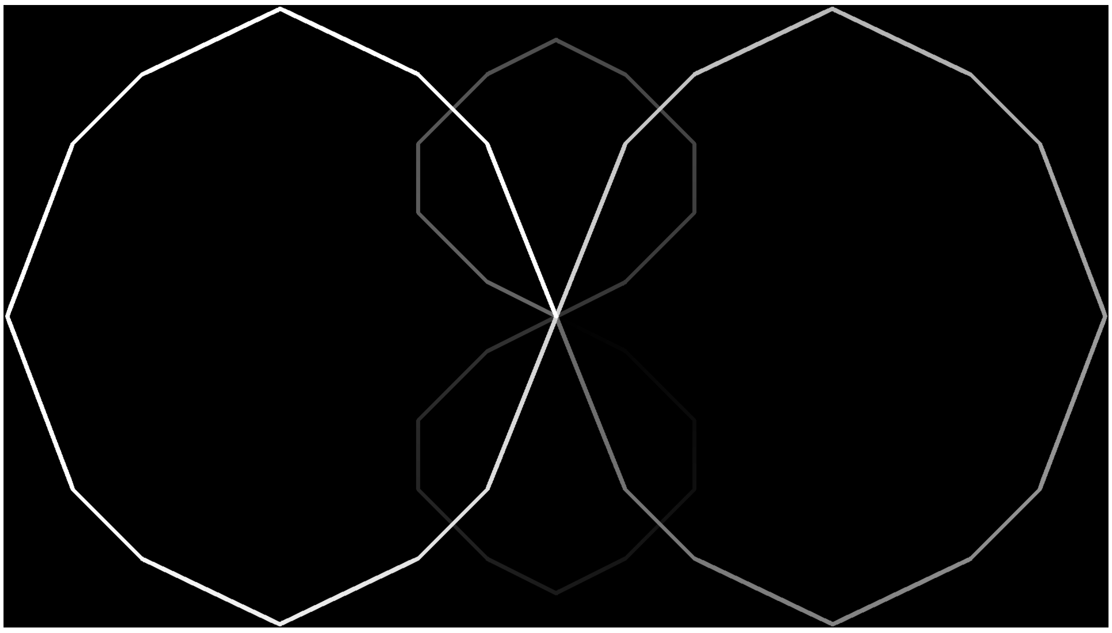


Figure .5: the trajectory labeled as *infinity_slow*

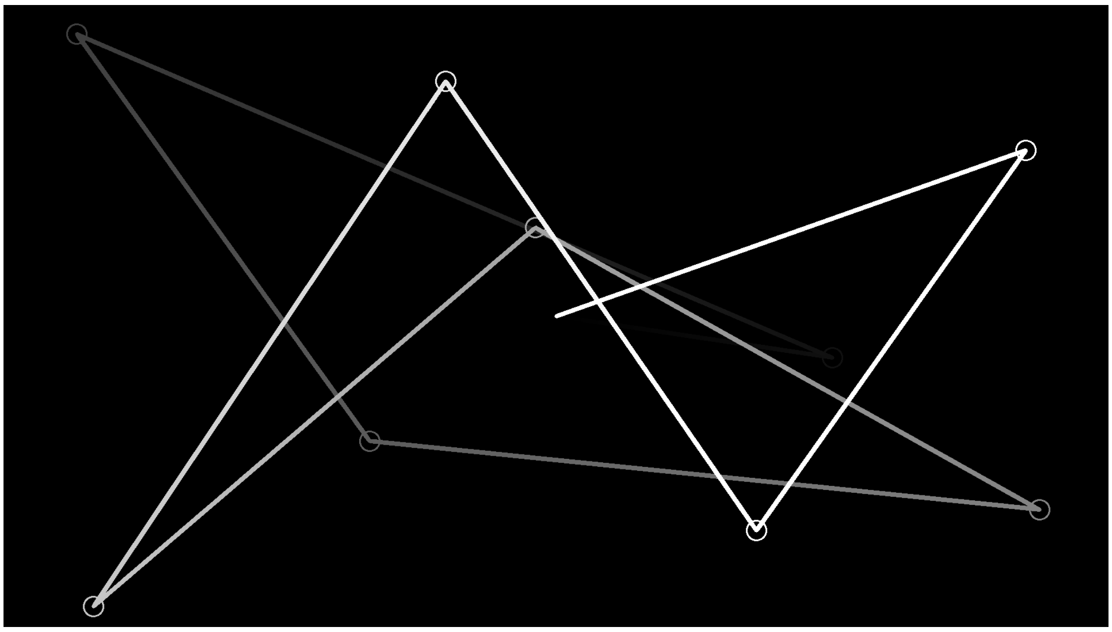


Figure .6: the trajectory labeled as *random1*

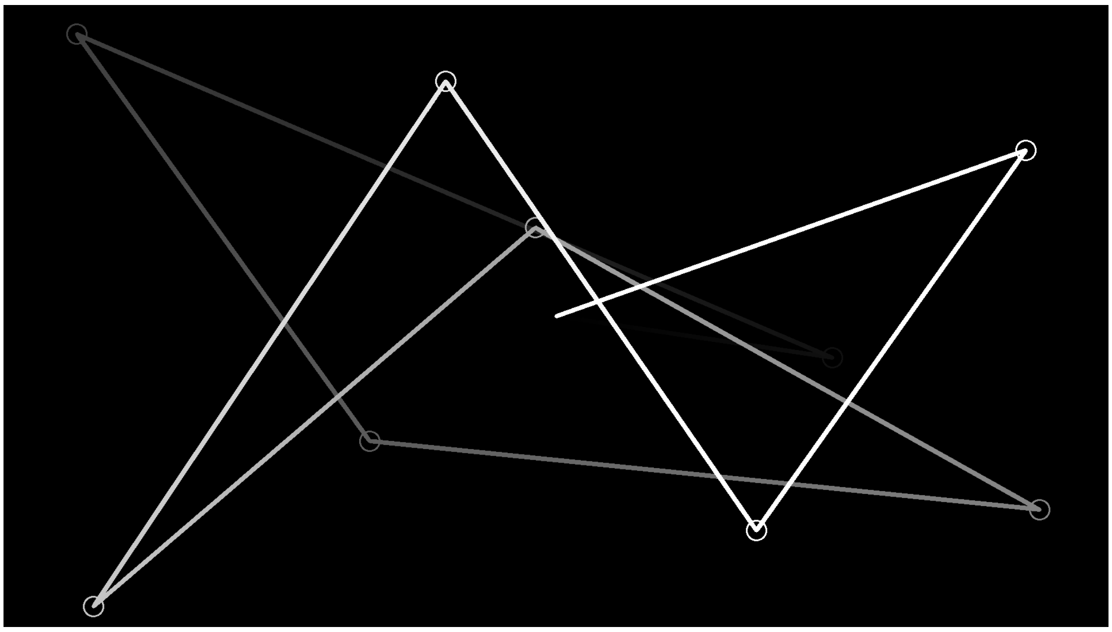


Figure .7: the trajectory labeled as *random1_slow*

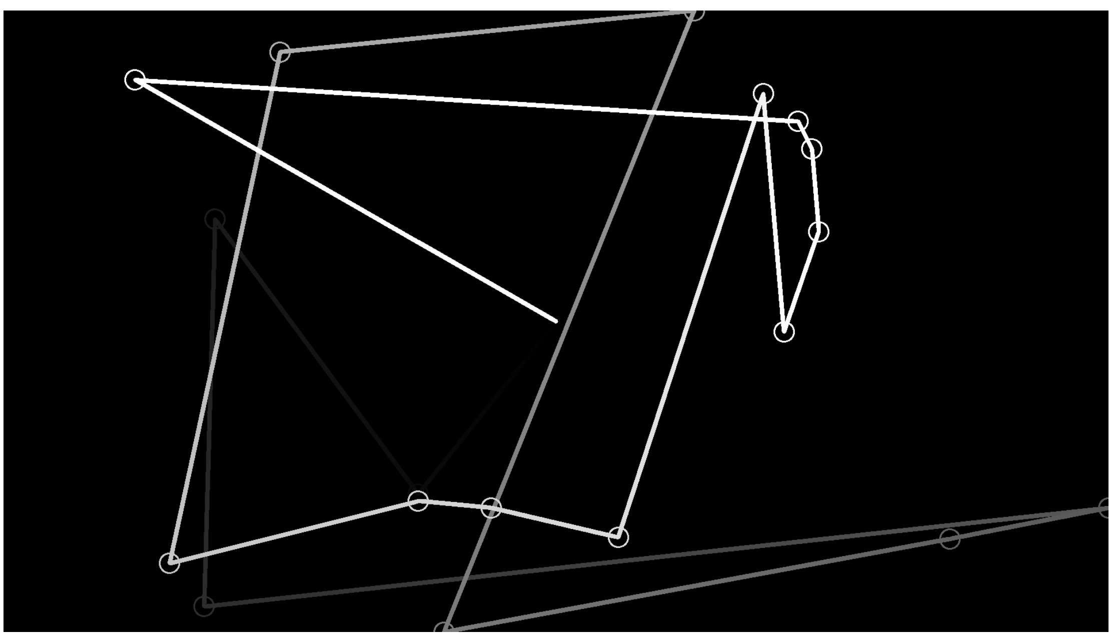


Figure .8: the trajectory labeled as *random4*

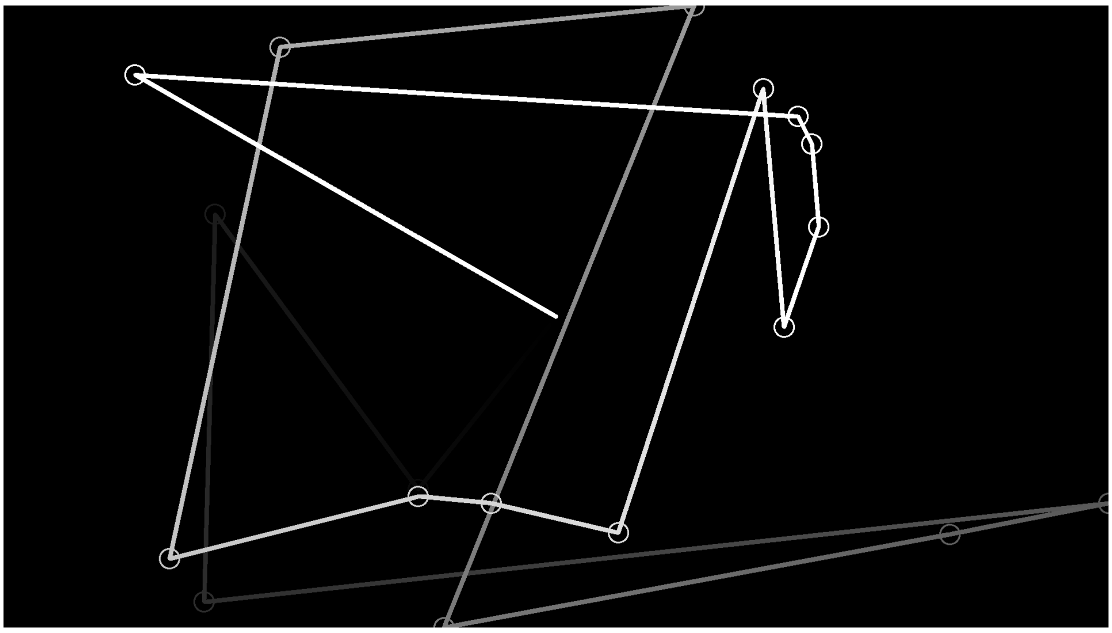


Figure .9: the trajectory labeled as *random4_slow*

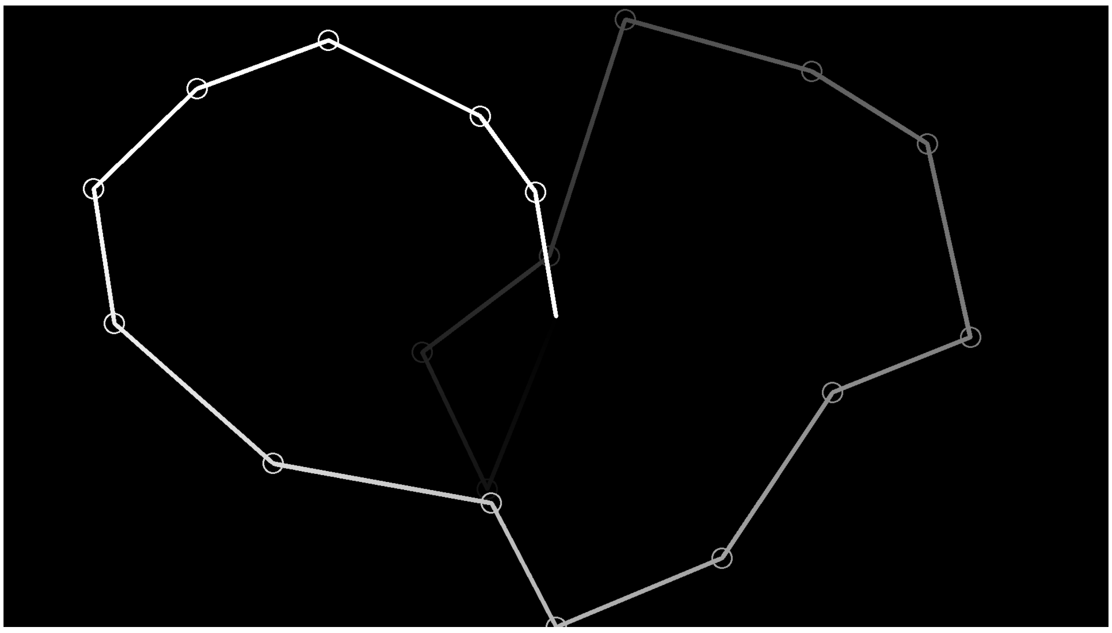


Figure .10: the trajectory labeled as *random5*

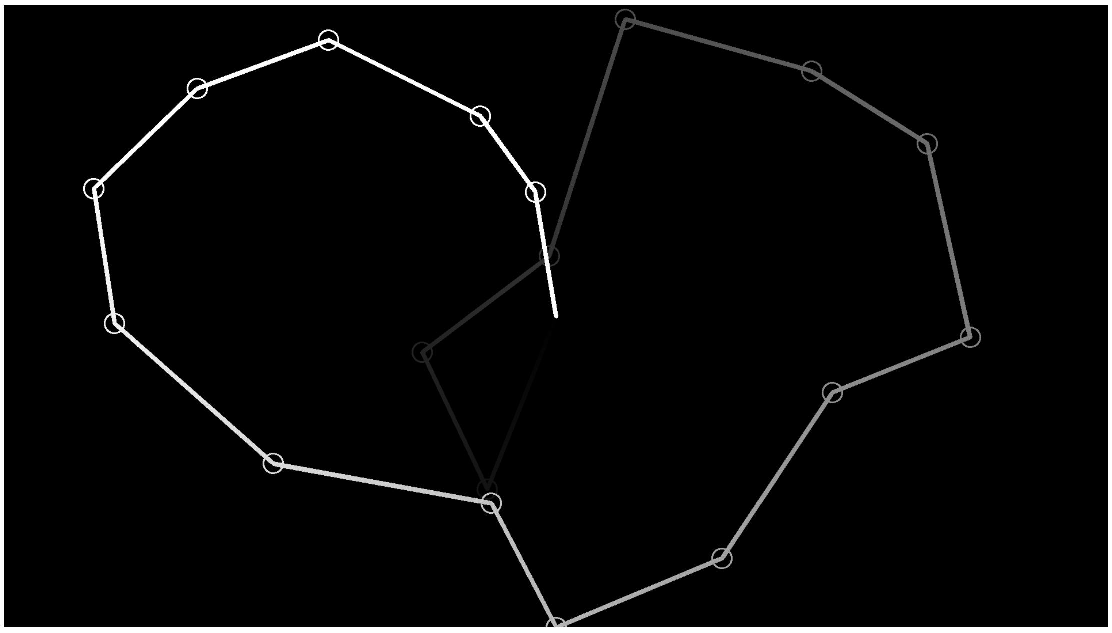


Figure .11: the trajectory labeled as *random5_slow*

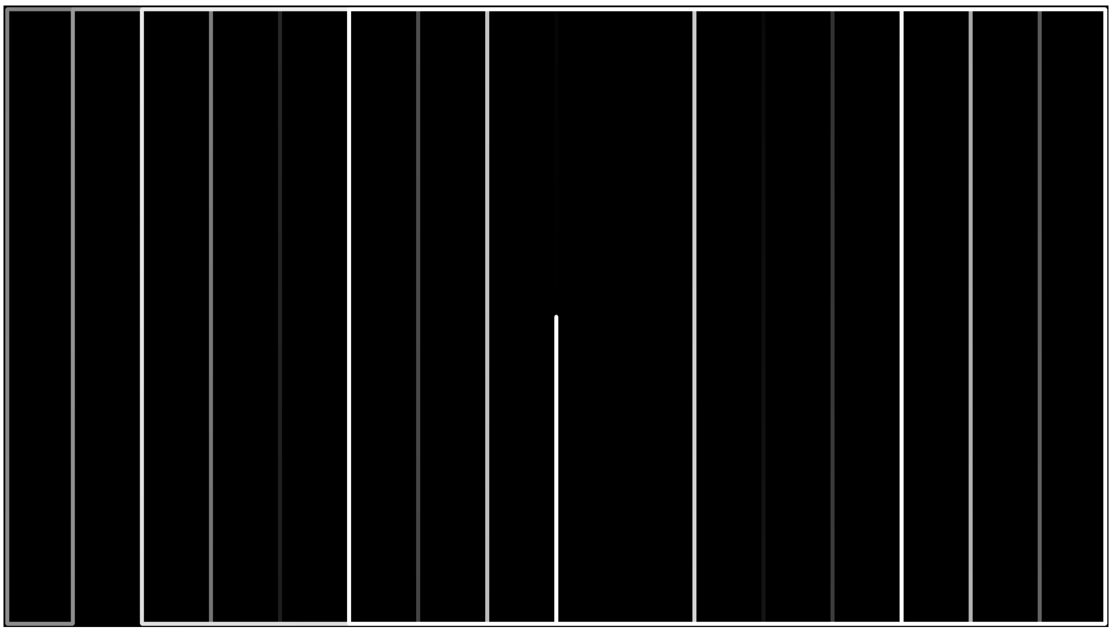


Figure .12: the trajectory labeled as *vertical*

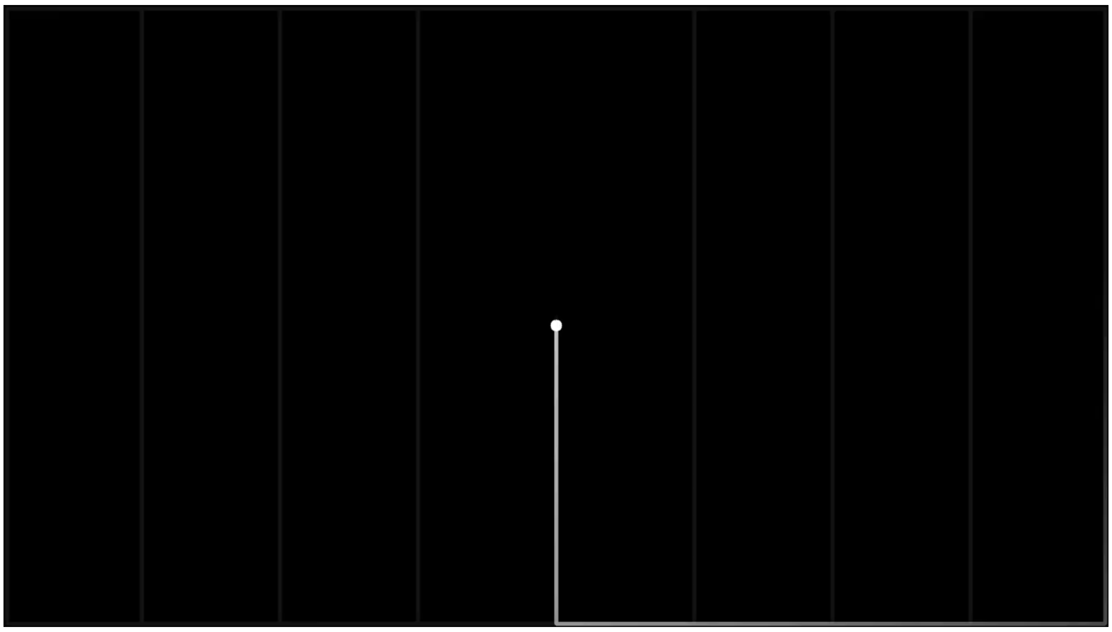


Figure .13: the trajectory labeled as *vertical_part1_slow*

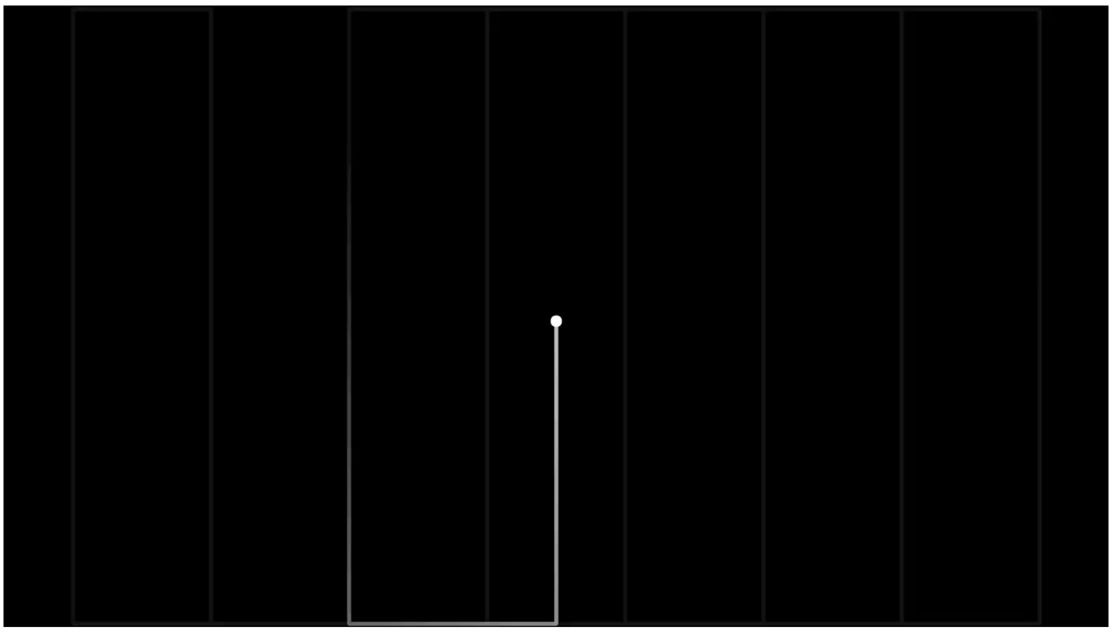


Figure .14: the trajectory labeled as *vertical_part1_slow*

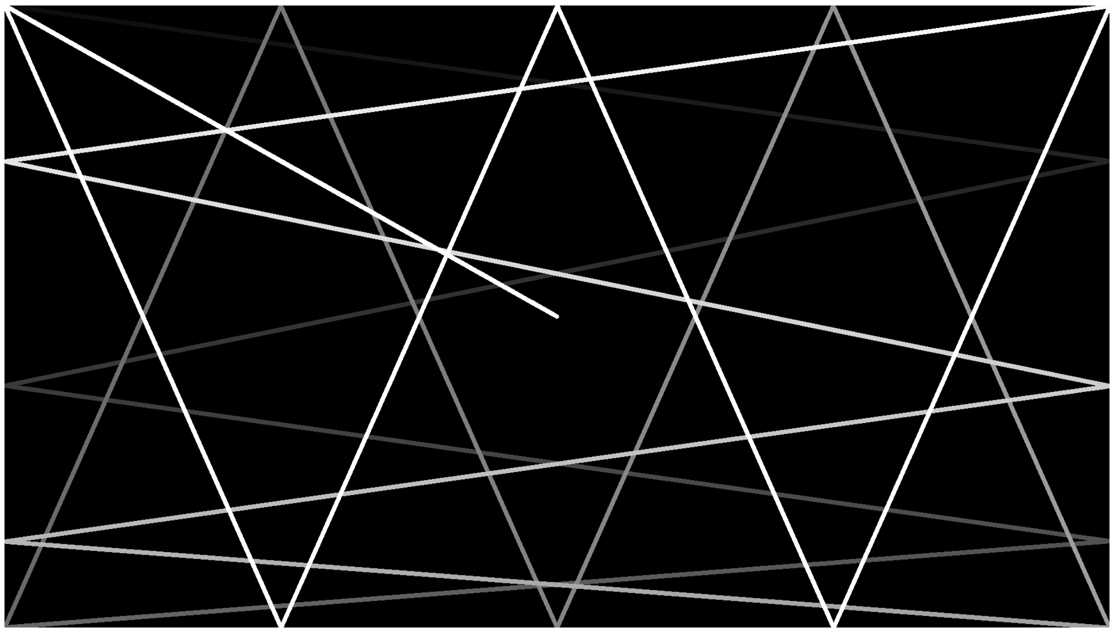


Figure .15: the trajectory labeled as *zigzag*

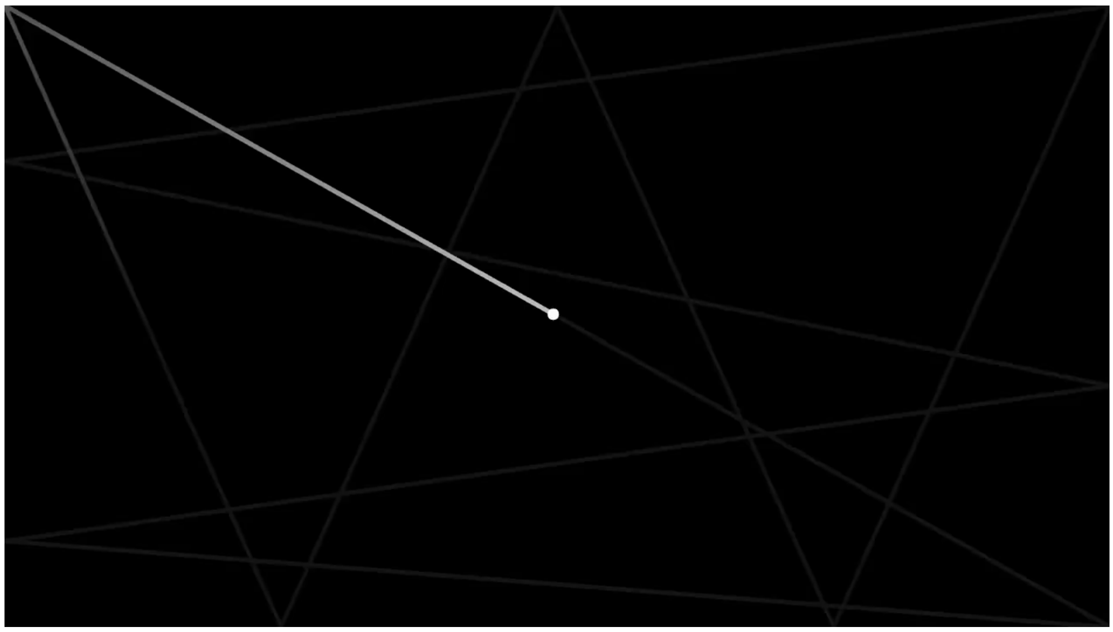


Figure .16: the trajectory labeled as *zigzag_part2_slow*

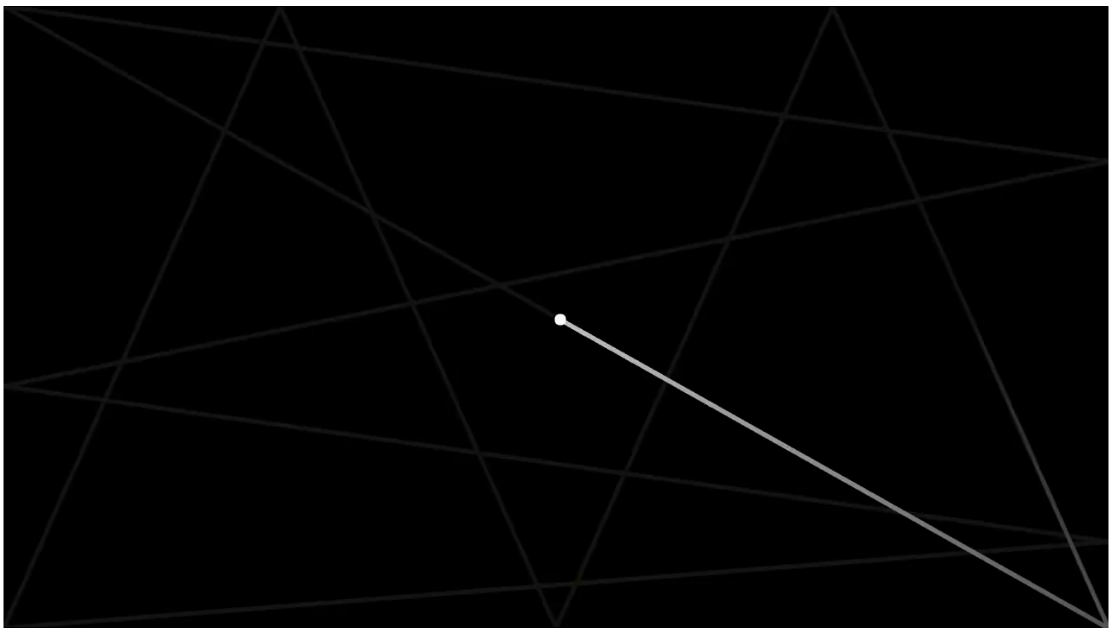
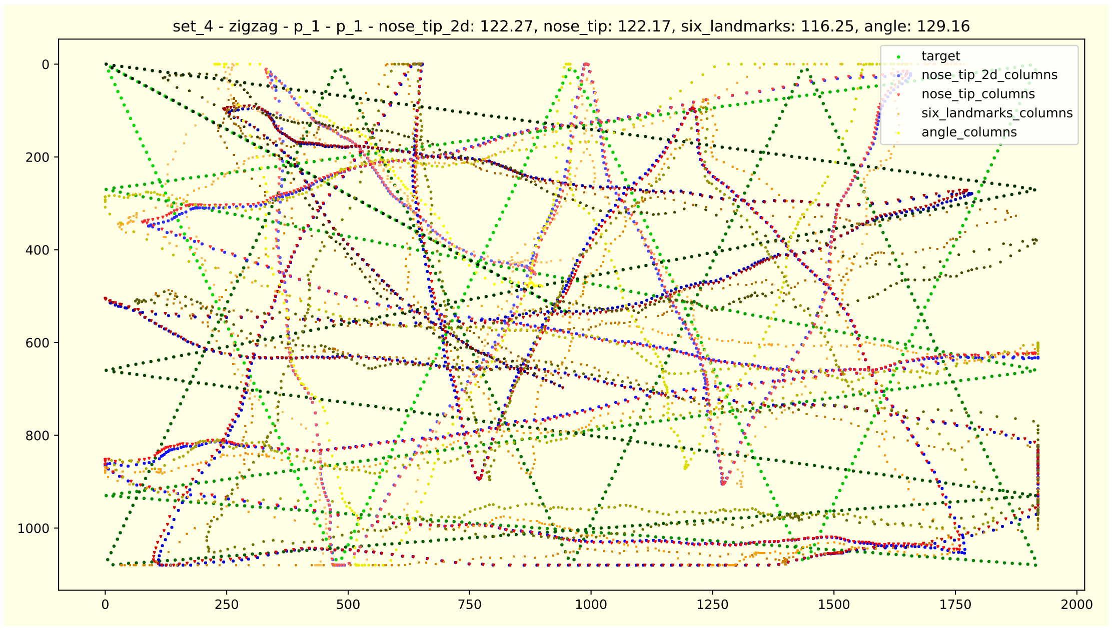
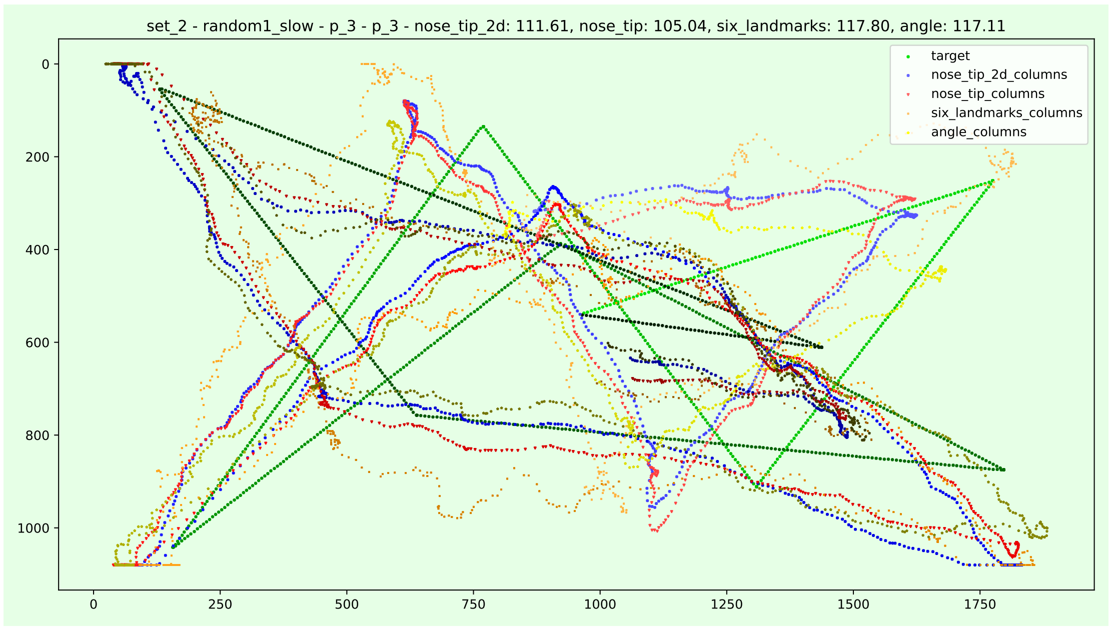


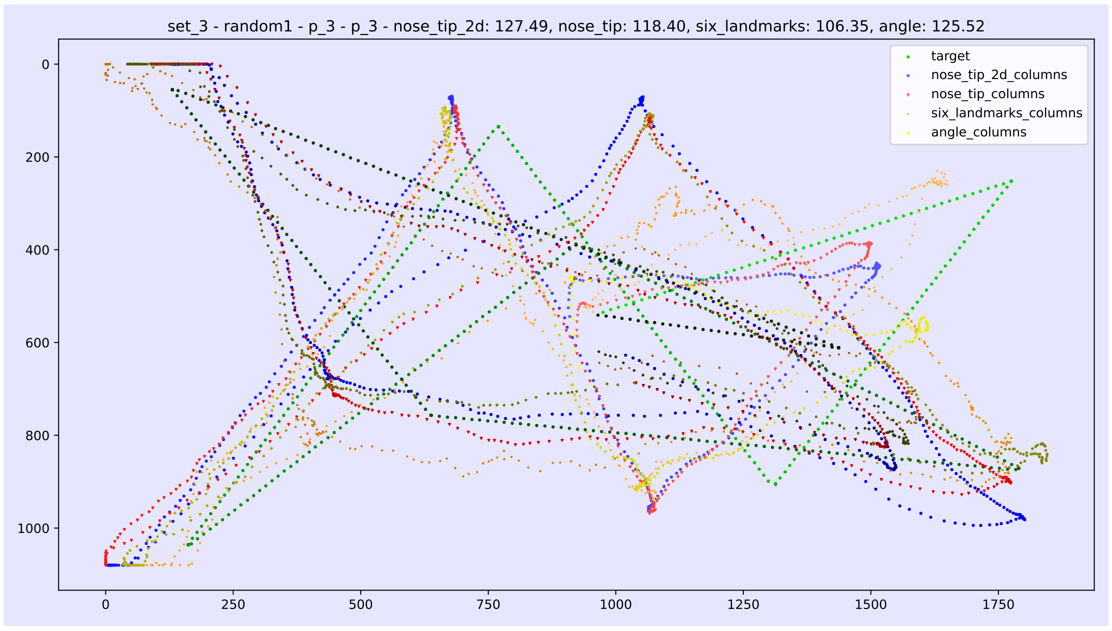
Figure .17: the trajectory labeled as *zigzag_part2_slow*

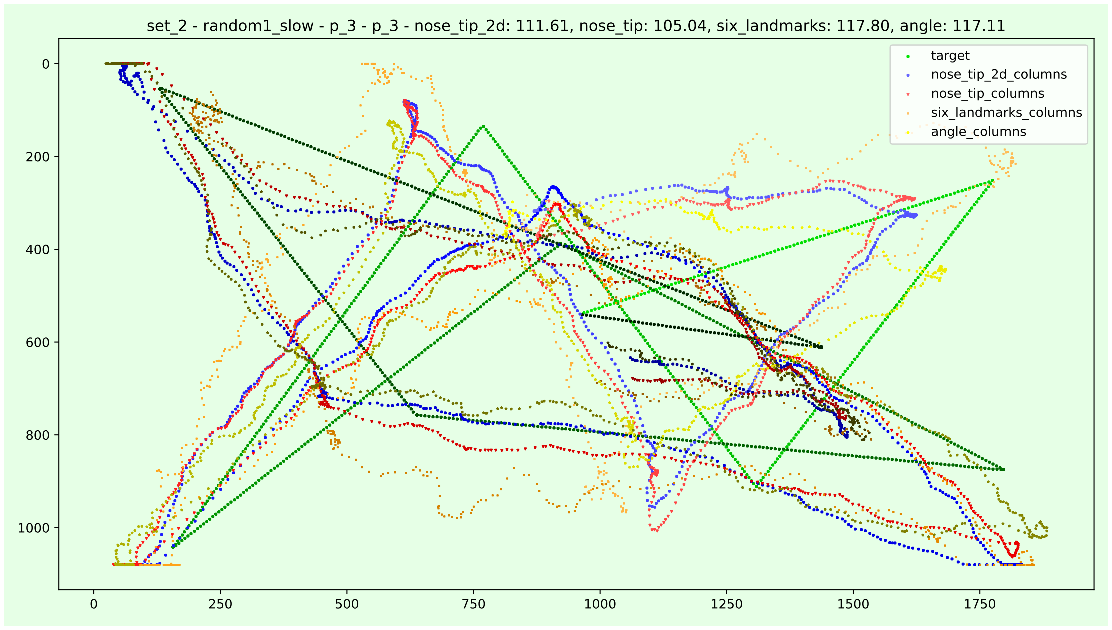
.2 Performance of Different Models

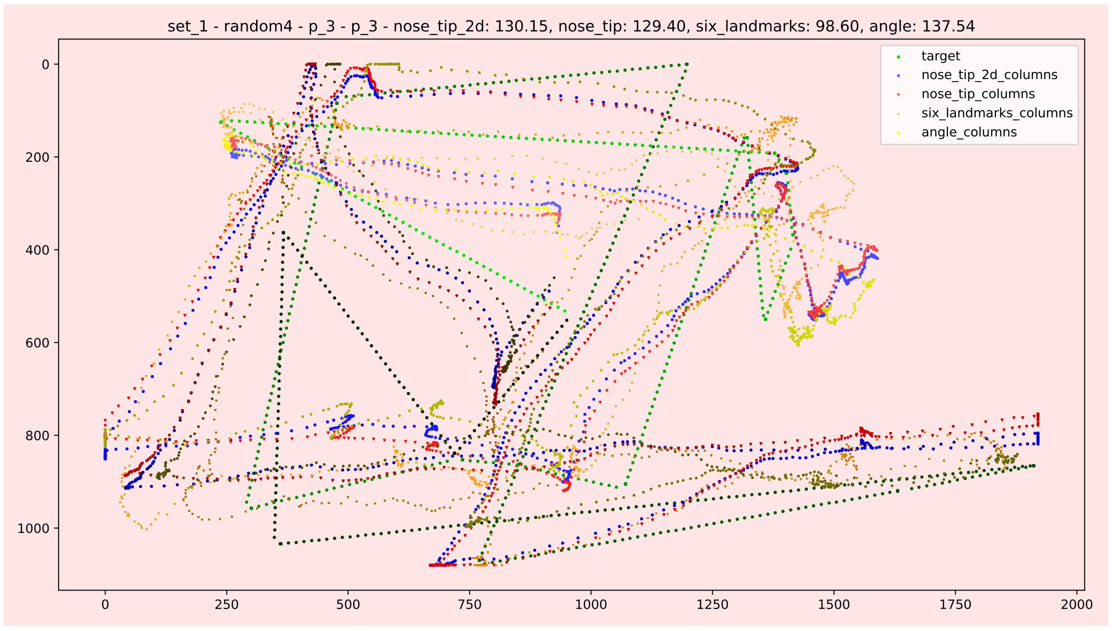
The following figures visualize the performance of different *personal* RNN models that are trained with *exact_values*. Note that the target trajectory is represented in green and moves from the brighter end to the darker one. The models estimate the desired pointing coordinates at each step. Different colors indicate the estimations based on different feature sets.

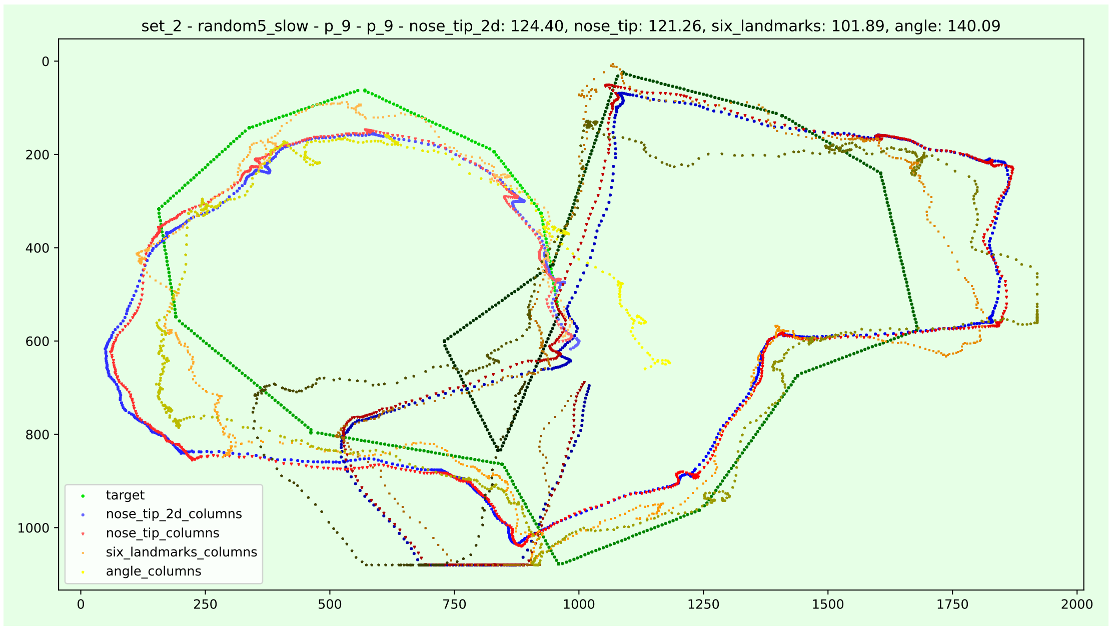

set_4 - zigzag - p_1 - p_1 - nose_tip_2d: 122.27, nose_tip: 122.17, six_landmarks: 116.25, angle: 129.16
target
nose_tip_2d_columns
nose_tip_columns
six_landmarks_columns
angle_columns
0
200
400
600
800
1000
0
250
500
750
1000
1250
1500
1750
2000


set_2 - random1_slow - p_3 - p_3 - nose_tip_2d: 111.61, nose_tip: 105.04, six_landmarks: 117.80, angle: 117.11
target
nose_tip_2d_columns
nose_tip_columns
six_landmarks_columns
angle_columns
0
200
400
600
800
1000
0
250
500
750
1000
1250
1500
1750


set_3 - random1 - p_3 - p_3 - nose_tip_2d: 127.49, nose_tip: 118.40, six_landmarks: 106.35, angle: 125.52
target
nose_tip_2d_columns
nose_tip_columns
six_landmarks_columns
angle_columns
0
200
400
600
800
1000
0
250
500
750
1000
1250
1500
1750


set_2 - random1_slow - p_3 - p_3 - nose_tip_2d: 111.61, nose_tip: 105.04, six_landmarks: 117.80, angle: 117.11
target
nose_tip_2d_columns
nose_tip_columns
six_landmarks_columns
angle_columns
0
200
400
600
800
1000
0
250
500
750
1000
1250
1500
1750


set_1 - random4 - p_3 - p_3 - nose_tip_2d: 130.15, nose_tip: 129.40, six_landmarks: 98.60, angle: 137.54
target
nose_tip_2d_columns
nose_tip_columns
six_landmarks_columns
angle_columns
0
200
400
600
800
1000
0
250
500
750
1000
1250
1500
1750
2000


set_2 - random5_slow - p_9 - p_9 - nose_tip_2d: 124.40, nose_tip: 121.26, six_landmarks: 101.89, angle: 140.09
target
nose_tip_2d_columns
nose_tip_columns
six_landmarks_columns
angle_columns
0
200
400
600
800
1000
0
250
500
750
1000
1250
1500
1750
2000