ADDRESSING SITUATIONAL AND PHYSICAL IMPAIRMENTS AND DISABILITIES WITH A GAZE-ASSISTED, MULTI-MODAL, ACCESSIBLE INTERACTION PARADIGM

A Dissertation

by

VIJAY DANDUR RAJANNA

Submitted to the Office of Graduate and Professional Studies of
Texas A&M University
in partial fulfillment of the requirements for the degree of

DOCTOR OF PHILOSOPHY

Chair of Committee,	Tracy Hammond
Committee Members,	Andruid Kerne
	Dylan Shell
	Steven M. Smith
Head of Department,	Dilma Da Silva

December 2018

Major Subject: Computer Science

ABSTRACT

Every day we encounter a variety of scenarios that lead to situationally induced impairments and disabilities, i.e., our hands are assumed to be engaged in a task, and hence unavailable for interacting with a computing device. For example, a surgeon performing an operation, a worker in a factory with greasy hands or wearing thick gloves, a person driving a car, and so on all represent scenarios of situational impairments and disabilities. In such cases, performing point-and-click interactions, text entry, or authentication on a computer using conventional input methods like the mouse, keyboard, and touch is either inefficient or not possible. Unfortunately, individuals with physical impairments and disabilities, by birth or due to an injury, are forced to deal with these limitations every single day. Generally, these individuals experience difficulty or are completely unable to perform basic operations on a computer. Therefore, to address situational and physical impairments and disabilities it is crucial to develop hands-free, accessible interactions.

In this research, we try to address the limitations, inabilities, and challenges arising from situational and physical impairments and disabilities by developing a gaze-assisted, multi-modal, hands-free, accessible interaction paradigm. Specifically, we focus on the three primary interactions: 1) point-and-click, 2) text entry, and 3) authentication. We present multiple ways in which the gaze input can be modeled and combined with other input modalities to enable efficient and accessible interactions. In this regard, we have developed a gaze and foot-based interaction framework to achieve accurate “point-and-click" interactions and to perform dwell-free text entry on computers. In addition, we have developed a gaze gesture-based framework for user authentication and to interact with a wide range of computer applications using a common repository of gaze gestures. The interaction methods and devices we have developed are a) evaluated using the standard HCI procedures like the Fitts’ Law, text entry metrics, authentication accuracy and video analysis attacks, b) compared against the speed, accuracy, and usability of other gaze-assisted interaction methods, and c) qualitatively analyzed by conducting user interviews.

From the evaluations, we found that our solutions achieve higher efficiency than the existing

systems and also address the usability issues. To discuss each of these solutions, first, the gaze and foot-based system we developed supports point-and-click interactions to address the “Midas Touch" issue. The system performs at least as good (time and precision) as the mouse, while enabling hands-free interactions. We have also investigated the feasibility, advantages, and challenges of using gaze and foot-based point-and-click interactions on standard (up to 24") and large displays (up to 84") through Fitts' Law evaluations. Additionally, we have compared the performance of the gaze input to the other standard inputs like the mouse and touch.

Second, to support text entry, we developed a gaze and foot-based dwell-free typing system, and investigated foot-based activation methods like foot-press and foot gestures. we have demonstrated that our dwell-free typing methods are efficient and highly preferred over conventional dwell-based gaze typing methods. Using our gaze typing system the users type up to 14.98 Words Per Minute (WPM) as opposed to 11.65 WPM with dwell-based typing. Importantly, our system addresses the critical usability issues associated with gaze typing in general.

Third, we addressed the lack of an accessible and shoulder-surfing resistant authentication method by developing a gaze gesture recognition framework, and presenting two authentication strategies that use gaze gestures. Our authentication methods use static and dynamic transitions of the objects on the screen, and they authenticate users with an accuracy of 99% (static) and 97.5% (dynamic). Furthermore, unlike other systems, our dynamic authentication method is not susceptible to single video iterative attacks, and has a lower success rate with dual video iterative attacks.

Lastly, we demonstrated how our gaze gesture recognition framework can be extended to allow users to design gaze gestures of their choice and associate them to appropriate commands like minimize, maximize, scroll, etc., on the computer. We presented a template matching algorithm which achieved an accuracy of 93%, and a geometric feature-based decision tree algorithm which achieved an accuracy of 90.2% in recognizing the gaze gestures. In summary, our research demonstrates how situational and physical impairments and disabilities can be addressed with a gaze-assisted, multi-modal, accessible interaction paradigm.

DEDICATION

This dissertation is dedicated to my brother, Vinaya Rajanna. Without his enduring support and encouragement, this work would not have been possible. I am humbled and thankful to my parents Renuka and Rajanna for all their sacrifices. They prioritized my dream to pursue a doctoral degree over their wishes. Lastly, my deepest gratitude to the almighty who keeps me motivated and going forward despite the hurdles.

ACKNOWLEDGMENTS

I express my deepest gratitude to my advisor, Dr. Tracy Hammond. I should be extremely fortunate to have you as an advisor. You have been a great mentor, you believed in me, extended great opportunities, and constantly encouraged me to give my best. You are an inspiration, a role model, you have made a lasting impact on my learning and research career. Thanks to my advisory committee Dr. Andruid Kerne, Dr. Dylan Shell, and Dr. Steven M. Smith. Dr. Kerne, learning Human-Centered Computing from you was one of my best learning experiences, it gave me an entire new outlook to the field of Computer Science. Dr. Smith, learning Cognitive Psychology from you helped me significantly in shaping my research around cognition in human-computer interaction. Dr. Shell, you have taught me to think deeper into the research problems by asking various questions, and reflecting back on my work addresses those questions. Thanks to Dr. Dilma Da Silva, Dr. John Keyser, Dr. Joseph Hurley, Dr. Aakash Tyagi and Dr. Vikram Kinra you all have mentored me and taught me how to instruct a class when I served as a graduate teaching fellow. Thanks to Dr. Michael Moore, you have encouraged my research with your new ideas, and consistently supported the user studies in our lab.

Thanks to Dr. John Paulin Hansen, from DTU, Denmark. Your suggestions and feedback were crucial as I was developing a gaze typing system. Importantly, thank you for providing an opportunity to intern in your lab at DTU, and introducing me to gaze interaction in virtual reality. Thanks to Dr. Scott I. Mackenzie for all your guidance while I worked on Fitts' Law experiments. My learning was significant.

Thanks to my amazing lab members at the Sketch Recognition Lab, you have been my inspiration and a strong moral support. Special thanks to Josh Cherian, you have reviewed and provided feedback on most of my publications. You have always made time to help me and your feedback always improved the quality of my works. Thanks to Paul Taele, Folami Alamudun, Stephanie Valentine, and Manoj Prasad. As senior lab members you all have constantly mentored me. You taught me how to pursue research, how to write research papers, how to make presentations, and

importantly how to deal to the stress. Thanks to Anna Stepanova, your continuous support and encouragement helped me to achieve my goals. Thanks to Murat Russel, Seth Polsley, Raniero Lara Garduno, Jung In Koh, Adil Malla, Matthew Runyon, Blake Williford, Hong-Hoe Kim, Larry Powell, Aqib Bhat, JorgeIvan Camara, Tianshu Chu, Shiqiang Guo, Siddhartha Karthik, Purnendu Kaul, Nahum Villanueva, Zhengliang Yin, and Jaideep Ray. You all have made my journey at Texas A&M memorable.

A special thanks to the department staff for their assistance over the years: Kathy Waskom, Karrie Bourquin, Elena Rodriguez, Sybil Popham, Valerie Sorenson, Leslie Darling, Dave Cote, Bruce Veals, Theresa Roberts.

CONTRIBUTORS AND FUNDING SOURCES

Contributors

This work was supervised by a dissertation committee consisting of Professor Tracy Hammond (advisor), Department of Computer Science and Engineering and Professor Andruid Kerne, Department of Computer Science and Engineering and Professor Dylan Shell, Department of Computer Science and Engineering and Professor Steven M. Smith, Department of Psychology.

Adil Hamid Malla and Rahul Ashok Bhagat implemented the random path generation algorithm used for gaze gesture-based authentication presented in Chapter 5. Murat Russell created a compact 3D printed case that houses the electronics for foot press sensing device used for gaze typing presented in Chapter 4. Jeff Zaho created the foot gesture recognition device used for gaze typing presented in Chapter 4. Mohammad Khan remodeled the 3D printed case of foot press sensing device to be used in upright stance presented in Chapter 3. All these collaborators were from the Department of Computer Science and Engineering.

All other work conducted for the dissertation was completed by the student.

Funding Sources

No outside funding was received for the research and compilation of this document.

TABLE OF CONTENTS

Page

LIST OF FIGURES

FIGURE Page

LIST OF TABLES

TABLE Page

1. INTRODUCTION

1.1 The Need for Hands-free, Accessible Interaction Methods

When personal computers were first introduced back in the 80s, they were meant to be used for specific tasks, in specific ways, and with specific input and output units. However, with the advancements in ubiquitous computing, i.e., computing available anywhere, anytime, and on any device, the notion that people use computers in structured spaces is becoming obsolete. Today, we live in a world where we are constantly interacting with computing devices while sitting (desktop/laptop), moving (mobile phone/fitness trackers), and even sleeping (sleep trackers). These scenarios pose many challenges on how these interactions should be designed? What is the minimal human effort required? How can we achieve efficiency? Can we multitask? and so on. While we try to answer these questions by developing novel interactions and supplementary devices, we must consider a "Human-centered" approach. Why? Because the best technology fits seamlessly into people's lives to the point where it is not even considered a technology and is forgiving of human errors.

While we have made computing devices available everywhere, and they have been increasingly used in various forms and at various places like computers in a household, automobiles, manufacturing industry, hospitals, etc., we have been consistently using the mouse and keyboard-based interactions and sometimes touch. These interactions make it essential that the user's hands are available to operate the system, which is not always possible. Here are a few examples of such scenarios. Advancements in surgical technology have enabled a surgeon to view the imagery of the part of the body being operated in greater details through high resolution cameras. However, if the surgeon wants to zoom, pan, switch images or perform any other operation, she is required to go through the cumbersome process of changing the gloves, working on the computer, sterilizing the hands again, putting on the glove, and continuing with the surgery.

As reported, these operations would take around ten minutes, and a surgeon cannot afford to

lose time during a surgery [1]. An alternative approach is to use gestures or to cover the touch enabled screens with plastic sheets/foils. Using gestures is inaccurate, and the doctor must become hands free by leaving all the surgical instruments. On the other hands, when using plastic sheets on a display, the surgeon must be at a reachable distance from the screen. In addition, the sheets need to changed often as it gets dirty with stains, importantly, this is not a hygienic solution. Hence, an ability to work on the computer while continuing to perform the surgery would be an efficient and ideal solution.

Another example, would be a driver being able to perform basic operations on a map, or control a car's dashboard without taking her hands off of the steering wheel and eyes off of the road. In modern day cars, we observe that the drivers constantly interact with touch enabled displays, while driving, and having only a single hand on the steering wheel. Similarly, musicians, soldiers, factory workers, or a person holding objects in the hand do come across scenarios where hands-free interactions are essential. However, the lack of hands-free interactions (and devices) result in poor interactions that either consume time or are inefficient.

While we discussed the needs for hands-free interactions in the context of situational impairments. On the other hand, there are users with permanent disability or impairment that need hands free, accessible interactions. Physically illiterate is a notion that indicates that a person with a disability faces challenges in acquiring mature movement patterns similar to their able-bodied peers [2, 3]. According to the 2016 disability status report (published in 2018) 12.8% of the people in the United States have at least one kind of disability [4].

Among the six major categories of disabilities, **7.1% of the United States population have an ambulatory disability**, and it is the most common disability observed [4]. Visual disability at 2.4% is the least common disability among the United States population. In addition, ambulatory disability is the major disability encountered by individuals in the age range of 21 to 64 at a rate of 5.4%, which restricts the ability of the individuals to work leading to unemployment due to disability. One in every 190 Americans lives with a lost limb, and about 80% of the amputations are due to vascular disease like diabetes, weight gain, or cardiovascular issues. Also, it is projected

that the number of people living with the loss of a limb will be nearly 3.6 million by 2050 compared to 1.6 million in 2005 [5]. Generally, ambulatory or motor disability is due to a physical impairment either by birth, injury, dysvascular amputation, or due to various medical conditions like a stroke, stiff or shaky hands etc. A major consequence of a disability **that restricts the mature movement patterns of an individual is their inability to work on computers using conventional input devices**.

Currently, to enable users with a physical impairment to work on computers prosthetic and various accessible input devices are developed. Though prosthetic technology is rapidly developing, most of these state-of-art technologies are neither attainable, nor well suited for day-to-day life [6]. Some of the prosthetics are body-powered and some are electronic devices powered by a battery, and both kinds of prosthetics are nowhere close to mimicking human functions yet [7]. Hence, using prosthetics to control conventional input devices like a mouse and keyboard has not been successful.

Similarly, alternative input devices are developed such that these devices leverage a person's unique abilities to function. Some of these devices include access switches, ergonomic keyboards, head mouse, breath/mouth control, large touch surfaces, hand stabilizers, speech to text, and so on. While each solution addresses a specific type of the impairment, the majority of these solutions become invasive with the severity of the motor impairments [8, 9, 10]. The accessible technologies are developed as separate interfaces or interaction methods, but are projected as equal technologies, which is not true. In addition, the use of a different input device will force a user to make use of different interfaces specifically customized for new input methods. Hence, users with a disability do not have the same experience as the able-bodied users when working on a computer. This calls for bridging the gap in input technologies where all users have a common experience while also accommodating their needs. Hence, this research focuses on developing gaze-assisted multimodal input methods that a) supports interactions in the scenarios of situational impairments, and b) enable users with physical impairment use the same interfaces and have the same experience as the able-bodied users when working on a computer. Our solutions are not just restricted to

supporting hands-free accessible solutions, but they also solve various interaction challenges, e.g., shoulder-surfing resistant authentication, that were unresolved with the existing standard input methods (mouse, keyboard, and touch).

1.2 Eye Movement-based Interactions

Interactions on a computer using the conventional input modalities like keyboard, mouse, and touch still remains a primary way of interactions [11, 12]. However, as discussed in Section 1.1 they are not suitable for every context, specifically these input methods are either inefficient or unusable in the scenarios of situational impairments and disabilities. While other accessible input methods like speech-based interaction [13, 14], Brain-Computer Interfaces [15, 16, 17], using active breathing [18, 19], and so on have been explored, but they have various limitations. Using speech-to-command or speech-to-text conversion for interaction relies heavily on the accuracy of the speech recognition system, and also, the speech input may not work well in noisy environments [20, 21]. Using brain-computer interaction is sophisticated, expensive, and the existing systems not matured to a point that they are accurate and reliable [22, 23].

Gaze tracking (eye tracking) refers to tracking and measuring eye movements using an eye tracker to determine the point of gaze (target location) on a computer screen or in 3D space [24, 25]. The real-time information of the eye movements gathered through eye trackers can be used for direct manipulation of interface elements; this forms the basis of gaze-assisted interaction [24, 26]. In the context of the constraints on accessible technologies, interactions using eye movements is highly promising. Gaze points are the manifestation of visual attention [26]. An ability to leverage gaze accurately, while inducing no cognitive load, will lead to highly contextual and richer interactions. Human-Computer Interaction involves numerous application contexts and scenarios where hands-free interaction is crucial, and in these scenarios, gaze-assisted interaction well serves the user needs.

First, gaze input modality enables hands-free interactions, which are useful in scenarios where the hands are engaged in other tasks, the user is working on a large (touch enabled) display, or the user does not want to reach out to a mouse [27]. Second, when using gaze and keyboard inputs

together there is no need to switch the hand to a mouse for pointing tasks unless precise pointing is required, hence the interactions are quicker [28]. Third, users with disability and impairments primarily rely on gaze-assisted interaction for communication [26]. For example, users with ALS, cerebral palsy, and so on use gaze-assisted interaction to talk to their caretakers, doctors, and so on. Similarly, users with impairments in hand, arm, lower back, and so on, find it difficult to use the conventional mouse- and keyboard-based interaction methods [29], and use gaze typing for text entry on a computer.

Lastly, in addition to being an accessible, hands-free input, gaze interaction enables novel solutions to various interaction problems, which were remained unresolved with the existing input methods. For example, using gaze-assisted interaction shoulder-surfing issues with authentication can be resolved [30], users can be liberated from remembering a complex set of shortcuts that vary across applications [31], gaze input can be used to control a wheelchair [32, 33], home automation systems, and even drones [34]. With improved gaze tracking accuracy and increased affordability of this technology, gaze-assisted interaction seems promising as we explore interaction paradigms beyond the conventional mouse- and keyboard-based interaction methods.

During the last few years, researchers have explored various ways to use gaze input to achieve hands-free, accessible interactions [35, 11]. Some of the notable works include [36, 37, 38]. Besides enabling hands-free, accessible interactions, gaze-assisted interaction has an inherent advantage because of the way humans interact with user interface elements on a computer screen. For example, to click a button, we first look at it, move the cursor from its current location onto the button, and finally click it. However, what if we could activate (click) the target, the first time the user looks at it. This avoids a) switching the hand between the keyboard and mouse, and b) a lot of mouse movements on a big screen or with multi-monitors. Hence, gaze-assisted interaction is particularly suitable for applications that make limited use of keyboard input but rely mainly on the mouse input. For example, web browsing, geo map interaction, reading tasks, and surveillance, etc. In recent years, extensive research has been directed toward leveraging gaze as the primary input modality. Gaze typing [39], gaze interaction in augmented reality [40], gaze-assisted reading

[41], gaze enhanced speech recognition [42], gaze to control drones [34], gaze assisted patient interaction [43], gaze based virtual task predictor [44], and gaze contingent computer games [45], etc., are few of the recent examples.

This extensive research in eye gaze-based interactions strongly supports the potential of using gaze-based input methods for hands-free interactions. However, these works also discuss various limitations with gaze-assisted interactions that limit the usability. Our research aims to address these limitations by combining gaze with other input modalities and also re-contextualizing gaze input as gestural input. We will be discussing the limitations of the existing systems and how we overcome those limitations in the future sections.

1.3 Introduction to Eye Tracking and the Physiology of Eye Movements

1.3.1 Human Visual System and the Anatomy of Eye

Human Visual System is composed of the eyes, visual pathways that connect retina to various regions of the brain involved in visual functions, and the connects ions between the brain regions, called streams, related to vision [26, 46]. The superior colliculus region of the brain is responsible for both saccadic and smooth pursuit eye movements, discussed in the further Section [26, 47]. The primary visual cortex is responsible for detection of the range of visual stimuli [48]. The visual information from the environment is captures through the eyes, carried through visual pathways to various regions of the brain. These brain regions further process the stimuli and generate a corresponding response. This way the eye functions as an input unit into the human visual system [26]. As an output unit, human eye reacts with mostly involuntary responses to the kind of stimuli which helps to understand cognitive state of the user through eye movement analytics. Figure 1.1 shows the anatomy of human eye [1].

Majority of the eye tracking systems function based on the corneal reflection. Cornea is the first layer through which the light passes, and traces its path on to the retina at the back of the eyeball [24]. Cornea is responsible for 2/3 of the refracting power of the human eyes, and is responsible for overall protection of the eyeball [49]. After the light is incident on the cornea, it

[1] wikimedia.org/wiki/File:Schematic_diagram_of_the_human_eye_en-edit.png

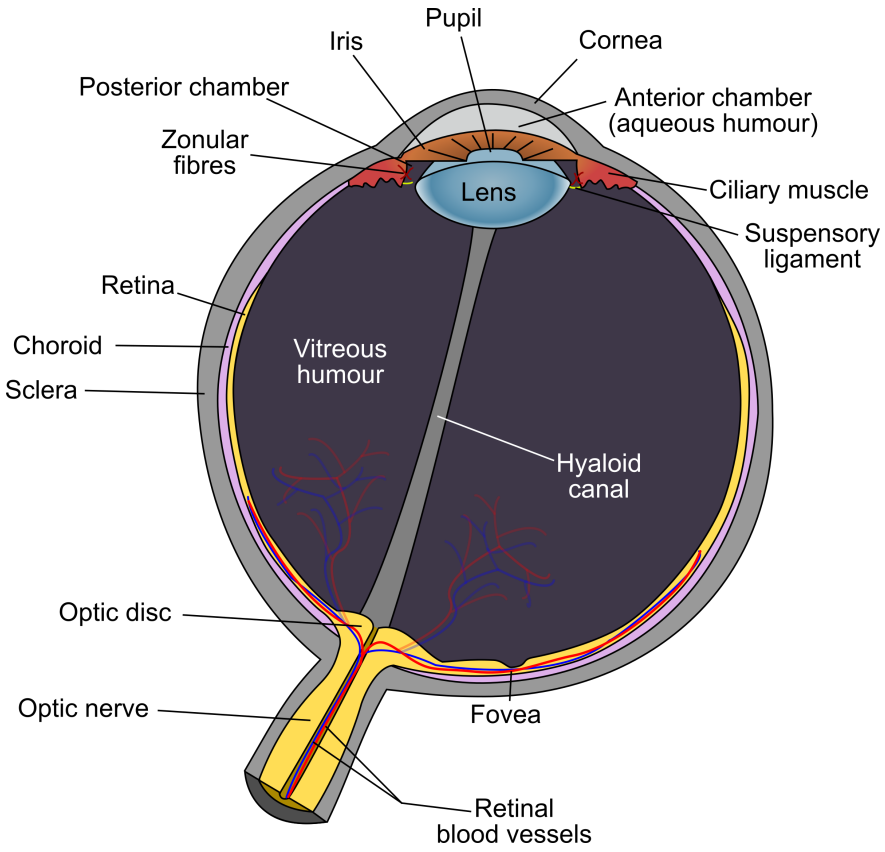

Figure 1.1: Anatomy of the human eye showing cornea, iris, and pupil, which are of interest for gaze tracking [Image: Wikimedia Commons, 2007].

then passes through the gap in the center of the iris (pupil). Iris regulates the amount of the light that enters the eye through contraction and expansion similar to an aperture setting in the camera. The opening, or aperture, in the center of the iris is called pupil, and its size varies from 1 to 8 mm. After passing through the pupil, light enters the lens that changes in its shape to focus on objects at varying distances [50]. This is called the accommodation phenomena. Figure 1.2 demonstrates how the lens changes its shaped to focus on objects at different distances [2].

While retina is responsible for 2/3 of the refractive property of the eye, lens accounts for 1/3 of the refractive property. The lens focuses the light exactly on the retina at the back of the eye [24] The photoreceptors, that are sensitive to light, present on the Retina converts light energy into electrical impulses creating the first stage of visual perception. Photoreceptors are further classified

[2] wikipedia.org/wiki/File:Focus_in_an_eye.svg

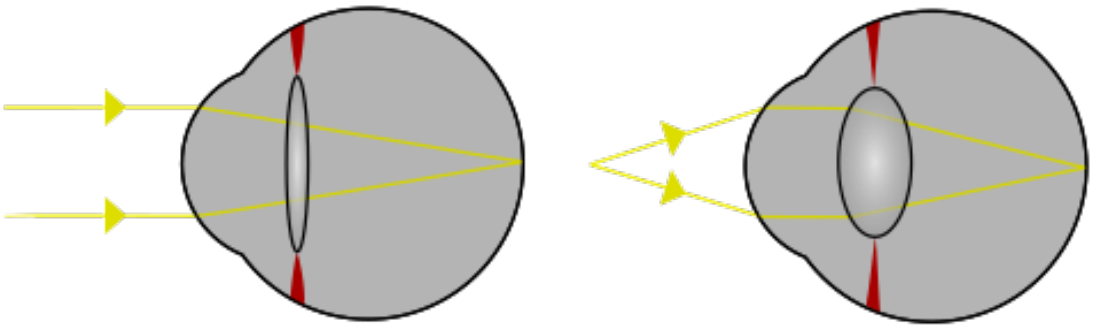

Figure 1.2: The accommodation phenomena in human eye: left - far away object, hence thin lens, right - nearer object, hence thick lens [Image: Wikimedia Commons, 2007].

into rods and cones. Rods perceive dim and achromatic light (night vision), and cones perceive brighter chromatic light (daylight vision) [26]. The central region of the retina on to which the light is focused by the lens is called fovea, this area contains most of the cone cells. The fovea varies in its size from 1 to 1.5 mm. Figure 1.3 represents the overall process of how light incidents on retina and in specific the cones and rods [3]. Figure 1.4 shows the distribution of cones and rods in and around fovea [4].

1.3.2 Types of Eye Movements

There are six muscles attached to the outer side of each eye, and these muscles functioning together helps to move the eyeball in six degrees of freedom, three translations and three rotations [51, 24] The six muscles responsible for these movements are the medial and lateral recti (sideways movements), the superior and inferior recti (up/down movements), and the superior and inferior obliques (twist) [51]. Figure 1.5 and Figure 1.6 show the lateral and anterior view of these muscles attached to the eyeball [5].

The positional eye movements can be classified as basic five types: saccadic, smooth pursuit, vergence, vestibular, and nystagmus. The non-positional eye movements can be classified as adaptation (pupil dilation and constriction) and accommodation (lens focusing).

[3] askabiologist.asu.edu/rods-and-cones
[4] commons.wikimedia.org/wiki/File:Human_photoreceptor_distribution.svg
[5] commons.wikimedia.org/wiki/File:1412_Extraocular_Muscles.jpg

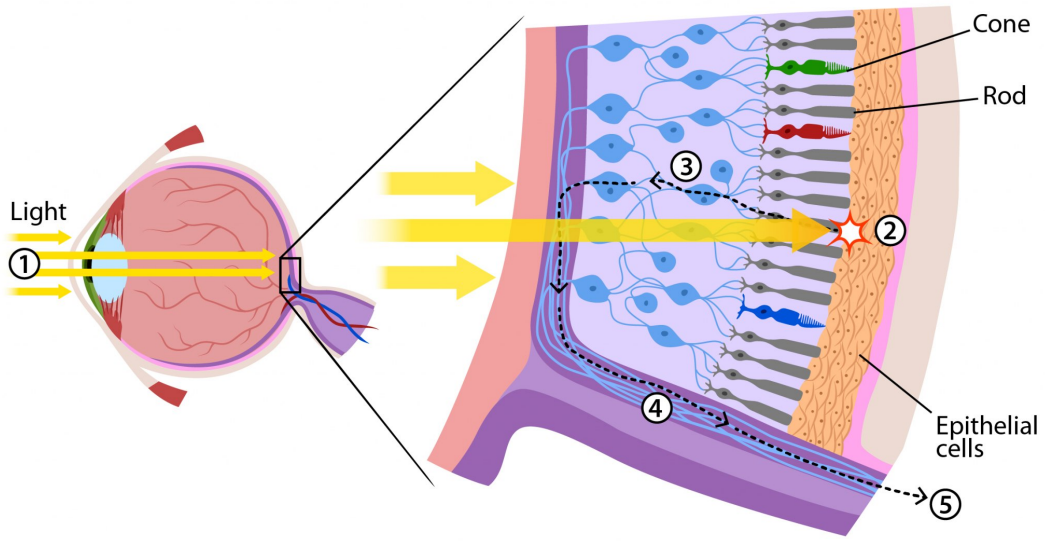

Figure 1.3: Light that is incident on retina is converted into electrical impulses by photoreceptors, also known as the cones and rods [Image: Wikimedia Commons - askabiologist.asu.edu].

1.3.2.1 Saccades

Saccadic eye movements are also known as orienting 'jumps' which direct the eyes towards the intended object/target or attention eliciting events [24]. Saccades internally re-positions the fovea to a new point in the visual environment. Saccadic eye movements are either voluntary or reflexive for a stimulus, and their duration range from 10 ms to 100 ms. The underlying neural mechanism of saccadic movement considers saccades as both stereotyped and ballistic [52, 53]. Stereotyped model of saccades states that the specific eye movement patterns are evoked repeatedly. However, ballistic eye movement states that the saccades are pre-programmed, i.e., 200 ms before a saccade the target location is fixed and the saccade cannot be altered [26]. Saccades play a crucial role in gaze tracking as it represents changes in an individual's visual attention for a given stimuli.

1.3.2.2 Fixations

Fixational eye movements allow the visual system to further process the visual stimulus. Fixations stabilize the retina on an object that is already focused, and the duration of fixation ranges between 150 – 600 ms [54]. In terms of visual angle, when fixating on an object, the movement

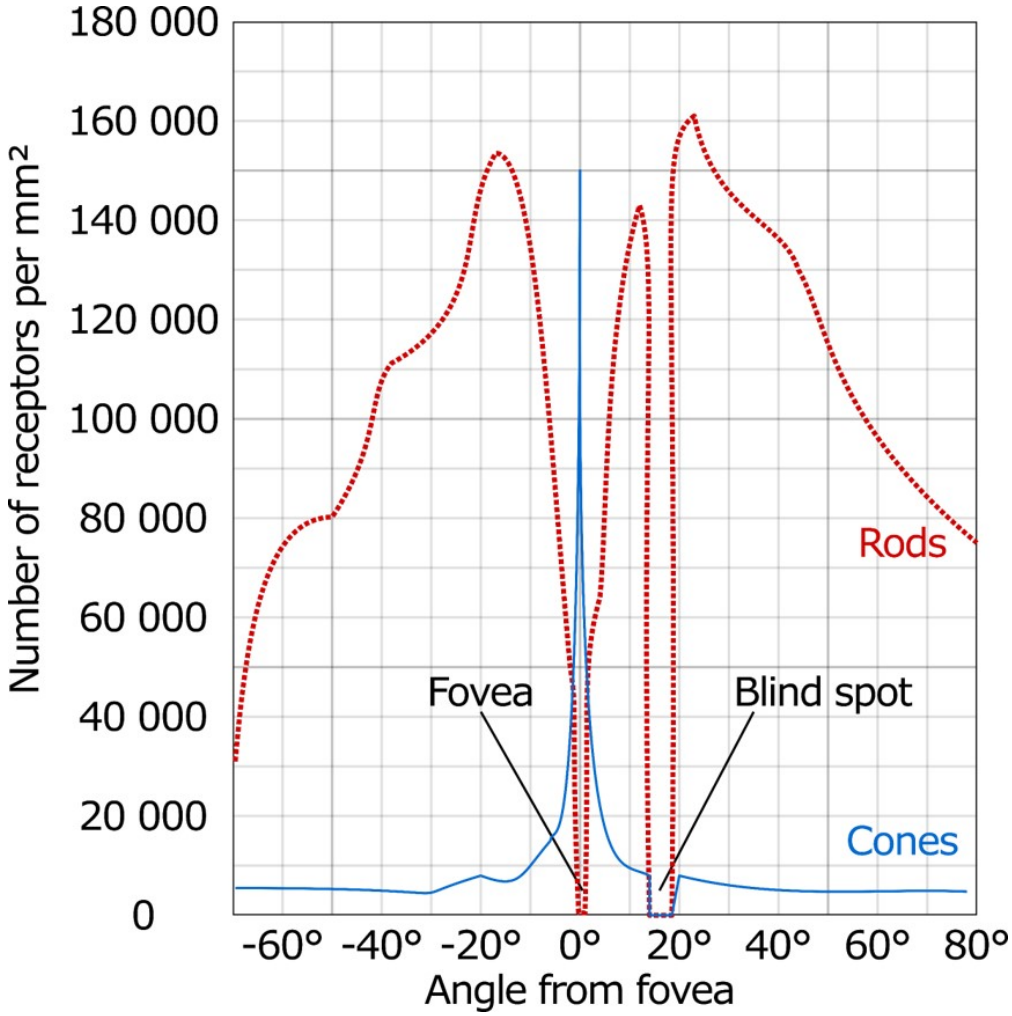

Figure 1.4: Distribution of cones and rods in and around fovea [Image: Wikimedia Commons, 2013].

rate is 15 deg/ms to 100 deg/ms [55]. While a fixation captures 25,000 square degrees of the visual world (180 deg horizontally, 130 deg vertically), only 3 to 4 square degrees falls on the fovea. The visual world captured on fovea is seen in great details with highest resolution, and eyes spend about 90% fixationing [56]. A fixation is the result of three types of eye movements: tremor, drift, and microsaccades [26]. Fixations appear as random fluctuations of the gaze around the target area at a distance not greater than 5 deg of visual angle [52]. Considering fixations as constrained eye movement within a limited period of time, or fluctuations around the target object within a specific range of visual angle provides us multiple ways to recognize fixations. Fixations serve both as an

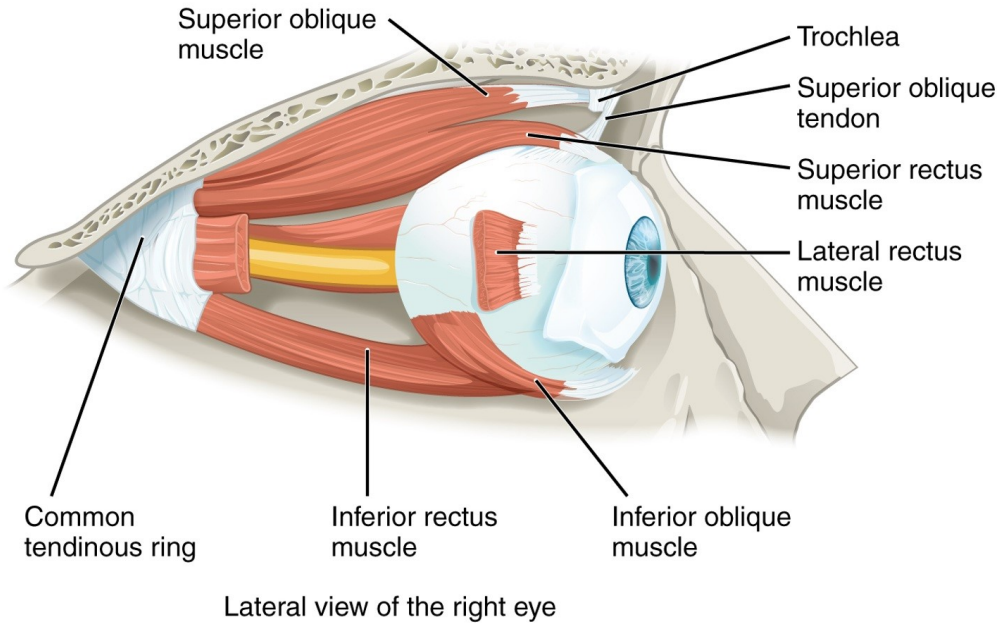

Figure 1.5: A lateral view of the eye muscles that are responsible for various types of eye movements [Image: Wikimedia Commons, 2013].

output signal or an input modality in Human-Computer Interaction. When considered as an output signal, fixations represents the visual attention, i.e., points of interest of the user on a scene. As an input modality, fixations with a pre-defined dwell time is used as an activation trigger, e.g., to click a button the user focuses on the button for 500 ms.

1.3.2.3 Smooth Pursuits

The movement of the eyes that enable the eyes to be focused on a moving target by matching eye movement to the speed and direction of the moving object/stimulus is called Smooth Pursuit [26, 24, 52]. Smooth pursuit or the lack of it reflect the cognitive state and the control of an individual over the visual system [52, 57].

1.3.2.4 Nystagmus

Nystagmus is voluntary wobbling or shaking of the eyes, these eye movements can be conjugate or dis-conjugate. In optokinetic nystagmus, saccades are interspersed to compensate for retinal movement of the eyes [58]. However, in vestibular nystagmus, saccades are interspersed to

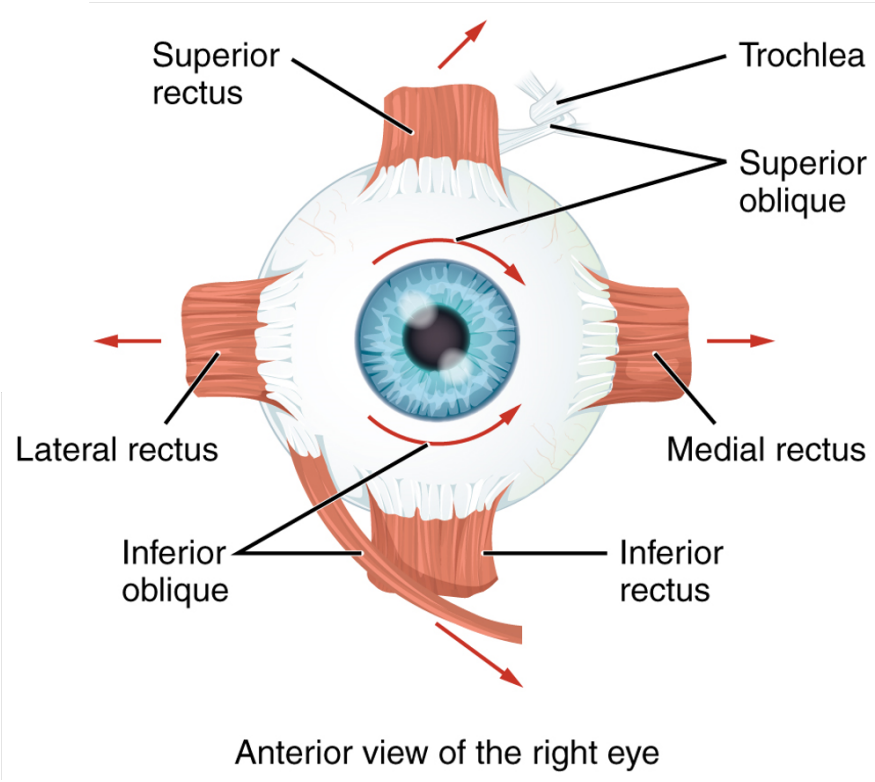

Figure 1.6: An anterior view of the eye muscles that are responsible for various types of eye movements [Image: Wikimedia Commons, 2013].

compensate for movements of the head [59]. Generally, Nystagmus poses a challenge to accurate estimation of the gaze.

1.3.2.5 Pupillary Movements

The changes in the pupil diameter, either contractions or dilation, represent pupillary movements. Pupillary movements are non-spatial eye movements that generally reflect the cognitive load on the user [60]. Changes in the pupil diameter are also caused by external sensory stimulus like light, touch, and sound. Two types of muscles in the iris control the pupil contractions and dilation. The sphincter muscles cause pupil contractions and dilator muscles cause dilation, i.e., an increase in the size of the pupil [61]. Pupil contractions are controlled by brain activity based on the stimulus. Pupil automatically contracts more as the object becomes brighter, i.e., more light entering the eye. Pupil dilation is generally in response to an outer stimulus, or internal metal load

(emotions, attention, and mental processes). In summary, pupillay movements offer a non-invasive way of understanding human cognitive processes.

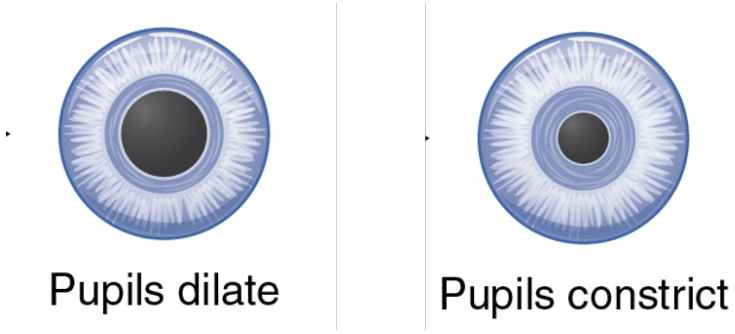

Figure 1.7: Pupillary Eye Movements: dilation and constriction

1.3.3 Calibration and Gaze Estimation

The eye tracking technology used today is based on processing the digital video stream of a user's face and eye in real-time. A typical eye tracking setup on a desktop computer is shown in Figure 1.8.

The gaze tracking system consists of mono or stereo digital cameras, near infra-red (NIR) light source, and a computer screen rendering the user interface. The video-based gaze tracking the entire process involves a) positioning the user in front of the screen, b) calibration, c) gaze estimation, d) continuous gaze tracking by capturing video frames of the face and eye regions [55]. Estimation of gaze requires, detecting the eyes and estimating their orientation, and this is achieved through detecting pupil and corneal reflections. The Pupil Center Corneal Reflection (PCCR) method uses the LEDs to produce glints on the corneal surface, and recognizes pupil center and reflection/glint on the cornea [6] as shown in Figure 1.9.

The curvature of the eye produces four reflections also known as Purkinje images, where the first Purkinje image shows the maximum/brightest reflection [7] (Figure 1.10). The gaze estima-

[6] wikiwand.com/en/Eye$_t$*racking*
[7] wikiwand.com/en/Purkinje_images

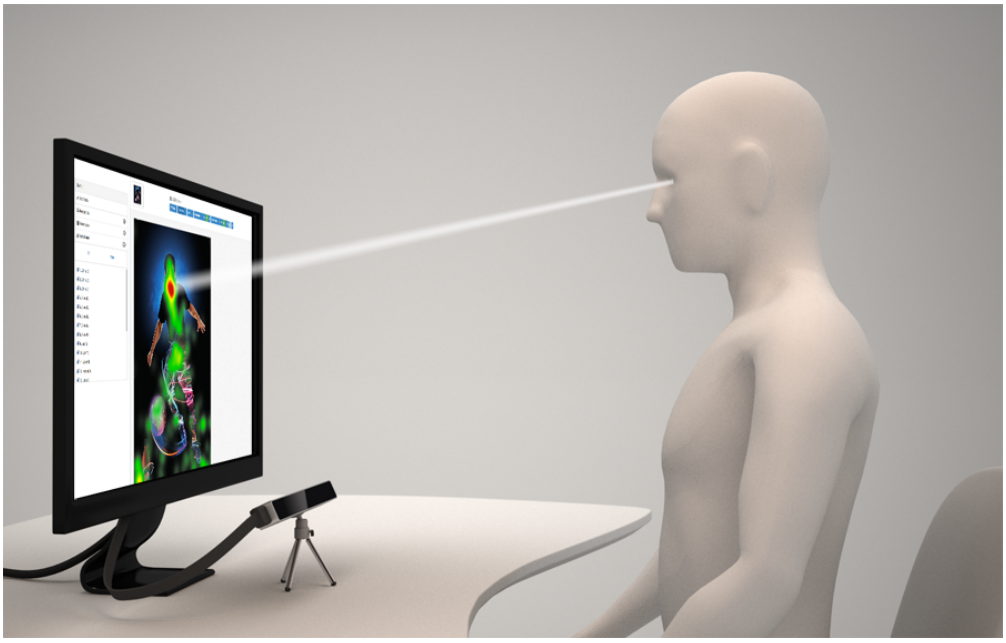

Figure 1.8: A standard setup of the table-mounted eye tracking system: the user sits in front of the screen while facing it, and the eye tracker is placed in front of the screen, facing at the user such that the user's face is within the tracking box of the eye tracker. This distance is generally 45 cm to 75 cm [Image: theeyetribe.com].

tion is achieved by computing the distance between the pupil center and the reflection point by performing linear mapping [62].

During the calibration phase, the user looks at n = 3, 5, 9, 13 target points on the screen (Figure 1.11), and the position of corneal reflection changes with respect to the pupil as shown in Figure 1.12. Following the calibration the gaze estimation is achieved by one of the three methods: 2D regression, 3D model, and Cross ratio based method [55].

The last phase of gaze tracking is to compute the gaze estimation error for each user by first computing the On Screen Distance (OSD) as shown in Equation 1.1. Then the gaze angle is computed as shown in Equation 1.2, and lastly, the mean gaze tracking error is computed as shown in Equation 1.3.

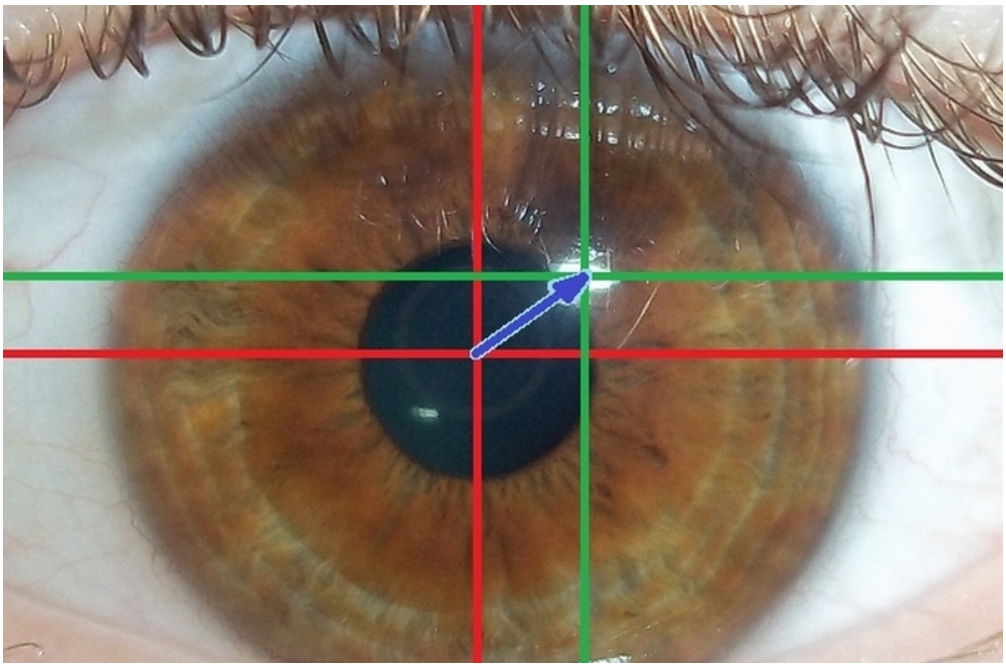

Figure 1.9: Corneal reflection (glint) due to an illumination from a near infra-red light source. To estimate the gaze point on the screen, the vector between the pupil center and the reflection point is calculated and then a linear mapping is applied [Image: Wikimedia Commons, 2015].

$$OSD = pixelsize \times \sqrt{(POG.X - \frac{X_{pixels}}{2}))^2 + (Y_{pixels} - POG.Y + \frac{offset}{pixelsize})^2} \tag{1.1}$$

$$Gaze\,angle\,(\theta) = \tan^{-1}(\frac{OSD}{dist}) \tag{1.2}$$

$$Ang\_Accuracy = \frac{pixelsize \times Pix\_acc \times \cos\left(mean\left(\theta\right)\right)^2}{mean\_dist} \tag{1.3}$$

1.4 Gaze-assisted Interactions: Point-and-click, Text entry, and Authentication

When developing gaze-assisted interactions for situational impairments and accessibility, from the existing literature, the three primary interactions considered are point-and-click [63, 64, 65, 66,

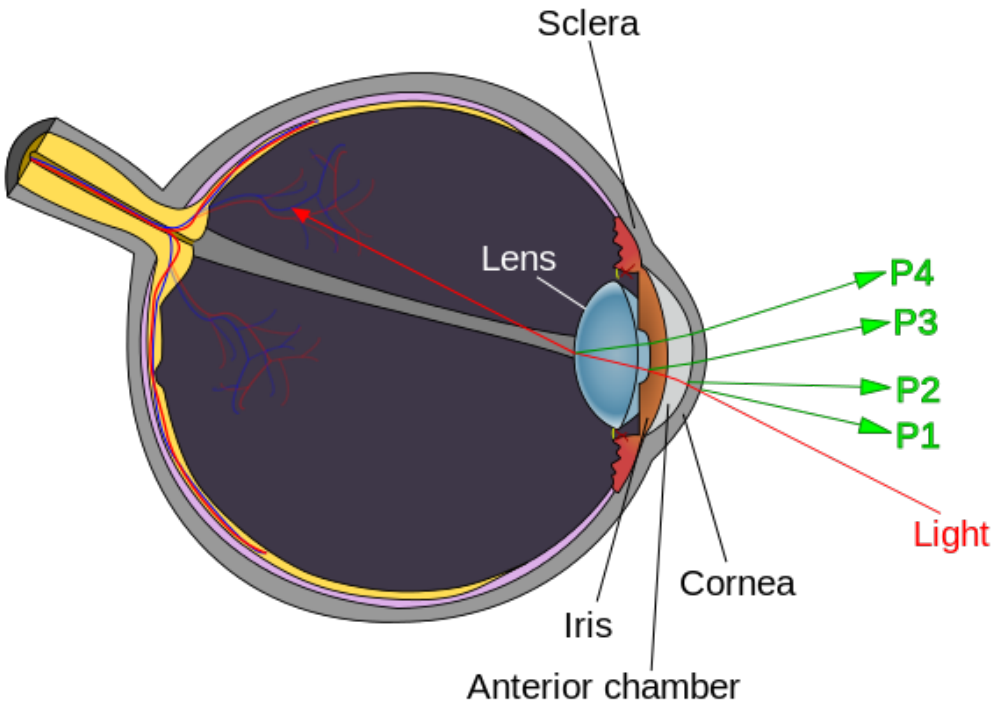

Figure 1.10: The four Purkinje images formed due to an illumination from an NIR light source. The first Purkinje image is considered for gaze estimation [Image: Wikimedia Commons, 2015].

67, 40, 68, 69, 70, 71], text entry [72, 73, 74, 75, 76], and authentication [37, 77, 78, 30, 79, 80]. In addition to these three primary interactions, gaze input has also been used to control drones [34], navigate robots [81], play music [82], and so on. These works demonstrate the a great number of possibilities with gaze-assisted interactions that are either unlikely or inefficient to achieve with other input methods.

First, the point-and-click interactions are the primary way of working with interface elements on the screen. Mouse and touch input modalities are primarily used for these interactions. Hence, an inability to use mouse or touch inputs poses a serious challenge to interacting with computing devices. As discussed, prior research works have investigated the usability of gaze input in these scenarios. In this regard, two types of interactive applications have been proposed: gaze selective applications, and gaze contingent applications [26]. In the case of gaze contingent applications, the

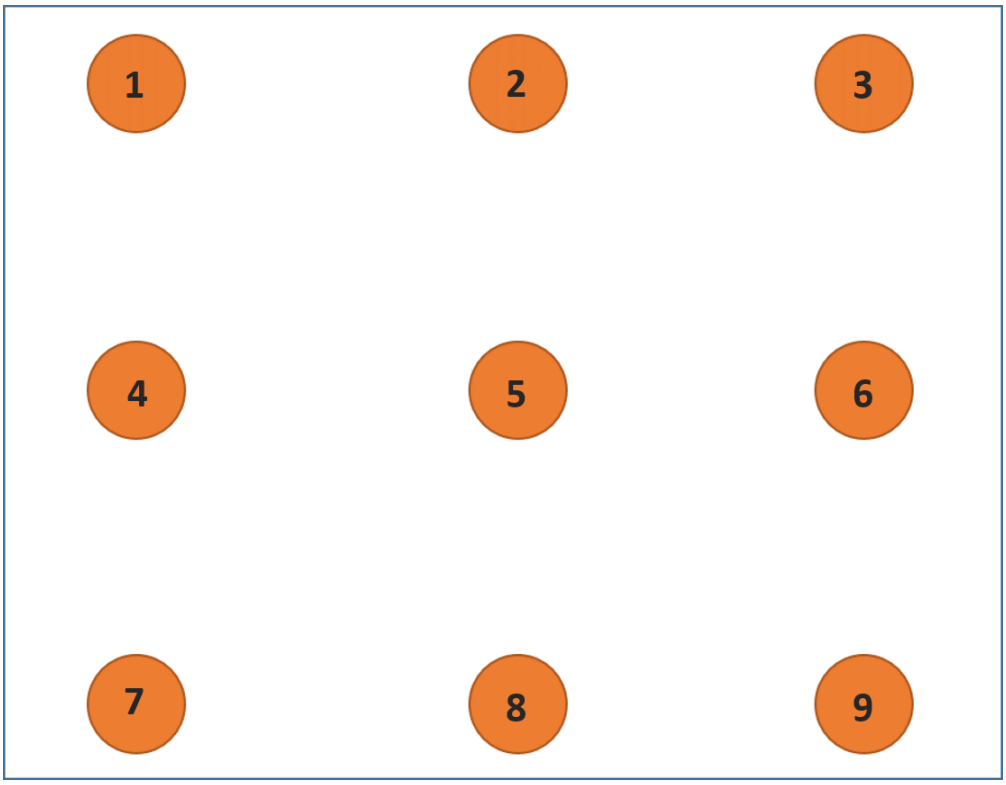

Figure 1.11: A standard 9-point calibration grid used for gaze estimation.

information presented to the user is manipulated based on where the user is currently looking on the screen. For example, virtual reality applications use the principle of gaze contingent information display to achieve foveated rendering [83, 84]. This significantly reduces the need for computing resources. Gaze selective applications use gaze input as a replacement for the mouse. Additionally, if the targets are sufficiently large, gaze pointing is much faster than the mouse pointing [85, 86]. In a few scenarios, gaze input has been used not for direct pointing, but to improve the speed of pointing interactions. For example, gaze input can be used to wrap the cursor on the target the user is looking at [87], or switch between multiple application windows [88, 89]. However using eyes as a mouse would lead to “Midas Touch" problem [65], and there are dwell-based and blink-based solutions proposed to counter the Midas Touch problem. We will further discuss the Midas Touch problem, existing solutions, our solution, and how our solution compares to the mouse input in Section 2.

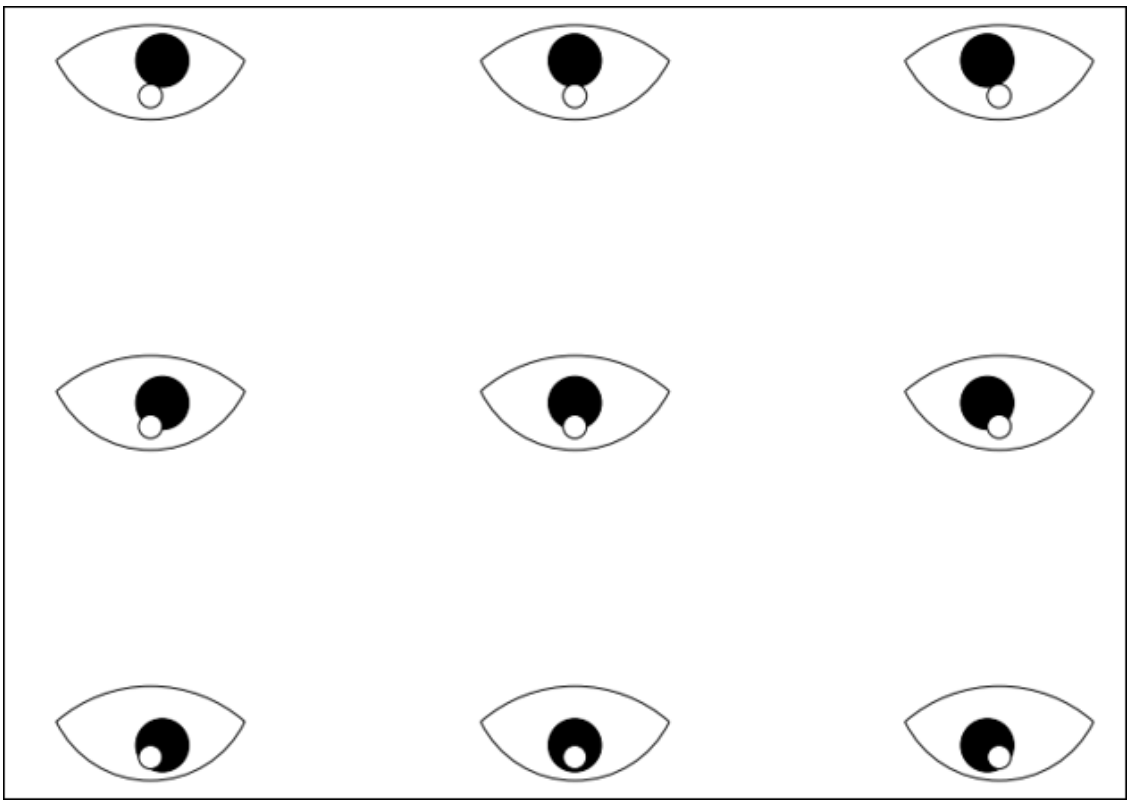

Figure 1.12: The position of the glint on cornea and the relative distance between pupil center and glint when the user looks at 9 calibration points [Image: Duchowski, 2007].

Second, individuals that have experienced serious accidents, diseases leading to imputations or instability of the limbs, brain-stem stroke, disability by birth, and so on primarily rely on text entry by gaze as their primary way of communication [24]. In fact, the first interactive applications of gaze were actually text entry systems where the user could focus on the desired character to enter it [90, 91, 92, 93]. Due to the low accuracy of gaze tracking, these early systems expected the user interface elements to be sufficiently large and fewer in number. Today, there are multiple ways of text entry by gaze based on if the user has the ability to fixate on the target or not. These methods include gaze typing, discrete gaze gestures, continuous pointing gestures, and eye switches [24]. In addition to proposing various gaze-based text entry methods, multiple ways of target selection have been explored. These methods include various dwell times [94], blink [95, 96], head-pointing [75], breathing, tooth-clicks [97], gaze-based multi-modal input [94], and so on. Furthermore, to im-

prove the speed and accuracy of typing current and future word prediction systems have also been developed [98, 99]. Further details regarding the various gaze-assisted text entry methods, key selection strategies, and typing evaluation metrics are discussed in Chapter 4.

Lastly, the gaze-assisted authentication was developed to serve two purposes 1) an accessible hands-free method of authentication, and 2) to counter shoulder-surfing attacks. As an accessible authentication method, user can enter their credentials without being required to use any input devices like a mouse, keyboard, or touch surface [30, 100]. On the other hand, gaze-assisted authentication tries to prevent shoulder surfacing attacks through casual observations [77, 101], or advanced attacks like using heat signatures and iterative video analysis attacks [102, 103, 104, 100]. Some of the authentication methods use alphanumeric passwords that are entered on a virtual keyboard through gaze typing [37]. Others use a completely different strategy like gaze gestures [78], following moving shapes [30], and so on. Further details regarding gaze-assisted authentication strategies, their accuracy, and vulnerability to video analysis attacks are discussed in Chapter 5.

1.5 Proposed Solutions

Despite the advantages of gaze-assisted interaction, there are still various limitations to using gaze-assisted interaction. One of the significant issues is the Midas Touch problem, which states that if eye position is used as an input modality, while directly substituting a mouse, with an expectation that user be able to simply look at what she wants and have it happen, everywhere a user looks a command gets activated and this is annoying [65]. Hence, a command (e.g., click) can not be executed wherever a user looks on the screen, this is because, a user first scans the display before fixating on the point of regard to execute an action [65]. Therefore, any gaze controlled interface that can not distinguish between an intentional and unintentional visual focus is hardly suitable for practical applications.

Existing solutions which at least support point-and-click interactions use either of the two main methods to address the Midas Touch issue: 1) eye blink and 2) dwell time. In dwell-based activation, to execute a command, like a click, on the interface element the user fixates on the point of regard for a predefine interval of time (150–200 milliseconds) [65]. Systems like [105, 67, 66,

40] use dwell time, with varying thresholds, as the target object selection method. Relatively a few implementations use an eye blink as a target activation method. In such systems, a user's intent to execute a command (click) on the target object (UI element) is achieved by blinking the eye. Some of the implementations that use an eye blink for target activation include [63, 64], and so on. Both dwell- and blink-based activations are limited by accuracy, speed, and inability for a prolonged usage.

We believe, an effective solution where gaze is used to perform mouse-like interactions, should use gaze input only for pointing, but a supplemental input should be used to execute commands at the point of regard. Hence, in this dissertation work, we present a multi-modal interaction paradigm that effectively combines gaze and foot input modalities to achieve point-and-click interactions on a computer. We discuss two systems, GAWSCHI and gaze typing, that utilize the gaze and foot interaction paradigm. Furthermore, we contextualize gaze input as gestural input to create gaze gesture-based interactions. In this method, we use sketch recognition principles and pattern matching algorithms to translate gaze input to gestural input and recognize the gesture. We discuss two systems, a gaze authentication system and a gaze gesture toolkit, that utilize gaze gesture-based interaction paradigms.

1.5.1 Gaze and Foot Input Framework

The basic principle behind gaze- and foot-based interaction framework is to achieve co-ordination between gaze and foot input to point-and-click precisely on the interface elements. In this method, we use an eye tracker that tracks a user's gaze on the screen and a wearable device that is operated with the user's foot. To click on the point of regard (POR) like a button, the user first looks at the POR and presses the pressure pad, with the foot, attached to the wearable device. The foot input is translated to mouse clicks that are executed on the interface element (POR) that the user is currently looking at.

1.5.1.1 GAWSCHI

GAWSCHI is a Gaze-Augmented, Wearable-Supplemented Computer-Human Interaction system that combines both gaze and foot input to effectively address the Midas Touch problem. GAWSCHI enables quick and accurate gaze-assisted interactions in a desktop environment, while being completely immersive and hands free. Interactions like point-and-click, double-click, click-and-hold, and hold-and-release are supported by the system. Foot-interaction is achieved through a quasi-mouse that has a small form factor, which allows it to be mounted on a user's footwear (shoe) or placed anywhere on the floor. The user executes a command by looking at an interface element (point of regard) and pressing with the foot, the pressure pad attached to quasi-mouse.

1.5.1.2 Gaze Typing

We present a dwell-free gaze typing system that comprises of a custom built virtual keyboard and a footwear augmented with pressure sensors or foot gesture recognition sensors. We present our findings from multiple usability studies and design iterations, through which we created appropriate keyboard layouts to support foot-interaction. We also present findings from a comparative study that discuss various gaze typing metrics when using dwell and dwell-free activation methods like foot press-based activation or foot gesture-based activation. We also discuss the advantages of dwell-free activation achieved by the separation of gazing and fixating through gazing and foot-interaction paradigm.

1.5.2 Gaze Gesture Framework

In this approach we contextualize gaze input as gestural input and each such gesture represents a unit of information based on the application context. For example, a gaze gesture-based authentication system will use multiple gestures as a password, or in a gaze gesture-based interaction method, each gesture can represent commands like window minimize, maximize, play, pause, etc. Generally, a gaze gesture consists of a series of fixation points connected with saccadic eye movements creating a scan-path (gaze-path). Such a scan-path can be further processed using sketch recognition and pattern matching algorithms for specific use cases. Based on these principles, we

have developed two systems that leverage gaze gestures: 1) an authentication system, and 2) gaze gesture-based interaction paradigm.

1.5.2.1 Gaze-assisted Authentication

The need for active involvement of the user to authenticate may not be suitable in various scenarios like situational and physical impairments. While the need for accessible and hands free authentication method is crucial, the existing standard (PIN or graphical) authentication methods suffer from shoulder surfing attacks. Shoulder-surfing is the act of spying on an authorized user of a computer system with the malicious intent of gaining unauthorized access. We present a gaze gesture-based, accessible, shoulder-surfing resistant authentication system that uses graphical passwords to authenticate a user. We show that our system is highly accurate, robust to calibration errors, and less susceptible to video analysis attacks.

1.5.2.2 Gaze Gesture Toolkit

Though gaze-assisted multi-modal systems enable accessible interactions, individuals with complete disability cannot use multi-modal interactions [86]. Such individuals could only move their eyes to express their active engagement [24]. Recent works have demonstrated the potential of gestures performed with eyes–gaze gestures–to interact with computer applications or for text entry [106, 107, 108]. We present a gaze gesture-based interaction toolkit, where a user can interact with a wide range of applications using a common set of gaze gestures. In our system, a gaze gesture can minimize, maximize, restore, or close an active window, or it can create a new tab, scroll, refresh, and so on a browser. To execute an action like minimizing a window, the user performs a pre-defined gesture on the window with their eyes. This framework enables a user to build a universal set of gestures that can be executed on any application, and the user need not remember application specific shortcuts. Additionally, we also present a gesture training toolkit, where a user can create a new gesture and associate specific actions to it. Lastly, we demonstrate the usability of the of the gaze gesture framework on a browser.

2. GAZE-ASSISTED POINT-AND-CLICK INTERACTIONS

Recent developments in eye tracking technology are paving the way for gaze-assisted interaction as the primary interaction modality. In the Introduction (Chapter 1) we discussed the two categories of gaze-based applications which are gaze selective and gaze contingent applications. In case of gaze selective applications, gaze input is used as a replacement for the mouse to achieve point-and-click operations [26]. Also, we briefly discussed the Midas Touch problem arising from using gaze as a replacement for the mouse, and how dwelling and blinking can address the Midas Touch issue. Unfortunately, despite successful efforts, existing solutions to the Midas Touch problem have two inherent issues: 1) lower accuracy, and 2) visual fatigue that are yet to be addressed. To achieve efficient point-and-click interactions while addressing the Midas Touch issue, we have developed GAWSCHI: a Gaze-Augmented, Wearable-Supplemented Computer-Human Interaction framework[1]. "GAWSCHI" enables accurate and quick gaze-assisted interactions, while being completely immersive and hands-free. The system uses an eye tracker and a wearable device (quasi-mouse) that is operated with the user's foot, specifically the big toe. GAWSCHI is the first system to seamlessly integrate gaze-assisted interactions on the native interface of an operating system to support common tasks on a computer. The system was evaluated with a comparative user study involving 30 participants, with each participant performing eleven predefined interaction tasks (on MS Windows 10) using both mouse and gaze-assisted interactions. We found that gaze-assisted interaction using GAWSCHI is as good (speed of task execution) as mouse-based interaction as long as the dimensions of the interface element are above a threshold (0.60" x 0.51"). Our study show that the performance and accuracy of the user gradually improves as the user gets more acquainted to the gaze-assisted interaction. In addition, an analysis of NASA Task Load Index post-study survey showed that the participants experienced low mental, physical, and temporal

[1] *Parts of this chapter are reprinted with permission from "GAWSCHI: Gaze-Augmented, Wearable-Supplemented Computer-Human Interaction" by Rajanna et al., 2016. Publisher and Copyright holder ACM Digital Library, 2016, New York. Conference - ETRA '16: 2016 Symposium on Eye Tracking Research and Applications Proceedings - doi.org/10.1145/2857491.2857499

demand; also achieved a high performance. We foresee GAWSCHI as the primary, accessible interaction modality for the physically challenged and an efficient, hands free interaction modality in the scenarios of situational impairments.

2.1 Introduction

Gaze-assisted interactions are a promising human-computer interaction paradigm as we move beyond the conventional mouse and keyboard based interaction methods. Interfaces driven by gaze not only enable hands-free, immersive interactions, but also are a way of communication for individuals with physical impairments, who otherwise find it difficult to use conventional interaction methods [29]. We are witnessing an increasing use of gaze input for gaze typing [39], gaze interaction in augmented reality [40], gaze-assisted reading [41], gaze enhanced speech recognition [42], gaze to control drones [34], art work evaluation [109], biometrics [110, 111], gaze-assisted patient interaction [43], gaze based virtual task predictor [44], and gaze contingent computer games [45], etc., are few of the recent examples.

Applications demanding rich interactions for efficiency, and users with accessibility needs to rely on eye tracking technology as a hands-free input modality [26]. As an accessible technology, gaze input serves as the primary mode of communication for individuals with severe motor and speech disability [112, 94]. Individuals with a disability use gaze input to point and select user interface elements on a computer screen, as well as for gaze typing using various key selection methods [24]. As gaze is increasingly used as an input modality, its applicability is not just limited to accessible technology, but there are emerging use cases such as leveraging gaze-assisted interaction in situationally-induced impairments and disabilities (SIID) [113]. In the case of SIIDs, a user's hands are assumed to be engaged in other tasks, and hence unavailable for selecting interface elements or typing by using touch, mouse, or keyboard. For example, the hands of a surgeon performing an operation, a musician playing music, a worker on a factory assembly line, and so on tend to be engaged in a specific task, and hence represent a case of SIID.

Despite the discernible advantages of the gaze-assisted interactions there is still no widespread adoption of this technology as the primary input modality. The existing solutions to the Midas

Touch problem are not effective and scalable enough to be applicable to the wider array of scenarios that we encounter daily. The Midas Touch problem states that if eye position is used as an input as a direct substitute for a mouse, with an expectation that user be able to simply look at what she wants and have it happen, everywhere a user looks a command gets activated and this is annoying [65]. A user first scans the display before fixating on the point of regard to execute an action [65]. Hence, any gaze-assisted display that can not accurately distinguish between the intentional and unintentional visual focus is hardly suitable for practical applications. Recent studies have proposed a few solutions that try to address the Midas Touch problem and open possibilities for using gaze as the primary input modality [114]. However, it is not currently feasible to seamlessly use these solutions across multiple applications.

We have developed GAWSCHI a Gaze Augmented, Wearable Supplemented, Computer Human Interaction framework that offers a precise, and effective solution to the Midas Touch problem. Unlike other solutions to the Midas Touch problem, which only rely on gaze as the input, we use a supplementary wearable device, along with the gaze input, to execute a user's intent at the point of regard. The wearable device is a quasi mouse that is controlled with user's foot, specifically the big toe. The design of the wearable allows it to be mounted on the user's footwear (shoe) or placed anywhere on the floor to enable effortless and hands-free interaction. Clicking actions like left click, double click, click and hold, and hold and release are supported by the quasi mouse. The quasi mouse communicates with the central GAWSCHI system running on the computer over Bluetooth, hence making it is mobile. GAWSCHI uses a table mounted eye tracker from The Eye Tribe for tracking the user's gaze on the screen. A central controlling system, gaze interaction server, running on the computer controls both the eye tracker and the quasi mouse. The cursor is always made visible to the user to provide a constant feedback on where the user is looking at, and what UI elements is the user interacting with. The activation of the intended commands like click, hold, drag, etc., are executed by looking at the object of interest and executing the corresponding actions through the foot controlled quasi mouse.

The efficiency and accuracy of GAWSCHI is evaluated by a comparative user study involving

30 participants. The participants performed eleven predefined daily computer tasks listed in Section 2.6, elicited from interviews and a pool of common activities performed on a computer. The participants used both gaze-assisted and mouse-based interactions to perform the eleven tasks. The results from this study show that the participants experienced no latency in moving the cursor on screen, and were able to precisely interact with the intended interface elements. The speed of task execution is at least as good as and in some cases even faster than mouse-based interaction. While participants with better gaze control showed consistently high performance throughout the study, others showed a gradual improvement as they got acquainted to the gaze-based interactions. An analysis of NASA Task Load Index post-study survey showed that the participants achieved high performance, and experienced low cognitive load, while using GAWSCHI. In addition, GASWCHI significantly reduces the need to constantly switch the hand between the mouse and keyboard. Lastly, a quantitative evaluation of the system design, showed higher ratings across various design aspects (Section 2.8) of the system.

One might argue that why GAWSCHI uses a foot-controlled quasi mouse, and why not use an explicit button, either on the keyboard, or mounted on the desk. Any use of an explicit button defeats the purpose of GAWSCHI for the following reasons:

- The primary goal of GAWSCHI is to provide a hands-free, immersive interaction, where the user is able simply lean back on the chair and perform mouse based actions.
- An explicit button will force the user to reach-out to the button, demanding hand movements, and bending action.
- While the user works on a computer the legs (specifically the feet) are immobile for most of the time, and this makes the foot an appropriate choice for controlling an external input device.
- Our study proves that a foot controlled quasi mouse that operates based on the amount of pressure applied is more convenient than a mouse (clicking).

- Lastly, the number of user actions supported by the quasi mouse is extensive as opposed to an explicit button that can support only a few actions.

Further sections of this chapter are organized as follows. Section 2.2 provides a brief background and review of research in gaze-assisted interactive systems. Section 2.3 discusses unique features of GAWSCHI by comparing it with the prior work. In Section 2.4 and 2.5 we describe system architecture and implementation. Section 2.6 covers our experimental protocol, evaluating the efficacy of GAWSCHI framework. Results from our laboratory studies are provided in Section 2.7, followed by a discussion section 2.8 that interprets the results and presents inferences.

2.2 Prior Work

Since Jacob presented his research on the potential of using eye tracking toward developing gaze based interactive systems [65], there has been significant research both in the development of accurate eye tracking systems and complementary applications that solve the "Midas Touch" problem. Implementations such as [68, 69, 71, 70], have already integrated gaze-assisted interactions on the native interface of an operating system (Windows). However, such implementations continue to use either dwell time, fixation, or magnify the point of regard for a target object selection, and hence they are not precise and do not match the speed of a mouse. Hence, an accurate implementation of the gaze-assisted interaction that integrates seamlessly with the native interface of an operating system to substitute mouse-based interactions still remains unresolved. Despite the lower accuracy levels, gaze-assisted interaction is adopted in various prototype systems that prove the feasibility of the gaze interaction modality. Such systems primarily focus on exploring the ways to solve the Midas Touch problem. The existing research toward leveraging gaze as an input modality can be classified across five broad categories.

2.2.1 Gaze and Foot-based Interaction

Foot-operated computer input devices have always been well studied among the Computer-Human Interaction community. Pearson and Weiser [115] conducted seminal work in this regard, as they presented the design of “Moles," foot-operated input devices similar to the mouse. Pakka-

nen and Raisamo [116] demonstrated the feasibility of feet input in non-accurate spatial tasks. Furthermore, Velloso et al. [117], present a comprehensive survey of foot-operated interaction modality. Only Göbel et al. [118], combined gaze and foot-input as an interaction modality. However, their implementation is specifically built for pan and zoom interactions; the authors mention that foot-interaction systems support coarse pointing interaction, and are not precise enough for pointing tasks.

2.2.2 Gaze and Blink-based Interactive Systems

Gaze based interactive systems use gaze as the input modality to move the cursor to a point of regard on the screen. In such systems, a user's intent to execute a command (click) on the target object (UI element) is achieved by blinking the eye. Adjouadi et al. [63], implemented an eye-gaze tracking system for individuals with severe motor disabilities. The system uses an eye-to-mouse pointer coordinate conversion system that uses left eye blink for a short time interval as the left mouse click action. The authors present a metric, 20 zeros, coming into stimulus computing unit as the clear distinction between an intentional blink as opposed to a normal blink. During the system evaluation the participants primarily used a web browser and navigated through the web. The authors also report that though the eye movements are faster than cursor movements, the eye tracking system is still less stable and less accurate.

Biswas et al. [64], presented a gaze based input interaction system for people with severe disabilities that uses eye tracking and single switch scanning interaction techniques. The system uses blink for the target object selection, as it can clearly classify between intentional and non-intentional eye blinks. It only recognizes intentional eye blinks through dwell time during which the user has to make another blink to select the target. During the evaluation phase participants used a smart home application for 10 minutes with no specific task defined. A comparative study with a gesture recognition system showed that the eye tracking system demands more cognitive load than the gesture-based system. However, temporal and performance scales are not significantly different between the systems indicating that participants performed equally on both the systems.

2.2.3 Gaze and Dwell-time Based Interactive Systems

Jacob in his seminal work on gaze based interactive system [65] proposed that dwell time (150–200 milliseconds) is more convenient for target object activation. Since then, many gaze based systems have adopted dwell time, with varying thresholds, as the target object selection method [105, 67, 66, 40]. Jacob et al. [105], conducted a study of interaction techniques to incorporate gaze in human computer interaction. They developed an eye tracker test bed that consists of several ships on a map, where whenever the user looks at a particular ship, details of the selected ship are displayed on the left screen. In this study, the authors used two object selection methods simultaneously: a) Explicit key press, b) Dwell time. The user has to compromise with speed if dwell time is used as the object selection method. Also, the authors proposed that a longer dwell time can be used to ensure that an inadvertent selection will not be made, however, this negates some of the speed advantage of gaze based interaction. In addition, the authors state that a minimum dwell time of 150–200 milliseconds can be used only if the selection of wrong target objects can be undone trivially. The authors revert back to the key press method in situations where target object selection can not be undone easily. The study also reports that the benefit of low cognitive load becomes attenuated if unnatural and conscious eye movements are required.

Heikkil et al. [66] presented EyeSketch, a gaze-assisted drawing application that uses drawing objects, which can be moved or resized. The application uses dwell time to select the tools and objects, and gaze gestures for moving and resizing objects. The implementation uses a dwell time of 400 ms; once the target object (button) is selected, the user is notified through a feedback sound. Since the design of the system places resize buttons close together, using dwell time to resize shapes may lead to misclicks. The authors report that eye gestures are a more suitable method for moving and resizing the objects than dwell time; also gaze gestures are less error prone and easier to use.

2.2.4 Gaze and Touch-based Interactive Systems

Gaze and touch-based interactive systems use gaze as the point of regard, or the target location to be manipulated, and combines gaze with touch input on the same display surface [119], or external touch input device [120, 121] to execute the user actions. Turner et al. [120], presented a method to manipulate objects on a multitouch surface, where a user can select the object of interest through gaze, and manipulate the object by touch input, anywhere on the screen. The advantage is that this method supports whole surface reachability and rapid context switching. The authors report the usability of the system in the context of four applications: 1) Map browsing, 2) Image gallery, 3) Multi object pinching, and 3) the MS Paint application. This implementation suffers from camera occlusion by the user's hand, and is not suitable for use with the fixed UI controls.

Turner et al. [120, 121], presented a method for moving objects between large screens and personal devices with the gaze and touch input. This interaction is achieved using three techniques eye cut & paste, eye drag & drop, and eye summon & cast. In this mode of interaction, content and destination selection is achieved through gaze, and content retrieval and publishing is achieved through pull and push gestures.

2.2.5 Gaze and Gesture-based Interactive Systems

Cha et al. [122], combined gaze with gesture inputs in multimodal interaction for multi display environments equipped with large displays toward achieving contact-less application control . An interpreter combines the data streams from both a gesture transformer and a gaze transformer, and recognizes if the user performs a gesture (pointing), while fixating at a point on the screen. In this work, the authors present no practical applications of gaze and gesture based interaction. Also, the literature on practical applications that combine gaze and hand gestures is limited.

2.2.6 Gaze and Voice-based Interactive Systems

Gaze and voice based interaction uses explicit vocal commands to confirm a user's action/intention. Elepfandt et al. [123], implemented a prototype system "Matchbox" that functions by combining gaze and voice commands. With gaze, a user selects the target object, and through voice com-

mands like drag & drop, rotate, and resize the desired action is performed on the target object. The authors present that short voice commands are better suited as opposed to longer sentences in natural dialogue. In addition, though no supporting evidence is provided, the authors state that this approach is better than dwell time in gaze based interactions.

Beelders et al. [124], integrated eye gaze and speech for typing in Microsoft Word. The authors report that the effectiveness and error rate of gaze and speech interface for typing are not affected by either the size or the spacing between the buttons. However, when comparing the performance of the same task using the keyboard, the keyboard interface is significantly ahead of the gaze and voice based interface. Even with extended exposure to the gaze and voice based interface, the performance was not better than keyboard based interface.

Hence, an accurate, gaze-assisted interaction that integrates seamlessly with the native interface of an operating system to substitute mouse, while supporting most of the common interaction tasks still remains unresolved. GAWSCHI seeks to explore these limitations to create a fully gaze and foot-based interaction modality to achieve precise point-and-click interactions, while also supporting other interactions (double-click, click-and-hold, hold-and-release). Furthermore, GAWSCHI seeks to improve the design of foot-operated devices through its wearable quasi-mouse that has a small form factor. In addition, the design of the quasi-mouse only requires a gentle press (minimal effort), with the foot, for executing the user commands (click), while still achieving a high performance.

2.3 GAWSCHI: The Unique Features

Despite a wide range of interactive systems that leverage gaze, there are still multiple aspects of gaze based interaction that are yet to be addressed. First, most of the existing gaze-assisted implementations are tested and have shown to work on the test-beds or prototype systems that the authors have created. Such systems are built with specific design considerations, such as UI elements with large dimensions and larger font sizes, to support gaze based interaction. However, some of the implementations like [68, 69, 71, 70] integrate gaze interaction with Windows operating system, but they suffer from speed, accuracy, and applicability limitations as discussed in

section 2.2.

An implementation that seamlessly integrates with the native interface of a widely used operating system, and supports most of the common tasks just through gaze does not exist. These common tasks may include: web browsing, image viewing, map navigation, reading web pages, code editors, page scrolling, interacting with an application, online video viewing, browsing folder structures, and playing games, etc. GAWSCHI is one such system that integrates seamlessly with the native interface of an operating system (Windows 10), and allows for the performance of all, but not limited to, the tasks listed above.

Second, the user actions supported by existing systems, which provide both hands free and gaze-assisted interaction, are limited to click (select) and double click [125]. On the other hand, the implementation of GAWSCHI supports click, double click, click and hold, hold and release, and importantly can be easily extended to allow for more user actions. Third, the existing implementations demand conscious efforts by the user; the studies that report NASA Task Load Index show a higher mental demand [64]. However, with the user actions isolated to a foot controlled quasi mouse, GAWSCHI shows a very minimal user effort, but much higher performance. Lastly, GAWSCHI has shown to work at least equally accurate, and in some cases even quicker than mouse based interaction, while being completely hands free. Such a design lends itself to enriched interactions among the general demographics, and an assistive technology for the physically challenged.

2.4 System Architecture

The goal of GASWCHI is to enable Gaze Augmented Computer Human Interaction that is noninvasive, inconspicuous, efficient, and accurate, while inducing no physical strain or cognitive load on the user. Hence, we believe any incremental improvement over the existing solutions, which use either dwell time or a blink to address the Midas Touch problem will not help toward achieving the goal of seamless integration, and efficient and accurate gaze mediated interaction on the native interface of a computer. In our approach, we move beyond the blink and dwell time based approaches for activation of the point of regard. We have invented a wearable device that

is controlled with with user's foot, and functions as a quasi mouse. The device has a small form factor, and is flexible enough to be worn on or attached to the user's foot-ware. The isolation of responsibilities across multiple modules (devices) lends GAWSCHI a unique design among gaze mediated interaction systems. The framework consists of three primary modules: 1) Gaze Interaction Server, 2) Eye Tracking Module, and 3) Foot-controlled Quasi Mouse. A working model of the system is depicted in Figure 2.1.

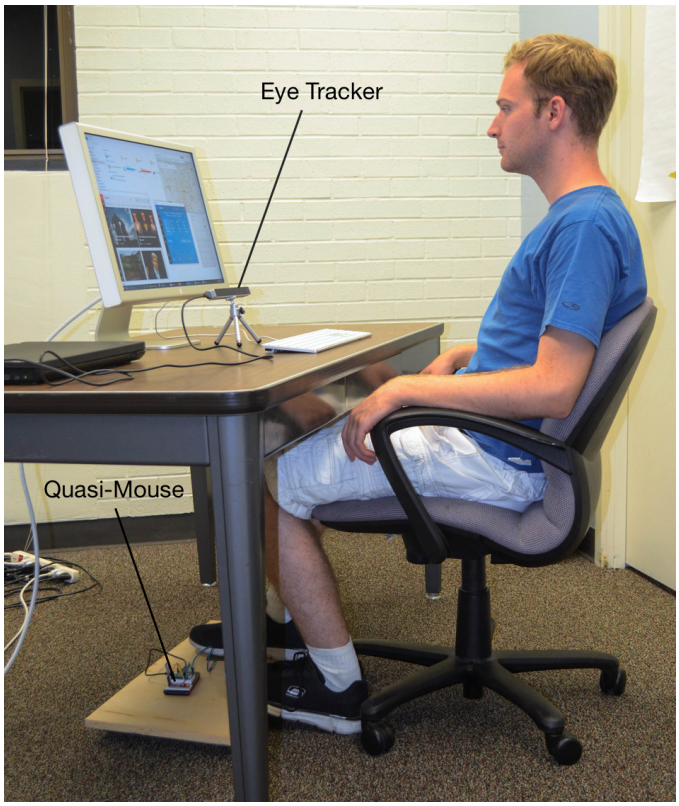

Figure 2.1: A user working on a computer using GAWSCHI. An eye tracker is placed in front of the user, and the user is facing the display. The user simply looks at the desired point of interest on the screen and the cursor moves to that location. To click, the user presses the pressure sensor attached to the foot-controlled Quasi Mouse.

2.4.1 Gaze Interaction Server

The Gaze Interaction Server, a central module, achieves the desired interactions by mediating between the Eye Tracking Module and Foot-controlled Quasi Mouse. To achieve the desired interactions between the modules, the gaze interaction server runs on a computer to which both the eye tracking module and foot-controlled quasi mouse are connected over USB and Bluetooth respectively. The gaze interaction server implements an algorithm to navigate the cursor on the screen by applying simple filtering computations on smoothed gaze coordinates, received from the eye tracker. In addition, the Gaze Interaction Server is also responsible for executing the commands issued by the user, at the point of regard. The user issues the desired command by operating the quasi mouse; these commands are then delivered over Bluetooth. The wireless connectivity between the quasi mouse and gaze interaction server provides the freedom of positioning the quasi mouse anywhere on the floor or on foot-ware, and being able to move freely, while supporting multiple configurations.

2.4.2 Eye Tracking Module

GAWSCHI uses the “Eye Tribe" tracker, an eye tracking system that provides a pair of (x,y) screen coordinates, based on where the user is looking at[2]. The Eye Tribe tracker is a table top eye tracker that is placed on a tripod, below the monitor. To work with the eye tracker, the user position is adjusted such that the face is centered in front of the monitor at a distance of 45 – 75 cm [2]. The eye tracker computes (x,y) coordinates by extracting the information from the user’s eyes and face while the user works on a computer. Prior to using the eye tracker with GAWSCHI, the system is calibrated for each user to develop a unique model for the user’s eye characteristics. The eye tracking module connects to the Gaze Interaction Server, and when the user activates gaze mediated interaction the eye tracking module begins streaming the gaze data to the Gaze Interaction Server.

[2]theeyetribe.com

2.4.3 Foot-operated Quasi Mouse

A significant contribution of this work is the development of a wearable quasi mouse that is operated with the user's foot, specifically the big toe. The user executes commands at the point of regard on screen by operating the quasi mouse. The quasi mouse allows for interactions like click, double click, click and hold, and hold and release by pressing on a flexible pressure pad (3.5" x 1.75"). The user's action (pressing the pressure pad) is encoded into an appropriate character (numeric) based on the amount of pressure applied, and the encoded command is delivered to the Gaze Interaction Server over Bluetooth. The Gaze Interaction Server subsequently decodes and translates the encoded message into an appropriate mouse event on the screen. Hence, both the Foot-controlled Quasi Mouse and Gaze Interaction Server work in conjunction to provide the infrastructure required to execute user commands.

2.4.4 Working Model

To initiate gaze augmented interaction through GAWSCHI, a user first starts the GAWSCHI desktop application shown in Figure 2.2, this initiates the Gaze Interaction Server for other modules to pair with. The quasi mouse is then powered on and connected to the gaze interaction server by clicking on the "Pair Wearable" button on the interface. The user then places the quasi mouse on the floor, or attaches it to his/her footwear, and positions himself/herself in a comfortable posture in front of the eye tracker for calibration. After the eye tracker is calibrated, the user connects the eye tracker to the gaze interaction server by clicking on "EyeTracker" button on the interface. The cursor onscreen starts moving according to the user's gaze, once eyetracker starts streaming the gaze data to the interaction server. At this point, the user can reach and interact with the interface elements on screen just with the gaze and inputs from the quasi mouse as shown in Figure 2.1.

2.5 System Implementation

2.5.1 Gaze Interaction Server

The Gaze Interaction Server that is responsible for on screen mouse navigation, and the execution of user issued commands at the point of regard is developed on the Eye Tribe SDK[2]. The

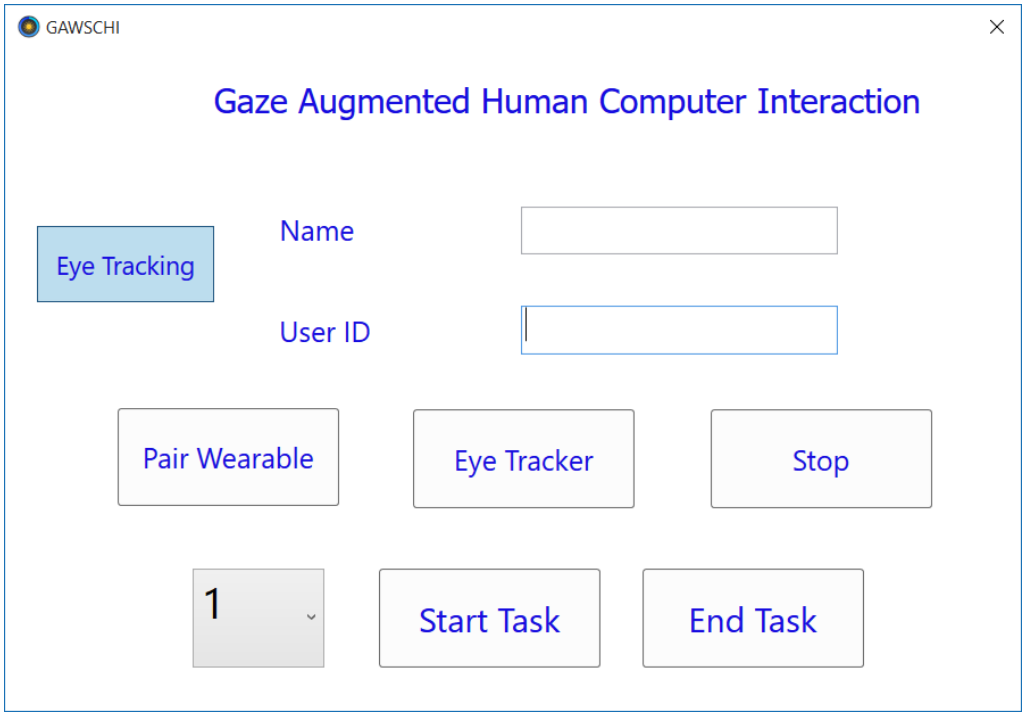

Figure 2.2: GAWSCHI desktop application used for system evaluation. This interface allows the central controlling unit to connect to both eye tracker and foot-controller. The application records the time it took to complete a selected task.

system is implemented in the C# programming language; it leverages the eye tribe C# libraries[3] to communicate and receive data from the eye tracker. The gaze interaction server registers itself with the Eye Tribe gaze manager to receive gaze data at a frequency of 60 Hz. With each gaze update the gaze interaction server receives both smoothed and raw (x,y) coordinates, and also fixation details. After receiving the (x,y) coordinates, the system transforms them to the exact (x,y) point on the screen by applying an offset if required. Furthermore, the gaze interaction server then moves the mouse pointer to the transformed (x,y) coordinate on screen by sending a mouse move command. The other primary responsibility of the Gaze Interaction Sever is to communicate with the quasi mouse to receive and execute user commands at the point of regard. The commands received are encoded using a predefined single byte character, which are then decoded into an appropriate mouse command, and executed instantaneously at the current (x,y) mouse coordinates on

[3] dev.theeyetribe.com/csharp/

the screen.

2.5.2 Foot-operated Quasi-mouse

The quasi-mouse is built with three main components: 1) Teensy Microcontroller[4], 2) Bluetooth Modem (BlueSMiRF)[5], and 3) Force Sensitive Resistor[5]. A pictorial depiction of the two versions of the quasi-mouse are shown in Figure 2.3 and Figure 2.4. Also, the complete circuit diagram is shown in Figure 2.5.

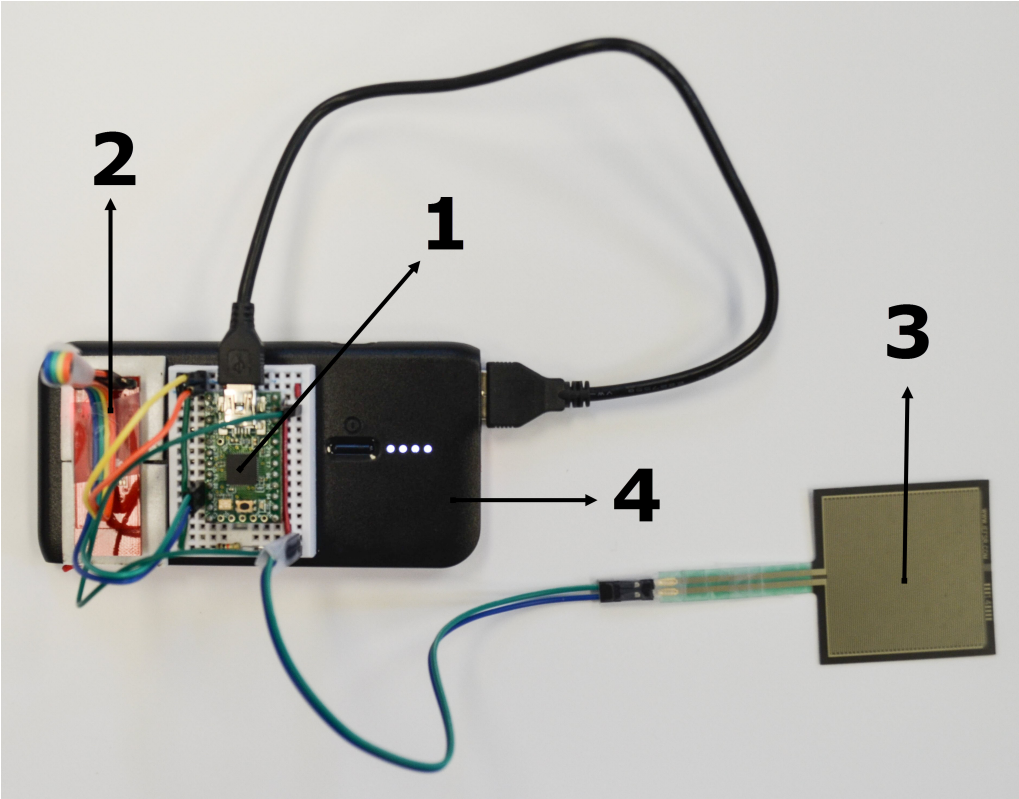

Figure 2.3: Quasi-mouse on the floor: this version of the foot-controller is placed on the floor and the user would only interact with pressure sensor. This device is good for stationary setup like a desktop.

The user input from pressing the pressure pad (Force Sensitive Resister), is sensed by measuring the output voltage of a voltage divider circuit. The minimum amount of pressure to be

[4] www.pjrc.com
[5] www.sparkfun.com/products

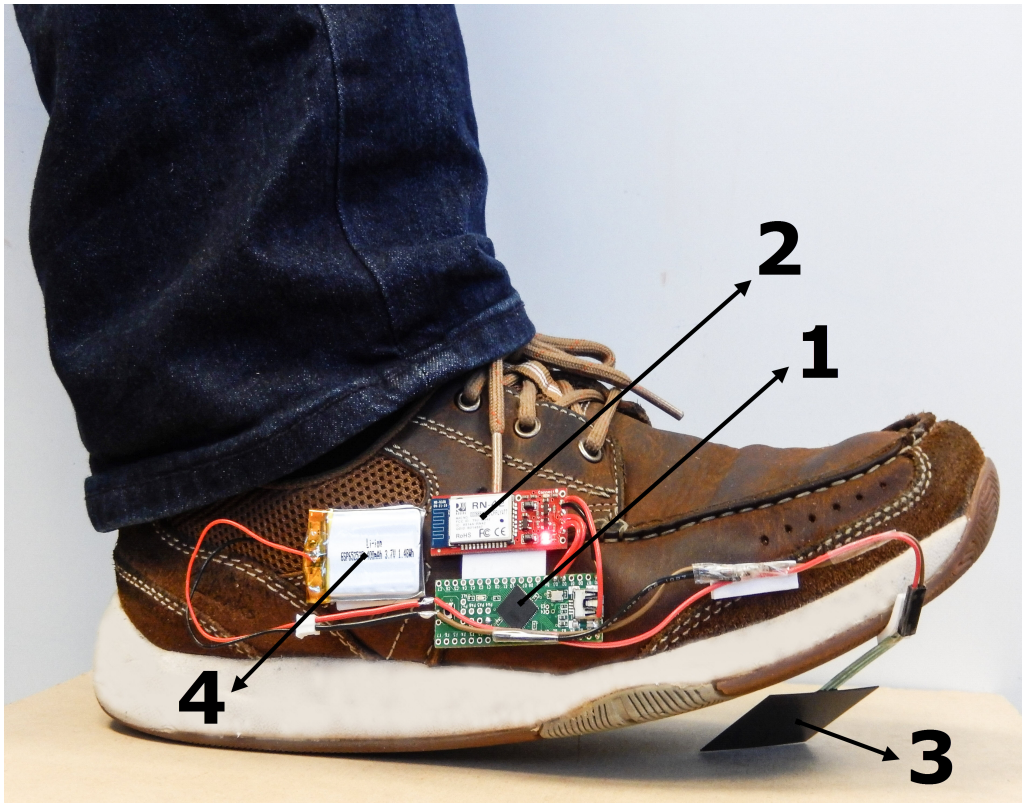

Figure 2.4: Quasi-mouse worn on the footwear. 1) Microcontroller, 2) Bluetooth Modem, 3) Force Sensitive Resistor, 4) Battery

applied, to be registered as an input action, can be adjusted by setting the output voltage threshold in equation 2.1. In equation 2.1, $R1$ and $R2$ are the resistance values, and V_{in} is the input voltage.

$$V_{out} = V_{in} \cdot \frac{R1}{R1 + R2} \tag{2.1}$$

Each user input, based on the pressure thresholds, is encoded into a character(numeric) value and communicated to the Gaze Interaction Server via Bluetooth Modem. The Gazer Interaction Server then decodes the message received and executes the respective command at the point of regard.

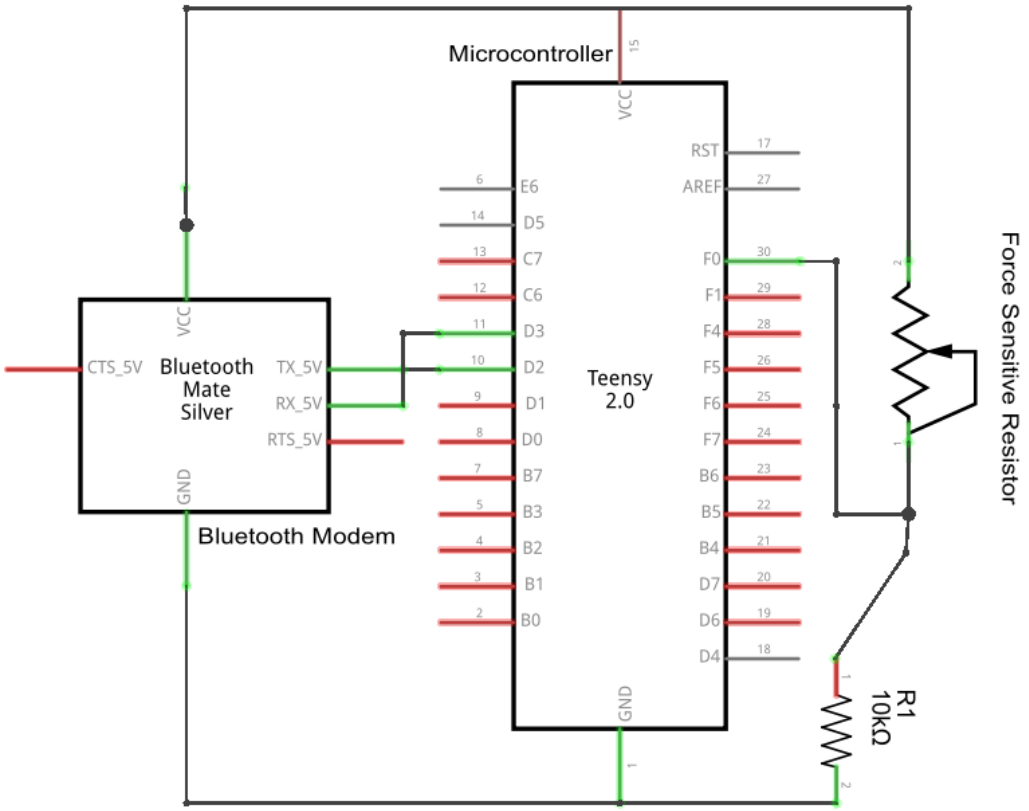

Figure 2.5: Circuit Diagram - Foot-operated Quasi Mouse

2.6 Experiment Design

GAWSCHI is evaluated with a comparative user study to understand its efficacy in substituting the mouse for performing common interaction tasks on a computer. Questions that we sought to answer through this user study are as follows:

1. Can gaze augmented interaction, supplemented with a wearable device for executing user commands, perform at least as good (speed of task execution) or better than the mouse based interaction?

2. Do some users exhibit better gaze control, and hence higher performance than others?

3. Will a user's performance improve as s/he becomes better acquainted with the gaze-assisted

interaction?

4. How is the accuracy of gaze-assisted interaction related to the dimensions of the interface element?

5. What effects does GAWSCHI have on the mental, temporal, and physical demand of the user? How is their performance impacted?

6. How well is the design of GAWSCHI accepted by the users?

To answer this broad range of questions, we recruited 30 participants, belonging to diverse ethnic groups (USA, Mexico, Europe, India, China, and Korea). The rationale behind the consideration of ethnicity is that the physiological structure of the eye differs between ethnicities and that impacts eye tracking accuracy. The participants pool consists of 26 males and 4 females, all either graduate or undergraduate students, with the ages varying between 20 and 43 ($\mu_{age} = 24.7$); none had previously used gaze-interaction. For the user study we used a 23" monitor, with a 1900 x 1200 resolution (19.5" x 12.19" screen size), and a PPI of 98.44 (Pixels Per Inch). Each user was first briefed about the experiment; subsequently presented with a demo of each required activity, using both the mouse and gaze-assisted interactions. Furthermore, each participant was put through a practice session for a maximum of five minutes. Following the completion of the experiment, each user completed a NASA TLX survey and a quantitative evaluation of the system. We have excluded data collected from nine participants because of one or more of the following reasons: a) calibration failure, b) the participant was using thick spectacles for vision correction, or c) the participant could not complete all the tasks.

During the experiment a user performs eleven interaction activities on a computer (running Windows 10). The task execution is counterbalanced, with 50% of the users performing the eleven tasks first with the gaze-assisted interaction, and subsequently performing the same tasks using the mouse; the remaining 50% of the users followed the reverse order. The interaction activities chosen are the common activities, performed routinely on a computer by a user; the tasks were elicited

from interviews and a pool of common tasks. The eleven tasks performed during the experiment are:

1. Click on the Windows start menu, open the News application, open the first article, and scroll through the article by clicking on the side scroll bar.
2. Click on the Windows start menu, open the Weather application, and report the current temperature.
3. Open Chrome web browser from the toolbar, log-in to Gmail (credentials pre-saved) by clicking on its bookmark, and open the first email.
4. Open Chrome web browser from the toolbar, open YouTube by clicking on its bookmark, play any video, pause, switch to full screen mode, and switch back to the normal screen mode.
5. Open Chrome web browser from the toolbar, open Google Maps by clicking on its bookmark, click on a point of choice, and switch to Google street view.
6. Open the Images folder on the desktop, and open an image.
7. Open the Documents folder on the desktop, and open the first pdf file.
8. Run the Notepad application by clicking on its icon on the desktop.
9. Click on the Windows start menu, open Calculator, and compute 5+7.
10. Open Calendar application from the toolbar, and view today's schedule by clicking on the toolbar item Day.
11. Click on the Task View icon on toolbar, and switch to the top left task.

For the user study we used a 23" monitor, with a 1900 x 1200 resolution (19.5" x 12.19" screen size), and a PPI of 98.44 (Pixels Per Inch). Each user is first briefed about the experiment;

the person conducting the experiment demos the activities to be performed both using the mouse and gaze-assisted interactions. Before performing the gaze-assisted interaction, the eye tracker is calibrated for each user, and the user is put through a practice session for a maximum of five minutes. During the practice session, the participant learns to move the cursor on screen with the gaze, and focus at various points as directed by the experimenter. In addition, the participant also learns how to operate the foot controlled quasi mouse. The interface used for the experiment is shown in Figure 2.2, where the participant is able to select the task ID, and initiate and end the task, by clicking on the “Start Task" and the “End Task” buttons respectively.

Following the experiment, the participant completes a quantitative system evaluation, based on the Likert scale, on various design aspects of the system. Furthermore, the participant also completes the NASA Task Load Index survey for the gaze-assisted interaction. We have excluded data collected from nine participants because of one or many of the reasons like, the eye tracker failed to calibrate for the participant, the participant was using thick spectacles for vision correction, the participant could not complete all the tasks, and one of the participants was legally blind in the right eye.

2.7 Results

During the user study the time taken to perform each task, both with the gaze-assisted interaction and mouse-based interaction is recorded for each participant. Figure 2.6 shows the mean time taken to perform each task, and the standard deviation as the error bar, for both mouse and gaze based interactions. Figure 2.6 also shows Δt, the time difference between the mean times to perform a task using gaze and mouse based interactions. $\Delta t = GazeInteractionMean - MouseInteractionMean$. Δt compares the time taken by gaze-assisted interaction to the mouse-based interaction, hence a positive value of Δt indicates that gaze-assisted interaction took Δt time more than the mouse-based interaction. Alternatively, a negative value of Δt indicates that gaze-assisted interaction took $|\Delta t|$ time less than the mouse-based interaction. The Δt values from Figure 2.6 indicate a marginal speed increase with mouse assisted interaction over gaze-assisted interaction for all the tasks, except for Task 1.

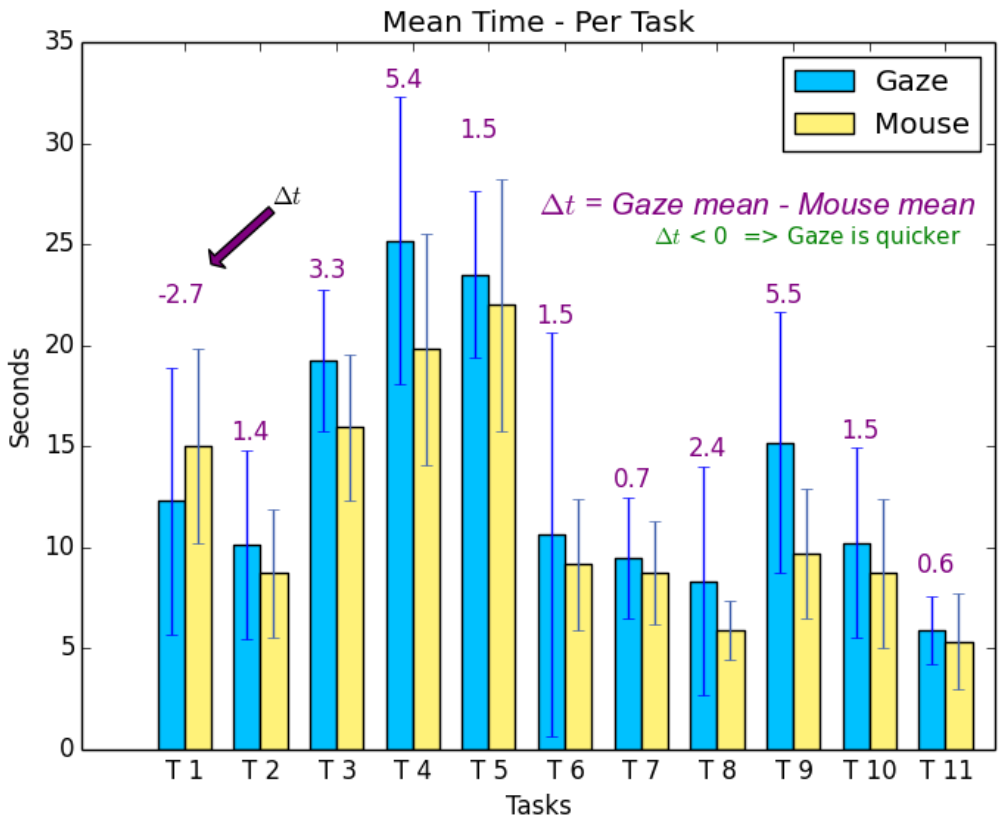

Figure 2.6: Mean Time Taken for Each Task, Std Dev, and Δt

To better understand the statistical significance of per task mean time differences, we conducted a matched-pairs t-test for each task. An analysis based on matched-pairs t-test is appropriate since our study involves the same participants performing a set of eleven tasks, using both mouse and gaze based interactions. Our null hypothesis is that the mean of matched-pairs time differences, for each task, using both mouse and gaze based interactions is zero. Results from the two tailed matched-pairs t-test with a confidence interval of 95% ($\alpha = 0.05$) is shown in Table 2.1. It can be observed that the p-value > 0.05 for all the tasks, except for tasks 3, 4, and 9. Hence (except for 3, 4, and 9), we fail to reject the null hypothesis that the mean of matched-pairs time differences, for each task, is zero. Since we fail to reject the null hypothesis, except for tasks 3, 4, and 9, we infer that the gaze-assisted interaction with GAWSCHI is at least as good as mouse based interaction. For tasks 3, 4, and 9, we reject the null hypothesis and accept the alternate hypothesis that the

mean of matched-pairs time differences is non-zero; **reasons for rejection of the null hypothesis** is elaborated in the discussion Section 2.8.

Table 2.1: Matched-pairs t-test, Two tailed, 95% Confidence Interval. We observe that there is no significant difference in the time taken to complete a task between mouse and gaze inputs except for Tasks 3, 4, and 5.

Task	**Mean Time Diff** $\bar{d}$	**Standard Error** $SE(\bar{d})$	**t-stat**	**p**
1	2.70	1.40	1.93	0.068
2	-1.39	1.09	-1.27	0.219
3	-3.32	0.91	-3.65	0.002
4	-5.38	1.47	-3.67	0.002
5	-1.52	1.31	-1.17	0.257
6	-1.48	2.04	-0.73	0.476
7	-0.73	0.59	-1.25	0.227
8	-2.43	1.24	-1.95	0.065
9	-5.52	1.34	-4.12	0.001
10	-1.49	1.21	-1.23	0.234
11	-0.59	0.40	-1.45	0.161

Furthermore, we verify our hypothesis that the design of a gaze-interaction system, like GAWSCHI, that isolates user actions to a foot-operated quasi-mouse, demands minimal user efforts (Figure 2.7). Though comparison of NASA TLX scores for an entirely new interaction modality like gaze against a highly familiar modality like mouse is impractical, from Figure 4.a it can be observed that, despite TLX scores for gaze being marginally higher than mouse, each TLX score for gaze is still lower than a high workload threshold value of 40 as used in previous studies, e.g., [126]. Hence, though interaction using GAWSCHI is not simpler than mouse, we infer that users do not experience physical or mental workload (fatigue) when using GAWSCHI.

Lastly, Figure 2.8 shows a quantitative evaluation of the system, based on the Likert scale

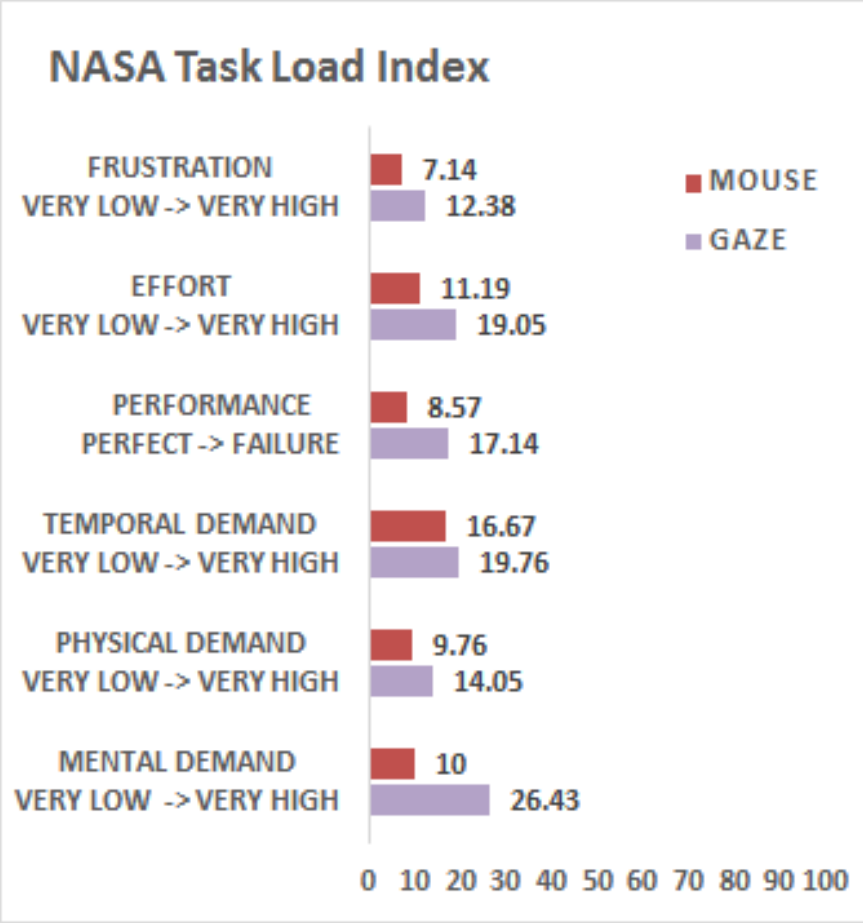

Figure 2.7: NASA Task Load Index for Mouse and Gaze Interaction (lower is better))

(lower score is better), on various design aspects. In can be observed that the users are highly satisfied with the overall interaction. In addition, users rated high for screen reachability, mouse speed, system feedback, quasi mouse ease of use, and their ability to click at the point of regard.

2.8 Discussion

Through the system evaluation, we have tried to answer the questions listed in Section 2.6. Based on the observations from the study, results of statistical tests, the nature of the task, interface elements involved in the task, and the user feedback, we have derived that the minimum dimensions of an interface element should be at least 0.60" x 0.51" for a user to conveniently interact with that UI element. Performance of a user with no prior experience using the gaze-assisted interaction is at least as good as mouse-based interaction, when the dimension of the interface element is at

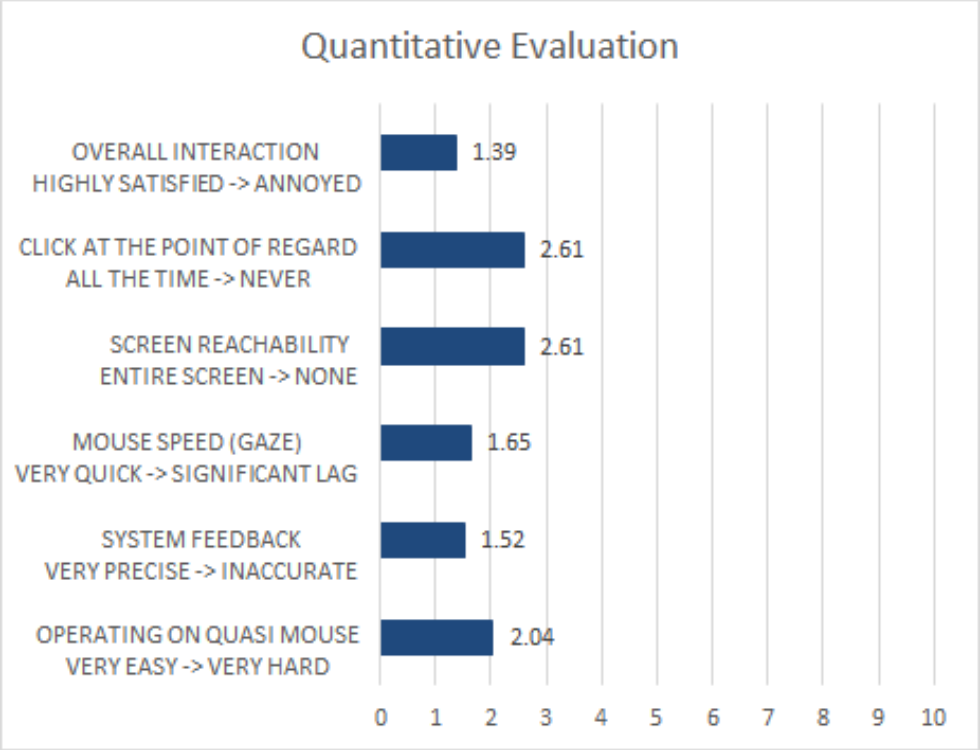

Figure 2.8: GAWSCHI - Quantitative Evaluation for Gaze Interaction (lower is better)

least 0.60" x 0.51" (this dimension is resolution and PPI agnostic), and the spacing between the UI elements is discernible. From both Figure 2.6 and t-test results from Table 2.1, we infer that GAWSCHI is at least as good as, and in some cases quicker than the mouse based interaction.

We observed two reasons for matched-pairs t-tests fail on tasks 3, 4, and 9: 1) Tasks 3 and 4 are browser based tasks, and the dimensions of most of the interface elements in all these tasks are less than 0.60" x 0.51", and 2) For Task 9, where a user performs a predefined, addition task on the calculator, the proximity of numbers and a minimal gaze drift leads to higher time consumption. In all the other tasks, where the gaze interaction matches the performance of a mouse, the dimensions of UI elements are higher than the threshold. In addition, users wearing vision correction devices, like spectacles, tend to execute misclicks, and also take marginally more time to interact with UI elements, when compared to normal users. It can be observed in Figure 2.6 that after the fifth task, a user's efficiency improves as the user gets more acquainted with the gaze based interaction; the same is also verbally confirmed by the participants.

3. A FITTS' LAW EVALUATION OF GAZE INPUT COMPARED TO MOUSE AND TOUCH INPUTS

In Chapter 2, we discussed how efficient point-and-click interactions can be achieved using our gaze-assisted interaction framework. Also, we discussed how our framework addresses the issue of Midas Touch by using a supplemental foot-based input. Our system was evaluated by comparing the performance of gaze input to mouse (time to complete) when performing 11 pre-defined interaction tasks on a computer. In this chapter, we evaluate the performance of gaze and foot-based interaction framework in comparison to the mouse and touch inputs through Fitts' Law [127, 128, 129]. Performance evaluation of any input modality through Fitts' Law is crucial since the output metrics from Fitts' Law evaluations are comparable to other such evaluations of the input methods [128, 129]. This chapter is divided into two parts. We begin with an introduction to Fitts' Law (Section 3.1) and in prior work (Section 3.2), we discuss all the Fitts' Law evaluations that included gaze as one of the inputs. Furthermore, the the first part (Section 3.3) discusses the Fitts' Law evaluation of the gaze input on standard display (up to 24"), and the second part (Section 3.4) discusses the evaluation on large displays (up to 84"). Lastly, the chapter will be concluded with a comparison of the performance of the gaze input on both standard and large monitors 3.5[1].

3.1 Introduction

Fitts' Law models the human movement analogous to the way information is transmitted [130]. Different kinds of movement tasks have different indices of difficulties expressed in bits/s. To perform a movement task, a certain number of bits of information is transmitted by the human motor system. The performance of a movement task can be quantified (throughput) by dividing the number of bits transmitted by the movement time (MT) [130, 131]. Furthermore, Fitts' Law

[1] *Parts of this chapter are reprinted with permission from "A Fitts' Law Evaluation of Gaze Input on Large Displays Compared to Touch and Mouse Inputs" by Rajanna et al., 2018. Publisher and Copyright holder ACM Digital Library, 2018, New York. Conference ETRA '18: 2018 Symposium on Eye Tracking Research and Applications Proceedings - doi.org/10.1145/3206343.3206348

has been used in HCI research in two ways, first, to predict the time it takes (movement time) for a user of a graphical interface to move the cursor to the target and click it. Second, to compare the speed and accuracy of different input methods through a single statistic called throughput [130]. The throughput of an input method is computed as follows:

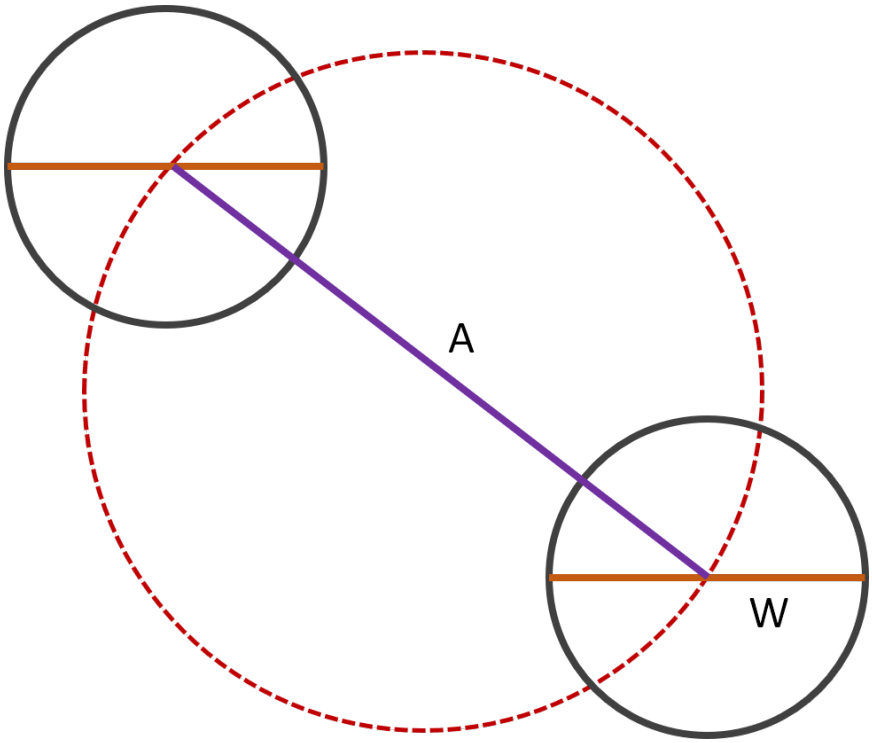

Figure 3.1: Fitt's Law Task Setup: N number of target circles with a width "W" pixels in diameter are arranged around a layout circle with a width (amplitude) "A" pixels in diameter.

$$Throughput = \frac{ID_e}{MT} \tag{3.1}$$

Where, ID_e is the effective Index of Difficulty, and MT is the mean Movement Time. The subscript e indicates "effective." While ID represents the Index of Difficulty considered for the tasks, in its effective form, i.e., ID_e, represents the difficulty of the task completed by the user rather than the what she was presented with [130]. The ID_e is calculated as shown in Equation 3.2.

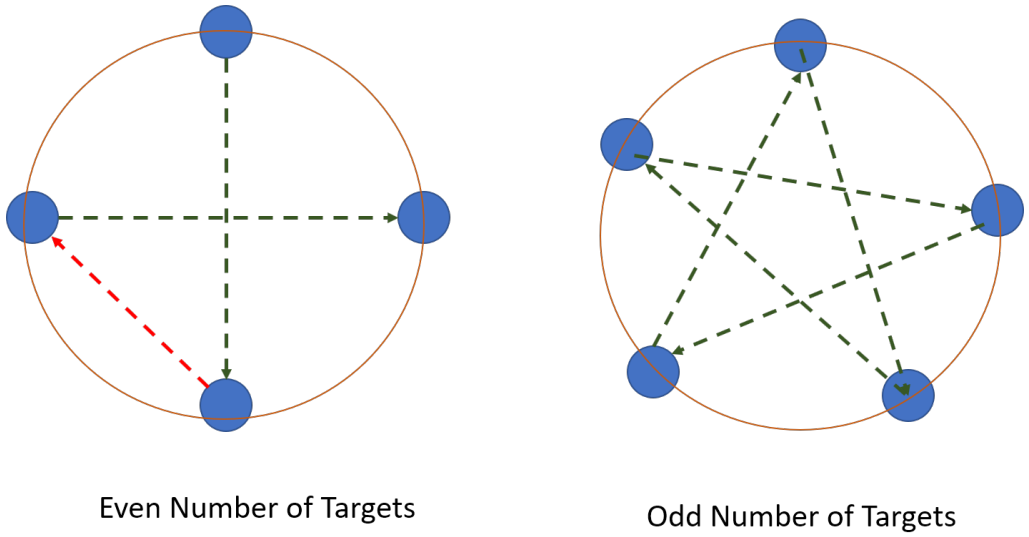

Figure 3.2: Fitt's Law Task Setup: demonstration of why odd number of targets should be considered and not even number of targets. With odd number of targets, the Euclidean distance between two opposite targets is the same. However, the distance two between opposite targets is not consistent if even number of targets were considered.

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

Where, A_e is the effective distance to the target (amplitude), i.e., the mean of the effective movement amplitudes measured along the task axis over a sequence of trials. The effective amplitude of a trial in a sequence is computed as $A + dx$, and the dx is computed as shown in Equation 3.7. W_e is the effective target width which is calculated as shown in Equation 3.3.

$$W_e = 4.133 \times SD_x \tag{3.3}$$

Where, SD_x is the standard deviation of the selection coordinates (dx - overshoot or undershoot) in a sequence a trials which is computed as shown in Equation 3.7.

To compute dx, first the selection coordinate of a trial is projected back on to the task axis, and this is done to maintain the inherent one-dimensionality of Fitts' Law [130, 131]. The task axis is

the line joining the center of source (from) to destination target (to).

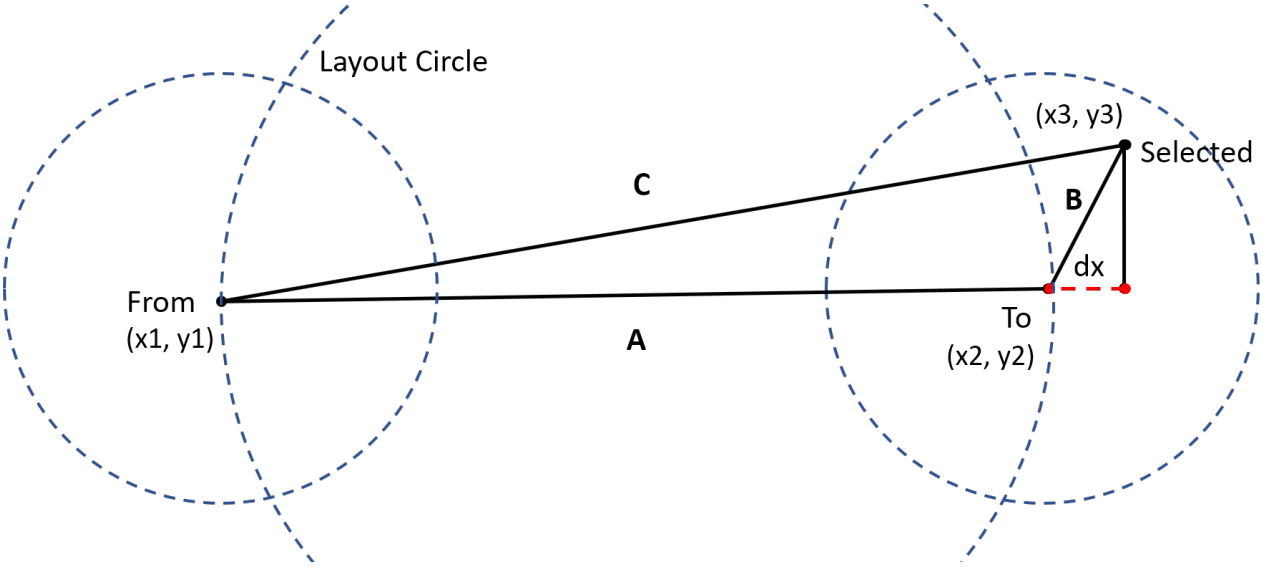

Figure 3.3: Computation of dx used in the calculation of the Effective Amplitude A_e and Effective Target Width W_e. The amount by which the user overshoots or undershoots from the center of the target is projected back on to the task axis. This a) ensures the inherent one-dimensionality of Fitts' Law, and b) the difficulty of the task is computed based the actual task completed by the user rather than what she was presented to do.

$$A = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2} \tag{3.4}$$

$$B = \sqrt{(x_3 - x_2)^2 + (y_3 - y_2)^2} \tag{3.5}$$

$$C = \sqrt{(x_1 - x_3)^2 + (y_1 - y_3)^2} \tag{3.6}$$

$$dx = (C * C - B * B - A * A)/(2.0 * A) \tag{3.7}$$

3.2 Prior Work

Fitts' Law evaluation of gaze input has been primarily conducted in desktop settings, and the common selection methods considered have been gaze+dwell, gaze+click, and mouse [129, 131,

132, 133, 130, 134]. Zhang et al., presented the first work on Fitts' law evaluation of gaze input that conforms to ISO 9241-9 [131]. The authors compared gaze input with short and long dwell times and gaze+Spacebar with mouse input. The target widths chosen were 75 px and 100 px, and amplitude chosen were 275 px. The Gaze+Spacebar eliminated the waiting time, it was the best selection method among gaze inputs with a throughput of 3.78 bits/s (mouse was 4.68 bits/s). Also, the participants liked Gaze+Spacebar out of the three gaze-based inputs.

Ware et al., presented a Fitts' law evaluation of gaze input [135]. Three selection methods were used along with gaze: a button press, dwelling, and an onscreen select button. In an experiment where the participants had to select one of the seven menu items arranged vertically, each item covering a visual angle of 2.0 to 1.65 degrees, the authors found that irrespective of the selection procedure, the gaze-based selection methods took less than 1 second for target selection. Also, target selection with eye movements fits the Fitts' law well.

Miniotas et al., tested the validity of the findings from Ware et al. [135], by comparing the performance of an eye tracker and a mouse in a simple pointing task [132]. The participants had to make rapid and accurate horizontal movements to targets that were vertical ribbons. The target amplitude included 26, 52, 104, and 208 mm and the target widths included 13 and 26 mm. The authors found that the selection time is longer for the eye tracker than for the mouse by a factor of 2.7.

Zhai et al. [136], proposed MAGIC: Manual and Gaze Input Cascaded Pointing to improve the usability of gaze input. A Fitts' law evaluation was conducted by using 3 input methods which included an isometric pointing stick and two versions of MAGIC pointing. The experiment included two target sizes (20, 60 px) and three target distances (200, 500, and 800 px). The authors found that the completion time and target distance did not completely follow Fitts' law when using MAGIC pointing, but when considering both target size and target distance the data fit the Fitts' law but relatively poorly. Pointing with two version of MAGIC achieved a higher performance (4.55 and 4.76 bits/s) than manual input (3.2 bits/s).

Vertegaal et al. [133], evaluated 4 input methods in a Fitts' task involving large visual tasks.

The input methods included a gaze and manual click, a gaze and dwell click, a stylus, and a mouse. Unlike the Fitts' law task in [137] that used ISO multi-directional tapping task, the authors in this experiment aimed at using gaze input to disambiguate between contexts of interaction, e.g., selecting one of the two large windows on a screen. Hence, the experiment involved alternate selection of one of the two large visual targets (tasks with low index of difficulty). The target widths included 70 px, 100 px, and 140px, and the amplitudes included 200 px, 400 px, and 800 px. The index of difficulty varied from 1.28 bits/s to 3.6 bits/s. In this experiment, gaze-based inputs outperformed manual input methods: mouse and stylus achieved an index of performance (IP corrected) of 4.7 and 4.2 bits/s respectively, but gaze with manual click and gaze with dwell (100 ms) achieved an IP of 10.9 and 13.8 bits/s respectively. Though gaze input outperformed manual input, it also had higher error rate: mouse 4.6%, stylus 6.2%, gaze with manual click 11.7%, and gaze with dwell click 42.9%. The authors concluded that gaze input with manual click provides the best trade-off between speed and accuracy.

Bleeders et al. [138], evaluated three gaze+speech inputs against a mouse in an ISO multi-directional tapping task. The three gaze+speech based inputs included eye gaze and speech (ETS), ETS with magnification (ETSM), and ETS with gravitational well (ETSG). The authors found that the mouse was far superior in performance when selecting the targets (throughput), compared to all gaze-based inputs.

Surakka et al. [139] compared target acquisition of gaze pointing and EMG selection (i.e., frowning) to the mouse. The mouse was most effective for short distances, but the gaze+EMG input combination showed a higher index of performance than the mouse for error-free data, suggesting that gaze+EMG may be faster at longer distances, but their data did not show any speed advantage of gaze+EMG over the mouse. San Agustin et al. [140] later confirmed, that gaze+button and gaze+EMG were in fact faster than mouse+button and mouse+EMG.

In the Prior Work (Section 3.2), we discussed Fitts' Law evaluation of gaze under various task difficulties, and gaze was being compared to various other input methods. The input methods considered were gaze+dwell, gaze+click, gaze+button-press, and mouse. However, we do not see

a comparison of gaze input to touch input. Also, gaze input was used in standard configurations like gaze and various dwell times, or gaze and a keyboard button press. But, a multimodal input combining gaze and a supplemental foot input was never considered.

3.3 A Fitts' Law Evaluation of Gaze Input Compared to Mouse and Touch Inputs on a Standard Display (up to 24")

From Chapter 2, we have learned the advantages of using foot input with gaze, and how the performance of such a multimodal system compares to the mouse input. Therefore, it is essential to compare the performance of gaze input to mouse through a standard evaluation method like the Fitts' Law. We conducted a Fitts' Law evaluation of gaze input compared to the mouse and touch inputs on a standard screen. The experiment conforms to ISO 9241-9 standardization. From a study involving 12 participants, we found that the gaze input has the lowest throughput (2.55 bits/s), and the highest movement time (1.04 s) of the three inputs. In addition, though touch input involves maximum physical movements, it achieved the highest throughput (6.67 bits/s), the least movement time (0.5 s), and was the most preferred input.

3.3.1 Fitts' Law Experiment Design

For the Fitts' Law experiment we used the software [2] developed by Soukoreff and MacKenzie [130, 129]. Specifically, we used Fitts' Task Two which is a multi-directional point-and-select task. For each trial the target to be selected is highlighted in red color, and once the highlighted target is selected, the target that is opposite to the current target gets highlighted. In accordance with the previous Fitts' law studies on gaze pointing, we used a nominal index of difficulty that ranged from 2.0 to 2.5 [131]. Hence, the amplitude, i.e., the distance to the target we chose were 1000 px and 1100 px, and the target widths were set to 230 px and 330 px. The computation of the index of difficulty is shown in Table 3.1. Figure 3.4 shows the experiment setup where the Fitts' Law task is shown on a standard 24" display.

[2] http://www.yorku.ca/mack/FittsLawSoftware/ [last accessed Jan 23rd 2018]

Table 3.1: Fitts' Law Evaluation - Standard display: Amplitude, Width, and Index of Difficulty

Amplitude (px)	Width (px)	Index of Difficulty (bits/s)
1100	230	2.53
1000	230	2.41
1100	330	2.11
1000	330	2.01

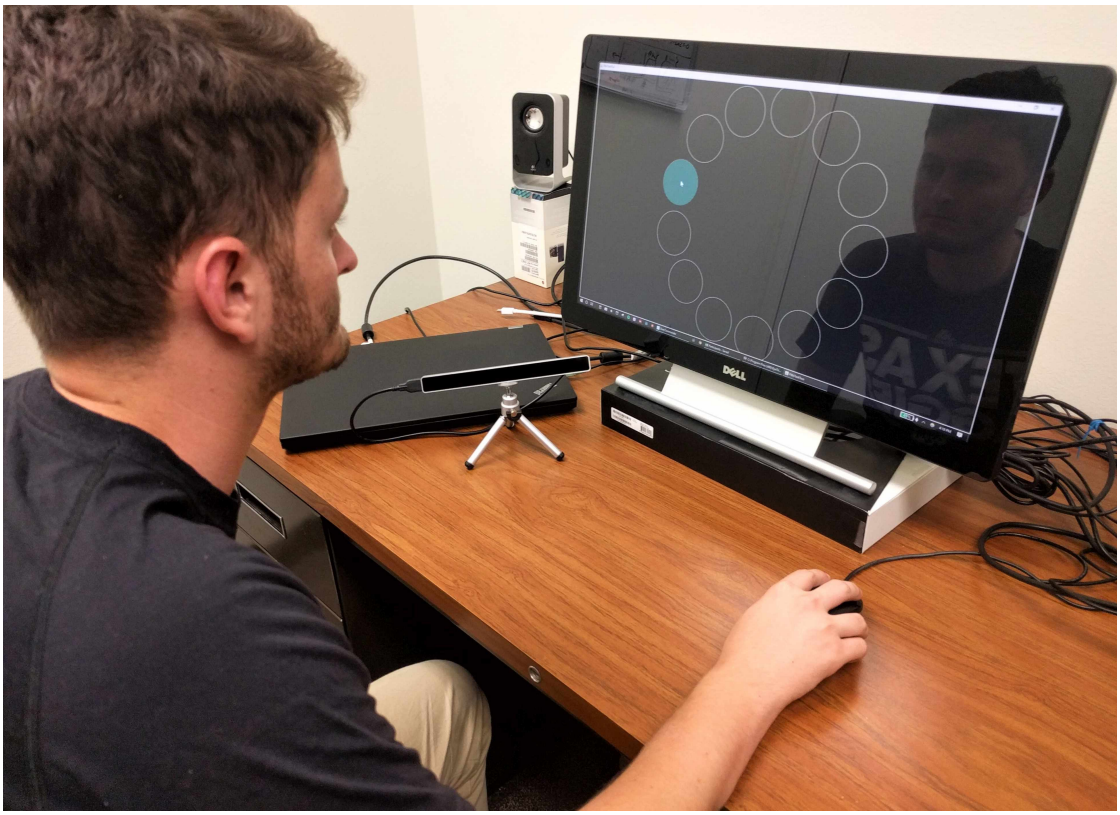

Figure 3.4: Fitts' Law Evaluation on a standard display: mouse input

3.3.2 Selection Methods

We chose three selection methods: 1) mouse, 2) touch, and 3) gaze+foot. For the mouse input, the participant used a standard mouse to select targets as shown in Figure 3.4. The cursor speed

was set to the default value. For the touch input, the participant directly touched the screen to select targets as shown in Figure 3.5. For gaze+foot input an eye tracker was placed in between the user and the display as shown in Figure 3.6. The eye tracker was removed when using the mouse and touch inputs.

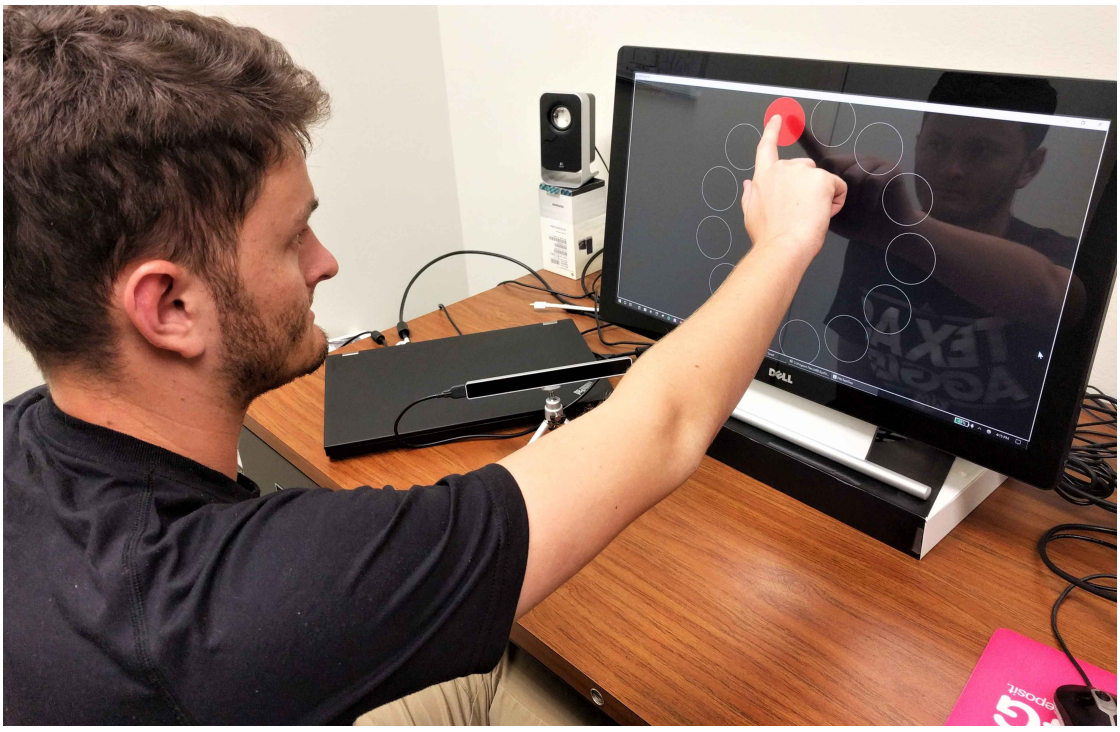

Figure 3.5: Fitts' Law Evaluation on a standard display: touch input

To achieve the gaze+foot interaction, we enhanced a gaze+foot input system developed by Rajanna et al. [141], which consists of an eye tracking module and a foot controller (Figure 3.7), and the on-screen cursor follows the user's gaze. To select a target the user first places the cursor on the target by focusing on it, and then selects it by pressing a pressure sensor, attached to the foot controller, with the foot. The foot controller connects to the eye tracking system over Bluetooth, and the entire circuitry is placed inside a portable 3D printed case.

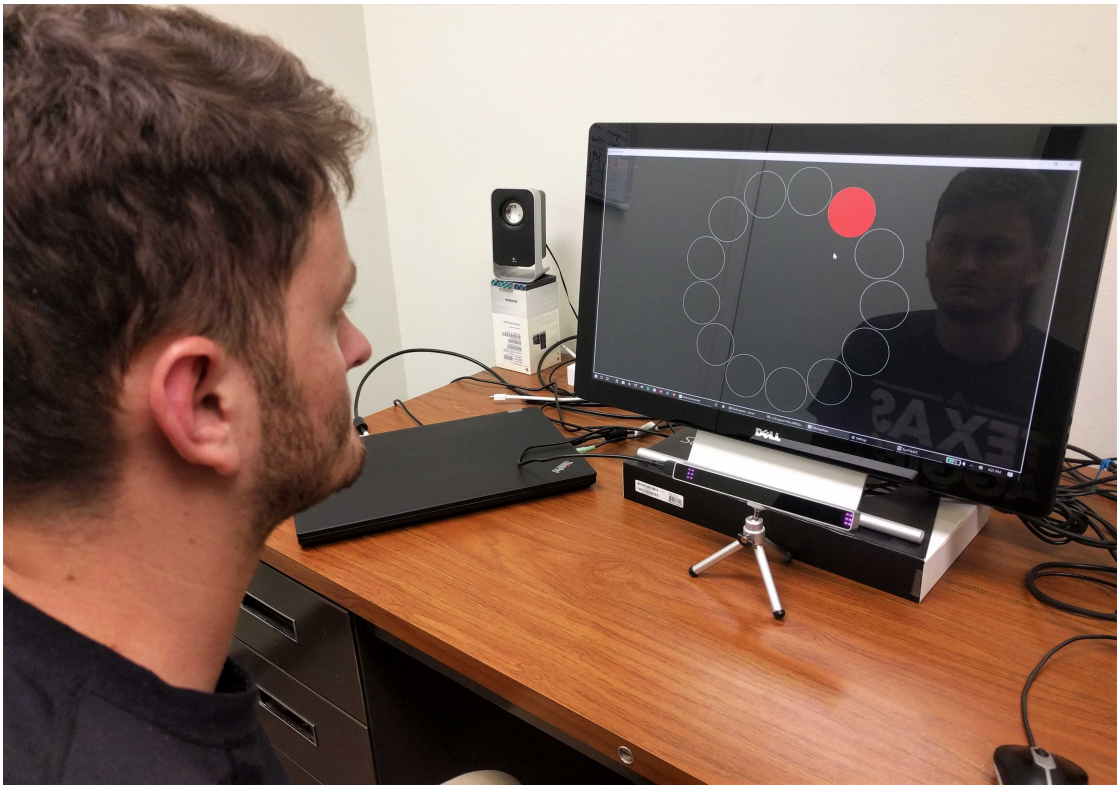

Figure 3.6: Fitts' Law Evaluation on a standard display: gaze input

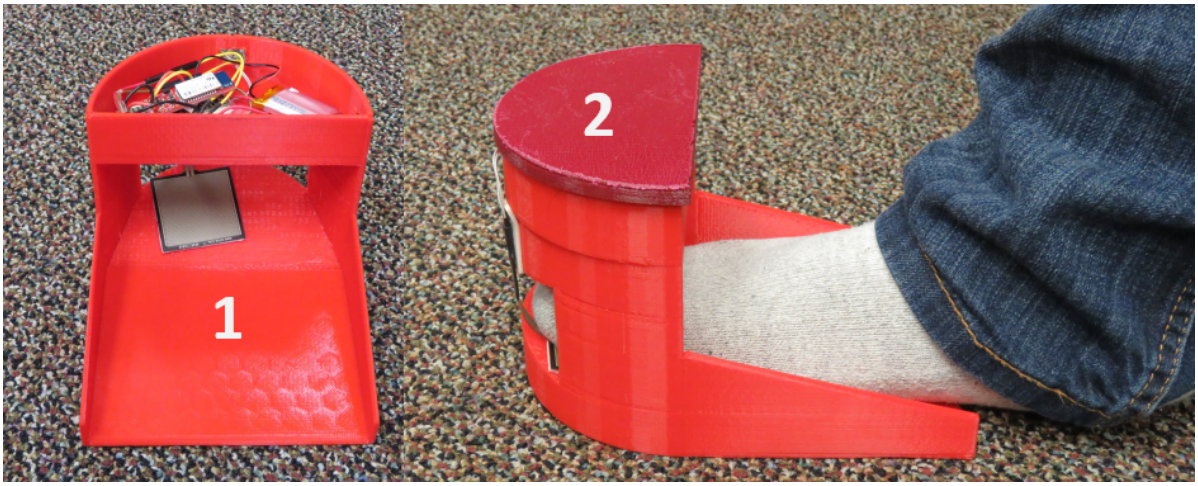

Figure 3.7: The foot controller used in the gaze+foot selection method. 1 - a force sensitive resistor, microcontroller, and bluetooth module in a 3D printed case, 2 - foot interaction.

3.3.3 Display and Gaze Tracking

The experiment was conducted on a Dell Monitor, a 24" touch enabled display [3]. We used an Eye Tribe tracker for eye tracking. The tracker had a manufacturer reported accuracy of 0.5° to 1.0° of visual angle, and had a sampling rate of 60 Hz.

3.3.4 Participants and Procedure

For the Fitts' Law experiment we recruited 12 participants (8 M, 4 F) with their ages ranging from 19 to 32 ($\mu_{age} = 23$). At the beginning of the study, each participant was briefed about the Fitts' Law task and the kind of inputs they would be using for target selection. For each input method (e.g., mouse) the participant completed one sequence of trials to familiarize themselves with the system before the actual data collection began. The participants used three input methods– gaze+foot, mouse, and touch–for target selection, and the order of input methods used by the participants was counterbalanced according to the Latin square design.

For each input method the participant completed 4 blocks of target selection task, and each block had four sequences of trials as we used two amplitudes (1000 px and 1100 px) and two target widths (230 px and 330 px). In each sequence, there were 13 trials, hence, a total of 2,496 trials (13 trials $\times$ 4 seq $\times$ 4 blocks $\times$ 12 participants) were completed for each input. Also, a total of 7,488 trials (2,496 $\times$ 3 inputs) were completed from all the three inputs. The participants were allowed to rest for a minute between each block, and in the case of gaze input, the participants were re-calibrated if the calibrated stance was disturbed between the blocks.

3.3.5 Results

We conducted a one-way ANOVA with replication on the four dependent variables (DVs): 1) movement time, 2) throughput, 3) error rate, and 4) effective target width. The independent factor was the 'selection method' which had three levels: 1) mouse, 2) touch, and 3) gaze. Table 3.2 shows the result of ANOVA on the DVs, and also the mean and standard deviation of the selection methods for each DV.

[3] missing

Table 3.2: Fitts' Evaluation - Standard Display: ANOVA and post-hoc analysis (p values highlighted in gray indicate significance at $\alpha = 0.05$).

Selection Method [Ms, Th, Gz]	Mean	Std. Dev	ANOVA
Movement Time (ms)	Ms = 683.912 Th = 490.759 Gz = 1040.456	148.539 101.692 294.870	F(2,382) = 633.144 *p = 0.000*
Throughput (bits/s)	Ms = 3.810 Th = 6.674 Gz = 2.550	0.865 1.563 0.930	F(2,382) = 797.830 *p = 0.000*
Error Rate (%)	Ms = 1.362 Th = 0.320 Gz = 3.084	3.691 1.541 5.165	F(2,382) = 26.448 *p = 0.000*
Effective Target Width (pixels)	Ms = 229.7937 Th = 148.126 Gz = 304.555	124.507 149.238 327.897	F(2,382) = 23.841 *p = 0.000*

We observe that the factor 'selection method' is significant ($p < 0.05$) for all the four DVs, i.e., the value of a DV differs among the selection methods. Out of all the selection methods, 'touch' achieves the highest throughput (6.67 bits/s), consequently it has the least movement time, error, and effective target width. Similarly, 'gaze' input has the lowest throughput (2.55 bits/s), consequently it has the highest movement time, error, and effective target width. Post-hoc tests with Bonferroni correction showed that for DVs movement time, throughput, and effective target width the difference between each pair of the selection methods, (mouse, touch) (mouse, gaze) (touch, gaze), was significant ($p < 0.05$). Figure 3.8, Figure 3.9, Figure 3.10, and Figure 3.11 compare the means of the three selection methods for each DV.

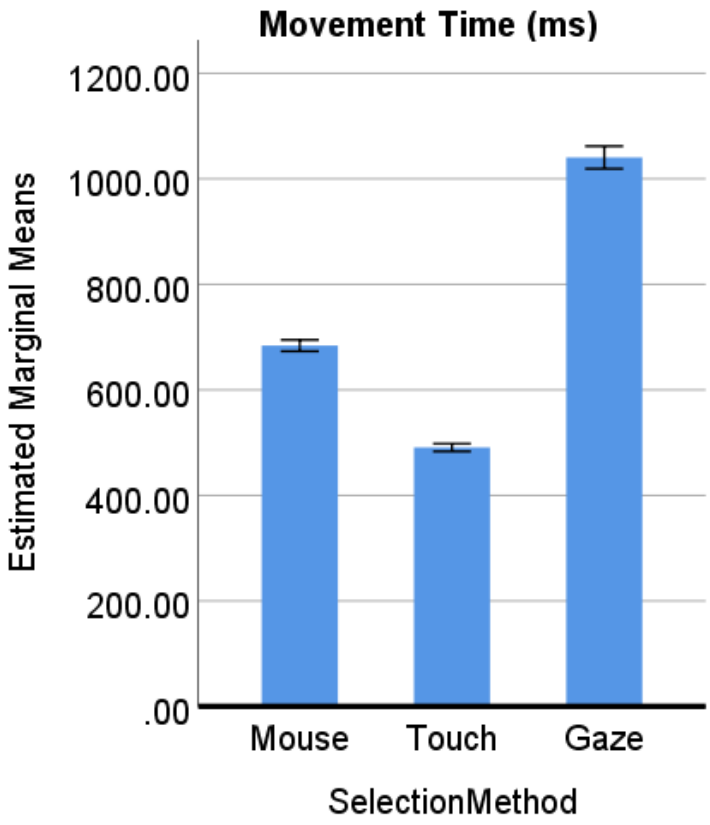

Figure 3.8: Movement time: comparison of the movement time between the three selection methods on the standard display.

3.3.6 Discussion

From the results in Table 3.2 we observe that touch input achieves the highest throughput, and it has the least movement time, error rate, and effective target width. Similarly, gaze has the lowest throughput, highest movement time, error rate, and effective target width. These results reflect that direct manipulation (input) method like touch is the fastest and most accurate input technique.

To reason, why does touch performs the best, consider Figure 3.12 that visualizes the points on the screen along which the mouse traverses, and compare it against the visualization of the mouse points (Figure 3.13) and gaze points (Figure 3.14). It can be observed from Figure 3.12 that though the user moves their hand from one target to the other, there are no points recorded along the path. The reason is obvious that the user lifts their finger between selecting two opposite targets, and

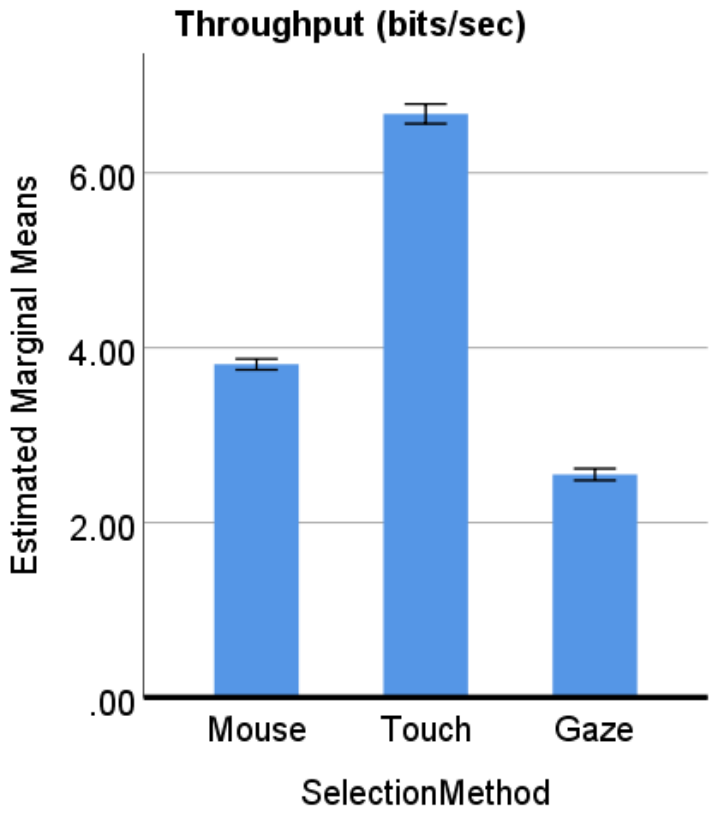

Figure 3.9: Throughput: comparison of throughput between the three selection methods on the standard display.

since the entire screen is within the view of the user the user could move quickly between the targets. This is the primary reason that contributes to highest throughput and lowest movement time. Next, we also observe that touch has the lowest error rate and effective target width. As humans have better motor control over the movement of their hand, the user always hits inside the target, hence the lowest error. Also, the touch-based selections are such that the user always selects the target at its center which leads to lower overshoot and undershoot values. Hence, touch input has the lowest effective target width.

Next, considering why does gaze input has the lowest throughput, we can observe from Figure 3.14 that gaze input is indeed quicker in moving between the source to destination targets, similar to the touch input. There are only a few cursor points along the path connecting the two

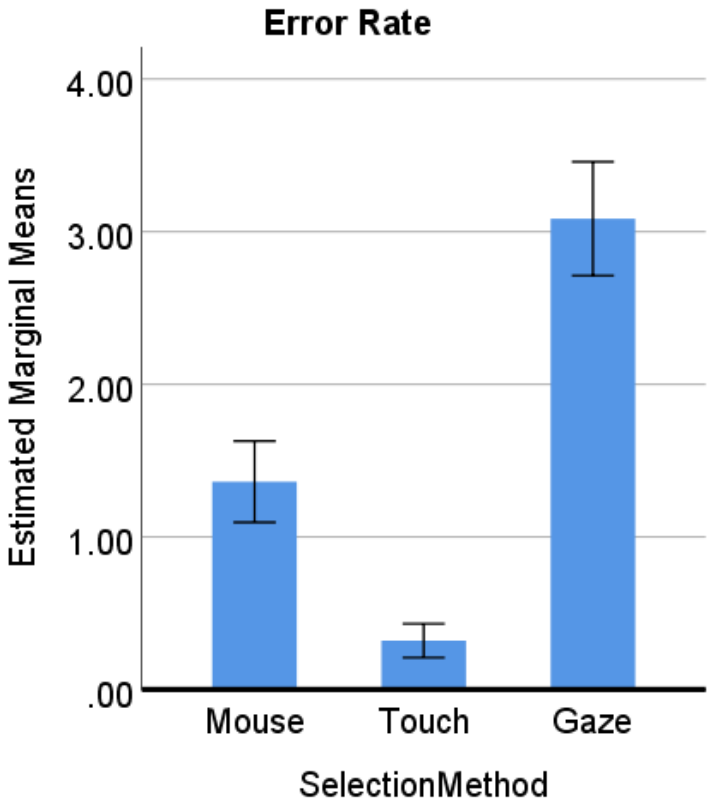

Figure 3.10: Error rate: comparison of error rate between the three selection methods on the standard display.

targets. However, when using gaze input, a maximum time is consumed in stabilizing the cursor (gaze) within the target, and selecting it. Hence, from Figure 3.14 we observe a cluster of points within the target. In addition, the higher effective target width results in lower index of difficulty. Lower index of difficulty coupled with higher movement time results in lower throughput. Hence, when using gaze input, adopting a border crossing strategy as the selection will result in highly efficient interaction.

3.4 A Fitts' Law Evaluation of Gaze Input Compared to Mouse and Touch Inputs on a Large Display

Gaze-assisted interaction has commonly been used in a standard desktop setting. When interacting with large displays, as new scenarios like situationally-induced impairments emerge, it is

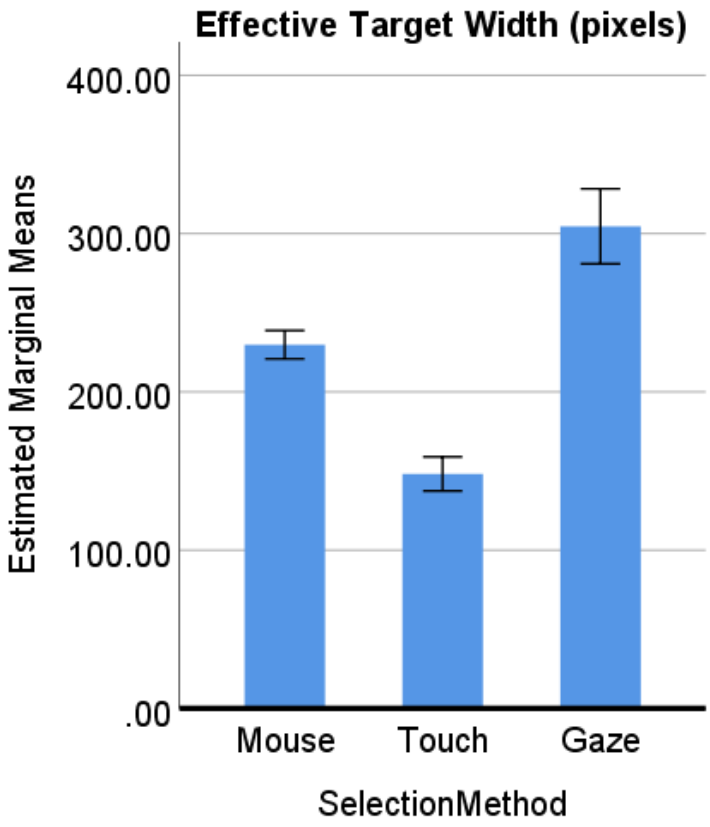

Figure 3.11: Effective target width: comparison of effective target width between the three selection methods on the standard display.

more convenient to use the gaze-based multi-modal input than other inputs. However, it is unknown as to how the gaze-based multi-modal input compares to touch and mouse inputs. We compared gaze+foot multi-modal input to touch and mouse inputs on a large display in a Fitts' Law experiment that conforms to ISO 9241-9. From a study involving 23 participants, we found that the gaze input has the lowest throughput (2.33 bits/s), and the highest movement time (1.176 s) of the three inputs. In addition, though touch input involves maximum physical movements, it achieved the highest throughput (5.49 bits/s), the least movement time (0.623 s), and was the most preferred input.

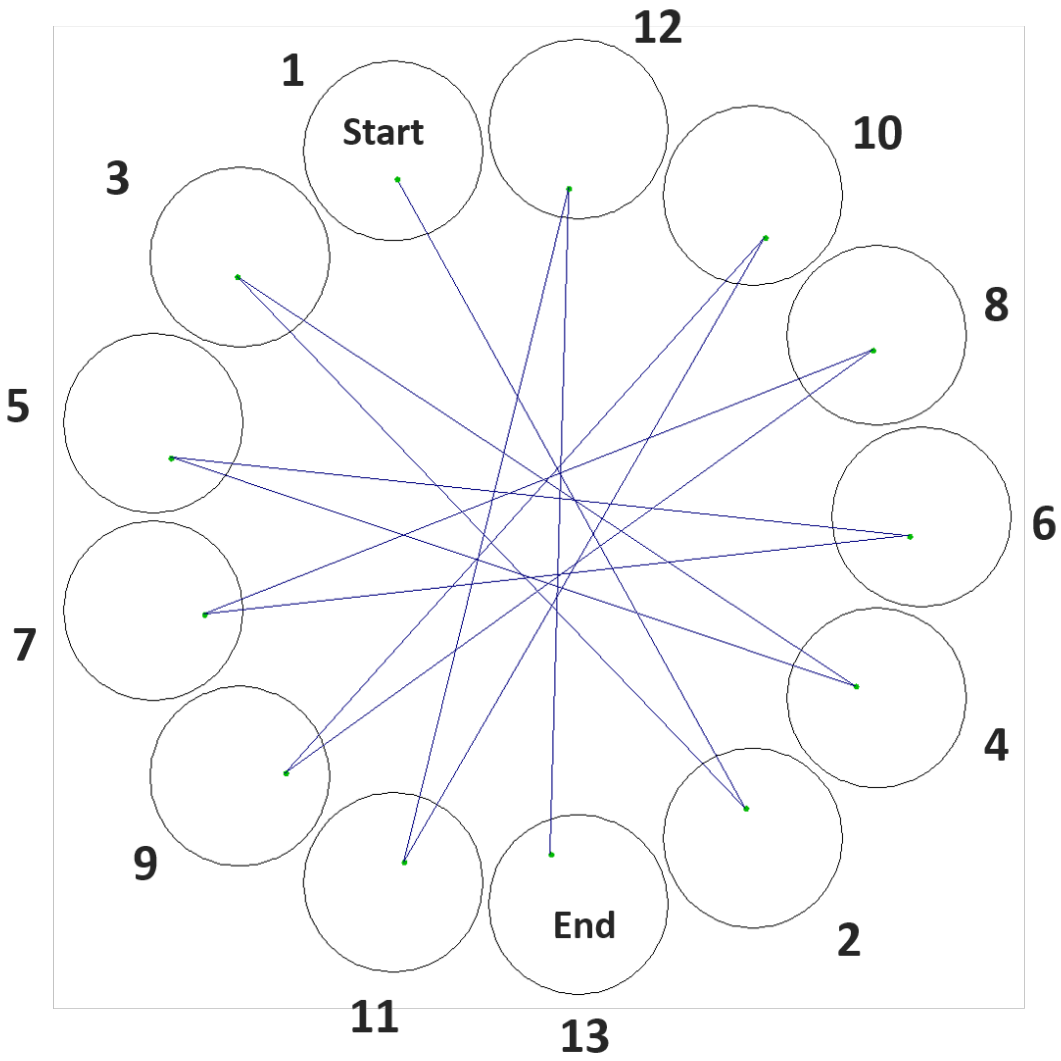

Figure 3.12: Touch input cursor points visualization: though the user moves their hand from one target its opposite target, there are no points recorded along the path (no delay). This is the primary reason that contributes to highest throughput and lowest movement time.

3.4.1 Introduction

Gaze-assisted interaction in a desktop setting as an efficient interaction method or a solution to SIID has been previously explored [142, 30, 143, 144], and also compared against other inputs [131, 132, 133, 130]. Gaze-assisted interaction on large displays has various applications as people can interact with public displays, screens in collaborative spaces, operation theaters, etc.

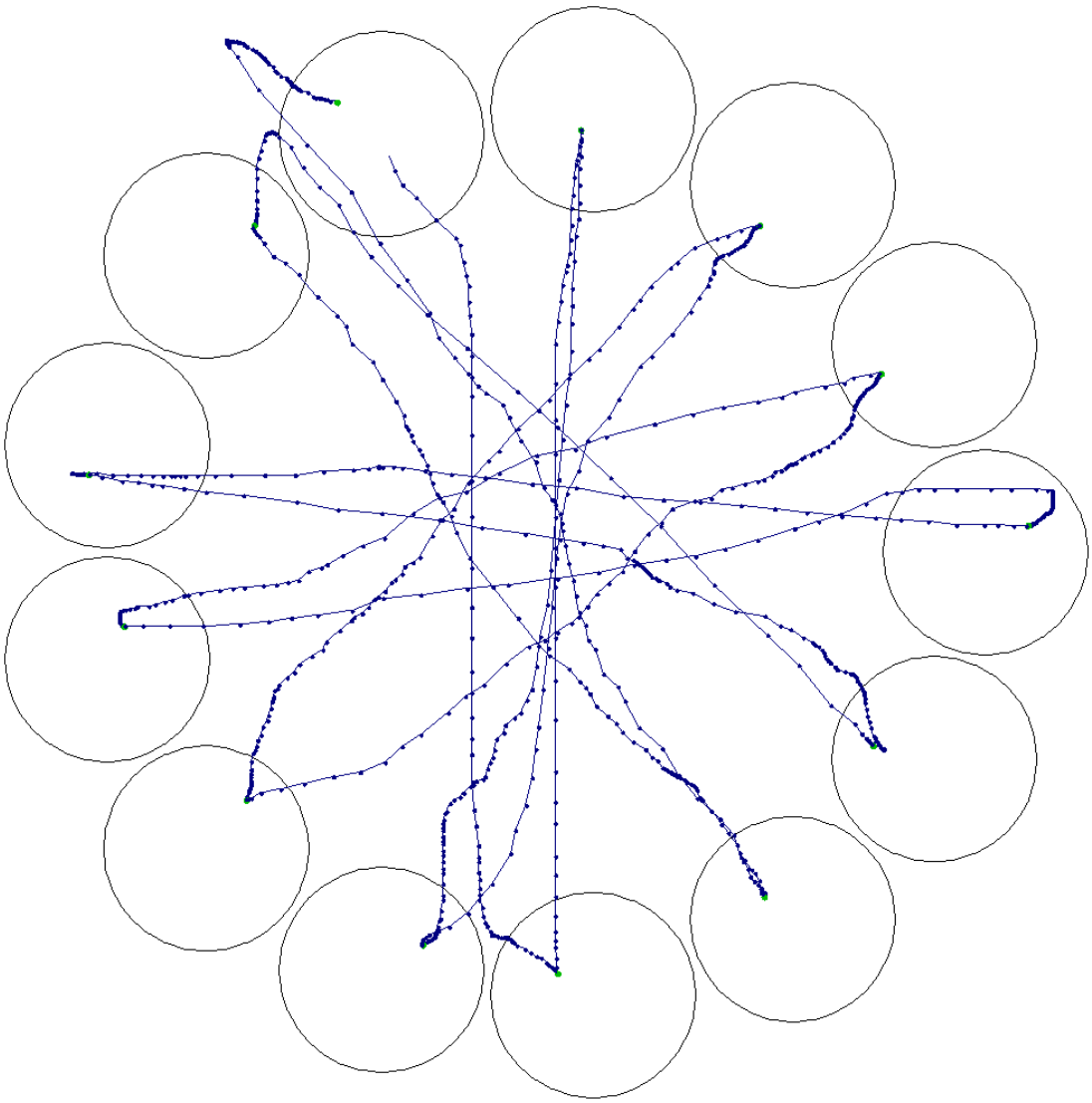

Figure 3.13: Mouse input cursor points visualization: out of the three inputs, when using the mouse input a maximum number of cursor points are recorded as the cursor moves from one target its opposite target. This introduces delay. Also, the user most of the time clicks closer to the edge of the target that results in increased effective target width and lower throughput compared to the touch input.

While there are various examples of using gaze input on large displays [1, 145, 146], its comparison to other commonly used inputs like touch and mouse are limited. To discuss a few relevant works that explored gaze-assisted interaction on large displays, in an upright stance, Hatscher et al., demonstrated the usability of gaze- and foot-based interaction on a large monitor in operation

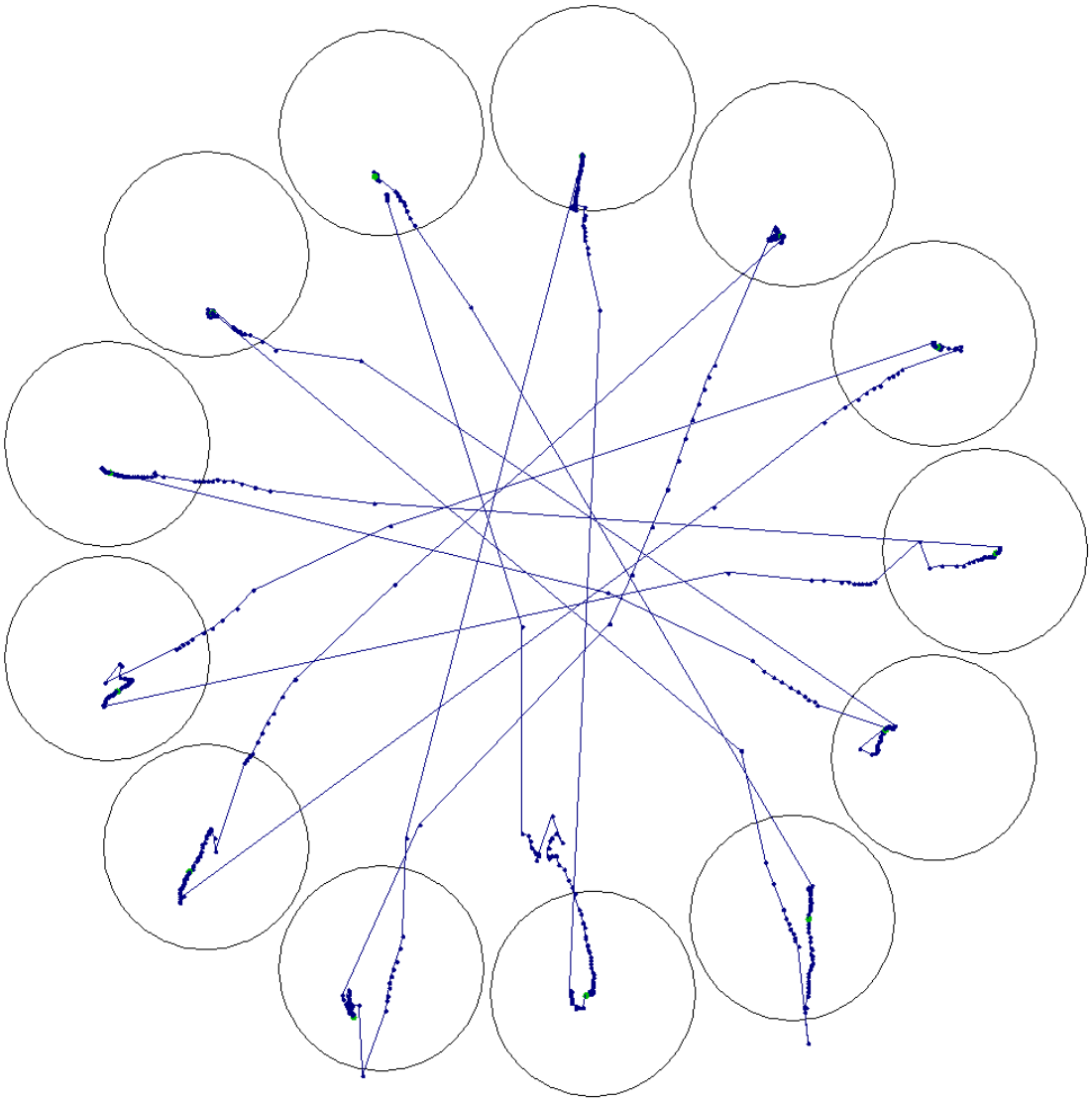

Figure 3.14: Gaze input cursor points visualization: compared to the mouse input there are only a few points are recorded as the cursor moves from one target its opposite target, this behavior is similar to the touch input. However, a maximum time is consumed in stabilizing the cursor (gaze) within the target, and selecting it. Hence, gaze has lower throughput than touch and mouse inputs.

theaters [1]. In this setup, a physician performing minimally-invasive interventions can look and interact with medical image data displayed on the large monitor with gaze input. San Agustin et al., developed gaze-enabled public display (55 inches) where users can interact with high-density information like a digital bulletin board with several notes on top of each other [145].

These works demonstrate that in the cases of SIIDs or from a usability perspective it is more

relevant to use gaze-assisted interaction on large displays ($\sim$ 84 inches). However, the majority of the table mounted trackers (Tobii, EyeTribe, GazePoint, SMI, etc.) are built for 24-inch screens. To achieve the best performance, the optics and IR lights are tuned for the viewing angles that correspond to screens up to 24 inches. This does not mean that we cannot use these trackers on large screens, but they do not track well. Therefore, for someone trying to use these commonly available eye trackers on large displays, it is unknown as to 1) How the accuracy and efficiency of pointing and selecting with gaze input compare against touch and mouse inputs that are used commonly on the large displays? 2) How does the usability of gaze input compare to mouse and touch inputs? 3) What should be size of the targets (this influences the index of difficulty)? 4) While the touch input requires a user to physically move in front of the display to reach different points on the display, and mouse input requires larger movements of the wrist, do users feel touch and mouse inputs stressful compared to the gaze input? The lack of answers to these queries motivated us to conduct a Fitts' Law evaluation that conforms to ISO 9241-9 standardization[4]. In this study we compared three input methods for pointing and selecting targets on a large display. The three input methods we used were gaze+foot, touch, and mouse.

We observe that all the Fitts' Law evaluations we discussed in prior work (Section 3.2) were conducted in a desktop setting, and the participant was always seated while using the various input methods. Contrary to this typical setup, our work performs Fitts' Law evaluation of the gaze input on a large display, while comparing it against mouse and touch inputs. Also, the participants used the three inputs in an upright stance.

3.4.2 Fitts' Law Experiment Design

In accordance with the previous Fitts' law studies on gaze pointing, we used a nominal index of difficulty that ranged from 2.0 to 2.5 [131]. Hence, the amplitude, i.e., the distance to the target we chose were 1250 px and 1650 px, and the target widths were set to 350 px and 450 px. The computation of the index of difficulty is shown in Table 3.3.

[4]https://www.iso.org/standard/30030.html [last accessed Jan 23rd 2018]

Table 3.3: Fitts' Law Evaluation - large display: Amplitude, Width, and Index of Difficulty

Amplitude (px)	Width (px)	Index of Difficulty (bits/s)
1650	350	2.51
1250	350	2.19
1650	450	2.2
1250	450	2.0

3.4.3 Display and Gaze Tracking

The experiment was conducted on a Microsoft Surface Hub [5], a large (84-inch) touch enabled display. We used a Gazepoint GP3 HD tracker for eye tracking. Since the tracking was not accurate enough around the left and right edges of the 84-inch screen, the interaction space was set to 69 inches. The tracker had a manufacturer reported accuracy of 0.5° to 1.0° of visual angle, and had a sampling rate of 150 Hz. However, to test the accuracy of the tracker for our setup with a large display, we recruited 7 (6 M, 1 F) participants, and repeatedly recorded the tracking accuracy values (following the standard calibration) on a 9-points grid interface we developed. A total of 39 accuracy values were recorded, and the average tracking accuracy was 4.6° of visual angle (min 2.6°, max 9°).

3.4.4 Participants and Procedure

For the Fitts' Law experiment we recruited 23 participants (19 M, 4 F) with their ages ranging from 19 to 32 ($\mu_{age} = 23$). Data from 4 participants were excluded since they could not complete the gaze input due to low tracking accuracy. Also, 3 participants who were wearing glasses removed their glasses (for better gaze tracking accuracy) during the experiment. At the beginning of the study, each participant was briefed about the Fitts' Law task and the kind of inputs they would be using for target selection. For each input method (e.g., mouse) the participant completed one

[5] https://www.microsoft.com/en-us/surface/devices/surface-hub/tech-specs

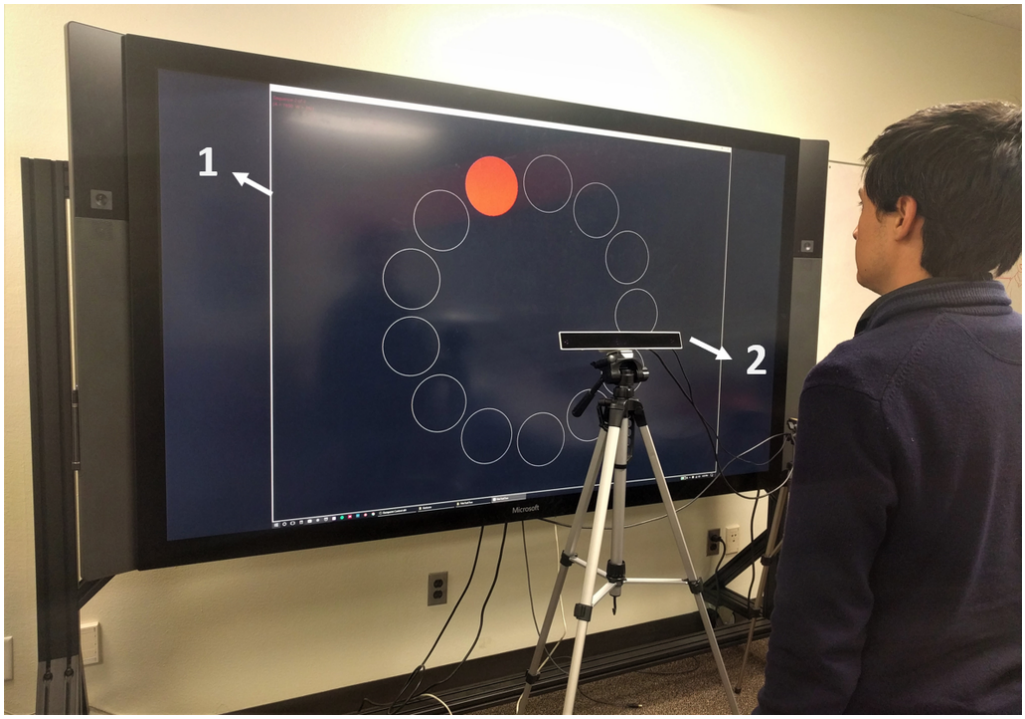

Figure 3.15: Fitts' Law Experiment: a participant, in an upright stance, performing a multi-directional point-and-select Fitts' Law task (1) shown on a large display. Also, an eye tracker is mounted on a tripod (2).

sequence of trials to familiarize themselves with the system before the actual data collection began. The participants used three input methods–gaze+foot, mouse, and touch–for target selection, and the order of input methods used by the participants was counterbalanced according to the Latin square design.

For each input method the participant completed 4 blocks of target selection task, and each block had four sequences of trials as we used two amplitudes (1650 px and 1250 px) and two target widths (350 px and 450 px). In each sequence, there were 13 trials, hence, a total of 3,952 trials (13 trials $\times$ 4 seq $\times$ 4 blocks $\times$ 19 participants) were completed for each input. Also, a total of 11,856 trials (3,952 $\times$ 3 inputs) were completed from all the three inputs. The participants were allowed to rest for a minute between each block, and in the case of gaze input, the participants were re-calibrated if the calibrated stance was disturbed between the blocks.

3.4.5 Results and Discussion

We conducted a one-way ANOVA with replication on the four dependent variables (DVs): 1) movement time, 2) throughput, 3) error rate, and 4) effective target width. The independent factor was the 'selection method' which had three levels: 1) mouse, 2) touch, and 3) gaze. Table 3.4 shows the result of ANOVA on the DVs, and also the mean and standard deviation of the selection methods for each DV.

Table 3.4: Fitts' Law evaluation on a large display: ANOVA and post-hoc analysis (p values highlighted in gray indicate significance at $\alpha = 0.05$).

Selection Method [Ms, Th, Gz]	**Mean**	**Std. Dev**	**ANOVA**
Movement Time (ms)	Ms = 777.884 Th = 623.253 Gz = 1176.54	152.23 139.50 561.14	F(2,606) = 242.196 *p = 0.000*
Throughput (bits/s)	Ms = 3.449 Th = 5.498 Gz = 2.331	0.885 1.487 1.001	F(2,606) = 755.789 *p = 0.000*
Error Rate (%)	Ms = 1.0374 Th = 0.6073 Gz = 8.7298	3.3501 3.0006 9.9549	F(2,606) = 161.763 *p = 0.000*
Effective Target Width (pixels)	Ms = 301.671 Th = 198.462 Gz = 407.084	207.508 272.184 285.128	F(2,606) = 51.659 *p = 0.000*

We observe that the factor 'selection method' is significant ($p < 0.05$) for all the four DVs,

i.e., the value of a DV differs among the selection methods. Out of all the selection methods, 'touch' achieves the highest throughput (5.49 bits/s), consequently it has the least movement time, error, and effective target width. Similarly, 'gaze' input has the lowest throughput (2.33 bits/s), consequently it has the highest movement time, error, and effective target width. Post-hoc tests with Bonferroni correction showed that for DVs movement time, throughput, and effective target width the difference between each pair of the selection methods, (mouse, touch) (mouse, gaze) (touch, gaze), was significant ($p < 0.05$). However, for the DV error, the difference between each pair of selection methods was significant except for the pair (mouse, touch) where $p = 0.32 > 0.05$. Figure 3.16, Figure 3.17, Figure 3.18, and Figure 3.19 compare the means of the three selection methods for each DV.

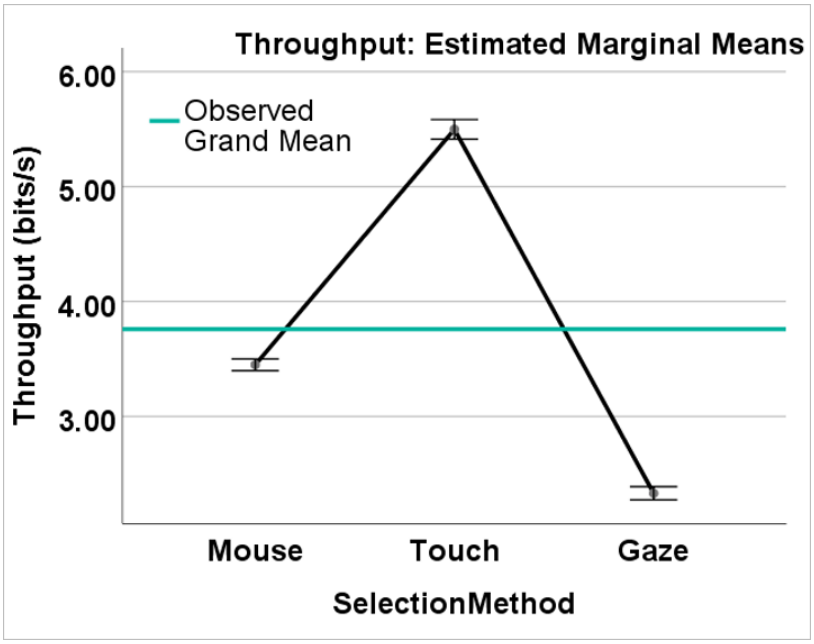

Figure 3.16: Comparison of estimated marginal means for DVs 'Throughput' and 'Error Rate' for the three selection methods. The error bars represent standard error of the mean.

Though theoretically it appears that gaze input should achieve a higher throughput than the mouse and touch inputs, since an eye movement between two distant targets is quicker [133] than the mouse or touch, the results contradict our assumption. This is due to the fact that though the

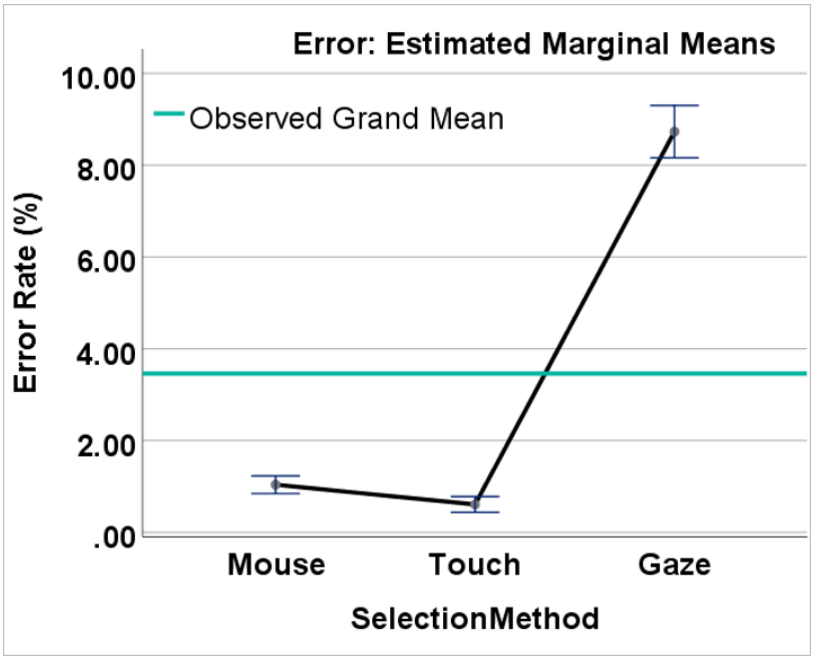

Figure 3.17: Comparison of estimated marginal means for DVs 'Throughput' and 'Error Rate' for the three selection methods. The error bars represent standard error of the mean.

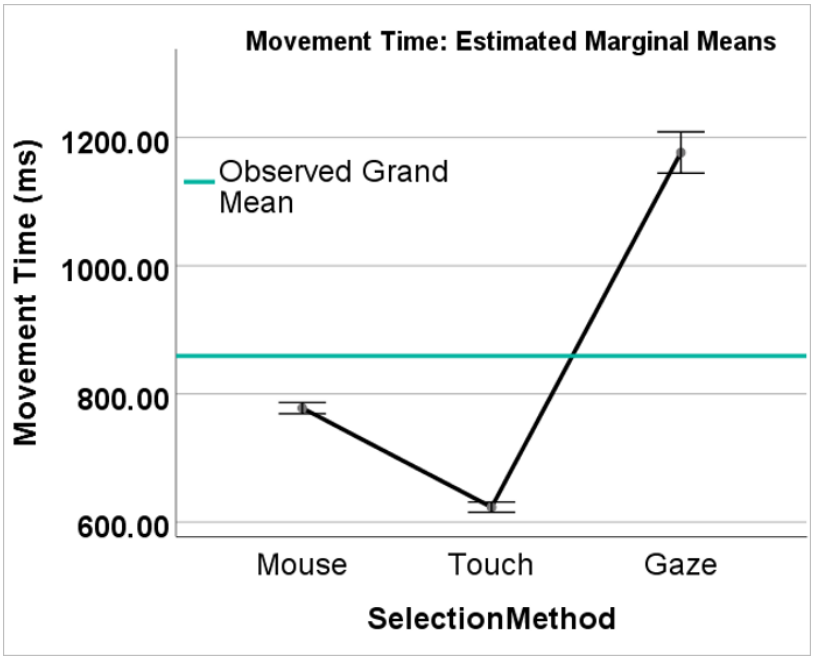

Figure 3.18: Comparison of estimated marginal means for DVs 'Movement Time' and 'Effective Target Width' for the three selection methods. The error bars represent standard error of the mean.

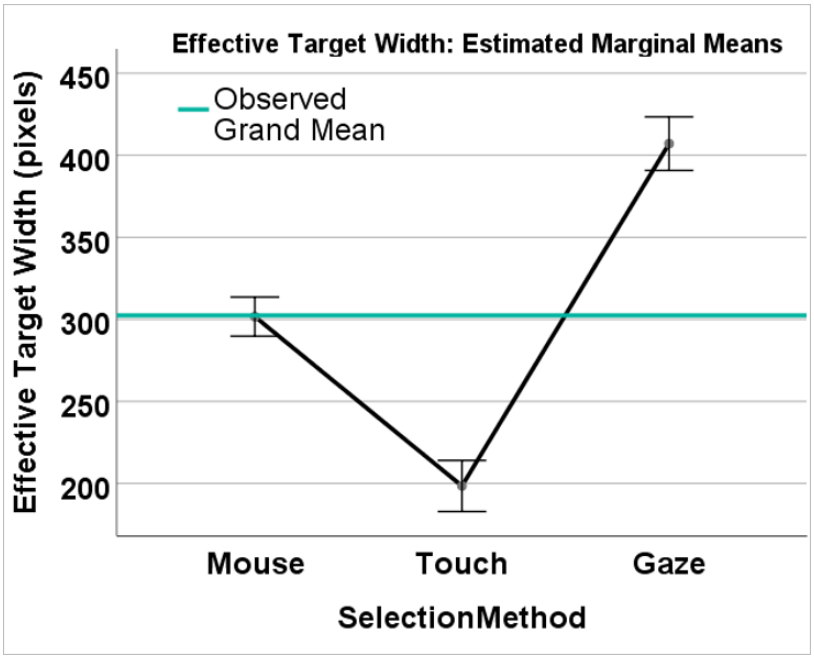

Figure 3.19: Comparison of estimated marginal means for DVs 'Movement Time' and 'Effective Target Width' for the three selection methods. The error bars represent standard error of the mean.

user may move the cursor quickly from target A to the vicinity of target B, placing the cursor exactly on target B and selecting it consumes more time due to lower tracking accuracy on the large display. Therefore, the results suggests that there are two ways to improve the throughput of gaze-based selection on large displays. First, by reducing the index of difficulty of the task, i.e., primarily by increasing the target width, and also by reducing the distance between the targets. Second, by developing eye trackers exclusively for the large displays (larger than 24 inches). Also, in the interviews the participants shared that with gaze+foot interaction it is essential to achieve the synchronization between pointing with gaze and selecting with foot.

Furthermore, we wanted to understand if the users' performance, specifically throughput and error, improve as they progress from block 1 to block 4 for a given selection method (e.g., touch). Hence, we conducted a one-way ANOVA with replication on the dependent variables 'throughput' and 'error' for each of the selection methods, and the independent factor was 'block.' Table 3.5 shows the block mean and standard deviation for various DVs, and corresponding ANOVA results.

We observe that the factor 'block' is significant for DVs gaze, mouse, and touch throughput.

Table 3.5: Fitts' Law evaluation on the large display: ANOVA of block performance (p values highlighted in gray indicate significance at $\alpha = 0.05$).

Block [B1 to B4]	**Mean [Std. Dev]**	**ANOVA**
Mouse Throug	B1 = 3.19 [0.91], B2 = 3.48 [0.80]	F(3,225) = 7.75
-hput (bits/s)	B3 = 3.65 [0.91], B4 = 3.46 [0.86]	*p = 0.000*
Touch Throug	B1 = 4.78 [1.39], B2 = 5.58 [1.22]	F(3,225) = 13.23
-hput (bits/s)	B3 = 5.68 [1.56], B4 = 5.93 [1.51]	*p = 0.000*
Gaze Throug	B1 = 2.09 [1.01], B2 = 2.49 [1.03]	F(3,225) = 5.61
-hput (bits/s)	B3 = 2.29 [0.97], B4 = 2.44 [0.95]	*p = 0.001*
Mouse Error	B1 = 0.91 [3.76], B2 = 0.91 [2.50]	F(3,225) = 1.68
Rate (%)	B3 = 0.70 [2.23], B4 = 1.61 [4.40]	*p = 0.172*
Touch Error	B1 = 0.80 [3.45], B2 = 0.10 [0.88]	F(3,225) = 1.38
Rate (%)	B3 = 1.01 [3.83], B4 = 0.50 [2.90]	*p = 0.250*
Gaze Error	B1 = 10.3 [10.7], B2 = 8.90 [9.23]	F(3,225) = 1.38
Rate (%)	B3 = 7.89 [9.31], B4 = 7.79 [10.4]	*p = 0.250*

The throughput generally increases as the user progresses from block 1 to block 4, which is an indication that the users' performance does improve with more exposure to the selection method. However, we also see that the difference in 'error' between the blocks is not significant for all the DVs. This suggests that the users get quicker in selecting targets with subsequent blocks, however, the accuracy of selection remains the unchanged.

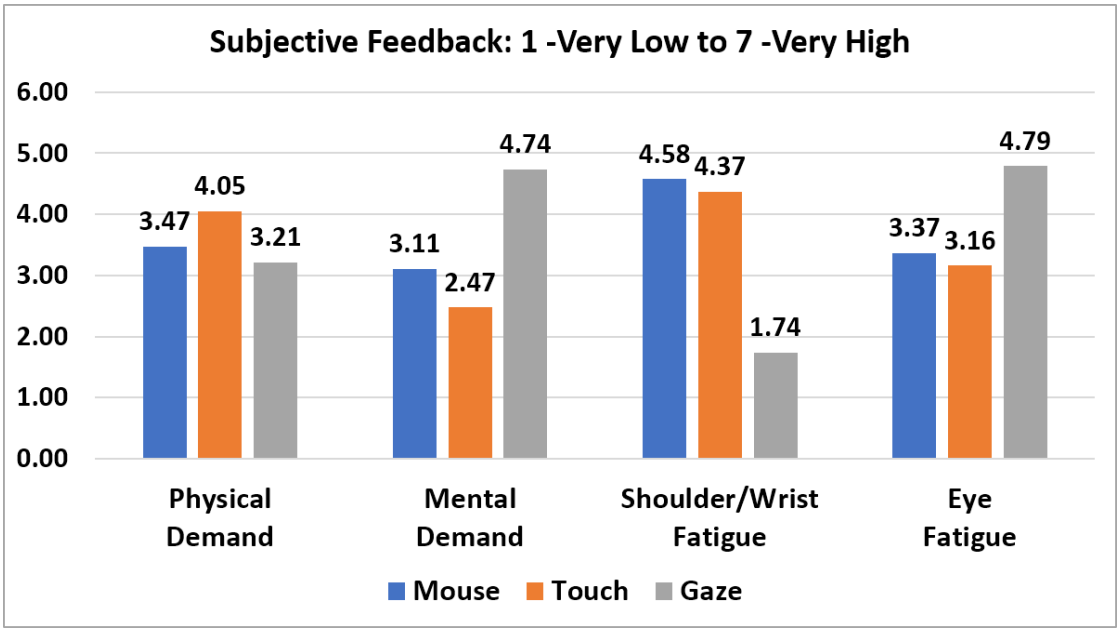

Figure 3.20: Subjective Feedback - lower score is better.

3.4.6 Subjective Feedback

Each participant rated their experience of using the three selection methods on a Likert scale (1-very low to 7-very high) for various physiological measures. Figure 3.20 summarizes the mean value of each measure. As we may expect, touch and mouse results in increased shoulder/wrist fatigue and physical demand on the large displays compared to the gaze input. On the contrary, the gaze input results in increased mental demand and eye fatigue compared to touch and mouse inputs. However, gaze input has an added advantage of enabling hands-free interactions that are crucial in the cases of SIIDs. We further analyzed the ratings using a one-way ANOVA with replication and considering 'selection method' as the independent factor. We found that 'selection method' is not a significant factor for DV 'Physical Demand' $[F(2,36) = 1.506, p > 0.05]$. However, 'selection method' is a significant factor for DVs 'Mental Demand' $[F(2,36) = 19.09, p < 0.05]$, 'Eye Fatigue' $[F(2,36) = 16.128, p < 0.05]$, and 'Shoulder/Wrist Fatigue' $[F(2,36) = 34.283, p < 0.05]$. Also, for DVs 'Mental Demand', 'Eye Fatigue', and 'Shoulder/Wrist Fatigue' where the ANOVA results are significant, post-hoc tests with Bonferroni correction showed that the effect was due

to the difference between (gaze, touch) and (gaze, mouse), but the difference between (touch, mouse) was not significant. In summary, though gaze enables faster cursor movements between the targets theoretically, we found that gaze input had the lowest throughput and highest error rate. On the contrary, although touch results in increased shoulder/neck fatigue, it achieves the highest throughput and lowest error rate, and was the most preferred input.

3.5 A Comparison of Fitts' Law Evaluation of Gaze Input on a Standard and Large Display

In Section 3.3 we discussed the performance on gaze input when compared to the mouse and touch inputs on a standard display. A similar comparison was also performed on a large display in Section 3.4. From both the experiments we found that the touch input has the highest throughput and the least error. Similarly, the gaze input has lower throughput and highest error. However, these experiments did not reveal if these input methods would achieve the same performance irrespective of the device dimensions, or the performance is dependent on the screen dimension. Hence, to understand the dependence of the three input methods on the screen dimension, we performed a mixed two-factor mixed model ANOVA with replication on the three dependent variables: Throughput, Movement Time, and Error Rate.

The two factors (independent variables) we considered were: 1) Screen Size, and 2) Selection Method. The factor 'Screen Size' is a between-subjects factor and it has two levels: 1) standard (up to 24"), and 2) large (up to 84"). 'Screen Size' is a between subjects factor since the participants who performed Fitts' Law task on the standard display did not participate in the evaluation of the three inputs on the large display. 'Selection Method' is a within-subjects factor and has three levels: 1) Mouse, 2) Touch, and 3) Gaze. A total of 3 ANOVA tests were performed, and for each ANOVA test we considered one dependent variable of the three dependent variables. Table 3.6 shows the results of the ANOVA tests for 3 dependent variables.

From Table 3.6, we observe that 'Screen Size' is a significant factor for all the three dependent variables ($p < 0.05$). This suggests that 'Screen Size' does influence the throughput, movement time, and error rate. Generally, throughput is higher on the standard screen than the larger screen. But, the movement time and error rate is higher on the large display than the standard display.

Table 3.6: Fitts' Law evaluation - standard and large display: mixed model ANOVA (p values highlighted in gray indicate significance at $\alpha = 0.05$).

	Screen Size [Standard, Large] Between-subjects Factor	**Selection Method [Mouse, Touch, Gaze] Within-subjects Factor**	**Interaction [SelectionMethod x ScreenSize]**
Throughput	F(1,494) = 63.038 *p = 0.000*	F(2,988) = 1546.49 *p = 0.000*	F(2,988) = 29.84 *p = 0.000*
Movement Time	F(1,494) = 42.361 *p = 0.000*	F(2,988) = 520.723 *p = 0.000*	F(2,988) = 0.893 p = 0.410
Error Rate	F(1,494) = 40.305 *p = 0.000*	F(2,988) = 138.944 *p = 0.000*	F(2,988) = 42.956 *p = 0.000*

Also, we see a strong interaction effect for throughput ($F(2,988) = 29.84, p < 0.05$) and error rate ($F(2,988) = 42.956, p < 0.05$)

Lastly, Figure 3.21, 3.22, 3.23 show the comparison of mean values of throughput, error rate, and movement between for the three input methods on standard and large screens.

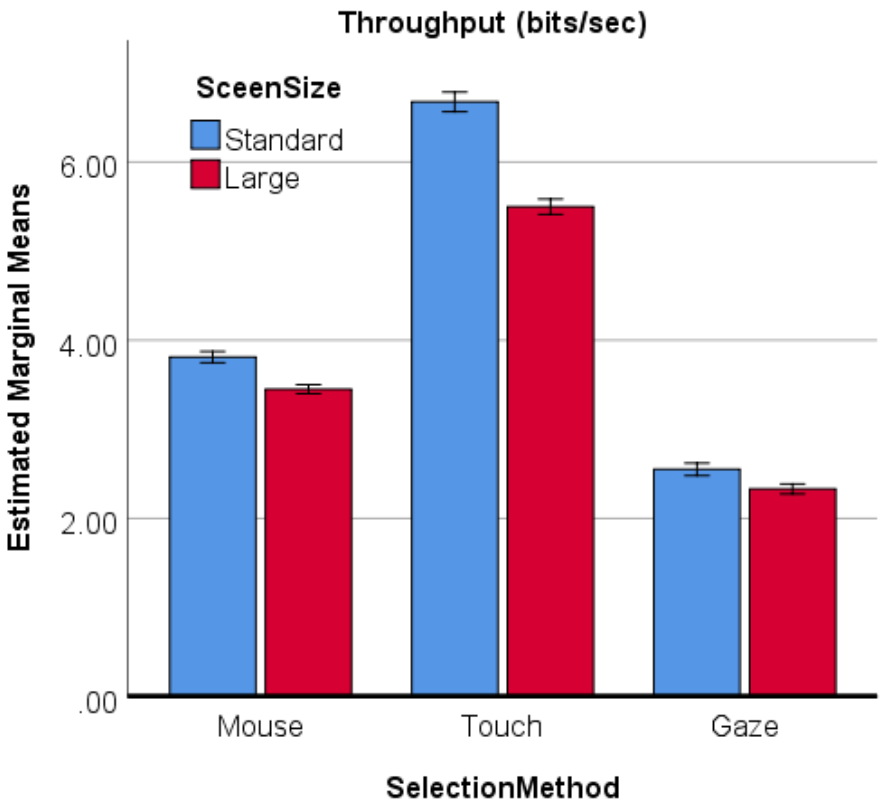

Figure 3.21: Fitts' Law evaluation on a standard and large display: comparison of throughput among the three selection methods.

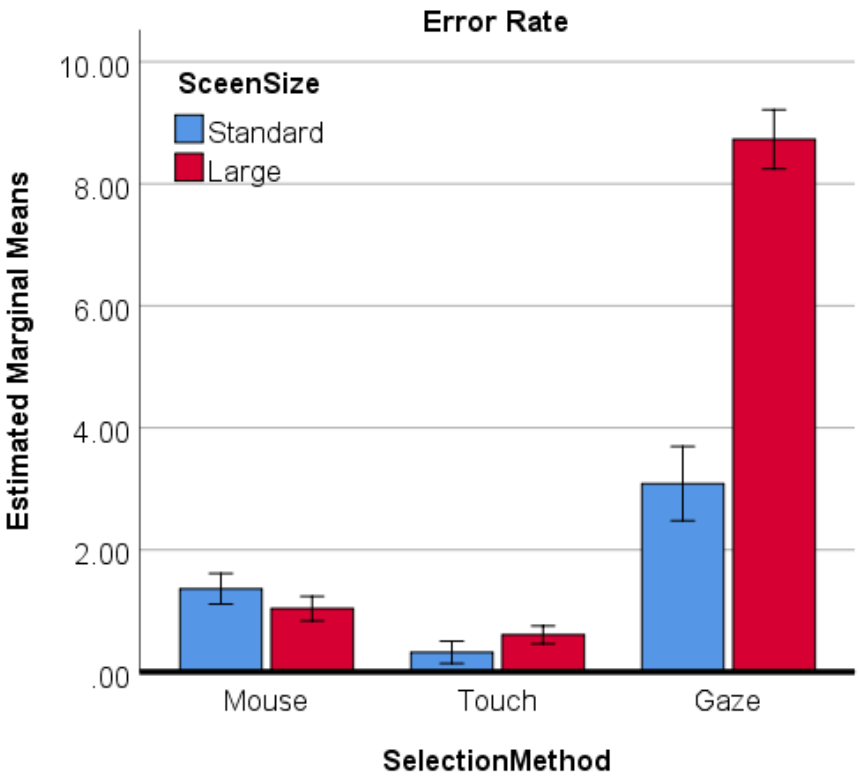

Figure 3.22: Fitts' Law evaluation on a standard and large display: comparison of error rate among the three selection methods.

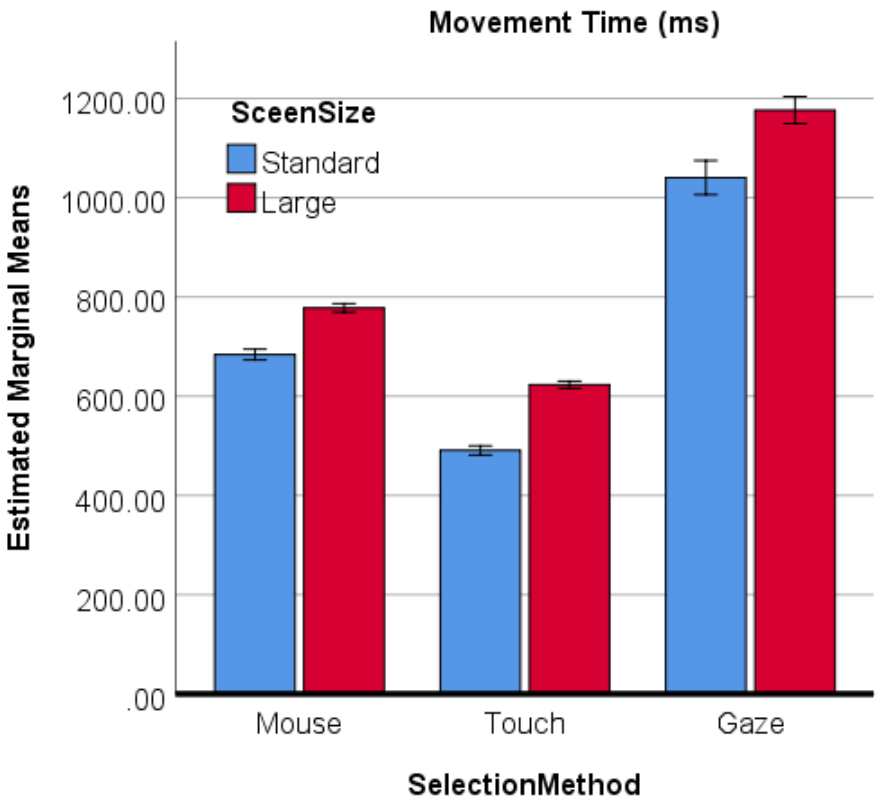

Figure 3.23: Fitts' Law evaluation on a standard and large display: comparison of movement time among the three selection methods.

4. GAZE-ASSISTED TEXT ENTRY METHODS

Text entry on a computer is achieved through different kinds of input modalities. While typing by hand is the most common text entry method, other methods like speech to text, gaze-assisted text entry, to name a few have been explored. In this work, we investigate a multi-modal approach to gaze-based text entry, and this is because of its two significant advantages. First, gaze-based text entry allows for hands-free text entry in the scenarios of situationally-induced impairments and disabilities (SIID), and second, it serves as a primary means of communication for those with either congenital, or trauma related speech and motor impairments. Gaze typing, a method of text entry by gaze, typically uses an on-screen keyboard, where the key selection is achieved by focusing the user's gaze (fixation) on the target character for a predefined duration called "dwell time." The majority of gaze typing systems use dwell time as the trigger for target selection, however, the use of dwell time leads to various usability issues, reduced typing speed, high error rate, and visual fatigue with prolonged usage. Addressing these issues is crucial for improving the usability, and efficiency of gaze typing.

In this regard, we present a dwell-free, multimodal approach to gaze typing where the gaze input is supplemented with a foot input modality. Our combined gaze and foot-based typing system comprises an enhanced virtual QWERTY keyboard (VKB), and a footwear augmenting wearable device that provides the foot input. In this multi-modal setup, the user points her gaze at the desired character, and the foot input selects the character. We further investigated two approaches to foot-based selection, a foot gesture-based selection and a foot press-based selection, which are compared against the standard dwell-based selection[1].

We evaluated our gaze typing system through a comparative study involving three experiments (56 participants), where each experiment used a different target selection method, and had 17 par-

[1] *Parts of this chapter are reprinted with permission from "Gaze Typing Through Foot-Operated Wearable Device" by Rajanna et al., 2016. Publisher and Copyright holder ACM Digital Library, 2016, New York. Conference - ASSETS '16: The 18th International ACM SIGACCESS Conference on Computers and Accessibility Proceedings - doi.org/10.1145/2982142.2982145

ticipants. In the first experiment the participants used dwell-based selection, second, foot gesture-based selection, and third, foot press-based selection for gaze typing. With dwell-based selection the participants used three different dwell times (1000 ms, 700 ms, 400 ms), and we found that the lowest mean typing speed of 6.19 WPM was achieved at 1000 ms, and the highest mean typing speed of 11.65 WPM at 400 ms dwell time. With foot-based selection methods, the participants achieved the lowest mean typing speed of 10.3 WPM using foot gestures, and 11.41 WPM using foot press-based selection in the first session. By the end of the fourth session, the highest typing speed of 13.82 WPM was achieved with foot gestures, and 14.98 WPM was achieved with foot press-based selection. Considering the top typing speeds and associated errors, ANOVA tests revealed that the difference in the typing speeds between the three selection methods is significant, however, no difference was found in the error rate.

Overall, considering both the typing performance and usability aspects, foot-based activation methods are highly preferred than dwell-based activation. Furthermore, toe tapping is the most preferred foot gesture of all the four gestures (toe tapping, heel tapping, right flick, and left flick) we used in the study. Lastly, when using foot-based activation, either foot gestures or foot press, the users quickly develop a rhythm in pointing at a key with the gaze and selecting it with the foot. This familiarity reduces the errors significantly, and also as shared by the participants, the foot interaction was so natural that within typing a few phrases they did not have to consciously remind themselves to use the foot. We believe our findings would encourage further research in leveraging a supplemental foot input in gaze typing, or in general, gaze-assisted interactions. In addition, our findings on using foot gestures and foot press-based selection as input methods would assist in the development of foot-based interactions and devices.

4.1 Introduction

Text entry is one of the basic operations performed on a computer, and is achieved through various input modalities. While text entry through typing on a physical keyboard is primarily used, the keyboard-based text entry has limitations under certain circumstances. These circumstances can be classified into two groups: 1) situationally-induced impairments and disabilities

(SIID) [113, 147, 148], and 2) physical impairments and disabilities [24].

In case of the SIIDs, a user's hands are assumed to be engaged in other tasks, and hence unavailable for typing on a physical keyboard. For example, a surgeon performing an operation, a musician playing music, a factory worker wearing thick gloves or with greasy hands, driving a vehicle, etc. Similarly, in the case of physical impairments, either by birth or injury, the hands are unavailable, or the user may not have enough control over their hands to type on a physical keyboard. In both of these scenarios, gaze typing plays a crucial role in assisting these individuals to enter text on a computer through their eye movements. Speech to text input also serves as one of the viable solutions in the above scenarios when using a physical keyboard is not possible. However, speech to text input has various limitations because of its accuracy and applicability. For example, speech to text can not be reliably used in noisy environments (public places), and the accuracy of speech recognition is constrained by the accent of the user. Importantly, while entering confidential information (passwords, unique IDs, etc.) and personal details (age, preferences, etc.), speaking the details out loud may not be an acceptable solution. Additionally, users with speech impairment can not use the speech to text input.

Text entry by gaze has gained significant focus because of its robustness, applicability, and the ability to customize the system to be appropriate for different kinds of impairments (accessible technology). The flexibility of gaze-assisted text entry resulted in multiple methods, where each method uses a unique strategy for text input. Irrespective of the text entry strategy used, all gaze-assisted text entry methods use gaze as the primary input modality. Among the various methods available for text entry by gaze, one method that has received maximum focus is "Gaze Typing." [24]. Gaze typing uses a virtual keyboard on the monitor, and to enter a character a user fixates his or her gaze on a specific key for a duration of time referred to as the dwell time. A constant fixation for the duration of dwell time, confirms the user's intent to press the target key.

In addition to gaze typing other gaze-assisted text entry methods are single character gaze gestures, continuous gaze gestures, and gaze switches [24]. Gaze typing through single character gaze gestures (discrete gaze gestures) uses a unique gesture for each character. A user enters a

character by drawing a character like shape (set of strokes) defined for the character [149]. The system interprets a set of strokes drawn by the user into a specific character. Furthermore, text entry based on continuous gaze gestures uses continuous pointing, zooming, and panning actions of the gaze to select the target characters. Text entry systems that use continuous gaze gestures have shown to work with low accuracy eye trackers and on smaller displays [150]. Lastly, the method of text entry with gaze switches uses a matrix of letters, and the focus moves automatically from line to line. To enter a character, first, a line is selected by blinking the eyes, and this makes the focus to switch to a single line scanning mode. Now the focus starts scanning each character in the selected line. In the line scanning mode, the user selects a character by blinking again [24]. Out of all these gaze-assisted text entry methods, “Gaze Typing" is a direct and majorly used text entry method [24].

Gaze typing systems can be broadly classified into two categories: 1) dwell-based systems and 2) dwell-free systems. Dwell-based gaze typing systems use visual fixation on the target key for the duration of dwell time, as the selection method. The duration of the dwell time varies among systems, and commonly varies from 400 ms to 1000 ms [151, 76]. In dwell-free gaze typing systems, gaze is still used to point at the target key location. However, the selection of the target key is triggered by a supplemental input modality, or by intelligently populating the probable words based on gaze positions on the keyboard [152, 94]. To type with dwell-free typing systems that do not use a supplemental input modality, the user gazes over the target keys of the word but does not fixate on them. The system uses an internal dictionary to generate a possible set of words which are then presented to the user. Now, the user fixates on the correct word for the duration of dwell time to select it. We will comment more on dwell-based, and dwell-free gaze typing systems in the prior work Section 4.2.

Existing dwell-based, and dwell-free gaze typing systems have various limitations that affect gaze typing efficiency. Firstly, considering dwell-based gaze typing systems, they place a high demand on the user’s attention, and sometimes results in inadvertent selection of keys due to the Midas Touch [65]. The issue of Midas Touch states that when eye position on the screen is used

as a direct substitute for the mouse, wherever the user fixates during a visual search, the point gets activated. This unintentional, or indiscriminate selection is inefficient, and leads to user frustration [65]. Secondly, text entry can be slow based on the dwell duration used, typical typing speeds are below 10 wpm [153, 112]. Additionally, the user is constantly thinking about which character to select next (cognitive processing), validate if the typed letters are correct, fix errors, etc., which generally demands more time [151]. Another drawback of dwell-based typing is that a single dwell time is not suitable for all users, hence it is hard to find the optimal dwell-time. If a shorter dwell time is used (150 to 400) to improve the typing speed, the user is constantly forced to perform a visual search for the target key without inadvertently fixating for too long before finding the correct target. [154, 151]. This results in more errors and a higher overproduction rate [74, 154, 151]. But, a longer dwell-time, though increases accuracy, reduces the typing speed, limits quicker users, and increases visual fatigue [74, 154, 151]. Also, some users simply can not focus at a point for a sufficiently long duration [155].

Similarly, dwell-free systems that use extra saccades for gaze typing only marginally increase the typing speed, with a major downside of increasing the keystrokes required per character [24]. Dwell-free typing systems that do not use a supplemental input rely on language modeling, and word and character prediction to support text entry. However, the systems that use word prediction induce cognitive, and perceptual load on the user. The user constantly switches focus from the keyboard to scanning predicted list of words to see if the desired word is populated. Hence, though word prediction may reduces keystrokes per character (KSPC), the improvement in typing speed achieved is minimal, and in some cases worse than non prediction system [24, 156]. Additionally, the prediction system consumes precious screen real estate to populate the word list. Another limitation when using word prediction is the lack of an extensive library. For this reason, these systems will under-perform when typing unknown words like family names or local places. Hence, they are limited for practical use in free communication, but work well under constrained input conditions [157].

While considering the applicability of gaze typing in the scenarios of SIIDs and impairments,

it is essential to address the current limitations, and improve the gaze typing experience, usability, and efficiency. In this research, we present a dwell-free, multimodal, gaze typing system that uses a supplemental foot input. The foot input is achieved through a footwear augmented with a wearable device, which communicates with the central system over a Bluetooth connection. We also present an enhanced virtual QWERTY keyboard where the key layout is customized to maximize the selection area for a few selected keys, and hence supports ease of interaction. Gaze and foot-based point-and-click interactions on a desktop has been explored by Rajanna et al. [141], Klamka et al. [158], and Hatscher et al. [1]. However, when considering gaze typing, except for a preliminary study by Rajanna et al. [94], there exists no study that thoroughly investigates gaze typing with a supplemental foot input. Hence, we wanted to investigate gaze typing using both foot gestures and subtle foot press-based selection methods. Our gaze typing system consists of three modules: 1) Virtual Keyboard (VKB), 2) Gaze Interaction Server, and 3) Foot-Operated Wearable Device. A pictorial depiction of the system is shown in Figure 4.1. The rational for choosing a supplemental foot input, implementation details, and the research questions are discussed in further sections.

Through the system evaluation, we wanted to determine the feasibility of a gaze and foot-based typing system, and compare its efficiency to existing gaze and dwell-based typing systems. While a new gaze typing method, in our case a gaze and foot-based typing, that improves typing efficacy at the cost of increased physical and mental demand would not be acceptable. Hence, in addition to aiming for improved gaze typing efficiency (typing speed and reduced errors), we focused on improving the overall gaze typing experience and usability through using foot input. In particular, we were interested in addressing the performance and usability issues associated with existing dwell-based and dwell free typing systems. Furthermore, we were interested in studying how users use various foot gestures, their preferred gesture, and how the performance of foot-press based selection compares to foot gestures. Lastly, we wanted to understand the learning effects of the gaze and foot-based typing system on the user. The specific research questions are discussed in Section 4.3. To evaluate these objectives, we conducted a three phase, comparative user study involving 51 participants.

Overall, our results suggest that an efficient gaze typing system that addresses most of the usability issues can be achieved by incorporating a supplemental foot input modality. Furthermore, the users appreciated the greater control over the interface with gaze and foot-based typing as inadvertent key selections were significantly reduced. We found that by dividing the responsibility, i.e., fixating on a key and its selection between two separate input modalities, helps to achieve a more usable and robust gaze typing system. Also, we learned from the user studies that the key to achieving a higher typing speed (WPM) on our system is the ability to synchronize fixating on a key, and pressing the pressure pad to activate it.

The remaining sections of this chapter are organized as follows. We will review various gaze assisted text entry methods, their advantages, and limitations in the Prior Work Section 4.2. The Research Questions Section 4.3 lists all the goals and aspects that are explored in this work. Design Motivation Section 4.4 discusses the rationale behind using a supplemental input modality, specifically foot input, for gaze typing The section also discuss how we arrived at the design of the foot-operated wearable device that enables foot input. System Design and Implementation Section 4.5 discusses the system components, and implementation details of the virtual keyboard and the two foot-based selection methods. Experimental protocol for the three experiments we conducted are discussed in the Experiment Design Section 4.6. Results from our experiments are discussed in Section 4.7, followed by a Discussion Section 4.8 that interprets the results and presents inferences. Sections 4.9 presents the Conclusion and compares the performance of foot-based selection with dwell-based selection. In addition, we also discuss how users use different gestures when they have the freedom of using and switching between multiple gestures.

4.2 Prior Work

Research in gaze-assisted text entry dates back more than 20 years [112]. There have been various gaze typing methods developed that use gaze as the primary input modality. In a minority of cases gaze has been combined with a supplemental input modality. Existing gaze-assisted text entry methods can be classified across four categories based on the approach used.

4.2.1 Gaze and Foot-based Interaction

Studies investigating the usability and permanence of gaze and foot-based interactions are limited. Rajanna et al. [141] presented GAWSCHI, a gaze and foot-based interaction framework that enables accurate and quick gaze-assisted interactions, The authors demonstrated that the gaze and foot-based interactions are as good (time and precision) as mouse-based interactions as long as the dimensions of the interface element are above a threshold. Klamka et al. [158] combined gaze input with a foot pedal to perform secondary mouse tasks like panning and zooming. The authors found that gaze-supported foot input allows for user-friendly navigation and is comparable to mouse input. Hatscher et al. [1] demonstrated the usability of gaze- and foot-based interaction on a large monitor in operation theaters. In this setup, a physician performing minimally-invasive interventions can look and interact with medical image data displayed on the large monitor with the gaze input.

4.2.2 Gaze Typing

As discussed in the Introduction Section 4.1, gaze typing uses an onscreen keyboard for text entry. In this method, the user is provided with live feedback of the key (alphabet, numeric, symbol, etc.) they are currently focusing on. This is done by either highlighting the key on the VKB, or by moving the cursor over the key on VKB as the user looks at it. Once the user's gaze is fixated on a key, the selection is performed through multiple ways like dwell time, blink, winks, or muscle activity using electromyography [72]. Majaranta et al. [73] studied how auditory and visual feedback affects eye typing. In their method, for "visual only" feedback, the symbol under focus shrinks as the dwell time elapses. The other feedback combinations included, speech only, click+visual, and speech+visual. The authors found that the kind of feedback method influences the text entry speed and error rate. The authors also found that auditory feedback is more effective than visual feedback. Majaranta et al. [74] further studied the effects of adjustable dwell time on the performance of gaze typing. Most gaze typing systems use a fixed dwell (450 ms to 1000 ms) time for key selection, and on average achieve a speed of 5 to 10 words per minute [74]. Using

adjustable dwell time, the authors found that novices' text entry rate increased from 6.9 wpm in the first session to 19.9 wpm in the tenth session. Furthermore, the dwell time decreased from an average of 876 ms to 282 ms, and the error rates decreased from 1.28% to 0.36%.

Hansen et al. [75] presented "GazeTalk," a gaze typing system that integrates both word and character prediction features. Character prediction feature dynamically changes the characters presented to a user based on the character already entered. The suggested characters are the most probable characters which are likely to follow the previous character. Word prediction works based on the same principles as character prediction. At any point the system displays eight most likely completion characters that are spread across different cells for easy access. Also, as shown by Hansen et al. [76], when the users are provided with predicted words they tend to choose the closer word (wrong ending), delete the last few characters of the word, and then type the right ending to fix the word. This method results in an increased over production rate.

Beelders et al. [159] implemented a gaze and speech-based multimodal system for gaze typing. The authors integrated a gaze and voice controlled online keyboard into Microsoft Word. To type a character, the user focuses on the desired character and then issues a verbal command in order to type the character. The authors showed that the physical keyboard is superior to gaze and voice-based text entry. Additionally, the typing speed of the gaze and voice-based system did not improve as the users were tested through multiple sessions (increased exposure). The users achieved an average typing speed of 0.2 to 0.3 characters per second.

Hansen et al. [76] conducted a comparative study by varying the selection method on a Danish on-screen keyboard (GazeTalk) with 10 large buttons. The character on each button changes according to a character prediction algorithm. The authors showed that dwell time selection on keys is a little slower, and has a higher overproduction rate than click-based selection. Based on the overproduction rates from the experiment, the authors report that dwell time selection introduces more erroneous actions and less efficient strategies. Also, when using a dwell time of 500 ms, each character entry takes 150 ms more time than mouse selection. The authors further found that a major problem with dwell-based selection is that the participants cannot just leave the mouse pointer

anywhere on the screen as they normally would do with mouse. Also, if they forgot to “park" the mouse in a text field, it would activate the button below it in inadvertently. This again adds to the overproduction rate, which is not seen with mouse selection. Also, a large group of people found that a dwell time of 500 ms is too short, especially in the beginning of the experiment. A comfortable speed was 750 ms.

MacKenzie et al. [152] implemented a dwell free gaze typing system with word and letter prediction. Like word prediction systems, the letter prediction algorithm highlights three probable next letters on the keyboard. The authors showed that letter prediction is as good and in some cases even better than word prediction. Also, when using word prediction, if the first letters of a word were wrong, it lead to increased error. The error rates were reduced when using the combination of a fixation algorithm and letter prediction.

Urbina et al. implemented a gaze typing interface based on the pi menus [155]. This was presented as an alternative to a single character entry and dwell time selection in gaze typing. The interface adopted [160] and included bigram text entry with one pie interaction. To enter a character, the user would move their gaze such that it crossed the selection border of the pi slice containing either a group of letters or a single letter. The authors reported that text entry with bigram and bigrams derived by word prediction has large advantages over single character entry methods in terms of speed and accuracy.

Pedrosa et al. [98] presented “Filteryedping," a dwell-free gaze typing system. The interface filters out unintentionally selected keys from the sequence of letters looked at by the user. Finally, a candidate list of words are presented to the user for selection of the right word. The results showed that the system allowed a typing speed of 15.95 words per minute after 100 min of typing. All the gaze typing systems discussed so far, either dwell-based or dwell-free, are limited in their efficiency or usability as elaborated in Introduction. Hence, a gaze typing system that addresses the usability issues while also achieving a good efficiency is critical.

4.2.3 Gaze Gestures

Text entry systems based on discrete gaze gesture leverage the principles of sketch recognition, where a few semantically associated strokes are interpreted as a shape [161, 30]. In this method, every character is encoded into a set of strokes such that each set is uniquely identified with an alphabet. To enter a character, the user draws strokes on a canvas in the order specified, and the system recognizes these set of strokes as a character [24]. This method needs no dwell time, though a short dwell time can be used to begin the gesture; also, short fixations are required to distinguish between the start and end of strokes. Advantages of this method include, strokes are independent of their position on the screen, and they rely only on related change in gaze direction. Since gaze gestures are independent of their location on the screen, the character recognition is not susceptible to calibration errors or inaccuracies. However, text entry systems that use discrete gaze gestures generally have a lower typing speed. Some well known gaze gesture implementations are discussed below.

Wobbrock et al. [149] presented “EyeWrite," a gaze gesture-based text entry system. In this setup, the system uses a character chart that encodes letter like gesture sets for each character. To enter a character, the user is presented with a square-framed canvas with four corners; the user has to draw a letter using four corners to map the character. As soon as the stroke crosses the line delimiting the corner, it is recognized as a specific alphabet. With “EyeWrite" the users achieved a speed of 5 wpm on average, whereas users achieved a speed of about 7 wpm on a virtual keyboard with dwell-based selection. But with “EyeWrite", users have significantly fewer errors in the final text, while the number of errors corrected during the entry is comparable to keyboard.

Bee et al. [162] adopted “QuickWriting," an interface originally developed for text input, for gaze controlled text input. This writing interface matches the continuous nature of visual gaze, and also enables text entry without requiring dwell to execute a command. The authors state that cursive writing comes closer to the nature of human visual gaze. Also, the authors showed that their system can compete with the common gaze-assisted writing system (GazeTalk or pEYEwrite) without word completion. Porta et al. [163] presented “Eye-S," a hidden interface to input text. The

system uses hot-spots, a collection of nine areas on the screen that are used to create letters (and general eye gestures) through a sequence of fixations. The hot-spots are made invisible, hence they do not interfere with the interface of any other application. The authors also mention that their system achieves a slower writing rate, when compared to other text input systems, which use visible graphical elements as targets.

David et al. [164] presented “Dasher," a system that uses continuous gaze gestures and language modeling to support efficient text entry. Initially, all the characters are aligned on the right-hand side of the interface in the alphabetical order. To select a character the user points the cursor at the desired character with their gaze, and this causes dynamic zooming of the area around the target character as the character moves to the left-hand side. Once the character that was pointed crosses a vertical delimiting line, it is registered by the system, and simultaneously probable next characters start appearing in the zoomed area. Dasher achieves a typing speed of up to 34 WPM. Another system that uses continuous gaze gestures is “StarGazer," presented by [150]. StarGazer uses a circular keyboard where all the letters are placed on two concentric circles, and continuous zooming and panning gestures are used to select the desired character. After five minutes of practice, the novice users achieved a typing speed of 8.16 WPM on StarGazer.

4.2.4 Gaze Switches

Text entry based on gaze switches use eye blinks or winks to confirm a user selection. Here the system does not track gaze to make a selection, but just watches for blinks or winks for a confirmation from the user. The system uses an alphabet matrix, where the focus moves from one line to other in a top down pattern. A user selects a line with a blink, and only the selected line is further scanned allowing to select a specific character, which requires another blink.

Grauman et al. [165] presented “BLINKLINK" a tool that automatically detects a user’s eye blinks, and also measures the duration. This system was developed as an alternative modality for people with severe disabilities to interact with a computer. The system recognizes voluntary long blinks as mouse clicks while involuntary short blinks are ignored. A sequence of long and short blinks are interpreted as a semiotic message. The system was tested with a scanning spelling

program using the same method described above. The authors report that a communication rate of about 9 seconds per letter was achieved.

Kate et al. [166] created an eye-controlled aid for nonvocal patients with paralysis. The system uses an eye switch with the partially defected eye for the selection of letters. The authors tested different selection procedures with and without visual feedback. In general, visual feedback reduced the number of selection errors significantly. Fejtová et al. [167] created “I4Control" which enables non-contact control of a personal computer through eye (or head) movement. The solution emulates mouse movement, and the cursor can be placed on a software keyboard on the screen. Selection of the key is achieved by an eye blink. In this setup, the authors compute the direction of the cursor movement as an appropriate deviation from the balanced position.

4.3 Research Questions

By building a gaze typing system with a supplemental foot input, we wanted to answer the following research questions.

1. Can users coordinate their gaze and foot input to enter text on a computer. In other words, is gaze and foot-based dwell-free typing system feasible?

2. Does a gaze and foot-based typing system achieve higher typing speed (WPM) and lower error rate than gaze and dwell-based typing system?

3. How the typing speed and error rate of foot gesture-based selection compare to foot press-based selection?

4. What foot gestures do the participates find convenient to use and why?

5. Do participants select a single foot gesture and use it throughout, or do they switch between using different gestures, to prevent stress on the foot?

6. Does a gaze and foot-based typing system provide user friendly interactions by addressing the interaction issues found with gaze typing systems that use only gaze but no other supplemental inputs?

7. Do users learn over repeated usage of the system, and develop a familiarity with the gaze and foot-based typing? Does this result in improved performance?

8. Does a gaze and foot-based typing system induce physical strain and cognitive load on the user?

9. From a usability perspective, what are the advantages and disadvantages of using a supplemental foot input with gaze typing?

4.4 Design Motivation

We believe that substituting dwell-based selection with a new modality is one of the key factors in addressing multiple limitations associated with gaze typing. Hence, in our solution, we intended to substitute dwell-based selection by a direct and instantaneous method of target key selection. Our solution leverages an input from the foot as a supplemental (selection method) input along with gaze to type on a VKB. We strongly believe using a supplemental input along with gaze input enhances user experience over just using only the gaze input. In addition, we hypothesize that the additional input modality does not strain the user under normal usage conditions, and is similar to using a mouse. The other advantage of the foot input over dwell is the elimination of the need to adjust dwell time for different users, or for the same user. Hence, our multimodal approach to gaze typing reduces the load on the visual channel, and distributes the responsibility to multiple input channels (foot and eye). The same can not be achieved by just using the dwell or blink based selection.

The next question we considered was why the foot as a supplemental input, and why not other input modalities were considered? Velloso et al. [168] discussed successful applicability of foot input in various use cases in human-computer interaction. Prior works have combined gaze and speech for text entry [159], which has resulted in limited typing speed. As discussed previously, works like [141, 158, 169, 1] have already explored gaze and foot-based coarse point and click interactions on a computer, and they found, the foot is one of the promising supplemental inputs to be combined with gaze. However, there exists only a preliminary study on gaze and foot-

based typing [94]. The lack of an extensive study in gaze and foot-based typing motivated us to thoroughly study the foot input in gaze typing achieved through either distinct foot gestures or subtle foot press actions.

The foot-based interaction, either foot gestures or foot press, was achieved through a wearable device that is attached to the footwear of the user. Considering the various design options for a foot-operated input device, we wanted a device that is easy to operate and should not strain the user with prolonged usage. The system presented in [158] used physical pedals (USB connected) which require tilting and lifting movements of the feet to generate user inputs. Prolonged use of these pedals could lead to fatigue. Hence, we aimed that our device should have a small form factor (wearable), and should communicate with the main system wirelessly (over Bluetooth). Considering these requirements, we improved on the design of a foot-operated device presented by Rajanna et al. [141]. We created a small 3D printed container to package the circuitry, and this foot-operated device is attached to the shoe of the user.

For achieving foot gesture-based interactions, the foot-operated device contains gyroscope that recognizes the foot gestures. In addition, the interaction system also contains a bluetooth receiver that connects to the computer over an USB port. For achieving foot press-based interactions, the foot-operated device contains a voltage differentiate circuit and a pressure sensor extending out from the device. The pressure sensor is placed inside the footwear such that it can be operated (activated) by applying pressure with the foot, specifically the toe. Since the device senses the pressure applied by the user, the system does not require physical movements of the foot, but a gentle press is enough, making it convenient to use. In addition, the wearable device approach allows for customizations, since it can be 3D printed to support various interaction paradigms (foot, hand, etc.).

4.5 System Design and Implementation

Based on our design decisions, we created a gaze and foot-based typing system that comprises of three primary modules: 1) Gaze Interaction Server, 2) Virtual Keyboard (VKB), and 3) Foot-Operated Wearable Device. A pictorial depiction of the system is shown in Figure 4.1.

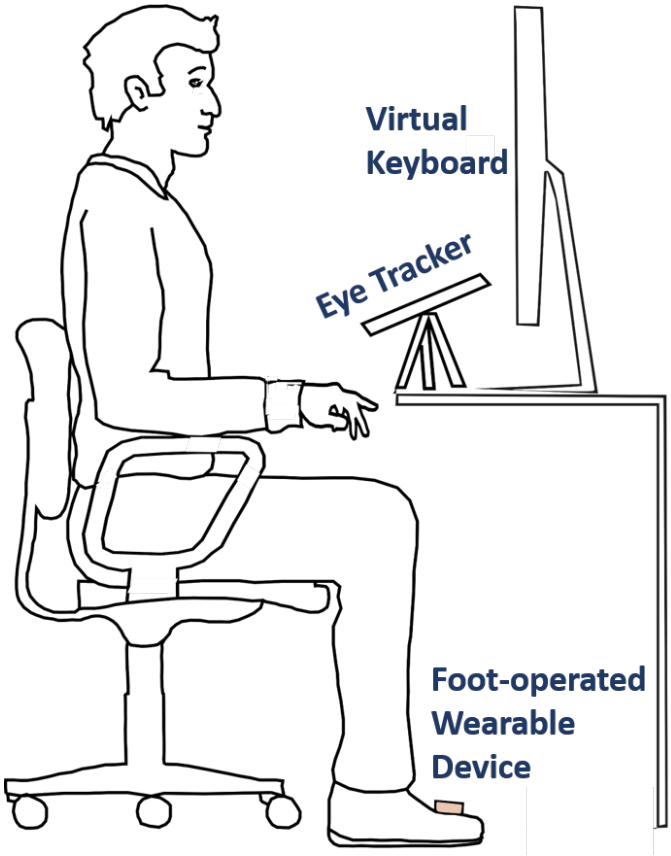

Figure 4.1: Gaze and foot-based typing system: an eye tracker is placed in front of a monitor displaying the virtual keyboard, and the user is wearing a footwear augmented with the wearable device.

4.5.1 Gaze Interaction Server:

The gaze interaction server is the central module that coordinates between the VKB and foot-operated input device to achieve gaze typing. It runs on the computer and receives input from the foot-operated device. The foot-operated device connects to the central module on the computer over a Bluetooth connection. The gaze interaction server converts the foot input received as a single byte characters into the key selection commands that are understood by the virtual keyboard.

4.5.2 Virtual Keyboard

In our experiment we used a QWERTY virtual keyboard developed from the open-source VKB "OptiKey[2]." We enhanced the standard QWERTY VKB layout to be suitable for gaze and foot-based typing, and to improve the typing efficiency. The keyboard layout was customized over multiple design iterations. These customizations can be categorized as a) regrouping, b) re-sizing, and c) redundancy. The enhanced keyboard layout is shown in Figure 4.2. As part of layout regrouping, we moved the infrequently used symbolic keys to the secondary screen which can be activated through a menu key, and also moved the numeric keys to the primary screen. As part of layout re-sizing, we emphasized the frequently used keys with larger dimensions, for example, we made some functional keys (space, enter, backspace) prominent (Figure 4.2). Specifically, the space bar was made significantly larger. Lastly, as part of introducing redundancy, we added two instances of backspace keys, one at the top row and the other at the bottom row, so that the user can correct errors quickly. The virtual keyboard constantly receives the user's gaze points on the screen as a pair of (x,y) co-ordinates from the eye tracker (Tobii EyeX[3]). As the user's gaze scans the keys on the keyboard, each key looked at by the user is highlighted along the border of the key with the red color. Once the key is selected either with a dwell time or input from the foot, the background of the key is highlighted in blue, and the character is printed in the writing space.

4.5.3 Foot Gesture Recognition Device

The foot gesture recognition device consists of two units, a master (sender) and a receiver as shown in Figure 4.3. We aimed at creating a small form factor foot-operated input device that can be attached to the user's footwear. Hence, the entire circuitry of the master unit is housed inside a 3D-printed container that is attached to the user's footwear as shown in Figure 4.4. The receiver is an USB enabled unit that is connected directly to the computer. The master unit is responsible for recognizing the foot gesture, and sending the appropriate command (e.g., click) to the receiver. The receiver is responsible for the executing the command on the computer.

[2] github.com/OptiKey
[3] tobiigaming.com/product/tobii-eyex/

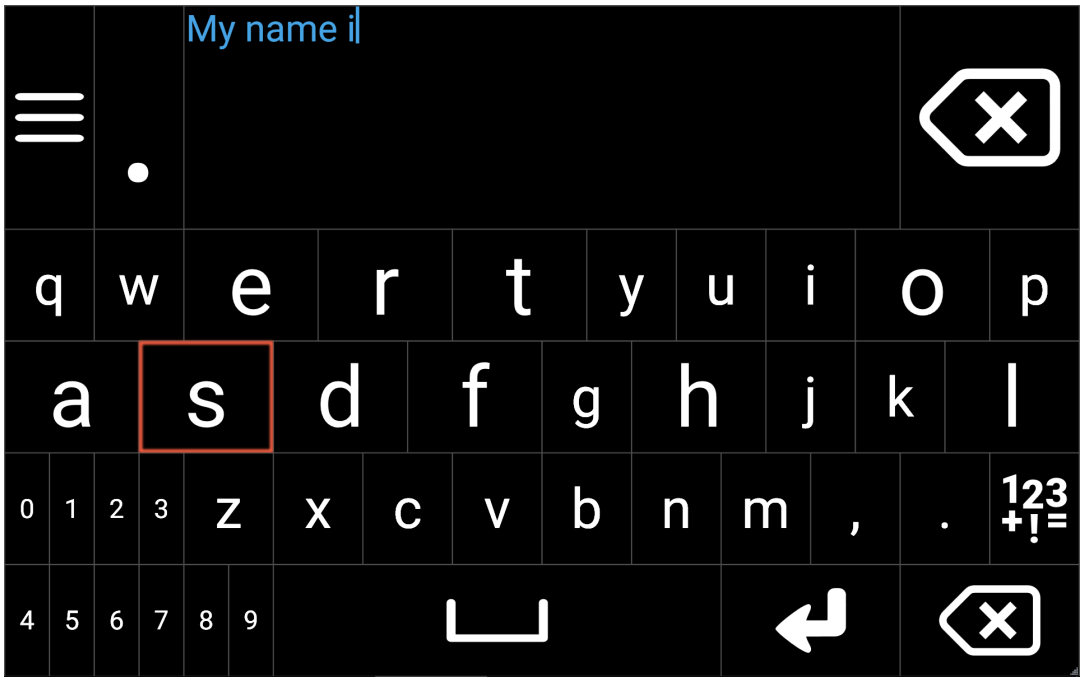

Figure 4.2: An enhanced QWERTY keyboard: the keyboard layout is customized such that the frequently used keys have larger dimensions, infrequently used symbolic keys are moved to a secondary screen, and the backspace key is made redundant to help correct errors quickly.

4.5.3.1 Master Unit

The circuit diagram of the master unit is shown in Figure 4.5. The unit is built using four main modules: 1) a motion processing unit (MPU-6050) with gyroscope and accelerometer, 2) an Arduino Pro Mini Microcontroller, 3) a Bluetooth Module (HC-05), and 4) a Battery Recharging Unit (Adafruit Powerboost 1000C). The device is powered by a rechargeable battery and can be turned on and off with a switch. The gyroscope provides foot orientation data, and the microcontroller constantly reads the changes in foot orientation and recognizes various foot gestures. Once a foot gesture is identified, this information is sent to the receiver unit connected to the computer. Figure 4.6 shows an outline of how the master unit is attached to the user's footwear, and Figure 4.7 shows the list of foot gestures that are recognized by the master unit.

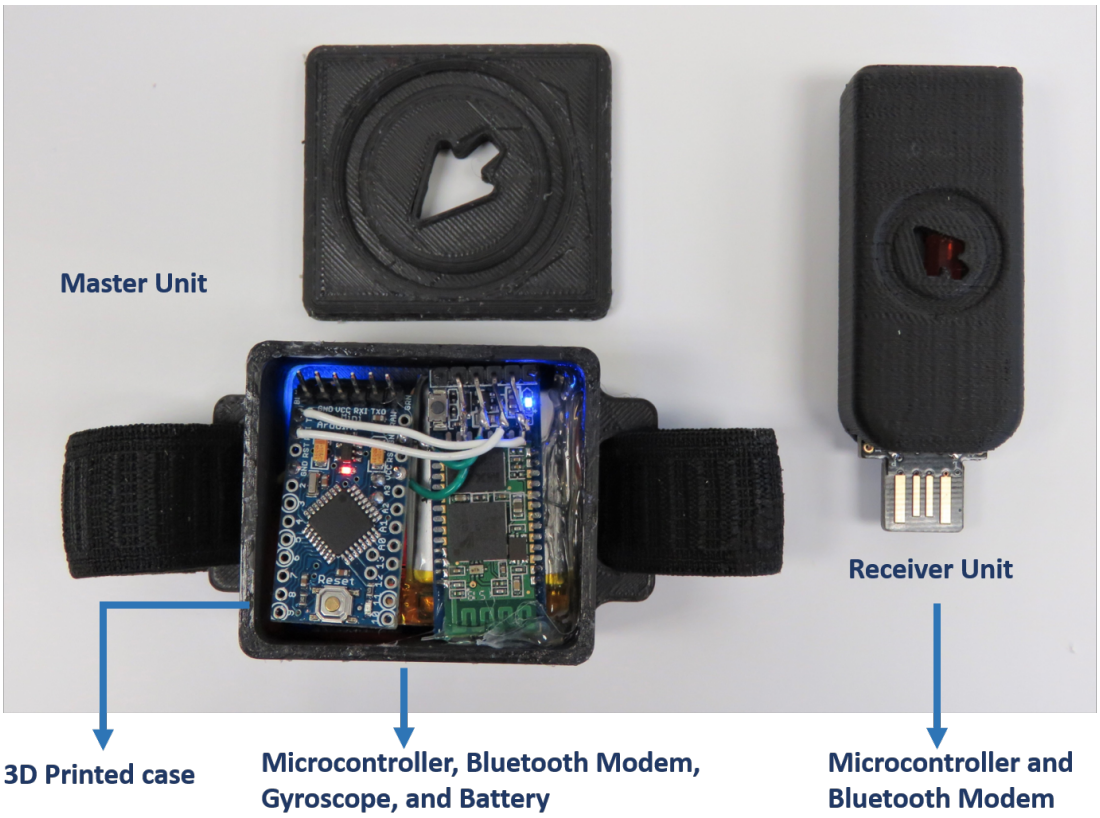

Figure 4.3: Foot Gesture Recognition Device: the master and receiver units. The master unit is attached to user's footwear, and the receiver unit is connected to the computer through USB port.

4.5.3.2 Receiver Unit

The receiver is a USB enabled, plug-and-play unit (Figure 4.3), and it consists of two modules: 1) Arduino Leonardo USB Microcontroller, and 2) a Bluetooth Module (HC-05). The circuit diagram of the receiver unit is shown in Figure 4.8. While the receiver unit can execute commands like single click, double click, right click, etc., in our system, irrespective of the gesture identified, a click action is performed at the cursor position.

4.5.4 Foot Press Sensing Device

Similar to the foot gesture recognition device, the foot press sensing device has a small form factor and is attached to the user's footwear as shown in Figure 4.9. The entire circuitry of the device is housed in a 3D printed container, while just exposing the pressure sensor. The circuitry

Figure 4.4: Foot Gesture Recognition Device - Master unit: the entire circuitry of the master unit is housed inside a3D-printed container that is attached to the user's footwear. The user is executing a toe tap gesture.

consists of three modules: 1) Teensy 2.0 Microcontroller[4], 2) Bluetooth Modem (BlueSMiRF)[5], and 3) Force Sensitive Resistor[6]. An outline how the device is attached to the user's footwear and the placement of the pressure sensor are shown in Figure 4.10.

The pressure sensor extends from the main circuit as shown in Figure 4.11 and the circuit diagram is shown in Figure 4.12. The user input from pressing the pressure sensor (Force Sensitive Resister), is sensed by measuring the output voltage of a voltage divider circuit. The minimum amount of pressure to be applied, to be registered as an input action, can be adjusted by setting the output voltage threshold (V_{out}) in equation 4.1. In equation 4.1, $R1$ and $R2$ are the resistance values, and V_{in} is the input voltage. The user input, based on the pressure thresholds, is encoded as a single byte characters and transmitted to the Gaze Interaction Server via the Bluetooth Modem.

[4]www.pjrc.com
[5]www.sparkfun.com/products/12577
[6]www.sparkfun.com/products/9376

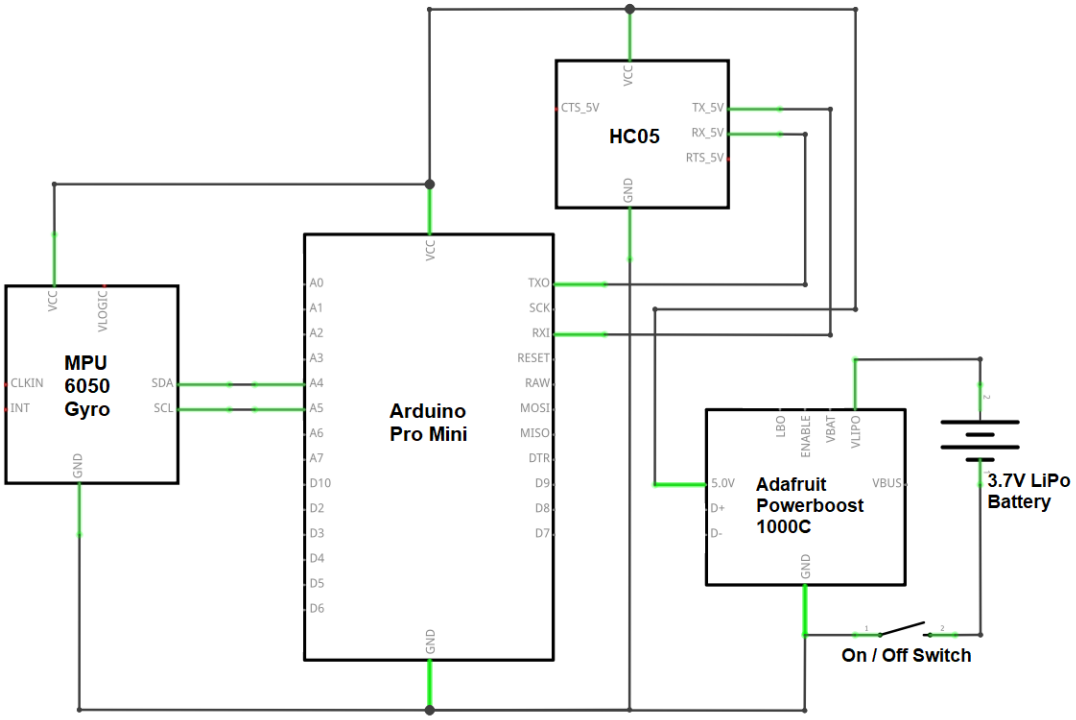

Figure 4.5: Foot Gesture Recognition Device: circuit diagram of the master unit showing the four main modules 1) a motion processing unit (MPU-6050) with gyroscope and accelerometer, 2) an Arduino Pro Mini Microcontroller, 3) a Bluetooth Module (HC-05), and 4) a Battery Recharging Unit (Adafruit Powerboost 1000C).

The Gazer Interaction Server then decodes the message received into key selection commands.

$$V_{out} = V_{in} \cdot \frac{R1}{R1 + R2} \tag{4.1}$$

4.6 Experiment Design

Through the system evaluation, we wanted to answer the questions discussed in the *Research Questions* section. At a broader level, we wanted to evaluate if a gaze and foot-based, multimodal, dwell-free typing system is feasible? And how does such a system compare against the gaze and dwell-based typing system? Also, from the usability perspective, we wanted to understand the advantages of using the supplemental foot input as a selection method in gaze typing. To specifically explore these questions, we conducted three experiments by involving a total of 51 participants.

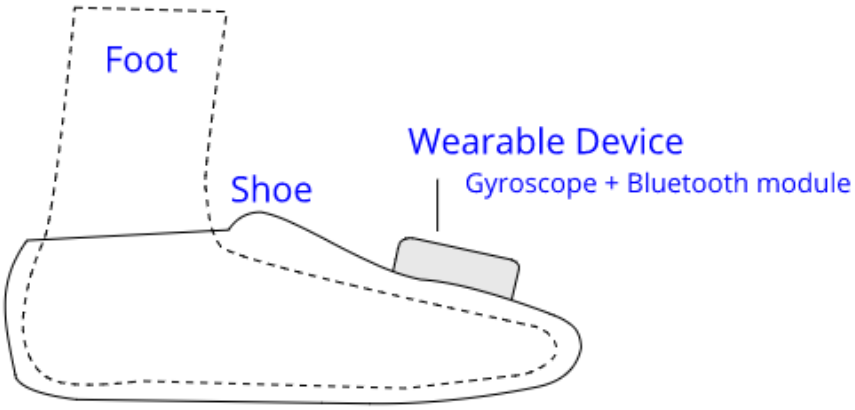

Figure 4.6: Foot Gesture Recognition Device: an outline of how the master unit is attached to the user's footwear.

In each experiment we used a unique key selection method, out of the three selection methods we developed: 1) dwell-based selection, 2) foot gesture-based selection, and 3) foot press-based selection. We also made sure that each subject participated in only one of the three experiments, which means we had a different set of users participating in each experiment. We followed this model to avoid any familiarity developed with the gaze typing system by participating in one experiment influencing the user's performance in subsequent experiments. The enhanced and standard keyboards used in our study did not have word or character suggestion features, and the participants were asked to correct all the errors in the entered text. The details of each phase, specifically the task performed, and the results are discussed in the results and discussion Section 4.7.

4.7 Results

The efficiency of our gaze typing system was evaluated based on a text-focused and key-selection-focused metrics. The two gaze typing metrics we considered were Words Per Minute (WPM), and Rate of Backspace Activation (RBA), shown in equation 2 and 3 respectively [24]. In our experiments, the participants entered phrases from Mackenzie et al. [170], a collection 500 phrases for evaluation of text entry techniques.

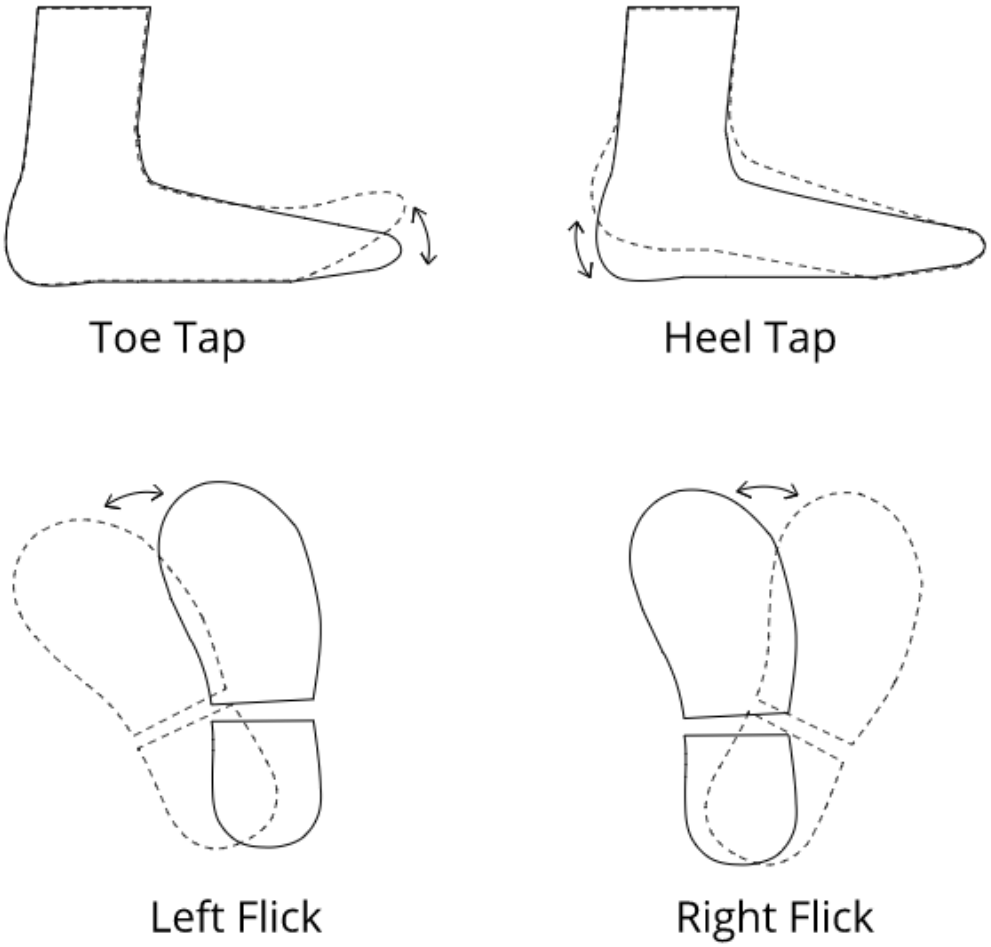

Figure 4.7: Foot Gesture Recognition Device: the list of foot gestures that are recognized by the device.

$$Words\,Per\,Minute\,(WPM) = \frac{Number\,of\,Characters}{Time\,Spent\,for\,Typing\,(min) \times 5} \tag{4.2}$$

$$Rate\,of\,Backspace\,Activation\,(RBA) = \frac{Number\,of\,Keystrokes\,for\,Backspace\,or\,Delete}{Number\,of\,Characters\,Typed} \tag{4.3}$$

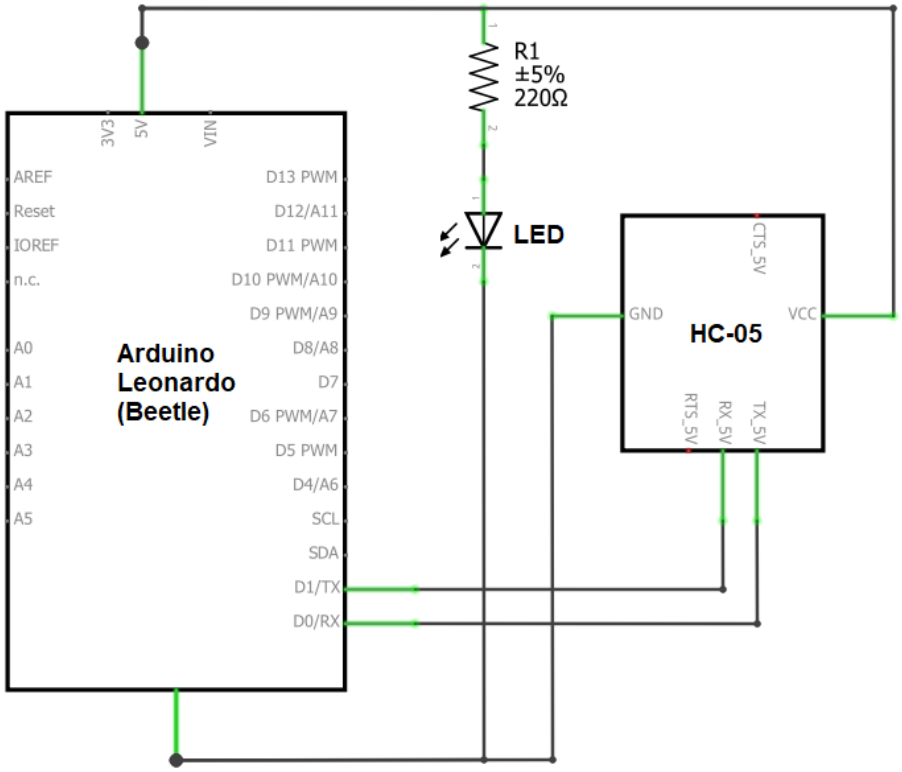

Figure 4.8: Foot Gesture Recognition Device: circuit diagram of the receiver unit showing the two primary modules 1) Arduino Leonardo USB Microcontroller, and 2) a Bluetooth Module (HC-05).

4.7.1 Experiment 1: Gaze and Dwell-based Typing

In this phase, the participants gaze typed using dwell-based selection. 17 participants (9 male, 8 female) with their ages ranging from 21 to 32 ($\mu_{age} = 23.5$) participated in this phase. We choose three different dwell times: 1000 ms, 700 ms, and 400 ms. The three different dwell times were chosen based on the most common least, average, and maximum dwell times used in the prior studies [171, 151, 76, 74, 157, 112]. Each participant typed 10 phrases with each of the three dwell times, first starting with 1000 ms, next 700 ms, and lastly with a dwell time of 400 ms. Hence, each participant typed a total of 30 phrases, and overall 510 phrases were entered by the 17 participants

Figure 4.9: Foot Press Sensing device attached to the user's footwear. The pressure sensor is placed inside the footwear.

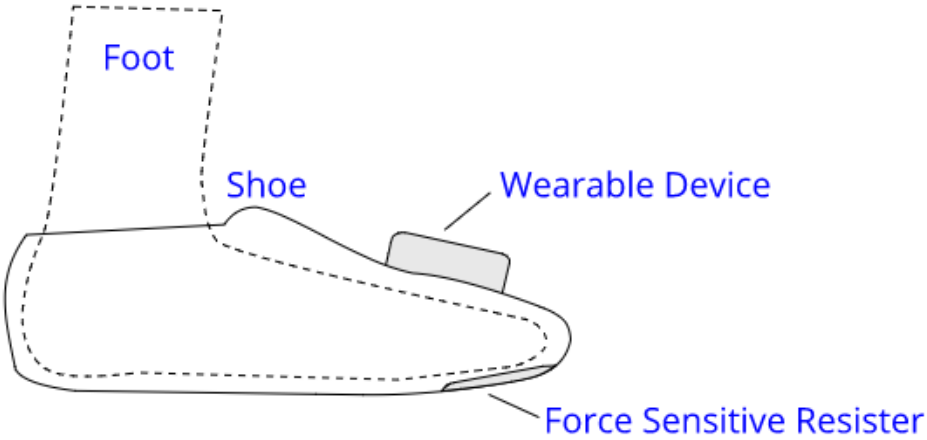

Figure 4.10: An outline of how the foot press sensing device is attached to the user's footwear, and the placement of the pressure sensor inside the footwear.

(17×30) across the three dwell times. Table 4.1 lists the mean and standard deviation of gaze typing metrics like WPM and RBA across the three dwell times.

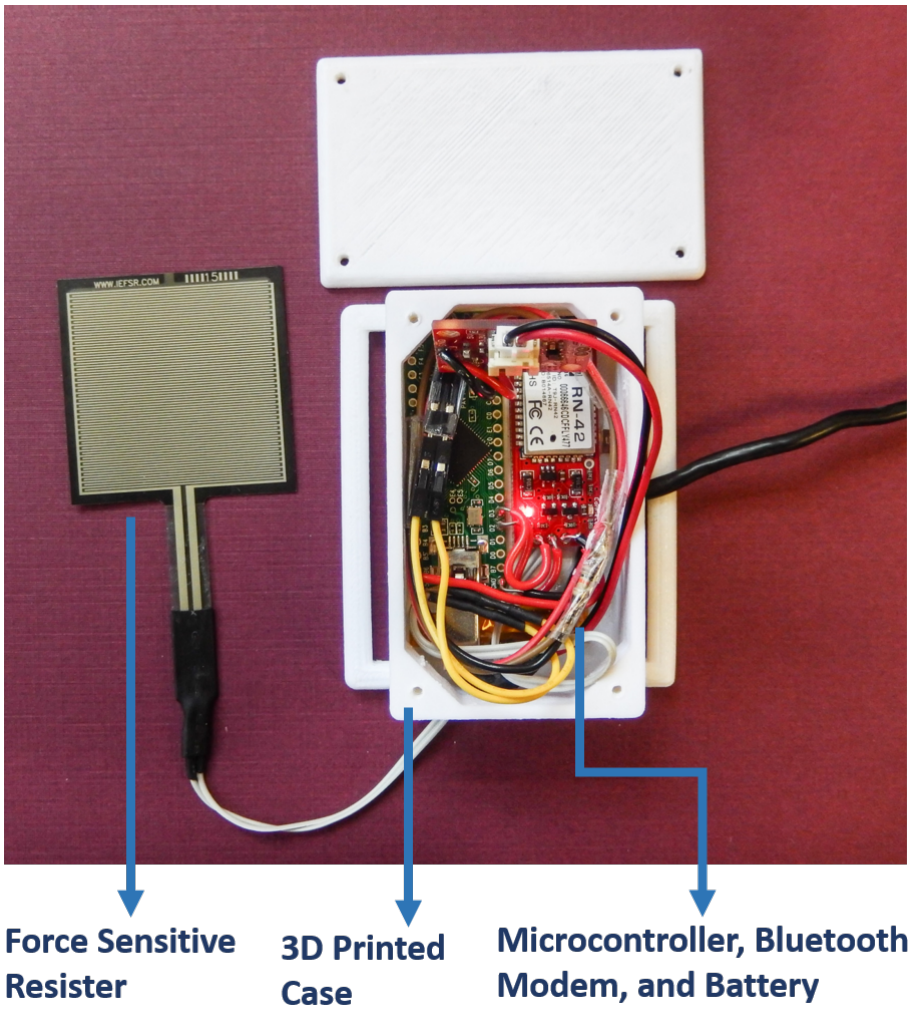

Figure 4.11: Foot Press Sensing Device: the entire circuitry is housed inside a 3D-printed container, and the force sensitive resister that senses foot press actions extends from the main circuit and is placed inside the footwear.

The highest mean typing speed of 11.65 Words Per Minute (WPM) was achieved with a dwell time of 400 ms, and the corresponding mean Rate of Backspace Activation (RBA) was 7% (highest error). The least mean error rate, i.e., a 1% RBA was achieved with a dwell time of 1000 ms, the corresponding mean WPM was 6.19 (lowest typing speed). Figure 4.13 and 4.14 show WPM and RBA respectively across the three different dwell times.

From Figure 4.13, we observe that the gaze typing speed increases with decreasing dwell time. While participants achieved the highest typing speed at 400 ms, during the post study interviews

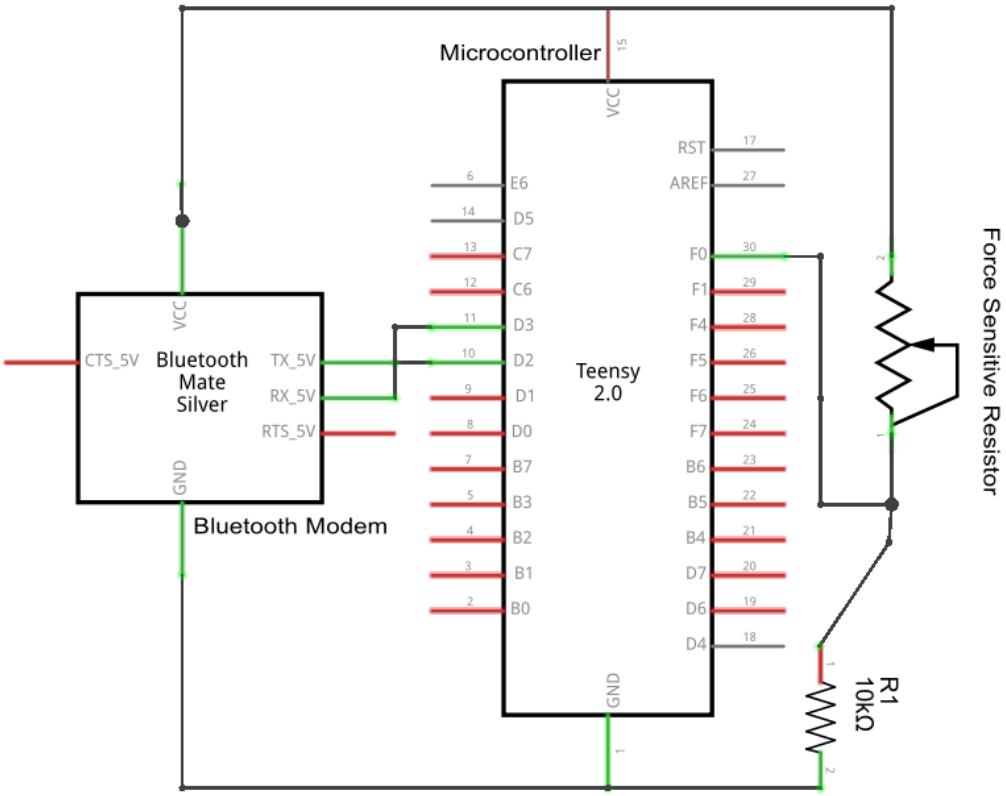

Figure 4.12: Foot Press Sensing Device: the three primary modules 1) Teensy 2.0 Microcontroller, 2) Bluetooth Modem (BlueSMiRF), and 3) Force Sensitive Resistor.

Table 4.1: Dwell-based Selection: typing speed (WPM) and error rate (RBA) across different dwell times

	WPM		**Error**	
Dwell Time	Mean	Std. Dev	Mean	Std. Dev
1000 ms	6.19	1.24	0.01	0.02
700 ms	8.71	1.38	0.01	0.01
400 ms	11.65	1.8	0.07	0.06

they commented that the shorter dwell time of 400 ms demanded extensive attention as there was very little, or no time for error recovery.

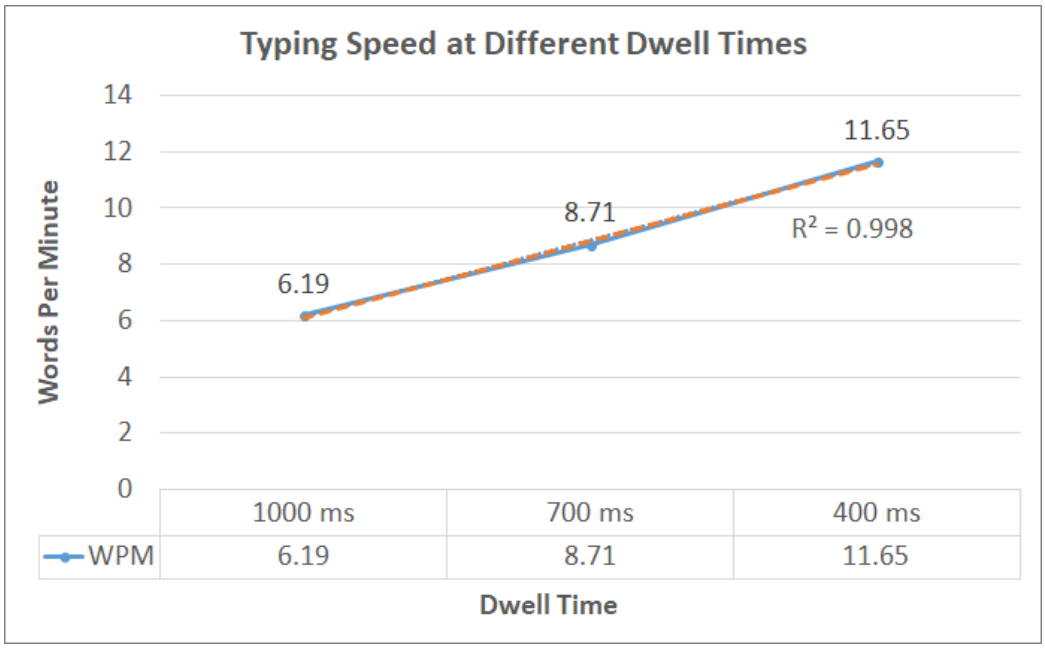

Figure 4.13: Dwell-based Selection: typing speed expressed in terms of Words Per Minute (WPM) across different dwell times. We observe that the typing speed increases with the decreasing dwell time, and the regression line has an R^2 value of 0.99.

From Figure 4.14, we observe that the error rate increases with the decreasing dwell time. The reason for this observation is two fold: first, with a higher dwell time such as 1000 ms, the user gets enough time to search for the target key, and also quickly recover from inadvertent selections by looking away from the character before the dwell time threshold elapses. This helps in achieving a significantly lower error rate with higher dwell time. Secondly, with a shorter dwell time like 400 ms, the user is expected to shift their gaze quickly between desired characters, without inadvertent selections during visual search. Hence, with decreasing dwell time the error rate increases.

4.7.2 Experiment 2: Gaze and Foot Gesture-based Typing

In this experiment, the participants gaze typed using foot gesture-based selection method. 17 participants (9 male, 8 female) with their ages ranging from 20 to 27 ($\mu_{age} = 22.29$) participated in this experiment. The experiment was further divided into four sessions, and in each session a participant typed 10 phrases. Hence, each participant typed a total of 40 phrases, and overall 680 phrases were entered by the 17 participants (17×40) across the four sessions. As discussed in

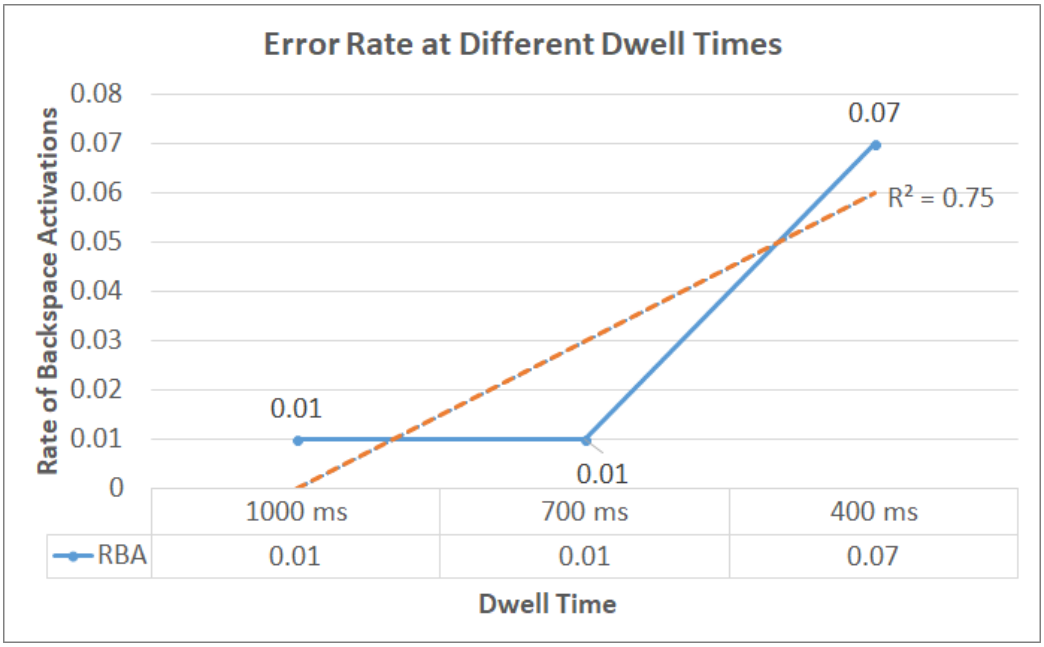

Figure 4.14: Dwell-based Selection: error rate expressed in terms of Rate of Backspace Activation (RBA) across different dwell times. We observe that the error rate increases with the decreasing dwell time, and the regression line has an R^2 value of 0.75

the research questions Section 4.3, the main goals of our study was to explore the usability and performance of foot gestures in gaze typing as this knowledge is unavailable. Hence, based on the prior studies, where foot input was used for interacting with computers, we considered four gestures: 1) toe tap, 2) heel tap, 3) right flick, and 4) left flick for key selection. Irrespective of the gesture performed, the key focused on by the user gets selected following the completion of the gesture.

At the beginning of the study, the user was asked to type a few practice phrases using different gestures to develop familiarization with the four gestures. During the study, the user had the freedom of using any of the four gesture or a combination of gestures for target selection. Table 4.2 lists the mean and standard deviation of gaze typing metrics like WPM and RBA across four sessions.

A highest mean typing speed of 13.82 WPM, with a lowest error rate of 7% RBA were achieved at the end of the fourth session. A lowest mean typing speed of 10.3 WPM, with a highest error

Table 4.2: Foot gesture-based Selection: typing speed (WPM) and error rate (RBA) across different sessions.

	WPM		Error	
Dwell Time	Mean	Std. Dev	Mean	Std. Dev
S1	10.3	2.48	0.1	0.04
S2	12.1	2.29	0.08	0.04
S3	13.49	2.65	0.07	0.03
S4	13.82	1.94	0.07	0.03

rate of 10% RBA were observed at the end of the first session. Figure 4.15 and 4.16 show WPM and RBA respectively across the four typing sessions.

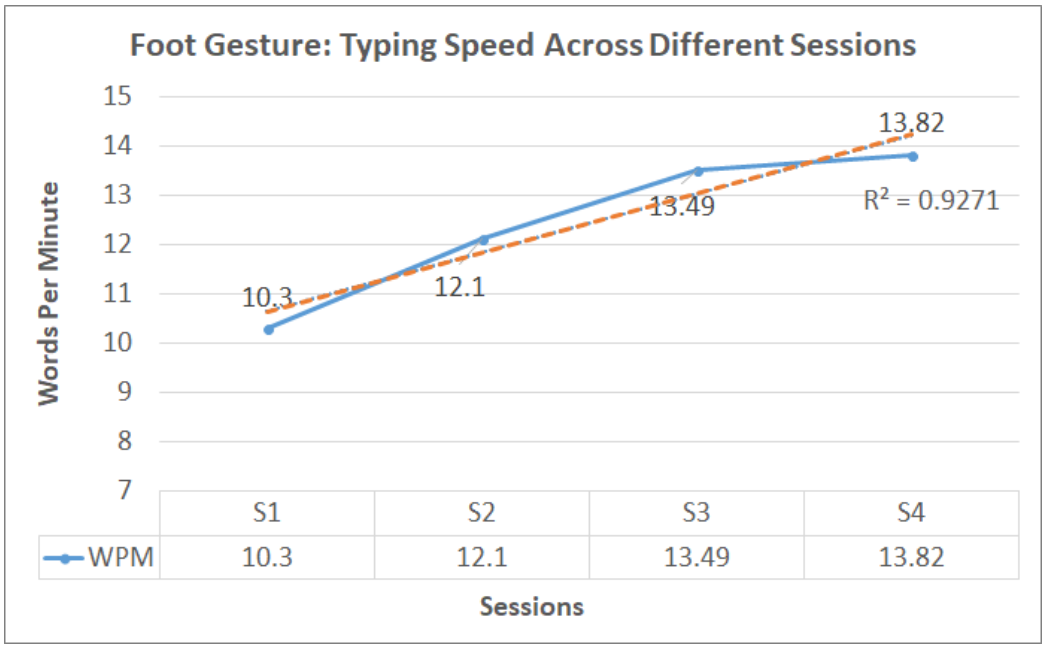

Figure 4.15: Foot gesture-based Selection: typing speed expressed in terms of Words Per Minute (WPM) across different sessions. We observe that the typing speed increases with subsequent sessions, and the regression line has an R^2 value of 0.92.

From Figure 4.15, we observe that typing speed increases with subsequent sessions with the

participants reaching the highest typing speed at the end of the fourth session. As learned from the post study interviews there are three reasons for this observation: 1) participants get familiar with the foot gestures, i.e., the learn how high the toe or heel need to be raised, or right and left flicks to be performed to achieve key selection, 2) generally, the participants choose a convenient gesture and use it throughout the study. This behavior contradicts our hypothesis that the participants switch to different gestures to reduce the strain on a single part of the feet, and 3) the participants achieve better synchronization of focusing their gaze on the target key and selecting it with a foot gesture.

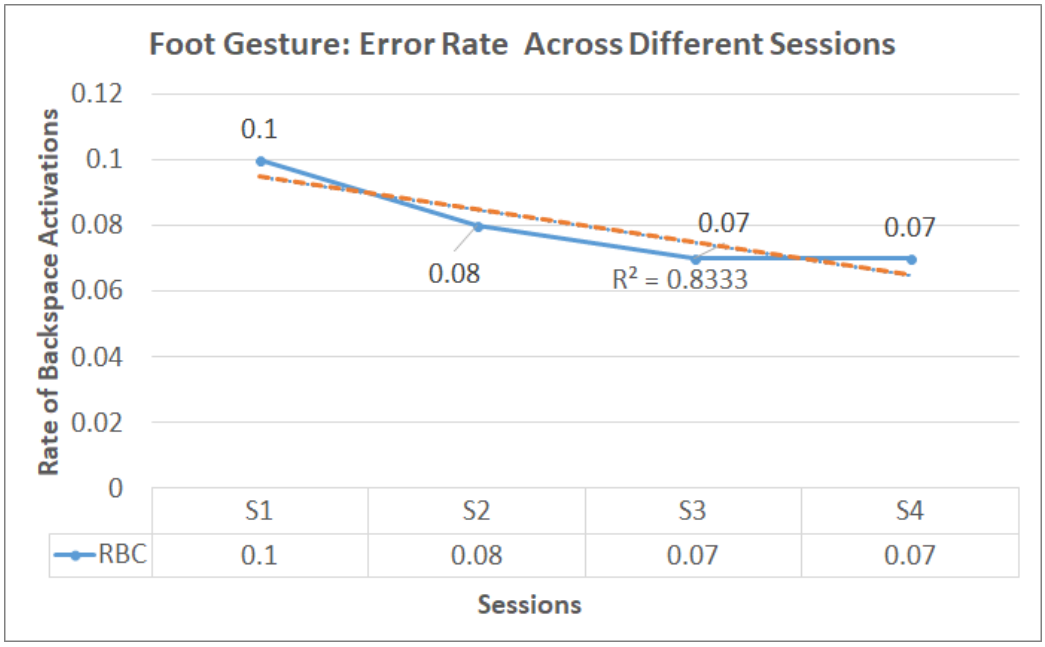

Figure 4.16: Foot gesture-based Selection: error rate expressed in terms of Rate of Backspace Activation (RBA) across different sessions. We observe that the error rate decreases with subsequent sessions, and the regression line has an R^2 value of 0.83

From Figure 4.16, we observe that error rate decreases with subsequent sessions with the participants reaching the lowest error rate at the end of fourth session. Similar to the increasing typing speed, the familiarity with the foot gestures results in reduced errors. Specifically, the participants learn to better synchronize gaze pointing with the gesture execution. During the initial sessions,

generally a user switches their gaze to the next character in the word before selecting the current word, i.e., the eyes moves faster than the foot gesturing. However, as shared in the post study interviews, the participants learn a repeated pattern of gaze pointing and foot gesturing so that these two actions are always in synchronization. This leads to reduced error rate with subsequent sessions.

Furthermore, we were interested in learning which of the four gestures is used the most and the variation in the use of different kinds of gestures across the sessions. Figure 4.17 shows the percentage of overall usage of each gesture throughout the study. Figure 4.18 (and Table 4.3) shows the percentage of usage of each gesture across the four sessions as the percentage of total gestures performed in that session, and it appears that toe tap is the most used gesture. We performed a one-factor ANOVA with replication. The independent factor was “Gestures" which had four levels: toe tap, heel tap, right flick, and left flick. The dependent variable was “Gesture Usage" which was the percentage of each gesture used in each session by each participant. The results show that there is a significant difference in the usage of different types of gestures with an $F(3,201) = 92.955, p < 0.001$. Also, post-hoc analysis with Bonferroni correction indicated that the gesture usage between any pair of gestures is significant ($p < 0.05$), except for left and right flicks ($p > 0.05$).

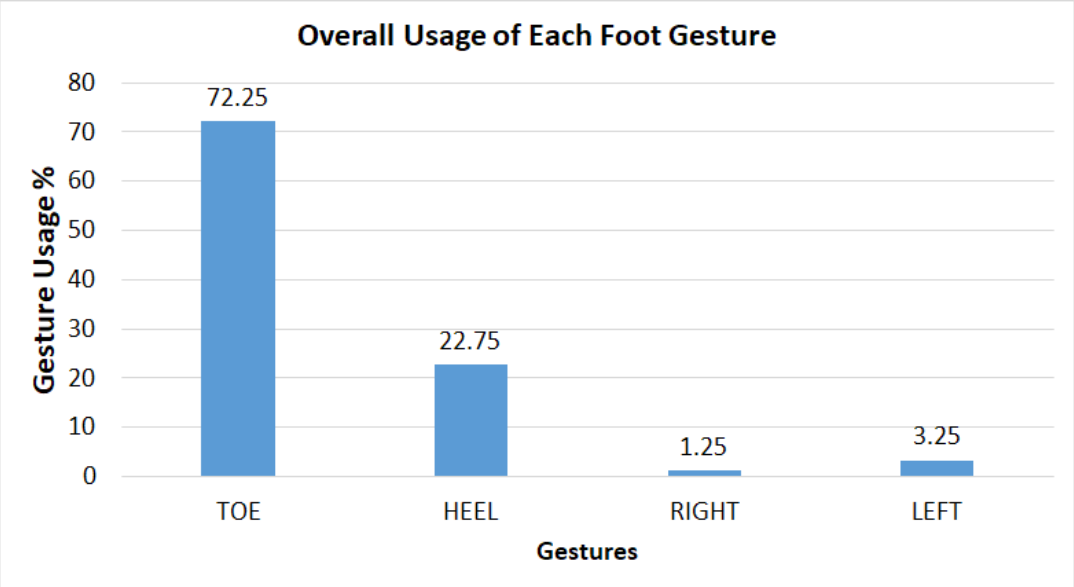

Figure 4.17: Overall usage of each gesture throughout the study

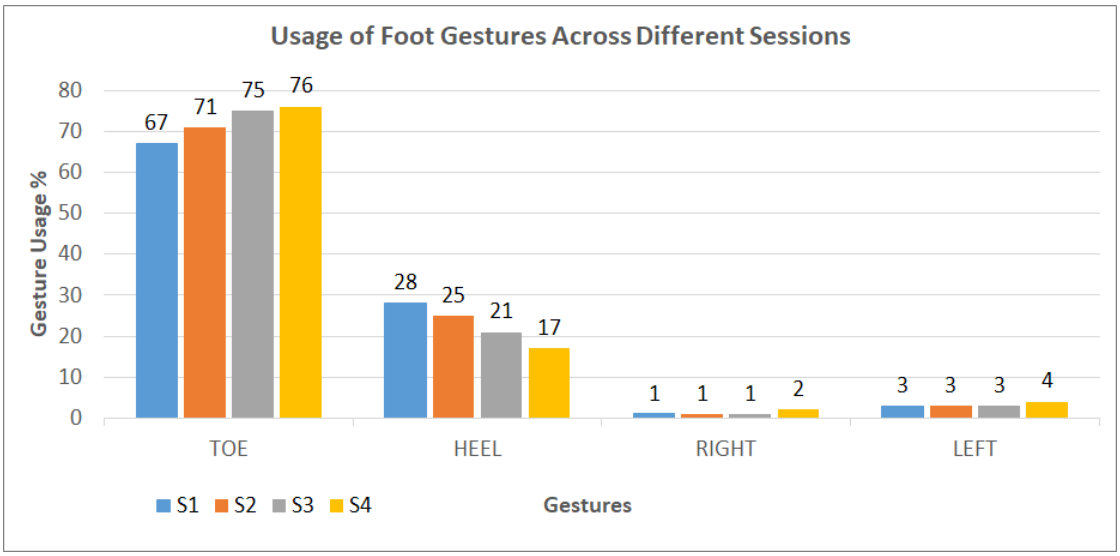

Figure 4.18: Usage of each gesture across the four sessions

Table 4.3: Gesture Usage Across Sessions

Sessions	Toe %	Heel %	Right %	Left %
S1	67	28	1	3
S2	71	25	1	3
S3	75	21	1	3
S4	76	17	2	4

Next, to answer the question that do users choose a single gesture initially and use the same gesture throughout the study, or do they change gestures as the study progresses, we conducted one-factor ANOVA with replication. The independent factor was “sessions" which had four levels S1, S2, S3, S4. The four dependent variables were toe tap, heel tap, right flick, and left flick. A total of four ANOVA tests were performed, and for each ANOVA test we considered one dependent variable of the four dependent variables. The results of the four ANOVA tests are presented in Table 4.4, and we observe that “sessions" is not a significant factor for dependent variables: toe

tap, heel tap, right flick, and left flick. This indicates that a gesture, e.g., toe tap, is used equally across four sessions. The same holds true for other gestures like heel tap, right flick, and left flick. In addition, post-hoc analysis with Bonferroni correction indicates that for a given gesture, its amount of usage does not differ across any two sessions. All these observations strongly suggest that though system supports multiple gestures, each participant selects a single, convenient gesture, and uses it throughout the study. Though we expected that the user may switch to different gestures to avoid stress involved in performing the same gesture for a long time, the users' behavior was opposite to our expectation. Just the availability of multiple gestures do not influence the user to switch to using different gestures, and this could be because a user does not want to lose the familiarity developed in using a gesture. Also, it takes some time and effort by the user in getting familiarized with a new gesture.

Lastly, we discuss the learning effects of gaze typing when using foot gesture-based selection. We performed a one-factor ANOVA with replication. The independent factor was 'sessions' which had four levels: S1, S2, S3, and S4. The dependent variables were typing speed (WPM) and error rate (RBA). From Table 4.5 we observe that 'sessions' is a significant factor for typing speed (WPM). The differences in typing speed across sessions is significant ($p < 0.05$), and the speed increases with subsequent sessions. Also, post-hoc analysis with Bonferroni correction indicated that the typing speed differs between any pair of sessions ($p < 0.05$), except for sessions 3 and 4. This indicates that the typing speed reaches a plateau at nearly 13.8 WPM. Similarly, the difference in error rate across sessions is significant ($p < 0.05$), and the error rate decreases with subsequent sessions. Post-hoc analysis with Bonferroni correction indicates that error rates between sessions 1 and 2, 1 and 3, and 1 and 4 are significant ($p < 0.05$). Since the difference in error between sessions 2 and 3, and 3 and 4 is not significant, these results suggest that the participants quickly learn (by session 1) to use the foot gesture-based selection and start making fewer errors, and this behavior continues throughout the study.

Table 4.4: Foot gesture-based Selection: ANOVA tests to understand if a gesture is used equally across the sessions (p values highlighted in gray indicate significance at $\alpha = 0.05$)

	Sessions [S1, S2, S3, S4]	**Std. Error**	**Post hoc Analysis**
Toe	F(3,48) = 1.376 *p = 0.261*	S1 = 8.450 S2 = 8.691 S3 = 8.828 S4 = 8.424	(S1, S2) *p = 1.000* (S1, S3) *p = 0.907* (S1, S4) *p = 1.000* (S2, S3) *p = 1.000* (S2, S4) *p = 1.000* (S3, S4) *p = 1.000*
Heel	F(3,48) = 2.180 *p = 0.103*	S1 = 8.276 S2 = 8.007 S3 = 7.648 S4 = 7.477	(S1, S2) *p = 1.000* (S1, S3) *p = 0.704* (S1, S4) *p = 0.593* (S2, S3) *p = 1.000* (S2, S4) *p = 0.773* (S3, S4) *p = 1.000*
Right	F(3,48) = 0.765 *p = 0.519*	S1 = 1.023 S2 = 1.374 S3 = 1.455 S4 = 1.417	(S1, S2) *p = 1.000* *(S1, S3) p = 1.000* *(S1, S4) p = 1.000* *(S2, S3) p = 1.000* (S2, S4) *p = 1.000* (S3, S4) *p = 1.000*
Left	F(3,48) = 0.524 *p = 0.668*	S1 = 2.240 S2 = 2.692 S3 = 2.744 S4 = 3.118	(S1, S2) *p = 1.000* (S1, S3) *p = 1.000* (S1, S4) *p = 1.000* (S2, S3) *p = 1.000* (S2, S4) *p = 1.000* (S3, S4) *p = 1.000*

Table 4.5: Foot gesture-based Selection: ANOVA tests to understand the learning effects across the sessions (p values highlighted in gray indicate significance at $\alpha = 0.05$).

	Sessions [S1, S2, S3, S4]	**Std. Error**	**Post hoc Analysis**
WPM	F(3,48) = 47.372 *p = 0.000*	S1 = 0.601 S2 = 0.556 S3 = 0.642 S4 = 0.471	(S1, S2) *p = 0.000* (S1, S3) *p = 0.000* (S1, S4) *p = 0.000* (S2, S3) *p = 0.001* (S2, S4) *p = 0.000* (S3, S4) p = 1.000
Error	F(3,48) = 9.793 *p = 0.000*	S1 = 0.010 S2 = 0.010 S3 = 0.008 S4 = 0.008	(S1, S2) *p = 0.005* (S1, S3) *p = 0.002* (S1, S4) *p = 0.004* (S2, S3) *p = 0.528* (S2, S4) *p = 0.960* (S3, S4) *p = 1.000*

4.7.3 Experiment 3: Gaze and Foot Press-based Typing

As experiment 2 demonstrated that the majority of participants find toe tapping more convenient than other gestures. Also, we observed that the participants generally do not switch to different gestures, however, they pick a gesture initially and use the same gesture throughout the study. These observations motivated us to develop a third key selection method: foot press-based selection. The reason for considering foot press-based selection was to achieve higher typing speed than gaze and foot gesture-based selection. In this method a Force Sensitive Resistor (FSR) is placed under the toe area of the foot, and for key selection, the user has to perform a subtle press action on the sensor. Unlike the foot gestures, the foot press-based selection does not require any movement of the foot, but subtle foot presses will achieve key selection. Hence, we hypothesized that the ease and convenience of the foot press-based selection would result in higher typing speed

than the foot gesture-based selection.

To test our hypothesis, the participants gaze typed using foot press-based selection method. 17 participants (15 male, 2 female) with their ages ranging from 21 to 26 ($\mu_{age} = 22.29$) participated in this experiment. Similar to experiment 2, each participant completed four typing sessions, and 10 phrases were typed in each session. Therefore, a total of 680 phrases were entered by the 17 participants (17×40) with each participant typing 40 phrases from four sessions. Table 4.6 lists the mean and standard deviation of gaze typing metrics like WPM and RBA across the three dwell times.

Table 4.6: Foot Press-based selection: typing speed (WPM) and error rate (RBA) across different sessions.

	WPM		**Error**	
Dwell Time	Mean	Std. Dev	Mean	Std. Dev
S1	11.41	1.95	0.09	0.04
S2	13.04	1.6	0.07	0.03
S3	13.93	1.54	0.07	0.04
S4	14.98	1.68	0.06	0.03

The highest mean typing speed of 14.98 WPM, the lowest error rate of 6% RBA were achieved at the end of the fourth session. The lowest mean typing speed of 11.41 WPM, the highest error rate of 9% RBA were observed at the end of the first session. Figure 4.19 and 4.20 show WPM and RBA respectively across the four typing sessions.

As observed in experiment 2, Figure 4.19 shows that the typing speed increases with successive sessions, and Figure 4.20 shows that the error rate decreases with subsequent sessions. This increased typing speed and decreased error rate is attributed to the increased familiarity with the foot press action, and the ability to synchronize gaze pointing and selection with the foot press. We learned from the post-study interviews that few participants positioned their toe within shoe such

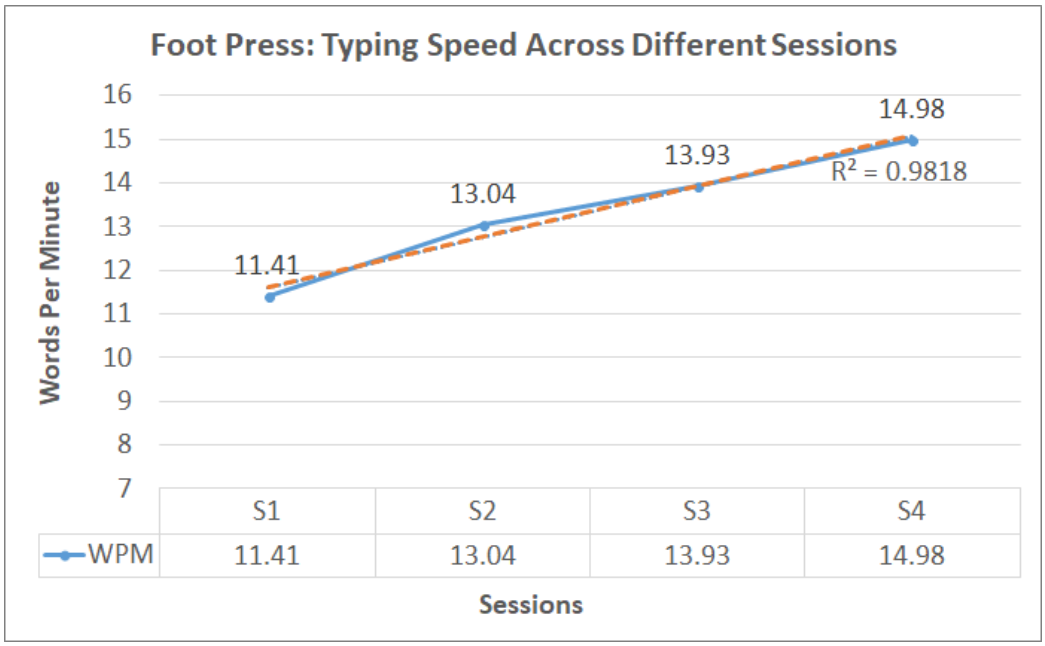

Figure 4.19: Foot press-based Selection: typing speed expressed in terms of Words Per Minute (WPM) across different sessions. We observe that the typing speed increases with subsequent sessions, and the regression line has an R^2 value of 0.98.

that they used only the big toe for pressing the pressure sensor. However, others used their entire toe to press the pressure sensor.

To understand the learning effects of foot press-based gaze typing, we conducted one-factor ANOVA with replication. Similar to experiment 2, "sessions" was an independent factor with four levels S1, S2, S3, and S4. The two dependent variables were WPM and error rate (RBA). From Table 4.7 we observe that 'sessions' is a significant factor for both WPM and error ($p < 0.05$). From post-hoc analysis with Bonferroni correction on the typing speed between pairs of sessions, we found that the typing speed differs between a pair of any two sessions ($p < 0.05$), except for sessions 2 and 3 ($p > 0.05$). Similarly, post-hoc analysis with Bonferroni correction on the error rate between a pair of any two sessions indicates that the difference between sessions S1 and S2, and S1 and S4 are significant ($p < 0.05$). Similar to foot gesture-based activation, since there is no significant difference in the error between session 2 and 3, and 3 and 4, these results suggest that the participants quickly learn to use the foot press-based activation.

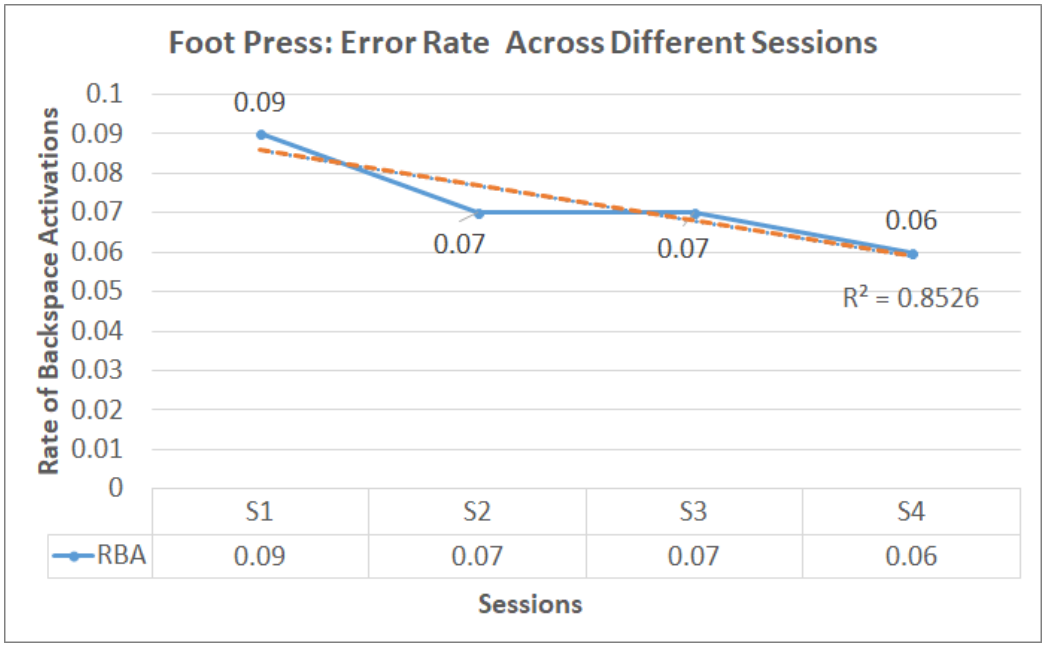

Figure 4.20: Foot press-based Selection: error rate expressed in terms of Rate of Backspace Activation (RBA) across different sessions. We observe that the error rate decreases with subsequent sessions, and the regression line has an R^2 value of 0.85

4.7.4 Gaze Typing Performance: Dwell Vs Gesture Vs Press

In the previous sections, we analyzed the typing performance of each selection method individually. In this section, we will analyze the typing performance of each selection method by comparing it against other methods. First, we analyzed the top typing speed and associated error rate of each selection method with one-factor ANOVA without replication. The independent factor was the "Selection Method" which had three levels Dwell (DW), Foot Press (FP), Foot Gesture (FG). The two dependent variables we considered are the typing speed (WPM) and error rate. The top typing speed and associated error rate for dwell-based selection was considered from the experiment where dwell time was set to 400 ms. Similarly, the top typing speed and associated error rate for foot gesture-based and foot press-based selections were considered from session 4 of the experiment for both types of foot-based selection methods.

Table 4.8 lists the results of ANOVA tests. We observe that the difference in typing speed between selection methods is significant ($p < 0.05$). Post-hoc analysis with Bonferroni correction

Table 4.7: Foot press-based Selection: ANOVA tests to understand the learning effects across the sessions (p values highlighted in gray indicate significance at $\alpha = 0.05$).

	Sessions [S1, S2, S3, S4]	**Std. Error**	**Post hoc Analysis**
WPM	F(3,48) = 44.324 *p = 0.000*	S1 = 0.473 S2 = 0.389 S3 = 0.373 S4 = 0.406	(S1, S2) *p = 0.001* (S1, S3) *p = 0.000* (S1, S4) *p = 0.000* (S2, S3) *p = 0.155* (S2, S4) *p = 0.000* (S3, S4) *p = 0.025*
Error	F(3,48) = 5.743 *p = 0.002*	S1 = 0.011 S2 = 0.008 S3 = 0.010 S4 = 0.008	(S1, S2) *p = 0.053* (S1, S3) *p = 0.150* (S1, S4) *p = 0.005* (S2, S3) *p = 1.000* (S2, S4) *p = 1.000* (S3, S4) *p = 0.985*

Table 4.8: Top Typing Speed: ANOVA for WPM and Error

	Selection Method [Dwell (DW), Foot Press (FP), Foot Gesture (FG)]	**Mean**	**Std. Error**	**Post hoc Analysis**
WPM	F(2,50) = 14.844 *p = 0.000*	DW = 11.648 FP = 14.98 FG = 13.814	DW = 0.437 FP = 0.406 FG = 0.470	(DW, FP) *p = 0.000* (DW, FG) *p = 0.003* (FP, FG) *p = 0.199*
Error	F(2,50) = 0.208 *p = 0.813*	DW = 0.069 FP = 0.060 FG = 0.068	DW = 0.014 FP = 0.008 FG = 0.007	(DW, FP) *p = 1.000* (DW, FG) *p = 1.000* (FP, FG) *p = 1.000*

indicate that the difference in typing speed is observed mainly due to the difference in typing speeds between dwell and foot gesture ($p = 0.003$), and dwell and foot press ($p = 0.000$), but not foot gesture and foot press ($p = 0.199$). We observe no difference in the error rates between the selection methods ($p > 0.05$). These results indicate that though users make the same amount of errors across selection methods, they differ by how fast they type using each method.

4.7.5 Gesture Vs Press-based selection

While we focused primarily on foot-based action, we further analyzed the gaze typing performance by only considering the foot gesture-based and foot press-based selection methods. Figure 4.21 and Figure 4.22 compare the gaze typing speed and error rate between the two foot-based selection methods.

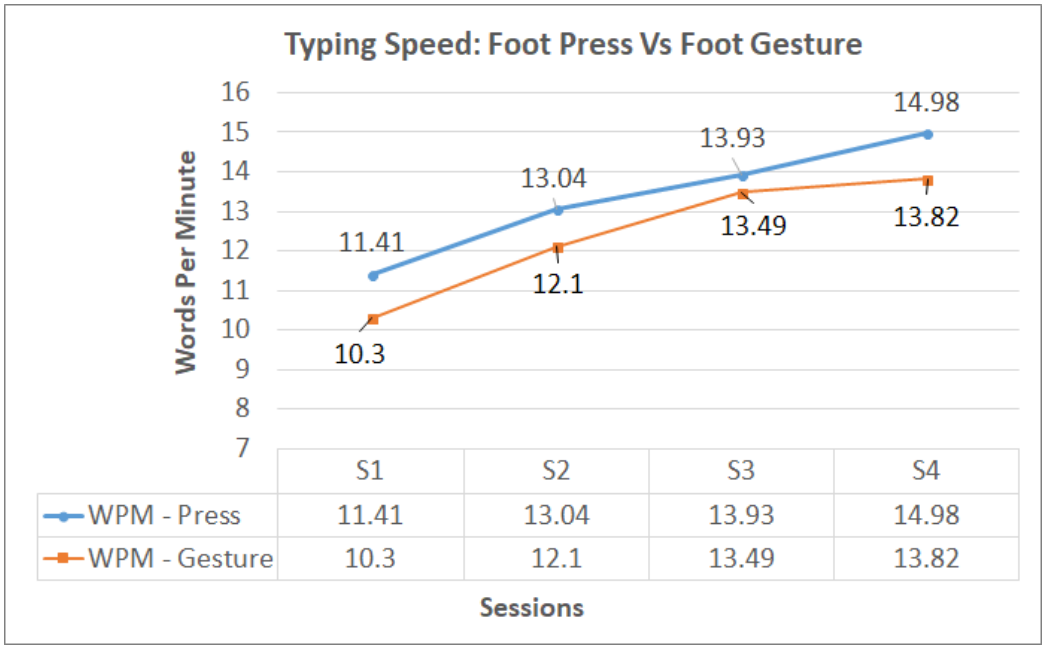

Figure 4.21: Typing speed comparison - foot press Vs foot gesture based selection: though generally the difference in typing speed between the selection methods is 1 WPM in any given session. From the two-way mixed factor model ANOVA we found that the difference in typing speed between the two selection methods is not significant ($F(1,32) = 2.008, p = 0.166$).

We performed two-way mixed model ANOVA with replication on the dependent variables:

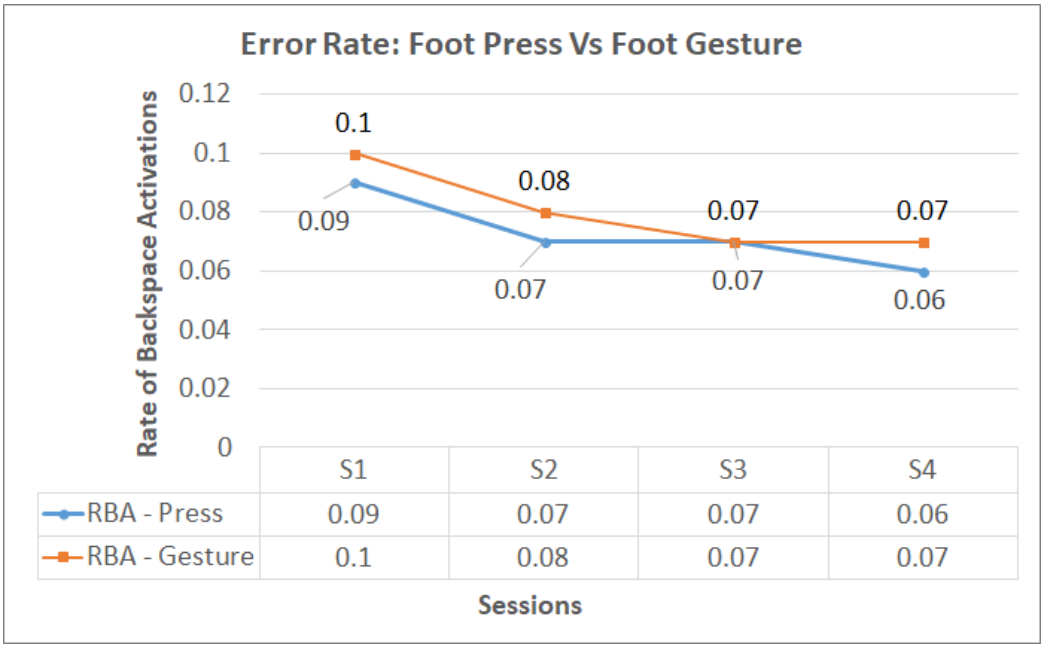

Figure 4.22: Error rate comparison - foot press Vs foot gesture based selection: though generally the difference in typing speed between the selection methods is 1% in any given session. From the two-way mixed factor model ANOVA we found that the difference in error rate between the two selection methods is not significant ($F(1,32) = 0.229, p = 0.635$).

WPM, error. The two factors (independent variables) we considered were: 1) selection method, and 2) sessions. The factor "selection method" is a between-subjects factor and it has two levels: 1) foot gesture-based selection, and 2) foot press-based selection. 'Selection method' is a between-subjects factor since the participants who gaze typed using the foot gestures were not involved in the evaluation of the foot press-based selection. "Sessions" is a within-subjects factor and has four levels: S1, S2, S3, and S4.

Table 4.9 lists results of ANOVA. We observe that neither the typing speed (WPM) nor error rate differ between the two foot-based selection methods ($p > 0.05$). However, consistent with previous analysis, the typing speed (WPM) and error rate differ between the sessions. Lastly, we observe no interaction effects between the $Selectionmethod \times Sessions$ for both WPM and error rate.

Since only factor 'sessions' was significant for both the dependent variables, the post-hoc analysis with Bonferroni correction for factor 'Sessions' is shown in Table 4.10. We observe that the

Table 4.9: Mixed Factor Anova: WPM and Error

	Selection Method [Foot press, Foot Gesture]	Sessions [S1, S2, S3, S4]	Interactions
WPM	F(1,32) = 2.008	F(3,96) = 90.179	F(3,96) = 1.060
	p = 0.166	*p = 0.000*	*p = 0.370*
Error	F(1,32) = 0.229	F(3,96) = 14.227	F(3,96) = 0.733
	p = 0.635	*p = 0.000*	*p = 0.535*

difference in typing speed between any pair of sessions is significant ($p < 0.05$), and the typing speed generally increases with subsequent sessions. We also observe that, difference in error between sessions S1 and S2, and S1 and S4 are significant ($p < 0.05$), but the difference is not significant between sessions S2 and S3, and S3 and S4. This observation with error is similar to what was observed in experiment 1 and 2, which indicates that irrespective of foot gesture-based or foot press-based selection, the participants reduce the error they make from session 1 to 2, from session 2 onward the changes in the errors observed are not significant.

4.7.6 Gaze Typing Usability - Qualitative Feedback

One of the main focuses of our work was to understand the advantages of using a supplemental input, specifically foot input, for gaze typing over just using dwell-based key selection. To understand the effectiveness of using foot over dwell input, we interviewed all the participants in our study. Following are the common feedback provided by the participants, based on the mode of input they used in the study.

4.7.6.1 Gaze and Dwell-based Typing

Positives:

- A longer dwell time, like 1000 ms, helps to avoid inadvertent selections.
- A shorter dwell time, like 400 ms, supports fast typing, and it is most suitable for users who cannot focus on a key for a longer time.

Table 4.10: Mixed Factor ANOVA: Post hoc Analysis for Sessions

	Mean [S1, S2, S3, S4]	Std. Error	Post hoc Analysis
WPM	S1 = 10.858 S2 = 12.570 S3 = 13.712 S4 = 14.398	S1 = 0.382 S2 = 0.339 S3 = 0.371 S4 = 0.311	(S1, S2) $p = 0.000$ (S1, S3) $p = 0.000$ (S1, S4) $p = 0.000$ (S2, S3) $p = 0.000$ (S2, S4) $p = 0.000$ (S3, S4) $p = 0.031$
Error	S1 = 0.095 S2 = 0.073 S3 = 0.070 S4 = 0.064	S1 = 0.007 S2 = 0.006 S3 = 0.007 S4 = 0.006	(S1, S2) $p = 0.000$ (S1, S3) $p = 0.907$ (S1, S4) $p = 0.000$ (S2, S3) $p = 1.000$ (S2, S4) $p = 0.325$ (S3, S4) $p = 1.000$

- Gaze typing greatly helps people with a disability to communicate with others.

General feedback, limitations and Suggestions:

- With a longer dwell time (1000 ms), it becomes harder for typing as one is required to focus on the target key for a longer time.
- With a shorter dwell time (400 ms), though the typing becomes fast, a lot of errors were made and corrected.
- With a shorter dwell time, it is difficult to correctly enter the same character twice consecutively.
- A shorter dwell time is more cognitively demanding.
- The user is forced to look away from the keys, or onto the text input area when not typing.

4.7.6.2 Foot Gesture-based Selection

Positives:

- “I thought it worked pretty well, I felt like you get into a rhythm after a while where you don’t even really think about typing, you just kind of do it every time you look at a new character."
- “It wasn’t hard to work with the device at all, it wasn’t straining, it was really easy not much thought involved with that. I did like it, it’s a pretty cool concept."
- “When I got the hang of it, there wasn’t much that bothered me. I got used to the rhythm, I would know to tap my foot."
- “Once I got the rhythm of it seemed like a pretty natural combination to me, but getting started, each sentence I had to remind myself to tap."
- “I thought it was really cool, pretty intuitive, I was able to pick it up pretty quickly."
- “At first I have to think about it, but after many sentences, I didn’t have to think about it, it was just natural."
- “I did the toe tap pretty much the whole time, it got easier. I would do the heel tap if I am leaning back in my chair more, I had to sit up really straight, so I would only do the toe tap."
- “After some practice, I definitely got faster, more accurate."
- “Foot gestures were comfortable, I prefer heel tap."
- “At the begging it was a little bit weird, after getting to play with it a little bit more, I figured out that I had to move the device a little bit and it detects it. That made it nicer."
- “I didn’t really have to think about it, it kind of just became a rhythm."
- “It was pretty automatic, I would get into a rhythm."

- “I had some difficulty with the consistency of the feedback, trying to figure out just how high up or how much force to go down, but towards the end it was a lot easier to have that kind of measure."
- “I was pretty used to tapping, but there were couple of movements I was not sure if it registered the tap, but generally I was able to keep tapping really fast."
- “Unless you are writing an essay it’s not strenuous. For what we did it’s not strenuous."

General feedback, Limitations, and Suggestions:

- “Sometimes I try to move my gaze fast before my foot tap that caused a lot of errors, that’s a coordination problem."
- “The error was not due to hitting the wrong letter, but having to stop at the right letter, your eyes are gone before its processed."
- “Tapping the floor feels like tapping the keyboard, I am pressing something, it gives more completion than right and left flicks."
- “Sometime it was difficult to tell whether moving the foot up or moving the foot down what was actually going to register as a foot tap."
- “I used the toe until my foot got tired and then I would use the heel a little bit to give my toe a break."
- “I found myself doing a combination of left and right flicks and I found that worked much better, it was a little more faster, my toe was mostly in the air."
- “My eyes are already moving to the next letter before I hit it with my foot."
- “I would get into a rhythm, once it broke, it was hard to get back into it. It takes a few seconds to get back into it."

- “In the first couple of rounds, I was like having slow it down a little bit, and make sure that my eyes were adjusting, I was getting each specific letter, making sure I was not going too fast picking wrong letter, eventually I was able to go faster, because my eyes would kind of like catch as I was going."
- “I did have to consciously think when switching gestures, but once it’s switched it wasn’t too bad."
- “I mainly used toe tap, I did try the heel tap at the end but I couldn’t figure out the consistency with with I had to actually push my foot down, I just stuck with toe tap."
- “I just used toe, and I guess that’s just the one I picked, flick I tried but I didn’t like it, heel was fine, but I just used toe because it was most comfortable."
- “I initially started using the heel tap the most, but overtime I found that the front foot tap was easier (toe tap) and less strenuous. I rarely used the left and right."
- “I preferred the toe tapping, I started with toe tapping, but switched to different ones to try them out, but I still liked the toe tapping the most, that’s the one that I am most comfortable with."
- “I tried using the heel, I liked the toe tap most, I did not use right and left at all."
- “I liked toe tap the most, heel is harder, now and then I used left flick."
- “I used the toe tap, that was the most comfortable to use, I did use other gestures every once in a while, but toe tap was most comfortable to me."
- “I used mostly heel tap, I found it feel most comfortable to me, there was some sync issues with left and right."

4.7.6.3 Foot Press-based Selection

Positives:

- “I enjoyed how simple the system was. You literally just look at the letter you want to type, and tap your toe to select it."
- “Gaze typing is a solid system, accuracy could be improved, but I felt overall that the system performed well for the functions it has."
- “The gaze typing experience was good, it worked well, and the user experience was pretty smooth."
- “The system was usable and pretty accurate. I was missing letters, because I was trying to move fast."
- “The concept was really cool, I liked that some letters were enlarged compared to other ones, it did help me a lot."
- “To type, I did not have to exactly look at the character. I was looking in between letters, and I could see the letter in my periphery."
- “It was a good way to allow people with disability use a keyboard in a more conventional way rather than all voice input (but it is getting better). Also, if you are tying something that is not in the dictionary you can’t do that with the voice, but you can do that with gaze typing."
- “I really enjoyed it, it was quite a novel experience, not using any hand held to device to go about it."
- “The input noises, and highlighting were good. You always knew what you are looking at."
- “The whole gaze typing was pretty neat and easy to use."
- “After typing few phrases, clicking with the foot became natural."
- “I really liked the idea, I thought it would be difficult at first, because I have to sync my eye movement with the big toe pressing on the sensor, but that came on very easy, that’s good."

- “It came by instinct, after the first sentence, it was just like an automatic response for the foot to keep on doing that, that was pretty easy."
- “It was comfortable enough to quickly and easily get the motions right, but at the same time I felt it is little bit sensitive. Sometimes when I tried to click one button, it would click twice."
- “Foot input wasn’t tiring, it takes some time getting used to. Because, I wasn’t used to use my foot that way before."
- “Once you know how much pressure you need to put, it is ok. At first I was double typing."
- “The feedback from the eye tracking system was sufficient."
- “For short phrases it is not straining, if I was googling something, or like having a short conversation it is fine."
- “One can sit, relax, and do the whole interaction."
- “The system is definitely useful for people with disability."
- “The system was intuitive, pretty good, and it is a really a good project."
- “Once the system is calibrated it is easy to work with."
- “For a casual use it is pretty good."
- “For majority of the time I was able to type everything."
- “I was getting faster as I kept typing. It felt pretty good."
- “It is pretty responsive, it is pretty easy to correct mistakes."
- “It is not hard to learn how to use. It did not take long to adopt."
- “I did not even think about the possibility of typing using one’s feet."
- “It was interesting and really useful to accessibility."

General feedback, limitations and Suggestions:

- “The foot sensor could have more feedback on it when pressed. It was too easy to press multiple times, and better feedback like a click or vibration when pressed would help that."
- “The system becomes a bit tiring after prolonged use (mostly my toe, my eyes were fine)."
- “I had to re-position my foot a lot because the pressure pad was large."
- “Initially, I had to learn a little bit to understand how much to press."
- “Need click or vibration in the foot as feedback."
- “After using the system consistently for some time I felt tired."
- “Sometimes it is hard to find where exactly is the pressure sensing area. Hence, sometimes you have to look for where to tap, bind it to some place."
- “Sometimes I pressed the same key twice when I didn’t need to. May be my eyes moved slow compared to my foot input. My foot input meant to press the next character, but my eyes were not there yet. More practice, and I will be constantly aware that I need to push down."
- “It gets tiring after a while because of not blinking and staring at the screen. It takes a lot of mental power focusing rather than typing with the hand because you have muscle memory with typing, you know where the keys are."
- “It is hard to verify what you are typing. You need to really stay focused."
- “Physical feedback like a switch would provide better feedback for the foot."
- “Erroneous selections can be avoided by placing the characters a little far away."
- “Space key functionality can be moved to left foot."
- “The shoe has to be a better fit for each individual person."

- "For the foot pad the clicker should be bigger and it shouldn't rely on the single toe. It would be easier to not misclick. You are focusing more on using that one toe."
- "Ability to use all your toes would be helpful."

4.8 Discussion

In this section we revisit the research questions discussed in Section 4.3. First, we wanted to answer if users can coordinate their gaze and foot input enter text on a computer. Results from experiment 2 and 3 demonstrate that, irrespective of the foot gesture or foot press based-selection, with a short learning curve, i.e., typing about 10 phrases, users do conveniently coordinate their gaze and foot input to enter text on the computer. Also, typing at a comfortable speed, the typing speed reaches a plateau at round 14 WPM. Regarding the error rate, from both experiment 2 and 3, we observe that the error steeply reduces from session 1 to 2, and there on no significant change is observed. This indicates that users quickly learn to coordinate their gaze and foot, and make significantly less errors. From the qualitative feedback, discussion in Section 4.7.6, we learn that after typing a few phrases, users develop a rhythm in pointing with gaze and selecting with foot such that the foot interaction becomes natural. The users do not have to consciously remind themselves to achieving the gaze and foot coordination.

Second, while we intended to compare the performance of dwell-based selection with foot-based selection, the comparison will not be fair since the lowest dwell time used was 400 ms. While 400 ms is in the range of the lowest dwell times used in the previous studies [171, 151, 76, 74, 157], and many participants also reported that dwell time of 400 ms was demanding and unnatural to use, the average typing speed achieved was 11.65 WPM in contrast to 13.82 WPM with foot gestures and 14.98 with foot press-based selection. ANOVA test discussed in Section 4.7.4 showed that when considering the top typing speeds of the three selection methods, the difference in typing speed was significant between dwell and foot-based selection methods ($p < 0.05$). However, there was no significant difference in the error rate ($p > 0.05$). All the methods had an error rate of nearly 6% when typing at the top speed. As the goal of our study was not just to compare the

performance, but was also to consider the usability aspects of the selection methods, the foot-based selection methods overall fairs better than dwell-based selection.

Third, focusing specifically on foot-based selection methods, ANOVA tests in Section 4.7.5 revealed that there is no significant difference in typing speed as well as error rate between the two selection methods ($p > 0.05$). While foot press is less physically intense than a foot gesture, we expected that foot-press based selection would achieve a higher typing speed. Though foot press-based selection generally achieved 1 WPM higher speed than foot gesture-based selection, the difference was not significant. Also, there was no significant difference in the error rate ($p > 0.05$). Hence, we infer that subtle foot press-based selection or distinctive foot gesture-based selection achieve the same performance.

Fourth, ANOVA tests from experiment 2 (Section 4.7.2) demonstrated that though users were provided with four gestures to be used as selection methods, users generally choose a single gesture and use the same gesture throughout the study. This observation again contradicts our hypothesis that availability of multiple gestures, i.e., the ability to orienting the foot in different directions, enables to user to switch between gestures when they get tired with one gesture. However, it appears that users prefer to use the same gesture with which they have developed a rhythm than switching to a new gesture and familiarizing with it. As toe tapping was used significantly higher than other gestures, and most uses shared that they preferred toe tapping out of all the gestures, we infer that toe tapping is the most efficient and convenient gesture for foot gesture-based interactions. Also, we suggest, unless multiple gestures are required to achieve different input actions, incorporate only toe tapping gesture. Redundancy of gestures may not improve performance.

Fifth, from both experiments 2 and 3, we observed a learning effect where the difference in the typing performance between the sessions was significant. The typing speed increases with subsequent sessions, and reaches a plateau after typing nearly 20 phrases. However, the error rate quickly reduces within typing 10 phrases and stays nearly unchanged from there on.

Sixth, does gaze and foot-based system induce physical strain and cognitive load. Though participants typed for one hour with intermittent breaks between the sessions, except for a few

participants the majority of participants (nearly 88%) reported that foot interaction, either gestures or press, was not strenuous when inquired during the post-study interview. Some participants did express that typing for nearly one hour was much strenuous on their eyes but not as much on their foot. Most of the participants reported that they got into a rhythm with foot gestures or press, and they needed to remind themselves to use their foot only in the beginning of the study. Specifically, participants felt that toe and heel tapping was natural and comfortable, as users have a natural inclination to foot tapping. We also observed that users hardly switched between foot gestures, but whenever they did, the switch was between toe and heel tapping or right and left flicks. So, they switched between symmetric gestures.

Lastly, regarding the advantages of using a supplemental foot input for gaze typing. From the interviews we found that the performance of gaze and foot-based typing is primarily dependent on coordination of pointing with gaze and selecting with foot (press or gestures). Often, participants reported that major reason for errors was not because they typed the wrong letters, but, their gaze moved fast before the current letter is selected. Irrespective of foot press or gesture based selection, the learning curve was short. The participants were comfortable with foot interaction within typing a few phrases. Overall, the participants seemed excited with the possibility of gaze and foot-based typing, and highly liked the usability and applicability of our system specifically in the scenarios of situationally induced impairments and disabilities.

4.9 Conclusion

Gaze typing is becoming one of the crucial input modalities for text entry in two scenarios: 1) situationally-induced impairments and disabilities (SIID), and 2) physical impairments and disabilities. In these two scenarios where a user would be unable to use their hands for typing, gaze typing provides a way for text entry on a computer with the help of an on-screen keyboard (virtual keyboard—VKB). The majority of gaze typing systems use dwell-based selection of the target key. This method has multiple limitations like increased attentional demand, high error rate with lower dwell time, and low typing speed with higher dwell time. Though dwell-free systems that use language modeling to suggest words or characters address the issue of finding an optimal dwell

time, or constantly adjusting the dwell time. These systems still result in inadvertent selections of unwanted keys. Additionally, the user is continually forced to switch focus between scanning the suggested words and typing. Furthermore, the improvement achieved in the typing speed is minimal. To address some of these concerns, we presented a dwell-free, multimodal, gaze typing system that uses a supplemental foot input for selection of the target keys. We implemented two methods of foot-based selection: 1) foot gestures, and 2) foot press. Additionally, we enhanced the standard QWERTY keyboard by modifying the layout, and the dimension of the keys to improve gaze typing performance. We tested the efficacy and usability of all three selection methods–dwell, foot gestures, and foot press–through three experiments. Each experiment had 17 participants, and a total of 51 participants took part in the study. We ensured that no participant that participated in one experiment took part in the other. This was done to prevent the chances of the familiarity developed by participating in one experiment influencing the other.

From the three experiments, we found the following observations. First, users can comfortably coordinate their gaze and foot input in achieving text entry on a computer. Overall, for gaze typing, foot-based key selection has higher efficacy and is a preferred method over dwell-based key selection. Second, toe tapping is the most preferred foot gesture for gaze typing, and we believe this also translates to point-and-click interactions on a computer. Third, though multiple gestures like toe tapping, heel tapping, right flick, and left flick, might be mapped to perform a single action (e.g., click), users prefer to select a single gesture and use it throughout. Users do not prefer to switch between using different gestures (to avoid the burden of learning a new gesture). Fourth, while subtle foot press-based selection may appear to be less straining and a faster selection method, we found no difference in the performance when using foot press-based selection or foot gestures. Lastly, when using foot-based selection, either foot gestures or foot press, the users quickly develop a rhythm between pointing on the target with their gaze and performing foot-based selection.

5. GAZE-ASSISTED AUTHENTICATION

We discussed in Introduction (Chapter 1) that authentication is one of the primary interaction tasks performed on a computer in addition to point-and-click and text entry interactions. Knowledge-based authentication, i.e., authenticating by password entry still remains a primary way of authenticating on a computer [172, 173]. However, other authentication methods like Touch ID [174, 175], graphical passwords [176, 177], face recognition [178], and various physiological and behavioral biometrics methods have been explored [179, 180]. Except for a few biometrics-based authentication methods like face recognition [178], iris recognition [181, 182], gait recognition [183, 184], etc., majority of the authentication methods today need an active involvement (motor functions) of the user. By active involvement, the user is required to either touch a sensor, type a password, or select objects on the screen with a mouse to authenticate. The need for active involvement of the user to authenticate may not be suitable in various scenarios like situational and physical impairments.

While the need for accessible and hands free authentication method is crucial, the existing standard (PIN or graphical) authentication methods suffer from shoulder surfing attacks. Shoulder surfing enables an attacker to gain authentication details of a victim through observations and is becoming a threat to visual privacy. To safeguard from shoulder surfing, numerous solutions like graphical passwords, tactile interfaces, gaze-based PIN entry, and so on have been proposed. Existing gaze-based solutions are limited by low accuracy, need for precise gaze input, and are susceptible to video analysis attacks. Hence, to authenticate in the scenarios of situational impairments and physical impairments, and to prevent shoulder surfing attacks, we have developed a gaze-assisted authentication method[1]. In specific, we have developed two gaze-assisted authentication strategies,

[1]*Parts of this chapter are reprinted with permission from "A Gaze Gesture-Based User Authentication System to Counter Shoulder-Surfing Attacks" by Rajanna et al., 2017. Publisher and Copyright ACM Digital Library, 2017, New York. Conference - CHI'17 Extended Abstracts: CHI Conference on Human Factors in Computing Systems Proceedings - doi.org/10.1145/3027063.3053070, and "DyGazePass: A Gaze Gesture-Based Dynamic Authentication System to Counter Shoulder Surfing and Video Analysis Attacks" by Rajanna et al., 2018. Publisher and Copyright holder IEEE, 2018. Conference - 2018 IEEE 4th International Conference on Identity, Security, and Behavior Analysis (ISBA), doi.org/10.1109/ISBA.2018.8311458

and they are 1) fixed transitions authentication, and 2) dynamic transitions authentication. The remaining sections of this chapter are organized as follows. In Introduction (Section 5.1) we will discuss the need for accessible, hands-free, and shoulder-surfing resistant authentication methods. In Prior Work (Section 5.2) we will discuss all the well known shoulder-surfing resistant authentication strategies that use gaze input. The fixed transitions authentication method is discussed in Section 5.3. We discuss the design motivation, system implementation, gesture recognition, experiment design, results, and advantages and disadvantages of fixed transitions authentication method. Next, in Section 5.4 we discuss the dynamic transitions authentication, and also discuss how the interface dimension, speed of transitions, and the level of randomness influence the recognition accuracy. Lastly, we will be discussing the system's susceptibility to video analysis attacks (Section 5.4.11.2) through single and dual video iterative attacks. These results from hacking studies are compared against hacking PIN-based authentication systems. The chapter will be concluded (Section 5.4.12) with a discussion on how various parameters influence the authentication accuracy, and the parameters should be optimized for better higher recognition accuracy.

5.1 Introduction

With the advancement of the internet and availability of affordable computing devices, most of the services are offered digitally. For example, online bank transactions, ATMS, self service kiosks, online library, ability to paying bills online, order food, shopping, and so on. However, the services offered online should be protected, and this is achieved by controlling who can access these services. An individual or a group of users that are authorized to use the services can do so by authenticating themselves. However, the major concern is that most of the authentication methods used are password-based, and the user is required to enter the password through a keypad or mouse [185, 186]. These authentication methods pose a challenge to people with disabilities as they experience a wide range of difficulties due to the lack of or inefficient accessible authentication methods [185]. Though there exists a few accessible authentication methods, they provide less security than intended [187]. Generally, for individuals with Parkinson's disease, dyslexia, vision impairment, motor impairments the security and usability of different authentication mechanisms

are significantly impacted [186]. This obviates the need for accessible, hands-free authentication methods.

On the other hand, shoulder-surfing is a significant issue for user authentication, due to its nature of attackers looking over a victim's shoulder to extract confidential information, and continues to be a growing problem [101, 188, 189]. The information targeted by an attacker (observer) comprises of a broad range of personal information of a victim (user) like user's interests, hobbies, sexual preferences, and login credentials [188]. ATMs and Kiosks will provide different kids of services based on entered credentials, for example, a manager will have higher privileges than an employee [190]. Keypad monitoring commonly occurs at public places like ATMs, Kiosks, airport lounge, coffee shops, and even airplanes, and semi-private spaces like offices to steal login credentials of the user [190]. Adeoti et al. [191] reported that shoulder surfing is one of major source of ATM frauds, and counted for 21.2% of ATM frauds in Nigeria. Shoulder surfing is considered as the second most ATM fraud (21.2%) after card jamming(24%) [191].

A report on global visual hacking, presented by Ponemon Institute in 2016 states that in business office environments attacks happen on laptops, tablets, and smartphones etc [192]. They conducted shoulder surfing attacks in eight countries, and a staggering 91% of visual attack were successful resulting in 613 units of breached data of various types [192]. 11% (69 units) of the breached data were login credentials that could further provide access to more sensitive information putting the company at risk. To further worsen this problem, the organizations are creating open work spaces (no wall or cubicle) to encourage collaboration and such an environment is conducive to visual hacking, as the prying eyes can easily see the computer screens and input devices without even the notice of the victim [192]. Shoulder-surfing attacks are becoming a common occurrence in crowded places, however the victim is hardly able to recognize the potential attacker [37, 193, 194, 188, 192]. Furthermore, the availability of vision-enhancing devices like long-range binocular, thermal camera, surveillance cameras [195], and the usage of drones [196] can further assist the attacker. Hence an effective system to counter shoulder-surfing attacks is imperative.

Multiple solutions have been proposed to prevent shoulder surfing attacks on authentication, and those solutions that use gaze-only input can be used as an accessible authentication method. Based on the authentication strategy used the existing solutions against shoulder surfing can be classified into multiple groups. For example, Stroke- or image-based graphical passwords [197, 198, 199, 200, 194], PIN entry methods through cognitive trapdoor games [201], PIN entry method based on vibration and visual information [202], SmudgeSafe authentication system [203], authentication by up or down touch gestures [204], stroke-based password on front and back of a smartphone [205], and gaze-assisted authentication [104, 79, 78, 37, 77]. Some of these solutions are built for mobile devices [77, 205, 203], and others target large screens like ATM or computer screens [197, 78, 79, 104]. Each of these systems have unique design advantages when considering the combination of authentication method and the target device. In this research we focus on gaze-based authentication, which has been previously explored by Kumar et al. [37], Bulling et al. [102], Luca et al. [78], Alain et al. [206], Vidal et al. [80], and Cymek et al. [79]. Similar to these solutions, our gaze gesture-based authentication system targets computer and ATM screens, or in general, systems that can be equipped with an eye tracker. Previously proposed gaze-based solutions use gaze input to enter an alpha numeric PIN [37], fixate on certain points on an image [102], draw specific gestures with eye movements [78], or follow moving objects in definite paths [80, 79]. Theses solutions are limited by low accuracy, the need for precise gaze input, requirement for users to remember the gestures, and the susceptibility to video analysis attack.

5.2 Prior Work

As previously discussed, authentication methods that use gaze input to prevent shoulder surfing attacks will also serve as accessible authentication methods. Solutions to shoulder-surfing use dynamic interfaces [207], and multi-modal input methods [78, 202]. In this section, we discuss some of the gaze-based solutions to address shoulder-surfing attacks. Prior research in gaze-based authentication methods to prevent shoulder surfing can be classified across four broad categories.

5.2.1 PIN and Gaze-based Authentication

Kumar et al. [37], presented “EyePassword," an authentication method where the user enters sensitive input like password or PIN by selecting from an onscreen keyboard using their gaze. The authors compared various combinations of input methods like Gaze + keyboard input based trigger and gaze + dwell time. They found that gaze + dwell - based password entry takes marginal additional time than keyboard-based password entry, while having similar error rates to keyboard-based password entry. Khamis et al. [77], presented “GazeTouchPass," which allows authentication on mobile phones though multiple switches between gaze and touch input modalities. The system uses the front camera of the phone to recognize the direction of the user’s gaze as left or right. The authors found that overall system accuracy with 0 input errors was 65%, and the system was usable and significantly secure than single-modal authentication.

5.2.2 Gaze Gesture-based Authentication

Luca et al. [78], presented "Eye-Pass-Shapes", where a user authenticates by drawing one of the eight gestures with their eye movements. The system evaluations showed that the eye gestures significantly increases security while being easy to use. Best et al. [104], presented a rotary interface for gaze-based PIN code entry. The solution eliminates dwell based numeral selection on a grid-based keypad by relying on weighted voting scheme of numerals whose boundaries are crossed by the streaming gaze points. The authentication accuracy was found to be 71.16% with PIN interface and 64.20% with rotary interface.

5.2.3 Gaze and Image-based Authentication

Bulling et al. [102] presented a novel gaze-based authentication system that makes use of cued-recall graphical passwords on a single image. During password selection, image areas which are likely to attract visual attention are masked. Through a threat model, the authors demonstrate that their method is significantly secure than a standard image-based and gaze-based authentication methods. Alain et al. [206], presented "Cued Gaze-Points", an eye gaze version of cued-recall graphical passwords. To authenticate, the user selects specific points on five sequence of images,

and the selection is achieved by looking at the desired point and holding the space bar. This method supports larger password space and the cued-recall nature helps users to remember multiple distinct points.

5.2.4 Gaze Pursuit-based Authentication

Vidal et al. [80] presented the idea and the design of authentication using eye pursuits. The authors proposed a pursuits-enabled screen that displays an animation of fishes swimming in the fish tank. The user can authenticate by looking at four specific fishes in the precise sequence. Cymek et al. [79], presented an authentication method, where the user follows the digits moving in vertical and horizontal directions to authenticate. The system achieved an accuracy of 97.57% in recognizing the digits entered.

As discussed above, the common limitation with gaze-based authentication systems summarizes to having low accuracy. As evaluated by DeLuca et al.[103], the error rate of various well known gaze-based authentication methods varied from 9.5% to 23.8%. Also, gaze-authentication is susceptible to video analysis attacks as demonstrated in [102, 78]. Though Cymek et al. [79] had an accuracy above 95%, however, this work was not evaluated for video analysis attacks. Lastly, authentication systems that expect the user to remember gestures or specific points on an image may be very overwhelming [102, 78]. The contribution of our work derives from trying to address these limitations. In the following sections, we will discuss two gaze-assisted authentication strategies: 1) fixed transitions authentication, and 2) dynamic transitions authentication. Both the strategies use gaze gestures, and they have high accuracy and robust to calibration errors as we leverage template matching to recognize the gestures.

5.3 Fixed Transitions Authentication

The fixed transitions authentication is a gaze gesture-based system that combines gaze with gesture recognition. The system authenticates users from their unique gaze patterns onto moving geometric shapes (Figure 5.1). The system authenticates the user by comparing their scan-path with each shapes' paths and recognizing the closest path. The interface comprises of 36 moving

shapes (Figure 5.2), and to authenticate, the user has to follow three shapes, one on each frame, on three consecutive frames. A frame is a five-second duration where all the shapes simultaneously move from their source location to destination location. Three secretly selected shapes constitute a user's password.

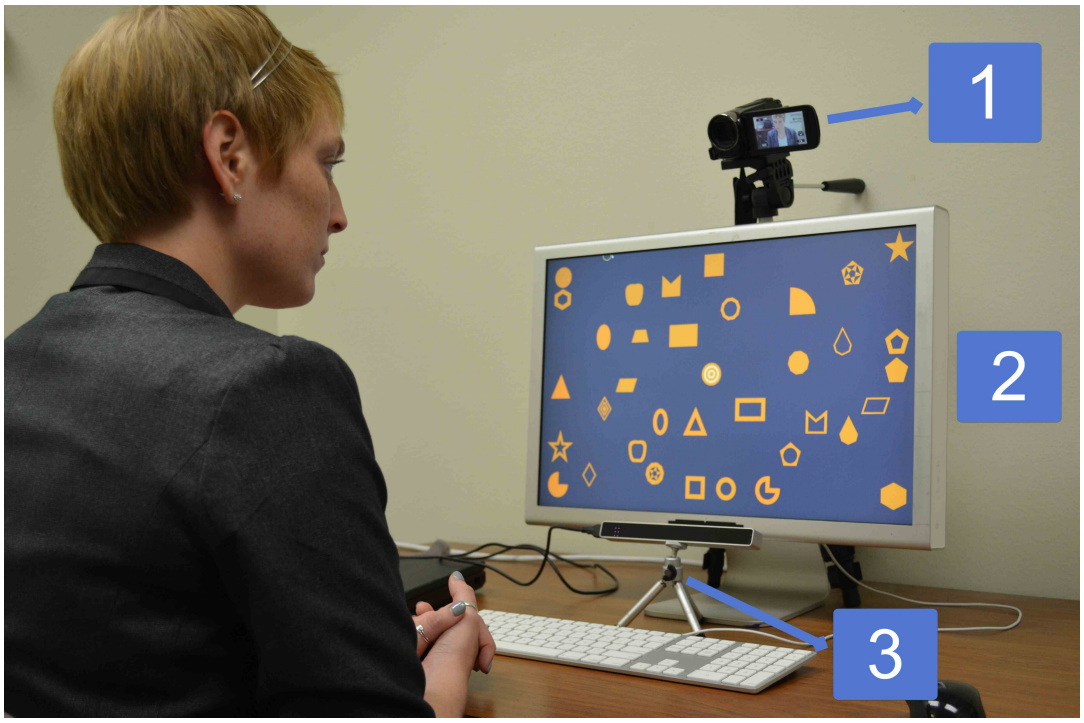

Figure 5.1: Gaze Gesture-Based Authentication System: A user is authenticating by following the three shape gaze password. [1 - Camera, 2 - Authentication interface, 3 - Eye tracker].

For successful authentication, the scan-paths of the user's gaze should match with the traversed paths of the correct shapes in the three frames. Also, of the 36 shapes only 12 shapes can be selected for a password, and the remaining 24 are fake shapes. Our approach is similar to the idea of pursuit-based authentication presented by Vidal et al. [80]. However, we use gesture recognition principles for scan-path matching, and this method supports high accuracy even with a large number of shapes and complex traversal paths. In addition, fake shapes introduce randomness to frustrate potential attackers from unauthorized access through guess work. In a study with 15

users, authentication accuracy was found to be 99% with true calibration and 96% with disturbed calibration. Also, our system is 40% less susceptible and nearly nine times more time-consuming to video analysis attacks compared to a gaze- and PIN-based authentication system.

Figure 5.2: Gaze gesture-based authentication interface with 36 shapes. Each shape has a fixed starting and ending points, and traverses along a predefined path. Out of the 36 shapes 12 are true shapes available for password selection, and the remaining 24 are fake shapes not considered during password selection

5.3.1 Design Motivation

Prior research by Wendy et al. [208], Hoanca et al. [209], Davis et al. [210], has shown that graphical passwords such as static images or user-drawn gestures are easier to remember than PIN passwords. Graphical passwords are mainly static images or user-drawn gestures. User-drawn passwords involve two processes: 1) visual recall of the drawn password, and 2) recall of the temporal order [211]. This concept has been adopted in Microsoft Windows picture log-in, which allows users to perform three gestures (tap, line, circle) on specific locations of an image to

authenticate. Our interface uses moving shapes like pentagons, triangles, and circles as opposed to pictures. One of the reasons for this design is a study by Madigan et al. [212], which showed that objects are more easily remembered than pictures in a free recall task. Also, a study by Bower et al. [213], showed that a series of line drawings are poorly remembered if the subject is unable to interpret the drawings in a meaningful way. Primarily, we wanted to liberate the user from remembering complex gestures and the order of strokes that constitute a gesture. Thus, we made the interface such that the user is only required to remember the shapes that constitute the password but not required to remember the gestures and their constituent strokes. Users can then follow the shapes' movements to authenticate. In addition, moving shapes take advantage of the mechanism of smooth pursuit (foveal pursuit), an ability of the eyes, with the goal of keeping the visual projection of a small moving target continuously on the center of the fovea [26, 214]. In addition, our approach of moving shapes takes advantage of the mechanism of smooth pursuit (foveal pursuit), an ability of the eyes, with the goal of keeping the visual projection of a small moving target continuously on the center of the fovea [26, 214].

5.3.2 Hypotheses

Considering the limitations with prior research and design motivations for our system, we form the following hypotheses: a) the gaze gesture-based authentication system achieves high accuracy and is robust to calibration errors, b) users commit fewer or no errors when entering passwords with successively repeated shapes (like pie, pie, circle) on our system, and c) our system is less susceptible and more time consuming to video analysis attacks than gaze- and PIN-based password entry systems.

5.3.3 System Architecture and Implementation

The gaze gesture-based authentication system (Figure 5.1) consists of two main modules: 1) Gaze Tracking Module, and 2) Authentication Engine.

Gaze Tracking Module: The system uses "The Eye Tribe" tracker[2], which is a table mounted

[2]theeyetribe.com

eye tracking sensor that provides (X,Y) coordinates of the user's gaze on the screen. For eye tracker to work efficiently, the user is positioned such that the face is centered in front of the monitor at a distance of 45 - 75 cm and the eye tracker error is 0.5°-1° of visual angle.

Authentication Engine: Authentication engine is the central module that runs on a computer and receives (X,Y) gaze-coordinates from the eye tracker. This module is responsible for positioning the circles at random locations on the interface, and generating a random path for each circle. The module also implements the scan-path matching algorithm to authenticate a user.

5.3.4 Authentication Procedure

5.3.4.1 Password Selection

To choose a password, a user selects three shapes from a password selection interface that lists the 12 true shapes. The first shape selected is followed on the first frame, the second on the next frame, and so on.

5.3.4.2 Authentication Interface

The authentication interface is shown in Figure 5.2. The interface is a canvas with 36 shapes placed at different locations on the screen: 12 are true shapes available for password selection, and the remaining 24 are fake shapes not considered during password selection. Each shape is assigned a predefined starting and ending points, and a path along which it traverses. Hence, the user is not required to search for password shapes once their initial locations are known.

For each true shape, there are two fake shapes placed at different quadrants on the screen which perform similar transitions as the true shapes. We introduced fake shapes for two reasons: 1) in brute force attacks, an attacker without knowledge of the fake shapes must assume a password complexity of $36 \times 36 \times 36 = 46,656$, whereas the true complexity is $12 \times 12 \times 12 = 1728$, and 2) in video analysis attacks, fake shapes introduce enough randomness in the system that it becomes hard or time-consuming to recognize the exact shape through guesswork.

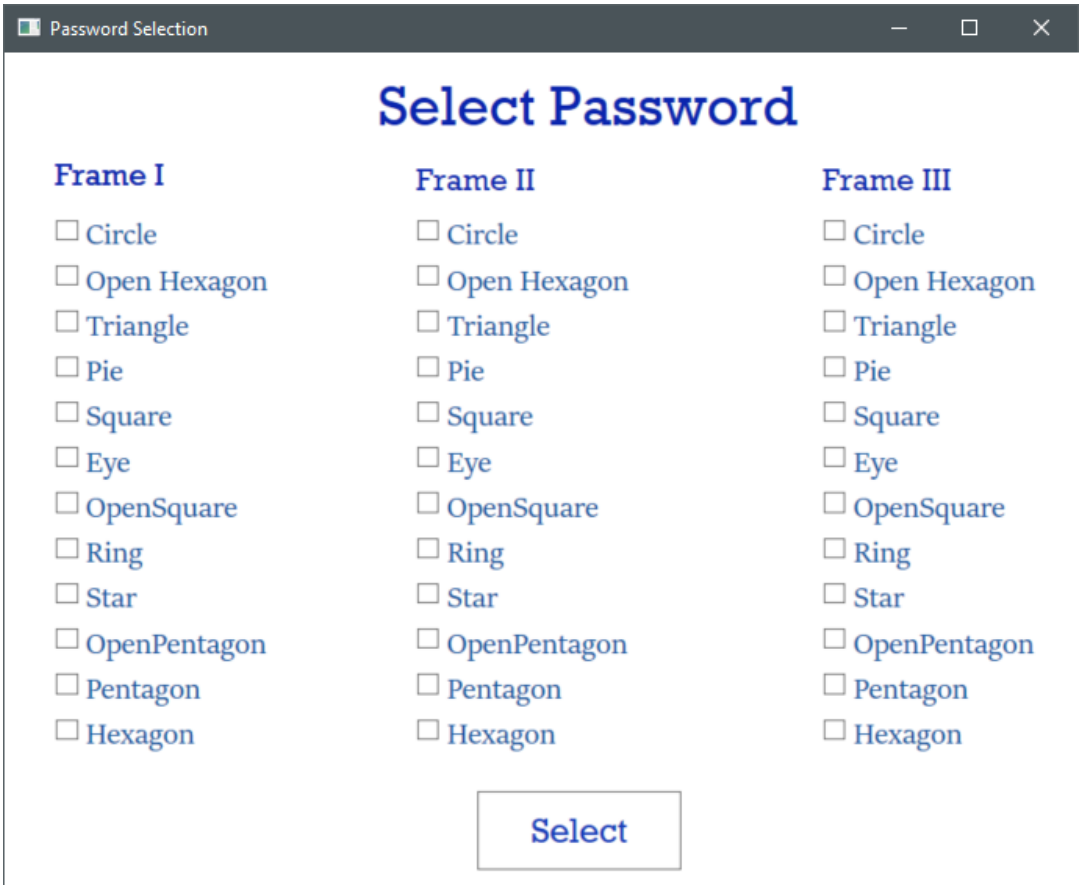

Figure 5.3: Password selection interface that lists the 12 true shapes. The user selects a single shape on each frame as a password (e.g., Start, Pie, Hexagon).

5.3.4.3 Authentication in Action

To control the interface, the user presses a set of hot-keys: 'A' to initiate movement of shapes and record gaze data, 'Z' to recover from user mistakes (blink, sneeze, losing the path) and discard recorded gaze data, and 'M' to submit the password after following 3 shapes. We minimize authentication failures since users have direct control over each frame. For example, if a user selected *Square-Star-Pie* as a password, then the user is authenticated by following each shapes' paths in their respective frames, as shown in the sequence of Figures 5.4, 5.5, 5.6. The user does not receive any feedback, since the gaze point and scan-path are hidden. If the user follows the correct password shapes in the correct sequence, she is authenticated, otherwise, the access is denied as shown in the Figure 5.7.

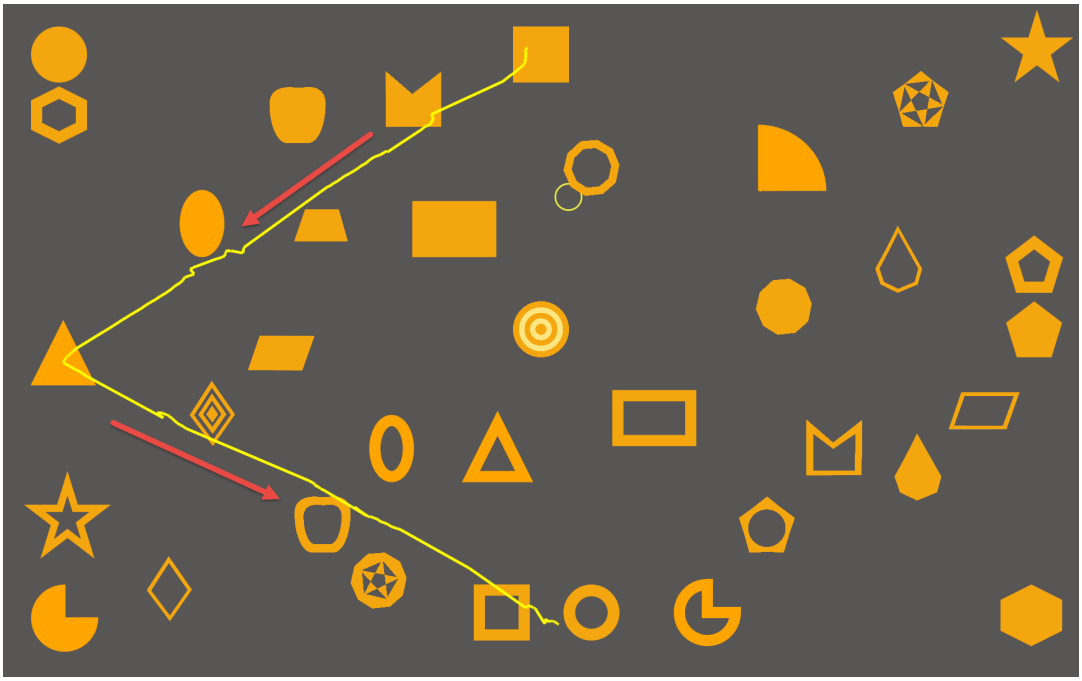

Figure 5.4: User's scan-path when following the traversed path of the **Square** shape (red - path of square, yellow - user's scan-path). The scan-path is shown here for representation, but the user does not see this.

5.3.5 Recognition System

We match the user's scan-path against a shape's traversed path through "Template Matching" algorithm, where we compute the root-mean-square distance of the candidate path (user's scan-path) from all the template paths (shapes' traversed paths). The template path of a shape that is at a least distance from the candidate path is chosen as the shape followed by the user. Our template matching algorithm is similar to $1 [215], but we perform only sampling, and calculate the average distance between the two paths.

5.3.5.1 Scan-Path Matching and Authentication

The template matching algorithm first samples the input scan-path to N = 64 points as depicted in Figure 5.8. We chose N=64, empirically derived considering the eye tracking frequency of 60Hz we used. To compute the average distance between a candidate path and a template path, as shown in Figure 5.9, we use equation 1, where P is a (X,Y) point on a path, C - candidate path, T -

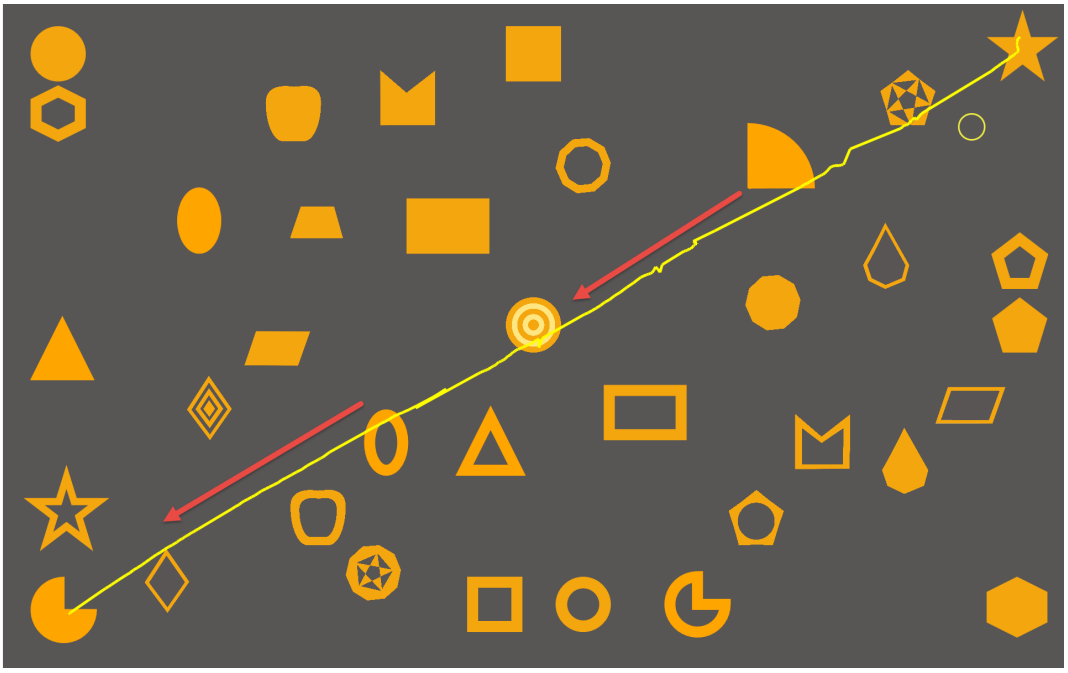

Figure 5.5: User's scan-path when following the path of the **Star** shape.

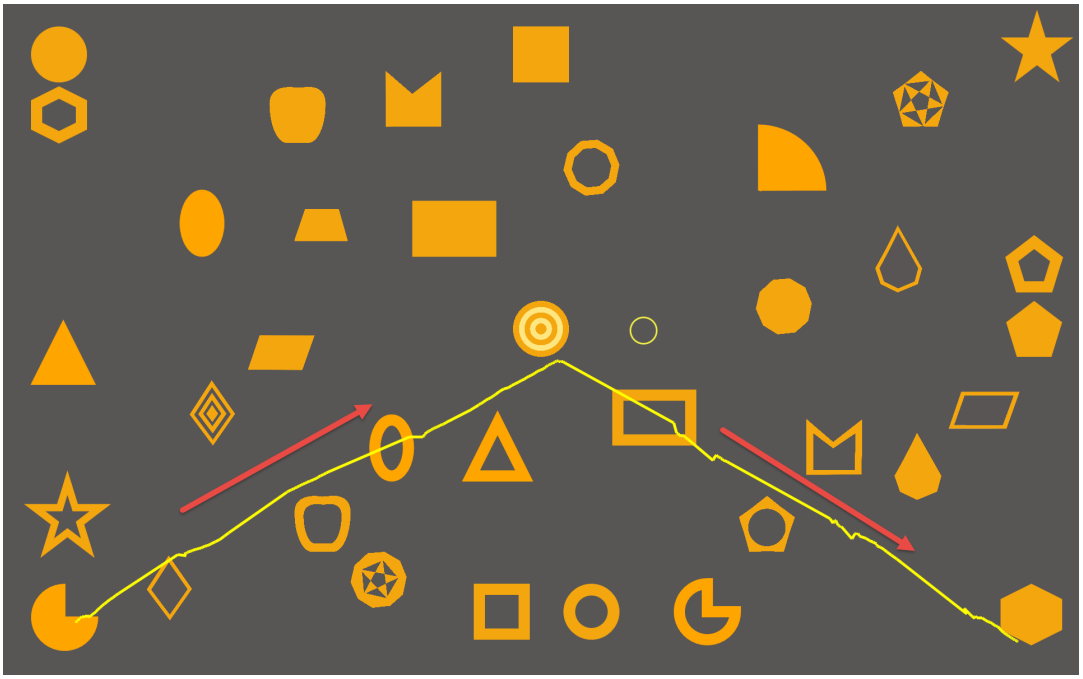

Figure 5.6: User's scan-path when following the path of the **Pie** shape.

Figure 5.7: Access granted or denied.

template path, and ΔDT - average distance to template.

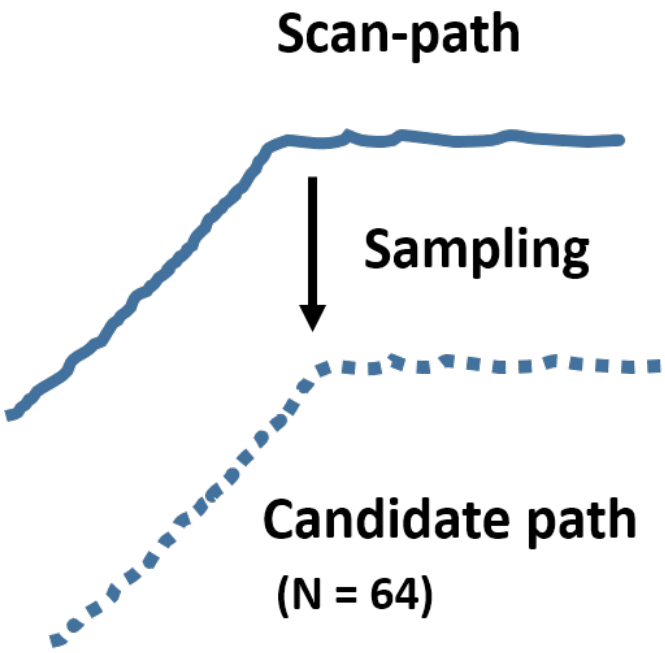

Figure 5.8: User's scan-path with ~300 points, scaled down to N = 64 points in the sampling stage. Sampling converts the scan-path to candidate path.

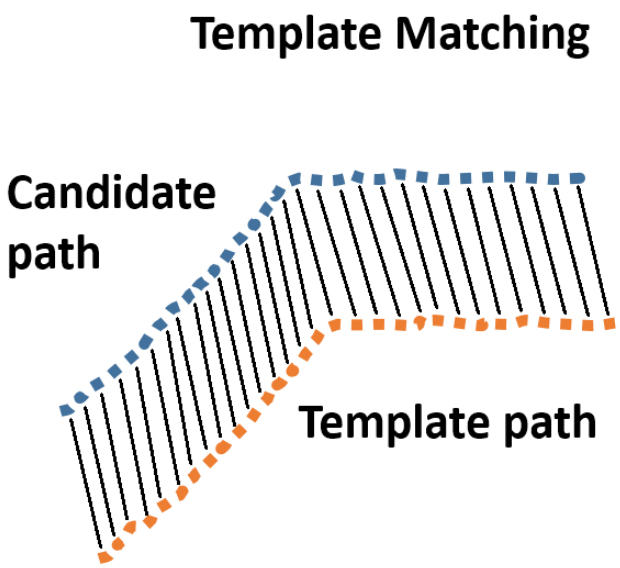

Figure 5.9: Template matching algorithm finding the Euclidean distance between each point on the candidate path (scan-path) to a corresponding point on the template path.

$$\Delta DT = \frac{\sum_{p=1}^{N} \sqrt{(C[p]_x - T[p]_x)^2 + (C[p]_y - T[p]_y)^2}}{N} \tag{5.1}$$

5.3.5.2 Template Construction

Our system was trained from traversed paths generated by seven users. First, each user generated paths for 12 shapes that are used as templates in the recognition phase, where the user again followed each of the shapes and the system recognizes the shape followed. For users who achieved more than 90% accuracy, their templates were retained. We repeatedly added and tested new paths until our final system achieved 100% accuracy from paths created by four of those users. Since eye movements involve fixations, saccades, and regressions [26, 216], we generate template paths against which the user's scan-path is matched, instead of using line paths of the shapes.

5.3.6 Experiment Design and Results

We tested the system in two phases.In the first phase, we evaluated system accuracy and robustness to calibration errors. In the second phase, we tested how strong our authentication system

is against video analysis attacks by comparing it against a gaze- and PIN-based password entry system.

5.3.6.1 PHASE 1: System Accuracy and Robustness

We recruited 15 participants (12 males and 3 females), some used vision correction devices like glasses and contact lens. All were either graduate or undergraduate students, with ages varying between 20 and 26 ($\mu_{age} = 22.53$). Before each study, the participant was briefed about the idea of gaze- and sketch-based authentication, given a small demo of the working system, and calibrated with the eye tracker on a 1900×1200 monitor.

5.3.6.2 Part 1: Scan-Path Recognition Accuracy

The goal of this study was to determine the recognition accuracy of the user's scan-path against the actual path of the shape. Hence, the user follows all 12 true shapes, one on each frame. After the completion of each frame, the system recognizes the shape followed by the user, and the shape's name is displayed through a pop-up message. In this phase, a small circle moves on the screen reflecting the user's gaze-point on the screen, and the user's scan-path is drawn as the gaze moves. This feedback (scan-path) was enabled to verify true positives, i.e., the path followed by the user for a given shape. However, no trial was repeated if the user didn't follow the true path resulting in recognition failure, as such errors may occur in real-world scenarios. Table 5.1 shows the confusion matrix for all the true shapes. We achieved a scan-path recognition accuracy of 99.44% at an F-measure of 0.99.

5.3.6.3 Part 2: Authentication Accuracy With True Calibration

In this part, the user was allowed to choose a password, by selecting three shapes from the password selection window. After selection, the user follows those shapes, one on each frame, but no feedback (scan-path) was shown. Providing no feedback simulates the real-world scenario, as feedback would enable shoulder-surfing attacks. To authenticate, the user should get all the three shapes correct. The user repeated this authentication procedure for three different passwords. We also recorded a video of the user's eye movements, while entering a password, to use in the com-

Table 5.1: Scan-Path Recognition - Confusion Matrix. Key: A - Circle, B - Open Hexagon, C - Triangle, D - Pie, E - Square, F- eye, G - Open Square, H - Ring, I - Star, J - Open pentagon, K - Pentagon, L - Hexagon

	A	B	C	D	E	F	G	H	I	J	K	L
A	1.0											
B		1.0										
C			1.0									
D				0.93	0.07							
E					1.0							
F						1.0						
G							1.0					
H								1.0				
I									1.0			
J										1.0		
K											1.0	
L												1.0

parative study. Lastly, to test the system's ability to invalidate wrong passwords, the experiment facilitator sets a different password (unknown to user), and the participant attempts to access the system by guessing the password; this was also repeated for three different passwords. This is similar to testing the system with true negatives. We achieved an authentication accuracy of 99%, and the confusion matrix is shown in Table 5.2.

Table 5.2: True Calibration: Confusion Matrix, Authentication Accuracy, and F-Measure

	True Password	False Password	Accuracy	F-Measure
True Password	97%	3%	99%	0.99
False Password		100%		

5.3.6.4 *Part 3: Authentication Accuracy with Disturbed Calibration*

To test robustness to calibration errors, the user was asked to get up and walk around for a few minutes. Upon return, the eye tracker was not re-calibrated, leaving the authentication system susceptible to calibation errors. Similar to part 2 of the study, the participant chooses three new passwords and enters them on three different trials. Again, the facilitator sets three new passwords, and the participant tries to access the system by guessing the passwords on three different trials, to test true negatives. We achieved an authentication accuracy of 96%, and the confusion matrix is shown in Table 5.3.

Table 5.3: Disturbed Calibration: Confusion Matrix, Authentication Accuracy, and F-Measure

	True Password	False Password	Accuracy	F-Measure
True Password	92%	8%	96%	0.96
False Password		100%		

5.3.6.5 *PHASE 2: Robustness Against Hacking*

Through a preliminary study, similar to previous studies [102, 78], we tested the susceptibility of our system to video analysis attacks in comparison to a gaze- and PIN-based password system. During phase 1, we recorded the videos of participants entering passwords (Figure 5.10) on both our system and a gaze- and PIN-based system (Figure 5.12) that used dwell-based selection. Four users, as shown in Figure 5.11, analyzed videos, chosen randomly, of the participants entering passwords. We found that gaze- and sketch-based authentication system was 40% less susceptible to video analysis attacks, and it took significantly more time–nearly 9 times longer–to guess the password on our system compared to gaze- and PIN-based authentication system. Users cracked 3/5 shape and 5/5 pin passwords in 4183 and 478 seconds respectively.

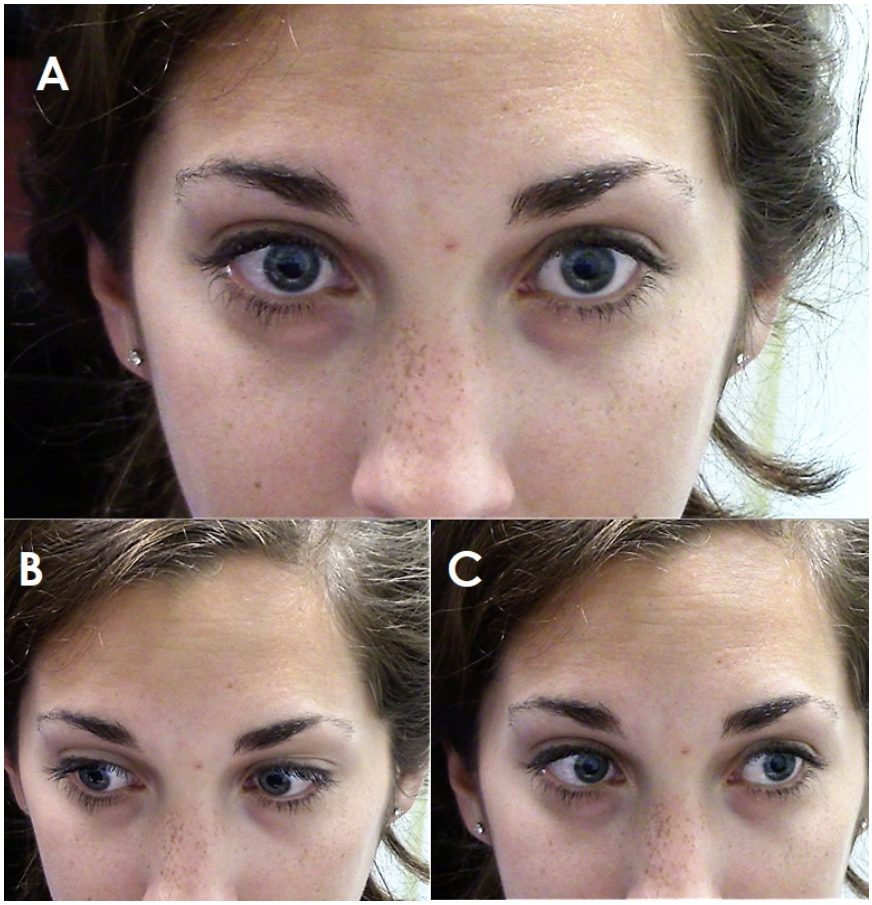

Figure 5.10: The range of the user's eye movements (gestures) when performing gaze authentication: A - looking at the center of the screen, B - looking at bottom left, C - looking at top right.

5.3.6.6 Qualitative Evaluation

Following are some of the feedback shared by the participants following completion of the study. **Positives:**

- **P02**: The nature of the actual password entry is really good.. I can't imagine a better way.
- **P05**: I liked that it's simple using shapes and that it's very secure that no one can really track your eye movements.. It's very innovative.
- **P12**: I like that you can use this for authentication and other people can't really tell that what you selected as your password, because they can't really follow your eyes.

Figure 5.11: Video Analysis Attack: A user is trying to guess the gaze password with the help of a video and authentication interface.

- **P09**: This is definitely something new.. It addresses the problem statement pretty nicely.
- **P07**: It was pretty impressive, it was able to distinguish between all those shapes when they were colliding with each other.. I would use it for everyday jobs.

Suggestions:

- **P09** : Someone sneezing could be an issue.. they have to do the password all over again.
- **P05**: The problem would be if people don't have concentrated eyes.
- **P15**: Adjust the eye tracker for different postures.

5.3.7 Discussion

In testing our hypotheses from our user studies, we first correctly hypothesized high accuracy for scan-path matching and the authentication system with true calibration. Also, the accuracy

Figure 5.12: Gaze- and PIN-based Authentication System

remained high even when the calibration was disturbed. We attribute high accuracy to relaxed precision on gaze input and unique paths for each shape traversal. However, we anticipate that multiple shapes with similar paths would reduce accuracy. Next, since the user follows a single shape in each frame, we found that the participants had no difficulty in entering a password with repeated shapes. Finally, results from video analysis attacks showed the advantage of fake shapes: although an attacker can guess the direction of a shape's movement from the user's eyes, they cannot pick the right shape from numerous options before the system locks out from failed attempts. From the interviews (side-table), we found that the users consider this solution innovative, secure, and simple. However, some expressed that sneezing, lack of attention during password entry, and so on would lead to incorrect gaze input.

While we expected 100% accuracy, we encountered two sources of scan-path distortion that affected accuracy. First, although our system authenticates with five-second shape movements compared to other gaze authentication systems that take from 7.5 seconds [104] to 54.0 seconds [103], users may blink during the shape's five-second movement and suggested reducing movement to 3 seconds. Hence, we hypothesize that reducing the overall authentication time to less than 10 sec-

onds avoids authentication failure due to erroneous gaze input. Second, the use of vision correction devices lead to imprecise gaze input [217, 218].

5.4 Dynamic Transitions Authentication

The fixed transitions authentication (Section 5.3) we discussed though achieves a higher accuracy than all the gaze-assisted authentication systems, the system is limited by various shortcomings. First, since the transition of the password shapes' are fixed for each shape, nearly 60% of the single video iterative attacks were still successful. Though this was an iterative attack and not bounded by time constraints, the fact that 60% of the passwords were hacked is unacceptable for an authentication system. Second, the system take nearly 15 seconds for three-shape and 20 seconds for four-shape passwords, and this delay is unacceptable if authentication is used frequently, e.g., unlocking a computer. Third, single all the shapes had the same color, it was hard to distinguish between the shapes when they all overlap during the transitions. Fourth, employing fake shapes does not always prevent but instead delays brute force attacks. If any enhancement is considered, the user should be able to select her own true and fake shapes. Lastly, the solution was screen resolution dependent, hence, may not work with the same accuracy across screens of different dimensions. To address these limitations with the fixed transitions authentication method, we developed a new strategy - dynamic transitions authentication.

Unlike the fixed transitions method, in dynamic transitions authentication the password elements (colors) move along random paths during an animation. Due to the dynamic nature of the interface, we show that our system is not susceptible to single video iterative attack, and has a low success rate with dual video iterative attack. We present two gaze gesture-based authentication strategies that rely on dynamic transitions. The core idea is a user authenticates by following uniquely colored circles that move along random paths on the screen. Through multiple evaluations, we discuss how the authentication accuracy various with respect to transition speed of circles, screen dimensions, and the number of moving and static circles. Furthermore, we evaluate the accuracy and resiliency of our authentication method against two threat models by comparing it against a gaze- and PIN-based authentication system. Overall, we found that of all the proposed

interfaces, the one with five static and five moving circles with a transition speed of two seconds was the most effective authentication method with an accuracy of 97.5%.

5.4.1 Introduction

We propose two authentication interfaces shown in Figure 5.13 and Figure 5.14 that use dynamic gaze gestures to authenticate users. The central idea of our authentication system is, the interface comprises of 10 uniquely colored circles which move along random paths during an animation of *N* seconds. An animation is a time interval where all the circles move from their source to destination locations. Analogous to a four digit PIN, a user selects a set of four colors (out of 10) as their password. To authenticate, the user follows the path of the colored circle during an animation, and the animation is repeated four times so that during each animation, the user follows the circle colored with his password color. For example, if the user's password is "red-blue-yellow-green" the user follows the red colored circle on the first animation, blue on the second, and so on. For a successful authentication, the scan-path of user's gaze should match with the path of the colored circle, in each animation, for all the four animations. In the above example, on animation 1, user's scan-path should match with the path of "red" circle, and the same is true for the remaining three animations. If the user fails to follow the correct color even in one animation, then the authentication fails. The two authentication interfaces we have developed are "dynamic interface" shown in Figure 5.13 and "static-dynamic interface" show in Figure 5.14. The main difference between the two interfaces is that in the dynamic interface all 10 circles move along random paths during an animation. However, on the static-dynamic interface, only five circles move and five remain static at fixed positions on the interface. While we initially created the dynamic interface, we later developed the static-dynamic interface because of various reasons that will be discussed in further sections.

We evaluated our solutions through a two phase user study. Both dynamic and static-dynamic interfaces were tested for their accuracy under two animation speeds 3 and 2 seconds. Furthermore, since static-dynamic interface was found to be a more practical solution, we tested for its accuracy under disturbed calibration. Lastly, we tested the static-dynamic interface interface under two

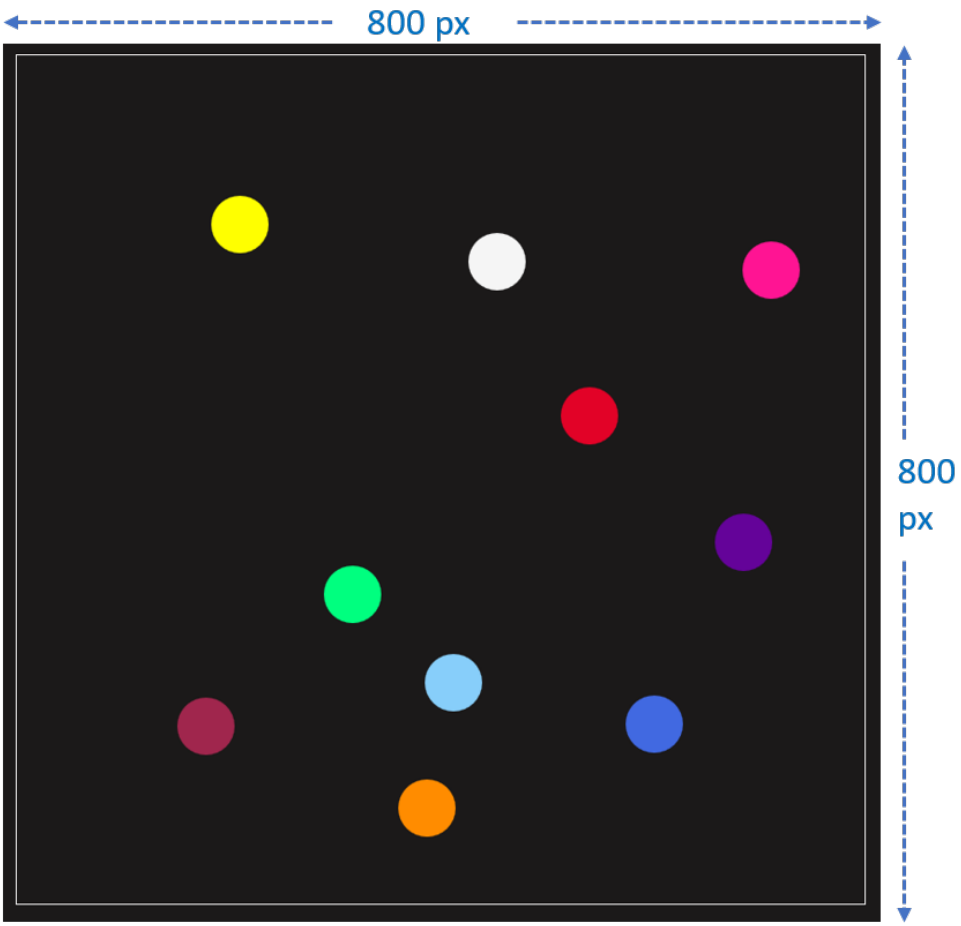

Figure 5.13: Dynamic authentication interface with 10 uniquely colored circles placed at random positions.

extreme conditions: 1) an animation speed of 1 second, and 2) a screen dimension of 400×400 px. Our results show that the static-dynamic interface with an animation speed of 2 seconds on an interface of 800×800 px is the most practical solution of all the interfaces and variations we have evaluated.

5.4.2 Design Motivation

While conceptualizing various design options for a dynamic transitions authentication system, we considered three key requirements. First, the interface should use graphical password as such methods have shown to help the user in remembering the password better than PIN based systems.

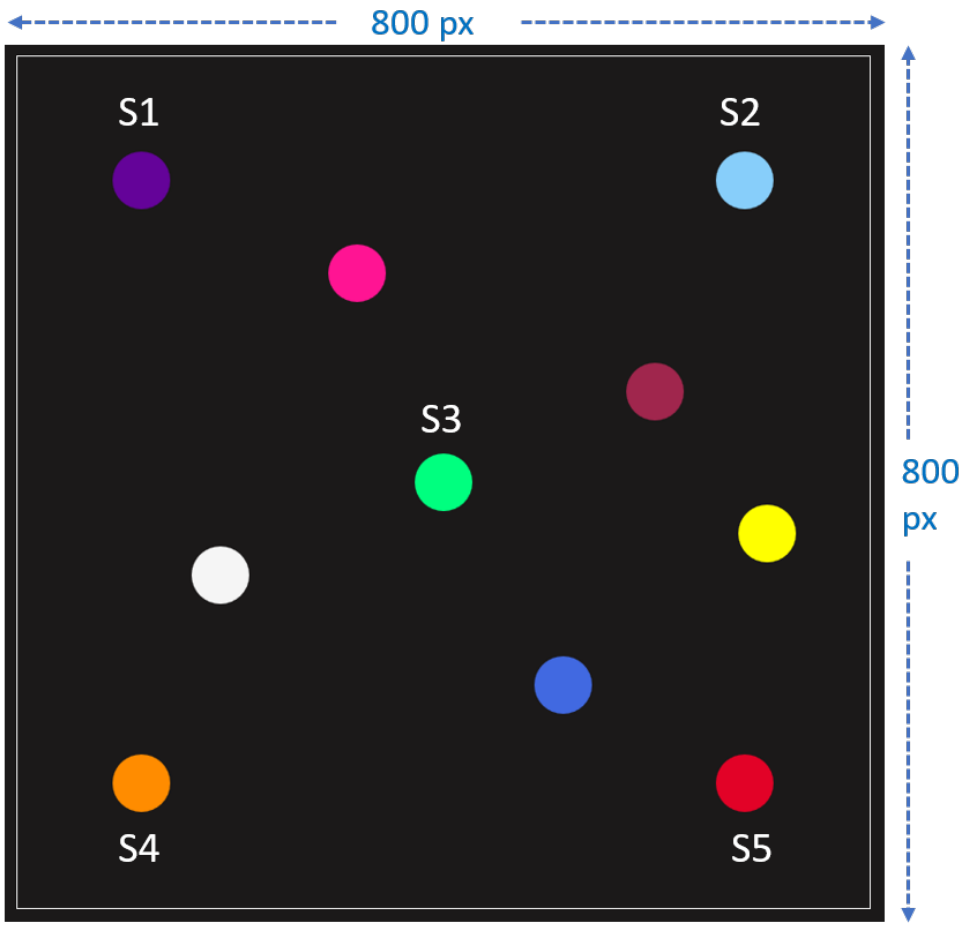

Figure 5.14: Static-dynamic authentication interface comprising of 5 static (S1, S2, S3, S4, S5) and 5 dynamic circles.

Second, the interface should be dynamic, unlike static interfaces, which ensures that it is extremely difficult to hack the system either by direct shoulder surfing or iterative video analysis attacks. Third, for any authentication system to be deployable at ATMs, kiosks, etc., the interface should work on smaller screen dimensions.

To achieve these goals, we improved on the gaze-based authentication system presented by Rajanna et al. [30] that used an interface with multiple geometric shapes (square, triangle, etc.). The shapes had fixed start-end positions and they would traverse along a fixed path during an animation. From [30], it was suggestive that though shapes had different geometries, but having

the same color for all the shapes was challenging for a user to keep track of the target shape when multiple shapes crossover during an animation. Moreover, all the shapes had a fixed path which makes the system highly susceptible to iterative video analysis attacks. In our design, to reduce visual cluttering, we focused on maintaining the uniformity of the interface by using circular shapes. Additionally, to distinguish each circle, we assigned a unique color for every circle. To keep the system analogous to a 10 digit numeric keypad based PIN entry system, we have included 10 circles.

We also thought about using numbered circles (0-9) instead of the colored circles similar to the design used in [79], however, using digits brings about the problem of common PIN passwords being hacked by intelligent attacks [3]. Hence, using circles with unique colors is an appropriate design choice, and we hypothesize that people are able to remember their password by associating the password colors with their favorite colors, colors of the objects they frequently use (car, cloth, etc). However, one limitation of this design is users with colorblindness will have limited set of colors to choose from as they can not distingush few colors. Since selection of right colors is important, we choose warm colors like yellow, pink, orange, etc., as they are more simulating and active [219, 220]. In addition, we ensured high-contrast colors as they attract more attention and better visibility which influence memory retention [221]. Lastly, we surveyed screen dimensions of ATMs by various vendors[4] [5].There is no single standard size of the ATM screen, but commonly used dimensions include 8", 10.1", 12.1", 15". We chose a median size of 11.5" which approximately translates to a dimension of 800×800 px on a screen with 98.44 PPI (screen size 1900×1200, 23") used in our experiments.

5.4.3 System Architecture

The gaze gesture-based authentication system using dynamic transitions has the same components as the fixed transitions authentication method: 1) Gaze Tracking Module, and 2) Authentication Engine. A working model of the system is depicted in Figure 5.15.

[3] www.datagenetics.com/blog/september32012/
[4] www.atmmarketplace.com
[5] www.ATMequipment.com

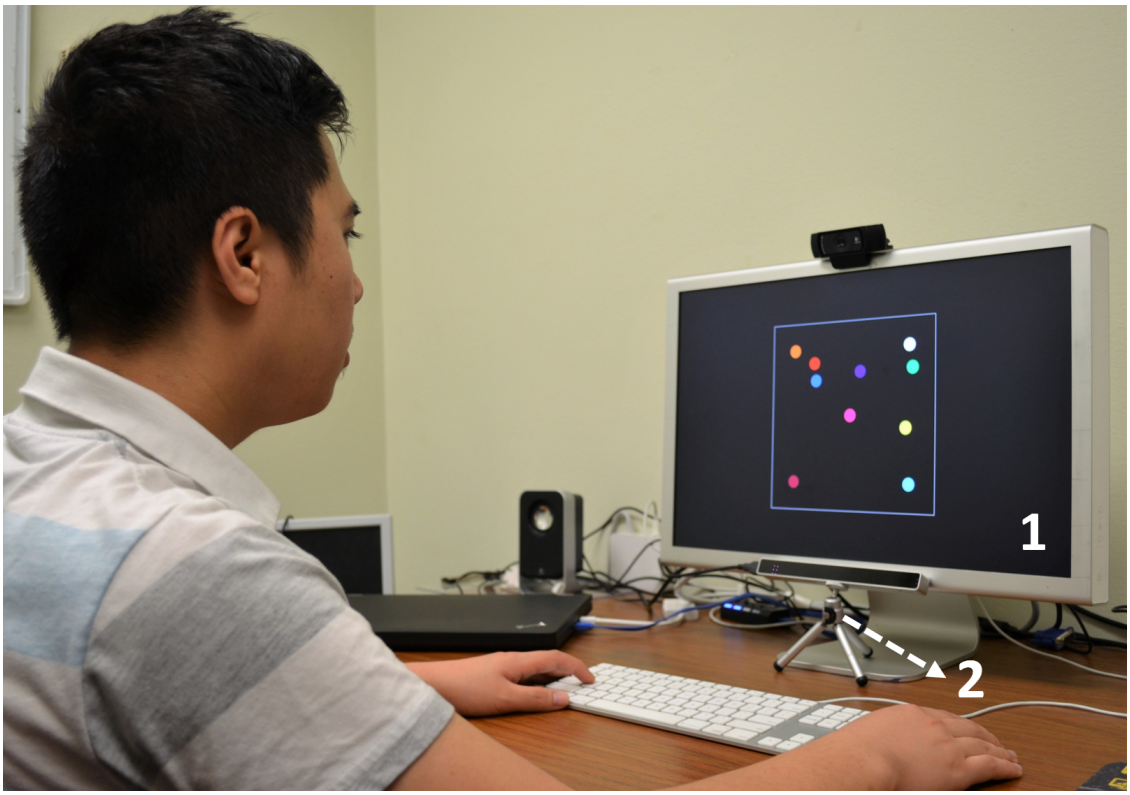

Figure 5.15: A user authenticating by following the paths of four uniquely colored circles chosen as the password (1 - Authentication interface, 2 - Eye tracker).

5.4.4 Authentication Procedure

The authentication procedure comprises of two processes: 1) one-time password selection, and 2) password entry.

5.4.4.1 One-time Password Selection

The user selects four colors as their password from the password selection window shown in Figure 5.16. Each color chosen is associated with an animation, i.e., if "red" color was chosen for animation 1, the user should follow the "red" color during animation 1 of the password entry phase . Password selection is a one-time procedure, and is only repeated if the user wants to change the password.

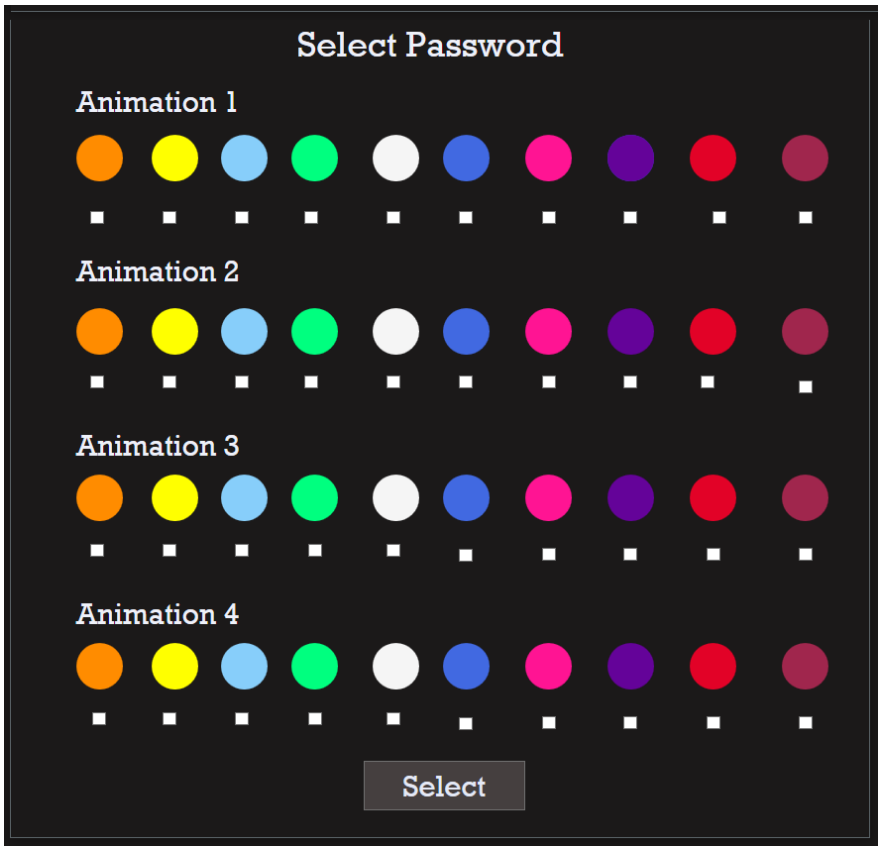

Figure 5.16: Password Selection Interface: a user selects a password by selecting one color for each animation, hence a total of four colors is chosen as the password.

5.4.4.2 Password Entry

To authenticate, the user follows four password colors in sequence, one during each animation, in four consecutive animations. The user controls the authentication interface through a set of hot-keys: 'A' to initiate movement of circles and record gaze data, 'Z' to recover from user mistakes (blink, sneeze, losing the path) and discard recorded gaze data, and 'M' to submit the password after following four circles.

5.4.5 Authentication Interface Dynamics

We discuss the dynamics and the algorithms used in the two authentication interfaces: 1) dynamic interface, and 2) static-dynamic interface. Both interfaces have 10 colored circles and have

a dimension of 800×800 px, but they differ in the number of moving circles in an animation. Irrespective of the interface, the two mechanisms that are common to both the interfaces are: 1) random point generation, 2) generation of animation path and template for each colored circle.

5.4.5.1 Random Point Generation Algorithm

To generate n number of random points that are uniformly distributed inside a circle with radius R_c, we employ the method proposed by Leon-Garcia et al. [222]. The joint probability distribution function (PDF) of the random points inside the circle, i.e., the joint distribution of random variable **X** and random variable **Y** representing the **x** and **y** coordinates of a random point is given by:

$$f_{X,Y}(x,y) = \begin{cases} \frac{1}{A} = \frac{1}{\pi R_c^2} & x^2 + y^2 \leq R_c^2 \\ 0 & \text{otherwise} \end{cases}$$

After transforming (x,y) into the polar coordinates and taking the Jacobian of the transformation, the joint PDF of random variables **R** and Θ is calculated using the joint PDF of **X** and **Y** as:

$$f_{R,\Theta}(r,\theta) = f_{X,Y}\left(h_1(r,\theta), h_2(r,\theta)\right) =$$

$$\begin{cases} \frac{r}{\pi R_c^2} & 0 \leq \theta \leq 2\pi, 0 \leq r \leq R_c \\ 0 & \text{otherwise} \end{cases} \tag{5.2}$$

The non-zero parts are rewritten as:

$$f_{R,\Theta}(r,\theta) = \frac{1}{2\pi} \times \frac{2r}{R_c^2} = f_{\Theta}(\theta) f_R(r)$$

where

$$f_{\Theta}(\theta) = \frac{1}{2\pi} \qquad f_R(r) = \frac{2r}{R_c^2}.$$

Now, Θ is uniformly distributed between 0 and 2π. The random variable $\mathbf{R}$ is generated by first calculating its cumulative distribution function as:

$$F_R(r) = \int_0^r \frac{2\alpha}{R_c^2} d\alpha = \frac{r^2}{R_c^2},$$

and then applying a uniformly distributed random variable $\boldsymbol{U}$ over the interval [0,1] we get

$$U = \frac{R^2}{R_c^2} \implies R = R_c \sqrt{U}.$$

Using the above distribution we generate n random points.

5.4.5.2 Initial Points for Circles

The initial position of the circles is random but uniformly distributed within the radius of 100 pixels from the center of the interface (with a dimension of 800×800 px). To generate the points, we use the Random Point Generation Algorithm discussed above with radius $R_c = 100$ pixels for each of the colored circle. We are using a small radius to make the initial positions of the circles closer which make video analysis attacks difficult.

5.4.5.3 Generating Animation Paths and Templates

Once the initial points are generated for the 10 circles, we generate random path for each circle. Each random path is a set of three points that the shape traverses from its initial point. To generate the three points for a random path, we use the same method of generating uniformly distributed random points discussed above by using a radius of $R_c = 400$ px. Additionally, we constrain that the three points to be beyond one-third of the distance from the center as shown in the Figure 5.17.

Based on the interface dimension, duration of animation, and the sampling frequency of the eye

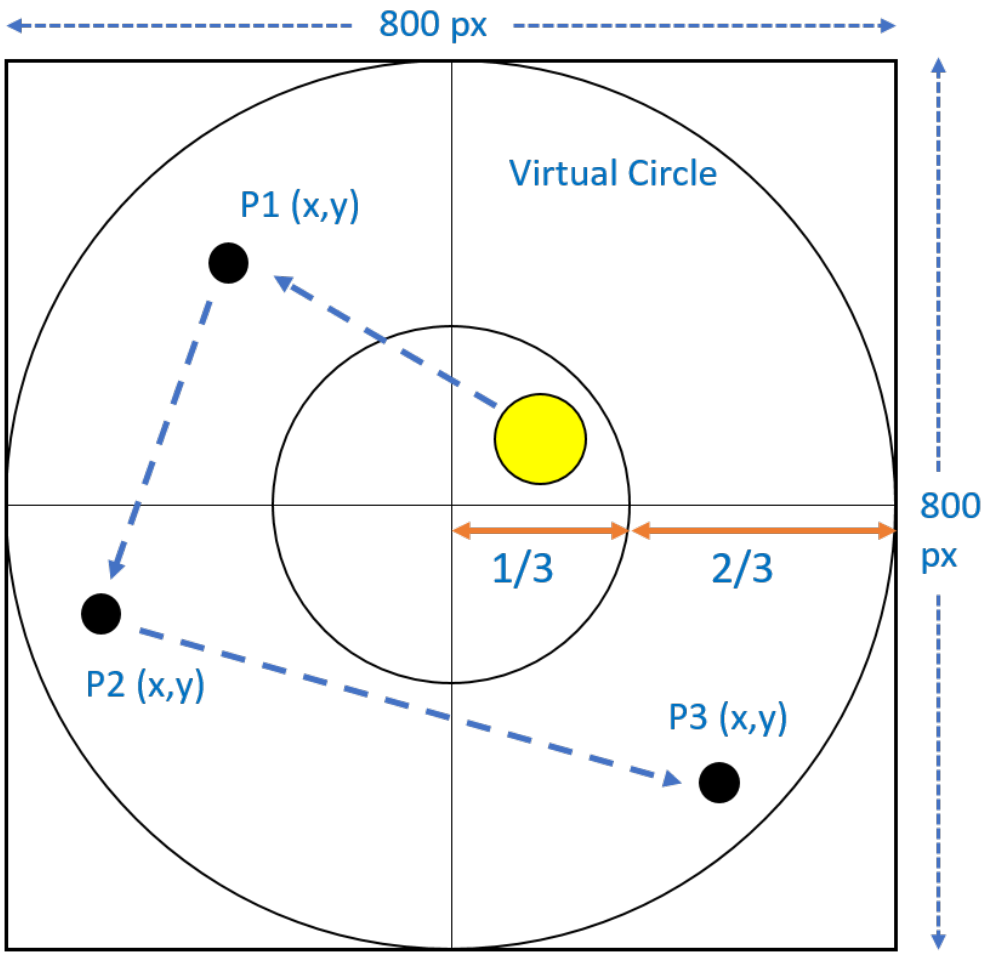

Figure 5.17: Distribution of Random Points: distribution of random points (P1, P2, P3) for the path of a circle (yellow). The random points are beyond 1/3 distance from the center along the radius of the virtual circular boundary.

tracker, we have established empirically that the template path should be made up of 300 points that are equally distributed along the its path. Since the animation path is made up of 4 points (1 starting + 3 random points) and 3 line segments joining these four points, the lengths of the line segments are uneven. Hence, we compute what fraction of 300 points are distributed along each line segment proportional to its length. In order to keep the path between points as a straight line, we used piecewise linear interpolation [223] to generate points along a line segment. Given two points **A** and **B**, this algorithm will put points on the line segment between **A** and **B** such that the

first point is 1 unit from **A**, the second point is 2 units from **A**, and so forth until the last point, which is a whole number of units from **A** and one unit or less from **B**. Suppose the Cartesian (x,y) coordinates of the points are $A = (x_A, y_A)$ and $B = (x_B, y_B)$. Let **d** be the distance between these two points, then by Pythagorean Theorem,

$$d = \sqrt{(x_A - x_B)^2 + (y_A - y_B)^2}.$$

The next point is at $\frac{2}{d}$ of the distance from **A** to **B**, the next at $\frac{3}{d}$ of the distance, and so forth. In general the n^{th} point that we place along the segment from **A** to **B** should be at coordinates (x_n, y_n), which represents the next point along the slope of the line joining the two points. where

$$x_n = x_A + \frac{n}{d}(x_B - x_A),$$
$$y_n = y_A + \frac{n}{d}(y_B - y_A).$$

We perform this for each integer n such that $1 \leq n < d$. We use the length **d** as normalizing factor to get a total of 300 points from three line segments generated by connecting the three random points and the initial point of the shape. All the points obtained from the above computation are later stored as the template for matching against a scan-path.

5.4.5.4 Scan-path Matching Algorithm

We match the user's scan-path against a circle's traversed path through "Template Matching" algorithm, where we compute the root-mean-square distance of the candidate path (user's scan-path) from all the template paths (circles' traversed paths). The template path of a circle that is at the least root-mean-square distance from the candidate path is chosen as the circle (color) followed by the user. Our template matching algorithm is similar to $1 [215], but we perform only sampling, and calculate the average distance between the two paths.

Sampling: We down-sample both the candidate path and the template paths to N=64 points,

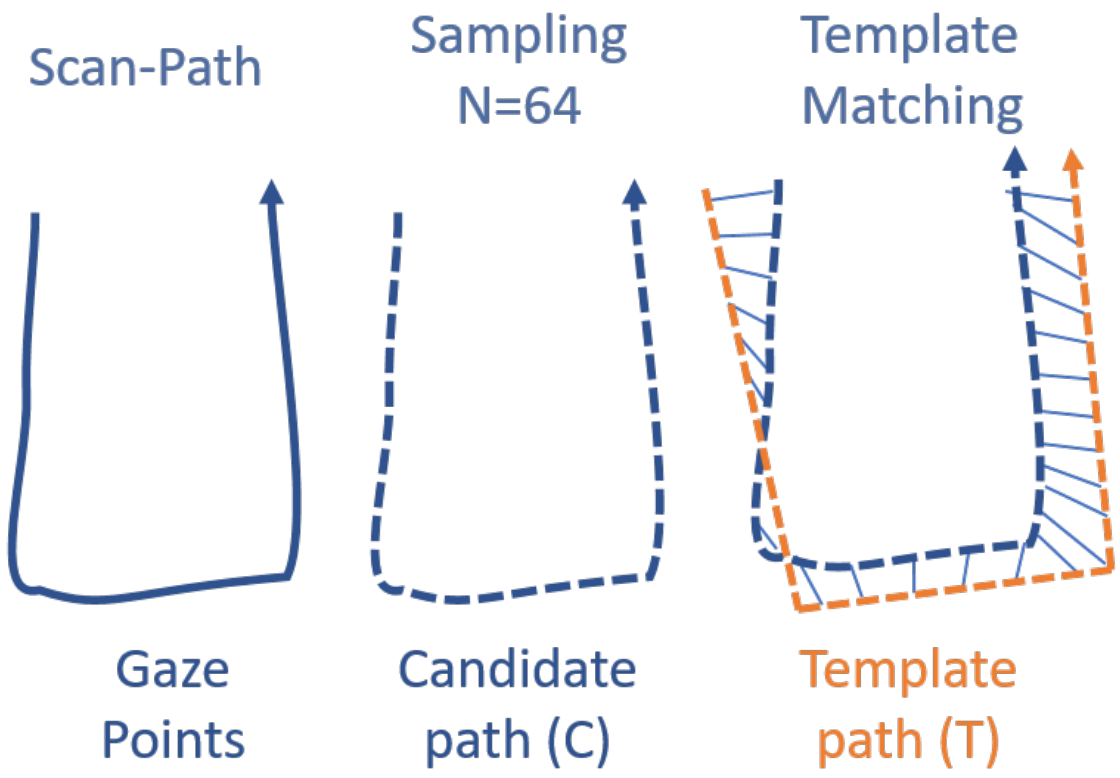

Figure 5.18: Sampling and Template Matching: a user's scan-path is sampled to N=64 points (candidate path), and matched against paths of all the random paths (template paths)

because of two reasons. First, sampling reduces the noise in the scan-path due to inherent jittery eye movements and approximates the path to a good extent. Second, down-sampling reduces the computation during the matching phase. Figure 5.18 shows the sampling of a scan-path.

Matching: To compute the average distance between a candidate path and a template path, as shown in Figure 5.18, we use equation 2,

$$\Delta DT = \frac{\sum_{p=1}^{N} \sqrt{(C[p]_x - T[p]_x)^2 + (C[p]_y - T[p]_y)^2}}{N} \tag{5.3}$$

where p is a point on path, C - candidate path, T - template path, and ΔDT - average distance to template.

5.4.6 Authentication Interfaces

5.4.6.1 Dynamic Authentication Interface

The dynamic authentication interface comprises of 10 uniquely colored circles that are distributed randomly on the interface. Most importantly, the circles traverse along random paths during an animation, and as they reach their final locations at the end of the animation, they interchange their positions in a random fashion.

5.4.6.2 Static-Dynamic Authentication Interface

We further modified the dynamic interface to develop static-dynamic interface for two reasons. Firstly, though the dynamic interface with 10 moving circles introduces enough randomness to prevent both shoulder surfing and video analysis attacks, a few users were overwhelmed by the visual cluttering of the interface as we found during the pilot studies. Though the user can start following a circle, once all the circles come closer or overlap during an animation, the user might loose the sight of the circle and this leads to recognition error. Secondly, since all the 10 circles move within a space of 800×800 px, two random paths might be similar which again leads to recognition failures affecting the accuracy.

Considering these factors, we designed static-dynamic authentication interface in such a way that only 5 out of the 10 circles move during the animation and the remaining 5 are static. However, a dynamic (moving) circle on the current animation can continue to be a dynamic circle or become a static circle on the subsequent animation. Similarly, a static circle on the current animation can continue to be a static circle or become a dynamic circle on the subsequent animation as the circles randomly interchange their positions at the end of the animation. Hence during the authentication, the user mostly ends up following a few dynamic circles and focusing at a few static circles. However, until the start of the animation, the user will not have any information if their password color (e.g., green) is going to be static or dynamic. Figure 5.14 shows the placement of static and dynamic circles. We fixed the locations for static circles along the virtual circular boundary at angles 45°, 135°, 225°, 315°, and at the center of the rectangle. The (x,y) coordinates

are calculated by $X = radius \times Cos(angle)$ and $Y = radius \times Sin(angle)$. We do not randomly select the positions of static circles since random placement may bring two circles closer and this would result in recognition failure as the static circles are selected based on visual fixation. We hypothesize that the static-dynamic authentication interface outperforms the dynamic interface mainly from the perspective from user friendliness and accuracy.

5.4.6.3 Authentication Procedure

Similar to authenticating on the dynamic interface, the user selects 4 colors as her password. To authenticate, during an animation, if the color chosen happens to be a dynamic circle, the user will follow the path of that circle. While on the contrary, if the color is a static circle, the user will constantly look at (fixate) the circle during the entire animation. If the user correctly follows or fixates on the colors in the consecutive animations in the order of the sequence of the password, the user is authenticated otherwise, the authentication fails.

5.4.6.4 Scan-path and Fixation Matching

Unlike dynamic interface, we need to first distinguish if the user followed a circle or fixated on a circle in the static-dynamic interface. To accomplish this, at the end of every animation, we compute the length of the user's scan-path. If the length of the scan-path is above the dispersion threshold ($length > d_{th} = 300$ pixels), we use the scan-path matching algorithm. However, if the length of the scan-path is below the dispersion threshold ($length < d_{th} = 300$ pixels), we use the centroid method to recognize the target circle that was focused on by the user. In the centroid method, we compute the centroid of all the gaze points recorded during the animation. To recognize the user-targeted circle, we calculate the Euclidean distance of all the static circles from the centroid and identify the circle which is nearest from the centroid as the recognized user-targeted circle.

5.4.7 Evaluation and Results

We recruited 20 participants (16 male, 4 female), all were either graduate or undergraduate students, with ages ranging from 18 to 31 ($\mu_{age} = 23.15$). Before the study, each participant was

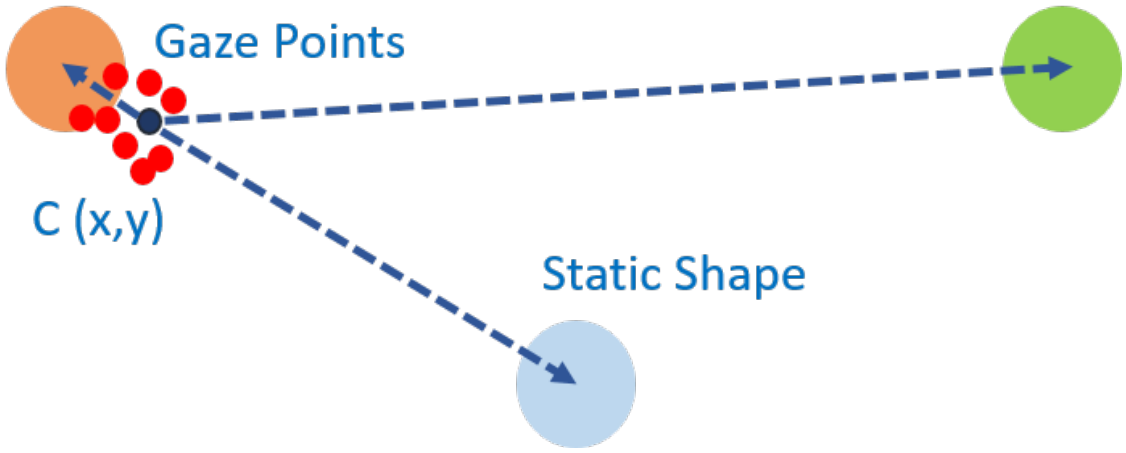

Figure 5.19: To recognize the static circle focused on by the user, centroid of gaze points is found and distances to all the static shapes are calculated to find the least distance.

briefed about the motivation behind gaze-based authentication system, and calibrated with an eye tracker on a 1900×1200 monitor. 1 participant was colorblind who could not distinguish between red and green colors, and hence did not choose these two colors during authentication. For each different interface we provided a training session for a maximum of 2 minutes.

5.4.7.1 Dynamic Authentication Interface

We evaluated the dynamic interface under two conditions by varying the animation time. To evaluate the system accuracy in recognizing the true password and authenticating the user, each user enters a password that was selected randomly by the experiment facilitator. To authenticate using a given password say "pink-orange-yellow-red", the user follows four colors in sequence, and the system authenticates the user if the four scan-paths of the user matches with the traversed paths of four colors selected as the password. This procedure was repeated for a total of two true passwords. Next, to test the system's ability to reject the false password, the experiment facilitator sets a different password (unknown to user), and the participant attempts to access the system by guessing the password. This is similar to testing the system with true negatives, and this procedure was repeated for a total of two passwords. We tested the above procedures, i.e., entering two true and two false passwords, under two experimental conditions. First the animation speed was

set to 3 seconds, and next the entire procedure was repeated by setting the animation speed to 2 seconds. We test two animation speeds since the goal is to achieve lower authentication time while supporting high accuracy.

5.4.7.2 System Accuracy

The system accuracy, F-measure, and confusion matrix for 3 and 2 second animations on the dynamic interface are listed in Table 5.4 and Table 5.5 respectively.

Animation time: 3 seconds

Table 5.4: Dynamic interface - 3 Seconds Animation: Confusion Matrix, Authentication Accuracy, and F-Measure

	True Password	False Password	Accuracy	F-Measure
True Password	85%	15%	92.5%	0.92
False Password		100%		

Animation time: 2 seconds

Table 5.5: Dynamic interface - 2 Seconds Animation: Confusion Matrix, Authentication Accuracy, and F-Measure

	True Password	False Password	Accuracy	F-Measure
True Password	82.5%	17.5%	91.25%	0.90
False Password		100%		

5.4.7.3 Recognition Error

We compute how many of the true passwords entered by the user are correctly recognized by computing the Levenshtein distance [224]. Levenshtein distance is a measure of the number of

entries in the password entered by the user that are correct and in the right place when compared to the actual password. For example, if the actual password is “pink-green-yellow-white" and the user enters “pink-green-yellow-white" the Levenshtein distance 0, i.e., the password has no errors. However, a password entered as “pink-red-yellow-white" has a Levenshtein distance of 1 since the user entered “red" in the place of “green." Similarly, a password of “maroon-blue-yellow-white" has a Levenshtein distance of 2, and so on. Table 5.6 shows the recognition errors for all the true passwords entered under two experiment conditions (3 and 2 second animations) on the dynamic interface.

Table 5.6: Dynamic Interface: recognition error based on the Levenshtein distance

Levenshtein distance	3 Seconds	2 Seconds
0 Error	**85%** (34/40)	**82.5** (33/40)
1 Error	**15%** (6/40)	**17.5%** (7/40)

5.4.7.4 Individual Path Recognition

To recognize the true password or to reject the false password, the system should be able to recognize the password entered by the user. Hence, we saved information of every single path entered by the user and the recognition result for both the true and false password entries. Table 5.7 shows the path recognition accuracy for all the entries on the dynamic interface.

5.4.7.5 Discussion

On the dynamic interface, sometimes, even when the participants followed the correct circle, the system wrongly recognized the circle followed. As discussed earlier, this kind of error is due to two circles having a similar path, and this is a limitation with the interface with all moving circles. Furthermore, the participants did not have difficulty in following the shapes when animation speed

Table 5.7: Dynamic Interface: path recognition accuracy

	3 Seconds	2 Seconds
Passwords Entered	80	80
Total Paths	320	320
Recognized paths	306	301
Accuracy	**95.63%**	**94.06%**

was set to 3 seconds. However, when the animation speed was reduced to 2 seconds, few participants felt it was challenging to keep track of the shapes. Losing the path or partially following the path of a circle during an animation leads to recognition failures due to incomplete gestures.

5.4.7.6 Static-Dynamic Authentication Interface

We followed the same evaluation procedure as the dynamic interface. Each participant entered two true and two false passwords under two experiment conditions: 1) 3 seconds animation, and 2) 2 seconds animation.

5.4.7.7 System Accuracy

The system accuracy, F-measure, and confusion matrix for 3 and 2 second animations on the static-dynamic interface are listed in Table 5.8 and Table 5.9 respectively.

Animation time: 3 seconds

Animation time: 2 seconds

5.4.7.8 Recognition Error

Table 5.10 show the recognition errors for all the true passwords entered under two experiment conditions (3 and 2 second animations) on the static-dynamic interface.

Table 5.8: Static-dynamic Interface - 3 Seconds Animation: Confusion Matrix, Authentication Accuracy, and F-Measure

	True Password	False Password	Accuracy	F-Measure
True Password	97.5%	2.5%	98.75%	0.99
False Password		100%		

Table 5.9: Static-dynamic Interface - 2 Seconds Animation: Confusion Matrix, Authentication Accuracy, and F-Measure

	True Password	False Password	Accuracy	F-Measure
True Password	95%	5%	97.5%	0.97
False Password		100%		

Table 5.10: Static-dynamic Interface: recognition error based on the Levenshtein distance

	3 Seconds	2 Seconds
0 Error	**97.5%** (39/40)	**95%** (38/40)
1 Error	**2.5%** (1/40)	**5%** (2/40)

5.4.7.9 Individual Path Recognition

Table 5.11 shows the path recognition accuracy for all the entries on the static-dynamic interface.

5.4.7.10 Discussion

As we found during the system development and pilot studies, the participants expressed that it was easy to follow the moving shapes even when the animation speed was 2 seconds. Furthermore, users do not lose the path of a circle in transition because of less or no overlapping of circles, since

Table 5.11: Static-dynamic Interface: path recognition accuracy

	3 Seconds	2 Seconds
Passwords Entered	80	80
Total Paths	320	320
Recognized paths	316	314
Accuracy	**98.75%**	**98.13%**

only 5 circles move during an animation. Few participants also felt that the combination of static and moving circles reduces the attention required compared to the dynamic interface. Also, to compare the accuracy, static-dynamic interface outperforms the dynamic interface both at 3 (dynamic 92.5%, static-dynamic 98.75%) and 2 (dynamic 91.25%, static-dynamic 97.5%) seconds animations. Furthermore, for static-dynamic interface “individual path recognition" accuracy almost remained the same both at 3 (98.75%) and 2 (98.13%) seconds as shown in Table 5.11. As we hypothesized, all these factors strongly suggest that the static-dynamic interface with 2 seconds animation is the most practical solution. Hence, we further tested the static-dynamic interface under two extreme conditions and with disturbed calibration as we discuss in the next section.

5.4.7.11 Static-Dynamic Extreme Interface

We tested the static-dynamic interface under two extreme conditions. First, we kept the interface dimension same as before, i.e., 800×800 px, but we set the animation time to 1 second. Second, we kept the animation speed as before, i.e., 2 seconds, but reduced the interface dimension to 400×400 px as shown in the Figure 5.20. A 400×400 px dimension on screen with 98.44 PPI (screen size 1900×1200 px, 23") translates to 5.75" screen. Under these two conditions, the user entered only two true passwords but no false passwords were entered since we were testing the limits of our authentication method.

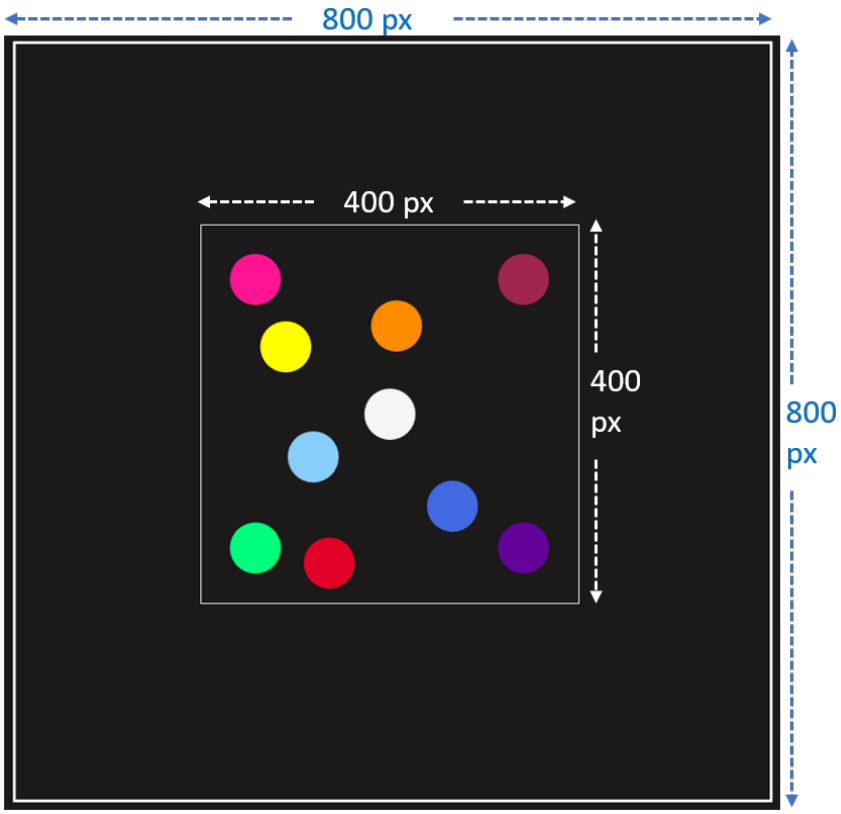

Figure 5.20: Static-Dynamic Authentication Interface with 400×400 px dimension (800×800 px boundary is shown for comparision).

5.4.7.12 System Accuracy

Table 5.12 shows the system accuracy in recognizing the passwords entered, and path recognition accuracy by considering all the entries on the static-dynamic extreme interface.

5.4.7.13 Recognition Error

Table 5.13 show the recognition errors for all the true passwords entered on the static-dynamic extreme interface.

5.4.7.14 Discussion

From the results we observe that under both the extreme conditions, the static-dynamic interface performs poorly. With an animation time of 1 second (800×800 px dimension), a user is

Table 5.12: Static-Dynamic Extreme Interface: system accuracy at extreme conditions

	1 second 800×800 **px**	400×400 **px** **2 seconds**
Passwords Entered	40	40
Password Recognition Accuracy	70%(28/40)	40%(16/40)
Path Recognition Accuracy	90% (144/160)	76.88% (123/160)

Table 5.13: Static-Dynamic Extreme Interface: recognition error based on the Levenshtein distance

	1 second 800×800 **px**	400×400 **px** **2 seconds**
0 Error	**70%** (28/40)	**40%** (16/40)
1 Error	**20%** (8/40)	**45%** (18/40)
2 Error	**10%** (4/40)	**12.5%** (5/40)
3 Error	**0** (0/40)	**2.5%** (1/40)

hardly able to follow the circle since it moves significantly fast. Also, reducing the dimension to 400×400 px (2 seconds animation) results in short gestures (scan-path) leading to recognition errors.

5.4.7.15 Static-Dynamic Disturbed Calibration

Gaze-based authentication systems are susceptible to calibration errors resulting in lower accuracy. This is because, if the authentication method requires precise pointing like entering PIN with gaze, fixating on certain points on an image, etc., any offsets in the calibration leads to im-

precise gaze input. However, in our gaze gesture-based approach we use template matching, and hence, errors in calibration and offsets in gaze input does not impact the accuracy as long as the user performs an approximate gesture. To test this assumption, the user was asked to get up and walk around for a few minutes. Upon return, the eye tracker was not re-calibrated, leaving the authentication system susceptible to calibration errors. Similar to previous experiments the user enters two true and two false passwords, and the animation speed was set to 2 seconds.

5.4.7.16 System Accuracy

The system accuracy, F-measure, and confusion matrix for 2 second animations on the static-dynamic interface with disturbed calibration are listed in Table 5.14.

Table 5.14: Static-Dynamic Disturbed Calibration - 2 Seconds Animation: Confusion Matrix, Authentication Accuracy, and F-Measure

	True Password	False Password	Accuracy	F-Measure
True Password	97.5%	2.5%	98.75%	0.99
False Password		100%		

5.4.7.17 Recognition Error

Table 5.15 show the recognition errors for all the true passwords entered on the static-dynamic interface with disturbed calibration.

Table 5.15: Static-Dynamic Disturbed Calibration - 2 Seconds Animation: recognition error based on the Levenshtein distance

0 Error	1 Error	2 Error
97.5% (39/40)	**0%** (0/40)	**2.5** (1/40)

5.4.7.18 Individual Path Recognition

Table 5.16 shows the path recognition accuracy for all the entries the static-dynamic interface with disturbed calibration.

Table 5.16: Static-Dynamic Disturbed Calibration - 2 Seconds Animation: path recognition accuracy

PINs Entered	Total Paths	Recognized Paths	Accuracy
80	320	309	**96.56%**

5.4.7.19 Recognition Error

As we hypothesized, since we use template matching to recognize the user's gesture, errors in calibration do not significantly impact the accuracy. Interestingly, it occurred that accuracy with disturbed calibration (98.75%) was slightly higher than with true calibration(97.5%). However, further analyzing the individual path recognition it can be found that disturbed calibration does reduce the accuracy (98.13% to 96.56%).

5.4.8 Gaze- and PIN-based Authentication

To compare the accuracy of our authentication interface and its susceptibility to video analysis attacks through multiple threat models, we developed a Gaze and PIN-based authentication system. The PIN-based authenticating interface uses a standard layout of numbers arranged in a 4×3 grid as seen at most of the ATMs. For consistency in comparison the numeric grid was also placed in a space of 800×800 square as shown in Figure 5.21. All the digits are placed at uniform distance on the horizontal as well as vertical directions.

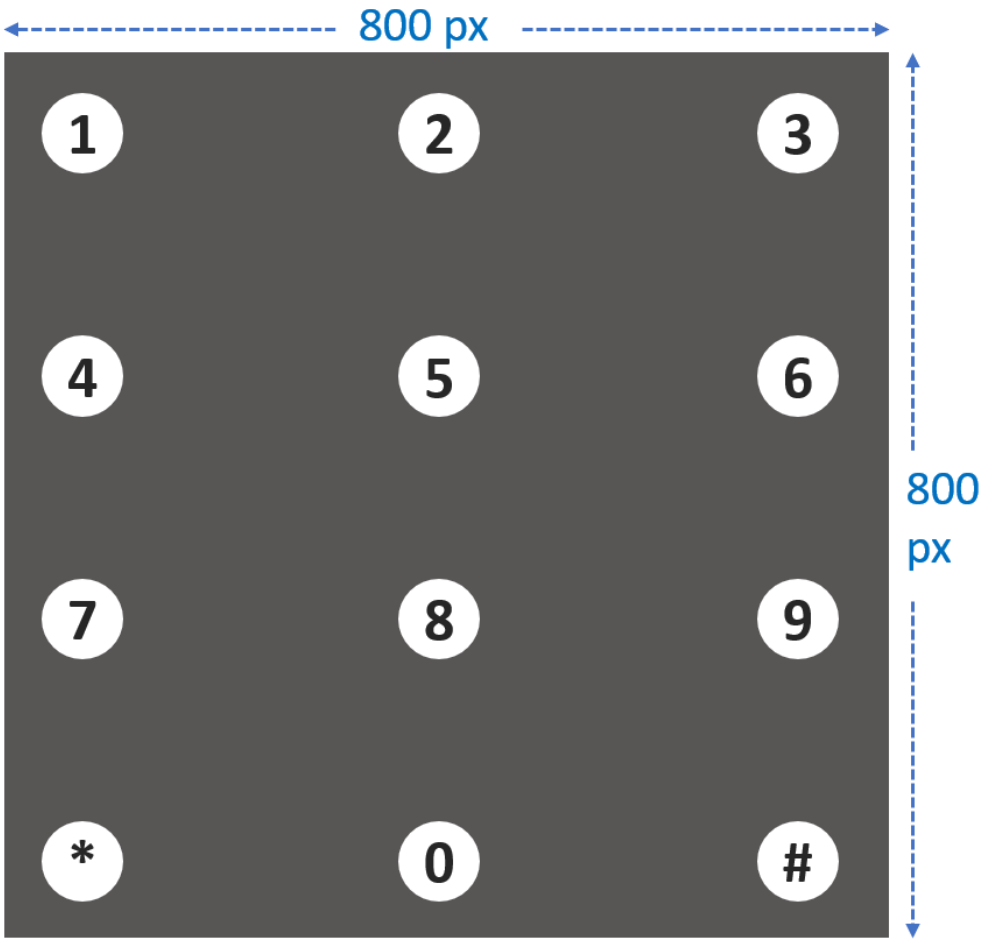

Figure 5.21: PIN Interface.

5.4.8.1 Authentication Procedure

A user authenticates with a PIN of 4 digits. To enter the PIN with gaze, there were two design choices: 1) on the authentication interface, the user fixates on each digit of the PIN for a dwell time of 150 to 200 ms, and 2) the user looks at the digit on the interface and selects it by pressing a hot key as used in [37]. We have used a hot key (A) for digit selection, and this is because of two reasons. First, in our authentication interface with colored circles, an animation is initiated by pressing a hot key (A), hence for the consistency of the activation method we used selection of the digit through a hot key. Second, fixation-based selection of digits needs high precision, and

is susceptible to errors since a user can not accurately perceive the dwell time. In most cases, the user may keep focusing at the same digit for more than the dwell time. Because of this, the user might have intended to enter say "3" but the system may recognize a stream of the same digit like "333". Situations like these raises uncertainty - whether the user indeed intended to enter "3" two times or it was an error. A key press based activation resolves this issue since for each key press only one gaze point is recorded.

5.4.8.2 PIN Recognition

PIN recognition is a simple process that uses Euclidean distance between the points. For each recorded gaze point, we compute the Euclidean distance to the center of every digit on the interface. The digit at the least distance from the gaze point is selected as the digit entered by the user. If all the 4 digits entered by the user match with the PIN, the user is authenticated. Figure 5.22 shows the distribution of gaze points when the user enters the PIN 1685.

5.4.8.3 Evaluation and Results

The same set of participants who evaluated gaze gesture-based interfaces also evaluated the gaze and PIN-based interface. Each participant entered two true and two false passwords.

5.4.8.4 System Accuracy

The system accuracy, F-measure, and confusion matrix for gaze and PIN-based interface are listed in Table 5.17.

Table 5.17: PIN Interface: Confusion Matrix, Authentication Accuracy, and F-Measure

	True Password	False Password	Accuracy	F-Measure
True Password	75%	25%	87.5%	0.86
False Password		100%		

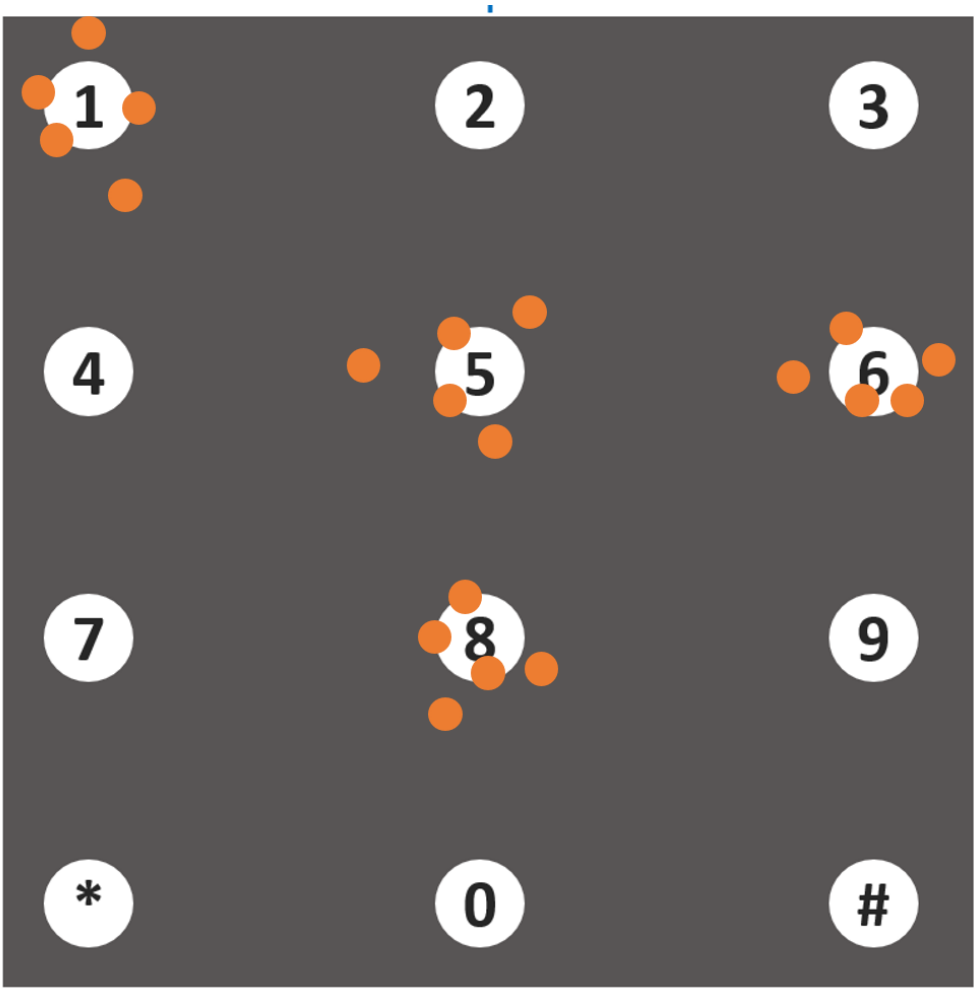

Figure 5.22: PIN Interface - Distribution of gaze points, after filtering, for pin 1685.

5.4.8.5 Recognition Error

Table 5.18 show the recognition errors for all the true passwords entered on the gaze and PIN-based interface.

5.4.9 Qualitative Evaluation

Each user completed a modified version of NASA-TLX questionnaire, a widely-used, subjective, multidimensional assessment tool that rates perceived mental workload and aspects of performance [225]. They rated both Gaze-PIN based entry system and our system (DyGazePass) across

Table 5.18: PIN Interface: Recognition error based on the Levenshtein distance

0 Error	1 Error	2 Error	4 Error
75% (30/40)	**17.5%** (7/40)	**2.5%** (1/40)	**5%** (2/40)

various metrics. We discuss each individual question and see if there exists differences between these authentication methods.

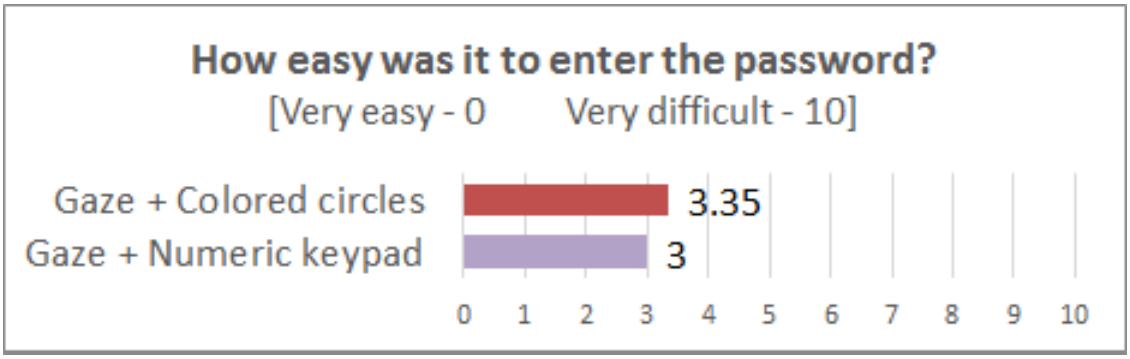

Figure 5.23: Q1: Matched-pairs t-test: P = 0.41 (P > 0.05)

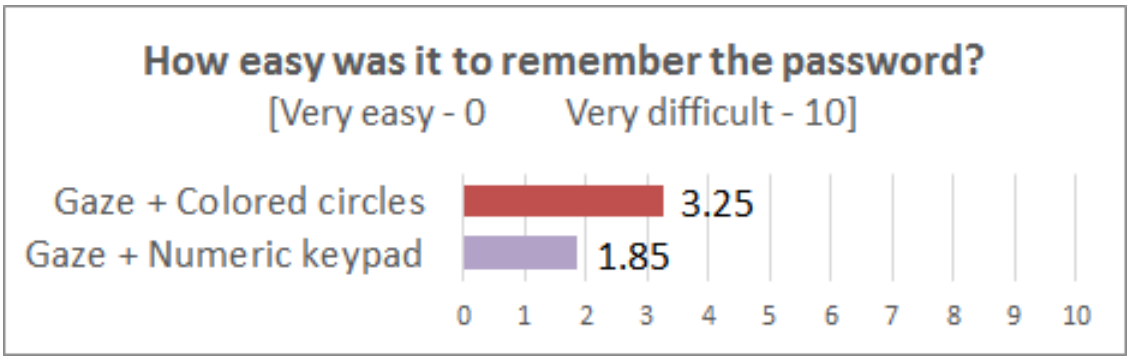

Figure 5.24: Q2: Matched-pairs t-test: P = 0.02 (P < 0.05)

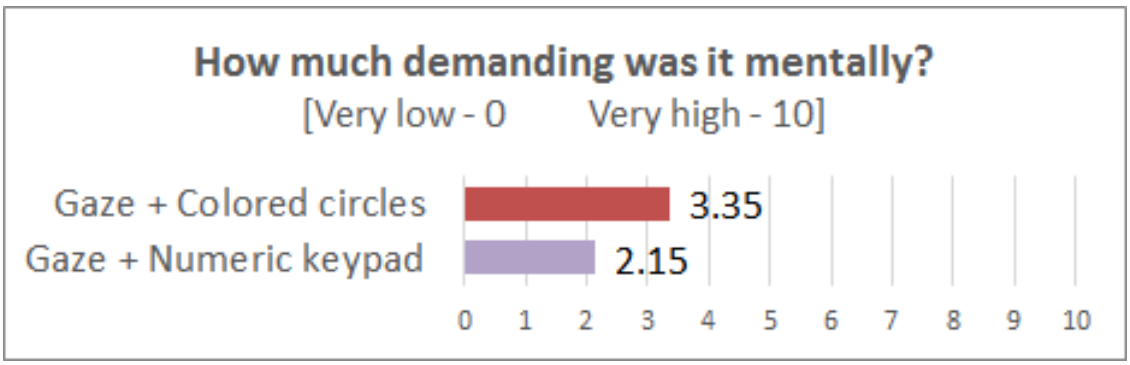

Figure 5.25: Q3: Matched-pairs t-test: P = 0.01 (P < 0.05)

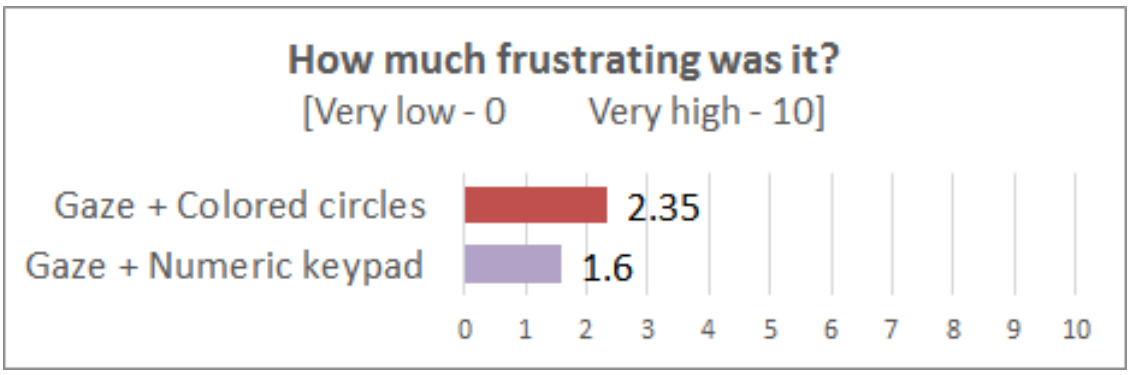

Figure 5.26: Q4: Matched-pairs t-test: P = 0.22 (P > 0.05)

Based on the responses to Q1 and Q4, it can be found that the participants do not find it difficult to enter password on one interface over the other (P > 0.05). Hence, it suggests that there is no difference in the workload on the user when entering a password on either of the interfaces. However, responses to Q2 and Q3 suggests that remembering a password as a set of colors and recollecting it during the authentication is mentally demanding when compared to remembering and recollecting a password as a set of numbers (P < 0.05).

5.4.10 Threat Models

For hacking numeric password and colored circles password, we assume two threat models: 1) single video iterative attack, and 2) dual video iterative attack. Figure 5.27 shows the placement of front and back cameras used to record videos of the users while authenticating.

Figure 5.27: Front and back camera positions for both single and dual video threat models.

5.4.10.1 Single Video Iterative Attack

In this model of attack, the attacker is made available a video stream of the user's face clearly showing the eye movements while authenticating. This is similar to a casual observer focusing at the eyes of the victim when the victim is authenticating using a gaze-based authenticating method. Here, the availability of the video extensively helps the attacker since the attacker can watch the video any number of times, and control the video playback for deeper analysis.

5.4.10.2 Dual Video Iterative Attack

In the dual video iterative attack, the attacker is made available two video streams: 1) a video stream of the user's face clearly showing the eye movements while authenticating, 2) a video stream of the authentication interface clearly showing any dynamic changes on the interface. For example, a video of the interface showing the transition paths of the circles, their colors, and the circles interchanging their positions for each new animation.

5.4.11 Password Hacking

We recruited 12 participants (9 male, 3 female), all were either graduate or undergraduate students, with ages varying between 18 and 30 ($\mu_{age} = 23.75$). Each participant hacked 2 numeric passwords and 2 colored circle passwords.

5.4.11.1 Numeric Password Hacking

Numeric password hacking is evaluated under a threat model of single video iterative attack (video of the user's face), and this attack requires no second video since the interface does not change. Each participant was provided a video of a user authenticating using a numeric password. The video was randomly chosen from a set of videos recorded in the first phase. The participant was given a maximum of 10 minutes or 3 tries, whichever is the earliest.

Table 5.19: Numeric Password Hacking: the number of passwords hacked across each try

Video	1st Try	2nd Try	3rd Try	Total
Single	4 (16.7%)	6 (25%)	9 (37.5%)	79.2%

A total of 24 passwords were attacked (12 x 2), and 79.2% (19/24) passwords were correctly recognized. The cumulative time taken to attack all the 24 passwords was 104 minutes, leading to

an average time of 4.33 minutes spent on hacking a password either successfully or not. Table 5.19 shows the number of passwords hacked across each try.

5.4.11.2 Colored Circles Password Hacking

For hacking authentication based on moving colored circles strategy, we used videos of users authenticating on static-dynamic interface (800×800 dimension and 2 seconds animation) as it was the most practical authentication method compared to dynamic interface. Hacking passwords on the static-dynamic interface was evaluated under two threat models.

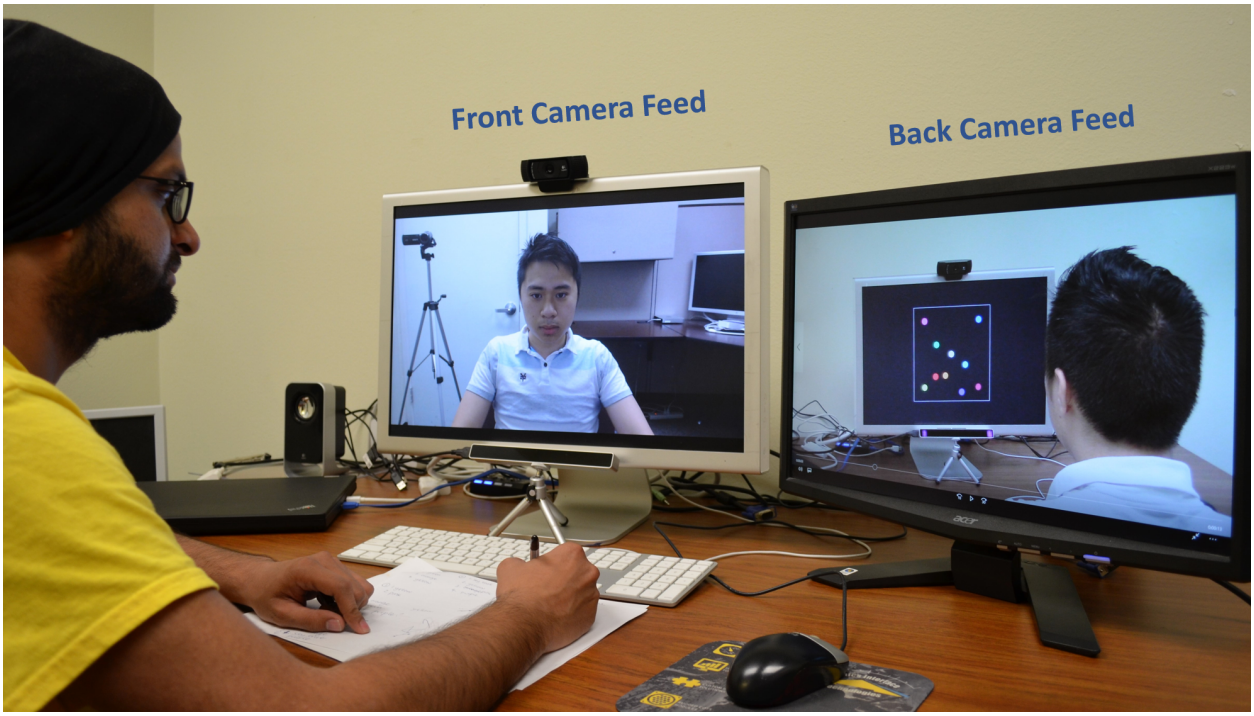

Figure 5.28: A hacker is trying to guess the colored circles password through dual video iterative attack (static-dynamic interface).

5.4.11.3 Single Video Iterative Attack

Each participant was provided a video of a user authenticating using colored circles password. As this is a single video iterative attack, the hacker was provided with the video is of the user's face,

Table 5.20: Colored Circles Password Hacking: number of passwords hacked across each try

Video	1st Try	2nd Try	3rd Try	Total
Single	0	0	0	0%
Dual	4 (16.7%)	0	0	16.7%

clearly showing the movement of eyes during authentication. The video was randomly chosen from a set of videos recorded in the first phase. The participant was given a maximum of 10 minutes or 3 tries, whichever is the earliest. A total of 24 passwords were attacked (12 x 2), and 0% (0/24) passwords were correctly recognized (Table 5.20). The cumulative time taken to attack all the 24 passwords was 58 minutes, leading to an average time of 2.42 minutes spent on hacking a password either successfully or not. Since no password was hacked, it suggests that the system is foolproof to single video iterative attack, and this was expected since the interface is dynamic, the path of the cirles are changing, and the circles are interchanging their positions at the end of an animation. Hence, without any details about the interface the user has to guess the password out of $10 \times 10 \times 10 \times 10$ permutations.

5.4.11.4 Dual Video Iterative Attack

This is an advanced attack, as we are assuming that the hacker has access to two videos one showing the user's face and the other showing the authentication interface during authentication session as shown in Figure 5.28. Each participant was provided with two videos (user's face and interface) of a user a user authenticating. The participant was given a maximum of 10 minutes or 3 tries, whichever is the earliest. A total of 24 passwords were attacked (12 x 2), and 16.7% (4/24) passwords were correctly recognized. The cumulative time taken to attack all the 24 passwords was 202 minutes, leading to an average time of 8.46 minutes spent on hacking a password either successfully or not. While this kind of attack is sophisticated, our system is still resilient to such attacks. Table 5.20 shows the number of passwords hacked across each try.

We hypothesized that participants will not be able to crack the password even with dual videos, and even if they do, they would take more than 3 tries. Interestingly, 4 passwords were cracked in the first try. 1 participant cracked 2 passwords, and 2 participants cracked 1 each. During the interview, these participants shared that they could crack the password since they were able to exactly sync both the videos and identify the start and end of each animation. For the passwords that were hacked, the victim (user authenticating) was unknowingly giving out the information about the start and end of the animation in either of two ways. First, the victim was hitting the hot key hard and the sound generated informs a hacker that the animation has just started so that the videos can be synced. Second, long pauses between each animation again helps the hacker to recognize the start of an animation. Hence, to avoid these kinds of intelligent attacks, the user should avoid giving out any information that helps the hacker in recognizing the start of an animation.

5.4.11.5 Qualitative Evaluation

Following the hacking session, participants were interviewed to understand the strategy they used to crack numeric and colored circles password and following were their feedback.

*"**P55:** PIN was like super easy. But the Dynamic interface was difficult. And while the circles crossover each other you loose the track of actual shapes. For Dynamic shapes it takes lot of effort to crack the password."*

*"**P57:** PIN cracking is easy and assumption is that the user was looking at center. Left and Right are easy to guess and since we know the positions of digits, it was easy. For Dynamic interface, i have no clue on password based on the eye moments. I tried to trace the moment of shapes with that of eye moments which was good strategy but the random positioning made it impossible."*

*"**P58**: The dynamic interface is very harder for me to crack and PIN was super easy. Static positioning of PIN makes it easy. Maybe more trials will help but not for sure."*

*"**P61:** The fact that the PIN has fixed location, we need to know where the eye line/level is, then it becomes easy to crack, but it is nearly impossible to crack the dynamic interface."*

*"**P62:** Pin Code was easy and obvious, but the dynamic circles/dots make it tougher. In PIN based*

I need to concentrate on only eye moment but in dynamic circle we need the placement as well as motion."

5.4.12 Discussion

Our primary goal was to create a gaze-based authentication system that addresses the limitations of existing systems by supporting high accuracy, being robust to calibration errors, and resilient to video analysis attacks. We discuss how these metrics are influenced by the various parameters of a gaze gesture-based authentication system.

5.4.12.1 Interface Dynamics Vs Accuracy

When comparing the accuracies and feedback of the users for both dynamic and static-dynamic interfaces, it is suggestive that as the interface becomes more dynamic users may find it overwhelming for a focused task like authentication which results in reduced accuracy. In addition, when gaze gestures are used for authentication, the interface should ensure that the margin for error is high. For example, the static-dynamic interface has only 5 moving circles during an animation, hence reducing the chances of generating similar paths, and this provides high margin of error for the user.

5.4.12.2 Authentication Time Vs Accuracy

Considering only the static-dynamic interface, it can be observed that for animation speed of 2 seconds or higher the accuracy does not differ much (3 seconds 98.75%, 2 seconds 97.5%). However, the accuracy reduces sharply to 70% when the animation speed was set to 1 second. Hence, with an animation speed of 2 seconds, considering a 4 color password, the least authentication time would be nearly 8 seconds. To reduce authentication time by reducing the animation speed to below 2 seconds will reduce the accuracy. However, when using 2 seconds animation, based on the level of security required, reducing the password length to less than 4 colors will reduce the authentication time to below 8 seconds.

5.4.12.3 Interface Dimension Vs Accuracy

Again considering the static-dynamic interface, it is evident from our experiments that (Table 5.12 and Table 5.13) reduction in screen dimension does reduce the accuracy. Reduced interface dimension causes two issues: 1) similar paths of moving circles, and 2) short gestures (scan-path), which are a source of recognition errors assuming that the gaze gestures are imprecise.

5.4.12.4 Interface Dynamics Vs Video Analysis Attacks

We discussed that as the interface becomes more dynamic, it impacts the accuracy. However, an interface with increased dynamic activities proportionally makes it harder to hack the password through video analysis attacks. As it was evident from our experiments that a dynamic interface is not susceptible to single video iterative attack, and has a low success rate with dual video iterative attack. We believe, further developments in gaze gesture-based authentication systems should consider these factors, and adjust each of these factors based on the environment where the authentication system will be used, and the level of security required.

6. GAZE GESTURE-BASED INTERACTIONS

As discussed in Introduction (Chapter 1), gaze-assisted interaction–an ability to interact with a computer using one's eye movements–is gaining momentum because of the availability of low cost eye trackers and improved gaze tracking accuracy. Furthermore, in Chapter 2 we discussed how gaze and foot-based multimedia input can help to achieve point-and-click interaction, and in Chapter 4 we discussed how gaze input can assist in text entry on computer.

Though gaze-assisted multi-modal systems enable accessible interactions, individuals with complete disability cannot use multi-modal interactions [86]. Such individuals could only move their eyes to express their active engagement [24]. Recent works have demonstrated the potential of gestures performed with eyes–gaze gestures–to interact with computer applications or for text entry [106, 107, 108]. However, these systems are limited by the number of supported gestures, recognition accuracy, need to remember the stroke order, lack of extensibility, and system complexity. Gaze gesture-based interactions would not only support individuals with disability or scenarios of situational impairment, but they also enable rich interactions.

We present a gaze gesture-based interaction framework where a user can design gestures and associate them to appropriate commands[1]. There are two significant advantages of our gaze gesture framework: 1) using gaze gestures, common interactions like minimize, maximize, scroll, and so on can be performed without switching the hand between keyboard and mouse, and 2) no need to remember a complex set of shortcuts like shortcuts on a code editor, which vary across applications. Figure 6.1 shows gaze pursuits, the building blocks of gaze gestures, where a user follows a moving object on the screen to create a gaze gesture.

Furthermore, we present two gaze gesture recognition algorithms: 1) a template matching algorithm, and 2) a geometric features based decision tree algorithm. Unlike the other gaze gesture

[1]*Parts of this chapter are reprinted with permission from "A Gaze Gesture-Based Paradigm for Situational Impairments, Accessibility, and Rich Interactions" by Rajanna et al., 2018. Publisher and Copyright holder ACM Digital Library, 2018, New York. Conference ETRA '18: 2018 Symposium on Eye Tracking Research and Applications Proceedings - doi.org/10.1145/3204493.3208344

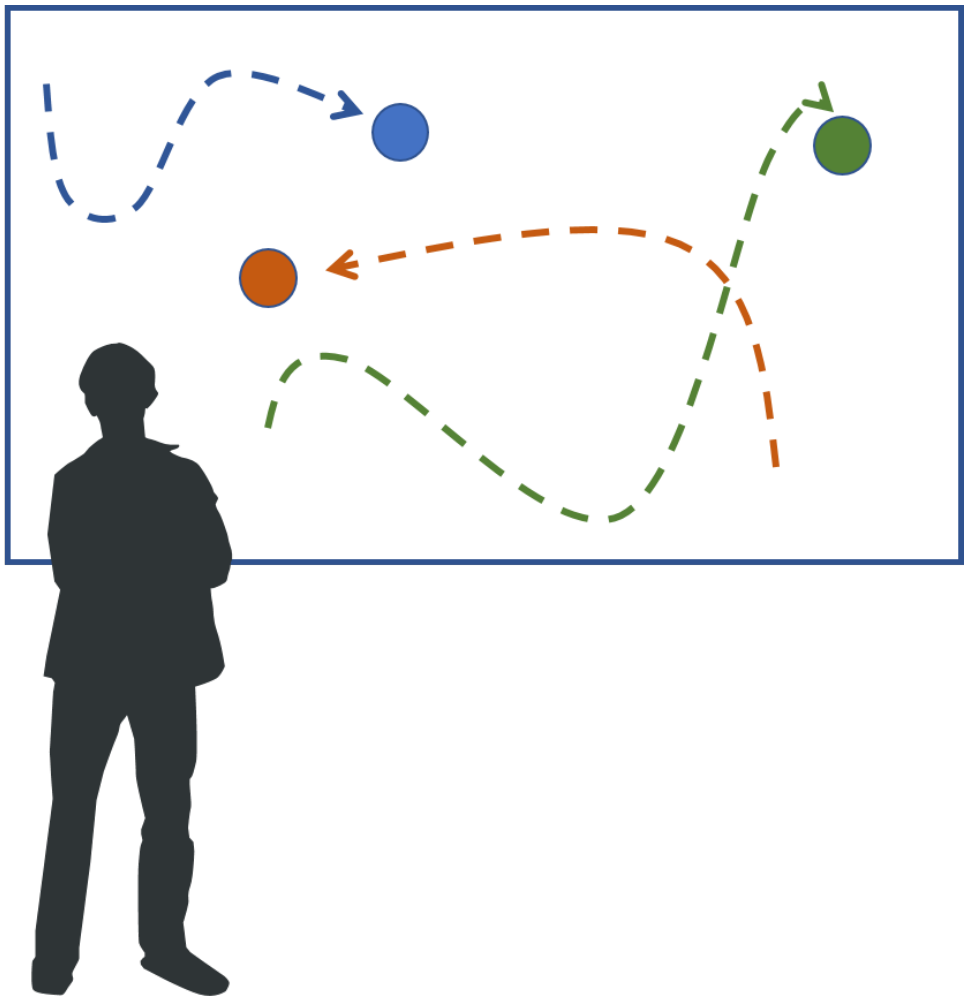

Figure 6.1: Performing gaze gestures through smooth pursuits eye movements, i.e., the user follows the transition of objects on the screen.

recognition frameworks, our gesture recognition framework is independent of the screen size, resolution, and the user can draw the gesture anywhere on the target application. Results from a user study involving seven participants showed that the system recognizes a set of nine gestures with an accuracy of 93% and a F-measure of 0.96. Next, the geometric features based decision tree algorithm functions by extracting the geometric features of the gesture, and recognizing the correct classification based on the training model. From a user study involving seven participants we found that the system accuracy is classifying 12 gestures was 90.2%. We envision, our algorithms can be leveraged in developing solutions for situational impairments, accessibility, and also for

implementing a rich interaction paradigm.

6.1 Introduction

Real-time information of the eye movements can be used for direct manipulation of the interface elements; this forms the basis of gaze-assisted interaction [24]. Gaze-assisted interaction is crucial in scenarios of situational impairments, i.e., the inability to work on a computer due to busy hands. Also, individuals with physical impairments or disabilities use eye movements for pointing, selecting, and typing tasks on a computer [226, 227]. With gaze-assisted interaction, an on-screen cursor navigation is achieved by mapping the eye movements to screen co-ordinates [26]. However, simply using gaze positions for target selection leads to inadvertent activations which is also known as the Midas Touch issue [65].

Multiple solutions like using dwell time, an eye blink, or using a supplemental input [86] have been proposed for target selection. In dwell-based activation, to execute a command, like a click, on the interface element the user fixates on the point of regard for a predefine interval of time (150-200 milliseconds) [65]. In blink-based activation a user's intent to execute a command (click) on the interface element is achieved by blinking the eye. In our approach, we re-contextualize gaze input as gestural input such that each gesture represents a user command (action) that can be executed on an application. For example, in the case of a situational impairment or disability, a gaze gesture can minimize, maximize, restore, or close an active window, or it can create a new tab, scroll, refresh, and so on an application like a browser. Also, as an example for a rich interaction paradigm, a user can create a set of gestures to execute code, debug, format, comment, etc., that can work across different code editors. This relieves the user from remembering different shortcuts on different code editors, or a time consuming option of using the mouse to select the action from the menu. Furthermore, an individual with speech impairment can use gaze gestures to make the computer speak specific phrases without switching to an assistive application. Therefore, an accurate gaze gesture recognition framework would allow for improved and extensible gaze-assisted interactions. Furthermore, while individuals with physical impairments primarily use gaze interactions on a computer, an individual with speech impairment can use gaze gestures to make

the computer speak specific phrases.

Currently, people use either the mouse or shortcuts to perform various actions available on the application menu. However, when using mouse, the user has to constantly switch the hand between keyboard and mouse, search for the menu item, and click it. This mode of interaction is slow and maybe unfavorable to some users. On the other hand, using shortcuts induces cognitive load on the users as they are required to remember various shortcuts across multiple applications. For example, different code editors use different shorts to execute code, however, using our gaze gesture framework a user can create a single gesture that can compile and execute code on multiple editors. Hence, the gaze interaction framework avoids switching the hand between input devices, and liberates users from remembering complex set of shortcuts.

6.2 Prior Work

The feasibility of gaze-assisted interaction was first demonstrated by Jacob [65]. Since then gaze-assisted interactions have been used for point-and-click [228, 229, 86], typing [75, 230], authentication [30, 100, 102], biometrics [231], performing secondary actions like zooming and panning [118], etc. Focusing specifically on some of the major research in gaze gesture-based interaction, Drewes et al., presented a framework to interact with computers using gaze gestures [106]. The authors implemented a gaze gesture algorithm based on mouse gestures, where users move their gaze in a combination out of eight directions to draw a gesture and execute an associated action. Wobbrock et al., presented EyeWrite: a gaze typing system, where characters are entered by performing predefined gestures for each character [107]. EyeWrite achieves an average typing speed of five words per minute, and the participants felt it was easier to use EyeWrite than on-screen keyboard. Bulling et al., presented a wearable EOG goggles using which gaze gestures can be performed as presented in [106] to interact with computers. Similarly, the usability of gaze gestures is demonstrated for gaming [108]. All the prior systems translate gaze gesture into a series of directional movements which limit the recognition accuracy because of jittery eye movements, and also, remembering a gesture as a sequence of directional changes is hard [106]. As we see, the amount of research toward utilizing gaze gestures is limited, and the existing systems have various

limitations. To address these limitations, our approach considers a gesture as a sketch stroke with a series of points, and the gesture is compared against a set of templates for recognition. This approach results in high accuracy, the gestures are independent of the screen size and resolution, the user is not required to remember the exact gestures, and the gestures can be executed anywhere on the screen.

6.3 System Architecture

The system consists of a Gaze Tracking Module, and Gesture Recognition Engine (Figure 6.2). The gaze tracking module uses a table-mounted “Gaze Point" eye tracker that provides (X,Y) gaze coordinates at 150 Hz. The gesture recognition engine constantly receives gaze points from the eye tracker, and is responsible for recognizing the gesture performed and executing the associated action on the target application. The beginning of the gesture is indicated by either pressing a hot-key (e.g., F2), or fixating for nearly 200 ms on the top left corner of the screen. Figure 6.2 shows the gaze gesture-based interaction framework in action where the user is performing actions on the browser using gaze gestures. Figure 6.3 shows the gesture design interface where the user creates gestures with eye movements and associates an action to each gesture.

6.4 Template Matching Algorithm

The gesture recognition engine performs template matching to match the gesture performed by the user (candidate gesture) to one of the various template gestures [232]. This is a multistage process (Figure 6.4), where the candidate gesture (6.4.A) is first sampled to N = 220 points (6.4.B). After sampling, the centroid of the gesture is calculated, and the centroid is moved to (0,0) co-ordinate, and also, all other points are moved to new points relative to the centroid (6.4.C). Finally, the transformed candidate gesture is compared with a set of template gestures by computing the Euclidean distance (6.4.D) between corresponding points as shown in Equation 1. The template gesture that is at a least root-mean-square distance from the candidate gesture is chosen as the gesture performed by the user.

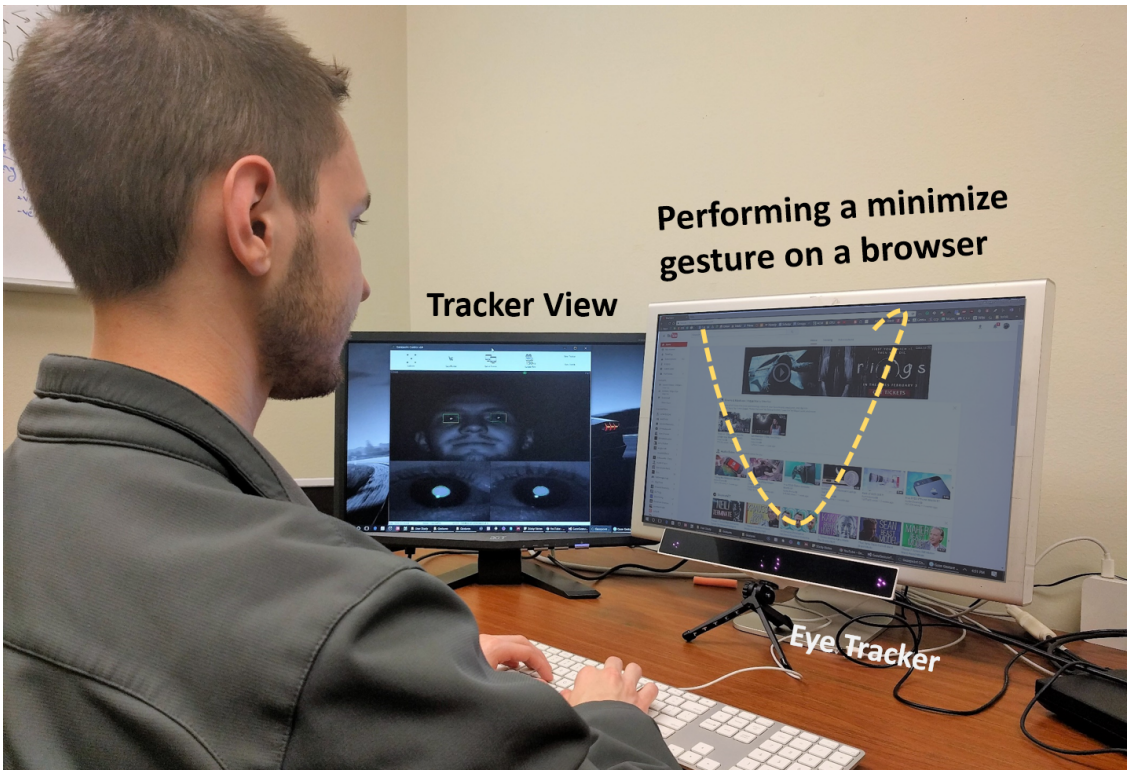

Figure 6.2: A user minimizing the browser with a gaze gesture.

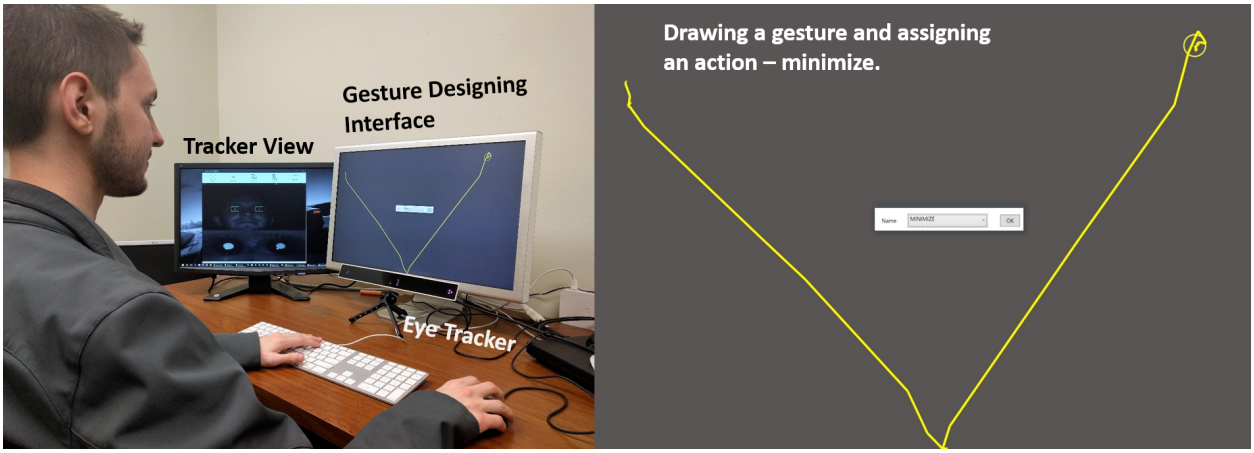

Figure 6.3: A user draws a gesture with their eye movements, and assigns a dedication action (e.g., minimize).

$$\Delta DT = \frac{\sum_{p=1}^{N} \sqrt{(C[p]_x - T[p]_x)^2 + (C[p]_y - T[p]_y)^2}}{N} \tag{6.1}$$

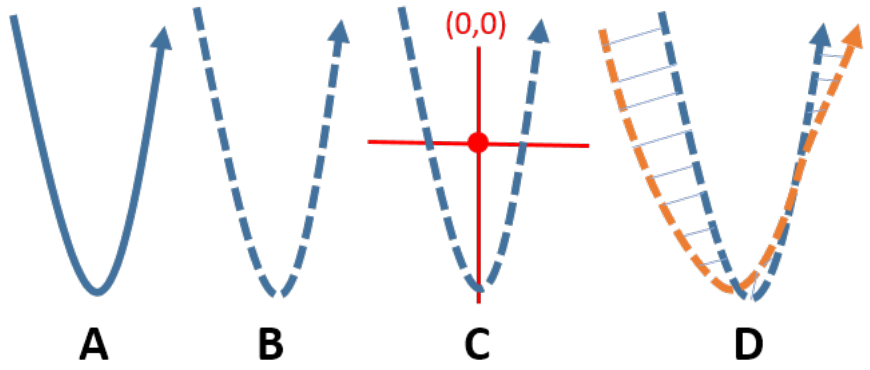

Figure 6.4: Pattern matching algorithm: A - Candidate gesture, B - Sampling, C - Centroid moved to origin (0,0), D - Computing Euclidean distance to a template gesture.

where p is a point on a gesture, C - candidate gesture, T - template gesture, and ΔDT - average distance to a template gesture.

6.4.1 System Evaluation and Results

We evaluated the gesture recognition accuracy and usability of the system through a preliminary study by involving seven users ($\mu_{age} = 26.28$). At the beginning of the study, each user was described about the gaze gesture-based interaction, and was asked to perform sample gestures on the screen. During the study, each user interacted with the browser by performing nine gestures, shown in Figure 6.5, to execute the associated actions like minimize, maximize, etc., on a browser as shown in Figure 6.2. The gaze interaction framework achieved a recognition accuracy of **93%** and a F-measure of **0.96**. The confusion matrix of the gestures performed is shown in Table 6.1. Also, the users shared that with practice, it was easy and quicker to perform gaze gestures than switching to a mouse and selecting the command from the menu, and the system was found to be responsive.

6.5 Decision Tree Algorithm

Another approach toward recognizing the path traversed by the user is through the decision tree algorithm [233]. In this method, we first create the model for the classification algorithm using multiple template paths. For creating the model, we extract the unique features of each path.

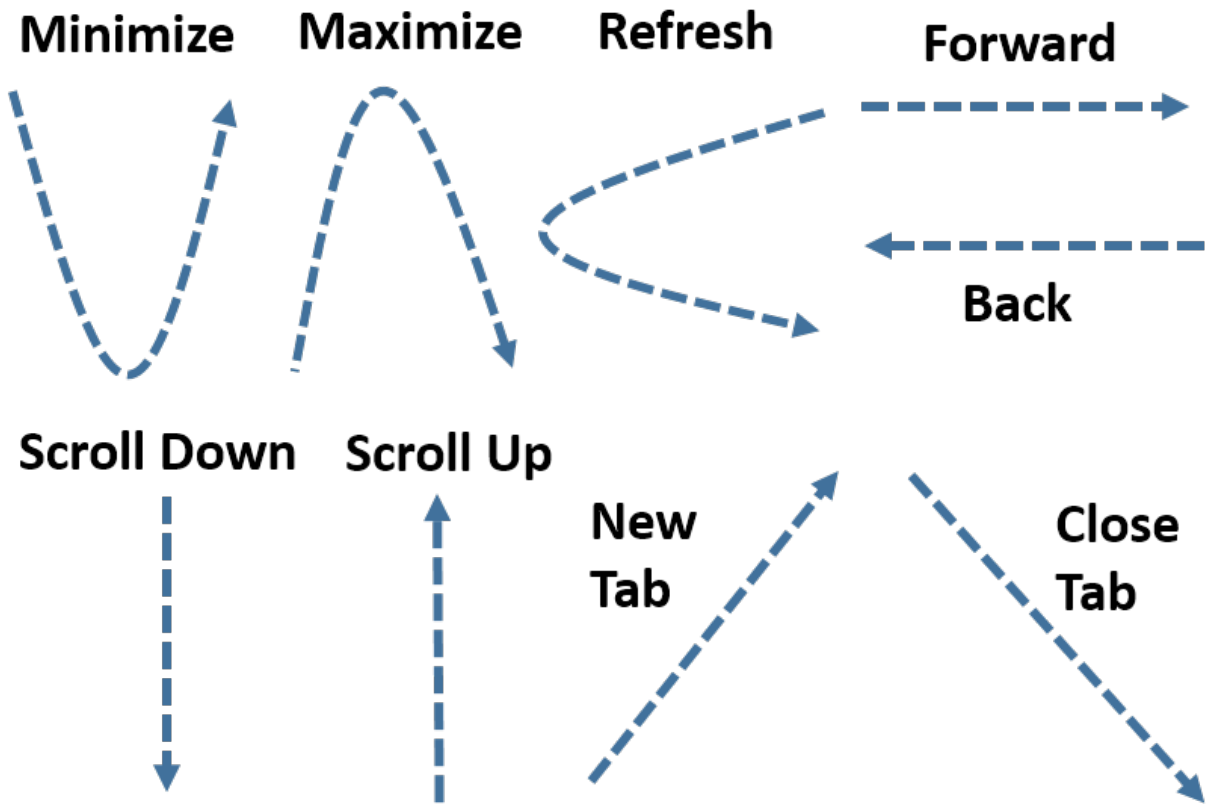

Figure 6.5: A set of gestures designed to interact with a browser.

The features we considered are:

- Starting Point
- Ending Point
- Area of the bounding box
- Length of the bounding box diagonal
- Slope of the bounding box diagonal

A pictorial depiction of the features considered are shown in Figure 6.6

Table 6.1: Confusion Matrix - Template Matching. Key: A - Minimize, B - Maximize, C - Forward, D - Back, E - Scroll Down, F- Scroll Up, G - Refresh, H - New Tab, I - Close Tab

	A	B	C	D	E	F	G	H	I
A	0.88		0.04	0.04			0.04		
B		1.0							
C			1.0						
D				0.84	0.16				
E					1.0				
F	0.04			0.04		0.88		0.04	
G				0.12			0.88		
H				0.04				0.96	
I		0.04		0.04					0.92

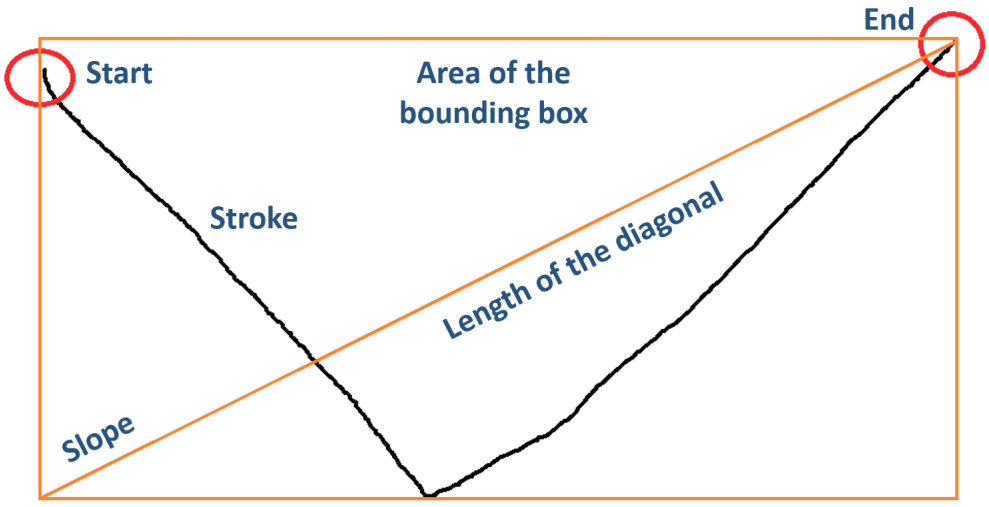

Figure 6.6: Decision tree features

6.5.1 System Evaluation and Results

To test the accuracy of the decision tree algorithm, we recruited seven participants and each participant performed 12 gestures through smooth pursuit eye movements. Data from one participant was discarded due to poor calibration. For classification using decision tree we used Accord .NET machine learning framework[2]. The decision tree algorithm achieved an accuracy of 90.2%, and the confusion matrix is shown in the Table 6.2.

Table 6.2: Confusion Matrix - Decision Tree

	G1	**G2**	**G3**	**G4**	**G5**	**G6**	**G7**	**G8**	**G9**	**G10**	**G11**	**G12**
G1	0.8			0.2								
G2		0.8								0.2		
G3			1.0									
G4				0.8					0.2			
G5					0.8						0.2	
G6						0.8		0.2				
G7							1.0					
G8								1.0				
G9				0.2					0.8			
G10										1.0		
G11											1.0	
G12											0.2	0.8

[2]accord-framework.net

6.6 Discussion

We presented a gaze gesture-based interaction framework for situational impairments, accessibility, and rich interactions. The gesture recognition algorithms used addresses the limitations of the existing gaze gesture recognition technique, and also supports high accuracy. We observe that the template matching algorithm has an accuracy of 93% and the decision tree algorithm has an accuracy of 90.2%. This result is not surprising since the template matching algorithm matches the user's scan-path against all the template paths, and finds the template path that is at the least Euclidean distance. On the other hand, the decision tree algorithm extracts the unique features of the template paths and their associated classes. Any candidate path whose features match closely with a given feature set in the model, the class of the matching template is assigned as the class of the candidate path. Hence, there are chances of errors, resulting in lower accuracy in case of the decision tree algorithm. Though template matching algorithm is more accurate than the decision tree algorithm, it is relatively slower. In our studies, we found that the decision tree algorithm is thrice as fast as the template matching algorithm. Again, this efficiency in speed is expected since the decision tree algorithm checks the features extracted against the decision tree (model) that is built from multiple templates.

7. SUMMARY AND CONCLUSIONS

In Introduction (Chapter 1) we discussed the need for accessible, hands-free interaction methods. Though the advancements in technology has made computing available anywhere, anytime, and on any device we still mainly use the mouse, keyboard, and touch inputs to interact with computing devices. Majority of the current interaction methods expect the user to be actively involved, and be using their hands to interact with the computing device. This poses a challenge as in the scenarios of situational and physical impairments and disabilities an individual cannot use their hands to work on a computer. Some of the examples of situational impairments and disabilities include driving, a surgeon operating, a factory worker wearing thick gloves or with greasy hands, a musician playing an instrument, a person holding objects, and so on. Also, we discussed that the existing accessible solutions for individuals with impairments and disability are invasive, bulky, expensive, or inefficient [8, 9, 10]. Prosthetic technology has not yet reached a state of maturity where prosthetics are affordable and effective for day-to-day life [2, 7]. Due to the lack of appropriate accessible solutions, users with impairments are forced to use alternative interfaces and they do not have the same experiences as the able-bodied users when working on computing devices.

Eye tracking technology has shown potential as an accessible input modality [36, 37, 38]. Previous research works have demonstrated the usability of gaze input in various contexts [39, 40, 41, 42, 44]. Hence, in this dissertation research we focused on developing gaze-assisted interaction paradigms for a) enabling individuals to work on computing devices in the scenarios of situational impairments, and b) enabling individuals with physical impairments and disabilities to use the same interfaces and have the same experience as the able-bodied users when working on a computer. In line with the majority of the gaze-assisted interactions developed, we have considered developing interaction paradigms for point-and-click [63, 64, 65, 66, 67, 40, 68, 69, 70, 71], text entry [72, 73, 74, 75, 76], and authentication [37, 77, 78, 30, 79, 80]. We have achieved this by primarily developing a gaze and foot interaction framework that supports point-and-click interactions and text entry, and a gaze gesture recognition framework to support authentication and gesture-based

interactions.

In Chapter 2, we presented GAWSCHI, a gaze and foot-based point-and-click interaction system that addresses the Midas Touch problem. Existing gaze-assisted solutions that tried to address the Midas Touch problem have lower accuracy, require large targets, time consuming, and induce visual fatigue. GAWSCHI integrates seamlessly with the native interface of an operating system (e.g., Windows 10) and enables hands-free interactions. Through an evaluation involving 30 participants that performed 11 pre-defined tasks on a computer, we showed that gaze-assisted interactions using GAWSCHI is as good (time and precision) as mouse-based interaction. In addition, we also showed that the minimum dimension of the interface element should be above 0.60" x 0.51" for gaze input to match the performance of the mouse. Lastly, through NASA TLX survey, we showed that a gaze-assisted multi-modal interaction method that separates point-and-click interactions into pointing with gaze and selecting with the foot leads to low metal, physical, and temporal demand. Further details regarding the results can be found in Section 2.7.

While in Chapter 2 we discussed gaze-assisted point-and-click interactions in a desktop environment, it provided a better understanding of the feasibility of gaze and foot-based interactions. It also helped us in understanding how the gaze input compares to the mouse input when performing the standard tasks on a computer. Also, we learned the constraints, in terms of user interface dimensions, under which the gaze-assisted interaction works well. As a next step, it is essential to compare the gaze input to other standard inputs like the mouse and touch inputs through the standard evaluation metrics used in Human-Computer Interaction. One such validation is the comparison of input methods through Fitts' Law evaluation. Hence, in Chapter 3 we first compared the gaze and foot-based input with the mouse and touch inputs on a standard screen (up to 24"). We found that the gaze input has the lowest throughput (2.55 bits/s), and the highest movement time (1.04 s) of the three inputs. Also, though touch input involves maximum physical movements, it achieved the highest throughput (6.67 bits/s), the least movement time (0.5 s). Furthermore, in Section 3.3.6 we discussed why the touch input performs well, and why the gaze input has the lowest throughput despite the fact that the cursor can be moved quickly between the targets. From

analyzing cursor points transitions, we found that the gaze moves quicker between the targets, however, placing the cursor inside the target and selecting it consumes time. Hence, when using gaze-assisted interaction for selection, adopting a border crossing methodology is would lead to efficient interaction (lower time). Further details regarding the results can be found in Section 3.3.5.

Next, for interactions on large displays (up to 84"), we hypothesized that it is more convenient to use the gaze-based multi-modal input than other inputs. Through a Fitts' Law evaluation that compared gaze+foot multi-modal input to touch and mouse inputs on a large display, we found that if the Index of Difficulty (ID) of the task is above 2.5 bits/sec, the participants could not complete the study. For for ID below 2.5 bits/sec, touch achieved the highest throughput at 5.49 bits/s and gaze achieved the lowest throughput at 2.33 bits/sec. From the qualitative evaluation, we found that touch and mouse inputs result in increased shoulder/wrist fatigue and physical demand on the large display compared to the gaze input. Further details regarding the results can be found in Section 3.4.5.

In Chapter 4, we focused on the second major usability of the gaze input, i.e., text entry. Gaze-assisted text entry enables hands-free text entry in the scenarios of situationally-induced impairments and disabilities (SIID), and individuals with speech and motor impairments rely primarily on gaze-assisted text entry for communication [234, 24, 112]. While gaze-assisted text entry has been studied for more than 20 years, currently used dwell-based and even dwell-free selection methods have limitations with speed, accuracy, and usability. We presented a gaze and foot-based dwell-free typing system, and investigated two approaches to foot-based selection: a) using foot gestures, and b) foot press-based selection. These selection methods were compared against the standard dwell-based selection. We found that foot press-based selection that achieves a typing speed of 14.98 WPM beats dwell-based selection that achieved a typing speed of 11.65 WPM. Also, foot-based selection methods are preferred over dwell-based selection considering speed, accuracy, and usability. Lastly, we found that toe tapping is the most preferred gesture of the four gestures we used, and the user quickly learns to synchronize pointing with gaze and selecting the target with foot. More details regarding the results can be found in Section 4.7

In chapter 5 we presented a gaze-assisted authentication method as an accessible, hands-free authentication method and as a solution for protecting authentication from shoulder-surfing attacks. We developed an authentication method that uses unique gaze gestures to authenticate a user. The core idea of the authentication method is that the user follows moving shapes or colors on the screen to create gaze gestures that are recognized by the system. We presented two authentication strategies that use gaze gestures: 1) fixed transitions authentication, and 2) dynamic transitions authentication. While fixed transitions authentication achieves a recognition accuracy of 99%, the system is 60% susceptible to single video iterative analysis attacks. On the other hand, the dynamic transitions authentication achieves an accuracy of 97.5% and is 0% susceptible to single video iterative analysis attacks. Furthermore, we discussed how the interface dimension, speed of transitions, and the level of randomness influence the recognition accuracy.

In Chapter 1 we discussed about point-and-click interactions using a gaze and foot-based interaction framework. We also demonstrated that such a multi-modal gaze-assisted system performs at least as good as the mouse. While such a system is significantly useful for in the scenarios of situational and physical impairments and disabilities, not all individuals can engage in foot-based interactions. Individuals with complete disability will not have a greater control over their foot. Hence, in Chapter 6 we presented a gaze gesture-based interaction paradigm, where a user can design gestures and associate them to appropriate commands. Using these common set of gestures, a user can interact with various applications on the screen just through eye movements. Furthermore, we presented two gaze gesture recognition algorithms: 1) a template matching algorithm, and 2) a geometric features based decision tree algorithm. Template matching algorithm recognized a set of nine gestures with an accuracy of 93%, and the geometric features based algorithm recognized a set of 12 gestures with an accuracy of 90.2%. While template matching algorithm is highly accuracy, the algorithm takes thrice the time it takes for the geometric features based decision tree algorithm to recognize a gesture. Overall, both the algorithms provide a framework for creating a gaze gesture-based accessible interaction system.

In summary, as discussed in the Introduction, our objective was to address situational and phys-

ical impairments and disabilities by developing gaze-assisted, multi-modal, accessible solutions. We specifically focused on developing interaction paradigm for point-and-click, text entry, and authentication. We evaluated our solutions by simulating real-world scenarios, and measured the system performance through standard HCI procedures like the Fitts' Law, text entry metrics, authentication accuracy and video analysis attacks. Also, we compared the system performance to other gaze-assisted accessible solutions, and demonstrated how our solutions improve the performance and usability. Furthermore, we collected users' feedback through post-study interviews, and reflected on the feedback to improve the usability of our solutions. To summarize, our accessible solutions enable individuals to perform point-and-lick, text entry, and authentication operations on a computer using eye-movements. This makes a huge difference in the scenarios of situational and physical impairments and disability.

8. A SUMMARY OF RESEARCH QUESTIONS ADDRESSED IN THIS THESIS

Our aim was to develop gaze-assisted, multi-modal, hands-free interaction methods as a solution to address situational and physical impairments and disabilities, this goal translates to addressing various research questions related to the development, and evaluation of the efficiency and usability of gaze-assisted interaction methods and supplemental devices. The three primary interactions we addressed were point-and-click, text entry, and authentication.

8.1 Point-and-Click Interactions

Existing gaze-based point-and-click solutions are limited by low task efficiency and visual fatigue [26, 87]. These solutions use eye gaze for pointing, and a pre-defined dwell time (or blink) for selecting the target [26, 229]. Hence, they do not perform as efficiently as mouse-based interactions. Therefore, we developed a gaze and foot-based multi-modal system to support point-and-click interactions, and we address the following research questions in Chapter 2 and 3.

1. Can gaze-assisted interaction be further enhanced with a supplemental input to achieve the speed of mouse-based interactions?

2. If an additional input modality is used along with the gaze, what should be the characteristics of such an input modality?

3. Is gaze and foot-based interaction system as quick as the mouse in all interaction tasks, or does it have limitations?

4. Does the gaze and foot-based interaction system overcome the visual, mental, and physical fatigue experienced with dwell-based solutions?

5. What are the values of throughput, movement time, error rate, and effective target width for gaze and foot-based interactions on a standard display (up to 24")?

6. What are the values of throughput, movement time, error rate, and effective target width for gaze and foot-based interactions on a large display (up to 84")?

7. How do the Fitts' Law metrics for gaze input compare to mouse and touch inputs on both standard and large displays?

8. Does the gaze input support interactions on a larger screen (up to 84")?

9. What are the advantages and challenges of using gaze input on large displays?

8.2 Text Entry

Users experiencing situational impairments and users with physical impairments in their hands, arm, spine, and lower back can not conveniently use a physical keyboard to enter text on a computer [75, 112]. As a solution, we developed a gaze and foot-based dwell-free typing system that uses a virtual keyboard and gaze input for text entry on a computer. Also, we investigated two foot-based activation methods like foot-press and foot gestures. Majority of the gaze typing solutions use dwell time to select a key on the keyboard, and dwell-based selection is limited by the same issues as we discussed earlier. Most importantly, different users find different dwell times convenient for gaze typing, and hence a common dwell time cannot be used [24]. A shorter dwell time introduces a lot of errors, but a longer dwell time, though limits errors, reduces the gaze typing speed [112]. Also, dwell-based selection of the keys results in unintentional selections. In developing a gaze and foot-based dwell-free typing system we have addressed the following research questions in Chapter 4.

1. Can an efficient gaze typing system be created that leverages a multimodal approach: gaze input for pointing at the key and a supplemental input for selection of the key?

2. Can users coordinate their gaze and foot input to enter text on a computer. In other words is gaze and foot-based dwell-free typing system feasible?

3. Does a gaze and foot-based typing system achieve higher typing speed (WPM) and lower error rate than gaze and dwell-based typing system?

4. Does the gaze typing system that uses a supplemental foot input completely eliminate unintentional key selections?

5. If foot input is used as an additional input modality along with gaze, what are the different approaches to performing target key selection?

6. How the typing speed and error rate of foot gesture-based selection compare to foot press-based selection?

7. If the system supports multiple foot gestures, how do users make use of the available gestures? Do they switch between using different gestures or chose a convenient gesture and use the same gesture throughout?

8. What are the most commonly used and least commonly used foot gestures?

9. What foot gestures the participates find convenient to use and why

10. Do participants select a single foot gesture and use it throughout, or do they switch between using different gestures, to prevent stress on the foot?

11. Does a gaze and foot-based typing system provide user-friendly interactions by addressing the interaction issues found with gaze typing systems that use only gaze but no other supplemental inputs?

12. Do users learn over repeated usage of the system, and develop a familiarity with gaze and foot-based typing? Does this result in improved performance?

13. Does a gaze and foot-based typing system induce physical strain and cognitive load on the user?

14. From a usability perspective, what are the advantages and disadvantages of using a supplemental foot input with gaze typing?

8.3 Gaze-assisted Authentication

Trying to authenticate in the scenarios of situational and physical impairments and disabilities is either challenging or not possible at all. Also, knowledge-based authentication like password entry is susceptible to shoulder surfing attacks. we addressed the lack of an accessible and shoulder-surfing resistant authentication method by developing a gaze gesture recognition framework, and presenting two authentication strategies that use gaze gestures. In developing our static and dynamic authentication solutions, we have addressed the following research questions in Chapter 5.

1. Can a gaze-assisted authentication method enable users with situational and physical impairments authenticate securely?

2. Does a gaze-assisted authentication method based on gaze gestures achieve high accuracy over gaze and PIN-based authentication?

3. Is the gaze gesture-based authentication method robust to calibration errors?

4. What are the advantages and disadvantages of using objects with predefined shapes that always move along a specified path as the elements of a password?

5. In addition to using shapes as authentication elements, what other approaches can be used, and what are their advantages?

6. What different approaches can be used for the transition of objects/colors on the screen? what are the advantages and limitations of each method?

7. How does the interface dynamics affect the accuracy of the system?

8. Does time to authenticate influence system accuracy? if so how?

9. Does dimension of the interface impact accuracy? if so how?

10. Is gaze gesture-based authentication resilient to casual shoulder surfing attacks?

11. When considering video analysis attacks, does an authentication system based on the dynamic transition of the password elements is more secure than the authentication system that uses static transitions of the password elements?

12. Does the gaze gesture-based authentication induce cognitive load on the user?

8.4 Gaze gesture-based Interactions

We developed a gaze gesture recognition framework that enables users to design gaze gestures and associate them to appropriate commands like minimize, maximize, scroll, etc., on a computer. Such a system enables users with complete disability to work on a computer. In Chapter 6, we addressed the following research questions related to a gaze gesture-based interaction system.

1. What are the multiple ways to recognize gaze-gestures?

2. How can a gaze gesture recognition system be made independent of screen size, resolution, and stroke order?

3. What are the advantages and limitations of a template matching algorithm for gaze gesture recognition?

4. What are the advantages and limitations of a geometric features-based algorithm for gaze gesture recognition?

REFERENCES

[1] B. Hatscher, M. Luz, L. E. Nacke, N. Elkmann, V. Müller, and C. Hansen, "Gazetap: Towards hands-free interaction in the operating room," in *Proceedings of the 19th ACM International Conference on Multimodal Interaction*, ICMI 2017, (New York, NY, USA), pp. 243–251, ACM, 2017.

[2] C. M. Capio, C. H. Sit, and B. Abernethy, "Fundamental movement skills testing in children with cerebral palsy," *Disability and rehabilitation*, vol. 33, no. 25-26, pp. 2519–2528, 2011.

[3] M. Whitehead, "Definition of physical literacy and clarification of related issues," *ICSSPE Bulletin*, vol. 65, no. 1.2, 2013.

[4] C. M. Capio, C. H. Sit, and B. Abernethy, "2016 disability status report: United states," *Ithaca, NY: Cornell University Yang Tan Institute on Employment and Disability(YTI)*, 2018.

[5] K. Ziegler-Graham, E. J. MacKenzie, P. L. Ephraim, T. G. Travison, and R. Brookmeyer, "Estimating the prevalence of limb loss in the united states: 2005 to 2050," *Archives of physical medicine and rehabilitation*, vol. 89, no. 3, pp. 422–429, 2008.

[6] A. B. Cobb, *The Bionic Human*. The Rosen Publishing Group, Inc, 2002.

[7] D. Desmond and M. MacLachlan, "Psychosocial issues in the field of prosthetics and orthotics," *JPO: Journal of Prosthetics and Orthotics*, vol. 14, no. 1, pp. 19–22, 2002.

[8] V. A. Freedman, E. M. Agree, L. G. Martin, and J. C. Cornman, "Trends in the use of assistive technology and personal care for late-life disability, 1992–2001," *The Gerontologist*, vol. 46, no. 1, pp. 124–127, 2006.

[9] S. Keates, P. Clarkson, and P. Robinson, "Developing a methodology for the design of accessible interfaces," in *Proceedings of the 4th ERCIM Workshop*, pp. 1–15, 1998.

[10] S. K. Kane, J. P. Bigham, and J. O. Wobbrock, "Slide rule: making mobile touch screens accessible to blind people using multi-touch interaction techniques," in *Proceedings of the*

10th international ACM SIGACCESS conference on Computers and accessibility, pp. 73–80, ACM, 2008.

[11] M. Elepfandt and M. Grund, "Move it there, or not?: the design of voice commands for gaze with speech," in *Proceedings of the 4th workshop on eye gaze in intelligent human machine interaction*, p. 12, ACM, 2012.

[12] F. Sasangohar, I. S. MacKenzie, and S. D. Scott, "Evaluation of mouse and touch input for a tabletop display using fitts' reciprocal tapping task," in *Proceedings of the Human Factors and Ergonomics Society Annual Meeting*, vol. 53, pp. 839–843, SAGE Publications Sage CA: Los Angeles, CA, 2009.

[13] D. Klatt, "The klattalk text-to-speech conversion system," in *Acoustics, Speech, and Signal Processing, IEEE International Conference on ICASSP'82.*, vol. 7, pp. 1589–1592, IEEE, 1982.

[14] W. R. Hutchison, "Speech recognition system with network accessible speech processing resources," Aug. 31 2004. US Patent 6,785,647.

[15] J. R. Wolpaw, N. Birbaumer, D. J. McFarland, G. Pfurtscheller, and T. M. Vaughan, "Brain–computer interfaces for communication and control," *Clinical neurophysiology*, vol. 113, no. 6, pp. 767–791, 2002.

[16] A. Kübler and N. Birbaumer, "Brain–computer interfaces and communication in paralysis: extinction of goal directed thinking in completely paralysed patients?," *Clinical neurophysiology*, vol. 119, no. 11, pp. 2658–2666, 2008.

[17] D. Afergan, "Using brain-computer interfaces for implicit input," in *Proceedings of the Adjunct Publication of the 27th Annual ACM Symposium on User Interface Software and Technology*, UIST'14 Adjunct, (New York, NY, USA), pp. 13–16, ACM, 2014.

[18] A. Edwards, *Extraordinary Human-Computer Interaction: Interfaces for Users with Disabilities*, vol. 7. CUP Archive, 1995.

[19] J. Cassell, "Embodied conversational interface agents," *Communications of the ACM*, vol. 43, no. 4, pp. 70–78, 2000.

[20] L. Deng and X. Huang, "Challenges in adopting speech recognition," *Communications of the ACM*, vol. 47, no. 1, pp. 69–75, 2004.

[21] D. O'Shaughnessy, "Automatic speech recognition: History, methods and challenges," *Pattern Recognition*, vol. 41, no. 10, pp. 2965–2979, 2008.

[22] J. d. R. Millán, R. Rupp, G. Müller-Putz, R. Murray-Smith, C. Giugliemma, M. Tangermann, C. Vidaurre, F. Cincotti, A. Kubler, R. Leeb, *et al.*, "Combining brain–computer interfaces and assistive technologies: state-of-the-art and challenges," *Frontiers in neuroscience*, vol. 4, p. 161, 2010.

[23] D. Tan and A. Nijholt, "Brain-computer interfaces and human-computer interaction," in *Brain-Computer Interfaces*, pp. 3–19, Springer, 2010.

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

[25] C. H. Morimoto and M. R. Mimica, "Eye gaze tracking techniques for interactive applications," *Computer vision and image understanding*, vol. 98, no. 1, pp. 4–24, 2005.

[26] A. Duchowski, *Eye tracking methodology: Theory and practice*, vol. 373. Springer Science & Business Media, 2007.

[27] V. Rajanna and T. Hammond, "A fitts' law evaluation of gaze input on large displays compared to touch and mouse inputs," in *COGAIN '18: Workshop on Communication by Gaze Interaction, June 14–17, 2018, Warsaw, Poland*, COGAIN '18, (New York, NY, USA), ACM, 2018.

[28] J. C. Mateo, J. San Agustin, and J. P. Hansen, "Gaze beats mouse: hands-free selection by combining gaze and emg," in *CHI'08 extended abstracts on Human factors in computing systems*, pp. 3039–3044, ACM, 2008.

[29] V. Raudonis, A. Paulauskaite-Taraseviciene, and R. Maskeliunas, "Vision enhancement technique based on eye tracking system," in *Exploring the Abyss of Inequalities* (K. Eriksson-Backa, A. Luoma, and E. Krook, eds.), vol. 313 of *Communications in Computer and Information Science*, pp. 150–160, Springer Berlin Heidelberg, 2012.

[30] V. Rajanna, S. Polsley, P. Taele, and T. Hammond, "A gaze gesture-based user authentication system to counter shoulder-surfing attacks," in *Proceedings of the 2017 CHI Conference Extended Abstracts on Human Factors in Computing Systems*, CHI EA '17, (New York, NY, USA), pp. 1978–1986, ACM, 2017.

[31] V. Rajanna and T. Hammond, "A gaze gesture-based paradigm for situational impairments, accessibility, and rich interactions," in *Proceedings of the Tenth Biennial ACM Symposium on Eye Tracking Research and Applications*, ETRA '18, (New York, NY, USA), ACM, 2018.

[32] M. Tall, A. Alapetite, J. San Agustin, H. H. Skovsgaard, J. P. Hansen, D. W. Hansen, and E. Møllenbach, "Gaze-controlled driving," in *CHI'09 Extended Abstracts on Human Factors in Computing Systems*, pp. 4387–4392, ACM, 2009.

[33] M. A. Eid, N. Giakoumidis, and A. El-Saddik, "A novel eye-gaze-controlled wheelchair system for navigating unknown environments: Case study with a person with als.," *IEEE Access*, vol. 4, pp. 558–573, 2016.

[34] J. P. Hansen, A. Alapetite, I. S. MacKenzie, and E. Møllenbach, "The use of gaze to control drones," in *Proceedings of the Symposium on Eye Tracking Research and Applications*, ETRA '14, (New York, NY, USA), pp. 27–34, ACM, 2014.

[35] J. San Agustin, H. Skovsgaard, J. P. Hansen, and D. W. Hansen, "Low-cost gaze interaction: ready to deliver the promises," in *CHI'09 Extended Abstracts on Human Factors in Computing Systems*, pp. 4453–4458, ACM, 2009.

[36] S. Laqua, S. U. Bandara, and M. A. Sasse, "Gazespace: eye gaze controlled content spaces," in *Proceedings of the 21st British HCI Group Annual Conference on People and Computers:*

HCI... but not as we know it-Volume 2, pp. 55–58, BCS Learning & Development Ltd., 2007.

[37] M. Kumar, T. Garfinkel, D. Boneh, and T. Winograd, "Reducing shoulder-surfing by using gaze-based password entry," in *Proceedings of the 3rd Symposium on Usable Privacy and Security*, SOUPS '07, (New York, NY, USA), pp. 13–19, ACM, 2007.

[38] S. Nilsson, T. Gustafsson, and P. Carleberg, "Hands free interaction with virtual information in a real environment: Eye gaze as an interaction tool in an augmented reality system.," *PsychNology Journal*, vol. 7, no. 2, 2009.

[39] J. P. Hansen, K. Tørning, A. S. Johansen, K. Itoh, and H. Aoki, "Gaze typing compared with input by head and hand," in *Proceedings of the 2004 symposium on Eye tracking research & applications*, pp. 131–138, ACM, 2004.

[40] J.-Y. Lee, H.-M. Park, S.-H. Lee, S.-H. Shin, T.-E. Kim, and J.-S. Choi, "Design and implementation of an augmented reality system using gaze interaction," *Multimedia Tools and Applications*, vol. 68, no. 2, pp. 265–280, 2014.

[41] S. Sharmin, O. Špakov, and K.-J. Räihä, "Reading on-screen text with gaze-based auto-scrolling," in *Proceedings of the 2013 Conference on Eye Tracking South Africa*, pp. 24–31, ACM, 2013.

[42] M. V. Portela and D. Rozado, "Gaze enhanced speech recognition for truly hands-free and efficient text input during hci," in *Proceedings of the 26th Australian Computer-Human Interaction Conference on Designing Futures: the Future of Design*, pp. 426–429, ACM, 2014.

[43] J. P. Hansen, J. S. Agustin, and H. Skovsgaard, "Gaze interaction from bed," in *Proceedings of the 1st Conference on Novel Gaze-Controlled Applications*, NGCA '11, (New York, NY, USA), pp. 11:1–11:4, ACM, 2011.

[44] c. Çığ and T. M. Sezgin, "Gaze-based virtual task predictor," in *Proceedings of the 7th Workshop on Eye Gaze in Intelligent Human Machine Interaction: Eye-Gaze & Multimodality*, GazeIn '14, (New York, NY, USA), pp. 9–14, ACM, 2014.

[45] P. A. Orlov and A. Nikolay, "The effectiveness of gaze-contingent control in computer games," *Perception*, p. 0301006615594910, 2015.

[46] I. P. Howard, *Human visual orientation*. J. Wiley Chichester; New York, 1982.

[47] J. M. Sprague and T. H. Meikle Jr, "The role of the superior colliculus in visually guided behavior," *Experimental neurology*, vol. 11, no. 1, pp. 115–146, 1965.

[48] F. Crick and C. Koch, "Are we aware of neural activity in primary visual cortex?," *Nature*, vol. 375, no. 6527, pp. 121–123, 1995.

[49] C. W. Oyster and N. Haver, *The human eye: structure and function*, vol. 1. Sinauer Associates Sunderland, MA, 1999.

[50] D. A. Atchison, G. Smith, and G. Smith, "Optics of the human eye," 2000.

[51] A. T. Duchowski, D. H. House, J. Gestring, R. I. Wang, K. Krejtz, I. Krejtz, R. Mantiuk, and B. Bazyluk, "Reducing visual discomfort of 3d stereoscopic displays with gaze-contingent depth-of-field," in *Proceedings of the acm symposium on applied perception*, pp. 39–46, ACM, 2014.

[52] R. H. Carpenter, *Movements of the Eyes, 2nd Rev*. Pion Limited, 1988.

[53] J. E. Hoffman and B. Subramaniam, "The role of visual attention in saccadic eye movements," *Perception & psychophysics*, vol. 57, no. 6, pp. 787–795, 1995.

[54] D. E. Irwin, "Visual memory within and across fixations," in *Eye movements and visual cognition*, pp. 146–165, Springer, 1992.

[55] A. Kar and P. Corcoran, "A review and analysis of eye-gaze estimation systems, algorithms and performance evaluation methods in consumer platforms," *IEEE Access*, vol. 5, pp. 16495–16519, 2017.

[56] D. O. Harrington, *The visual fields: A textbook and atlas of clinical perimetry*. Mosby St. Louis, 1956.

[57] E. Kasneci, G. Kasneci, T. C. Kübler, and W. Rosenstiel, "Online recognition of fixations, saccades, and smooth pursuits for automated analysis of traffic hazard perception," in *Artificial neural networks*, pp. 411–434, Springer, 2015.

[58] B. Cohen, V. Matsuo, and T. Raphan, "Quantitative analysis of the velocity characteristics of optokinetic nystagmus and optokinetic after-nystagmus," *The Journal of physiology*, vol. 270, no. 2, pp. 321–344, 1977.

[59] G. Fitzgerald and C. Hallpike, "Studies in human vestibular function: I. observations on the directional preponderance ("nystagmusbereitschaft") of caloric nystagmus resulting from cerebral lesions," *Brain*, vol. 65, no. 2, pp. 115–137, 1942.

[60] D. Kahneman, *Attention and effort*, vol. 1063. Citeseer, 1973.

[61] J. Beatty and B. Lucero-Wagoner, "The pupillary system, handbook of psychophysiology, cacioppo, tassinary & berntson," 2000.

[62] H. Drewes and A. Schmidt, "Interacting with the computer using gaze gestures," in *IFIP Conference on Human-Computer Interaction*, pp. 475–488, Springer, 2007.

[63] M. Adjouadi, A. Sesin, M. Ayala, and M. Cabrerizo, *Remote eye gaze tracking system as a computer interface for persons with severe motor disability*. Springer, 2004.

[64] P. Biswas and P. Langdon, "A new interaction technique involving eye gaze tracker and scanning system," in *Proceedings of the 2013 Conference on Eye Tracking South Africa*, ETSA '13, (New York, NY, USA), pp. 67–70, ACM, 2013.

[65] R. J. Jacob, "The use of eye movements in human-computer interaction techniques: what you look at is what you get," *ACM Transactions on Information Systems (TOIS)*, vol. 9, no. 2, pp. 152–169, 1991.

[66] H. Heikkilä, "Eyesketch: A drawing application for gaze control," in *Proceedings of the 2013 Conference on Eye Tracking South Africa*, ETSA '13, (New York, NY, USA), pp. 71–74, ACM, 2013.

[67] L. E. Sibert and R. J. Jacob, "Evaluation of eye gaze interaction," in *Proceedings of the SIGCHI conference on Human Factors in Computing Systems*, pp. 281–288, ACM, 2000.

[68] C. Lankford, "Effective eye-gaze input into windows," in *Proceedings of the 2000 Symposium on Eye Tracking Research & Applications*, ETRA '00, (New York, NY, USA), pp. 23–27, ACM, 2000.

[69] D. Fono and R. Vertegaal, "Eyewindows: Evaluation of eye-controlled zooming windows for focus selection," in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '05, (New York, NY, USA), pp. 151–160, ACM, 2005.

[70] M. Porta, A. Ravarelli, and G. Spagnoli, "cecursor, a contextual eye cursor for general pointing in windows environments," in *Proceedings of the 2010 Symposium on Eye-Tracking Research & Applications*, ETRA '10, (New York, NY, USA), pp. 331–337, ACM, 2010.

[71] M. Kumar, A. Paepcke, and T. Winograd, "Eyepoint: Practical pointing and selection using gaze and keyboard," in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '07, (New York, NY, USA), pp. 421–430, ACM, 2007.

[72] V. Surakka, M. Illi, and P. Isokoski, "Gazing and frowning as a new human–computer interaction technique," *ACM Trans. Appl. Percept.*, vol. 1, pp. 40–56, July 2004.

[73] P. Majaranta, I. S. MacKenzie, A. Aula, and K.-J. Räihä, "Auditory and visual feedback during eye typing," in *CHI '03 Extended Abstracts on Human Factors in Computing Systems*, CHI EA '03, (New York, NY, USA), pp. 766–767, ACM, 2003.

[74] P. Majaranta, U.-K. Ahola, and O. Špakov, "Fast gaze typing with an adjustable dwell time," in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '09, (New York, NY, USA), pp. 357–360, ACM, 2009.

[75] J. P. Hansen, K. Tørning, A. S. Johansen, K. Itoh, and H. Aoki, "Gaze typing compared with input by head and hand," in *Proceedings of the 2004 Symposium on Eye Tracking Research & Applications*, ETRA '04, (New York, NY, USA), pp. 131–138, ACM, 2004.

[76] J. P. Hansen, A. S. Johansen, D. W. Hansen, K. Itoh, and S. Mashino, "Command Without a Click : Dwell Time Typing by Mouse and Gaze Selections," *Proceedings of Human-Computer Interaction – INTERACT´03*, no. c, pp. 121–128, 2003.

[77] M. Khamis, F. Alt, M. Hassib, E. von Zezschwitz, R. Hasholzner, and A. Bulling, "Gazetouchpass: Multimodal authentication using gaze and touch on mobile devices," in *Proceedings of the 2016 CHI Conference Extended Abstracts on Human Factors in Computing Systems*, CHI EA '16, (New York, NY, USA), pp. 2156–2164, ACM, 2016.

[78] A. De Luca, M. Denzel, and H. Hussmann, "Look into my eyes!: Can you guess my password?," in *Proceedings of the 5th Symposium on Usable Privacy and Security*, SOUPS '09, (New York, NY, USA), pp. 7:1–7:12, ACM, 2009.

[79] D. H. Cymek, A. C. Venjakob, S. Ruff, O. H.-M. Lutz, S. Hofmann, and M. Roetting, "Entering pin codes by smooth pursuit eye movements," *Journal of Eye Movement Research*, vol. 7, no. 4, 2014.

[80] M. Vidal, A. Bulling, and H. Gellersen, "Pursuits: Spontaneous interaction with displays based on smooth pursuit eye movement and moving targets," in *Proceedings of the 2013 ACM International Joint Conference on Pervasive and Ubiquitous Computing*, UbiComp '13, (New York, NY, USA), pp. 439–448, ACM, 2013.

[81] J. P. Hansen, A. Alapetite, M. Thomsen, Z. Wang, K. Minakata, and G. Zhang, "Head and gaze control of a telepresence robot with an hmd," in *Proceedings of the 2018 ACM Symposium on Eye Tracking Research & Applications*, ETRA '18, (New York, NY, USA), pp. 82:1–82:3, ACM, 2018.

[82] N. Davanzo, P. Dondi, M. Mosconi, and M. Porta, "Playing music with the eyes through an isomorphic interface," in *Proceedings of the Workshop on Communication by Gaze Interaction*, p. 5, ACM, 2018.

[83] A. Patney, M. Salvi, J. Kim, A. Kaplanyan, C. Wyman, N. Benty, D. Luebke, and A. Lefohn, "Towards foveated rendering for gaze-tracked virtual reality," *ACM Transactions on Graph-*

ics (TOG), vol. 35, no. 6, p. 179, 2016.

[84] Y. S. Pai, B. Tag, B. Outram, N. Vontin, K. Sugiura, and K. Kunze, "Gazesim: simulating foveated rendering using depth in eye gaze for vr," in *ACM SIGGRAPH 2016 Posters*, p. 75, ACM, 2016.

[85] J. C. Mateo, J. San Agustin, and J. P. Hansen, "Gaze beats mouse: Hands-free selection by combining gaze and emg," in *CHI '08 Extended Abstracts on Human Factors in Computing Systems*, CHI EA '08, (New York, NY, USA), pp. 3039–3044, ACM, 2008.

[86] V. Rajanna and T. Hammond, "Gawschi: Gaze-augmented, wearable-supplemented computer-human interaction," in *Proceedings of the Ninth Biennial ACM Symposium on Eye Tracking Research & Applications*, ETRA '16, (New York, NY, USA), pp. 233–236, ACM, 2016.

[87] S. Zhai, C. Morimoto, and S. Ihde, "Manual and gaze input cascaded (magic) pointing," in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '99, (New York, NY, USA), pp. 246–253, ACM, 1999.

[88] D. Fono and R. Vertegaal, "Eyewindows: evaluation of eye-controlled zooming windows for focus selection," in *Proceedings of the SIGCHI conference on Human factors in computing systems*, pp. 151–160, ACM, 2005.

[89] T. Jenkin, J. McGeachie, D. Fono, and R. Vertegaal, "eyeview: Focus+ context views for large group video conferences," in *CHI'05 extended abstracts on Human factors in computing systems*, pp. 1497–1500, ACM, 2005.

[90] J. Ten Kate, E. E. Frietman, W. Willems, B. T. H. Romeny, and E. Tenkink, "Eye-switch controlled communication aids," in *Proceedings of the 12th International Conference on Medical & Biological Engineering*, pp. 19–20, 1979.

[91] M. Yamada and T. Fukuda, "Eye word processor (ewp) and peripheral controller for the als patient," *IEE Proceedings A (Physical Science, Measurement and Instrumentation, Management and Education, Reviews)*, vol. 134, no. 4, pp. 328–330, 1987.

[92] M. B. Friedman, G. J. Kiliany, and M. R. Dzmura, "Eye-tracker communication system," Mar. 3 1987. US Patent 4,648,052.

[93] T. E. Hutchinson, K. P. White, W. N. Martin, K. C. Reichert, and L. A. Frey, "Human-computer interaction using eye-gaze input," *IEEE Transactions on systems, man, and cybernetics*, vol. 19, no. 6, pp. 1527–1534, 1989.

[94] V. Rajanna, "Gaze typing through foot-operated wearable device," in *Proceedings of the 18th International ACM SIGACCESS Conference on Computers and Accessibility*, ASSETS '16, (New York, NY, USA), pp. 345–346, ACM, 2016.

[95] M. Su, C. Yeh, S. Lin, P. Wang, and S. Hou, "An implementation of an eye-blink-based communication aid for people with severe disabilities," in *Audio, Language and Image Processing, 2008. ICALIP 2008. International Conference on*, pp. 351–356, IEEE, 2008.

[96] T. Ohno, N. Mukawa, and S. Kawato, "Just blink your eyes: A head-free gaze tracking system," in *CHI'03 extended abstracts on Human factors in computing systems*, pp. 950–957, ACM, 2003.

[97] X. A. Zhao, E. D. Guestrin, D. Sayenko, T. Simpson, M. Gauthier, and M. R. Popovic, "Typing with eye-gaze and tooth-clicks," in *Proceedings of the Symposium on Eye Tracking Research and Applications*, pp. 341–344, ACM, 2012.

[98] D. Pedrosa, M. D. G. Pimentel, A. Wright, and K. N. Truong, "Filteryedping: Design Challenges and User Performance of Dwell-Free Eye Typing," *ACM Transactions on Accessible Computing*, vol. 6, pp. 1–37, mar 2015.

[99] I. S. MacKenzie and X. Zhang, "Eye typing using word and letter prediction and a fixation algorithm," in *Proceedings of the 2008 Symposium on Eye Tracking Research & Applications*, ETRA '08, (New York, NY, USA), pp. 55–58, ACM, 2008.

[100] V. Rajanna, A. H. Malla, R. A. Bhagat, and T. Hammond, "Dygazepass: A gaze gesture-based dynamic authentication system to counter shoulder surfing and video analysis at-

tacks," in *2018 IEEE 4th International Conference on Identity, Security, and Behavior Analysis (ISBA)*, pp. 1–8, Jan 2018.

[101] K. Bagchi and G. Udo, "An analysis of the growth of computer and internet security breaches," *Communications of the Association for Information Systems*, vol. 12, no. 1, p. 46, 2003.

[102] A. Bulling, F. Alt, and A. Schmidt, "Increasing the security of gaze-based cued-recall graphical passwords using saliency masks," in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '12, (New York, NY, USA), pp. 3011–3020, ACM, 2012.

[103] A. De Luca, R. Weiss, and H. Drewes, "Evaluation of eye-gaze interaction methods for security enhanced pin-entry," in *Proceedings of the 19th Australasian Conference on Computer-Human Interaction: Entertaining User Interfaces*, OZCHI '07, (New York, NY, USA), pp. 199–202, ACM, 2007.

[104] D. S. Best and A. T. Duchowski, "A rotary dial for gaze-based pin entry," in *Proceedings of the Ninth Biennial ACM Symposium on Eye Tracking Research & Applications*, ETRA '16, (New York, NY, USA), pp. 69–76, ACM, 2016.

[105] R. J. Jacob, "Eye movement-based human-computer interaction techniques: Toward non-command interfaces," *Advances in human-computer interaction*, vol. 4, pp. 151–190, 1993.

[106] H. Drewes and A. Schmidt, "Interacting with the Computer Using Gaze Gestures," in *Human-Computer Interaction – INTERACT 2007*, pp. 475–488, Berlin, Heidelberg: Springer Berlin Heidelberg, 2007.

[107] J. O. Wobbrock, J. Rubinstein, M. W. Sawyer, and A. T. Duchowski, "Longitudinal evaluation of discrete consecutive gaze gestures for text entry," in *ETRA '08*, (New York, New York, USA), p. 11, ACM Press, mar 2008.

[108] "Wearable EOG Goggles: Eye-Based Interaction in Everyday Environments," *Wear*, pp. 3259–3264, 2009.

[109] B. Bauman, R. Gunhouse, A. Jones, W. Da Silva, S. Sharar, V. Rajanna, J. Cherian, J. I. Koh, and T. Hammond, "Visualeyeze: A web-based solution for receiving feedback on artworks through eye-tracking," in *IUI'18 23rd International Conference on Intelligent User Interfaces Tokyo, Japan. Web Intelligence and Interaction (WII) workshop*, (Tokyo, Japan), IUI, March 2018. `http://ceur-ws.org/Vol-2068/wii4.pdf`.

[110] F. Alamudun, H.-J. Yoon, K. B. Hudson, G. Morin-Ducote, T. Hammond, and G. D. Tourassi, "Fractal analysis of visual search activity for mass detection during mammographic screening," *Medical Physics*, vol. 44, no. 3, pp. 832–846.

[111] T. Hammond, G. Tourassi, H.-J. Yoon, and F. T. Alamudun, "Geometry and gesture-based features from saccadic eye-movement as a biometric in radiology," tech. rep., Oak Ridge National Lab.(ORNL), Oak Ridge, TN (United States), 2017.

[112] P. Majaranta and K.-J. Räihä, "Twenty years of eye typing: Systems and design issues," in *Proceedings of the 2002 Symposium on Eye Tracking Research & Applications*, ETRA '02, (New York, NY, USA), pp. 15–22, ACM, 2002.

[113] S. K. Kane, J. O. Wobbrock, and I. E. Smith, "Getting off the treadmill: Evaluating walking user interfaces for mobile devices in public spaces," in *Proceedings of the 10th International Conference on Human Computer Interaction with Mobile Devices and Services*, MobileHCI '08, (New York, NY, USA), pp. 109–118, ACM, 2008.

[114] B. Velichkovsky, A. Sprenger, and P. Unema, "Towards gaze-mediated interaction: Collecting solutions of the "midas touch problem"," in *Human-Computer Interaction INTERACT'97*, pp. 509–516, Springer, 1997.

[115] G. Pearson and M. Weiser, "Of moles and men: the design of foot controls for workstations," in *ACM SIGCHI Bulletin*, vol. 17, pp. 333–339, ACM, 1986.

[116] T. Pakkanen and R. Raisamo, "Appropriateness of foot interaction for non-accurate spatial tasks," in *CHI'04 extended abstracts on Human factors in computing systems*, pp. 1123–1126, ACM, 2004.

[117] E. Velloso, D. Schmidt, J. Alexander, H. Gellersen, and A. Bulling, "The feet in human–computer interaction: A survey of foot-based interaction," *ACM Computing Surveys (CSUR)*, vol. 48, no. 2, p. 21, 2015.

[118] F. Göbel, K. Klamka, A. Siegel, S. Vogt, S. Stellmach, and R. Dachselt, "Gaze-supported foot interaction in zoomable information spaces (interactivity)," in *Proceedings of the Conference on Human Factors in Computing Systems - Extended Abstracts*, ACM, 4 2013.

[119] K. Pfeuffer, J. Alexander, M. K. Chong, and H. Gellersen, "Gaze-touch: combining gaze with multi-touch for interaction on the same surface," in *Proceedings of the 27th annual ACM symposium on User interface software and technology*, pp. 509–518, ACM, 2014.

[120] J. Turner, J. Alexander, A. Bulling, D. Schmidt, and H. Gellersen, "Eye pull, eye push: Moving objects between large screens and personal devices with gaze and touch," in *Human-Computer Interaction–INTERACT 2013*, pp. 170–186, Springer, 2013.

[121] J. Turner, A. Bulling, J. Alexander, and H. Gellersen, "Cross-device gaze-supported point-to-point content transfer," in *Proceedings of the Symposium on Eye Tracking Research and Applications*, pp. 19–26, ACM, 2014.

[122] T. Cha and S. Maier, "Eye gaze assisted human-computer interaction in a hand gesture controlled multi-display environment," in *Proceedings of the 4th Workshop on Eye Gaze in Intelligent Human Machine Interaction*, Gaze-In '12, (New York, NY, USA), pp. 13:1–13:3, ACM, 2012.

[123] M. Elepfandt and M. Grund, "Move it there, or not?: The design of voice commands for gaze with speech," in *Proceedings of the 4th Workshop on Eye Gaze in Intelligent Human Machine Interaction*, Gaze-In '12, (New York, NY, USA), pp. 12:1–12:3, ACM, 2012.

[124] T. Beelders and P. Blignaut, *The Usability of Speech and Eye Gaze as a Multimodal Interface for a Word Processor*. INTECH Open Access Publisher, 2011.

[125] J. C. Mateo, J. San Agustin, and J. P. Hansen, "Gaze beats mouse: Hands-free selection by combining gaze and emg," in *CHI '08 Extended Abstracts on Human Factors in Computing Systems*, CHI EA '08, (New York, NY, USA), pp. 3039–3044, ACM, 2008.

[126] B. Knapp and B. Hall, "High performance concerns for the trackwolf system (ari research note 91-14)," *Alexandria,(VA): ARI*, 1990.

[127] P. M. Fitts, "The information capacity of the human motor system in controlling the amplitude of movement.," *Journal of Experimental Psychology*, vol. 47, no. 6, pp. 381–391, 1954.

[128] F. Law and I. S. Mackenzie, "Fitts' Law," *Handbook of human-computer interaction*, vol. 1, pp. 349–370, 2018.

[129] I. S. MacKenzie, "Fitts' law as a research and design tool in human-computer interaction," *Human-computer interaction*, vol. 7, no. 1, pp. 91–139, 1992.

[130] R. W. Soukoreff and I. S. MacKenzie, "Towards a standard for pointing device evaluation, perspectives on 27 years of fitts' law research in hci," *International Journal of Human-Computer Studies*, vol. 61, no. 6, pp. 751 – 789, 2004. Fitts' law 50 years later: applications and contributions from human-computer interaction.

[131] X. Zhang and I. S. MacKenzie, *Evaluating Eye Tracking with ISO 9241 - Part 9*, pp. 779–788. Berlin, Heidelberg: Springer Berlin Heidelberg, 2007.

[132] D. Miniotas, "Application of fitts' law to eye gaze interaction," in *CHI '00 Extended Abstracts on Human Factors in Computing Systems*, CHI EA '00, (New York, NY, USA), pp. 339–340, ACM, 2000.

[133] R. Vertegaal, "A fitts law comparison of eye tracking and manual input in the selection of visual targets," in *Proceedings of the 10th International Conference on Multimodal Interfaces*, ICMI '08, (New York, NY, USA), pp. 241–248, ACM, 2008.

[134] D. Miniotas, O. Špakov, I. Tugoy, and I. S. MacKenzie, "Speech-augmented eye gaze interaction with small closely spaced targets," in *Proceedings of the 2006 symposium on Eye*

tracking research & applications - ETRA '06, (New York, New York, USA), p. 67, ACM Press, 2006.

[135] C. Ware and H. H. Mikaelian, "An evaluation of an eye tracker as a device for computer input2," in *Proceedings of the SIGCHI/GI Conference on Human Factors in Computing Systems and Graphics Interface*, CHI '87, (New York, NY, USA), pp. 183–188, ACM, 1987.

[136] S. Zhai, C. Morimoto, and S. Ihde, "Manual and gaze input cascaded (magic) pointing," in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '99, (New York, NY, USA), pp. 246–253, ACM, 1999.

[137] X. Zhang and I. S. MacKenzie, *Evaluating Eye Tracking with ISO 9241 - Part 9*, pp. 779–788. Berlin, Heidelberg: Springer Berlin Heidelberg, 2007.

[138] T. R. Beelders and P. J. Blignaut, "Using eye gaze and speech to simulate a pointing device," in *Proceedings of the Symposium on Eye Tracking Research and Applications*, ETRA '12, (New York, NY, USA), pp. 349–352, ACM, 2012.

[139] V. Surakka, M. Illi, and P. Isokoski, "Gazing and frowning as a new human–computer interaction technique," *ACM Transactions on Applied Perception (TAP)*, vol. 1, no. 1, pp. 40–56, 2004.

[140] J. San Agustin, J. C. Mateo, J. P. Hansen, and A. Villanueva, "Evaluation of the potential of gaze input for game interaction.," *PsychNology Journal*, vol. 7, no. 2, pp. 213–236, 2009.

[141] V. Rajanna and T. Hammond, "Gawschi: Gaze-augmented, wearable-supplemented computer-human interaction," in *Proceedings of the Ninth Biennial ACM Symposium on Eye Tracking Research and Applications*, ETRA '16, (New York, NY, USA), pp. 233–236, ACM, 2016.

[142] D. W. Hansen, H. H. T. Skovsgaard, J. P. Hansen, and E. Møllenbach, "Noise tolerant selection by gaze-controlled pan and zoom in 3d," in *Proceedings of the 2008 Symposium on Eye*

Tracking Research & Applications, ETRA '08, (New York, NY, USA), pp. 205–212, ACM, 2008.

[143] S. Stellmach and R. Dachselt, "Look & touch: Gaze-supported target acquisition," in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '12, (New York, NY, USA), pp. 2981–2990, ACM, 2012.

[144] J. P. Hansen, H. Lund, F. Biermann, E. Møllenbach, S. Sztuk, and J. S. Agustin, "Wrist-worn pervasive gaze interaction," in *Proceedings of the Ninth Biennial ACM Symposium on Eye Tracking Research & Applications*, pp. 57–64, ACM, 2016.

[145] J. San Agustin, J. P. Hansen, and M. Tall, "Gaze-based interaction with public displays using off-the-shelf components," in *Proceedings of the 12th ACM International Conference Adjunct Papers on Ubiquitous Computing - Adjunct*, UbiComp '10 Adjunct, (New York, NY, USA), pp. 377–378, ACM, 2010.

[146] M. Vidal, A. Bulling, and H. Gellersen, "Pursuits: spontaneous interaction with displays based on smooth pursuit eye movement and moving targets," in *Proceedings of the 2013 ACM international joint conference on Pervasive and ubiquitous computing*, pp. 439–448, ACM, 2013.

[147] H. Qian, R. Kuber, and A. Sears, "Tactile notifications for ambulatory users," in *CHI '13 Extended Abstracts on Human Factors in Computing Systems*, CHI EA '13, (New York, NY, USA), pp. 1569–1574, ACM, 2013.

[148] B. Schildbach and E. Rukzio, "Investigating selection and reading performance on a mobile phone while walking," in *Proceedings of the 12th International Conference on Human Computer Interaction with Mobile Devices and Services*, MobileHCI '10, (New York, NY, USA), pp. 93–102, ACM, 2010.

[149] J. O. Wobbrock, J. Rubinstein, M. W. Sawyer, and A. T. Duchowski, "Longitudinal evaluation of discrete consecutive gaze gestures for text entry," in *Proceedings of the 2008 Sym-*

posium on Eye Tracking Research & Applications, ETRA '08, (New York, NY, USA), pp. 11–18, ACM, 2008.

[150] D. W. Hansen, H. H. T. Skovsgaard, J. P. Hansen, and E. Møllenbach, "Noise tolerant selection by gaze-controlled pan and zoom in 3d," in *Proceedings of the 2008 Symposium on Eye Tracking Research & Applications*, ETRA '08, (New York, NY, USA), pp. 205–212, ACM, 2008.

[151] P. Majaranta and K.-J. Räihä, "Text entry by gaze: Utilizing eye-tracking," *Text entry systems: Mobility, accessibility, universality*, pp. 175–187, 2007.

[152] I. S. MacKenzie and X. Zhang, "Eye typing using word and letter prediction and a fixation algorithm," in *Proceedings of the 2008 symposium on Eye tracking research & applications - ETRA '08*, (New York, New York, USA), p. 55, ACM Press, mar 2008.

[153] L. A. Frey, K. White, and T. Hutchison, "Eye-gaze word processing," *IEEE Transactions on systems, Man, and Cybernetics*, vol. 20, no. 4, pp. 944–950, 1990.

[154] P. Isokoski, "Text input methods for eye trackers using off-screen targets," in *Proceedings of the 2000 symposium on Eye tracking research & applications*, pp. 15–21, ACM, 2000.

[155] M. H. Urbina and A. Huckauf, "Alternatives to single character entry and dwell time selection on eye typing," in *Proceedings of the 2010 Symposium on Eye-Tracking Research & Applications - ETRA '10*, (New York, New York, USA), p. 315, ACM Press, mar 2010.

[156] H. H. Koester and S. P. Levine, "Modeling the speed of text entry with a word prediction interface," *IEEE transactions on rehabilitation engineering*, vol. 2, no. 3, pp. 177–187, 1994.

[157] J. P. Hansen, K. Tørning, A. S. Johansen, K. Itoh, and H. Aoki, "Gaze typing compared with input by head and hand," in *Proceedings of the Eye tracking research & applications symposium on Eye tracking research & applications - ETRA'2004*, (New York, New York, USA), pp. 131–138, ACM Press, mar 2004.

[158] K. Klamka, A. Siegel, S. Vogt, F. Göbel, S. Stellmach, and R. Dachselt, "Look and pedal: Hands-free navigation in zoomable information spaces through gaze-supported foot input," in *Proceedings of the 2015 ACM on International Conference on Multimodal Interaction*, ICMI '15, (New York, NY, USA), pp. 123–130, ACM, 2015.

[159] T. R. Beelders and P. J. Blignaut, "Measuring the performance of gaze and speech for text input," in *Proceedings of the Symposium on Eye Tracking Research and Applications - ETRA '12*, (New York, New York, USA), p. 337, ACM Press, mar 2012.

[160] A. Huckauf and M. Urbina, "Gazing with pEYE," in *Proceedings of the 4th symposium on Applied perception in graphics and visualization - APGV '07*, (New York, New York, USA), p. 141, ACM Press, jul 2007.

[161] T. Hammond and R. Davis, "Ladder, a sketching language for user interface developers," *Computers and Graphics*, vol. 29, no. 4, pp. 518 – 532, 2005.

[162] N. Bee and E. André, "Writing with your eye: A dwell time free writing system adapted to the nature of human eye gaze," in *Perception in Multimodal Dialogue Systems*, pp. 111–122, Springer, 2008.

[163] M. Porta and M. Turina, "Eye-s: A full-screen input modality for pure eye-based communication," in *Proceedings of the 2008 Symposium on Eye Tracking Research & Applications*, ETRA '08, (New York, NY, USA), pp. 27–34, ACM, 2008.

[164] D. J. Ward, A. F. Blackwell, and D. J. C. MacKay, "Dasher—a data entry interface using continuous gestures and language models," in *Proceedings of the 13th Annual ACM Symposium on User Interface Software and Technology*, UIST '00, (New York, NY, USA), pp. 129–137, ACM, 2000.

[165] K. Grauman, M. Betke, J. Lombardi, J. Gips, and G. R. Bradski, "Communication via eye blinks and eyebrow raises: Video-based human-computer interfaces," *Universal Access in the Information Society*, vol. 2, no. 4, pp. 359–373, 2003.

[166] J. ten Kate, E. Frietman, F. Stoel, and W. Willems, "Eye-controlled communication aids," *Medical progress through technology*, vol. 8, no. 1, p. 1—21, 1980.

[167] M. Fejtová, J. Fejt, and O. Štěpánková, "Eye as an actuator," in *Computers Helping People with Special Needs*, pp. 954–961, Springer, 2006.

[168] E. Velloso, D. Schmidt, J. Alexander, H. Gellersen, and A. Bulling, "The feet in human–computer interaction: A survey of foot-based interaction," *ACM Comput. Surv.*, vol. 48, no. 2, pp. 21:1–21:35, 2015.

[169] V. D. Rajanna, "Gaze and foot input: Toward a rich and assistive interaction modality," in *Companion Publication of the 21st International Conference on Intelligent User Interfaces*, IUI '16 Companion, (New York, NY, USA), pp. 126–129, ACM, 2016.

[170] I. S. MacKenzie and R. W. Soukoreff, "Phrase sets for evaluating text entry techniques," in *CHI'03 extended abstracts on Human factors in computing systems*, pp. 754–755, ACM, 2003.

[171] J. P. Hansen, D. W. Hansen, and A. S. Johansen, "Bringing Gaze-based Interaction Back to Basics," *Universal Access In HCI*, pp. 325–328, 2001.

[172] S. Chiasson, E. Stobert, A. Forget, R. Biddle, and P. C. Van Oorschot, "Persuasive cued click-points: Design, implementation, and evaluation of a knowledge-based authentication mechanism," *IEEE Transactions on Dependable and Secure Computing*, vol. 9, no. 2, pp. 222–235, 2012.

[173] M. D. H. Abdullah, A. H. Abdullah, N. Ithnin, and H. K. Mammi, "Towards identifying usability and security features of graphical password in knowledge based authentication technique," in *Modeling & Simulation, 2008. AICMS 08. Second Asia International Conference on*, pp. 396–403, IEEE, 2008.

[174] R. K. Rowe, K. A. Nixon, and S. P. Corcoran, "Multispectral fingerprint biometrics," in *Information Assurance Workshop, 2005. IAW'05. Proceedings from the Sixth Annual IEEE SMC*, pp. 14–20, IEEE, 2005.

[175] D. Bonalle and G. Salow, "Method and system for fingerprint biometrics on a smartcard," Jan. 5 2006. US Patent App. 10/710,310.

[176] S. Chiasson, R. Biddle, and P. C. van Oorschot, "A second look at the usability of click-based graphical passwords," in *Proceedings of the 3rd symposium on Usable privacy and security*, pp. 1–12, ACM, 2007.

[177] R. Biddle, S. Chiasson, and P. C. Van Oorschot, "Graphical passwords: Learning from the first twelve years," *ACM Computing Surveys (CSUR)*, vol. 44, no. 4, p. 19, 2012.

[178] A. Rattani, D. R. Kisku, M. Bicego, and M. Tistarelli, "Feature level fusion of face and fingerprint biometrics," in *Biometrics: Theory, Applications, and Systems, 2007. BTAS 2007. First IEEE International Conference on*, pp. 1–6, IEEE, 2007.

[179] C. Bo, L. Zhang, X.-Y. Li, Q. Huang, and Y. Wang, "Silentsense: silent user identification via touch and movement behavioral biometrics," in *Proceedings of the 19th annual international conference on Mobile computing & networking*, pp. 187–190, ACM, 2013.

[180] R. M. Bolle, C. Dorai, and N. K. Ratha, "System and method for distortion characterization in fingerprint and palm-print image sequences and using this distortion as a behavioral biometrics," May 30 2006. US Patent 7,054,470.

[181] J. Daugman, "How iris recognition works," in *The essential guide to image processing*, pp. 715–739, Elsevier, 2009.

[182] R. P. Wildes, "Iris recognition: an emerging biometric technology," *Proceedings of the IEEE*, vol. 85, no. 9, pp. 1348–1363, 1997.

[183] J. Han and B. Bhanu, "Individual recognition using gait energy image," *IEEE Transactions on Pattern Analysis & Machine Intelligence*, no. 2, pp. 316–322, 2006.

[184] L. Wang, T. Tan, H. Ning, and W. Hu, "Silhouette analysis-based gait recognition for human identification," *IEEE transactions on pattern analysis and machine intelligence*, vol. 25, no. 12, pp. 1505–1518, 2003.

[185] B. Dosono, J. Hayes, and Y. Wang, "Toward accessible authentication: Learning from people with visual impairments," *IEEE Internet Computing*, vol. 22, pp. 62–70, Mar 2018.

[186] K. Helkala, "Disabilities and authentication methods: Usability and security," in *2012 Seventh International Conference on Availability, Reliability and Security*, pp. 327–334, Aug 2012.

[187] J. D'Arcy and J. Feng, "Investigating security-related behaviors among computer users with motor impairments," *Poster abstracts of SOUPS*, vol. 6, 2006.

[188] M. Eiband, M. Khamis, E. von Zezschwitz, H. Hussmann, and F. Alt, "Understanding shoulder surfing in the wild: Stories from users and observers," in *Proceedings of the 35th Annual ACM Conference on Human Factors in Computing Systems*, CHI '17, (New York, NY, USA), ACM, 2017.

[189] H. Zafar and J. G. Clark, "Current state of information security research in is," *Communications of the Association for Information Systems*, vol. 24, no. 1, p. 34, 2009.

[190] J. Long, *No tech hacking: A guide to social engineering, dumpster diving, and shoulder surfing*. Syngress, 2011.

[191] J. O. Adeoti, "Automated teller machine (atm) frauds in nigeria: The way out," *Journal of Social Sciences*, vol. 27, no. 1, pp. 53–58, 2011.

[192] P. Institute, "Global Visual Hacking Experimental Study: Analysis," 2016.

[193] A. H. Lashkari, S. Farmand, O. B. Zakaria, and R. Saleh, "Shoulder Surfing Attack in Graphical Password Authentication," *International Journal of Computer Science and Information Security*, vol. 6, no. 2, pp. 145–154, 2009.

[194] S. Wiedenbeck, J. Waters, L. Sobrado, and J.-C. Birget, "Design and evaluation of a shoulder-surfing resistant graphical password scheme," in *Proceedings of the Working Conference on Advanced Visual Interfaces*, AVI '06, (New York, NY, USA), pp. 177–184, ACM, 2006.

[195] J. Schiff, M. Meingast, D. K. Mulligan, S. Sastry, and K. Goldberg, *Respectful Cameras: Detecting Visual Markers in Real-Time to Address Privacy Concerns*, pp. 65–89. London: Springer London, 2009.

[196] Y. Wang, H. Xia, Y. Yao, and Y. Huang, "Flying eyes and hidden controllers: A qualitative study of people's privacy perceptions of civilian drones in the us," *Proceedings on Privacy Enhancing Technologies*, vol. 2016, no. 3, pp. 172–190, 2016.

[197] N. H. Zakaria, D. Griffiths, S. Brostoff, and J. Yan, "Shoulder surfing defence for recall-based graphical passwords," in *Proceedings of the Seventh Symposium on Usable Privacy and Security - SOUPS '11*, (New York, New York, USA), p. 1, ACM Press, 2011.

[198] M. A. S. Gokhale and V. S. Waghmare, "The Shoulder Surfing Resistant Graphical Password Authentication Technique," in *Procedia Computer Science*, vol. 79, pp. 490–498, 2016.

[199] F. Alt, S. Schneegass, A. S. Shirazi, M. Hassib, and A. Bulling, "Graphical Passwords in the Wild," in *Proceedings of the 17th International Conference on Human-Computer Interaction with Mobile Devices and Services - MobileHCI '15*, (New York, New York, USA), pp. 316–322, ACM Press, 2015.

[200] S. Wiedenbeck, J. Waters, J.-C. Birget, A. Brodskiy, and N. Memon, "Passpoints: Design and longitudinal evaluation of a graphical password system," *International journal of human-computer studies*, vol. 63, no. 1, pp. 102–127, 2005.

[201] V. Roth, K. Richter, and R. Freidinger, "A PIN-entry method resilient against shoulder surfing," in *Proceedings of the 11th ACM conference on Computer and communications security - CCS '04*, (New York, New York, USA), p. 236, ACM Press, oct 2004.

[202] T. Kuribara, B. Shizuki, and J. Tanaka, "Vibrainput," in *Proceedings of the extended abstracts of the 32nd annual ACM conference on Human factors in computing systems - CHI EA '14*, (New York, New York, USA), pp. 2473–2478, ACM Press, apr 2014.

[203] S. Schneegass, F. Steimle, A. Bulling, F. Alt, and A. Schmidt, "Smudgesafe: Geometric image transformations for smudge-resistant user authentication," in *Proceedings of the 2014*

ACM International Joint Conference on Pervasive and Ubiquitous Computing, UbiComp '14, (New York, NY, USA), pp. 775–786, ACM, 2014.

[204] E. von Zezschwitz, A. De Luca, B. Brunkow, and H. Hussmann, "SwiPIN," in *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems - CHI '15*, (New York, New York, USA), pp. 1403–1406, ACM Press, 2015.

[205] A. De Luca, M. Harbach, E. von Zezschwitz, M.-E. Maurer, B. E. Slawik, H. Hussmann, M. Smith, A. De Luca, M. Harbach, E. von Zezschwitz, M.-E. Maurer, B. E. Slawik, H. Hussmann, and M. Smith, "Now you see me, now you don't," in *Proceedings of the 32nd annual ACM conference on Human factors in computing systems - CHI '14*, (New York, New York, USA), pp. 2937–2946, ACM Press, 2014.

[206] A. Forget, S. Chiasson, and R. Biddle, "Shoulder-surfing resistance with eye-gaze entry in cued-recall graphical passwords," in *Proceedings of the 28th international conference on Human factors in computing systems - CHI '10*, (New York, New York, USA), p. 1107, ACM Press, apr 2010.

[207] A. De Luca, K. Hertzschuch, and H. Hussmann, "ColorPIN," in *Proceedings of the 28th international conference on Human factors in computing systems - CHI '10*, (New York, New York, USA), p. 1103, ACM Press, apr 2010.

[208] W. Moncur and G. Leplâtre, "Pictures at the atm: Exploring the usability of multiple graphical passwords," in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '07, (New York, NY, USA), pp. 887–894, ACM, 2007.

[209] B. Hoanca and K. Mock, "Secure graphical password system for high traffic public areas," in *Proceedings of the 2006 Symposium on Eye Tracking Research &Amp; Applications*, ETRA '06, (New York, NY, USA), pp. 35–35, ACM, 2006.

[210] D. Davis, F. Monrose, and M. K. Reiter, "On user choice in graphical password schemes.," in *USENIX Security Symposium*, vol. 13, pp. 11–11, 2004.

[211] P. C. v. Oorschot and J. Thorpe, "On predictive models and user-drawn graphical passwords," *ACM Trans. Inf. Syst. Secur.*, vol. 10, pp. 5:1–5:33, Jan. 2008.

[212] S. Madigan, "Picture memory," *Imagery, memory and cognition*, pp. 65–89, 2014.

[213] G. H. Bower, M. B. Karlin, and A. Dueck, "Comprehension and memory for pictures," *Memory & Cognition*, vol. 3, no. 2, pp. 216–220, 1975.

[214] R. Eckmiller and E. Bauswein, "Smooth pursuit eye movements," *Progress in brain research*, vol. 64, pp. 313–323, 1986.

[215] J. O. Wobbrock, A. D. Wilson, and Y. Li, "Gestures without libraries, toolkits or training: A $1 recognizer for user interface prototypes," in *Proceedings of the 20th Annual ACM Symposium on User Interface Software and Technology*, UIST '07, (New York, NY, USA), pp. 159–168, ACM, 2007.

[216] D. Robinson, "The mechanics of human saccadic eye movement," *The Journal of physiology*, vol. 174, no. 2, p. 245, 1964.

[217] K. Holmqvist, M. Nyström, R. Andersson, R. Dewhurst, H. Jarodzka, and J. Van de Weijer, *Eye tracking: A comprehensive guide to methods and measures*. OUP Oxford, 2011.

[218] A. Poole and L. J. Ball, "Eye tracking in hci and usability research," *Encyclopedia of human computer interaction*, vol. 1, pp. 211–219, 2006.

[219] D. Hornung, *Colour: a workshop for artists and designers*. Laurence King Publishing, 2005.

[220] T. C. Greene, P. A. Bell, and W. N. Boyer, "Coloring the environment: Hue, arousal, and boredom," *Bulletin of the Psychonomic Society*, vol. 21, no. 4, pp. 253–254, 1983.

[221] M. A. Dzulkifli and M. F. Mustafar, "The influence of colour on memory performance: a review.," *The Malaysian journal of medical sciences : MJMS*, vol. 20, pp. 3–9, mar 2013.

[222] A. Leon-Garcia and A. Leon-Garcia, *Probability, statistics, and random processes for electrical engineering*. Pearson/Prentice Hall 3rd ed. Upper Saddle River, NJ, 2008.

[223] R. H. Bartels, J. C. Beatty, and B. A. Barsky, *An introduction to splines for use in computer graphics and geometric modeling*. Morgan Kaufmann, 1995.

[224] V. I. Levenshtein, "Binary codes capable of correcting deletions, insertions, and reversals," in *Soviet physics doklady*, vol. 10, pp. 707–710, 1966.

[225] S. G. Hart and L. E. Staveland, "Development of nasa-tlx (task load index): Results of empirical and theoretical research," *Advances in psychology*, vol. 52, pp. 139–183, 1988.

[226] J. P. Hansen, A. W. Andersen, and P. Roed, "Eye-gaze control of multimedia systems," in *Symbiosis of Human and Artifact* (Y. Anzai, K. Ogawa, and H. Mori, eds.), vol. 20 of *Advances in Human Factors/Ergonomics*, pp. 37 – 42, Elsevier, 1995.

[227] J. P. Hansen, A. S. Johansen, D. W. Hansen, K. Itoh, and S. Mashino, "Command without a click: Dwell time typing by mouse and gaze selections," in *Proceedings of Human-Computer Interaction–INTERACT*, pp. 121–128, 2003.

[228] C. Lankford, "Effective eye-gaze input into windows," in *Proceedings of the 2000 symposium on Eye tracking research & applications*, pp. 23–27, ACM, 2000.

[229] M. Kumar, A. Paepcke, and T. Winograd, "Eyepoint: practical pointing and selection using gaze and keyboard," in *Proceedings of the SIGCHI conference on Human factors in computing systems*, pp. 421–430, ACM, 2007.

[230] V. Rajanna, "Gaze typing through foot-operated wearable device," in *Proceedings of the 18th International ACM SIGACCESS Conference on Computers and Accessibility*, ASSETS '16, (New York, NY, USA), pp. 345–346, ACM, 2016.

[231] T. Hammond, G. Tourassi, H.-J. Yoon, and F. T. Alamudun, "Geometry and gesture-based features from saccadic eye-movement as a biometric in radiology," 7 2017.

[232] J. O. Wobbrock, A. D. Wilson, and Y. Li, "Gestures without libraries, toolkits or training: A $1 recognizer for user interface prototypes," in *Proceedings of the 20th Annual ACM Symposium on User Interface Software and Technology*, UIST '07, (New York, NY, USA), pp. 159–168, ACM, 2007.

[233] M. A. Friedl and C. E. Brodley, "Decision tree classification of land cover from remotely sensed data," *Remote sensing of environment*, vol. 61, no. 3, pp. 399–409, 1997.

[234] P. Majaranta, U.-K. Ahola, and O. Špakov, "Fast gaze typing with an adjustable dwell time," in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '09, (New York, NY, USA), pp. 357–360, ACM, 2009.