Off-the-Shelf Gaze Interaction

Javier San Agustin Lopez

September 2009

Preface

This dissertation documents parts of my PhD studies conducted at the IT University of Copenhagen in the period from August 2006 to September 2009. The work was developed under the supervision of associate professors John Paulin Hansen and Dan Witzner Hansen from the IT University of Copenhagen, and associate professor Arantxa Villanueva Larre from the Public University of Navarra. Most research was done at the IT University of Copenhagen and the Public University of Navarra, with the exception of a visit to the Tokyo Institute of Technology hosted by professor Kenji Itoh, and a visit to Wright State University hosted by professor John Flach.

Off-the-Shelf Gaze Interaction

Copyright 2009 by Javier San Agustin.

The IT University of Copenhagen.
ISSN:
ISBN:

"Destiny is not a matter of chance,
it is a matter of choice;
it is not a thing to be waited for,
it is a thing to be achieved."

William J. Bryan (1860-1925)

Abstract

People with severe motor-skill disabilities are often unable to use standard input devices such as a mouse or a keyboard to control a computer and they are, therefore, in strong need for alternative input devices. Gaze tracking offers them the possibility to use the movements of their eyes to interact with a computer, thereby making them more independent. A big effort has been put toward improving the robustness and accuracy of the technology, and many commercial systems are nowadays available in the market.

Despite the great improvements that gaze tracking systems have undergone in the last years, high prices have prevented gaze interaction from becoming mainstream. The use of specialized hardware, such as industrial cameras or infrared light sources, increases the accuracy of the systems, but also the price, which prevents many potential users from having access to the technology. Furthermore, the different components are often required to be placed in specific locations, or are built into the monitor, thus decreasing the flexibility of the setup.

Gaze tracking systems built from low-cost and off-the-shelf components have the potential to facilitate access to the technology and bring the prices down. Such systems are often more flexible, as the components can be placed in different locations, but also less robust, due to the lack of control over the hardware setup and the lower quality of the components compared to commercial systems.

The work developed for this thesis deals with some of the challenges introduced by the use of low-cost and off-the-shelf components for gaze interaction. The main contributions are:

- Development and performance evaluation of the ITU Gaze Tracker, an off-the-shelf gaze tracker that uses an inexpensive webcam or video camera to track the user's eye. The software is readily available as open source, offering the possibility to try out gaze interaction for a low price and to analyze, improve and extend the software by modifying the source code.

- A novel gaze estimation method based on homographic mappings between planes. No knowledge about the hardware configuration is required, allowing for a flexible setup where camera and light sources can be placed at any location.

- A novel algorithm to detect the type of movement that the eye is performing, i.e. fixation, saccade or smooth pursuit. The algorithm is based on eye velocity and movement pattern, and allows to smooth the signal appropriately for each kind of movement to remove jitter due to noise while maximizing responsiveness.

Resumen

Las personas con disfunciones motoras severas son, en muchos casos, incapaces de utilizar periféricos como el ratón o el teclado y, en consecuencia, necesitan disponer de dispositivos de comunicación alternativos.

Las tecnologías de seguimiento de la mirada (*gaze tracking*) ofrecen a estas personas la oportunidad de utilizar los movimientos oculares para interactuar con el ordenador, haciéndolas, por tanto, más independientes. Los avances de la tecnología, tanto en robustez como en precisión, han permitido la aparición de varios sistemas comerciales.

A pesar de las mejoras introducidas en los últimos años en los sistemas de gaze tracking, los altos precios de puesta en el mercado han impedido que la tecnología se universalice. El uso de hardware especializado, como cámaras industriales o luces infrarrojas, incrementa la precisión y robustez del sistema, pero también el coste asociado, lo cual impide a muchos usuarios potenciales el acceso a esta tecnología. Añadido a lo anterior, estos sistemas ven reducida su flexibilidad al tener que colocar los diferentes componentes en lugares específicos, o incluso por estar éstos incorporados de forma permanente en el monitor.

Los sistemas de seguimiento de la mirada construidos con componentes de bajo coste pueden, potencialmente, facilitar el acceso a la tecnología y bajar los precios. Estos sistemas también suelen ser más flexibles, puesto que ofrecen la posibilidad de colocar los componentes en distintos lugares, dependiendo de la configuración del equipamiento y de las necesidades específicas del usuario. No obstante, suelen ser menos robustos, debido al menor control sobre la configuración del hardware y la peor calidad de los componentes usados en comparación con los sistemas comerciales.

El trabajo desarrollado en esta tesis aborda diferentes retos introducidos por el uso de componentes de bajo coste. Tres son las contribuciones principales:

- El desarrollo y la evaluación del ITU Gaze Tracker, un sistema de seguimiento de la mirada de bajo coste que utiliza una webcam o una cámara de vídeo para determinar dónde mira el usuario, y cuyo software está disponible como código abierto. Este sistema ofrece la posibilidad tanto de probar la tecnología de gaze tracking por un bajo precio, como de analizar, mejorar y adaptar el software, modificando el código fuente.

- Un novedoso método de estimación de la mirada basado en transformaciones homográficas entre planos. Dicho método no requiere de información sobre la configuración del hardware, lo cual aumenta la flexibilidad a la hora de colocar la cámara y las fuentes de iluminación.
- Un novedoso algoritmo de detección del tipo de movimiento que el ojo está llevando a cabo: fijación, movimiento sacádico o seguimiento suave. El algoritmo se basa en la velocidad ocular y el patrón de movimiento del ojo, y permite suavizar la señal de manera apropiada a cada tipo de movimiento, reduciendo el ruido y maximizando la responsividad.

Acknowledgments

It is amazing how fast the last three years have passed. Doing a PhD in Denmark has turned out to be one of the most important decisions in my life. What I have learned, experienced and lived during this time cannot be described in words, and wouldn't have represented so much without the companion, encouragement and support of many people. It is my wish to thank some of them here.

First of all, I would like to thank my three supervisors, John Paulin Hansen, Dan Witzner Hansen and Arantxa Villanueva Larre. John, for guiding me along the paths of gaze interaction, for introducing new fields of research and for always being so enthusiastic; Dan, for providing me with a good theoretical foundation on eye tracking and gaze estimation and for always being so inspiring with new ideas; and last but not least, Arantxa, for convincing me to start a PhD at ITU and for being patient enough to explain the theory of gaze estimation over and over again.

A really special thanks to Julio Mateo for his efforts in editing our articles, for proofreading this dissertation, for the interesting discussions about the results of our experiments, and for making my stay in Dayton an unforgettable experience. I would also like to thank John Flach for inviting me to Wright State University and for giving me the opportunity to present my work to their group.

A big thanks to the COGAIN Network of Excellence for organizing fruitful conferences and courses, for covering the expenses of traveling to such events, and especially for granting me the money to visit the Tokyo Institute of Technology. I would also like to thank both Professor Kenji Itoh and Hirotaka Aoki for inviting me and providing everything I needed during my stay in Tokyo.

I would like to thank the different office mates I have had during this time. Søren Mørk Petersen, for the warm welcome to ITU and for making my first months in Denmark easier; Henrik Skovsgaard, for the fruitful discussions about gaze interaction and the nice stays at conferences; and Emilie Møllenbach, for being so cheerful and for always trying to provide as much help as she can. Although unfortunately I didn't share an office with him, Martin Tall deserves a big thanks for designing a flashy interface for the ITU Gaze Tracker. Those long working days before the public release of the software won't be forgotten.

I would also like to thank my colleagues and fellow PhD students at the Innovative Communication group at the IT University of Copenhagen for the nice discussions, seminars, and coffee and lunch breaks in the corridor. They have definitely made my time at ITU much more enjoyable. A very special thanks to Tanja Belinda Andersen and to Nis Johansen for becoming more than just colleagues.

Not everything has been work, and therefore I would like to thank my friends, both in Pamplona and in Copenhagen, for making my social life more interesting, listening to my problems and cheering me up when I needed it. Thanks Christian, Dani, Eneko, Idoia, Olaia, Saioa, Teresa, Ida, Miguel, Julien, Joaquín. I'd like to give a special thanks to Elena, for encouraging me to start a PhD in Copenhagen and for all the support she provided when I moved to Denmark. I would not be where I am if it had not been for her. Thanks.

Finally, I would like to thank my parents, José Luis and Aurora, for supporting me in every way they can, for keeping me entertained over Skype, and for contributing to this work by proofreading the Spanish translation of the Abstract.

Finalmente, me gustaría dar las gracias a mis padres, José Luis y Aurora, por apoyarme en todo momento, por mantenerme entretenido en Skype, y por contribuir a esta tesis corrigiendo el Resumen.

Contents

Outline of the Thesis **xv**

I Introduction and Motivation **1**

1 Overview **2**
- 1.1 Motivation . . . 2
- 1.2 Introduction to Gaze Interaction . . . 3
- 1.3 Research Question . . . 4

2 The Nature of the Eye **6**
- 2.1 Physiology of the Eye . . . 6
- 2.2 Movements of the Eye . . . 9
  - 2.2.1 Fixations . . . 9
  - 2.2.2 Saccades . . . 10
  - 2.2.3 Smooth Pursuit . . . 10

3 Eye Tracking and Gaze Estimation **12**
- 3.1 Eye Tracking Techniques . . . 14
  - 3.1.1 Electro-Oculography (EOG) . . . 14
  - 3.1.2 Contact Lenses . . . 14
  - 3.1.3 Video-Oculography (VOG) . . . 14
- 3.2 Basics of Gaze Estimation . . . 17
- 3.3 Eye Movement Detection . . . 17
- 3.4 Gaze as an Input . . . 18

II Low-Cost Gaze Tracking: Challenges and Engineering Considerations **21**

4 Low-Cost Gaze Tracking: The ITU Gaze Tracker **22**
- 4.1 Previous Work . . . 23

4.2 The ITU Gaze Tracker 24
4.2.1 Objective 24
4.2.2 Hardware 25
4.2.3 Software 26

5 Performance Evaluation of Gaze Input 30
5.1 Target-Acquisition Tasks 31
5.1.1 Fitts' Law 31
5.1.2 Target-Acquisition Tasks in the ISO 9241 - Part 9 Standard . . 33
5.1.3 Previous Research 33
5.2 Eye-Typing Tasks 34

6 Assessing Low-Cost Gaze Interaction 39
6.1 Desktop Scenario 39
6.1.1 Lack of Head-Pose Invariance 40
6.1.2 Performance Evaluation in a Desktop Scenario 42
6.1.3 Discussion and Limitations 47
6.1.4 Pilot Study with a Person with ALS 47
6.2 Mobile Scenario 49
6.2.1 Mobile Gaze Interaction: Challenges and Workarounds 50
6.2.2 Performance Evaluation in a Mobile Scenario 51
6.2.3 Limitations of the System 54
6.3 Conclusions 54

III Gaze Estimation Using Homographies 57

7 Gaze Estimation 58
7.1 Introduction 58
7.2 Interpolation Methods 60
7.3 Geometry-Based Methods 61
7.3.1 Fully Calibrated Setups 62
7.3.2 Uncalibrated Setups 63

8 Eye Model Based on Homographies 68
8.1 Planar Homographies 68
8.2 Eye Model 69
8.2.1 Estimation of the PoR 71
8.2.2 Non-Linear Extension 71
8.3 Model Assumptions 72
8.3.1 Affine Projection 73
8.3.2 Pupil and Cornea Are Coplanar 74
8.3.3 Planar Cornea 74
8.3.4 The Corneal Reflections Are Created by a Projection 74

8.3.5 Optical and Visual Axes Are Collinear 75
8.3.6 No Refraction 75
8.3.7 Location of Light Sources 76

9 Eye Model Assessment on Simulated Data 77
9.1 Methodology 78
9.2 Estimation Bias due to Model Assumptions 79
9.2.1 Zero-Error Case. Model0 79
9.2.2 Effect of Perspective Projection. Model1 80
9.2.3 Effect of Non-Coplanarity of Pupil Plane and Reflections Plane. Model2 82
9.2.4 Effect of Reflection. Model3 83
9.2.5 Effect of Curvature of the Cornea. Model4 85
9.2.6 Effect of Angular Offset between Optical and Visual Axes. Model5 85
9.2.7 Effect of Refraction. Model6 87
9.3 Effect of Number of Calibration Targets 88
9.4 Head-Pose Tolerance 90
9.4.1 Effect of Head Moving on a Plane Parallel to the Screen 90
9.4.2 Effect of Head Moving in Depth 92
9.5 Real Scenario 93
9.6 Conclusions 94

10 Eye Model Assessment on Real Data 96
10.1 Hardware Setup and Procedure 96
10.2 Eye Feature Extraction 98
10.3 Estimation Bias with Head Still 99
10.4 Effect of Number of Calibration Targets 101
10.5 Effect of Head Movements 102
10.6 Conclusions 103

IV Enhancing Gaze Interaction 107

11 Fixation Detection 108
11.1 Fixation Detection Algorithms 109
11.1.1 Velocity-Based Algorithms (I-VT) 109
11.1.2 Dispersion-Based Algorithms (I-DT) 110
11.1.3 Area-Based Algorithms (I-AOI) 110
11.2 Velocity- and Dispersion-Based Algorithm 110
11.2.1 Detection of Fixations 111
11.2.2 Smoothing Stage 111
11.2.3 Evaluation 112
11.3 Discussion 113

12 Smooth Pursuit Detection **115**
12.1 Detection of Smooth Pursuit Movements 115
12.1.1 Sources of Error in Dynamic Interfaces 116
12.2 Previous Work 118
12.3 Eye Movement Detection Algorithm 118
12.3.1 Detection Algorithm 119
12.3.2 Smoothing and Prediction 121
12.3.3 Evaluation 123
12.4 Conclusions 128

13 Hands-Free Pointing and Clicking **130**
13.1 Selection in Gaze-Based Applications 130
13.1.1 Selection by Gaze Only 131
13.1.2 Multimodality: Combining Gaze Pointing and EMG Clicking . 132
13.2 Performance Evaluation of Gaze Pointing and EMG Clicking 134
13.2.1 Participants 134
13.2.2 Apparatus 134
13.2.3 Design and Procedure 135
13.2.4 Results and Discussion 136
13.3 Dealing with False Activations in a Gaze-EMG Combination 139
13.3.1 Integrated Multimodal Interface 140
13.3.2 Participants and Apparatus 141
13.3.3 Design and Procedure 141
13.3.4 Results 143
13.4 Discussion 144
13.5 Further Reading 145

V Conclusion and Future Work **147**

14 Conclusion **148**
14.1 Contributions of this Thesis 148
14.2 Future Work 151

A Kalman Filter **153**

References **155**

Outline of the Thesis

The work presented in this thesis is part of the research carried out by the Gaze Group at the IT University of Copenhagen. The objective of this group is to develop and evaluate gaze tracking systems built from low-cost components, and to explore new gaze-based interaction paradigms that are robust and tolerant to noise.

The work presented in this thesis focuses on eye gaze information used as an input to control an interface, and covers the development and testing of a low-cost gaze tracking system, the investigation of a novel gaze estimation technique, and the exploration of ways to enhance gaze pointing. The thesis is divided into four major parts:

Part I presents the motivation and the research question of the thesis and describes the physiology and movements of the eye. The part concludes with an introduction to the technology used in gaze interaction.

Part II provides a description of the ITU Gaze Tracker, a gaze tracking system built from low-cost components and released as open source. The software is divided into three modules: the camera class, the gaze tracking library, and the graphical user interface. The gaze tracking library has been developed as part of this thesis. In order to assess the feasibility of using low-cost components for gaze interaction, a performance evaluation of the ITU Gaze Tracker has been carried out in two different scenarios: a desktop scenario and a mobile scenario. Two tasks have been used for this evaluation: target acquisition under the Fitts' Law framework, and eye typing, thus covering most of the interaction paradigms used in gaze-controlled applications. Some of the results presented in Chapter 6 have been published in [64] and [66].

Part III introduces a novel gaze estimation technique that allows for flexible hardware setups while providing a high degree of head-pose invariance, and extends the research developed by Hansen in [23]. This technique is based on planar homographies, and requires four light sources that can be placed in any location. An exhaustive analysis of the characteristics and limitations of this novel technique has been carried out on both simulated and real data (Chapters 9 and 10, respectively). Different eye models have been constructed in the simulations to provide an insight on the effect that the assumptions and simplifications of the model have on its accuracy to estimate gaze. This investigation includes a comparison with two state-of-the-art gaze estimation methods.

Part IV explores ways to improve and enhance gaze interaction. The inherent noise of gaze tracking systems will cause a gaze-controlled pointer to jitter, which can be distracting for the user. In order to smooth the signal without affecting the responsiveness of the system, the system needs to identify how the eye is moving while the user is interacting, so fast movements are not smoothed while slow movements are filtered to reduce the jitter in the cursor. Chapters 11 and 12 present different algorithms to identify the eye movements and smooth the cursor position. Chapter 13 investigates the combination of gaze pointing with facial-muscle activation through the use of an electromyography (EMG) system, allowing for fast and robust selections in gaze-controlled applications. The results presented in Chapter 13 have been partly published in [65] and [1].

Part I

Introduction and Motivation

Chapter 1

Overview

This chapter presents the motivation for the work developed in the thesis and describes the research questions that are sought to be addressed.

1.1 Motivation

Birger Jeppesen has Amyotrophic Lateral Sclerosis (ALS), a fatal motor neurone disease caused by the degeneration of motor neurons and characterized by a progressive muscle weakness and atrophy. He has had ALS for 12 years and is now in the late stages of the disease. He is on a ventilator and apart from the eyes and his forehead, he cannot move any other muscle in his body. His eyes are his only means of communication, and his cognitive skills have remained unaltered.

People that suffer from severe motor-skill impairments like Birger are often unable to use standard input devices such as mouse and keyboard. They therefore have a strong need for alternative communication devices that allow them to have control over their environment and to communicate with others. Gaze tracking systems provide the technology to interact with a computer by using the eye movements. During the time he has been paralyzed by ALS, Birger has written a book and several articles using a commercial gaze tracking system (QuickGlance) provided by the health authorities [35]. However, many people with ALS and other motor-skill disabilities do not have access to this technology due to the high prices of commercial systems, usually above $10.000. Furthermore, commercial systems are often very rigid and the hardware cannot be modified to meet specific needs. Although most often these systems are used placed on a desk, in some situations a disabled person might need to use the system in a less conventional scenario, such as a wheelchair or a hospital bed. The rigidity of the hardware components makes it challenging to mount and successfully use these systems in such situations.

A reliable and flexible gaze tracker built from off-the-shelf components has been the dream of people with motor-skill impairments, caretakers and researchers alike, since it would facilitate access to the technology, bring prices down and promote research within the field and the development of new applications that make use of gaze information [28].

The main motivation for this thesis is people with severe motor-skill disabilities. The work carried out aims to develop and analyze a gaze tracking system that is low-cost and flexible regarding the geometry of the hardware components, and that can be easily used by a disabled person. As a secondary motivation, providing such a system might increase the interest on gaze tracking technology. Using gaze interaction in other fields, for example in videogames, can further encourage research and the development of better and cheaper systems, ultimately benefiting people with disabilities.

1.2 Introduction to Gaze Interaction

The use of gaze input in human-computer interaction has been regarded as promising for the last two decades. Most efforts in the eye tracking community are headed towards improving the technology for people with severe motor-skill disabilities, since they have a need for alternative input devices that can substitute the mouse. Hutchinson et al. [31] presented in 1989 one of the first gaze tracking systems that allowed disabled people to communicate using their eye movements. Since then, there have been a plethora of systems and gaze-based applications developed. Jacob and Karn [34] argued in 2003 that gaze interaction technology had reached a point where it could become mainstream and be used in general human-computer interaction, possibly as an addition to mouse and keyboard input. However, in hindsight eye tracking was not as "ready to deliver the promises" as the authors indicated in the title of their article. Even though the technology is evolving at a fast pace and commercial systems are very robust and accurate, we yet have to see a spread of general applications where gaze information is used.

The pioneering user group of gaze-controlled applications has been people with severe motor-skill disabilities. In situations where eye movements represent the only means of communication, gaze interaction provides an invaluable communication tool that allows disabled people to control their environment, write documents or call for assistance. Input speed becomes crucial for individuals using alternative input devices in general, and gaze interaction in particular, since it sets the upper limit of their communication possibilities. Improving input speed for locked-in patients is one of the main goals of the gaze tracking community. To this effort, a wide range of eye-typing applications has been developed with the dual objective of dealing with the inherent noise in gaze tracking technology while maximizing the efficiency of the writing process [27], [86].

Although our eyes are well suited for pointing, they lack a selection mechanism [33]. Selections are usually performed by looking at the desired object for a prespecified time, and this makes gaze-only interaction slow and inefficient. Input speed in gaze-based application can be increased by complementing gaze pointing with alternative modalities that allow the user to perform selections. Ware and Mikaelian [88] showed that gaze pointing could be twice as fast as mouse pointing when selections were performed with a button. However, severely disabled users are often unable to control manual devices. Alternative input techniques, such as facial-muscle activation through the use of electromyographic signals, might become a feasible and low-cost solution that can enhance gaze interaction in these cases where manual devices cannot be employed to perform selections [57].

Commercial gaze trackers make use of specialized hardware, such as industrial cameras, filters, lenses and infrared light sources. Most often, the different hardware components are fixed and built into the monitor, making the system robust and tolerant to head movements, but decreasing its flexibility. On the other hand, systems built from off-the-shelf components often have a higher degree of flexibility, and the hardware components can usually be placed in the most convenient location. However, a lack of control over the hardware configuration introduces more challenges to infer where the user is looking, since fewer assumptions about the setup can be made. Furthermore, the lower quality of low-cost components increases the noise in the estimated user's gaze.

1.3 Research Question

This thesis deals with gaze interaction using low-cost and off-the-shelf components. Rather than focusing on solving a specific problem, a range of challenges associated with gaze interaction is investigated and discussed. First and foremost, is it possible to develop a gaze tracking system built from low-cost and off-the-shelf components that has a similar performance as commercial systems? What are the limitations of such a system and how can they be addressed? To provide an opportunity to address these questions, the ITU Gaze Tracker has been developed, based on low-cost and off-the-shelf hardware. In April 2009 it was released as open source software. Prior to the public release of the system, an analysis of its characteristics and limitations was carried out in order to assess its performance and explore possible areas of improvement.

A high degree of flexibility is often required in non-desktop scenarios where using a conventional gaze tracking system becomes challenging. For example, it might be difficult to place a commercial system in front of a patient lying in bed. Estimating the user's eye gaze in situations in which the location of the different components is unknown becomes more challenging, since fewer assumptions about the hardware setup can be taken for granted. In this thesis, I investigate a novel gaze estimation method and study the effect of the different model assumptions to identify the main sources of error. A comparison with state-of-the-art methods proposed in the literature is carried out using both simulated and real data.

Gaze pointing is a noisy input method compared to e.g. a mouse. Selection of objects on the screen becomes even more challenging when low-cost components are used, due to the lower quality compared to high-end hardware. Will efficient smoothing of the input signal provide better results? An eye movement detection algorithm that provides a robust smoothing mechanism under a noisy input has been developed and investigated.

Selection methods based on gaze only are slow and unnatural, due to the need of staring at a target or performing eye gestures, and devices such as the mouse or other hand-activated buttons are often not usable by people with severe motor-skill disabilities. A need for alternative and efficient hands-free selection techniques arise. Are there any methods that can complement and enhance gaze pointing in gaze-controlled applications? A combination of gaze pointing and facial-muscle activation has been investigated in order to explore its potentials and limitations.

Before addressing these questions, we first need to gain insight into the characteristics of the human eye. The next chapter presents an introduction to the basic physiology and movements of the eye.

Chapter 2

The Nature of the Eye

The eyes constitute the main organ that humans use to perceive the world. We use our eyes to obtain information about our surroundings by directing our eye gaze to objects of interest. Furthermore, our eyes play an important role in social interaction: by looking at another person's eyes we gather information about their mood, their attention and their emotional state. Eyes are most often regarded as an input organ, where the information flows unidirectionally from the world into the eye. However, we also use our eyes to show emotions and interest. In general, the direction of gaze reflects the focus of our attention, although in some cases we might stare at something and have our attention elsewhere [37]. If we are able to track a person's eye movements, we can deduce, at least to some degree, what holds the attention of the observer.

We also use our eyes when we interact with the world. Before grabbing an object, we look at it to examine its characteristics. For example, if we grab a cup of coffee from the desk, we will glance at it and then we will move our arm to grab it. When leaving the mug back on the desk, we will look at the location where we are positioning it. Therefore, eye gaze provides a sense of the intention a person has to interact with the physical world [42].

This chapter describes the physiology of the eye and the basic eye movements that occur when we look around. It is not the intention to provide a thorough review of the characteristics of the eye, but rather to point out the important aspects that are relevant in gaze interaction and gaze tracking technology.

2.1 Physiology of the Eye

A schematic view of the human eye is shown in Figure 2.1[1]. The eyeball is approximately a spherical object with a radius of around 12 mm. There are three structures

[1]Figure taken from http://en.wikipedia.org/wiki/Eye

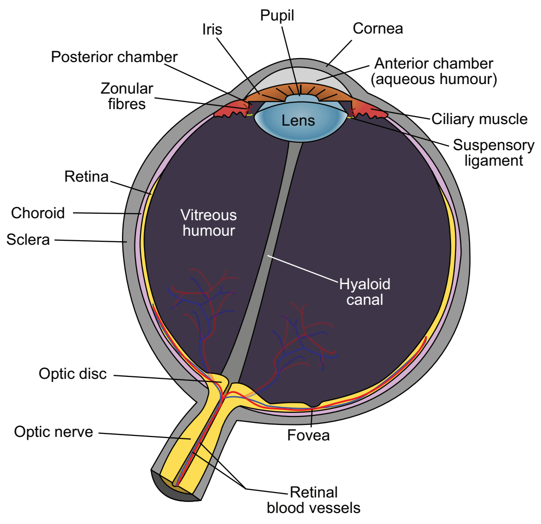


Figure 2.1: Diagram of the eye.

in the eye that can be observed from the outside: the *iris* (colored part), the *pupil* (black part in the center of the iris) and the *sclera* (white part). Pupil and iris are covered by the *cornea*, a transparent layer that refracts light before it enters the eye. The boundary between cornea and sclera is known as *limbus* (not depicted in Figure 2.1). The pupil is the aperture through which light enters the eye, and it can regulate the amount of light by expanding and contracting. When light enters the eye, it is refracted by the *lens*, a transparent structure located behind the pupil. The lens changes its shape in order to focus the rays of light onto the *retina*, a surface in the rear part of the eye that contains *photoreceptors*. Photoreceptors are a type of neurons sensitive to light, and convert light into electrical impulses that are sent to the brain through the *optic nerve*. The photoreceptors are divided into two main groups, *rods* and *cones*. Rods provide monochromatic vision and are very sensitive to light, therefore providing most of the visual information in dim light conditions (night vision). Cones are less sensitive to light than rods, but on the other hand they perceive colors and are more sensitive to rapid changes in the scene thanks to a lower response time to stimuli. Figure 2.2 shows the distribution of rods and cones across the retinal surface. The highest concentration of cones occurs in the *fovea*, a region in the retina that is responsible for sharp vision. The angle subtended by the fovea is around 2°. Visual acuity is approximately constant within those 2°, and drops linearly from there to the 5° border of the fovea, and exponentially beyond 5° [13].

Figure 2.3 shows a schematic view of the eye, adapted from [82]. The eye is not completely symmetric; however, an approximated symmetry axis can be considered as the line joining the different lenses of the eye. This is called the *optical axis*, or

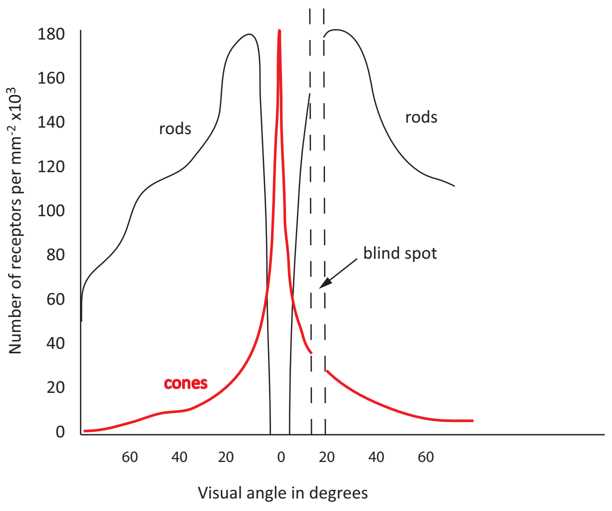


Figure 2.2: Density distribution of rod and cone photoreceptors across the retinal surface. Adapted from Duchowski [13].

Line of Gaze (LoG), and it does not coincide with the actual Line of Sight (LoS), or *visual axis*. The visual axis is the line that connects the fovea with the object we are looking at. Optical and visual axes intersect at the cornea center, with an angular offset that is subject dependent. There is a high variability among humans. Typical values are 5° in the horizontal direction and 1.5° in the vertical direction.

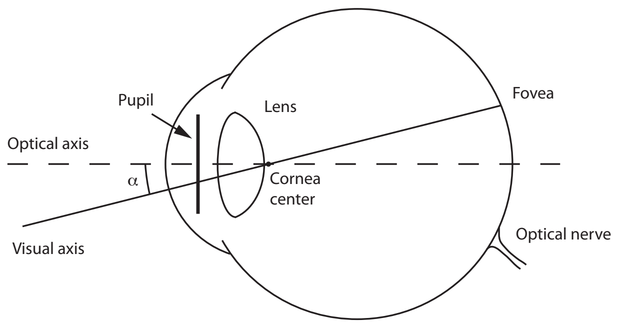


Figure 2.3: Schematic top-down view of the right eye. Adapted from Villanueva [82].

There are two structures in the eye that refract light: the cornea and the lens. Apart from refracting light, each surface of these structures will also reflect some of the light back, thus forming secondary images of the source, the so-called *Purkinje images* (see Figure 2.4). The first Purkinje image is created on the anterior surface of the cornea, and is the most visible of the four.

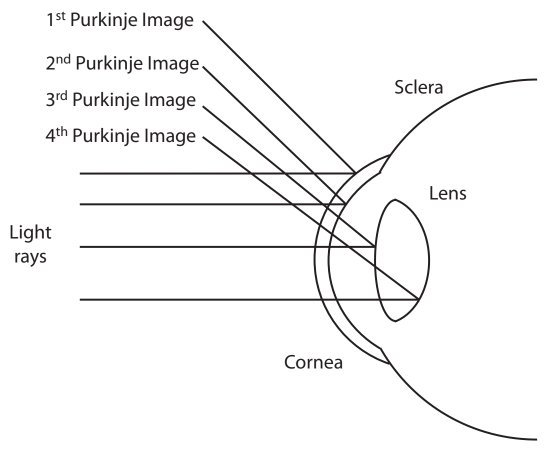


Figure 2.4: Formation of the four Purkinje images on the eye.

2.2 Movements of the Eye

During our everyday activities, we are presented with a vast amount of visual information. Our eyes are constantly moving in order to select the most relevant information on the basis of: (1) our goals and expectations, for example when we look for a specific object in the scene (endogenous or goal-driven control), and (2) the intrinsic salience of the objects in the visual scene, for example a moving object in a static scene will draw our attention (exogenous or stimulus-driven control) [20]. Two types of eye movements are used by the visual system to place relevant objects in the visual scene on the fovea: *saccades* and *smooth pursuit*. When the eye is stable, a *fixation* takes place.

2.2.1 Fixations

A fixation occurs when we stare at an object of interest, and is usually considered to have a duration of at least 100 to 150 ms [63]. It is during a fixation that we gather data from our surroundings.

Although we perceive the image as being completely still, there are three different micromovements taking place during a fixation, namely drifts, tremors and microsaccades. The objective of these micromovements is to keep the photoreceptors in the retina constantly excited, and to bring the visual target to the center of the fovea. Figure 2.5 illustrates the occurrence of these movements [7].

The dispersion due to these movements is usually around 1° of visual angle [94]. The velocity of the eye during a fixation is considered to be lower than 20°/s [63].

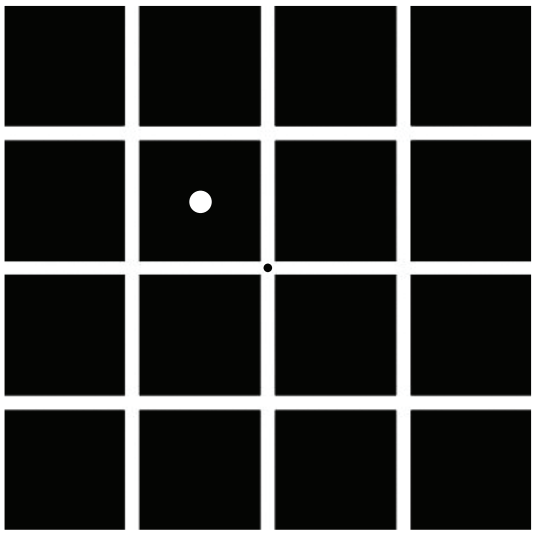


Figure 2.5: Demonstration of the micromovements taking place during a fixation (adapted from [7]). After looking at the white dot for around 20 seconds, the reader should fixate on the black dot. Relative displacements between the afterimage and the figure itself will be noticeable, following slow drifting movements and microsaccades.

2.2.2 Saccades

A saccade is a fast conjugate eye movement that repositions the eye so that the object in the visual scene is projected on the fovea. Saccades usually occur between two fixations. No information is gathered by the visual system during a saccade.

During a saccade the eye rotates at a high velocity, often reaching a peak velocity above 700°/s for large amplitudes. The duration of a saccade is not constant, but increases as the amplitude of the movement increases. The duration of saccades larger than 5° is around 30 ms plus 2 ms per additional degree of amplitude [7]. Saccadic movements involved in searching are in the range of 1 to 40°, and for amplitudes higher than 30° some head motion will take place. There is a refractory period of 100 to 200 ms between saccades, and a latency of 100 to 300 ms when the eyes respond to a visual stimulus [94].

2.2.3 Smooth Pursuit

Smooth pursuit takes place when our eyes track a moving object. The visual system is able to track objects in the range of 1 to 30°/s [94], [61]; however, velocities above 100°/s are also reported [7]. Smooth pursuit movements are not subject to endogenous control, and require a moving stimulus.

Smooth pursuit movements consist of two different components: a motion component and a saccadic component. The former is used to maintain the fovea stabilized on the moving object; the latter is used to perform minor corrections and to reposition the fovea on the target[2]. A smooth pursuit movement is composed of three stages [61]: (1) an initial smooth pursuit movement takes place around 125 ms after the stimulus commences movement; (2) a subsequent correcting saccade that places the object on the fovea takes place approximately 100 ms later; (3) a final smooth pursuit movement occurs, and the eye slowly matches the target velocity.

Saturation velocity is around 25 to 30°/s. For target velocities below saturation, the visual system is capable of tracking the object using only the motion component. For higher velocities, correcting saccadic movements will occur [61].

Figure 2.6 shows the approximate distribution of eye velocities for each of the three types of eye movements presented above. Chapters 11 and 12 describe methods to classify a sequence of gaze coordinates as belonging to each type of eye movement in real-time applications. Techniques to separate fixations from saccades based on eye velocity are common in gaze tracking systems. However, classification of smooth pursuit movements is more challenging, as the velocity distributions of fixations and smooth pursuit overlap.

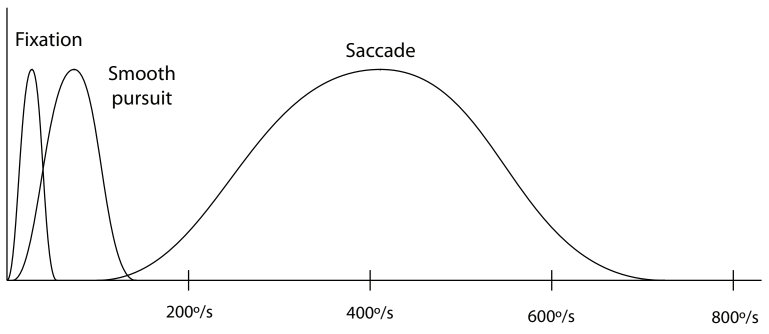


Figure 2.6: Approximate distribution of eye velocities in degrees per second for each type of eye movement: fixation, smooth pursuit and saccade.

The next chapter introduces the basic techniques used to track the eye and estimate where a person is looking.

[2]Parafoveal tracking can also occur [7].

Chapter 3

Eye Tracking and Gaze Estimation

A gaze tracking system seeks to find where a person is looking by means of information obtained from the eye. Depending on the methods employed and the information obtained, it is possible to determine the person's eye gaze as a 3D vector in space (the Line of Sight) or as a 2D point on a plane (the Point or Regard, PoR) [24].

The process of finding gaze can be divided into two subprocesses, eye tracking and gaze estimation. Eye tracking involves measuring the eye movements as the eye moves in its orbit. This is often achieved by detecting and tracking eye features such as the pupil or the iris over time. Gaze estimation uses these eye features to estimate the user's eye gaze. The information of the eye used to determine the user's gaze will depend on the technology employed.

Depending on the use of the gaze tracking system, two different applications can be considered [13]: *diagnosis* and *interaction*. Diagnostic applications analyze eye movements to gather information on the person's cognitive processes and attention when performing a specific task, such as reading a newspaper or driving. This analysis is usually carried out *a posteriori*, and therefore the eye movements do not have an effect on the scene observed. Diagnostic applications are widely employed in different fields, such as usability research, marketing and psychology. Yarbus [91] showed that humans examine a scene by performing a sequence of eye movements on the regions of interest. The eye gaze is positioned on a salient feature of the scene during a short period of time before jumping to the next point. Yarbus demonstrated that the path followed by the observer's eyes and the regions of interest depends on the task the user is performing (endogenous control, [20]). In his work, he used a picture and asked the observers to answer different questions. A different task produced a different scanpath over the image. Figure 3.1 shows the results when one subject was asked to perform different tasks. For example, in path number 1 the subject was asked to freely examine the picture, while in path number 3 the subject was asked to estimate the ages of the people in the picture. As can be observed, the observer payed more attention to the faces of the people in the latter case.

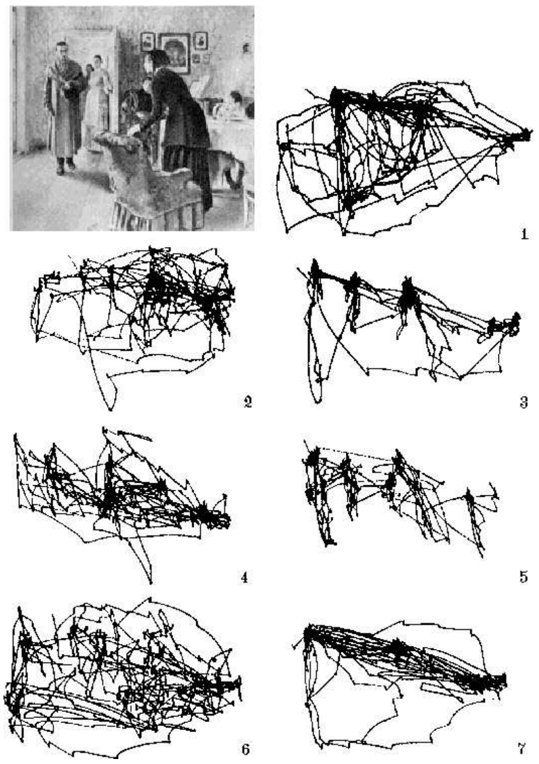


Figure 3.1: Scanpaths of the same subject when asked to perform different tasks. Taken from Yarbus [91].

The present work focuses on using gaze for interacting with computers. The gaze information provided by a gaze tracker is therefore used as an input to control a graphical user interface on a screen. The PoR coordinates estimated by the system are usually noisy, mainly due to inaccuracies in the extraction of the eye features. When the PoR is used as a pointer (i.e., as a substitute for the mouse) the jitter in the pointer can be distracting for the user (if the cursor is displayed). Some degree of smoothing is usually applied to the signal to make the cursor more stable.

The process to use eye gaze as an input to control an interface consists of three steps: (1) Data regarding the eye is captured, and relevant eye features are detected and extracted. (2) This information is employed to estimate gaze (either Line of Sight or PoR) using a gaze estimation algorithm. (3) A post-processing algorithm is applied to smooth the signal; this step usually requires identifying the type of eye movement the user is performing (see Section 2.2). The resulting PoR is then used to control a gaze-based application. for instance by acting as a pointer. Gaze information can be complemented with other input devices to enhance the interaction with the gaze-controlled interface.

The next sections present an overview of the three steps introduced above and provide some examples of the methods employed.

3.1 Eye Tracking Techniques

The movements of the eye can be tracked using different technologies, and the accuracy and invasiveness of a gaze tracking system will depend on the method and hardware employed. Eye tracking methods are commonly divided into three categories [13]:

- Electro-oculography, or EOG.
- Contact lenses.
- Video-oculography, or VOG.

3.1.1 Electro-Oculography (EOG)

The electro-oculographic technique is based on the existence of an electrical field that changes its potential as the eye moves in its orbit. To detect these changes in electric potential, electrodes are placed on the skin around the eyes [94]. The system is regarded as invasive, due to the necessity of placing the electrodes directly on the user's face. However, a big advantage is the possibility of using EOG with contact lenses and glasses without these affecting the performance, and its high tolerance to head movements. Also, the equipment used is rather inexpensive.

3.1.2 Contact Lenses

The most accurate method to track the user's eye gaze is achieved by means of special contact lenses. The accuracy is as high as 10 arc-seconds [94]. However, this method is extremely invasive. The contact lenses that the user needs to wear are usually connected to wires, making the system very uncomfortable to use. The use of this technology is therefore limited to laboratory research.

3.1.3 Video-Oculography (VOG)

The video-oculographic method uses a camera to record the eye movements of the user, and extracts the information of different eye features to determine the Point of Regard or the Line of Sight. The great advantage of video-based gaze trackers lies in their non-intrusiveness. In most cases the user does not need to wear any extra gear, and generally systems are tolerant to head movements, at least to a certain extent.

Arguably, video-oculographic systems are the most popular gaze tracking systems nowadays, and is the technique employed in this thesis.

Depending on the hardware configuration of the different components, it is possible to classify gaze tracking system as *remote* or as *head mounted*. In remote systems the camera and light sources are placed at a distance from the user, usually below or around the computer screen, making for a non-intrusive configuration. The user is often given the possibility of moving in front of the screen, as long as the eyes are kept within the field of view of the camera. For increased accuracy, some laboratory setups make use of a bite bar to ensure that the position of the user during an experiment is the same as the position during calibration.

Head-mounted systems, on the other hand, place the camera and light sources on the head of the user, usually mounted on a helmet or a pair of glasses [2]. Such systems make mobile gaze interaction on head-mounted displays possible.

VOG systems use computer vision techniques to detect and track the movements of the eye over time. The methods can be divided into three categories: *shape-based*, *appearance-based* and *hybrid* (for a complete review, see [24]):

- Shape-based methods rely on a prior model of the eye structures, which can be constructed from features of the eye (fixed-shape models) or from contours (deformable-shape models). The pupil, iris and eye corners are commonly used features. The image data are fitted to a predefined eye model, and therefore these methods are bottom-up.
- Appearance-based methods rely on building a model of the eye region that is fit to the eye by matching the template to the target feature. Eye models are usually built from a large set of training images with different subjects, illumination conditions and head poses that allow to build a classifier to detect the eye features in any given image.
- Hybrid methods combine different techniques, such as shape, appearance and color, to exploit the benefits of each method and overcome their limitations.

The method employed to track the eye will largely depend on the type of illumination used. Most systems make use of active infrared (IR) light, although passive systems that use natural light exist [24]. Systems based on active illumination use IR light to improve the quality of the image by providing a more stable illumination. Since it is invisible to the human eye, it is not distracting nor annoying to the user. The reflections produced on the cornea surface are also used to determine gaze in some gaze estimation methods. Figure 3.2 shows an eye under infrared light, and the same eye with no active illumination. Many reflections occurring on the eye due to external light sources are eliminated when IR light is applied. Moreover, the contrast between pupil and iris is greatly increased.

When only natural light is available, few assumptions on the image data can be made. Appearance-based methods that build a model of the eye region are often employed [24]. The iris is the most prominent feature due to its high contrast with

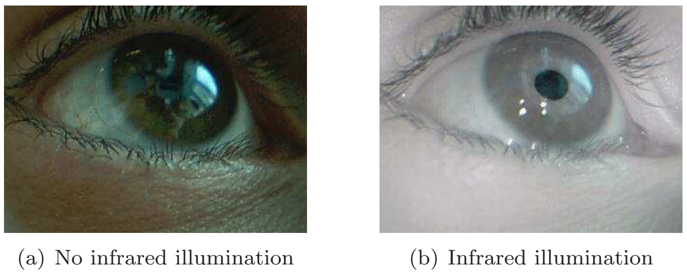


Figure 3.2: Eye without (a) and with (b) infrared light.

the sclera. However, it might be partially covered by the eyelids, making its detection more difficult and inaccurate.

Methods that rely on infrared illumination are the most popular in both commercial systems and in research. The pupil becomes a salient eye feature when IR light is applied (see Figure 3.2(b)). Depending on the location of the light sources with respect to the camera, two techniques can be employed. If the light source is co-axial with the camera, a *bright pupil* effect is obtained. This effect is similar to the red-eye effect that occurs when using a flash in photography. The light coming into the eye is reflected on the retina, and the camera 'sees' a bright pupil. If the light sources are off-axis with the camera, a *dark pupil* effect is achieved. In this case, the pupil appears darker than the surrounding iris. Figure 3.3 shows the bright pupil and dark pupil effects when using on-axis and off-axis illumination, respectively. Some systems synchronize the on-axis and off-axis light sources with the camera and apply an image difference between subsequent bright/dark pupil images to track the eye more robustly [18], [55], [98]. This method cannot, however, be applied in off-the-shelf systems, since additional electronics are required to synchronize light sources and camera.

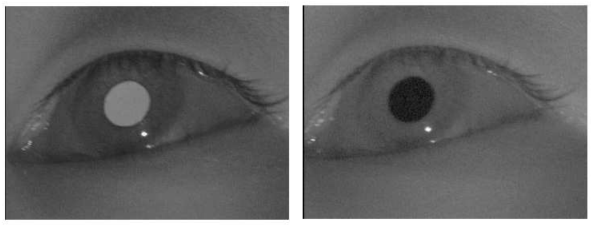


Figure 3.3: Bright (on-axis illumination) and dark (off-axis) pupil images [56].

Methods based on active IR light provide good tracking results in indoor, controlled scenarios. However, they are sensitive to external light sources that introduce additional reflections on the eye region, and are less robust under sun light or when the user wears glasses. Outdoor eye tracking is challenging, and efforts are devoted to improve the reliability under varying light conditions [17], [98].

3.2 Basics of Gaze Estimation

The objective of gaze estimation is to provide an accurate measure of the user's eye gaze given some information of the eye. In the case of image-based systems, this information is usually given in the form of eye features such as pupil center and corneal reflections. A gaze estimation method will therefore seek to find a correspondence between image data and eye gaze.

Finding this correspondence between eye data and gaze is the core problem of gaze estimation. Different approaches to find this relationship can be taken depending on the hardware employed and the knowledge available about geometric relationships between hardware components. These approaches can be divided into two different techniques:

- Geometry-based methods.
- Interpolation methods.

Geometry-based methods (e.g., [21]) rely on a three-dimensional model of the framework, including the eyeball and the camera and screen setup, and establish a relationship between image data and gaze using the geometry of the system. In order to build this relationship, a certain knowledge of the 3D locations of the different hardware components is usually required.

Interpolation methods (e.g., [56]), on the other hand, use general purpose equations, such as linear or quadratic polynomials, to map the image data to gaze coordinates. Interpolation methods are often used when little information on the hardware setup is available.

All methods require calculating some parameters, these being model-specific or the coefficients of a polynomial. Parameters are calculated through a calibration procedure in which the user is asked to look at certain points on the screen, usually between 4 and 16. During the calibration, a set of data is collected for each calibration target. Each set will contain information about the coordinates of the calibration target and the corresponding eye features, and the parameters being calibrated can then be determined.

3.3 Eye Movement Detection

Chapter 2 introduced the basic eye movements employed by the visual system to position the objects of interest on the fovea. These movements are of great relevance in diagnostic applications. For example, analyzing the eye movements of the user while he or she performs different tasks (e.g., reading or driving) allows to extract higher level information of the visual and cognitive processes taking place. Eye movements are often divided into fixations and saccades, and different metrics are employed in the analysis, for example fixation duration, saccade amplitude or saccade velocity. This analysis is usually carried out *a posteriori*.

Identifying the type of eye movement the user is performing can also improve the interaction in interactive gaze-controlled applications. Due to the noise in the extraction of eye features and in the gaze estimation process, a certain degree of jitter appears in the cursor position. If it is displayed, a jittery cursor can distract the user. Smoothing the signal over time during a fixation is therefore commonly seen in most commercial gaze trackers. An efficient smoothing algorithm is of special importance in low-cost systems, since the use of low-quality hardware usually adds more noise in the extraction of the eye features, making the cursor position more jittery.

Salvucci and Goldberg [63] presented in 2000 a taxonomy of fixation detection algorithms, and divided them into three main categories: velocity-based, dispersion-based and area-based. Chapter 11 describes these techniques and presents an implementation of an algorithm that combines velocity and dispersion characteristics. The limitations of the different methods will be described and discussed.

Smooth pursuit detection has not been fully explored in real-time, gaze-controlled applications. Although most interaction in graphical user interfaces is based on point-and-click, applications that display moving objects might benefit from a smooth pursuit detection algorithm. For instance, in video games it is common to see targets moving on the screen that the user has to select, e.g. to shoot at them. Some recent eye-typing applications also present a dynamic interface [27], [86]. In Chapter 12 a novel smooth pursuit detection algorithm that can be applied to gaze-controlled applications is presented and evaluated. This algorithm is integrated with the fixation detection algorithm presented in Chapter 11, allowing for the detection of the three basic eye movements: fixations, smooth pursuit, and saccades.

3.4 Gaze as an Input

The present work focuses on using gaze for interacting with computers in gaze-controlled applications. The PoR provided by a gaze tracker is therefore used as an input to control a graphical user interface on a screen. The user's eye gaze becomes a pointing device that substitutes (or complements) the mouse.

Eye movements can reveal our intention to interact with an object in the physical world. Interactive applications use eye movements as an input to control an interface. In these applications, the user looks around to select items and activate objects, and the interface reacts in real time by modifying the information displayed. However, it must be noted that even though our eyes can show our *intention* to interact with real-world objects, these objects do not react upon gaze. Jacob and Karn [34] proposed a taxonomy of gaze-driven applications, and regarded the eye movements required to control an interface and the response of the system to these movements as *natural* or *unnatural*.

- *Natural eye movement / Natural response*: A natural response requires the interface to react in a natural way to the user's gaze. Since physical objects do not react upon gaze, there are limited applications where the interface can respond naturally to eye movements. Applications of this kind react upon the person's visual attention, and the user has no direct control over the interface. Starker and Bolt [74] presented a display where information about the scene is given depending on the viewer's eye patterns. Vertegaal et al. [80] designed a videoconferencing system where eye movements are seamlessly transmitted to the different participants. This conveys information about who is talking or listening to whom by showing the attention of each participant. For a review of gaze-contingent displays and attentive user interfaces, see [14].
- *Natural eye movement / Unnatural response*: In the real world, we usually look at the objects before interacting with them. In an interactive gaze-based application, we look at the interface elements to interact with them, and these react upon our gaze in an unnatural behavior. Arguably, this way of gaze interaction is the most common: eye gaze direction is used for pointing at the objects we want to interact with [33], [34].
- *Unnatural eye movement / Unnatural response*: Trained, conscious eye movements can be used to operate an interface. Some applications require performing a specific eye pattern (i.e., gaze gestures) to confirm selections. Interaction with gaze gestures is not popular in gaze-based applications; it is, however, relatively new, and holds certain potential, as the requirement for an accurate system is relaxed [11], [54]. Isokoski discussed the feasibility of using off-screen targets for gaze-typing [32].
- *Unnatural eye movement / Natural response*: This category cannot occur in practice.

The use of eye movements as a direct input to control an interface has been extensively studied. Jacob [33] discusses different interaction techniques where eye movements can provide an additional input channel, and points out one of the main issues when using only eye movements for interaction, the *Midas Touch* problem. Although our eyes can be used for pointing after some training, they lack an activation mechanism, and therefore anywhere we look, an activation is issued. Ideally, an external switch would allow the user to perform selections, in the same way as the button does in a mouse. However, locked-in individuals like Birger Jeppesen might be unable to use a switch, and in these cases gaze-only selection techniques are required. Dwell-time clicking or long blinks are the most common alternatives when no selection devices are available. The former requires the user to stare at the desired interactive object for a predefined amount of time in order to issue an activation, while the latter requires a long blink.

Different studies have shown the speed advantage of gaze over manual input devices [88], [69]. Sibert and Jacob [69] found that gaze pointing was faster than mouse pointing for very short dwell times (150 ms); however, such short dwell time is unrealistic and would produce a high number of undesired selections in everyday interaction (usual dwell times are in the range of 250 to 1000 ms [51]). Zhang and MacKenzie [95] carried out the first performance evaluation of gaze pointing under the ISO 9241 - Part 9 Standard, comparing it to mouse pointing, and found gaze pointing combined with key activation to be as fast as the mouse. However, participants made more mistakes when using gaze. Provided good tracking conditions and after some training, our eyes can be used for pointing, although not as efficiently as with the mouse [95]. Section 5.1 presents some previous work on the performance evaluation of gaze pointing.

A large amount of research has been dedicated to developing and improving eye-typing applications. Most of these systems use an on-screen keyboard that present buttons on which the user needs to focus in order to activate them [31], [28], [3]. Interaction is achieved through a classical point-and-click paradigm. However, selection by dwell time slows down input speed. Dynamic paradigms for eye typing have also been explored [87], [27]. In these, the interface responds dynamically to the eye movements of the user. Majaranta and Räihä [52] presented a survey of eye-typing applications and design considerations that need to be taken into account to increase the efficiency and usability of such systems. Section 5.2 describes some of these applications.

The following chapters describe the ITU Gaze Tracker, a gaze tracking system built from low-cost components, and present an evaluation of its performance when used as an input device in target-acquisition tasks and for eye typing.

Part II

Low-Cost Gaze Tracking: Challenges and Engineering Considerations

Chapter 4

Low-Cost Gaze Tracking: The ITU Gaze Tracker

The use of off-the-shelf hardware components in image-based gaze tracking is an emergent research field. In recent years, the development of better and cheaper webcams and video cameras has led to a growing interest in the use of low-cost components for gaze interaction. As opposed to commercial systems where all the hardware is fixed and usually expensive, low-cost systems are built from inexpensive components that can be easily bought in any computer store.

There are many advantages of building systems from off-the-shelf hardware: it might allow users to have access to a system for a lower price than commercial systems, it might encourage research in the gaze interaction field and in the end it might help bring gaze tracking technology into the mainstream. However, using off-the-shelf components also introduces many challenges. For instance, the quality of the components is usually degraded with respect to the high-end hardware used in commercial systems. These lower quality components might introduce more noise in the extraction of eye information. The characteristics and configuration of the system components are also unknown, making it more difficult to use geometric models to estimate gaze that would allow to obtain 3D information on eye location and gaze direction.

Although there are many gaze tracking systems based on off-the-shelf components developed by enthusiasts, few of them have been described in the literature. This chapter presents the previous attempts and analyses their limitations to help identify the challenges that need to be solved in order to build robust low-cost systems. Finally, the ITU Gaze Tracker is described.

4.1 Previous Work

In 2004, Babcock and Pelz [2] presented a head-mounted eye tracker that uses two small cameras attached to a pair of safety glasses. One camera is pointing at the user's eye and is used for eye tracking, while the other one records the scene in front of the user. A small infrared LED is attached to the arm that holds the eye tracking camera, providing a dark-pupil effect and a corneal reflection. Figure 4.1 shows their system.

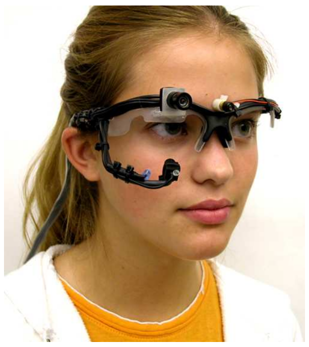


Figure 4.1: Head-mounted eye tracker built by Babcock and Pelz [2]. Courtesy of Babcock and Pelz.

Babcock and Pelz's system can be used for off-line analysis of eye movements, i.e. sequences are recorded and analyzed *a posteriori*. Li et al. [44] extended their work and built a similar system that worked in real time, called OpenEyes, which used the Starbust algorithm [45] for pupil detection. It is open hardware and open source, and the authors provide instructions on how to build the system and run the software on a Linux operating system. Being head-mounted, both systems are affected by head movements when interacting with a desktop screen.

Ryan et al. [62] improved the Starbust algorithm to increase the robustness in varying lighting conditions. Points on the contour of pupil and limbus are used to fit an ellipse, and the algorithm detects whether the fitted points correspond to either eye structure, allowing to switch between pupil and limbus tracking automatically.

Although the hardware components used in the systems described above are inexpensive, assembling the hardware requires advanced knowledge of electronics, which might prevent an average user from building and using such systems. A simpler hardware approach was taken by Zielinski in Opengazer[1], a system that uses a single webcam placed on the desktop. The gaze estimation method is not tolerant to head movements, and therefore the user needs to keep the head still after calibration. Unfortunately, there is no study evaluating the accuracy of this gaze tracker.

[1]http://www.inference.phy.cam.ac.uk/opengazer

Furthermore, it only runs on the Linux operating system and its development was discontinued in 2007.

Hansen and Pece [26] presented a system that uses a video camera to track the iris of the user. The system works in both natural and infrared light conditions, and is able to cope with extreme changes in illumination conditions. They report an accuracy of 4° of visual angle when the user keeps the head still.

4.2 The ITU Gaze Tracker

Despite the emergent interest in gaze tracking systems built from low-cost components, there is a need for systems that are easy to build and use. The previous attempts to build low-cost systems discussed above might be used for laboratory research, but in general advanced skills in electronics and computer science are required to build and use the systems.

The ITU Gaze Tracker aims to provide a low-cost system that can benefit both the end-user and the research community. Its performance has been evaluated and compared to other commercial systems in order to assess its strengths and identify its limitations. The present section provides a description of the system, while Chapter 6 presents an evaluation of the system when interaction is performed on a desktop monitor and on a head-mounted display in a mobile scenario.

4.2.1 Objective

The approach taken to develop the ITU Gaze Tracker aims to avoid the mistakes of previous low-cost systems. In order to provide a fully accessible system, a series of requirements was established:

- *Use of low-cost and off-the-shelf components.* Ideally, the hardware should be easy to buy in any electronics store or on-line.

- *No hardware modifications needed.* Assembling hardware can be difficult for an average user and should be avoided.

- *Flexible setup.* Since the components are not integrated like in most commercial systems, the user should be able to place the different components (camera, infrared lights) in different locations to meet specific requirements.

- *Plug'n'play.* The user should be able to just run the software after arranging the components. It should not be required to deal with source code.

- *Open-source software.* People working in the gaze interaction field might be interested in modifying the source code to fit specific needs. Providing an open-source solution can make this possible.

4.2.2 Hardware

The gaze tracking software has been developed so two possible hardware configurations can be used: a head-mounted setup with a small camera placed close to the user's eye, and a remote setup, in which case a video camera is placed on the desk. The hardware that can be employed to use the ITU Gaze Tracker will depend on the configuration chosen.

Head-Mounted Setup

In a head-mounted setup, the camera and possible external light sources must be mounted on the user's head. Therefore, these hardware components need to be as small and lightweight as possible, so the user does not get physical discomfort from wearing the headgear.

To keep the hardware as inexpensive as possible, using a webcam was considered to be the most feasible solution. Like most commercial systems, the ITU Gaze Tracker is based on active infrared illumination. Contrary to other low-cost solutions ([2], [44]) that build an external infrared LED close to the camera, the system proposed in this work makes use of a webcam with built-in infrared LEDs. This allows to simplify the hardware configuration and to make the system easy to build. Two different night vision webcameras were used during the development of the system, a Sandberg Nightcam and a Genius iSlim 321R[2]. Both cameras include a set of infrared LEDs and have an approximate cost of $20.

Due to the low resolution and broad field of view of webcams, the camera is required to be placed close to the user's eye in order to obtain a good image and extract the eye features robustly. In the initial configuration of the head-mounted setup, the webcam was mounted on an old ASL eye tracker helmet. This allows for a good fit in the user's head that prevents the camera from slipping when the user moves his or her head. A Sandberg Nightcam camera was mounted on the helmet and placed close to the user's eye. Such system was tested in a mobile setup where the users interacted with a head-mounted display. The results are presented in Section 6.2.

Although the strap helmet provides a good initial step to build and evaluate the low-cost webcam-based system, it cannot be considered to be off-the-shelf hardware. It is common to mount the camera using more conventional equipment, such as caps or safety glasses [2], [44]. In order to maximize the simplicity of the system, a piece of balsa wood was used as a mounting "device", as shown in Figure 4.2. The user bites on this piece of wood, and the camera is mounted on it close to the eye. In that way the camera can be hold very still with respect to the head, since effectively it is attached to the skull. Since the user has to bite the piece of balsa wood, it is not intended to be used for long periods of time. Rather, it provides an easy and

[2]The gaze tracking community at http://forum.gazegroup.org has reported having used the software employing modified webcams with external infrared illumination.

accessible way to try gaze interaction.

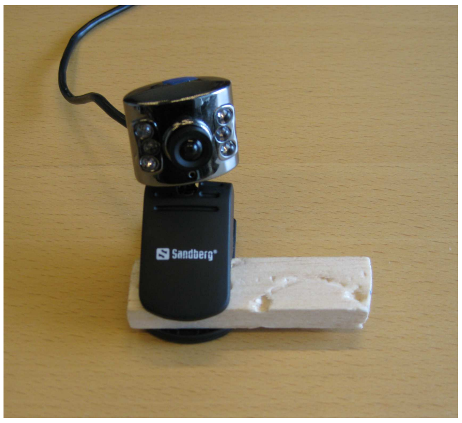


Figure 4.2: Webcam mounted on a piece of balsa wood that the user bites.

The performance of the gaze tracker in target-acquisition and eye-typing tasks using the head-mounted setup is evaluated and compared with other gaze tracking systems in Section 6.1.

Remote Setup

Most commercial gaze trackers that are used for interaction are *remote*, where remote means that the hardware is placed at a distance from the user. In this case, a camera that zooms into the user's eye must be used. Most often, a lens is mounted on the camera to obtain a big picture of the eye. Several videocameras make it possible to zoom into the user's face, providing a good image of the eye where it is possible to extract the features required to estimate gaze. Furthermore, they often include a night shot mode with built-in infrared LEDs.

The ITU Gaze Tracker has been tested with a Sony HDR-HC5 camera. This camera has a built-in infrared LED that proved not to be powerful enough to create a corneal reflection that could be detected robustly in the image. Therefore, external light sources were considered. A setup with two Sony IVL-150 infrared light sources is shown in Figure 4.3. These light sources can be bought in any electronics store, and run on batteries, so no extra wiring is required.

4.2.3 Software

The gaze tracking software runs under the Windows operating system, and is developed in C# using Visual Studio 8. The image processing is based on OpenCV[3], an

[3]http://sourceforge.net/projects/opencvlibrary

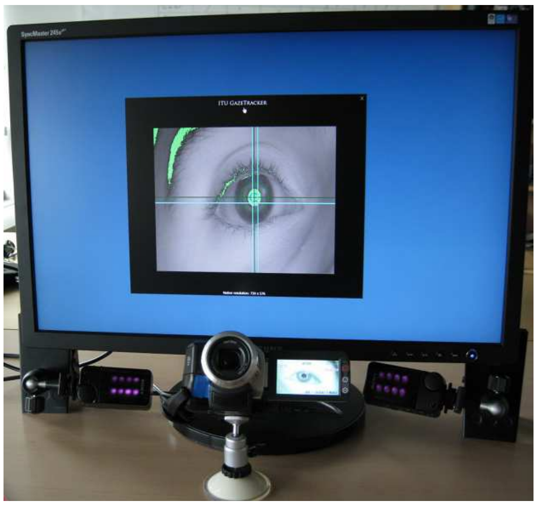


Figure 4.3: ITU Gaze Tracker in a remote configuration. The hardware components used are a videocamera with night vision mode and a pair of infrared light sources.

open source computer vision library originally developed by Intel and released with a BSD license. In order to be able to use managed code, OpenCV is used via EmguCV[4], a wrapper for OpenCV written in C#.

Figure 4.4 shows the main interface of the ITU Gaze Tracker. The main window displays the image grabbed by the camera, and a cross on the estimated pupil center. The 'pupil detector' slide bar allows to adjust the threshold level for the pupil detection. A similar setup screen is available when glint tracking is activated.

Software Components

The architecture of the software application is divided into three components: the gaze tracking library (written as a DLL and developed by the author of this thesis), the camera class (developed by Henrik Skovsgaard), and the user interface (UI, developed by Martin Tall). The UI communicates with the gaze tracking library using Windows events. The gaze tracking library uses the camera class to obtain the images from the camera.

The gaze tracking library includes methods to start the tracker, extract eye features, run a calibration, estimate gaze coordinates, and detect the type of eye movement the user is performing. Some of the methods are overloaded in order to allow for different hardware configurations; for example, depending on the number of light sources employed the calibration and gaze estimation procedures are different.

[4]http://www.emgu.com

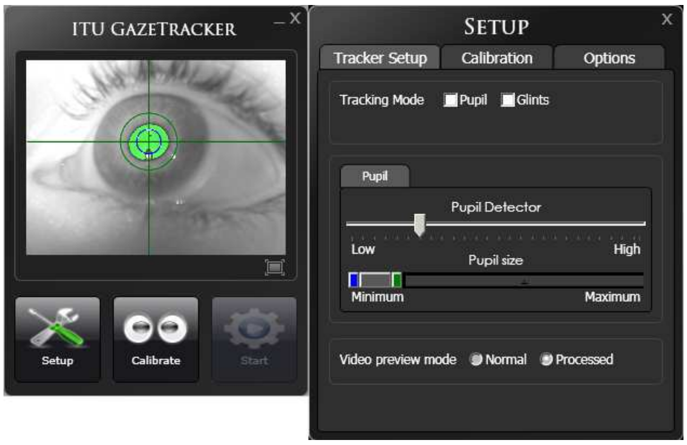


Figure 4.4: ITU Gaze Tracker main interface. The *Tracker Setup* tab allows to configure the parameters for pupil and glint detection.

Figure 4.5 shows a detailed diagram of the architecture. The UI instances the Tracker class, and the gaze tracking software automatically starts grabbing and processing images. The user interface allows to setup different configuration options and start a calibration procedure, control the cursor position using the estimated gaze coordinates, etc.

Eye Tracking Methods

The ITU Gaze Tracker can track the pupil and one or two corneal reflections (also known as glints). The pupil is detected by thresholding the image with the value given by the slide bar *Pupil Detector*. Increasing and decreasing that slide bar will modify the pupil threshold. As shown in Figure 4.4, a green pupil is sought after. The center of the pupil is then estimated and a cross hair is drawn on the image.

If desired, corneal reflection tracking can also be enabled. In the current release it is possible to track one or two corneal reflections. A different threshold controls the glint detection. It is assumed that the corneal reflections produced by the light sources are the ones closest to the pupil, and therefore reflections far from the pupil are regarded as undesired.

The glint detection procedure can easily be extended and modified to suit specific hardware configurations and make it more robust. For example, if the light sources are placed on a horizontal line, the corneal reflections on the image would appear on a somewhat horizontal line (distorted due to the curvature of the eye). This could be used as a constraint to reject other possible glint configurations that occur due to other light sources.

Since the hardware setup is completely flexible, no knowledge about the locations

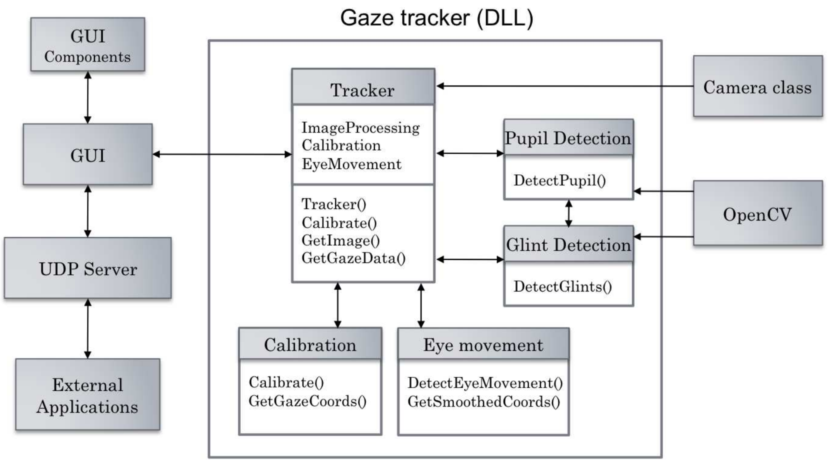


Figure 4.5: Software architecture of the ITU Gaze Tracker.

of camera and light sources with respect to the screen can be assumed. Therefore, the gaze estimation technique employed must be as generic as possible, and should not assume any specific configuration of the hardware components. The gaze estimation method implemented is an interpolation technique (see Chapter 7) and uses a pair of second order polynomial equations as described in [56]. During the calibration procedure, the coefficients of the polynomial equations are calculated, and subsequently used to estimate gaze.

Using an interpolation technique with one or more corneal reflections to estimate gaze provides some degree of head-pose tolerance in a remote setup; however, large head movements will affect the estimated gaze coordinates. Part III describes and evaluates a novel technique to estimate gaze that can be used in remote setups where no information about the hardware configuration is available.

Eye Movement Detection

A gaze tracking system usually provides a noisy signal, due to inaccuracies in the extraction of the eye features and the constant movement of the eye, even during fixations (see Chapter 2). When the user is fixating on a point, the estimated gaze coordinates calculated from consecutive images usually jump around that point. In order to remove this jitter and make the cursor more stable, a fixation detection algorithm has been implemented. When a fixation is detected, the signal is smoothed over a number of samples. When a saccade is detected, the smoothing is stopped to improve the responsiveness of the system. Fixation detection algorithms are discussed in more detail in Chapter 11. A method based on eye velocity and angular dispersion has been developed and integrated into the ITU Gaze Tracker.

Chapter 5

Performance Evaluation of Gaze Input

The present chapter introduces the theory underlying the performance evaluation of a pointing device in target-acquisition tasks, and describes some of the eye-typing applications and the metrics used to measure input performance.

Graphical User Interfaces (GUIs) are often based on interactive elements that the user has to select, such as buttons, menu items or hyper links. Interaction is usually performed by means of a pointing device, for example a mouse or a touchpad. The process of interacting with an object can be divided into two subprocesses: (1) a pointing process, during which the user moves the pointing element (e.g., the on-screen cursor) and places it on the desired object; and (2) a selection process, in which the user activates the object, for example by clicking with a mouse button. These kinds of tasks are called *point-and-select* or *target-acquisition* tasks.

There exist other more complex tasks, such as drag-and-drop, in which the user points at the desired object, selects it and maintains the selection button activated while the object is 'dragged' by moving the pointer. Once the final desired location is reached, the button is released, 'dropping' the object. In essence, this is the concatenation of two successive point-and-select tasks.

Eye typing represents one of the most common ways to interact using the eye movements. Input speed is crucial for people with severe motor-skill disabilities, and therefore a big effort by the gaze tracking community is devoted to improving eye-typing applications.

The next section introduces the performance evaluation in target-acquisition tasks, while Section 5.2 describes some of the existing eye-typing applications and the metrics used to evaluate typing performance.

5.1 Target-Acquisition Tasks

This section presents the fundamental theory underlying the performance evaluation of a pointing device and describes the previous work on evaluating gaze interaction.

5.1.1 Fitts' Law

Modern human-computer interaction (HCI) has a great interest in modeling and predicting human performance that can help to design better input devices and interfaces. In 1954, Fitts [19] described his model of human psychomotor behavior, introducing the idea that, when carrying out a movement task, the human motor system can be considered analogous to a communication channel of limited capacity [48]. The model is derived from Shannon's Theorem 17 on the transmission of information, which expresses the effective information capacity C (bits/s) of a communication channel with bandwidth B (Hz) as follows:

$$C = B \log_2 \frac{S+N}{N} \tag{5.1}$$

where S is the signal power and N is the noise power. The effect of the noise is to limit the information capacity of the channel below its theoretical maximum.

Fitts investigated the capacity of the human motor system, which he called index of performance (IP). IP is calculated as the ratio between the index of difficulty ID of a motor task and the movement time MT required to complete the task:

$$IP = \frac{ID}{MT} \tag{5.2}$$

Equation 5.2 is analogous to Equation 5.1, with IP matching the channel capacity C, ID matching the log term, and MT matching the inverse of the channel bandwidth, i.e. $1/B$. The index of difficulty of a task is measured in bits and is given by the following expression:

$$ID = \log_2 \left(\frac{A}{W} + 1 \right) \tag{5.3}$$

ID depends on the distance to the target, or amplitude A, and the width of the target measured along the axis of movement, W. The ratio between A and W within the logarithm is without units, and after applying the base 2 logarithm units of bits are obtained. Equation 5.3 was proposed by MacKenzie in [47] and is a reformulation of the original formula proposed by Fitts that presents a more accurate analogy with Shannon's information theorem. Furthermore, it avoids having negative IDs.

Equation 5.2 can be rewritten so that the predicted variable is the movement time MT, giving

$$MT = \frac{ID}{IP} \tag{5.4}$$

The IP can be determined as in Equation 5.2, or as a regression of MT on ID, which gives the following equation of a line

$$MT = a + b\,ID = a + b \log_2 \left(\frac{A}{W} + 1 \right) \tag{5.5}$$

where a and b (intercept and slope, respectively) are regression coefficients that are calculated empirically. The reciprocal of the slope, $1/b$, corresponds to the IP in Equation 5.4. The intercept a is usually considered to include additive factors unrelated to the index of difficulty. For example, target-acquisition tasks include a selection operation performed by pressing a button that contribute to the intercept but not to the slope of the regression line.

In 2000, the ISO 9241 - Part 9 standard based on Fitts' Law was introduced. It establishes the guidelines for evaluating computer input devices in terms of performance and comfort. The metric used to measure the performance is *throughput*, with units of bits/s. It combines both the speed and the accuracy of the input device. The equation for throughput is based on the IP in Fitts's Law, but it uses an *effective* index of difficulty (ID_e) instead of the standard ID, giving the expression

$$Throughput = \frac{ID_e}{MT} \tag{5.6}$$

where ID_e is determined as follows:

$$ID_e = \log_2 \left(\frac{A}{W_e} + 1 \right) \tag{5.7}$$

ID_e is calculated using the *effective width* W_e instead of the nominal width of the target W. W_e is derived from the distribution of movement endpoints, or 'hitpoints'. The underlying information theory assumes that the signal is perturbed by white Gaussian noise. The analogous requirement in target-acquisition tasks is that endpoints follow a Gaussian distribution. By normalizing the output measures with W_e, we take into account what the users actually did (i.e., distribution of movement endpoints) instead of what was expected (i.e., nominal target width), therefore incorporating the variability in performance across users [89].

Assuming a Gaussian distribution of endpoints, we can calculate the entropy, or information, which is given by $log_2(\sqrt{2\pi e}\sigma) = \log_2(4.133\sigma)$, where σ is the standard deviation of the distribution. The constant 4.133 can be splitted into a pair of z scores of 2.066; 96% of the total area of the gaussian unit curve ($\sigma = 1$) is bounded by -2.066 $< z <$ +2.066. This means that when the distribution of endpoints is normal, 96% of the hitpoints fall within the target, while 4% fall outside the target. In this case, $\log_2 W$ is an accurate representation of the information contained in the distribution of hitpoints. Fitts's Law assumes such a consistent error rate of 4% to maintain the analogy with the information theory. When the error rate exceeds 4% the effective target width is greater than W, and if it is lower than 4% the effective target width is less than W. Therefore, if an error rate different from 4% is observed, the width of the target needs to be adjusted to the effective target width.

W_e can be calculated in two ways. If the endpoints coordinates are known, W_e is determined as follows:

$$W_e = 4.133 \times SD \tag{5.8}$$

where SD is the standard deviation of movement endpoints across participants measured along the line from the origin of movement to the center of the target. If the endpoints are not known, W_e can be derived from the error rate in order to adjust the target width for a deviation from the nominal 4%. If an error rate of p percent is observed, a z score is calculated so that the interval $\pm z$ contains 100 - p percent of the area under the unit Gaussian curve. W_e is then obtained by multiplying W by $2.066/z$. It should be noted that the correction for the target width should be performed for each condition and each subject, because it makes use of within-subject variability [73].

Soukoreff and MacKenzie proposed seven recommendations to use Fitts' Law to evaluate the performance of pointing devices [73]. The analyses conducted in this thesis have followed all recommendations except for the one concerning using a wide range of indexes of difficulty. They suggest using a range of IDs from 2 to 8 bits. High IDs are difficult to achieve with gaze pointing, since the accuracy is significantly lower than the accuracy of a mouse, and therefore rather big targets must be used.

5.1.2 Target-Acquisition Tasks in the ISO 9241 - Part 9 Standard

To measure the performance of a pointing device, a target-acquisition task is commonly employed. This task requires the participants to point at a target and select it as quickly and accurately as possible. The ISO 9241 - Part 9 standard suggests using a circular array of targets in 16 different directions, as shown in Figure 5.1. The target width W is given by the diameter of the target. The amplitude A corresponds to the radius of the circle, that is, the distance from the 'home position' to the target. Both W and A need to take different values to ensure a wide range of indexes of difficulty in the task.

The order in which the targets are displayed can be randomized to avoid learning effects. The different conditions given by the different W's and A's should also be counterbalanced, for example following a balanced Latin square.

5.1.3 Previous Research

Fitts' Law and the ISO 9241 - Part 9 standard have been extensively used to evaluate the performance of pointing devices. Since the adoption of the ISO 9241 - Part 9 standard, the results obtained by different researchers have become more homogeneous, allowing for a better comparison across studies [73].

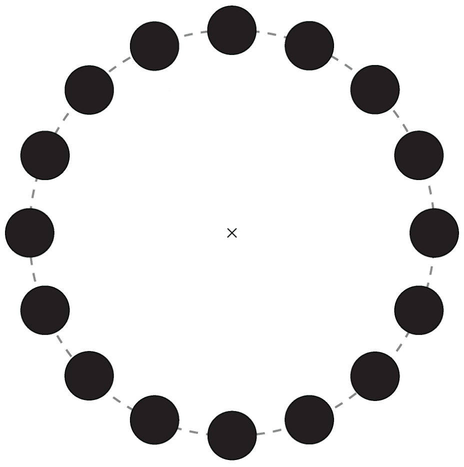


Figure 5.1: Circular layout of the 16 targets in a target-acquisition task.

Very few investigations have been conducted to evaluate gaze pointing in target-acquisition tasks using Fitts' Law. In 1987, Ware and Mikaelian [88] conducted the first study of gaze pointing under the Fitts' Law framework. They evaluated the movement time and error rate of a gaze tracker and compared its performance with a mouse. Three selection methods were considered: dwell (400 ms), a physical button, and an on-screen button to confirm a selection. Average movement times were below 1000 ms for the three activation techniques, with dwell and physical button being faster than the on-screen button.

Twenty years later, Zhang and MacKenzie [95] conducted the second evaluation of gaze pointing using Fitts' Law. Following the ISO 9241 - Part 9 standard, they studied the throughput of a gaze tracking system with three different selection methods: short dwell (500 ms), long dwell (750 ms) and space bar. The throughput obtained when using gaze with the space bar was close to the throughput of the mouse, although the error rate was significantly higher.

MacKenzie et al. proposed seven new metrics to measure performance in target-acquisition tasks [49]. These metrics can help explain some of the differences in performance obtained using different input devices; however, the authors argue that the new metrics should complement but not replace the traditional measures.

5.2 Eye-Typing Tasks

While using our eyes for pointing can be regarded as a substitution for the mouse, the accuracy provided by most eye trackers is not as good as the accuracy of the mouse. Interfaces specifically developed to be controlled by gaze can potentially exploit the characteristics of the eye movements more efficiently than normal applications and thus provide a more user-friendly way of interaction using our eyes. An example of gaze-controlled interfaces is eye-typing applications.

Hutchinson et al. [31] introduced in 1989 one of the first gaze tracking systems that included an eye-typing application, the Erica system. The user was presented with an on-screen keyboard consisting of six buttons. Letters were organized in a tree-structure hierarchy, and up to three selections were required to select a single character. The system had a character prediction feature to speed up the process. Although the typing performance was not evaluated, the authors wrote that "it can take an experienced user nearly 85 minutes to enter an entire page of text".

The development of more accurate gaze tracking systems has permitted more on-screen buttons to be displayed, thus decreasing the number of selections required for each character and increasing the typing speed. Much progress has also been done toward improving the efficiency of the eye-typing applications themselves. Reducing the number of the required keystrokes per character and including letter and word predictions allow the users to increase the typing speed.

The most common metric to measure performance of eye-typing applications is *words per minute* (WPM), where one word is considered as any five characters including spaces (this definition allows comparing the typing performance across different languages). Typical typing speeds obtained with a hands-on QWERTY keyboard are in the range of 40 to 60 WPM. It must be noted that some studies that use non-letter based languages (e.g., Japanese) report characters per minute (CPM), [29].

Soukoreff and MacKenzie proposed two metrics to measure the amount of errors, the *minimum string distance* (MSD) and the *keystrokes per character* (KSPC) [71]. The MSD measures the number of primitives (insertions, deletions and substitutions) required to transform the typed sentence into the correct sentence. A weakness of this metric is that it does not take into account corrected errors, as they will not appear in the final sentence. The KSPC is the ratio between the number of keystrokes employed and the minimum number of keystrokes necessary to type the sentence. For example, a user committing no errors while typing on a keyboard requiring one keystroke per character (e.g., a QWERTY keyboard) would obtain a KSPC of 1.00. The lowest KSPC possible (when no errors are committed) would be greater than one for keyboards requiring more than one keystroke to type each character (e.g., numeric keypad on a mobile phone). The higher the KSPC, the more errors and corrections were made. In a later work, Soukoreff and MacKenzie combined both metrics into the total error rate (TER) [72]. Both corrected and non-corrected errors are taken into account in the TER measure.

In 2001, Hansen et al. introduced GazeTalk [28], an eye-typing applications frequently used by people with severe disabilities. The interface presents an on-screen keyboard with the buttons arranged on a 4×3 grid, with the 6 most probable characters given the input text, and features word prediction (see Figure 5.2). The big buttons partly compensate for jitter and calibration offsets, and makes the system usable in noisy environments. However, the tree-structure organization of the characters decreases the input speed as up to 3 keystrokes might be required to type a single character.

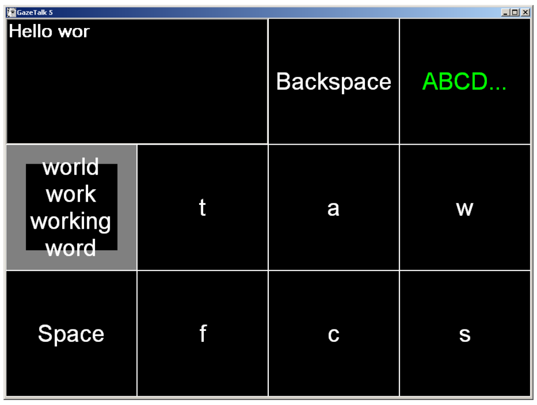


Figure 5.2: GazeTalk interface. The characters are arranged on a grid that shows the 6 most probable letters and the predicted word(s).

In 2000, Ward et al. presented Dasher [86], an eye-typing application in which characters are selected in continuous time instead of by point-and-click. The characters move from right to left until they cross the selection area. The user points at the desired character to select it (see Figure 5.3). It must be noted that this paradigm does not require an explicit selection mechanism. A probabilistic model is employed to modify the size of the characters: more probable characters are bigger than the less probable ones. A longitudinal study by Tuisku et al. [78] reported average typing speeds of up to 17 WPM after 2.5 hours of practice.

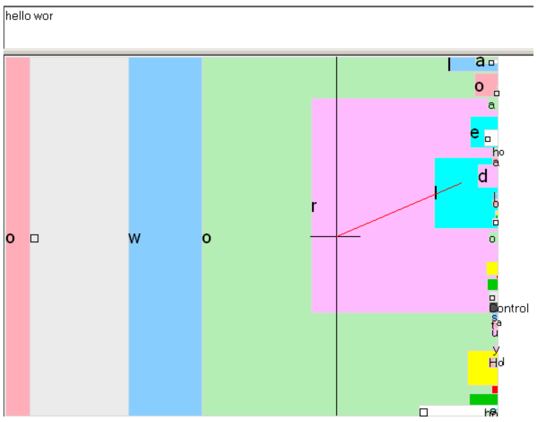


Figure 5.3: Dasher interface. Characters flow from right to left. The size of the containing box increases with the probability of the letter.

Hansen et al. presented in 2008 StarGazer [27], which takes Dasher's paradigm into a three-dimensional space. The user is presented a ring of characters in 3D space, as shown in Figure 5.4. The system zooms into the character currently pointed by the user; when collision occurs, the character is selected and a new ring of characters is displayed. Zooming allows to perform robust selections when targets are small [4]; this is of special importance when a low-accurate gaze tracker is used. An average typing speed of 4.7 WPM was reported using StarGazer without language modeling [27].

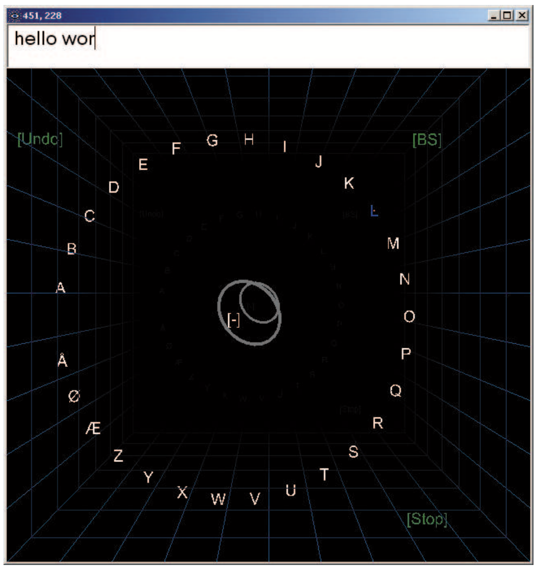


Figure 5.4: StarGazer's 3D interface. The letters are arranged on a circular layout. The system zooms into the pointed letter.

Other paradigms that do not include on-screen keyboards have been proposed. Isokoski presented a system that uses gaze gestures to input text [32]. The user types a character by looking at a specific sequence of off-screen targets. Two are the main advantages of the proposed method: (1) the typing application does not take any space on the screen, and (2) the gaze tracker employed needs not be highly accurate. However, in order to type the full alphabet the user is required to learn a high number of gesture sequences. The author does not report a performance evaluation of the technique in terms of WPM.

Inspired by Isokoski's work, Wobbrock et al. carried out a longitudinal study where they compared the performance of an on-screen keyboard with EyeWrite, a gesture-based typing system [90]. Figure 5.5 shows the gestures required to type each character of the alphabet. The authors found the on-screen keyboard to be significantly faster than the gesture-based technique, 7.03 vs 4.87 WPM, with no differences on the total error rate. Although no advantage of the gesture-based was found, it must be noted that the on-screen keyboard used requires a highly accurate gaze tracking system, whereas gesture-based typing interfaces do not.

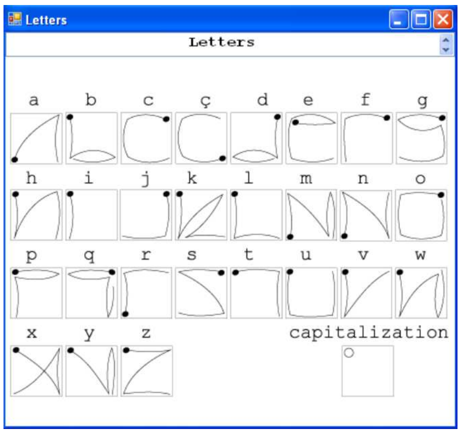


Figure 5.5: EyeWrite's alphabet. Each character is assigned a specific gesture.

The metrics described in this chapter are used to measure the performance of the ITU Gaze Tracker and compare it with other commercial systems in the next chapter, and to investigate the combination of gaze pointing and EMG clicking in Chapter 13.

Chapter 6

Assessing Low-Cost Gaze Interaction

The performance of the webcam-based system described in Chapter 4 has been evaluated in order to assess the feasibility of using low-cost components in gaze interaction. This evaluation is carried out in two different scenarios: a desktop scenario and a mobile scenario.

Section 6.1 presents a study where the ITU Gaze Tracker is used to interact with a desktop computer. The performance of the system is measured in the two tasks described in the previous chapter: target acquisition and eye typing. The section is concluded with the description of a pilot experiment carried out in the "real world" with Birger Jeppesen, the person with ALS introduced in Chapter 1. Section 6.2 investigates the potential of using the ITU Gaze Tracker to interact with a head-mounted display in a mobile scenario. A target-acquisition task is used to measure the performance.

Some of the results presented in this chapter have been previously published by San Agustin et al. in [66] and [64].

6.1 Desktop Scenario

The performance of the ITU Gaze Tracker has been evaluated when the interaction takes place on a desktop monitor. It must be noted that the system under evaluation is the head-mounted system presented in Section 4.2, in which a webcam is mounted on a piece of balsa wood that the user bites, thereby giving a hybrid desktop/head-mounted setup as shown in Figure 6.1.

In order to reduce the noise introduced during the extraction of eye features, the webcam has been placed very close to the eye, thereby providing a large image of the eye region. In effect, this is equivalent to using a zooming lens that is employed

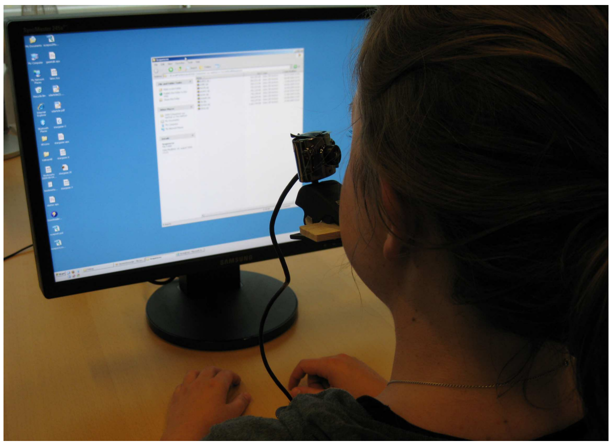


Figure 6.1: Head-mounted version of the ITU Gaze Tracker used in a desktop scenario.

in most commercial systems. The accuracy of the head-mounted system has been informally measured in such setup by calibrating on a set of 16 points, and running a test sequence afterwards. The measured average accuracy when the head is still is 0.53°, a value that is comparable to the typical average reported by commercial gaze tracking systems of around 0.5°. However, it must be noted that commercial systems allow for head movements, while the webcam-based ITU Gaze Tracker is not head-pose invariant, and therefore head movements introduce calibration offsets. This is explained next.

6.1.1 Lack of Head-Pose Invariance

In remote gaze tracking systems the camera is usually in a fixed position with respect to the screen. However, the user is often allowed to move freely, as long as the eyes are kept within the field of view of the camera[1]. Depending on the gaze estimation technique employed, head-pose invariance can be achieved, meaning that head movements will not affect the estimated gaze coordinates. A higher knowledge on the location of the different hardware components will allow to extract more information from the eye images, which can help to provide a head-pose invariant system (see Chapter 7).

The webcam-based system proposed in this thesis is not head-pose invariant: since the camera is mounted on the user's head, the two will move together. Therefore, a relative movement of the head/camera with respect to the screen will take place under head movements. As a result, when the user maintains his gaze on the same point on the screen and moves the head, the estimated gaze coordinates will be affected. Furthermore, the effect is counter-intuitive: a movement of the head to the right will

[1]Some systems include tilt units that rotate the camera to keep the user's eye within the field of view of the camera.

move the cursor to the left. The reason for this is that the relative movement of the pupil in the image when looking left and when moving the head to the right is the same.

Figure 6.2 illustrates this behavior. In the left column the user is looking straight at the center of the screen, with the head in the same position as during calibration. When the user looks left (middle column), the pupil moves in the image. As a result, the cursor (shown in red) moves to the location where the user is looking. If the user maintains the point of regard on the center of the screen and moves the head to the right (right column), the pupil moves in the same fashion as when looking left. Therefore, the cursor will move to the left. A similar effect can be observed when the user moves the head up and down.

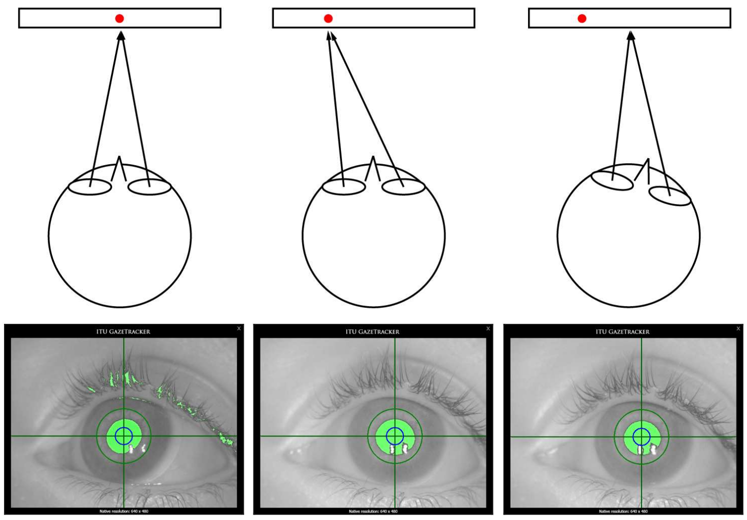


Figure 6.2: Behavior of the estimated PoR (shown as a red dot) when the user looks at the center of the screen with head still (left column), looks left with head still (middle column), and looks at the center of the screen but moves the head to the right (right column). Note that the two pictures to the right are similar to each other, while the one on the left is different.

As a consequence of this behavior, it is possible to compensate for inaccuracies of the gaze tracker by performing slight corrections of the head position. When an offset is noticed, the user can move the head to adjust the cursor position to the location where he or she is actually looking.

A simple way to deal with the lack of head-pose invariance is to require users to keep their head very still during a session. This can be achieved by using a bite bar or a very comfortable chair with a headrest. A bite bar ensures that the head remains completely still, thus increasing the internal validity of the results. However, in a real scenario users would not accept such an invasive setup. Using a chair with a headrest provides a more natural situation. Users will, however, perform slight head movements that will introduce an offset in the estimated gaze coordinates. This offset can be compensated for by the user by performing controlled head movements.

6.1.2 Performance Evaluation in a Desktop Scenario

An experiment was conducted to explore the possibilities of the ITU Gaze Tracker to interact in a desktop setup in two different tasks: target acquisition and eye typing. The performance of the ITU Gaze Tracker was compared to two commercial systems and a mouse. The target-acquisition task was performed as described in Section 5.1. In the eye-typing tasks, participants used StarGazer [27] to type a set of sentences they were given.

Participants

A total of five volunteers (four male and one female), ranging from 26 to 49 years old, participated in the experiment. All of them had previous experience with gaze interaction and the eye-typing application used in the experiment.

Apparatus

The target-acquisition task and the StarGazer typing interface were displayed on a 19" screen with a resolution of 1280×960. The three gaze trackers tested as input devices were the ITU Gaze Tracker, an SMI IViewX RED and a Tobii 1750, while the mouse was a Dell optical mouse. In the case of the ITU Gaze Tracker, the camera was mounted on a piece of balsa wood that the user bit to maintain it fixed to the head. Figure 6.3 shows the experimental setup.

As we will see in Chapter 11, it is convenient to include fixation detection algorithms that smooth the cursor position when the user fixates on a single point. This makes the cursor more steady and less jittery. However, if the smoothing is not implemented properly, small delays could affect the responsiveness of the system. In order to maximize throughput, all smoothing was eliminated where possible to avoid lag, i.e. only raw data of the estimated gaze coordinates was used to control the cursor. This was possible on the SMI and the webcam-based gaze tracker. The Tobii system was set to maximum speed, but some unknown automatic smoothing was performed during fixations.

Design and Procedure

To measure performance, participants completed two different tasks in the experiment: target acquisition and eye typing. The experiment was conducted employing a within-subjects full factorial design. The factors under study were *input device*

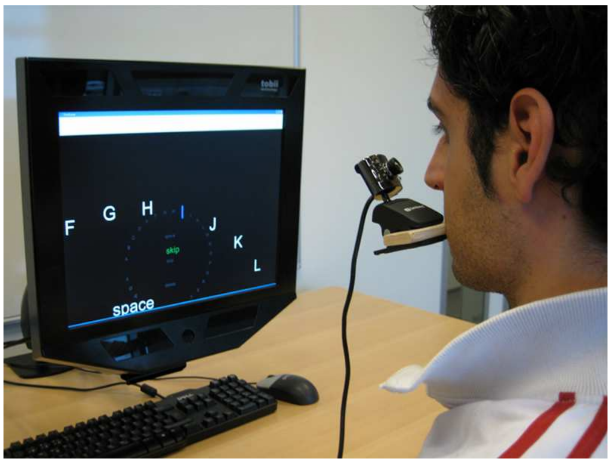


Figure 6.3: Experimental setup. The screen shows StarGazer on the monitor (which corresponds to the Tobii 1750 system).

(mouse, SMI, Tobii and webcam-based eye tracker), and, in the case of the target-acquisition task, also *target size* (75 and 150 pixels; roughly 2 and 4 degrees of visual angle, respectively). The orders of input device and task were counterbalanced across users to neutralize learning effects. The cursor was always visible to the user.

The target-acquisition tasks required the participants to point at a target as quickly and accurately as possible and activate the mouse button to select it. In each trial, completion time and erroneous selections were measured. Sixteen targets were arranged in a circular layout with a radius of 375 pixels. Targets could be 75 or 150 pixels in diameter; therefore, the nominal indexes of difficulty were 1.8 and 2.6 bits (calculated using Equation 5.3). The performance metrics measured were *completion time*, *throughput* and *error rate*.

The eye-typing tasks required the participants to type 5 random sentences using StarGazer. The sentences were taken from [50] and translated into Danish language. Prior to starting the experiment, participants had time to read the 5 sentences. After typing each sentence, participants selected the "Stop" button at the lower right-hand corner of the StarGazer interface, and the next sentence was then shown on the text field on top of the interface. Participants could take breaks between sentences, but no recalibrations of the gaze trackers were performed. The selection time was fixed to 1500 ms for all users. The performance metrics were WPM and TER (see Chapter 5).

Results in the Target-Acquisition Tasks

Analysis of the target-acquisition task was performed using three 4×2 within-subjects ANOVAs, with input device and target size as the independent variables. Completion time, throughput and error rate were analyzed as the dependent variables. All data were included in the analysis.

Overall mean throughput was 2.92 bits/s. Input device did not have a significant effect on throughput, F(3, 12) = 3.14, $p > 0.05$. The maximum throughput was obtained using the webcam-based gaze tracker (M = 3.33 bits/s, SD = 0.68), and minimum using the SMI system (M = 2.12 bits/s, SD = 0.72). Target size had a significant effect on throughput, F(1, 4) = 11.51, $p < 0.05$. Throughput was higher for large targets (M = 3.09 bits/s, SD = 0.56) than for small targets (M = 2.76 bits/s, SD = 0.49). Figure 6.4 shows the mean throughput obtained with each input device.

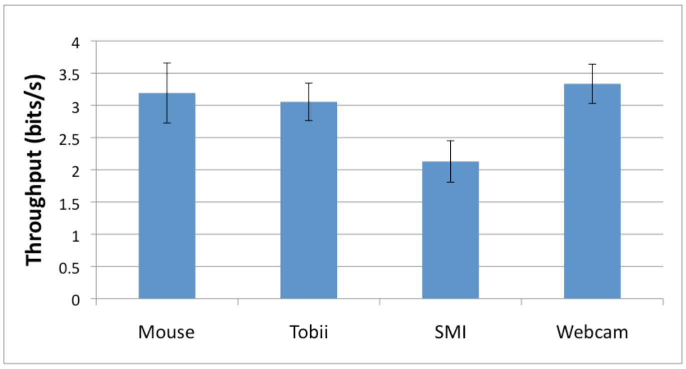


Figure 6.4: Mean throughput for each input device in a target-acquisition task. Error bars show the standard error of the mean.

Overall mean completion time was 825 ms. Input device did not have a significant effect on completion time, F(3, 12) = 0.59, $p > 0.05$. The webcam gaze tracker had the lowest completion time (M = 751 ms, SD = 91) while the mouse had the highest (M = 891 ms, SD = 216). Target size had a significant effect on completion time, F(1, 4) = 14.47, $p < 0.05$. Completion time was lower for large targets (M = 756 ms, SD = 91) than for small targets (M = 893 ms, SD = 117).

Overall mean error rate was 26.4%. Input device had a significant effect on error rate, F(3, 12) = 22.63, $p < 0.05$. Error rate was lowest using the mouse (M = 5.0%, SD = 4.2), and highest using the SMI system (M = 53.8%, SD = 12.8). Target size had a significant effect on error rate. Error rate was lower for large targets (M = 8.7%, SD = 6.8) than for small targets (M = 44%, SD = 6.01).

Figure 6.5 shows the completion time vs error rate for the 4 devices. Performance is best towards the origin of coordinates. Completion time was similar for all devices, but mouse had a lower error rate. On the other hand, the SMI system had a high error rate. As mentioned above, smoothing was deactivated in the SMI and the webcam-based systems in order to maximize the responsiveness (it is not possible to deactivate the smoothing in the Tobii system, and therefore its speed setting was set to highest). Deactivating the smoothing algorithm had a negative impact on the performance of the SMI system, affecting the accuracy and consequently increasing the error rate. A similar performance to the Tobii system could be expected when using the default settings.

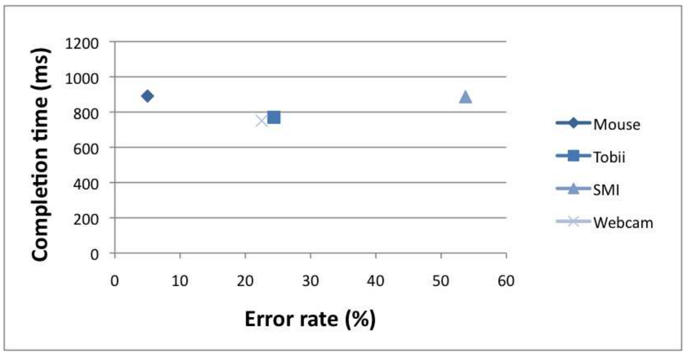


Figure 6.5: Completion time vs error rate for each input device in a target-acquisition task.

The throughput of 3.33 bits/s obtained with the webcam-based system is similar to values obtained in previous studies on gaze input performance. Zhang and MacKenzie [95] performed a similar evaluation and obtained a throughput of 3.78 bits/s when using a key for activation. In their experiment they used a *ViewPoint* system, a head-fixed gaze tracker that is not tolerant to head movements. They used a chin-rest to ensure that participants did not move with respect to calibration position, whereas in this work participants had the ability to move the head. The error rate obtained with the webcam-based gaze tracker was higher than the error rate obtained in the study by Zhang and MacKenzie (22.5% vs 16.94%). Calibration offsets introduced by participants moving their heads can help explain this difference.

The throughput obtained with the mouse (3.19 bits/s) is rather low, as it is usually in the range of 4 to 5 bits/s [73]. A possible explanation of this low value might be the big targets used in this experiment. Completion time should decrease as target size increases, but the observed completion times for mouse are rather high despite the large targets. Therefore, the calculated throughput decreases.

Results in the Eye-Typing Tasks

Analysis of the eye-typing task was performed using two one-way ANOVAs with input device as the independent variable. Words per minute (WPM) and total error rate (TER) were analyzed as the dependent variables. All data were included in the analysis.

Overall mean WPM was 4.68. There was no significant effect from input device on WPM, $F(3, 12) = 1.93$, $p > 0.05$. Mean WPM was highest with mouse ($M = 4.94$, $SD = 0.43$) and lowest with SMI ($M = 4.52$, $SD = 0.63$). Figure 6.6 shows the mean WPM for each input device.

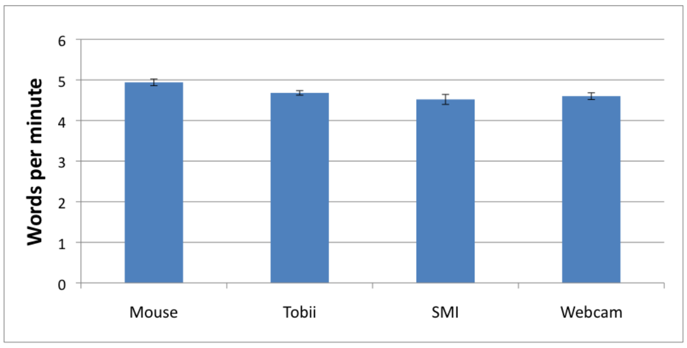


Figure 6.6: Mean WPM for each input device in an eye-typing task. Error bars show the standard error of the mean.

Overall mean TER was 0.47%. There was no significant effect from input device, F(3, 12) = 0.66, $p > 0.05$. Mean TER was lowest with the low-cost gaze tracker (M = 0.25%, SD = 0.9) and highest with mouse (M = 0.69%, SD = 1.7). Figure 6.7 shows the mean TER for each input device.

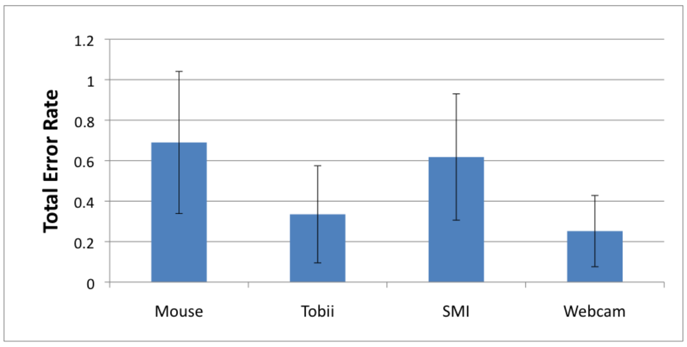


Figure 6.7: Mean TER for each input device in an eye-typing task. Error bars show the standard error of the mean.

The words per minute obtained in the eye-typing task was similar for the four devices. The overall mean WPM of 4.68 is consistent with the value of 4.70 reported by Hansen et al. [27]. However, it is low compared to other studies ([51], [78], [87]), where typing speeds above 15 WPM are reported when using other typing applications. There are two main reasons that explain the difference in typing performance. First, adding a language model allows for higher input rates, and in this experiment no letter or word prediction has been used. Second, the selection time of 1500 ms employed in this test is very high, especially as participants got acquainted with the system. Furthermore, the total error rate values are very low compared to other studies (for example, Soukoreff and MacKenzie report an average TER value of 8% [72]). The high selection time of 1500 ms allowed participants to commit very few mistakes. A longitudinal study should be performed to explore the maximum typing performance that can be achieved as participants decrease the selection time and become more proficient.

6.1.3 Discussion and Limitations

The target-acquisition and eye-typing experiments indicate that the low-cost gaze tracker has a performance indistinguishable from commercial systems. The main limitation of the webcam-based system is the lack of head-pose invariance. However, it had no impact in the measured performance. Furthermore, participants would often use this as a feature and compensate for calibration offsets by making small adjustments of their head position at each selection. Nevertheless, in a normal every-day usage scenario it might be difficult to keep the head still for long periods of time.

To solve the lack of head-pose invariance, gaze information can be combined with head-pose information. This can be achieved by using a head tracker that gives the location of the head in 3D. By calibrating both gaze and head trackers, the system could detect when the user moves the head and modify the estimated gaze coordinates accordingly so the cursor remains where the user is looking. The feasibility of this solution is yet to be explored.

The webcam can also be mounted on a head-mounted display. The camera and the display are then fixed together and to the user's head, solving the head-pose limitation. Such a system would allow the user to interact with mobile devices hands-free and using eye movements only. Section 6.2 explores this possibility and presents a study in which a head-mounted solution is tested in a mobile scenario where the user is walking and interacting with the interface shown on a head-mounted display.

One situation where the head-pose invariance does not play a relevant role is when the user is unable to move the body, e.g. due to having a locked-in syndrome. The webcam can be mounted on the user's wheelchair or bed and placed close to the eye. Once the calibration procedure is completed, the accuracy is maintained due to the inability of the user to move (assuming that the camera has a fixed position). The next section describes a pilot study with Birger Jeppesen, a person with ALS, in which gaze interaction in such a scenario is explored.

6.1.4 Pilot Study with a Person with ALS

The study presented above has been carried out in a laboratory scenario. In order to investigate the feasibility of using the ITU Gaze Tracker by a person with ALS and to explore its limitations, a pilot study was conducted with Birger Jeppesen. As described in Chapter 1, Birger has suffered from ALS for 12 years, and his eyes represent his only means of communication.

Everyday communication with his family and caretakers is based on face-to-face interaction. Birger and his helpers and close relatives have memorized a spelling grid consisting of letters grouped in numbered rows. The individual communicating with him goes through each row number, and Birger selects the row where the desired letter is by looking up. The helper then goes through the individual letters of that group until the preferred one is found: this way Birger spells out his intentions. A large part of the communication is based on the helper's ability to predict meaning and sentences.

Birger can also use a computer-based gaze tracking system that allows him to type using his eyes, although nowadays he uses it infrequently because he becomes tired quickly. His preferred gaze tracking system is called QuickGlance, and it costs around $10.000. Birger has been lucky enough to be provided with this commercial gaze tracking system by the local health authorities.

Birger tested the ITU Gaze Tracker and gave feedback to contribute in the development process. The webcam was mounted on the arm of an office lamp, which in turn was mounted on Birger's wheelchair. Figure 6.8 shows Birger lying on his wheelchair with the webcam pointing toward his eye. Figure 6.9 shows the interface of the eye-typing application that he normally uses, GazeTalk [28] projected on a wall of his living room.

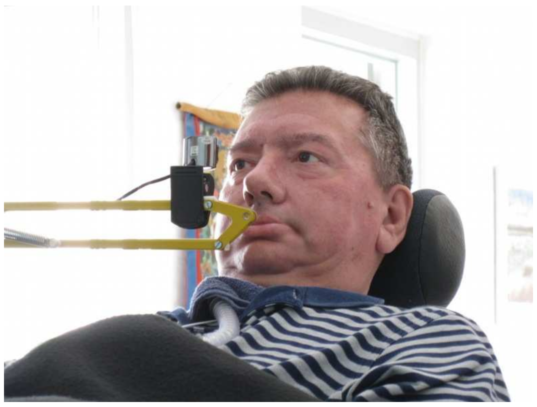


Figure 6.8: The webcam is mounted on the arm of an office lamp and placed close to Birger's eye.

As can be depicted from Figures 6.8 and 6.9, the hardware setup of the system is simple and flexible. The camera was easily placed in front of Birger's eye by attaching it to the wheelchair. This opens the possibility for gaze interaction in scenarios where the rigidity of commercial systems might pose a problem. For example, if a patient is lying in a hospital bed, placing a computer screen in front of him or her might be challenging. On the contrary, placing a webcam close to his or her eye and projecting the interface on a wall or on the ceiling becomes a feasible solution. In the case

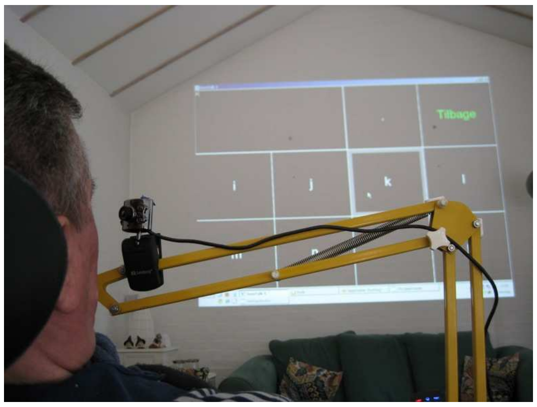


Figure 6.9: Birger sits on his wheelchair and types on the wall-projected interface using his eye movements.

of Birger, the wall-projected interface allows for more social interaction, as family members and friends can see what he is writing in real time, thus eliminating the need for a speech synthesizer or another person reading aloud what he writes.

Birger was able to use the ITU Gaze Tracker in combination with GazeTalk to type different sentences using his eye movements. Although the system was not as stable as the commercial gaze tracker he normally uses, he saw a great potential of such low-cost system for people who cannot afford or do not have access to commercial systems. More thorough experiments with people with ALS and other severe motor-skill disabilities should be conducted in order to compare the performance and the suitability of the ITU Gaze Tracker to other systems available in the market.

6.2 Mobile Scenario

Head-mounted displays are becoming increasingly popular as a means to obtain on-the-spot information in the fields of medicine, entertainment, augmented reality, maintenance or telerobotics [46], [77], [58]. In some scenarios the user needs to have access to certain information right in front of his or her eyes. For instance, a technician repairing a defective wire in a building can benefit from looking at maps and diagrams of the electrical installation on a head-mounted display, offering him the possibility of accessing important information without moving. During an operation, a doctor might need to look at different images and information of the patient being operated, and having them at a glance on a head-mounted display might be more efficient than turning towards a desktop computer.

Displaying the information right in front of the user's eye(s) holds interesting potentials, but it also creates a need for an efficient way to interact with the display. In a mobile scenario, the hands might be required for other tasks, and the use of standard input devices such as keyboard and mouse becomes awkward and sometimes not feasible. Gaze tracking can potentially provide a hands-free pointing technique that allows for information access on-the-move. However, few studies investigate the possibilities of using gaze pointing in head-mounted displays. Nilsson [58] presented an augmented reality system where the interface was controlled by eye movements, but no results on the performance were reported in her paper. Tanriverdi and Jacob [77] studied the use of eye tracking in a virtual reality (VR) environment, and gaze-based pointing was found to be faster than hand-based interaction. Duchowski et al. [16] built a similar VR system to investigate the effect of training in aircraft inspection. In this case, the eye movements were analyzed *a posteriori* and not used to control the interface.

This section aims to provide insight on the possibilities of using gaze input to interact with the information displayed on a head-mounted display in a mobile scenario. The study is conducted using the low-cost, webcam-based ITU Gaze Tracker.

6.2.1 Mobile Gaze Interaction: Challenges and Workarounds

The system in this mobile scenario consists of two different components: the gaze tracker and the head-mounted display. Ideally, these two components would be integrated into one system. In order to allow for easy construction of the system, the hardware modifications were kept at a minimum.

The head-mounted display used was a Vuzix DV920. It is a binocular display that can be connected to a standard computer or a video device such as an iPod. It weighs around 100 grams and provides a resolution of 1024×768 pixels. The 2.5 cm thick frame prevents ambient light from reaching the user's eye, eliminating a high number of undesired reflections on the sclera and iris produced by disturbing light sources. However, it is still possible for the user to see the surrounding environment by looking above or below the display.

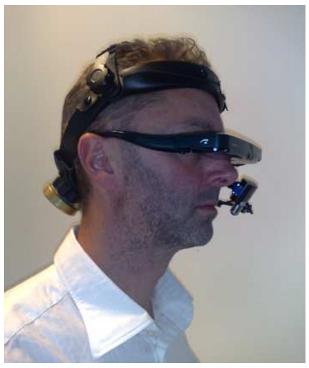


Figure 6.10: The user wears a head-mounted gaze tracker and interacts with the information shown on the head-mounted display.

A great challenge in a mobile setup is to mount the camera in a stable position to record the user's eye movements while not slipping as the user moves. Relative movements between camera and eye will be recorded by the gaze tracking software as an eye movement, and will therefore affect the estimated coordinates on the screen. An old ASL head-mounted eye tracker was disassembled and only the strap helmet and a small arm was kept. The webcam was mounted on this arm and placed so it pointed at the user's right eye, as depicted in Figure 6.10. The arm has enough degrees of freedom to allow for a flexible and stable positioning of the camera.

6.2.2 Performance Evaluation in a Mobile Scenario

An experiment was conducted to evaluate the potential of the low-cost, head-mounted gaze tracker in a mobile scenario. Participants interacted with the head-mounted display and completed a target-acquisition task (see Section 5.1) both standing still and walking at constant speed on a Nautilus treadmill. Interaction was performed by means of two pointing methods, gaze and mouse.

Participants

A total of six male volunteers, with ages ranging from 24 to 49 years, took part in the experiment. None of them used glasses but two used contact lenses. Four of them had previous experience with gaze tracking.

Apparatus

Figure 6.11 shows the equipment used in the experiments. As mentioned above, the webcam was mounted on a strap helmet. Participants had to interact on a Vuzix DV920 binocular head-mounted display. A Logitech MX Air mouse was used for mouse pointing. This mouse has a built-in gyroscope that tracks movements in 3D space, allowing for a full 6 degrees of freedom interaction (similar to the one found in the Nintendo Wii remote controller). Therefore, the user could control the cursor without the need for a flat surface.

Design and Procedure

The experiment was conducted using a 2×2×2 within-subjects full factorial design. Factors were *position* (standing or walking at a constant speed of 3.5 km/h), *pointing method* (MX Air mouse or gaze), *and target width* (150 or 200 pixels). The orders were counterbalanced across users according to a balanced Latin square to neutralize learning effects.

Prior to testing, the experimenter explained the purpose of the experiment to the participant and, in the case of the novice participants, how to use eye movements to control the cursor. Participants had the opportunity to take a sequence of warm-up

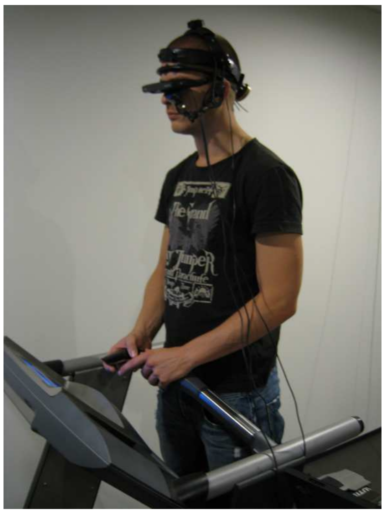


Figure 6.11: Experimental setup of the mobile gaze interaction evaluation. The user wears a head-mounted gaze tracker and interacts with the information shown on a head-mounted display while walking on a treadmill.

trials in an office to get acquainted with the system. Afterwards, the experiment was conducted at the university gym.

The task required the participants to point at a target as quickly and accurately as possible and activate the mouse button to select it. In each block, 16 data points were presented randomly for each of the 2 target sizes, one for each of the 16 possible directions, as specified in Section 5.1. The distance to the targets was fixed at 200 pixels; therefore, the nominal indexes of difficulty were 1 and 1.2 bits (see Equation 5.3). Each participant completed a block of trials for each position and pointing method, i.e. a total of 4 blocks. For each trial both *completion time* and *unsuccessful activations* were measured. Throughput was calculated using Equation 5.6.

Results

Overall mean throughput was 1.12 bits/s. Mean throughput was significantly higher when participants stood still (M = 1.37 bits/s) than when they walked (M = 0.93 bits/s), F(1, 5) = 15.65, $p < 0.05$. There was also a significant effect of pointing device, with throughput of mouse pointing (M = 1.4 bits/s) being higher than gaze pointing (M = 0.94 bits/s), F(1, 5) = 7.11, $p < 0.05$. Target size had no impact on throughput, F(1, 5) = 0.07, $p > 0.05$. Figure 6.12 shows the throughput for each combination of pointing method and position.

Overall mean completion time was 1063 ms. Mean completion time was lower when participants stood still (M = 927 ms) than when they walked (M = 1199 ms), F(1,

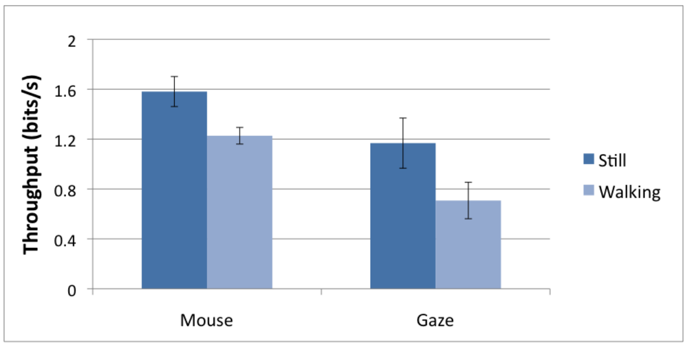


Figure 6.12: Throughput for mouse pointing and gaze pointing in still and walking situation. Error bars show the standard error of the mean.

5) = 10.21, p < 0.05. Pointing method had no significant effect on mean completion time, F(1, 5) = 2.47, p > 0.05, although gaze was slower than mouse (1234 and 893 ms respectively).

Similar results could be found for error rate, which was significantly lower when participants stood still (M = 13%) than when they walked (M = 26%), F(1, 5) = 29.45, p < 0.05. Pointing device had an effect on error rate, F(1, 5) = 13.83, p < 0.05, with error rate of gaze pointing (M = 31%) being higher than mouse pointing (M = 7%).

Figure 6.13 shows the completion time vs the error rate for each combination of pointing device and position. The error rate increases significantly when the participants were interacting with gaze while walking.

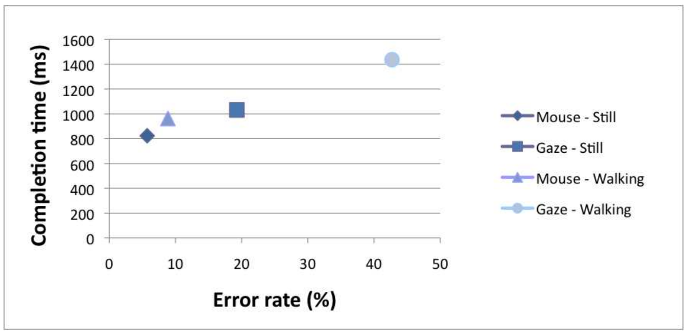


Figure 6.13: Completion time vs error rate for each combination of pointing device and position.

The throughput obtained in this mobile scenario was rather low (around 1 bit/s). Although completion times were not particularly high, the error rate was especially high when users were walking and pointing with their eyes. Interaction while walking represents a difficult task, and selecting even rather big targets is very challenging. Participants performed better when standing still, but the big target sizes and short distances made up for a task with a low index of difficulty. Thus, the throughput obtained was low compared to the 3.33 bits/s obtained in the study presented in the previous section.

6.2.3 Limitations of the System

Although the camera seems firmly attached to the strap helmet, it moves with respect to the eye when the user is walking. Even slight movements affect the estimated gaze coordinates, adding an offset that prevents the user from correctly selecting the target. A possible solution to partially solve this issue is to track the pupil and one or more corneal reflections instead of only the pupil center. Pupil and corneal reflections move differentially under eye movements, but simultaneously in case of headgear slippage.

The camera and the display are not attached to each other, and relative movements between both components take place as the user walks. This has a great effect on the cursor position in the current setup. A direct solution is to integrate camera and display into one unit, thereby eliminating these movements. However, such an approach might require complex hardware modifications, which would leave it out of the scope of this thesis.

A great drawback of the design proposed is the use of the strap helmet, which is not an off-the-shelf component. However, a similar system could be built using other mounting solutions, such as a pair of safety glasses or a bike helmet [2], [44].

6.3 Conclusions

The two studies presented in this chapter show the potential of the ITU Gaze Tracker for gaze interaction. The results obtained indicate that in a desktop scenario the webcam-based system described in Chapter 4 has a similar performance as two other commercial systems in both a target-acquisition task and an eye-typing task. The measured throughput, 3.33 bits/s, is comparable with the throughput obtained by Zhang and MacKenzie in [95], 3.78 bits/s. The measured words per minute in the eye-typing task is 4.6, a rather low value that can be explained by the long activation time of 1500 ms employed. However, no differences in typing performance are found in comparison with the two commercial gaze trackers included in the study.

Mobile interaction on a head-mounted display using a gaze tracker has proven to be a challenging task, and the throughput obtained is as low as 1 bit/s. In the prototype used in the study presented in Section 6.2 the camera is not attached to the display, and the movement of the body when the user walks produces relative movements between both components, thereby introducing big offsets in the estimated gaze coordinates.

The main benefits of the webcam-based system investigated in this chapter are the very low cost camera employed (around \$20) and the simplicity of the hardware setup. However, the system poses a big limitation: the estimated gaze coordinates are affected by head movements. Although in some situations this might not be an issue (e.g. when the user has a locked-in syndrome), a desktop gaze tracker should be tolerant to head movements to allow the user to move freely. Part III deals with gaze estimation techniques and presents and evaluates a novel method robust to head movements in remote setups.

Part III

Gaze Estimation Using Homographies

Chapter 7

Gaze Estimation

The results presented in Part II show the potential of the low-cost, webcam-based ITU Gaze Tracker to be used for gaze interaction. However, the system has an important limitation regarding head movements: since the camera is attached to the user's head, it moves together with it, and therefore head movements affect the estimated gaze coordinates. Moreover, the system is invasive due to the requirement of mounting the camera on the head.

A remote version of the ITU Gazer Tracker has been developed. It uses a high-resolution camera that is placed on the desk pointing at the user's face. An important characteristic of the system is the high degree of flexibility. As a result, no knowledge on the location of the hardware components can be assumed. To improve the accuracy and the tolerance to head movements, a novel gaze estimation algorithm is proposed in this thesis, extending the work by Hansen [23]. The performance of this new method is compared to other gaze estimation methods that have few or no assumptions on the locations of the hardware components.

This chapter introduces the theory of gaze estimation, with a special focus on techniques that do not require any knowledge on the hardware configuration. Chapter 8 describes the novel gaze estimation method based on homographies, and Chapters 9 and 10 present an evaluation of the technique using both simulated and real data, comparing its performance and limitations to two state-of-the-art methods based on the cross-ratio invariant of projective space and to an interpolation-based method.

7.1 Introduction

A video-based gaze tracking system pursues establishing a relationship between the eye features extracted from the image and the user's eye gaze by means of a gaze estimation algorithm. In general, this mapping can be expressed as a function from an m-dimensional space (eye features) to a 3-dimensional space (world coordinates) [23],

$$\Phi : \mathbb{R}^m \rightarrow \mathbb{R}^3 \tag{7.1}$$

When only the PoR on the screen is obtained, the mapping is from $\mathbb{R}^m$ to $\mathbb{R}^2$. Gaze estimation techniques can be divided into two categories:

- Interpolation-based techniques: they assume that Φ is a generic function that maps the eye features extracted from the images to the point of regard. This function is deduced after a user-calibration procedure.
- Geometry-based techniques: these methods use a model of the eye to provide information of the PoR or, in some cases, of the direction of gaze in space (the PoR is then calculated as the intersection between the gaze vector and the screen plane). Some of these methods require knowledge of the relative locations of the different hardware components of the setup.

Finding the function Φ is the core problem of gaze estimation, and it requires calculating a set of parameters by means of one or more calibration processes. Depending on the nature of these parameters, four types of calibration can be considered [24]:

- *Camera calibration*, i.e. the estimation of the camera parameters such as focal length or principal point.
- *Geometry calibration*, i.e. calibration of the relative locations and orientations in 3D space between the different components of the setup such as camera, light sources or screen.
- *Human-specific parameters*, such as cornea curvature or offset between optical and visual axes.
- *Model parameters*, such as regression coefficients in interpolation-based methods.

The characteristics and limitations of the gaze estimation method largely depend on how much information is extracted through calibration. Human-specific parameters and model parameters are usually estimated by having the user look at a set of points on the screen. This process is normally performed before a session starts. Camera and geometry calibration require physical measurements of the elements or the use of extra hardware such as camera calibration patterns or mirrors [30], [96], [68]. When a system requires both camera and geometry calibration the system is denoted as *fully calibrated*. In setups with a fixed geometry where the location of the components does not change with respect to each other geometry calibration needs to be performed only once. Calibration of the camera is required every time any of the parameters of the camera (e.g., the focus) is modified.

There is a trade-off between the characteristics of the model (e.g., regarding accuracy and head-pose invariance) and the hardware information available *a priori*: as more knowledge of the location of the hardware components is available, more information can be extracted from the image data, and therefore more complex eye models can be employed. In general, interpolation-based methods are less demanding in terms of hardware calibration, and no need for camera parameters or relative locations between components are necessary [56]. Geometry-based techniques, on the other hand, usually require camera and geometry calibration in order to estimate the parameters of the 3D eye model [21], [67], although some methods only require a partial geometry calibration [92]. The accuracy obtained by fully-calibrated methods is usually higher, and they are often more tolerant to head movements.

Head movements represent a challenge in gaze estimation. Many gaze tracking systems calculate the relative movement of the eye with respect to the head. However, we also move our heads while we look around. Therefore, the direction of gaze is determined by the orientation of the eye and the head pose. It is possible to achieve head-pose invariance in geometry-based methods. In the case of interpolation-based methods, the accuracy decays significantly with large head movements from the calibration position.

The eye features used to estimate gaze have an effect on the accuracy and the degree of head-pose invariance of the method. Contours, reflections on the cornea and eye corners are among the most common features extracted from the eye image. When active illumination is used, the pupil and the corneal reflections produced by the IR light sources are typically employed to estimate gaze.

7.2 Interpolation Methods

Interpolation methods assume that Φ is a set of generic expressions that map the eye features to the user's point of regard. These methods require to calculate the model parameters through a user-calibration procedure, in which a sequence of calibration points, usually arranged on a 3×3 or 4×4 grid, is shown on the screen. It is assumed that these generic expressions can model the behavior of the eye gaze for all points over the limited region of the screen, at least to a certain extent. The goodness of the interpolation largely depends on the function employed and the number of calibration points used to deduce the parameters of the function.

The function Φ maps the eye features to a two-dimensional space (the screen), and can be expressed as a linear combination of a set of eye features, as shown in Equation 7.2

$$\Phi = A \cdot E \tag{7.2}$$

where A represents the coefficients of the linear combination and E represents the features of the eye used in the regression. During calibration, a set of eye features corresponding to each sample point is extracted from the images. This information is then employed to calculate the coefficients A, which are used to estimate the PoR during a working session.

The most common functions used in interpolation methods are polynomial expressions, usually of second order degree. Although a higher degree might provide a better accuracy, it also requires more calibration points to properly calculate all the coefficients. Cerrolaza et al. [8] investigated polynomial regression varying the eye features used, the degree of the polynomial and the terms employed, and concluded that increasing the degree beyond second order does not improve accuracy. Since the calibration procedure is regarded as cumbersome for the user, a strong effort should be put into minimizing the number of calibration points.

Interpolation methods have been extensively employed for gaze estimation in systems where little or no information of the geometry of the different components is available, and where constructing a model of the eye is too difficult. Morimoto et al. [56] used a pair of second-order polynomials to map the pupil-corneal reflection vector (x_{pc}, y_{pc}) to on-screen coordinates (x_s, y_s), as expressed in Equation 7.3. The authors do not provide, however, an evaluation of the accuracy of this technique.

$$\begin{aligned} x_s &= a_0 + a_1 x_{pc} + a_2 y_{pc} + a_3 x_{pc} y_{pc} + a_4 x_{pc}^2 + a_5 y_{pc}^2 \\ y_s &= a_6 + a_7 x_{pc} + a_8 y_{pc} + a_9 x_{pc} y_{pc} + a_{11} x_{pc}^2 + a_{12} y_{pc}^2 \end{aligned} \tag{7.3}$$

Other interpolation methods include non-parametric forms such as neural networks. Zhu and Ji [97] use a generalized regression neural network to map pupil parameters to screen coordinates under active IR illumination. The advantage of the system is that it does not require user calibration. Although the method is head-pose invariant, the accuracy is between 5 and 8°, rather low compared to other regression methods where accuracies below 1° are reported [8].

7.3 Geometry-Based Methods

Geometry-based methods use a model of the eye and some information about the geometry of the system to estimate the user's gaze. Constructing a model of the eye and the hardware components might be more difficult than using generic expressions as in interpolation-based techniques. However, geometry-based methods generally provide more accurate gaze estimates and a useful theoretical basis to gain insight into the behavior of the gaze estimation technique. Modeling the system allows us to investigate the performance of the method and to predict theoretically the estimation errors.

Most geometry-based methods are fully calibrated, and make use of information about the hardware components of the system, such as camera parameters and the location of camera, light sources and screen with respect to each other [21], [81], [67]. The process to obtain this information involves camera and geometry calibrations. Since this represents a cumbersome process that needs to be performed every time the location of a component is modified, fully calibrated systems usually have all hardware components fixed.

Some geometry-based techniques use a model of the eye, but eliminate the need for camera and geometry calibration, thus increasing the flexibility of the system as the components can be placed in the most convenient location [92], [10]. Although a model and user calibration might still be required, these methods are known as *uncalibrated*.

7.3.1 Fully Calibrated Setups

Fully calibrated methods assume that all the hardware information is known. Therefore, before using the system the location of screen and light sources with respect to the camera must be measured (geometry calibration) and the camera parameters need to be calculated (camera calibration). Once this information is gathered, it is possible to estimate the Line of Sight (or visual axis) with respect to the camera. The on-screen coordinates are then calculated as the intersection between the 3D vector that represents the direction of gaze with the screen plane.

The model of the eye used to estimate gaze also requires calculating some human-specific parameters. Most characteristics of the eye are subject-dependent, and while some of them (e.g. optical axis, angular offset between visual and optical axes) can be calculated through a calibration process, others (e.g. refraction indices, cornea radius) can be difficult to measure for each individual, and anthropomorphic average values are often used [21].

A number of geometry-based methods have been described in the literature. Depending of the hardware employed methods have different characteristics: as the number of cameras and corneal reflections increase, features like head-pose invariance can be obtained. Guestrin and Eizenman [21] proposed a general model for gaze estimation in fully calibrated setups and analyzed different possible hardware configurations. A more exhaustive review of gaze estimation techniques can be found in [24].

One camera and no light sources

Although most gaze estimation methods employ one or more corneal reflections, there have been attempts to determine gaze without the use of light sources. Wang and Sung presented a method based on finding the 3D location of the irises of both eyes [85]. Assuming the iris is a circle, gaze direction is calculated as the vector normal to that circle. Later on, they improved the method to work with only one eye using some anthropomorphic values of the eyeball structure. Both systems need to be combined with head pose information. Villanueva et al. [84] estimate gaze direction from the image of the pupil, obtaining an accuracy of around 1° when the head remains still.

One camera and one light source

It is possible to determine gaze direction in fully calibrated systems employing one camera and one light source, assuming that the head is kept fixed or that the distance from camera to eye can be calculated [21]. However, most implementations of this setup assume that the head movements are negligible. Ohno et al. [59] use a model of the eye to calculate the optical axis and a 2-point calibration to correct for the angular offset. They use average values for cornea curvature and distance between pupil and cornea center, obtaining an accuracy below 1° when users keep the head inside a rather small area of 4 cm^2 at a constant distance from the screen.

One camera and multiple light sources

Guestrin and Eizenman [21] show that it is possible to determine the optical axis using two or more light sources and assuming that the cornea curvature and distance between pupil and cornea center are known. The angular offset can be calculated with a single calibration point.

Multiple cameras and multiple light sources

In the case where cornea curvature and distance between pupil and cornea center are unknown, the optical axis can be determined if at least two cameras and two light sources are employed [21]. The angular offset between visual and optical axes can then be determined using one calibration point.

Some methods that employ multiple cameras often use a wide field of view camera to track the position of the head, and a narrow field of view camera to extract the features of the eye, usually mounted on a pan-and-tilt unit in order to maintain the eye inside the field of view under large head movements [97], [5], [92].

Beymer and Flickner [5] presented a high-accuracy gaze tracking system that makes use of 2 sets of stereo cameras, one for facial tracking and the second one for eye tracking. Calibration of the stereo setups must be carried out prior using the system. Shih and Liu [68] reduced the hardware to a single pair of stereo cameras and determined the optical axis from the pupil center and three corneal reflections (two corneal reflections are sufficient, but a third light source is added to ensure that at least two corneal reflections appear on the image). One calibration point is used to calculate the angular offset between optical and visual axes. The accuracy obtained is below 1° of visual angle.

7.3.2 Uncalibrated Setups

Uncalibrated systems use, at least to some extent, information about the geometry of the system and a model of the eye in order to estimate gaze. However, the requirement for camera and geometry calibrations is eliminated. Since the 3D location of the hardware components is unknown, the estimation of gaze is performed in 2D and obtained in the form of the PoR coordinates on the screen.

Cross-Ratio Methods

The cross-ratio method is a geometry-based gaze estimation technique for uncalibrated setups. It was introduced by Yoo et al. in [93], where they described a new technique to map the center of the pupil in the image to a corresponding PoR on the screen. It is based on the cross-ratio, the only invariant of projective space. The main advantage of the method is that it does not require camera or geometry calibration while theoretically being head-pose invariant. Four LEDs are placed in the corners of the monitor, and a fifth LED is placed coaxial with the camera. In the first version of the method the fifth LED is only used to create a bright pupil effect. The LEDs are synchronized with the camera and dark and bright pupil images are captured alternatively. Ideally, the difference of two consecutive images contains only the pupil, which is therefore easy to segment.

Figure 7.1 shows a schematic diagram of the gaze estimation method. The four LEDs placed on the corners of the screen are projected onto the cornea, producing four corneal reflections. A virtual plane tangent to the cornea surface is assumed to exist, and the four corneal reflections ($\mathbf{v}_1$, $\mathbf{v}_2$, $\mathbf{v}_3$, $\mathbf{v}_4$) lie on this reflection plane. The polygon formed by the four corneal reflections is therefore the projection of the screen. A second projection takes place from the corneal plane to the image plane, obtaining the points ($\mathbf{u}_{v1}$, $\mathbf{u}_{v2}$, $\mathbf{u}_{v3}$, $\mathbf{u}_{v4}$) and the projection of the pupil center, $\mathbf{u}_p$.

When the user looks at the screen, the gazed point is projected onto the corneal plane by the same projection as the LEDs are projected on the corneal reflections. Therefore, the image of the pupil center can be used to estimate gaze. The property used by the method is the cross-ratio of 4 points on a line, which is invariant under a planar projective transformation [30]. The cross-ratio is defined as follows:

$$\text{Cross}\left(\mathbf{p}_1, \mathbf{p}_2, \mathbf{p}_3, \mathbf{p}_4\right) = \frac{|\mathbf{p}_1\mathbf{p}_2|\,|\mathbf{p}_3\mathbf{p}_4|}{|\mathbf{p}_1\mathbf{p}_3|\,|\mathbf{p}_2\mathbf{p}_4|} \tag{7.4}$$

where

$$|\mathbf{p}_i\mathbf{p}_j| = \det\begin{bmatrix} p_{i_x} & p_{j_x} \\ p_{i_y} & p_{j_y} \end{bmatrix} \tag{7.5}$$

In order to estimate the gaze coordinates, the center of the pupil needs to be mapped to the screen. To that effect, the method uses the relation between the cross-ratio of a configuration formed by the corneal reflections and the center of the pupil in the image, and the cross-ratio of the light sources and the (unknown) PoR.

The cross-ratio of the screen is calculated as follows:

$$CR^x_{screen} = \frac{\left(w - \frac{w}{2}\right)\hat{g}_x}{\left(w - \hat{g}_x\right)\frac{w}{2}} = \frac{\hat{g}_x}{w - \hat{g}_x} \tag{7.6}$$

where w is the width of the screen and $\hat{g}_x$ is the x coordinate of the estimated gaze point $\mathbf{g}$.

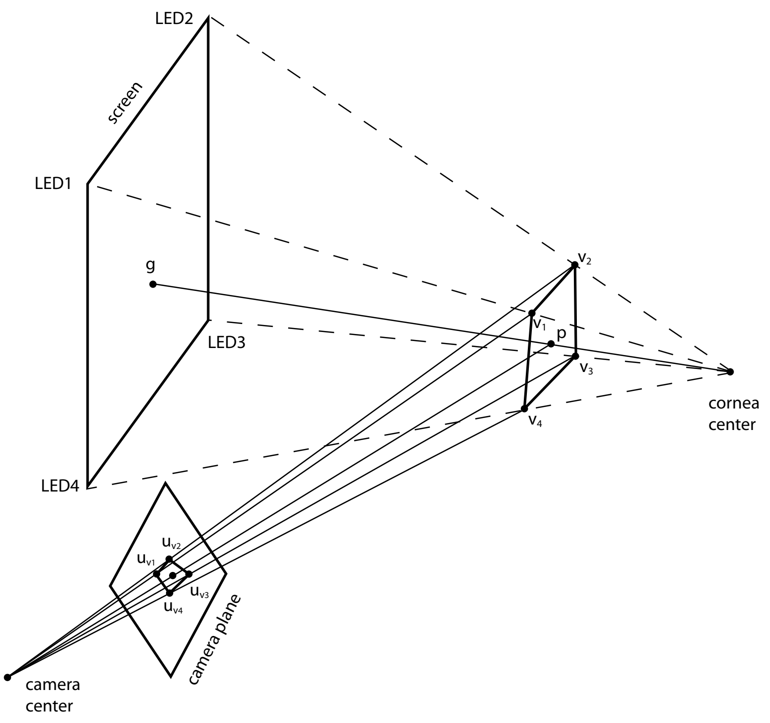


Figure 7.1: The four light sources are projected onto a reflection plane. The corneal reflections are then projected onto the image plane.

The cross-ratio of the virtual corneal reflections and the pupil is calculated as follows (see Figure 7.2 and refer to [92] for further details):

$$CR^x_{image} = \frac{|\mathbf{u}_{v_1}\mathbf{u}_{m_1}|\,|\mathbf{u}_{m_2}\mathbf{u}_{v_2}|}{|\mathbf{u}_{v_1}\mathbf{u}_{m_2}|\,|\mathbf{u}_{m_1}\mathbf{u}_{v_2}|} \tag{7.7}$$

Since the cross-ratio of both configurations have to be the same, it is possible to determine the PoR by means of Equation 7.8:

$$\hat{g}_x = \frac{w \cdot CR^x_{image}}{1 + CR^x_{image}} \tag{7.8}$$

A similar derivation gives the estimated y coordinate of the PoR, $\hat{g}_y$.

The original method by Yoo et al. [93] used the coordinates of the corneal reflections in the image to estimate gaze, following the calculations presented above. However, the method was shown to introduce large estimation errors for some subjects, [92], [10], [22]. To reduce the bias in the estimation of the PoR, error compensation techniques using a subject-specific calibration were proposed by Yoo and Chung in

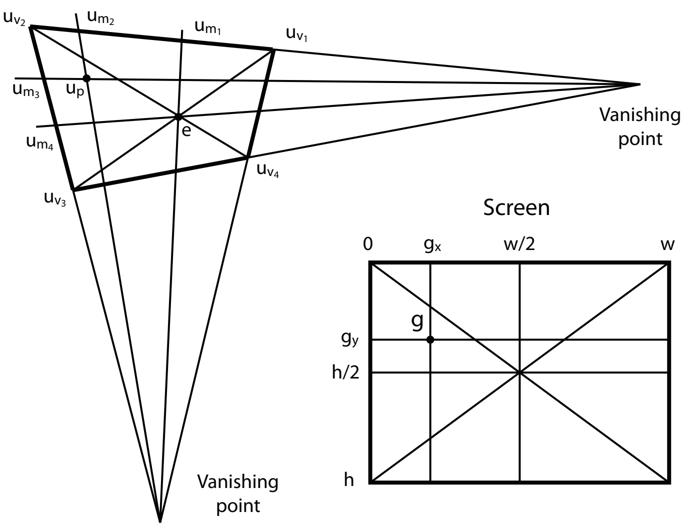


Figure 7.2: Cross-ratio of corneal reflections and pupil center in the image, and cross-ratio of the light sources and PoR on the screen.

[92] and by Coutinho and Morimoto in [10]. An investigation of the main causes of the estimation bias in the original cross-ratio method is presented by Kang et al. in [38]. Two main sources of error are identified: the angular offset between optical and visual axes, and the non-coplanarity between pupil and corneal reflections.

An enhancement of the method was presented by Yoo and Chung in [92], where the coordinates of the corneal reflections in the image are modified to compensate for gaze estimation bias. Information of the on-axis light source is used to calculate the *virtual corneal reflections* as follows:

$$\mathbf{u}_{v_i} = \mathbf{u}_c + \alpha_i(\mathbf{u}_{r_i} - \mathbf{u}_c) \tag{7.9}$$

where $\mathbf{u}_{v_i}$ are the coordinates of the virtual corneal reflection, $\mathbf{u}_c$ describes the coordinates of the corneal reflection generated by the on-axis light source, $\mathbf{u}_{r_i}$ represents the coordinates of the i*th* corneal reflection in the image, and α_i are four constant terms that control the magnitude of the change of the coordinates of each corneal reflection. The four α_i are subject-dependent and are calculated through a calibration process where the user is asked to look at the four light sources.

The original method assumes that pupil and reflection plane are coplanar. As shown by Kang et al. in [38], this correction of the corneal reflections provides a compensation for the error introduced by this assumption. The study carried out in Chapter 9 provides an insight into the effects of this assumption, and confirms the findings by Kang et al. [38].

The method proposed by Yoo et al. in [92] does not take into account the angular offset between optical and visual axes, which leads to a high error in the estimated gaze coordinates, as noted by Coutinho and Morimoto in [10] and by Kang et al. in [38]. Coutinho and Morimoto [10] extended the method to compensate for the offset between axes. They calibrate a single α parameter and an on-screen vector m_α, the error between the PoR estimated by the cross-ratio method and the calibration targets. The α parameter is calculated so that the error vector m_α is minimized. They report an accuracy of $0.5°$ in simulated data and around $1°$ in real data. It must be noted that since the compensation using m_α is introduced on the screen, the method is sensitive to head movements in depth, thus not being head-pose invariant.

Although in principle the cross-ratio methods do not require geometry calibration, it is assumed that the light sources are placed exactly on the corners of the screen, which can be difficult to achieve in a non-laboratory scenario. Furthermore, the methods require a symmetric configuration of the light sources. In case the lights are not placed on the corners of the screen, their position with respect to the screen would need to be measured, and the correct width and height (and possible offset from top left corner) should be introduced in Equation 7.8. Another limitation regarding the hardware employed is the need to place a light source on-axis with the camera. Although this is easily achievable in a laboratory setup, it can be difficult to build for an average user, since it requires attaching a ring of LEDs (light emitting diodes) to the objective of the camera.

The next chapter presents a novel geometry-based, uncalibrated method based on planar homographies. The method is analyzed and compared to the methods based on the cross-ratio presented above ([92], [10]) using simulated data (Chapter 9) and real data (Chapter 10).

Chapter 8

Eye Model Based on Homographies

In this chapter a novel method to estimate the PoR is described. The method is based on homographic mappings between camera plane, pupil/cornea plane, and screen plane, and is an extension to the work by Hansen [23]. The model is geometry-based and does not require camera calibration nor geometry calibration of the hardware setup.

A description of planar homographies is given in the next section. Section 8.2 presents the model of the eye based on homographic mappings. The chapter is concluded with a description of the assumptions made by the gaze estimation method. The effect of these assumptions on the accuracy of the gaze estimation method is investigated in the next chapter using simulated data. In this investigation the two uncalibrated geometry-based methods based on the cross-ratio described in Section 7.3.2 are also analyzed. The analysis of the estimation bias of the method is completed in Chapter 10 using real data.

8.1 Planar Homographies

A planar homography H is a projective mapping that maps points from one plane in 3D space to another. For any given point $\mathbf{x}$ on a plane $\mathbf{\Pi}$ in 3D space, it is possible to find the corresponding point $\mathbf{x}'$ on another plane $\mathbf{\Pi}'$ by means of a 2D homography H, as expressed in Equation 8.1. Figure 8.1 shows the geometry involved in such configuration.

$$\mathbf{x} = \mathbf{x}'H \tag{8.1}$$

H is a 3×3 matrix. Since it is defined up to scale, it has 8 degrees of freedom [30]. Given a set of four or more points lying on one plane and their corresponding

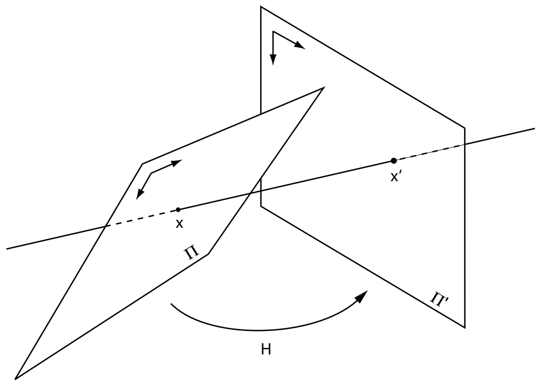


Figure 8.1: Any point $\mathbf{x}$ on plane $\mathbf{\Pi}$ is projected to a point $\mathbf{x}'$ on plane $\mathbf{\Pi}'$ by means of the homography H.

points on another plane, it is possible to determine the homography H that relates both planes, since each point-to-point correspondence accounts for two constraints.

8.2 Eye Model

The gaze estimation method presented in this chapter is based on homographic mappings between different components of the system. Since a model of the system is constructed, the gaze estimation method is geometry-based. However, no geometry calibration of the hardware components is required.

The system uses one uncalibrated camera and four light sources that can be placed at any location in space. The four light sources are projected onto the cornea surface. Although the cornea is a spherical surface, the method assumes that the four corneal reflections lie on the same plane, $\mathbf{\Pi_C}$, tangent to the cornea surface. The four corneal reflections are projected onto the camera image plane $\mathbf{\Pi_I}$ using the camera center as center of projection, as shown in Figure 8.2.

For any point belonging to a plane in 3D space, its corresponding point in any other plane can be calculated by means of a homography H. Therefore, there exists a homography that relates the image plane $\mathbf{\Pi_I}$ and the screen plane $\mathbf{\Pi_S}$. This homography from image to screen is denoted $\mathbf{H_{IS}}$. For any given point on the image, its corresponding point on the screen plane can be found provided $\mathbf{H_{IS}}$ is known.

Assuming a planar cornea, the four corneal reflections lie on the same plane, $\mathbf{\Pi_C}$. Furthermore, assuming that the formation of the corneal reflections is given by a projection of the light sources on the corneal surface with the cornea center as center of projection, light sources and corneal reflections are related by a projective transformation. Therefore, there exists a homography that maps points on the cornea to points on the screen. This homography is denoted $\mathbf{H_{CS}}$. The effect of both

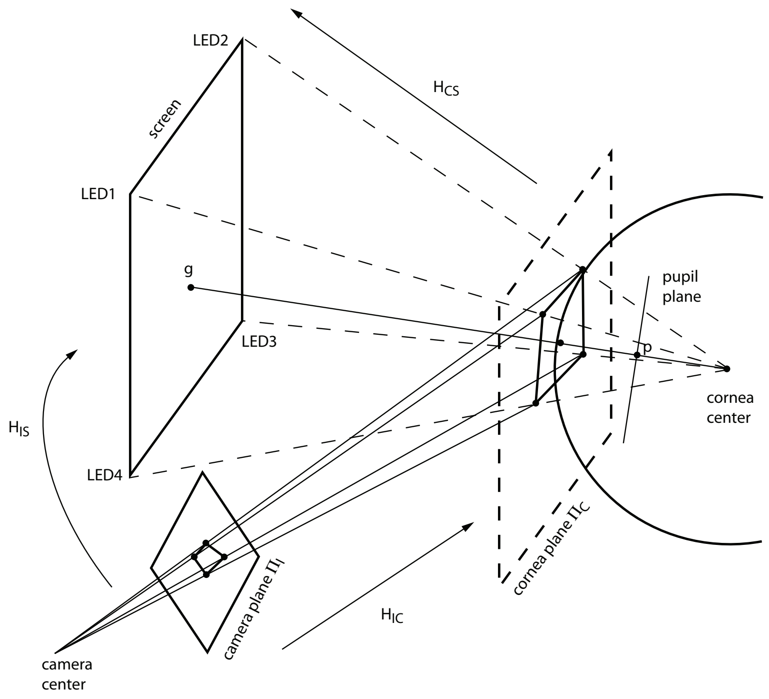


Figure 8.2: The four light sources are mapped onto the cornea plane by means of a homography. Another homography maps the corneal reflections onto the image plane. The concatenation of both homographies maps points on the image plane to points on the screen plane.

assumptions (planar cornea and projection instead of reflection) on the estimated gaze coordinates will be studied in the next chapter.

The corneal reflections and the pupil are projected onto the camera image plane by means of the camera projection matrix [30]. Since an uncalibrated setup is being considered, the camera projection matrix is not known. Assuming that both the pupil and the four corneal reflections lie on the same plane, the projection can be considered to be a homographic mapping from cornea plane in 3D space to the image plane. This homography is denoted as $\mathbf{H_{IC}}$.

The concatenation of two homographic mappings is also a homographic mapping. Therefore, the mapping from image to screen given by $\mathbf{H_{IS}}$ can be constructed by concatenating the mapping from image plane to cornea plane and the mapping from cornea plane to screen plane, $\mathbf{H_{IS}} = \mathbf{H_{IC}} \circ \mathbf{H_{CS}}$.

8.2.1 Estimation of the PoR

Gaze coordinates on the screen are estimated from the coordinates of the pupil center and the corneal reflections in the eye images. In order to estimate gaze, both homographic mappings $\mathbf{H_{IC}}$ and $\mathbf{H_{CS}}$ need to be determined.

Determination of mapping from image plane to cornea plane $\mathbf{H_{IC}}$

Since the 3D locations of the corneal reflections are unknown, the cornea plane $\mathbf{\Pi_C}$ cannot be determined, and therefore it is not possible to calculate the mapping from image plane to cornea plane, $\mathbf{H_{IC}}$. To help build the relation described in the previous section, an virtual *normalized plane* $\mathbf{\Pi_N}$ is considered to be located in 3D space with coordinates (0,0), (0,1), (1,0), (1,1). This plane is equivalent to the screen up to an affine transformation. The four corneal reflections in the image, u_1 to u_4, are mapped to the corners of this normalized plane, u_{n1} to u_{n4}, as displayed in Figure 8.3. The mapping constructed can be described by a homography denoted as $\mathbf{H_{IN}}$, and is calculated independently of the calibration points.

The pupil center $\mathbf{p}$ in the image can be mapped to normalized space using the mapping $\mathbf{H_{IN}}$, and therefore normalized with respect to the corneal reflections, obtaining the *normalized pupil center* $\mathbf{p_n}$.

Determination of mapping from cornea plane to screen plane

Once the center of the pupil in the image has been mapped to the normalized plane, the mapping from normalized plane to screen plane $\mathbf{H_{NS}}$ can be determined. This homography is used to map the normalized pupil center, $\mathbf{p_n}$, to a pair of coordinates on the screen, **gaze**.

A minimum of four corresponding points are needed on both planes in order to calculate the homography [30]. These corresponding points are determined by means of a calibration process where the user looks at four calibration targets on the screen. For each calibration point, the normalized coordinates of the pupil center are mapped to the screen coordinates of the calibration target.

Once both homographic mappings are determined, the concatenation $\mathbf{H_{IS}} = \mathbf{H_{IN}} \circ \mathbf{H_{NS}}$ is used to estimate the PoR coordinates on the screen. The process is illustrated in Figure 8.3.

8.2.2 Non-Linear Extension

The gaze estimation method described above is a linear model in which the PoR is determined as the concatenation of two consecutive homographic mappings between camera plane and normalized plane, and between normalized plane and screen plane. Therefore, the model assumes that all processes are linear. These assumptions are described in the next section and their effects analyzed in the next chapter.

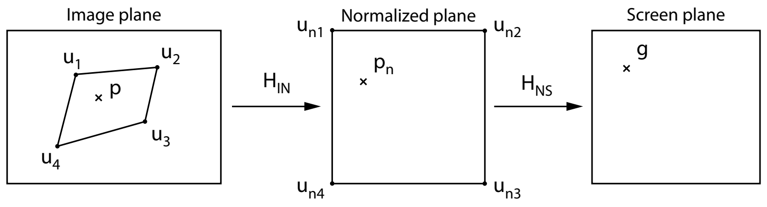


Figure 8.3: The pupil center $\mathbf{P}_c$ is mapped to the PoR coordinates on the screen, **gaze** by means of the concatenation of the 2 homographies $\mathbf{H_{IN}}$ and $\mathbf{H_{NS}}$.

A series of non-linearities that are not modeled will introduce gaze estimation errors. In order to partially compensate for these non-linearities, the model has been extended with an interpolation-based technique using a pair of third order polynomials like the ones described in Section 7.2, one for each axis. The PoR is estimated using a concatenation of homographic mapping and polynomial regression. The homography method described above is first applied, and a pair of on-screen coordinates obtained. A polynomial regression is then applied to correct for inaccuracies. Since the correction is applied on the screen, it is not scale invariant, and movements of the head in depth will affect the estimated gaze coordinates. The effect of head movements is investigated in Section 9.4.

The calibration procedure first calculates the homographic mapping using four calibration points placed on the corners of the screen, as explained in the previous section. Then, a set of points arranged in a grid is employed to calculate the coefficients of the polynomials.

The combination of homography and interpolation can provide a more accurate estimation of the gaze coordinates than the pure homography method when enough calibration points are used. However, it might be affected by the two main issues that affect pure interpolation methods (e.g., [56]): (1) a high estimation error when an insufficient number of calibration points is used and the coefficients of the polynomials cannot be properly estimated (usually 9 to 16 points are employed for calibration), and (2) the lack of head-pose invariance, as head movements introduce a notable estimation bias. As will be shown in Sections 9.3 and 9.4 in the next chapter, these two issues do not affect the combination of homography and interpolation.

8.3 Model Assumptions

The gaze estimation method based on homographies makes a series of assumptions and simplifications that result in an error in the estimated gaze coordinates. Chapter 9 presents an analysis of the effect of each assumption on the estimated PoR.

The homography method proposed in this thesis is compared to two other uncalibrated geometry-based methods, both of them based on the cross-ratio projective invariant and presented in Section 7.3.2. The first one is the method proposed by

Yoo et al. in [92], and will be referred to as the 'Yoo method'. The second method is the extension of the Yoo method proposed by Coutinho and Morimoto in [10], and is denoted as 'Coutinho method'.

Table 8.1 summarizes the assumptions made by each of the three methods, and shows whether the method compensates for the error introduced by each assumption or not.

	Assumed			Compensated for		
	Yoo	Cou	Hom	Yoo	Cou	Hom
Affine projection	✓	✓	✓	✗	✗	✗
Pupil and cornea are coplanar	✓	✓	✓	✓	✓	✗
Planar cornea	✓	✓	✓	✗	✗	✗
Corneal reflections generated by projection	✓	✓	✓	✓	✓	✗
No offset between optical and visual angles	✓	✓	✗	✗	✓	-
No refraction	✓	✓	✓	✗	✗	✗
Lights on screen corners	✓	✓	✗	✗	✗	-

Table 8.1: Assumptions and possible compensations made by the three uncalibrated geometry-based methods: 'Yoo method', 'Coutinho method' and homography method.

8.3.1 Affine Projection

The pupil is a circle in 3D space, and it is projected onto the image plane by means of the nodal point of the camera **O**. When the user is not looking directly at the camera, the pupil forms an ellipse on the image plane. The center of this ellipse is taken as the center of the pupil. As illustrated in Figure 8.4, in projective perspective it can easily be assumed that the projection of the pupil center in 3D space is not the center of the projection of the pupil in the image. The error between the center of the projected pupil and the projection of the center can introduce an estimation bias in the gaze coordinates.

In fully calibrated setups it is possible to unproject the pupil contour by means of the projection center of the camera **O** and calculate the center of the pupil in 3D space. This allows to calculate the projection of the pupil center. On the other hand, in uncalibrated setups the camera parameters are unknown and thus it is not possible to determine the correct pupil center. The accuracy of the cross-ratio methods and the homography methods will be affected by the difference between the projection of the pupil center and the center of the projection of the pupil.

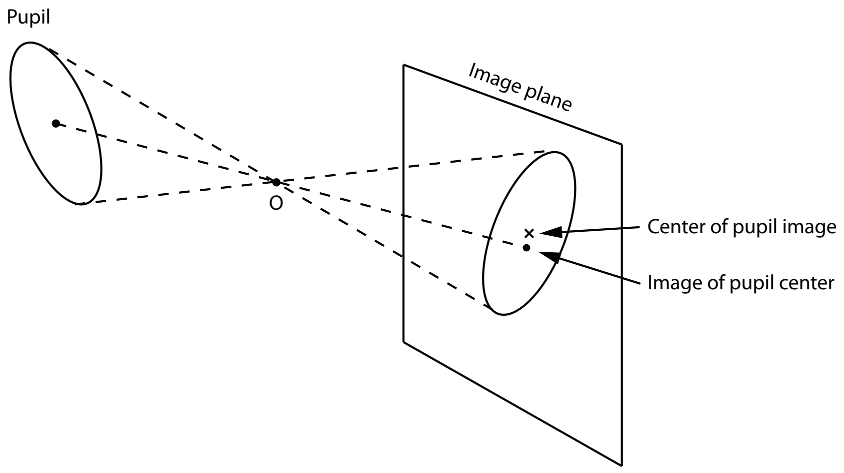


Figure 8.4: The pupil contour is projected onto the image plane by means of the projection center of the camera **O**. The calculated center of the projected ellipse does not coincide with the projection of the center.

8.3.2 Pupil and Cornea Are Coplanar

When the eye features are projected onto the camera plane, the image of the corneal reflections and the image of the pupil lie on the same plane, the image plane. However, in 3D space the corneal reflections and the pupil do not belong to the same plane. Even under the assumption that all corneal reflections are on the same plane, the pupil is located behind the cornea plane at a distance *d*. This is illustrated in Figure 8.5. The non-coplanarity of cornea plane and pupil plane introduces an error in the estimation of the gaze coordinates [22].

8.3.3 Planar Cornea

The cornea is approximately a sphere with a radius of 7.98 mm. The light sources are projected onto the cornea spherical surface by means of the center of curvature of the cornea. Therefore, it cannot be assumed that the four reflections lie on the same plane. However, the cross-ratios and the homography methods assume a planar cornea with all four reflections belonging to the same plane.

8.3.4 The Corneal Reflections Are Created by a Projection

A light source is reflected on the surface of the cornea and generates a corneal reflection $\mathbf{cr}$. This corneal reflection creates the glint $\mathbf{u_{cr}}$ on the image plane. However, the models assume that the corneal reflection is generated by a projection from the light sources onto the cornea surface using the cornea center as the center of projection,

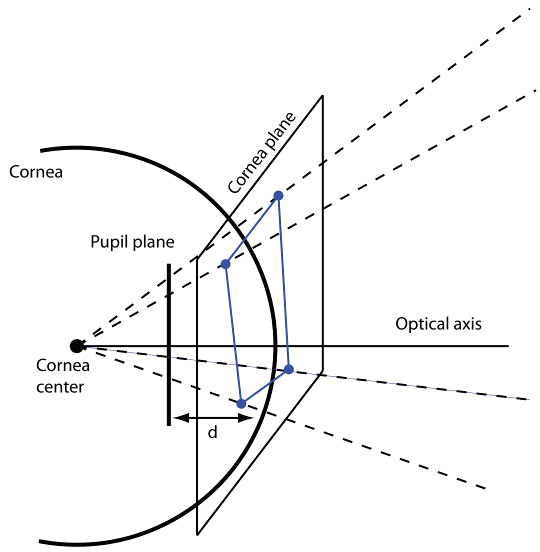


Figure 8.5: Assuming that the corneal reflections lie on the same plane, the pupil plane is located behind the cornea plane at a distance *d*.

thus creating point $\mathbf{v}$ in 3D space and $\mathbf{u_v}$ on the image plane. Figure 8.6 illustrates the difference between projection and reflection.

In order to use the cross-ratio property of projective space, the points must be generated by a projection. Therefore, the cross-ratio methods calculate the virtual point $\mathbf{v}$ of each corneal reflection. The four virtual points are used to calculate the cross-ratio, as explained in Section 7.3.2.

8.3.5 Optical and Visual Axes Are Collinear

As explained in Chapter 2, there exists an angular offset between the optical axis and the actual direction of gaze, the visual axis. The cross-ratios methods use the pupil center to estimate gaze. The pupil center is aligned with the center of curvature of the cornea and therefore lies on the optical axis. The PoR is defined as the intersection of the visual axis with the screen, and therefore an estimation error will be introduced by the offset between optical and visual axes.

8.3.6 No Refraction

The light entering and exiting the eye is refracted in the different layers of the eye due to the different indices of refraction. The main refraction takes place in the interface between air and cornea, which has an index of refraction of 1.376. An effective index of refraction of 1.3375 for cornea and aqueous humor is often considered [21]. Refraction

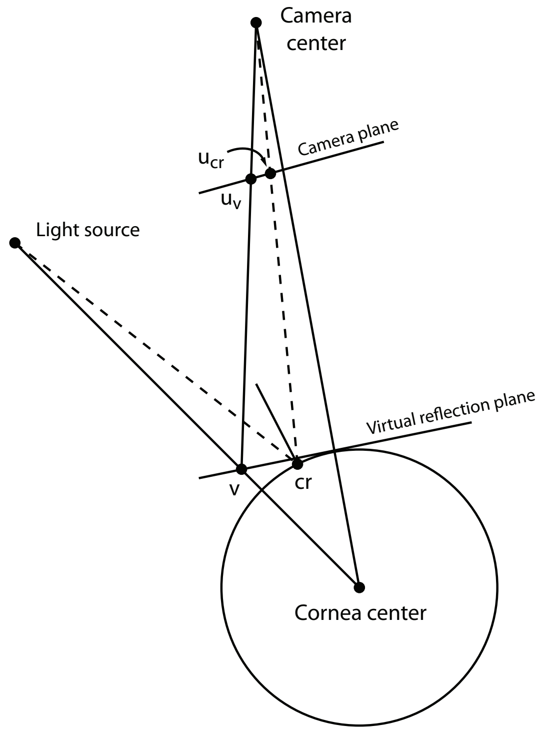


Figure 8.6: The light source is reflected on the cornea surface, creating the point $\mathbf{u_{cr}}$ on the image. However, the methods assume that the light source is created by a projection by means of the cornea center.

on a spherical surface such as the cornea is a non-linear process that makes the location of the pupil change non-linearly. Feature-based models, and in particular the cross-ratios methods and the homography method, do not model refraction explicitly.

8.3.7 Location of Light Sources

As explained in Chapter 7, the cross-ratio method exploits the relation between the shape formed by the light sources and the shape formed by the corneal reflections on the image plane. The cross-ratio is invariant in projective space, and therefore the cross-ratio of the light sources is the same as the cross-ratio of the images of the corneal reflections. In order to estimate gaze, the pupil center is mapped to the screen using both cross-ratios (screen and image). Therefore, the method assumes that the light sources are located exactly on the corners of the screen. If the light sources are placed somewhere else, a correction is required by means of a geometry calibration procedure.

Chapter 9

Eye Model Assessment on Simulated Data

This chapter presents an exhaustive analysis of the estimation bias of the homography method presented in Chapter 8. The performance of the method is compared to two state-of-the-art geometry-based gaze estimation methods that do not require geometry calibration: the Yoo method (as described in 2005 by Yoo and Chung [92]) and the Coutinho method [10]. Both methods were described in Chapter 7. The analysis has been conducted using the eye tracking simulator by Böhme et al. [6], developed in Matlab[1] and released under a GPL license. The software has been extensively modified to suit the requirements to conduct the different tests. Data in Section 9.2 have been generated using a mathematical model developed in Mathematica[2] by Arantxa Villanueva.

Section 9.2 investigates the effects of the assumptions made by the models on the estimation bias (see Section 8.3). The results allow insights into the behavior of the different models and help identify the main sources of error. Section 9.3 investigates the effect of increasing the number of calibration targets on the accuracy of each method, while Section 9.4 presents an analysis of the effects of head movements on the accuracy. In Section 9.5 a realistic scenario with slight head movements and measurement noise on the camera is introduced, and the performance of the gaze estimation methods evaluated. In these three tests the non-linear extension of the homography method presented in Section 8.2.2 is introduced in the analysis. For completeness, a pure interpolation method has also been analyzed, thus covering regression-based methods and uncalibrated geometry-based methods.

[1]http://www.mathworks.com/products/matlab
[2]http://www.wolfram.com/products/mathematica

9.1 Methodology

The on-screen accuracy was measured by calculating the average error and the standard deviation of the error for each gaze estimation method when looking at 256 locations of the screen arranged on a 16×16 grid. Tests have been conducted using two different hardware configurations, denoted as *pos0* and *pos1*. Figure 9.1 depicts the two hardware configurations employed.

- *pos0* is a symmetric configuration. The screen is a 300×300 mm square, the camera is located in the center of the screen, and the eye is placed at a distance of 600 mm from the screen and centered with respect to it. This allows to study the symmetry properties of the different methods.
- *pos1* represents a more realistic scenario. The screen has a 4:3 ratio, with a size of 400×300 mm. The camera is located below the monitor, closer to the user and slightly off-center.

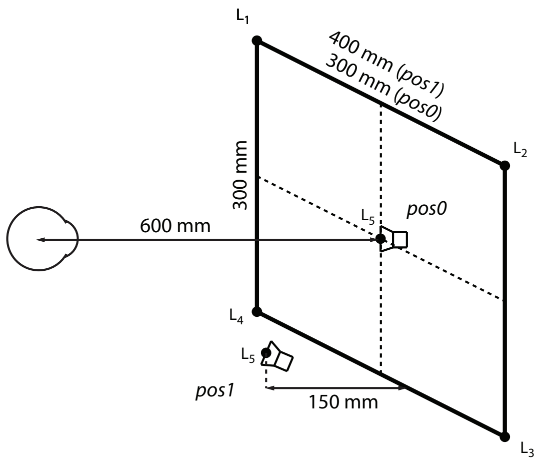


Figure 9.1: Hardware configurations employed (not in scale). In *pos0* the camera is centered on the screen, while in *pos1* the camera is placed closer to the user and off-center.

The cornea is modeled as a perfect sphere with a radius of 7.98 mm. The center of the eye globe is placed at a distance of 600 mm from the screen and centered with respect to it. The coordinate system of the eye is centered on the eye ball center. The eye points towards the negative Z axis. The following values for the eye parameters are given with respect to the eye coordinate system.

- Cornea radius = 7.98 mm
- Cornea center = [0 0 -4.35] mm
- Pupil radius = 3 mm

- Pupil center = [0 0 -8.79] mm
- $\alpha = 5^\circ$
- $\beta = 1.5^\circ$
- $n_{\text{effective}} = 1.3375$
- Camera focal length = 2880 pixels

9.2 Estimation Bias due to Model Assumptions

Eye models make assumptions about the characteristics of the eye. The assumptions made by the cross-ratios methods and the homography method were introduced in Section 8.3 and summarized in Table 8.1. In this section the effect of the different assumptions on the error in the estimated gaze coordinates is analyzed with the objective of identifying the main sources of error and investigating ways to compensate for such errors.

The analysis is carried out by building a series of "eye" models, with each model introducing a different assumption. A perfect model (denoted as Model0) where all the assumptions are met is taken as the initial step in the process. Each assumption is then introduced to the model and its effects on the gaze estimation error analyzed. Each subsequent model is therefore more realistic and provides a better model of a real eye.

The list of tested eye models is the following:

- Model0: perfect model.
- Model1: perspective projection is used instead of affine projection.
- Model2: the pupil is placed at its natural location, and therefore the pupil plane and the cornea plane are not coplanar.
- Model3: corneal reflections are calculated using reflection instead of projection.
- Model4: the cornea is modeled as a spherical surface and therefore the corneal reflections are no longer coplanar.
- Model5: the angular offset between optical and visual axis is introduced.
- Model6: refraction in the interface between air and cornea surface is introduced.

9.2.1 Zero-Error Case. Model0

A perfect model where there is no error is analyzed. The characteristics of the model are the following:

- The cornea is a plane.
- Pupil and cornea are coplanar.
- Affine projection is used. The image of the pupil center is the center of the image of the pupil.

- The LEDs are projected onto the cornea plane using the cornea center as projection center, instead of reflected on the cornea surface.
- Optical and visual axes are coincident.
- There is no refraction.

A schematic drawing of the setup is shown in Figure 9.2. The camera is located below the monitor.

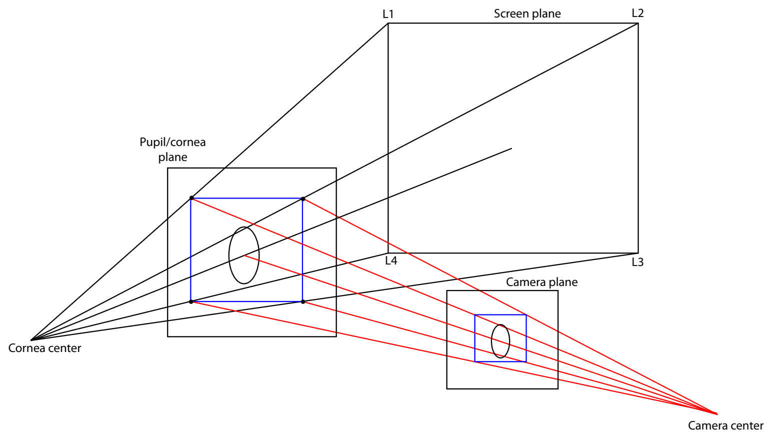


Figure 9.2: Model0.

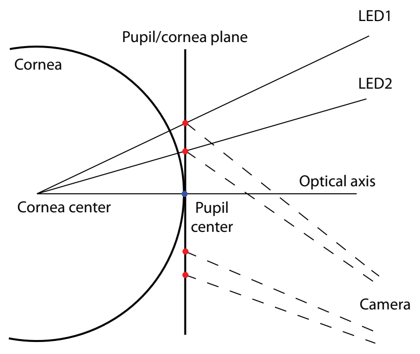
As expected, the error in the estimated gaze coordinates is zero for all three models.

9.2.2 Effect of Perspective Projection. Model1

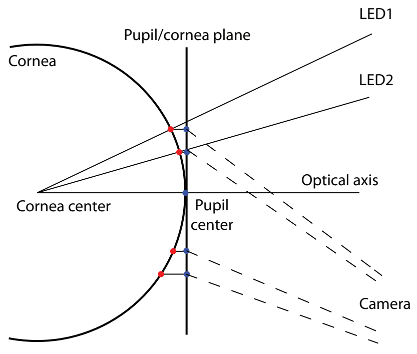
In affine projection, the image of the pupil center is mapped to the center of the image of the pupil. Under perspective projection it cannot be assumed that the projection of the center is the center of the projection. Therefore, using the center of the image of the pupil to estimate gaze under perspective projection will introduce an error.

Two different methods to calculate the 3D location of the corneal reflections have been considered. The first one assumes that the cornea is a plane. The light sources are projected onto the cornea plane using the cornea center as projection center (same case as in Model0). This is denoted Model1a, and is shown in Figure 9.3(a). Since cornea plane and pupil plane are perpendicular to the optical axis, both planes are coplanar.

A more realistic way to simulate a planar cornea is to project the light sources onto the cornea sphere and then project these corneal reflections onto the pupil/cornea plane, which is located perpendicular to the optical axis and tangent to the cornea surface, as shown in Figure 9.3(b). This method is denoted Model1b.



(a) Model1a



(b) Model1b

Figure 9.3: Model1. (a) Model1a. The light sources are projected onto the pupil/cornea plane. (b) Model1b. The light sources are projected onto the spherical surface of the cornea, and then projected onto the pupil/cornea plane.

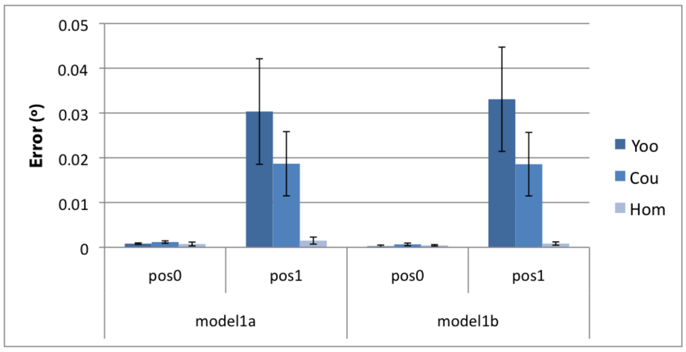


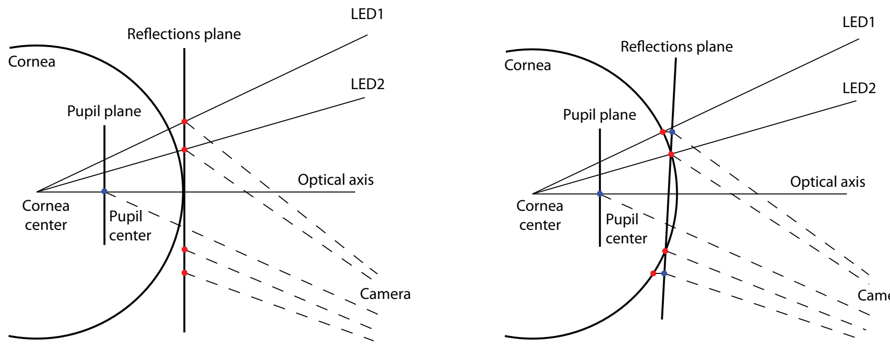
Figure 9.4: Average error in degrees when perspective projection is introduced (Model1).

The use of perspective projection instead of affine projection introduces an error in the estimated gaze coordinates. However, this error is very low, below 0.04° of visual angle for the three methods. This is in accordance with the results presented by Kang et al. in [38], where it is shown that the difference in the estimated PoR coordinates between using the image of the pupil center and the center of the pupil image is up to two orders of magnitude lower than the estimation bias due to the method. Therefore, it is possible to assume that the projections of light sources onto the cornea surface and the eye onto the image plane occur in affine space and use the center of the pupil image to estimate the PoR without increasing the estimation error.

9.2.3 Effect of Non-Coplanarity of Pupil Plane and Reflections Plane. Model2

The three gaze estimation techniques assume that pupil and the corneal reflections are located on the same 3D plane. The cross-ratios methods calculate the cross-ratio using the corneal reflections and the center of the pupil. In the case that the pupil and corneal reflections are not coplanar in 3D space, there will be an error in the estimated PoR. According to the analysis of the cross-ratio method carried out by Kang et al. in [38], the non-coplanarity of pupil plane and corneal reflections plane is one of the main sources of error in the aforementioned method.

To analyze the influence of the non-coplanarity between pupil plane and cornea plane, the pupil plane is placed in its natural location. As in the previous model, there are two ways to calculate the 3D location of the corneal reflections. In Model2a the light sources are projected onto a plane perpendicular to the optical axis (and therefore parallel to the pupil plane), as shown in Figure 9.5(a). In Model2b (Figure 9.5(b)) a more realistic approach is followed: the light sources are projected onto the cornea sphere, and the reflections plane is created using three corneal reflections; the fourth reflection is mapped to the reflections plane to simulate a planar cornea. Note that in Model2a the reflections plane is forced to be perpendicular to the optical axis, something unlikely to happen in a real scenario. In Model2b the reflections plane can take an arbitrary orientation.



(a) Model2a (b) Model2b

Figure 9.5: Model2. (a) Model2a. The light sources are projected onto the cornea plane. (b) Model2b. The light sources are projected onto the corneal surface; three of the corneal reflections are used to calculate a plane, and the fourth corneal reflection is projected onto that plane.

The methods based on the cross-ratio are affected by the way the reflections plane is computed. In Model2a the error obtained is similar to the values obtained in Model1 when pupil and cornea are coplanar, i.e. the estimation error is very low (below $0.04°$). However, in Model2b the error increases significantly to $0.37°$ in *pos0* and

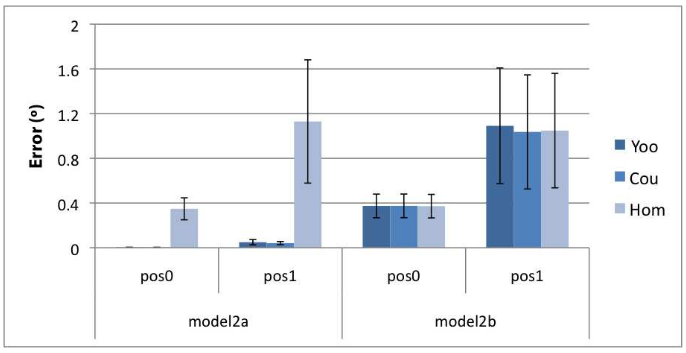


Figure 9.6: Average error when pupil plane and cornea plane are not coplanar (Model2).

around 1° in *pos1*. The difference between both models lies on the way the reflections plane is calculated. In Model2a, the reflections plane is forced to be perpendicular to the optical axis. Therefore, the center of the pupil is always located on the same position with respect to that plane. The cross-ratio formed by the virtual corneal reflections and the pupil center on the image is approximately the same as the cross-ratio formed by the PoR on the screen and the light sources. The cross-ratio methods thus provide an accurate estimation of the PoR (the error is due to the assumption of affine projection).

On the contrary, in Model2b the reflections plane can take any orientation, and is therefore not perpendicular to the optical axis. As the angle between optical axis and reflections plane increases, the estimation error increases. This behavior is illustrated in Figure 9.7. The reflections plane is rotated on the x axis from -20 to 20°, and the estimation error is calculated for a single point on the screen. The error is approximately 0 when reflections plane and optical axis are perpendicular, and increases as the angle deviates from 90°.

Kang et al. point out in [38] that the distance from pupil plane to reflections plane is one of the main sources of error (together with the angular offset between optical and visual axes). In their study they assume that the reflections plane is perpendicular to the screen. According to the analysis carried out in this section, the estimation bias occurs when two conditions are met: (1) the pupil plane and reflections plane are not coplanar, and (2) the reflections plane is not perpendicular to the optical axis. This results provides an insight into the geometrical behavior of the cross-ratio method. Nevertheless, in a real situation the reflections plane is not perpendicular to the optical axis, and therefore an estimation error is introduced.

9.2.4 Effect of Reflection. Model3

In the models presented so far the locations of the corneal reflections in 3D space are calculated by means of a projection with the center of the cornea being the center of

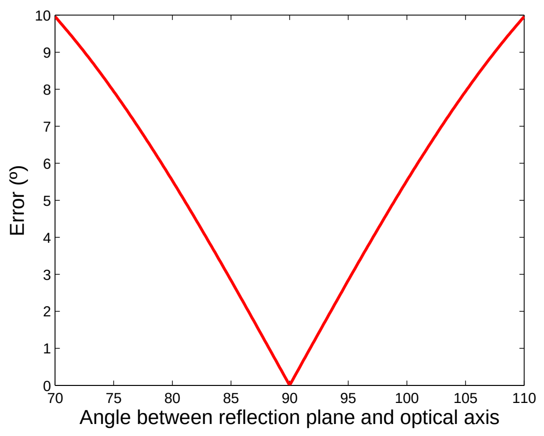


Figure 9.7: Increase in estimation bias for the Yoo method as the angle between reflections plane and optical axis increases.

projection. In Model3 the corneal reflections are calculated by means of a reflection.

As in the previous models, two possible ways to calculate the location of the corneal reflections in 3D space have been considered. In Model3a the light sources are reflected on a cornea plane perpendicular to the optical axis, while in Model3b they are reflected onto the cornea sphere. To maintain the coplanarity of the corneal reflections in Model3b, a reflection plane is calculated with three reflections, and the fourth reflection is projected onto that plane.

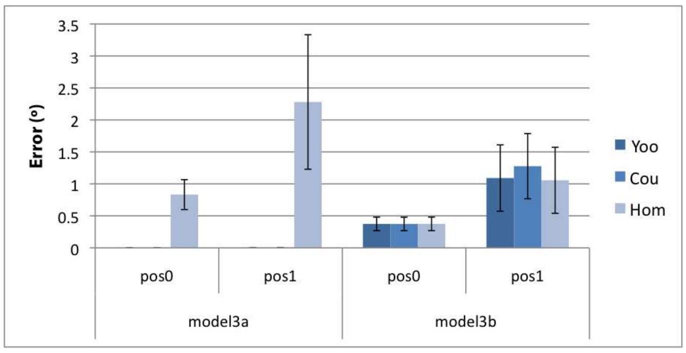


Figure 9.8: Average error in degrees when the corneal reflections are calculated by means of reflection (Model3).

The cross-ratio methods experiment a similar behavior as with Model2, with a low error in Model3a and a higher error in Model3b. As in Model2, in Model3a the reflections plane is forced to be perpendicular to the optical axis, while in Model3b it is not, and therefore the error increases. In *pos1* and using Model3b, the estimation error for the Yoo method and the homography method does not increase, while in the case of the Coutinho method there is an increase of 0.24°.

9.2.5 Effect of Curvature of the Cornea. Model4

The models studied so far assume a planar cornea, and the reflections plane is calculated in two different ways. In this section a spherical cornea is considered. Figure 9.9 shows the errors obtained for each method in each configuration.

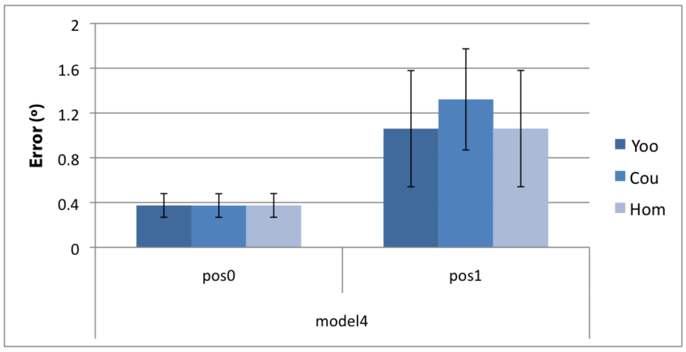


Figure 9.9: Average error in degrees when the cornea is spherical (Model4).

The errors obtained for Model4 are similar to the ones obtained for Model3b. In *pos1*, the increase of estimation error due to the curvature of the cornea is below 0.04° for the three methods. Although the cornea is now a sphere, it is possible to consider the effect of its curvature to be negligible. At a normal working distance of a remote gaze tracker (50 to 60 cm from the screen), the size of the cornea is small with respect to the screen, and the curvature can be neglected.

9.2.6 Effect of Angular Offset between Optical and Visual Axes. Model5

As described in Chapter 2, the optical axis lies in the direction of the line that joins the center of the cornea and the center of the pupil. When we look at one point, it is the visual axis that actually points to the observed point, and not the optical axis. In this section the influence of the angular offset between both axes is analyzed. A horizontal offset of 5° and a vertical offset of 1.5° have been considered (values suggested by Guestrin and Eizenman in [21]). The results are shown in Figure 9.10.

Kang et al. analyze in [38] the original method proposed by Yoo et al. to estimate gaze using the cross ratio [93], and conclude that the angular offset between optical and visual axes introduces an error on the estimated gaze point on the screen. Yoo

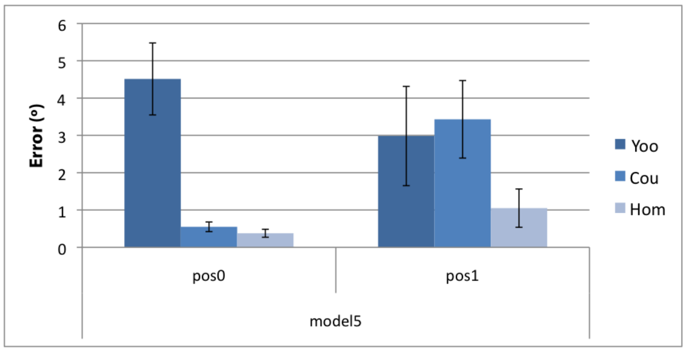


Figure 9.10: Average error in degrees when an angular offset between optical and visual axes is introduced (Model5).

and Chung propose in [92] a correction of the location of the corneal reflections using four α parameters as described in Chapter 7. The four α parameters are calculated during calibration, and their objective is to correct for the misalignment between the corneal reflection and the pupil center, thereby compensating for the non-coplanarity between pupil and reflection plane. Since the actual visual axis is not employed, it does not compensate for the angular offset between axes. As a result, the estimation error in *pos1* increases significantly from 1.06° to 2.98° of visual angle.

Some degree of compensation is achieved by the method proposed by Coutinho and Morimoto in [10], in which an on-screen offset vector that is calculated through a calibration process is applied to compensate for the angular offset. However, the error increases in *pos1*, with an estimation bias of 3.43°, which represents an increase of 2.11° from the case where there is no offset between axes. The homography method models the offset between axes implicitly, and therefore the accuracy is not affected when the visual axis is introduced. The accuracy in *pos1* is 1.05°.

Error as a function of horizontal axes offset

In order to study the effect of the angular offset in each of the methods, an analysis has been conducted where the horizontal angular offset between optical and visual axes increases from 0 to 5° (the vertical offset is 0°). Figure 9.11 shows the obtained average error.

As expected, the homography method has a flat response due to the implicit modeling of the angular offset. The Yoo method is greatly affected by the offset, and presents a linear response. The Coutinho method partially compensates for the offset by applying an on-screen correction. However, this correction does not completely model the error. Furthermore, since the vector is applied on on-screen coordinates, it is affected by head movements in depth.

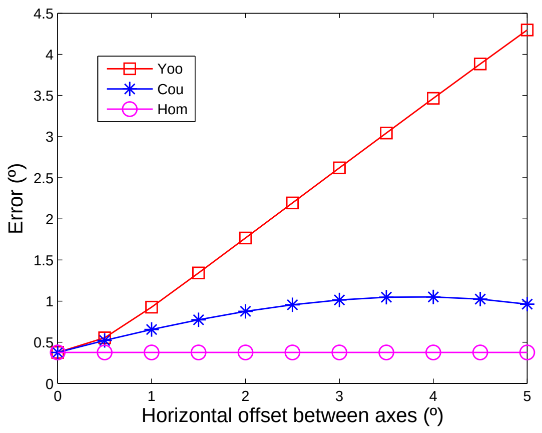


Figure 9.11: Average error in degrees for the three gaze estimation methods as the horizontal offset between optical and visual axes increases for a symmetric configuration *pos0*.

9.2.7 Effect of Refraction. Model6

When the light enters the eye, it is refracted on 4 different refracting surfaces due to the different indices of refraction. Most of the refraction takes place at the interface between air and cornea, which is considered to have an index of refraction of 1.376. Villanueva and Cabeza pointed out that due to refraction the image of the pupil center is not coincident with the center of the pupil image [83]. In this section the effect of refraction in the estimation error is analyzed.

Figure 9.12 shows the estimation bias for each method. It is assumed that there is no angular offset between optical and visual axes.

The accuracy obtained with the three methods is similar to the accuracy achieved when no refraction is considered. The effect of refraction is therefore negligible and needs not be modeled. This result is in accordance with the findings by Kang et al. in [38].

Figure 9.13 shows the accumulated errors introduced by each assumption for the three methods under study in the realistic hardware configuration *pos1*. The cross-ratios methods are mostly affected by two assumptions: (1) the non-coplanarity of pupil and reflection plane, and (2) the angular offset between optical and visual axes. This result is in accordance with the findings by Kang et al. [38]. In the case of the homography method, the non-coplanarity of pupil and reflection plane accounts for 98% of the total estimation error.

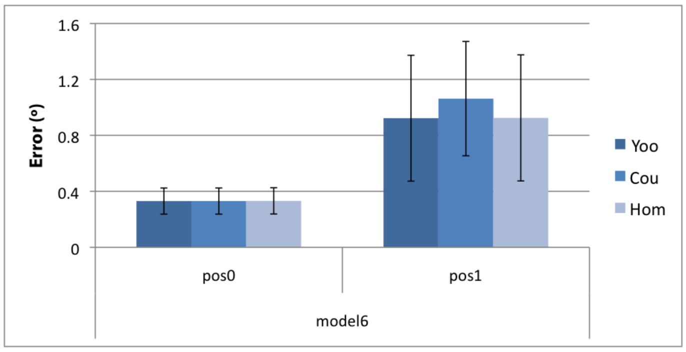


Figure 9.12: Average error in degrees when refraction on the cornea surface is introduced (Model6).

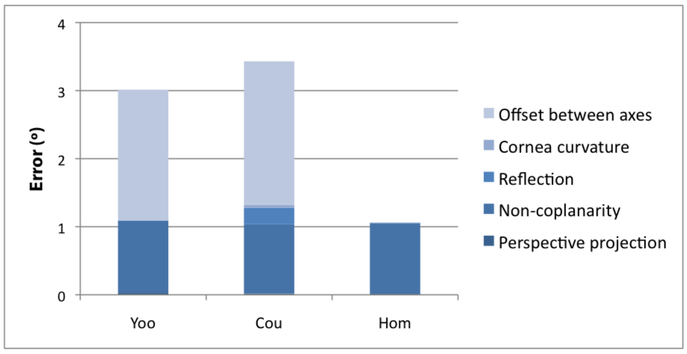


Figure 9.13: Error introduced by each assumption for the three methods under study (in *pos1*).

9.3 Effect of Number of Calibration Targets

Increasing the number of calibration targets usually results in a reduction of the estimation bias, as information is gathered on more points on the screen that can be then used to provide a more accurate estimation of where the person is looking. In this section the effect of such increase is analyzed. Two new models are introduced in the analysis: the combination of homography and interpolation that was described in Section 8.2.2 and a pure interpolation technique.

The combination of homography and interpolation allows for partial compensation of the non-linearities that take place as the eye rotates in its orbit. A pair of third-order polynomials is used to correct for inaccuracies in the estimated gaze coordinates after the homography method is applied. The pure interpolation technique is described in Section 7.2, and represents the most common gaze estimation method when no information about the hardware setup is available.

The Yoo method uses only four calibration points (the four light sources), and does not have any way of adding more. The Coutinho and homography methods can use as few as four points, but using more points to better estimate the calibration parameters is possible. The homography and interpolation and the pure interpolation methods require a minimum number of points in order to properly calculate the coefficients of the polynomials (six for second order, ten for third order).

An eye model in which all assumptions are taken into account is employed. The realistic hardware setup (*pos1*) is used. Figure 9.14 shows the average error for each method as the number of calibration points is increased. Only regular patterns have been considered, from 2×2 to 6×6 grids.

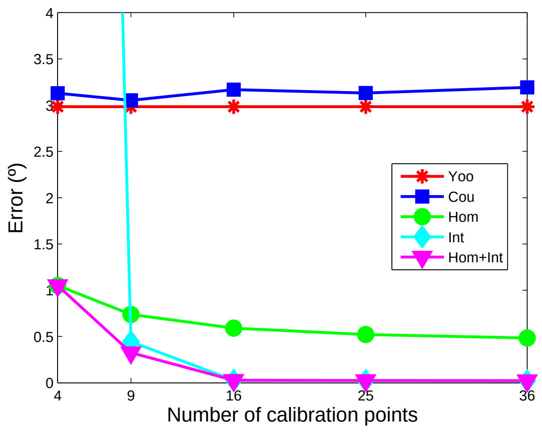


Figure 9.14: Average error for each gaze estimation method as the number of calibration points increases.

The error is constant for the Yoo method, since it is always calibrated using four points. The Coutinho method does not show any improvements when more calibration points are used, obtaining a similar estimation error as the Yoo method. In the case of the homography-based method the estimation error is approximately halved when 36 calibration targets are employed, and the accuracy improves from around 1° to 0.5°.

The coefficients of a pure interpolation technique cannot be properly determined when the system of linear equations is underdetermined. A second-order polynomial has 6 coefficients, while a third-order one has 10 coefficients. For 2×2 and 3×3 grids a second-order polynomial is used in the interpolation technique, while for 4×4 or more a third-order polynomial is used. The estimation bias is high for the 2×2 calibration pattern (above 30°) due to the underdetermination of the system. As the number of calibration points increases, the error decreases and reaches 0.03° of visual angle when 16 or more targets are used. The combination of homography and interpolation technique has the advantage that the error is constrained by the bounds of the homography method. Therefore, an accuracy of around 1° is obtained when only 4 targets are used and the coefficients of the third-order polynomials cannot be determined. As with the pure interpolation method, the error reaches 0.03° for 16 targets or more. A combination of homography and interpolation has the advantage of providing a low estimation bias when only 4 calibration points are used, and a very low error when the number of calibration points increases.

9.4 Head-Pose Tolerance

Eye movements with respect to the screen might have a notable impact on the estimation bias in uncalibrated setups. Contrary to fully calibrated setups, the location of the eye in 3D space is not known, and the parameters calculated during user-calibration (e.g. coefficients of a polynomial regression) are usually only valid for calibration position. When the user moves the head, the accuracy of the estimated gaze coordinates decreases.

An ideal gaze estimation method should be head-pose invariant in order to be robust against naturally occurring head movements. Prior knowledge of the geometry of the system and the camera parameters can be used to impose constraints that ensure head-pose invariance. However, in non hardware-calibrated setups ensuring head-pose invariance is a challenging research issue.

To test the theoretical head-pose invariance of the methods, two different tests have been carried out: (1) the head moves on a plane parallel to the screen covering an area of ±300 mm, and (2) the head moves in depth along the z axis. The following configuration has been used:

- All model assumptions are taken into account.
- The realistic hardware configuration (*pos1*) is used.
- The Yoo, Coutinho and homography methods are calibrated using 4 points.
- The interpolation methods are calibrated using 16 points arranged on a 4×4 grid.

9.4.1 Effect of Head Moving on a Plane Parallel to the Screen

To simulate the head moving on a plane parallel to the screen, the position of the eye is moved on a 5×5 grid that covers an area of 600×600 mm, with the calibration position being in the center of the grid. Such big movements are unlikely to occur in a real scenario, since the eye would fall outside the field of view of a common camera[3].

Figure 9.15 shows the error obtained for the five methods in each of the 25 locations, divided in 5 sections where each section represents one row, from top to bottom. Figure 9.16 shows the average error and standard deviation over all the locations.

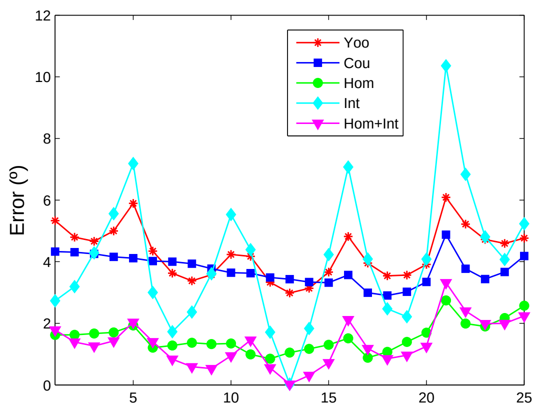


Figure 9.15: Average error in degrees for the five gaze estimation methods when the eye moves on a plane parallel to the screen. The x axis shows the 25 locations of the eye (arranged on a 5×5 grid), with the first 5 points show the first row.

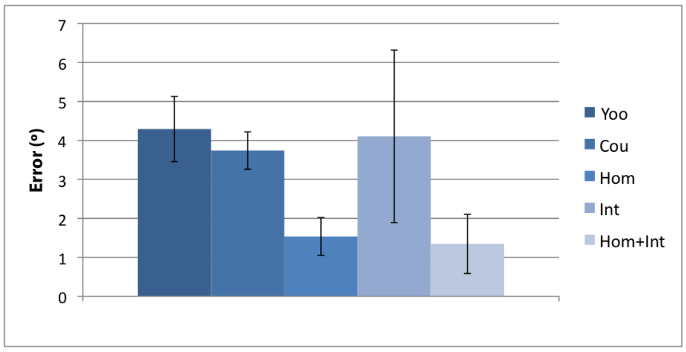


Figure 9.16: Average error in degrees for the five gaze estimation methods when the eye moves on a plane parallel to the screen.

The average error for the Yoo method is 4.29° with a standard deviation of 0.84. The Coutinho method is less affected by head movements, and the average error is 3.74° with a standard deviation of 0.47. In the case of the homography method, the average error is 1.53° with a standard deviation of 0.49, thus being quite robust to lateral and vertical head movements.

The accuracy of the pure interpolation method is affected by head movements. The average error is 4.10° with a standard deviation of 2.21. It is the method that presents most variation in the error. As expected, the error is almost 0° for calibration

[3]Pan-and-tilt cameras can be used to track and follow the user's eyes.

position (point 13 in Figure 9.15) but increases significantly as the eye moves. The homography and interpolation method is less affected by the movements, with an average error of 1.34° with a standard deviation of 0.76.

9.4.2 Effect of Head Moving in Depth

Movements of the eye in depth with respect to the camera might have a notable impact on the estimated gaze coordinates on the screen in uncalibrated setups, as regression-based gaze estimation methods cannot usually compensate for scale changes.

In this test, the eye is moved along the z axis, ±300 mm from calibration position, thus covering a total of 600 mm. Figure 9.17 shows the average error in degrees for each estimation method as the eye moves in depth.

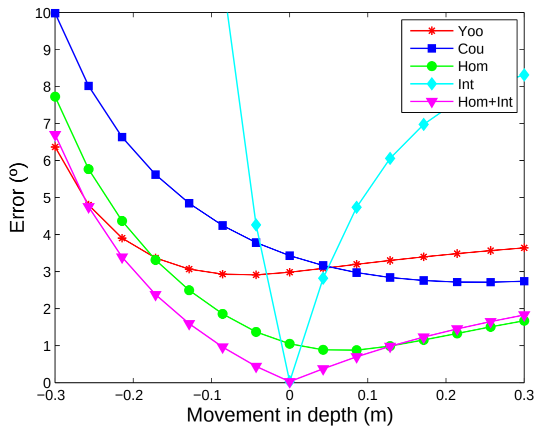


Figure 9.17: Average error in degrees for the five gaze estimation methods when the eye moves along the z axis.

The Yoo method is quite robust against head movements, and has an almost planar response in a ±10 cm range from calibration position. Although also based on the cross-ratio invariant, the error in the Coutinho method is more affected by head movements, since the correction for angular offset is performed directly on screen coordinates. The homography method has a higher accuracy than the Yoo and Coutinho method. The error increases as the eye moves from calibration position, but it is below 2° for head movements of up to 10 cm. In the case of the pure interpolation method, the error increases significantly even for small head movements (4° when the eye has moved 5 cm from calibration position). The homography and interpolation combination has a high accuracy while not being as affected by head movements as the pure interpolation. The error is below 1° for relative movements of up to 10 cm from calibration position.

9.5 Real Scenario

In a real situation, the effects of model assumptions, naturally occurring head movements and noise in the detection of the eye features in the image will add up and affect the accuracy of the estimated gaze coordinates, increasing the error. The test carried out in this section analyzes the performance of all the gaze estimation models in a realistic scenario under the following assumptions:

- All model assumptions are taken into account.
- The realistic hardware setup *pos1* is employed.
- Natural head movements are simulated by introducing Gaussian noise on the eye coordinates in the three axes of space. The standard deviation is 30 mm.
- The camera has a Gaussian measurement noise with a standard deviation of 0.5 pixels.
- The Yoo, Coutinho and homography methods are calibrated using 4 points, while the interpolation methods are calibrated using 16 points.

The measurement noise and the noise in the eye location is introduced for both calibration and test. Figure 9.18 shows the estimation bias for the five methods under study, in both a head fixed and realistic condition.

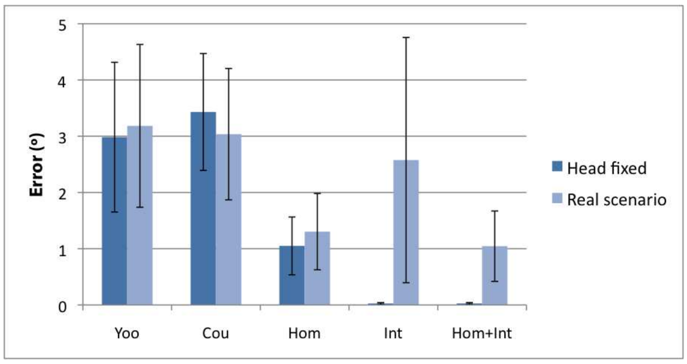


Figure 9.18: Average error in degrees for the five gaze estimation methods in a head fixed and a realistic scenario with all the assumptions taken into account and noise introduced in eye position and measurements in camera.

The average error for the Yoo method is 3.18°, which represents an increase of 6.7% with respect to the head fixed situation, while for the Coutinho method the error is 3.04°, an improvement of 11.5%. The slight differences indicate that most of the error is due to the angular offset between axes, as the small head movements do not affect the error significantly. In the case of the homography method the error increases from 1.05° in the head fixed condition to 1.30° in the realistic condition.

As expected, the pure interpolation method is affected by head movements and the estimation error increases from 0.03° to 2.57° (three orders of magnitude). This is in accordance with empirical experience with systems that use interpolation methods to estimate gaze: the accuracy is high right after calibration, but the error increases as the user works with the system and moves from calibration position. After some time, a recalibration is usually required. The combination of homography and interpolation is also affected by head movements, and the error increases from 0.03° to 1.04°. However, the accuracy is the highest among the methods under study, indicating that the homography and interpolation is not affected by head movements as much as a pure interpolation technique.

The results presented above provide an estimate of the theoretical estimation error for each of the methods in a realistic scenario where the user is allowed to make slight head movements. In the next chapter these results are compared with the error obtained when real data is employed.

9.6 Conclusions

The investigation of the estimation bias using simulated data presented in this chapter accomplishes two main objectives: (1) it provides an insight into the main sources of error of the homography method described in Chapter 8 and of two geometry-based, uncalibrated methods based on the cross-ratio invariant in projective space ([92], [10]); and (2) it shows the advantages that the homography method offers over methods based on the cross-ratio and interpolation regarding accuracy and tolerance to head movements.

Section 9.2 presents a thorough analysis of the error due to the assumptions made by the homography method and by the two other geometry-based methods under study, the Yoo method [92] and the Coutinho method [10], both based on the cross-ratio. The main characteristic shared by the three methods is that no geometry or camera calibration are required. The results obtained using simulated data show that the main source of error of the homography method is due to the non-coplanarity of corneal reflection plane and pupil center, while the two methods based on the cross-ratio are also affected by the angular offset between optical and visual axes. This is in accordance with the findings by Kang et al. [38]. However, the results shown in Section 9.2 indicate that the calibration method proposed by Coutinho and Morimoto in [10] does not fully compensate for the angular offset between axes, as opposed to the findings presented in [38]. When all assumptions are taken into account and four points are used for calibration in the three methods, the estimation error of the homography method is 1°, while the error of the methods based on the cross-ratio is around 3°.

Coutinho and Morimoto suggest that increasing the number of calibration points reduces the estimation bias of their method [10]. However, the analysis carried out in Section 9.3 shows that the accuracy remains constant when the number of calibration points increases, suggesting that calibrating with more than four points is unnecessary. In the case of the homography method, the error is halved when the number of calibration points is increased from 4 to 25 or more. The investigation carried out in Section 9.3 includes the analysis of the estimation error of an interpolation method and the non-linear extension of the homography method described in Section 8.2.2, and indicates that the estimation error of the polynomial regression can be as low as 0.03° provided that the number of calibration points is enough to give a proper estimation of the regression coefficients (i.e. the system should not be underdetermined); the homography and interpolation method, however, is theoretically not affected by a low number of calibration points.

Interpolation methods provide a very high accuracy with a sufficiently high number of calibration points. However, the calculated coefficients of the polynomials are only valid for calibration position. In Section 9.4 it is shown that movements of the head from the initial position greatly affect the accuracy of the method, increasing the estimation bias. This is especially relevant in movements along the z axis. The homography and cross-ratios methods are less sensitive to head movements; they are not, however, head-pose invariant. Nevertheless, the analysis carried out in Section 9.5 indicates that slight head movements of around 30 mm from calibration position do not increase the error significantly.

The next chapter evaluates the estimation error of the gaze estimation methods under study employing real images recorded with five users. The results obtained are compared to the values shown in this chapter.

Chapter 10

Eye Model Assessment on Real Data

The previous chapter presented an analysis of the effect of the different assumptions made by the eye models in the accuracy of the gaze estimation methods under consideration. The study was carried out using simulated data. In this chapter the performance of the gaze estimation methods is analyzed and compared using real data.

10.1 Hardware Setup and Procedure

Five participants (four male, one female) participated in the evaluation of the gaze estimation methods. One of them wore contact lenses.

A 24" monitor with a resolution of 1920×1200 pixels was used in the tests. The participants sat approximately 600 mm away from the screen. The head was not fixed, and therefore users had the ability to perform slight head movements. This provides a more realistic usage scenario, thereby increasing the ecological validity of the results and making it possible to extrapolate the results to non-laboratory setups.

Figure 10.1 shows the equipment employed. Images are grabbed using a Point Grey Flea2[1] camera with a 35 mm lens and a resolution of 800×600 pixels. The camera is located below the monitor and slightly closer to the user. The camera is connected to a laptop computer using a FireWire 800 connection.

In order to create the necessary corneal reflections, 4 Sony IVL-150 infrared lamps are placed on the corners of the screen. The methods based on the cross-ratio require a fifth light source located on-axis with the camera. This is achieved by placing a ring of LEDs around the lens of the camera.

[1]http://www.ptgrey.com/product/flea2/

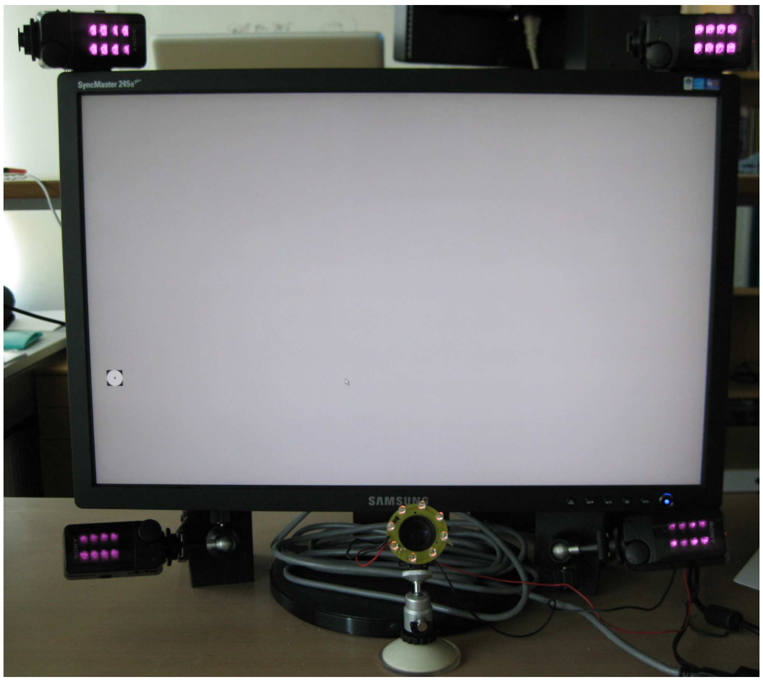


Figure 10.1: Experimental setup. Four light sources are placed on the corners of the screen and a ring of infrared LEDs is placed on-axis with the camera.

Participants were presented a sequence of targets on the screen, in a similar way as common calibration procedures found in commercial gaze tracking systems. Each user recorded 4 sequences, 2 of them for calibration purposes and another 2 for testing. The first sequence consists of 4 points, while the rest consist of 25 points arranged on a 5×5 grid. Approximately 10 images are used in each point to extract the eye features (the 5 corneal reflections and the pupil center).

- Sequence 1: calibration of the Yoo method. The user is asked to look at the four light sources.
- Sequence 2: calibration of the rest of the methods (25 points). The user is asked to keep the head as still as possible.
- Sequence 3: head still test (25 points). The user is asked to keep the head as still as possible (however, small movements might still occur as the head is not fixed).
- Sequence 4: head movement test (25 points). The user is asked to stand up and sit down again. The camera is adjusted to the new position of the user. The user is also asked to move the head in a natural way while looking at the points.

10.2 Eye Feature Extraction

An off-line eye tracker has been developed in Matlab and used to analyze each sequence in order to extract the different eye features from the images. The algorithm finds the configuration of glints corresponding to the reflections produced by 5 light sources. The locations of the corneal reflections are then used to find the pupil and fit an ellipse to its contour.

The corneal reflections are found by thresholding the image and exploiting certain geometrical constraints. Blobs that are too small or too big are eliminated. Since 4 light sources are placed on the corners of the screen, the 4 corneal reflections produced form a quadrangle with a rectangular shape. This type of shape is sought among all possible combinations of 4 glints in the image. The last corneal reflection is found by proximity to the other 4. Figure 10.2 shows the original image and the thresholded image with the candidate glints in red and the selected glints in blue.

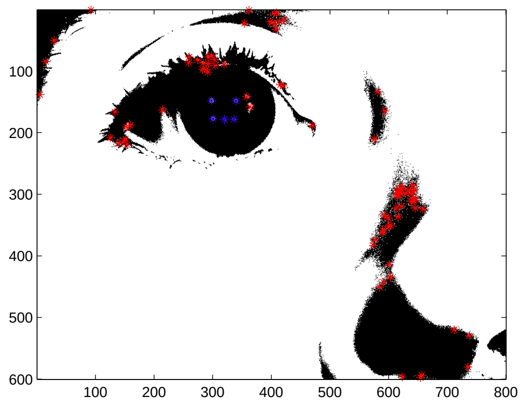


Figure 10.2: Thresholded image. The candidate glints are shown in red, and the selected configuration of five glints is shown in blue.

If the glints are properly detected, the pupil is sought in their vicinity. The image is thresholded and the blob that meets a criteria of minimum and maximum size is chosen as the pupil. A set of points along the contour is calculated by finding the maximum gradient along lines sent from the center of the blob in all directions, a technique based on active shape models [9]. Since the glints affect the contour of the pupil, points close to the glints are removed from the contour. The remaining contour points are then fitted to an ellipse using a RANSAC algorithm [30]. Figure 10.3 shows the contour points and the fitted ellipse.

Since the objective is to test the gaze estimation methods, the extraction of the eye features was supervised to ensure that all data obtained are correct. Data from

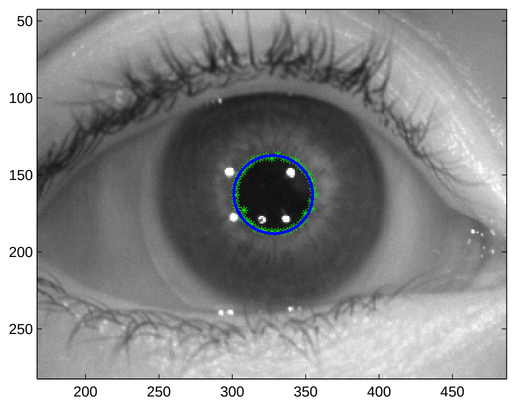


Figure 10.3: Pupil detection. A set of points on the contour is fitted to an ellipse using a RANSAC algorithm.

images where the eye features were not detected properly, e.g. due to eye blinks, were removed from the analysis.

10.3 Estimation Bias with Head Still

The calibration sequences were used to calculate the parameters for each gaze estimation method. Depending on the method a variable number of calibration points was used:

- In the Yoo method 4 calibrations points are used (light sources).
- In the Coutinho and homography methods 4 points are used (corners of the 5×5 grid).
- In the interpolation methods 16 points are used. The points are not evenly distributed as they are taken from a 5×5 grid.

A test was conducted on 25 points using Sequence 2, recorded with the head still. Figure 10.4 shows the average accuracy and standard deviation obtained with each method, using both simulated and real data. The error for each user and gaze estimation method is shown in Table 10.1.

The average accuracy for all participants using the Yoo method is 5.02° with a standard deviation of 1.49. This value is around double than the 2.27° obtained by Coutinho and Morimoto in [10] when the head is fixed. A possible reason for the increase is the smaller screen they used, 17” vs 24”. However, the value obtained with real data is also higher than the value of 2.98° obtained using simulated data.

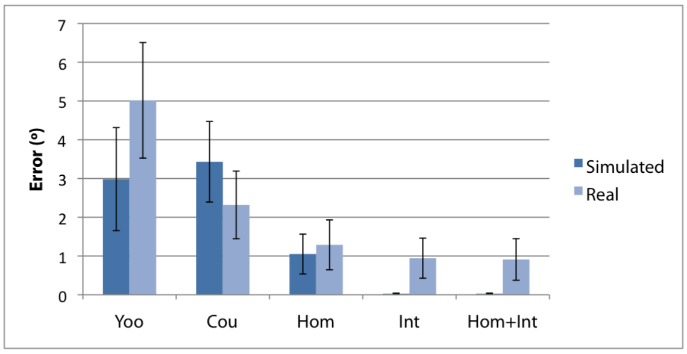


Figure 10.4: Average error in degrees for the five gaze estimation methods when users kept their head still for real and simulated data.

	Yoo	**Cou**	**Hom**	**Int**	**Hom + Int**
User 1	7.56	5.30	2.05	1.27	1.31
User 2	4.72	2.10	1.08	0.71	0.62
User 3	3.85	1.44	1.33	0.95	1.06
User 4	4.28	0.93	0.91	0.73	0.72
User 5	4.69	1.82	1.06	1.06	0.83

Table 10.1: Average error for each gaze estimation method when users kept their head still.

The average error for the Coutinho method is 2.32° with a standard deviation of 0.87, and is lower than the expected value of 3.43° that is obtained using simulated data. Coutinho and Morimoto report an average error of 0.91°. In this case, the increase in estimation error can be attributed to the different screen sizes and to the higher number of calibration points used in their study. However, the results presented in the next section indicate that increasing the number of calibration points does not improve the accuracy of the method.

The average error for the homography method is 1.29° with a standard deviation of 0.64 using 4 calibration points. This value is in accordance with the result obtained using simulated data in the previous chapter, 1.05°. The pure interpolation method has an error of 0.94° with a standard deviation of 0.52 when using 16 calibration points. The combination of homography and interpolation has the lowest error, 0.91° with a standard deviation of 0.54 using 16 calibration points (4 of which are used to calculate the homography). The results obtained with simulated data were as low as 0.03°. However, in this case noise in the estimation of the eye features and slight head movements increase the error.

Although the participants were asked to keep the head still, they still performed slight movements as they looked at the points presented on the screen. Therefore, the results presented above might be less accurate than what could be expected in a completely head-still situation. However, this was preferred over using a chin rest in order to increase the ecological validity of the analysis and be able to extrapolate the results to a real usage scenario.

10.4 Effect of Number of Calibration Targets

According to the results obtained in Section 9.3 using simulated data, an increase in the number of calibration points improves the accuracy of the homography and the interpolation methods. However, contrary to the results reported by Coutinho and Morimoto in [10], the accuracy of the Coutinho method is not improved as the number of calibration points increases. In this section a similar analysis is carried out using real data. The methods are calibrated using Sequences 1 and 2, and tested on Sequence 3 (i.e., head still).

Figure 10.5 shows the average error for the five gaze estimation methods as the number of calibration points increases from a 2×2 grid to a 5×5 grid.

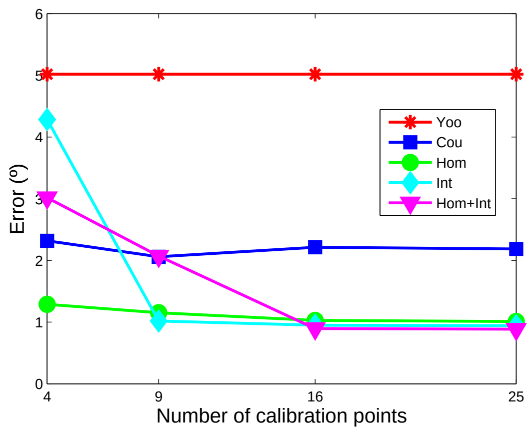


Figure 10.5: Average error for each gaze estimation method as the number of calibration points increases (using real data).

The number of calibration points for the Yoo method is constant, and so are its results. The accuracy obtained with the Coutinho method does not improve as the number of calibration points increases. This is in accordance with the results presented in Section 9.3, but disagrees with the results obtained by Coutinho and Morimoto in [10]. The estimation error for the homography method decreases a 20% from 1.29° to 1.01°.

In the case of the methods that use interpolation, a minimum number of points is required to properly estimate the regression coefficients. The pure interpolation method uses a second-order polynomial for 2×2 and 3×3 grids, and a third-order one for 4×4 and 5×5. As expected, the error is high for the 2×2 grid, 4.28°, and decreases to approximately 0.95° for 16 calibration points or more. In the case of the homography and interpolation method, a third-order polynomial is used regardless of the number of calibration points. Therefore, the error decreases from 3.01° when 4 calibration points are used, to 0.89° for 16 calibration points or more.

10.5 Effect of Head Movements

The performance of the different gaze estimation methods under slight head movements was investigated using simulated data in Section 9.5. In this section a similar analysis is carried out with real data using Sequences 1 and 2 for calibration, and Sequence 4 for testing. The users were asked to stand up and sit down before recording Sequence 4, and therefore the position and focus of the camera needed to be adjusted. Furthermore, participants were asked to perform natural head movements as they looked at the points displayed on the screen.

Figure 10.6 shows the average error in degrees for each method. The results obtained using empirical data are compared to the results using simulated data presented in Section 9.5, where Gaussian noise with a standard deviation of 30 mm is introduced in the eye coordinates. The error for each user is displayed in Table 10.2.

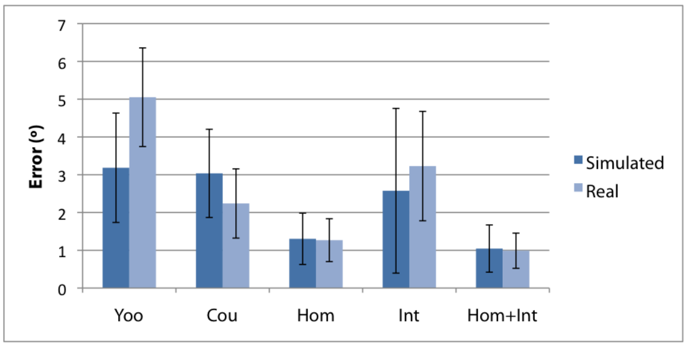


Figure 10.6: Average error in degrees for the five gaze estimation methods under head movements. Results for simulated data from Section 9.5.

The Yoo method has a higher average estimation error using real data, 5.05°, than the error obtained using simulated data, 3.18°. Furthermore, this value is double than the error obtained in the study conducted by Coutinho and Morimoto of 2.40° in the test trials where the users were asked to move from calibration position [10]. In the case of the Coutinho method, the accuracy improves using real data over the results

	Yoo	**Cou**	**Hom**	**Int**	**Hom + Int**
User 1	7.34	4.65	1.33	2.12	1.06
User 2	5.18	2.14	1.09	2.68	0.63
User 3	4.04	1.85	1.89	3.48	1.58
User 4	4.37	1.03	1.13	6.02	0.99
User 5	4.33	1.52	0.89	1.84	0.68

Table 10.2: Average error for each gaze estimation method when the users moved their heads.

obtained in simulation, 2.24° vs 3.04°. However, the error is higher than the error reported by Coutinho and Morimoto of 0.95°. As in the case of head fixed, the differences in these values can be attributed to the different screen sizes used.

The homography method has an average estimation error of 1.27°, a similar result as the value obtained using simulated data, 1.30°. Furthermore, the error in the head fixed case is 1.29°, which indicates that small head movements have a small impact on the accuracy of the method.

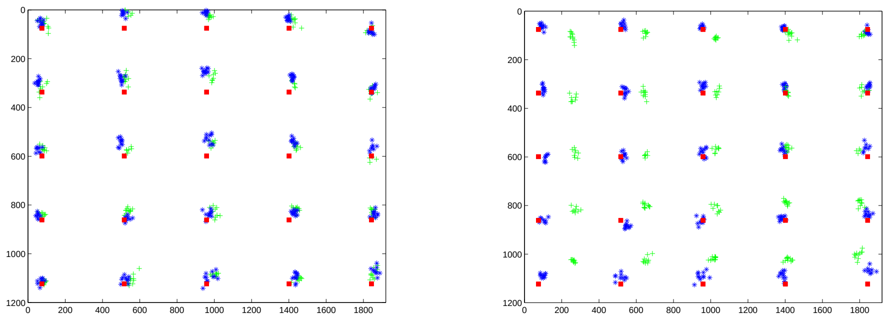
The study carried out in Section 9.5 using simulated data suggests that the interpolation method is greatly affected by slight head movements, as the estimation error increases from 0.03° to 2.57°. The error obtained using real data is 3.22°, confirming that a pure interpolation method is not head-pose invariant.

The homography and interpolation method, however, has an error of 0.99°. The error in the head fixed condition is 0.91°. The slight increase in the estimation bias between head-movement and head-fixed conditions indicate that small, natural head movements do not affect the accuracy of the method notably despite using an interpolation technique.

To illustrate the effect of head movements on the homography and interpolation methods, Figure 10.7 shows the estimated gaze coordinates of user 2 for both methods. The blue stars show the estimated gaze coordinates for the sequence with the head still, while the green crosses show the gaze coordinates in the sequence where the head moves. In the interpolation method the head movements introduce an offset in the estimated gaze coordinates, while in the homography method the estimated PoRs are more concentrated around the observed target coordinates.

10.6 Conclusions

The evaluation of the estimation error performed in this chapter using empirical data confirms the results obtained in the previous chapter where a simulator was employed. In the head-fixed condition, the homography method has an estimation error of 1.29° using four calibration points. Increasing the number of calibration points decreases the error to 1.01°. Depending on the application, this improvement might not be



(a) Homography (b) Interpolation

Figure 10.7: Estimated gaze coordinates for (a) homography (4 calibration points) and (b) interpolation method (16 calibration points). Blue stars show the estimated gaze coordinates for head still, green crosses for head movement. Red squares show the coordinates of the observed points.

worth it, since it requires the user to spend a significantly longer time calibrating.

The methods based on the cross-ratio have a higher estimation error when compared to previous results by Yoo and Chung [92] and Coutinho and Morimoto [10]. The differences might be due to the different screen size used. Contrary to the result by Coutinho and Morimoto [10], increasing the number of calibration points does not decrease the estimation error of the Coutinho method. This is in accordance with the results presented in the previous chapter using simulated data.

Although the methods that include an interpolation technique have a low estimation error in the head-still condition, they are affected by head movements. The pure interpolation method should therefore be avoided in setups where the user can move his or her head freely. The combination of homography and interpolation, however, is not affected by head movements, and has a higher accuracy than the pure homography method. The downside is that it requires more calibration points.

The tests in both simulated and real data have shown the main benefits of the homography method.

- Average accuracy of 1.28° of visual angle using 4 calibration points, and 1.03° using 16 points.
- Low number of calibration points required. Only four points are necessary to calculate the homography that maps the pupil to the point of regard. Compared to the methods based on the cross-ratio, the homography method offers a lower estimation error.
- Low effect of head movements on the accuracy. The average estimation error decreases from 1.29° to 1.27° when the user moves from calibration position and

performs natural head movements while looking at the on-screen targets.

- Flexibility regarding the location of the light sources and calibration points. The methods based on the cross-ratio require five light sources, four of them placed in the corners of the screen and one placed co-axial with the camera. The homography method simplifies the hardware configuration by requiring only four light sources, thereby increasing the flexibility of the setup. A limitation of the method, however, is the requirement of four light sources, whereas interpolation-based methods require only one.
- Geometry calibration of the hardware is not necessary. The camera and the light sources can be placed in any location, and their position with respect to screen needs not be measured.

Despite the good properties of the homography method to estimate gaze, noise in the extraction of eye features introduce jitter in the estimated gaze coordinates, and smoothing is usually required. Part IV explores ways to enhance gaze interaction that can benefit both low-cost and high-quality gaze tracking systems. Fixation and smooth pursuit detection are explored in Chapters 11 and 12 as a means to smooth the signal depending on the type of eye movement. In Chapter 13 a combination of gaze pointing and EMG clicking is investigated in order to explore the potential of multimodality to improve the performance of gaze interaction in target-acquisition tasks and avoid the Midas Touch problem [33].

Part IV

Enhancing Gaze Interaction

Chapter 11

Fixation Detection

Analysis of eye movements to study cognitive processes has gained popularity over the last years [12]. In psychological research on eye movements, the raw data provided by an eye tracking system are usually post-processed in order to cluster a set of gaze points into areas of interest and extract higher-level information, thus reducing the complexity of the eye-movement protocol [63]. This is generally achieved by applying an *a posteriori* algorithm on the gaze coordinates that identifies different eye movements, mainly fixations and saccades.

A post-processing stage that smooths the signal is commonly used in gaze-based interaction, with the difference that in this case the identification of the eye movements is done in real time. In most video-based gaze tracking systems, the estimated gaze coordinates suffer from jitter when the user is fixating a single point. This noise has two main sources: (1) inaccuracy in the extraction of the eye features (e.g. pupil center or corneal reflections), and (2) miniature eye movements that occur during a fixation, such as microsaccades and tremors (see Section 2.2). As a consequence of this noise, the pointer does not remain steady during a fixation, which might distract and confuse the user (if the cursor is displayed). Although it affects all systems, this jitter can be especially relevant in systems that make use of low-cost components, since noise tends to be higher than in systems built from high-end hardware.

Due to this inherent noise in the estimated gaze coordinates, gaze-based interaction can benefit from smoothing the cursor position by using the information of the last N images and not only the last one. This can help to reduce the jitter and make the cursor more stable. Such smoothing is usually performed by a fixation detection stage that can differentiate between fixations and saccades. When a fixation is detected, the cursor position is averaged over time, reducing the apparent noise perceived by the user. Once the fixation is finished and a saccade is detected, the algorithm stops smoothing the cursor position until the next fixation in order to avoid the “lag-behind” effect during a fast eye movement.

This chapter presents a review of fixation detection algorithms and their limitations, and describes an implementation of an algorithm that combines two different techniques to make the detection more robust. This algorithm is integrated in the eye movement detection algorithm presented in the next chapter.

11.1 Fixation Detection Algorithms

The literature regarding algorithms for real-time fixation identification is scarce. Salvucci and Goldberg [63] presented a taxonomy of fixation detection algorithms based on spatial and temporal characteristics. They classified the algorithms according to five different criteria: velocity, dispersion, area, whether the algorithm is duration sensitive, and whether it is locally adaptive. A summary of the techniques that can be implemented in real time is given here, together with some examples from the literature.

11.1.1 Velocity-Based Algorithms (I-VT)

Velocity-based algorithms use the eye velocity as the criteria to identify a fixation, based on the fact that the eye has a low velocity during a fixation, whereas during a saccade the velocity is very high. For each given pair of on-screen coordinates (x_i, y_i), the velocity is calculated with respect to the previous point (x_{i-1}, y_{i-1}). Given the distance between the points $d_i = \| (x_i, y_i), (x_{i-1}, y_{i-1}) \|$, the corresponding angle θ_i is calculated using Equation 11.1.

$$\theta_i = 2 \tan^{-1} \left(\frac{d/2}{D} \right) \tag{11.1}$$

where D is the distance from eye to screen, usually around 60 cm. The velocity is then calculated as $\dot{\theta}_i = \theta_i / \Delta t$, where Δt is the time elapsed between samples. The algorithm requires specifying one parameter, the maximum velocity allowed $\dot{\theta}_{max}$. This single threshold will classify a point (x_i, y_i) as belonging to a fixation when $\dot{\theta}_i < \dot{\theta}_{max}$; otherwise it will be classified as a saccade.

The value of $\dot{\theta}_{max}$ can be estimated from the characteristics of the eyeball kinematics (see Section 2). However, there is no agreement on the correct value. Salvucci and Goldberg [63] propose a threshold of 20°/s, whereas Duchowski et al. [15] employ a value of 130°/s. The characteristics of the gaze tracking system used play an important role in the calculated point-to-point velocities, and therefore the value of $\dot{\theta}_{max}$ can be optimized for a specific system by conducting an empirical study.

In their system, Duchowski et al. [15] use a velocity-based method to detect fixations for binocular gaze tracking a virtual reality environment, with the only difference of using a 3D vector for gaze. They note that point-to-point velocities can be greatly affected by noise in the signal, as more noise will result in higher velocities during a fixation. In their article they propose a fifth-order filter, but no significant differences with the second-order filter are found.

Koh et al. [39] use a Kalman filter to separate saccades from fixations by comparing the observed and the predicted eye velocities. A system-dependent threshold is used to make that classification. The authors report an accuracy increase of 10% of this method over the I-VT technique presented above.

11.1.2 Dispersion-Based Algorithms (I-DT)

Dispersion-based algorithms use the spatial information of the cursor location to identify a fixation, and are based on the fact that the dispersion of the gaze coordinates during a fixation is low. For a sequence of N points, the dispersion of the points in the window is calculated and, if lower than a pre-specified threshold, a fixation is reported.

Dispersion-based algorithms require specifying two parameters, a minimum duration time t_{min} and a maximum dispersion threshold D_{max}. As mentioned in Chapter 2, a fixation is considered to have a minimum duration of 100 to 200 ms. Once that time has elapsed, the maximum dispersion of the gaze coordinates in a temporal window is calculated and compared to the threshold. As long as the measured dispersion is lower than D_{max}, the algorithm reports a fixation; when it is higher, the fixation ceases. The maximum dispersion during a fixation is considered to be 1° of visual angle; however, as with velocity-based algorithms, the dispersion will be affected by the noise in the gaze tracking system, and a different threshold might be required.

11.1.3 Area-Based Algorithms (I-AOI)

Area-based algorithms identify fixations only when they occur within specified regions of interest, for example interactive elements such as buttons or icons. Gaze points occurring outside these target areas are labeled as saccades. A benefit of this method is that when a fixation is detected on a region of interest, the cursor can be placed on the center of the region, effectively eliminating all the jitter coming from the noisy gaze coordinates.

However, in order to implement this technique the fixation detection algorithm must know the location of the regions of interest for every user interface that is displayed, and this information might not be provided by the operating system.

11.2 Velocity- and Dispersion-Based Algorithm

A fixation detection algorithm has been implemented and incorporated into the ITU Gaze Tracker presented in Chapter 4. The algorithm is composed of two stages: (1) a detection stage based on a combination of a velocity and a dispersion-based technique; and (2) a smoothing stage that calculates the average gaze coordinates over the last N samples by means of a moving average. Both stages are described below.

11.2.1 Detection of Fixations

The identification of fixations is based on two properties: the low eye velocity and the low dispersion of the gaze coordinates during a fixation. As shown in Chapter 2, the angular velocity of the eye during fixations is low. Due to noise in the extraction of eye features, the gaze coordinates provided by the gaze tracking system might be noisy, increasing the measured velocity. Combining a velocity-based stage with a dispersion-based stage allows to increase the robustness in the classification. These two stages are described below.

Velocity-Based Stage

This first stage is based on the I-VT algorithms presented in Section 11.1.1. For each sample i, the velocity of the eye movement $\dot{\theta}_i$ of point (x_i, y_i) with respect to the previous point (x_{i-1}, y_{i-1}) is measured and compared to the threshold $\dot{\theta}_{max}$. The value of this threshold in the current implementation is set to 40°/s. If the measured velocity is below the threshold, the new pair of coordinates is added to the window and the algorithm proceeds to the second stage. Otherwise, the window is cleared and the system reports a saccade.

Dispersion-Based Stage

The second stage of the algorithm consists of a dispersion-based technique, and checks for the dispersion of the points in the current window. A minimum duration of t_{min} = 120 ms is required before the algorithm reports a fixation is taking place. Once the time span in the window is greater than t_{min}, the dispersion of the coordinates in the window is computed and compared to the maximum dispersion D_{max} value of 1° of visual angle.

A pure velocity-based method can yield incorrect results when the velocity is shifting around the threshold value $\dot{\theta}_{max}$ [63], [15], and a series of short fixations are therefore reported by the algorithm. Adding the temporal and spatial constraints in this second stage avoids such very short fixations, increasing the robustness of the algorithm.

In order to calculate the eye velocity in degrees per second, the size of the screen and the distance from screen to user need to be known. The size of the screen in mm and pixels can be either measured physically, or obtained through the API[1] of the computer's graphics device. The distance between user and screen is assumed to be 60 cm.

11.2.2 Smoothing Stage

Once a fixation has been detected, the cursor position is smoothed in order to reduce the jitter produced by noise in the image and eye micro movements. The smoothed

[1] Application Programming Interface

coordinates (x_s, y_s) are calculated by means of a moving average applied to a window containing the last N gaze coordinates, as expressed in Equation 11.2.

$$x_s = \frac{1}{N} \sum_{i=0}^{N-1} x_i \tag{11.2}$$

The y coordinate would be calculated with a similar expression. In the current implementation the window size is set to $N = 15$ images. Assuming a frame rate of 30 images per second, the averaging is performed over 500 ms. When a saccade is detected, the window is cleared and the most recent sample is used to update the cursor position.

11.2.3 Evaluation

The algorithm described above is integrated with the smooth pursuit detection algorithm presented in Chapter 12. Therefore, only an informal evaluation of the smoothing properties of the algorithm is given here.

Figure 11.1 shows a sequence of gaze coordinates composed of two consecutive fixations separated by a saccade. Due to the noise, the points given by the gaze tracking system (shown as red squares) are jittery. When the fixation is detected, the coordinates are smoothed (blue stars) and the signal tends to be concentrated around the same point.

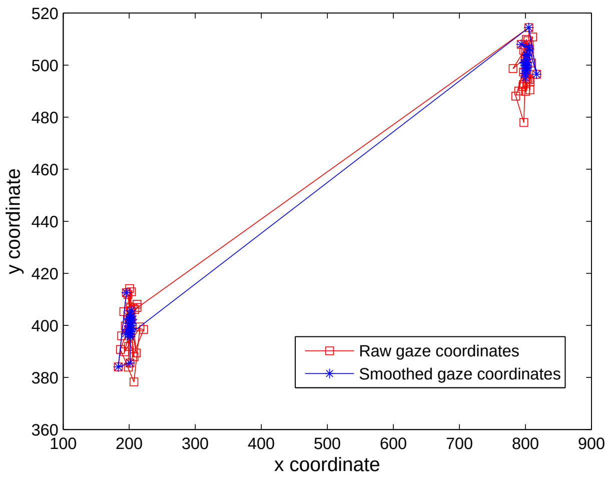


Figure 11.1: Fixation detection. Red squares show the raw gaze coordinates; blue stars show the smoothed coordinates, which tend to be more concentrated around the same point. (Note the different scale in x and y).

Figure 11.2 shows the behavior in the y coordinate over time. Smoothing reduces the spread around the fixated point, thereby making the cursor more stable.

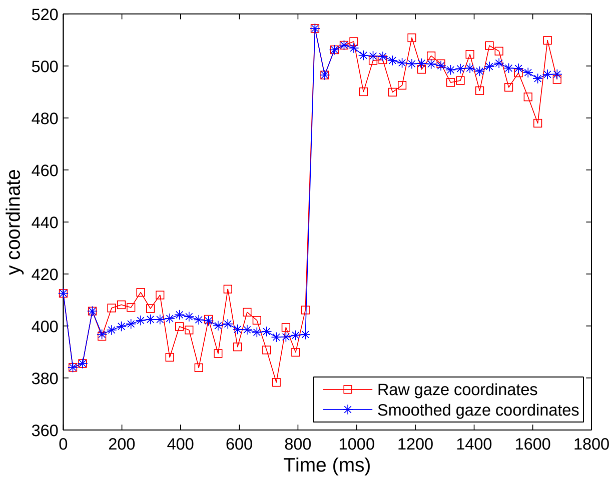


Figure 11.2: Smoothing of the y coordinate. Red squares show the raw y coordinate; blue stars show the smoothed y coordinate.

11.3 Discussion

The fixation detection algorithm presented in this chapter allows to robustly classify the gaze coordinates given by the gaze tracking system as belonging to either a saccade or a fixation. This identification is performed on basis of the measured eye velocity between consecutive samples and the dispersion of the samples in a window containing the last N points.

The algorithm proposed detects fixations in real time and allows to smooth the signal during a fixation to reduce the noise in the cursor position. It has been informally evaluated by using it together with the ITU Gaze Tracker with successful results. It is also possible to incorporate the algorithm in any gaze tracking system; however, the parameters might need to be modified to suit the different characteristics of the system, e.g. regarding noise.

The algorithm requires the specification of three parameters: maximum eye velocity $\dot{\theta}_{max}$, minimum fixation duration t_{min}, and maximum dispersion D_{max}. Since these parameters are used as thresholds to classify a series of samples as belonging to a fixation, the algorithm might be prone to errors when measured values are around the threshold values. For this reason, a low number of thresholds is desired, and ideally no thresholds should be required. However, although combining velocity and dispersion techniques requires specifying more thresholds, it also increases the robustness in the detection of fixations [13], [63].

The smoothing stage introduced by the fixation detection algorithm has a negative impact on the interaction during smooth pursuit movements, since the cursor position is averaged over the last N samples, thereby introducing a lag. This is explained in more detail and addressed with the inclusion of a smooth pursuit detection algorithm in the next chapter.

Chapter 12

Smooth Pursuit Detection

Chapter 11 presents an overview of real-time fixation detection algorithms and describes an implementation of a velocity- and dispersion-based algorithm. This chapter extends the eye movement detection algorithm by incorporating a novel smooth pursuit detection technique, which is described and evaluated in Section 12.3. The motivation for a smooth pursuit detection algorithm and the previous work described in the literature are presented first.

12.1 Detection of Smooth Pursuit Movements

Interactive elements in current graphical user interfaces are primarily static, and therefore most of the interaction is based on pointing and selecting objects on the screen. To ease point-and-select tasks, gaze tracking systems often integrate a fixation detection algorithm that allows to smooth the cursor position when the user is looking at an object, thus removing the jitter associated with gaze pointing (see Chapter 11).

However, there are applications that include dynamic environments where objects move on the screen, requiring the user to maintain the pointer on the moving object in order to interact with it. The most obvious example is video games. For instance, a first person shooter game presents enemies that move following different paths; the player is required to point at these enemies in order to shoot at them. But there also exist eye-typing applications that present dynamic environments. Dasher [86] and StarGazer [27] (see Section 5.2) require tracking the letters on the screen while they move towards the center of the interface, where they are selected and written in the textbox. These kinds of interfaces require performing smooth pursuit movements, and might therefore benefit from a smooth pursuit detection algorithm that can smooth and predict the signal accordingly.

12.1.1 Sources of Error in Dynamic Interfaces

When dynamic objects are present in the user interface, gaze interaction is greatly affected by time delays occurring in the system. Two main sources of error can be identified: jitter and lag.

Jitter

Even if the eyes are following a target moving on a straight line, the noise in the system will make the cursor position jump around that line. Figure 12.1 shows the x coordinate of a target moving on a straight line, and the estimated PoR while the user tracks the target. Data have been recorded with the ITU Gaze Tracker. Gaussian noise with $\sigma = 30$ pixels has been added to the signal. Due to the noise, the signal does not follow the target smoothly.

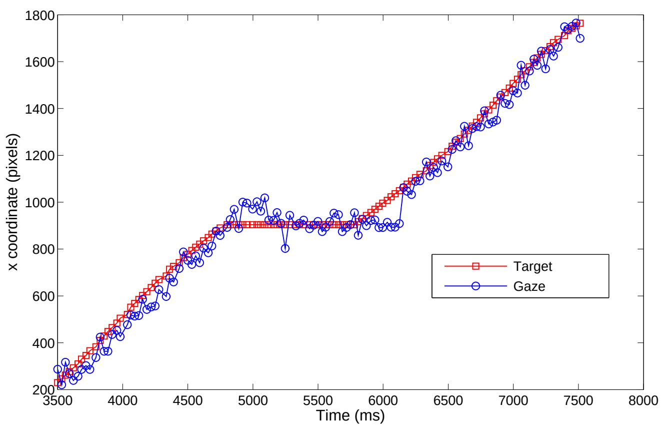


Figure 12.1: Smooth pursuit movement and fixation (data recorded on ITU Gaze Tracker). The x coordinates of the target (red star) and the estimated PoR (blue circle) are plotted against time.

Lag

In a gaze tracking system, there is always a time delay between the moment the eyes move, and the moment the cursor position is updated. In point-and-select tasks this delay is very often not noticeable, but it does have an effect during smooth pursuit movements. Due to the continuous movement, the delay will make the cursor lag behind the target. We can identify two main sources of delay: (1) system delay, and (2) smoothing delay.

- System delay: there is a delay between the moment an image of the eye is recorded and the moment the information is used to drive the cursor, due to the time required to grab the image, analyze it and extract the eye features, and calculate the new gaze coordinates. Low-cost cameras (e.g. webcams) are usually slower than high-end cameras, therefore increasing the time delay.

- Smoothing delay: smooth pursuit movements can easily be mistaken with fixations, especially when the eye velocity is low. A smooth pursuit movement will be regarded by the system as a series of fixations, and the smoothed cursor position will make it lag behind the target. Figure 12.2 shows the x coordinate of the target position (red square) and of the estimated PoR (blue circle) during a smooth pursuit task after applying a fixation detection algorithm. Data have been recorded with a Tobii 1750 system with parameters set to highest responsiveness. Due to the low eye velocity, the system detects a fixation, and the cursor position is smoothed until the dispersion is too high. The cursor position is then corrected, and a new fixation detected. The pattern produced has a saw-like shape. Deactivating the smoothing can solve the issue, but the cursor will then be jittery as shown in Figure 12.1 (note that in some systems, such as in the Tobii 1750, it is not possible to deactivate the smoothing).

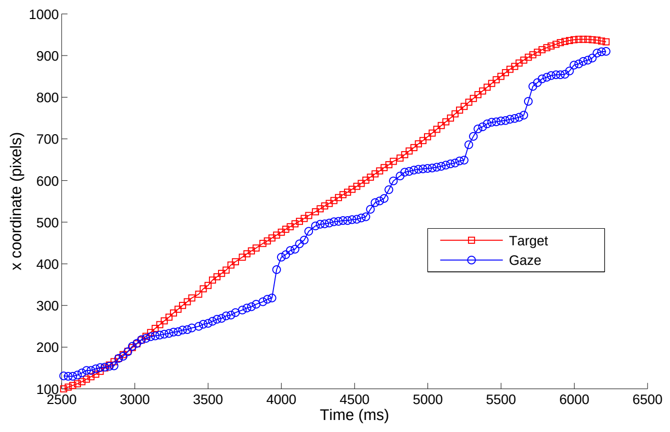


Figure 12.2: Smooth pursuit movement (data recorded on a Tobii 1750). The x coordinates of the target (red star) and the estimated PoR (blue circle) are plotted against time. A saw-like pattern in the cursor position is produced due to the smoothing of the fixation detection algorithm.

Most gaze tracking systems include a fixation detection algorithm to smooth the signal during fixations. However, few (if any) include methods to detect smooth pursuit movements. Interaction in dynamic environments such as video games will be affected, as behaviors like the one shown in Figure 12.2 will occur. In order to smooth the signal properly, it is necessary to apply a different smoothing algorithm suited to the type of movement that the eye is performing.

Dealing with the inherent lag between the target position and the PoR given by the gaze tracking system is also a challenging topic. Komogortsev et al. describe in [41] an eye movement detection algorithm that makes use of a Kalman filter to predict the next sample, and propose a metric to measure the accuracy of the prediction based on the root mean square error (RMSE) between the predicted gaze coordinates in time *k*, $x_{k|k-1}$ (predicted in time *k-1*) and the observed coordinates, z_k:

$$RMSE = \sum_{k=i}^{j} \frac{\sqrt{(\hat{x}_{k|k-1} - z_k)^2}}{j - i} \tag{12.1}$$

Ideally, the RMSE should be 0°, since that would mean perfect prediction.

12.2 Previous Work

The detection of smooth pursuit movements for real-time, gaze-based applications has not been as extensively investigated as the detection of saccades and fixations. There are, however, some previous attempts to classify the signal provided by the eye tracking system as smooth pursuit. Sauter et al. presented an algorithm to separate smooth pursuit and saccades. They use a Kalman filter to predict the state in the next sample, and a χ^2 test on the error between the recorded and the predicted eye velocities. When the χ^2 value is higher than a pre-specified threshold, a saccade takes place. A limitation of their method is the inability to separate fixations from smooth pursuit movements.

Komogortsev and Khan [40] followed a similar approach to detect eye movements. They term their Kalman filter as Attention Focus Kalman Filter (and Two State Kalman Filter, or TSKF, in a later work, [41]). A χ^2 test is used to classify saccades. Fixations are detected by imposing an eye velocity below 0.5°/s during a minimum duration of 100 ms. Smooth pursuit movements are detected when the eye state is not a saccade nor a fixation, and the eye velocity does not exceed 140°/s.

Komogortsev and Khan extended their method in [41] with an Oculomotor Plant Kalman Filter (OPKF). The TSKF is used to predict low-velocity movements and detect the onset of saccades, while the OPKF is used to predict the trajectory during the saccade. The RMSE between observed and predicted value given by Equation 12.1 is around 1° during fixations and around 3° during smooth pursuit movements.

12.3 Eye Movement Detection Algorithm

The eye movement detection algorithm proposed in this thesis consists of two steps:

- Eye movement detection: classifies the gaze samples as fixations, saccades or smooth pursuits based on the eye velocity and movement pattern.
- Signal smoothing: the cursor position is smoothed according to the type of eye movement: no smoothing for saccades, moving average for fixations, and Kalman filter for smooth pursuits. During smooth pursuits the next sample is also predicted by means of the Kalman filter.

These steps are described next.

12.3.1 Detection Algorithm

The detection of the type of eye movement is based on the eye velocity and the movement pattern of the gaze coordinates given by the gaze tracker.

- Velocity: As seen in the previous chapter, the measured eye velocity can be used to classify gaze samples as fixations or saccades. Smooth pursuit movements, however, can have similar velocities to fixations (see Figure 2.6). Therefore, a maximum angular velocity threshold $\dot{\theta}_{max}$ is employed to separate high-speed movements (saccades) from low-speed movements (fixations and smooth pursuits). As noted in Chapter 11, the value of this threshold depends on the characteristics of the gaze tracker.
- Distribution of eye movements: during a fixation, the noise in the gaze tracking system introduces jitter in the cursor position. This can be regarded as random noise around the fixation position. During a smooth pursuit, the eye follows a path while tracking the target. Therefore, even in the presence of noise, the estimated gaze coordinates will also follow a path.

Figure 12.3 illustrates this behavior of the eye movements. An initial smooth pursuit takes place, followed by a fixation, and finalized by another smooth pursuit (movement occurs from right to left). There is noise in the estimated gaze coordinates due to inaccuracies in the extraction of eye features. The estimated instant velocity of the eye $\dot{\theta}$, measured in $^\circ$/s, is shown in Figure 12.3(b). Although it tends to be higher during the smooth pursuit movements, the velocities during fixations and smooth pursuits overlap, making it difficult to differentiate between both eye movements on the basis of the instant velocity.

When the gaze tracking system provides a new PoR p_i, the eye movement detection algorithm determines the angular velocity $\dot{\theta}_i$ between the new point and the previous point, and compares it to the velocity threshold $\dot{\theta}_{max}$. If $\dot{\theta}_i < \dot{\theta}_{max}$, the point is added to a moving window of size N.

In order to determine whether the eye is performing a fixation or a smooth pursuit movement, the direction of movement is analyzed. Figure 12.4 shows a sequence of 4 points p_i. A line is fitted to the points belonging to the window. This line gives an indication of the direction of movement. When the window contains 2 points, the line r_1 that connects both points is calculated. When the new pair of gaze coordinates p_3

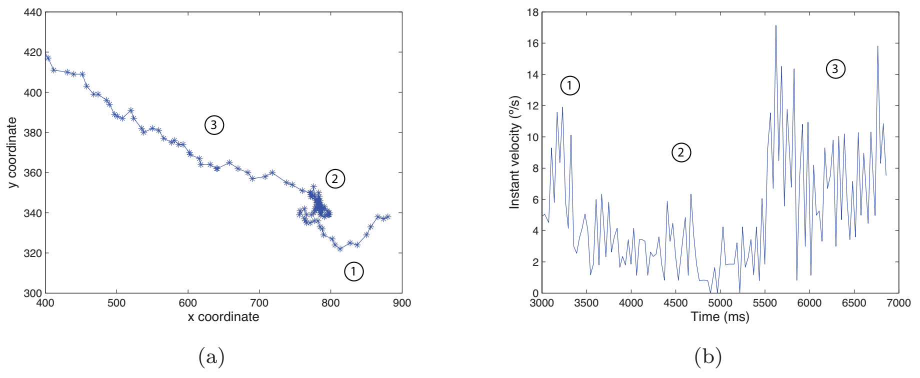


Figure 12.3: Sequence consisting of (1) a smooth pursuit movement, (2) a fixation, and (3) a second smooth pursuit (movement occurs from right to left). a) Gaze coordinates. b) Instant velocity between subsequent samples.

is calculated, a new line r_2 is fitted to the three points. The angle between the new line and the previous line indicates the amount of change in direction.

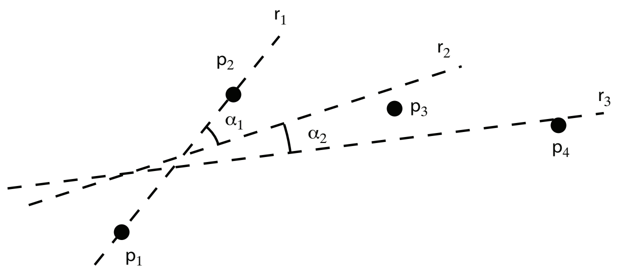


Figure 12.4: Sequence of 4 points in a smooth pursuit movement. A line is fitted to the points in the window and the angle with the previous direction calculated.

When the estimated gaze coordinates points are following a path, the change in direction is low. To illustrate this behavior, consider the gaze path shown in Figure 12.3 above. Figure 12.5 shows the standard deviation of the angles between the lines formed by the gaze points in the window of size N. When the eye is performing a fixation, the noise introduces jitter in the cursor position, and the line that fits to all points produces a bad fit. The angles α_i formed by the different lines will therefore take a wide range of values, producing a high standard deviation, as can be seen in the middle part of the graph in Figure 12.5.

On the other hand, if the eye is tracking a moving target, the points tend to follow a linear trend. It is possible to fit a line to the points, and the angles that all the lines fitted to the points in the window are low (ideally 0, if the target follows a rectilinear movement and there is no noise in the tracker). Therefore, the distribution of the angles α_i is narrow, and the standard deviation is low. This is shown in the beginning and ending parts of the graph in Figure 12.5.

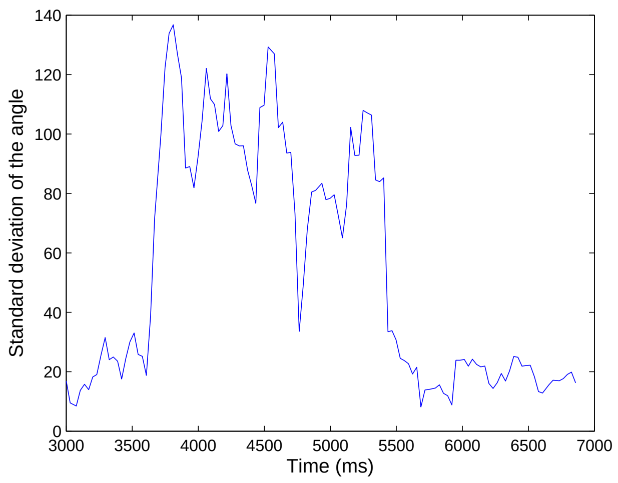


Figure 12.5: Standard deviation of the angles between the lines formed by the points in the current window. The standard deviation is low during smooth pursuit movements (beginning and end) and high during fixations (middle part).

Since the direction of the target can change over time, the calculations of direction and angles are performed over a small window of N samples. The points belonging to the window are classified as either a fixation or a smooth pursuit by calculating the standard deviation of the angles α_i in the window and comparing the value to a threshold α_{max}. This threshold is calculated empirically, and its value will differ depending on the characteristics of the gaze tracker.

12.3.2 Smoothing and Prediction

The algorithm introduced above gives the type of movement that the eye is performing: saccade, fixation or smooth pursuit. Each type of eye movement requires a different type of smoothing to avoid introducing delay.

- Saccades: the signal is not smoothed in order to preserve the responsiveness of the system during these fast eye movements.

- Fixations: a moving average is applied on a window of N samples, as explained in Chapter 11.

- Smooth pursuit: a moving average smoothing introduces a delay, producing a lagging cursor (see Figure 12.2). In order to maximize the responsiveness, a Kalman filter is applied to a window of N points. Furthermore, the Kalman filter is used to predict the next sample.

Kalman Filter

A short introduction to the Kalman filter is given in Appendix A.

The observable variables given by the gaze tracking system are gaze coordinates and time. During smooth pursuit movements, the coordinates of the PoR at any given time step k, (x_k, y_k), can be computed from the coordinates in the previous time step, (x_{k-1}, y_{k-1}), and the measured angular eye velocity $(\dot{x}_{k-1}, \dot{y}_{k-1})$. Equation 12.2 represents this calculation for the x axis.

$$x_k = x_{k-1} + \dot{x}_{k-1}\Delta t \tag{12.2}$$

where Δt is the time between samples, 33 ms in the current implementation. Here we assume that the velocity can be computed as $\dot{x}_k = \frac{x_k - x_{k-1}}{\Delta t}$.

We can rewrite Equation 12.2 in matrix form, giving:

$$x_k = A_k x_{k-1} + B_k u_k \tag{12.3}$$

In this model, we assume the control input u_k to be zero. The state transition matrix is assumed to be constant, and is expressed as:

$$A = \begin{bmatrix} 1 & \Delta t \\ 0 & 1 \end{bmatrix} \tag{12.4}$$

The observation matrix H is also assumed to be constant. Only the coordinates can be measured, hence H is given by the following expression:

$$H = \begin{bmatrix} 1 & 0 \end{bmatrix} \tag{12.5}$$

Therefore, the system state constructed for the x axis is given by:

$$\begin{bmatrix} x_k \\ \dot{x}_k \end{bmatrix} = \begin{bmatrix} 1 & \Delta t \\ 0 & 1 \end{bmatrix} \begin{bmatrix} x_{k-1} \\ \dot{x}_{k-1} \end{bmatrix} + \begin{bmatrix} w_p \\ w_v \end{bmatrix} \tag{12.6}$$

where the system noise vector w has been divided into w_p, the noise related to the PoR, i.e. the gaze coordinates provided by the gaze tracking system, and w_v, the noise related to the eye velocity. The model for the measurements is:

$$x_k = \begin{bmatrix} 1 & 0 \end{bmatrix} \begin{bmatrix} x_k \\ \dot{x}_k \end{bmatrix} + v_p \tag{12.7}$$

The covariance matrix of the system noise, Q, is calculated as $Q = E[(w - E(w))(w - E(w))^T]$. It is assumed that both variables w_p and w_v are uncorrelated. Therefore, the covariance matrix Q is given by:

$$Q = \begin{bmatrix} \delta_p & 0 \\ 0 & \delta_v \end{bmatrix} \tag{12.8}$$

where δ_p is the variance of variable w_p and δ_v is the variance of variable w_v. According to Komogortsev et al. [41] w_p is related to the "noise" in the eye during fixations. Due to the micromovements that take place during fixations (see Chapter 2), the standard deviation δ_p can be considered to be 1°. The standard deviation of w_v is more difficult to measure. The value chosen is $\delta_v = 1°/\text{s}$. The values of both δ_p and δ_v are in accordance with the values used by Komogortsev et al. in [41].

The covariance matrix of the measurement noise, R, is related to the noise in the gaze tracking system [41]. Since only the gaze coordinates are measured, it is a scalar, v_p. The value will depend on the actual measurement noise of the system. For simplicity, the error is assumed to be 1° of visual angle. The value could, however, be measured during calibration of the gaze tracking system, thereby allowing to construct a more accurate model.

The filter needs to be initialized with position and velocity at time step $k = 0$. When a new smooth pursuit movement is detected, the initial position is initialized with the current gaze coordinates, and the initial velocity is computed as the average velocity in the current window. Note that each filter (one for each axis) is initialized independently.

Reconstruction and prediction of the signal are carried out as described in Appendix A. The gaze coordinates $(\hat{x}_{k+1}^-, \hat{y}_{k+1}^-)$ in the next time step are predicted using Equation A.3 for each axis. As explained in Section 12.1, the RMSE between the coordinates predicted by the Kalman filter and the observed gaze coordinates can be used to measure the accuracy of the prediction by means of Equation 12.1.

12.3.3 Evaluation

The eye movement detection algorithm presented above has been evaluated using a task in which participants had to perform the three types of eye movements.

Participants and Apparatus

Five participants, with ages ranging from 28 to 49, volunteered to take part in the experiment. Three of them had previous experience with gaze tracking systems. One of them wore glasses and another one contact lenses.

An Alea Technologies IG-30[1] system was employed to track the participants' eye movements. The system works at 50 Hz and has a typical accuracy of 0.5°. No smoothing was applied to the gaze coordinates, which increased the jitter in the gaze coordinates significantly compared to the smoothed signal. The device was mounted on a 19" monitor with a resolution of 1280×1024 pixels.

[1]http://www.alea-technologies.de

Experimental Procedure

The experiment required the participants to look at a series of targets displayed on the screen by a software application programmed in C#. Each block consisted of 4 trials, while each trial consisted of 2 targets that were presented on the screen one at a time. One target was static and required a click by the user, while the other was dynamic and moved from one side of the screen to the other with a constant velocity and following a straight line. The starting x coordinate was fixed, while the y coordinate was chosen randomly. The direction of movement (left to right or right to left) was random. The target stopped during two seconds at a random time after starting movement, and then continued moving until reaching the end of the screen.

Static and dynamic targets were displayed alternatively. When a static target was shown, the participant was instructed to fixate on it and select it by pressing the mouse button. Once selected, the target disappeared and a dynamic target was shown. The participant was instructed to follow it with his or her eyes. Figure 12.6 shows a hypothetical sequence in one trial. Participants completed 3 blocks, each with a different target velocity: 4, 8 and 12°/s.

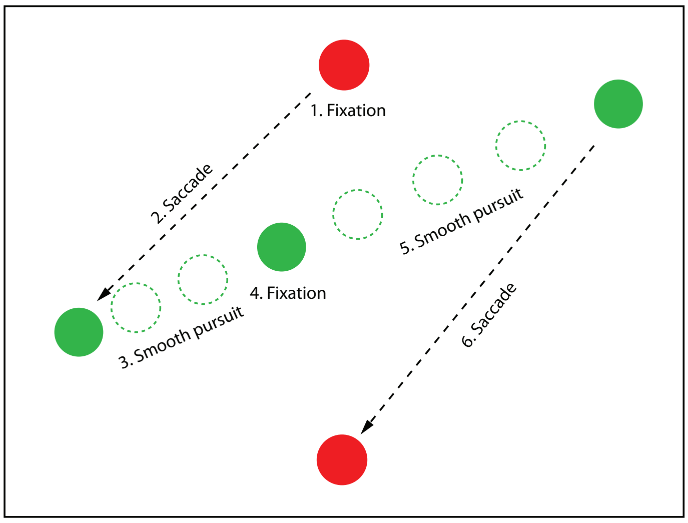


Figure 12.6: Example of one trial. The static target (red) is displayed. When the user clicks, the dynamic target (green) appears on one side and moves to the other side, stopping during two seconds around mid way. A new static target then appears and a new trial begins.

The following parameters were used in the algorithm:

- Saccade velocity $\dot{\theta}_{max} = 100°/s$
- Minimum fixation duration $t_{min} = 120$ ms
- Maximum fixation dispersion $D_{max} = 3°$

- Maximum angular dispersion for smooth pursuit $\alpha_{max} = 20°$
- Time span between samples $\Delta t = 33$ ms
- Window size $N = 12$ images

Three metrics were measured in each trial: proportion of correctly detected samples, the delay between stimuli and detection, and the angular error in the prediction of the next sample during smooth pursuit movements (measured as the RMSE, see Equation 12.1).

The proportion of correctly detected samples was measured by comparing the type of eye movement reported by the algorithm with the type of movement of the stimuli ("fixation" or "smooth pursuit"). When both coincided, the type of eye movement was considered to be successfully detected. The proportion of *samples correctly classified* and *total number of samples* was calculated for each block. A result of 1 would mean perfect classification.

Results

Figure 12.7 shows the proportion of samples where the eye movement was detected correctly for each type of eye movement and target velocity. Results are given separately for fixations and smooth pursuits, and for each target velocity. Due to the delay between stimuli and gaze coordinates, saccadic movements are not considered.

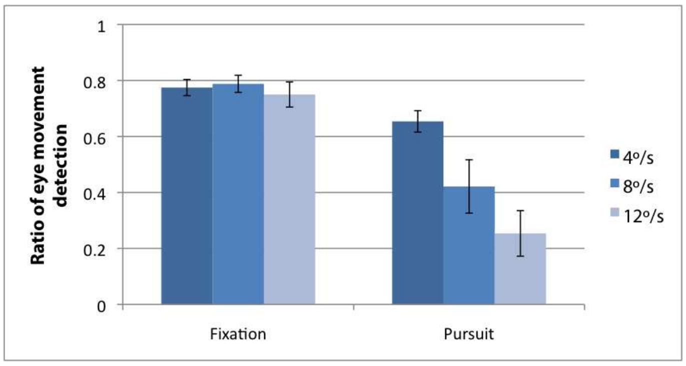


Figure 12.7: Average proportion of properly detected fixations and smooth pursuits for each target velocity.

The overall mean of fixations successfully detected is 77%. There are two explanations for the samples detected incorrectly: (1) the delay between stimuli and gaze coordinates; and (2) during a fixation the eye drifts and a sequence of samples on a line is recorded and interpreted by the algorithm as a smooth pursuit movement.

In the case of smooth pursuit, the overall mean of samples correctly classified is 44%. The ratio of properly detected smooth pursuits decreases as the target velocity increases. For low target velocities (4°/s) the percentage of samples correctly classified as smooth pursuit is 65%, while for high target velocities (12°/s) this value drops to 25%.

Figure 12.8 shows an example of one trial consisting of 140 samples where the target moves at a velocity of 4°/s.

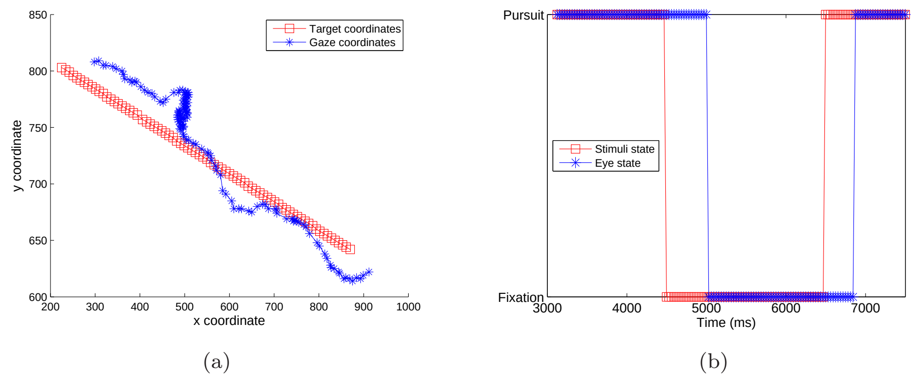


Figure 12.8: Sequence consisting of a smooth pursuit movement, a fixation, and a smooth pursuit. a) Target coordinates (red square) and gaze coordinates (blue star). b) Stimuli state and detected eye state. Note the delay in the detection of the eye state.

Figure 12.8(a) shows the target coordinates and the gaze coordinates given by the gaze tracking system. The target moves from right to left, stops during 2 s, and continues moving. There is an offset between gaze coordinates and target due to the inaccuracy of the gaze tracking system. Nevertheless, this offset is irrelevant for the current analysis.

Figure 12.8(b) shows the type of movement the eye is performing over the time lapse under study. The y axis shows the type of movement, i.e. fixation or smooth pursuit. Red squares show the state of the target, while blue stars show the state of the eye reported by the algorithm. Even though the type of eye movement is properly detected, there is a delay between stimuli and reaction. Due to the delay, the proportion of properly detected type of eye movement in this sequence is 0.79. The delay has therefore an impact on the detection ratio, and is analyzed next.

Figure 12.9 shows the average delay in the detection of fixations and smooth pursuits for each target velocity. The delay is measured as the time between the stimulus starts and the proper eye movement is detected.

The average delay in the detection of fixations is 374 ms (SD = 44), while in the case of smooth pursuit the average delay is 576 ms (SD = 79). This delay is caused by: (1) a *human* delay due to the reaction time to the new stimuli, and (2) an *algorithm* delay due to the time required to detect the type of eye movement. The delay is longer in the detection of smooth pursuit movements than fixations. The reason is that often a fixation is detected before the smooth pursuit movement. Figure 12.10 illustrates

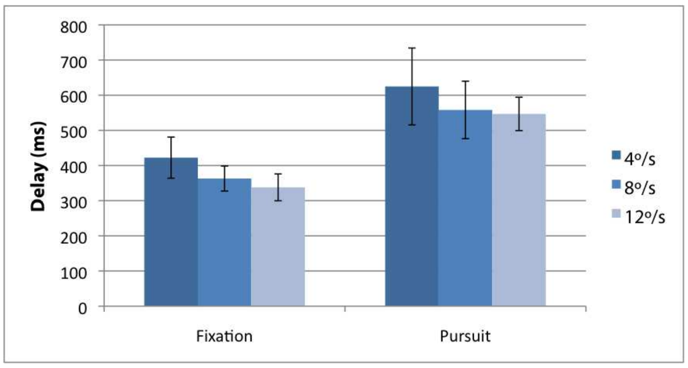


Figure 12.9: Delay in the detection of fixations and smooth pursuits for each target velocity.

this behavior. The dynamic target is displayed after the static target, and the eye executes a saccade to reach the moving target. After the saccade, a new fixation is considered before the algorithm detects that the eye gaze coordinates follow a linear trajectory.

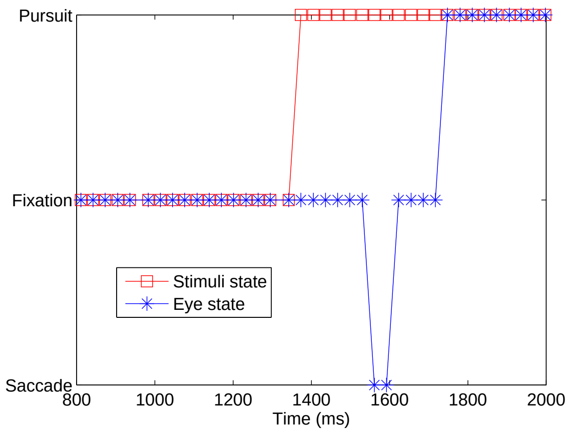


Figure 12.10: Average delay in the detection of a smooth pursuit movement.

As seen in Section 12.1, the delay in the gaze tracking system makes the cursor lag behind the target. In order to deal with this delay, it is possible to use the predicting properties of the Kalman filter and update the cursor position with the predicted gaze coordinates instead of the current ones.

Figure 12.11 shows the average root mean square error (RMSE) between the observed gaze coordinates and the coordinates predicted by the Kalman filter (using

Equation A.3).

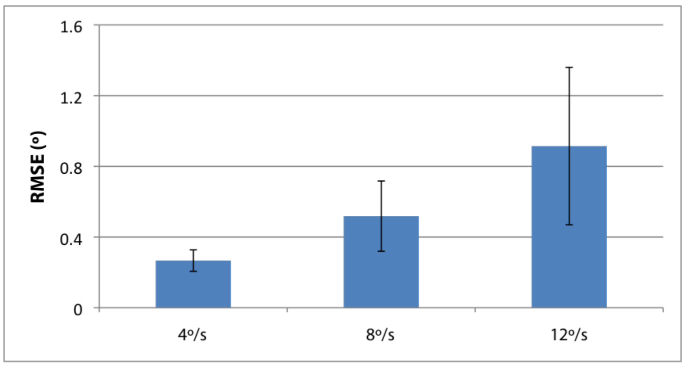


Figure 12.11: Average RMSE between predicted and observed gaze coordinates.

The overall average RMSE is 0.57°. The prediction performance is affected by the target velocity: as the velocity increases, the accuracy of the predicted gaze coordinates decreases. For low velocities (4°/s), the average RMSE is 0.27°, while for high velocities (12°) the RMSE is 0.91°. The RMSE values obtained in this study are notably lower than the ones reported by Komogortsev et al. in [41], where the average RMSE values are in the range of 2.65 - 3.12°.

12.4 Conclusions

An eye movement detection algorithm has been presented in this chapter. The algorithm is composed of two stages: a detection stage, where the gaze coordinates given by the gaze tracking system are classified as belonging to a fixation, saccadic movement, or smooth pursuit; and a smoothing stage, where the signal is smoothed to reduce the noise without introducing lag.

Classification of the type of eye movement is performed on basis of the measured eye velocity and the movement pattern of the gaze coordinates. Fixations are correctly detected in 77% of the cases, while smooth pursuits are detected in 44% of the cases. The proportion of properly detected smooth pursuits decreases as target velocity increases.

The gaze coordinates are smoothed during fixations and smooth pursuits. The moving average technique used during fixations is described in the previous chapter. During smooth pursuit movements, the trajectory followed by the measured gaze coordinates is calculated by means of a Kalman filter, thus allowing to smooth the signal without introducing delay. Furthermore, the delay introduced by the gaze tracker is partly compensated by predicting the next sample. The average error between prediction and observed value is 0.57°. It must be noted that these tests have been performed off-line, and therefore it is not possible to know how it feels to have a cursor that moves “ahead” of time. An experiment where the cursor position is updated with the predicted coordinates should be carried in real time to further investigate the potential of prediction during smooth pursuit movements.

The algorithm described is based on thresholds, and is therefore dependent on the characteristics of the gaze tracking system. The values of the thresholds need to be tweaked depending on parameters such as noise and delay. For instance, if the noise in the gaze coordinates is high, the measured eye velocity and the dispersion during fixations will take higher values, and the $\dot{\theta}_{max}$ used to classify samples as saccadic movements or fixations or smooth pursuits might need to be modified. It must be noted that the algorithms have been developed using gaze tracking systems with a low temporal resolution (15 to 50 Hz), whereas many new commercial systems have sampling rates of 125 Hz. This increase in frame rate can have a big impact on the performance of the algorithm proposed.

As noted above, the methods to classify the eye movements and smooth the cursor position accordingly presented in this chapter can help reduce the noise and thereby provide a more stable pointer. However, selections based on gaze only are still slow. The next chapter explores multimodality as a way to provide a fast selection technique in gaze interaction, and a combination of gaze pointing and EMG clicking is investigated and evaluated.

Chapter 13

Hands-Free Pointing and Clicking

As discussed in Chapter 5, interaction with graphical user interfaces is based on pointing and selecting objects presented on the interface, usually making use of a pointing device such as a mouse or a touchpad. When we interact with an interface, our eyes normally point at the desired object before we actually the mouse to place the cursor over it. Therefore, our eyes are on the object long before the cursor arrives. Gaze pointing can take advantage of this: if the cursor moves following our eye movements, the time required to point at the object with the mouse is eliminated. However, contrary to usual input devices, our eyes lack a selection mechanism that can confirm a selection (e.g. a button).

This chapter explores multimodality as a means to provide a fast selection mechanism that can complement and enhance gaze pointing. Since people with severe motor-skill disabilities are often unable to use hand-controlled devices, the selection techniques investigated here are hands-free. Namely, an EMG switch is used to perform selections. Two studies where gaze pointing is combined with EMG clicking are presented. The first study explores the speed advantage of using a combination of gaze pointing and EMG clicking over using a mouse, while the second study describes a technique that aims to reduce the undesired activations that take place in noisy environments.

We first begin by presenting different approaches that allow the user to perform selections, both using the eyes only and through the use of an EMG switch.

13.1 Selection in Gaze-Based Applications

Eye movements provide a fast pointing technique. A selection technique that can complement gaze pointing by providing fast selections would be desirable. However, finding a reliable method to perform selections in gaze-based applications without losing the hands-free advantages is not a trivial problem. Two approaches can be taken: (1) relying on the eyes only, and (2) adding a second device (e.g. a switch) that decouples pointing and selecting.

13.1.1 Selection by Gaze Only

In gaze-based systems, the two most common selection methods are dwelling and blinking. When using dwelling as the selection method, the system issues an activation every time the user stares at a target for longer than a predefined threshold duration (i.e., dwell time). Relying only on dwell time for activation leads to the Midas Touch problem (see Chapter 3 and [33]): undesired activations occur when the user stares at a target with the only purpose of looking at it and not activating it. A short dwell time might improve performance and responsiveness, since waiting time is reduced, but more undesired activations will occur.

Selection by blinking allows the user to actively confirm every activation by blinking an eye, thereby eliminating the Midas Touch problem. However, blinks also occur naturally and frequently when the user does not intend to issue an activation, leading to some natural blinks being mistaken for activations. Furthermore, blinking for every selection can become tiring for the user after some time using the system.

Modern GUIs present icons and objects as small as 9×9 pixels. The accuracy offered by most current video-based gaze trackers makes it difficult to select such small objects. Furthermore, low-cost systems have a lower accuracy than high-end commercial systems, and it can be challenging to select even rather big objects. Alternative gaze-based selection methods that deal with the inherent noise of a gaze tracker have been proposed. Two-step dwell [43] consists of two activations: the first activation magnifies the area surrounding the location where the activation was issued, while the second activation performs the actual selection. This allows selecting small objects that would be difficult to select otherwise.

An obvious disadvantage of this method is the increased final selection time compared to normal dwell: for every selection, two activations are required, one to magnify and another one to perform the selection. To alleviate this problem while still dealing with a noisy input, a zooming technique can be considered [4]. Using a zooming paradigm for selection, the region around the gazed position is increased in size in a continuous way as long as the user maintains a fixation. The activation is issued when a maximum predefined magnification is reached. Zooming allows the user to dynamically control the final selection coordinates during the magnification process. Skovsgaard et al. [70] conducted a thorough study to evaluate the performance of dwell, 2-step dwell and zooming selection techniques on small targets (6×6, 9×9 and 12×12 pixels). According to their results, error rates decreased by 70% when 2-step dwell was used compared to normal dwell and 44% compared to zooming. Although selection times with zooming were 96% lower than with 2-step dwell, participants subjectly perceived 2-step dwell to be faster and easier to use.

13.1.2 Multimodality: Combining Gaze Pointing and EMG Clicking

Arguably, gaze-only selection techniques are unnatural and slow down interaction. The speed advantage that the fast eye movements offer compared to hand or arm movement in the pointing process is lost due to an inefficient selection process. A way to address this issue is adding an extra input "device" that can perform selections. If we take the mouse as an example, we can consider it as a pointing device, while the buttons constitute the selection device. In a similar way, gaze pointing can benefit from adding an extra input modality with the sole objective of performing selections. As mentioned in Chapter 5, Zhang and MacKenzie [95] evaluated gaze pointing in combination with short dwell, long dwell and key activation. Activation by key yielded the best results in terms of throughput and error rate, emphasizing the fact that multimodal input outperforms dwell activation, even for short dwell times. Furthermore, multimodality provides the user more control of when to perform an activation, thus eliminating the Midas Touch problem.

Using a key or mouse button in combination with eye movements removes the hands-free advantage of gaze interaction. In some instances, the user might not be able to operate a hand-controlled device. For example, a severely disabled person might have lost the ability to move his or her hands. While operating a patient, a doctor might interact with a head-mounted display using their eyes to obtain relevant information regarding the patient and the operation, being unable to control a keyboard or a mouse. In such cases alternative selection devices are to be considered.

An electromyographic (EMG) switch constitutes a promising selection device that can compete with the speed of a mouse button while still maintaining the hands-free advantage of gaze pointing versus conventional pointing devices. Furthermore, the technology is becoming mainstream and commercial systems can be found for around $100 as of this writing. An EMG switch is based on electromyographic signals, which can be recorded and analyzed in real-time to detect contraction and relaxation of muscle cells by measuring the difference in electric potential. Two ways of obtaining EMG signals can be considered: intramuscular EMG, in which a needle is inserted into the muscle tissue; and surface EMG, where some electrodes are placed on the skin of the user. Intramuscular EMG is very invasive and therefore surface EMG is preferred for HCI applications. Figure 13.1 shows a user wearing two electrodes on his forehead. These electrodes are connected to an amplifier that enhances the signal, and then to a computer that is in charge of analyzing the signal and detecting muscle contractions. The system shown in Figure 13.1 has been developed at the Public University of Navarra and is used in the experiment described in Section 13.3.

The Neural Impulse Actuator[1] (NIA) is a commercial low-cost (around $100 as of this writing) device that allows the user to interface with a computer by means of three different biosignals, namely electromyogram (EMG), electrooculogram (EOG) and electroencephalogram (EEG). These signals are detected on the user's forehead

[1]http://www.ocztechnology.com/products/oczperipherals/nia_neural-impulse-actuator

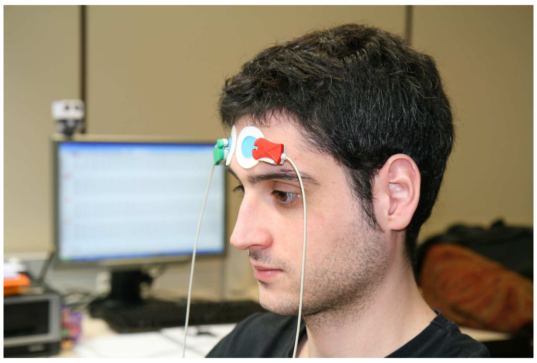


Figure 13.1: User wearing a pair of electrodes on the forehead, corresponding to an EMG system developed at the Public University of Navarra.

by three dry electrodes mounted on a headband. The software provided with the system analyzes the biosignals and decodes them in eleven frequencies. Figure 13.2 shows an example of the EMG signal recorded with the NIA. While the muscle is relaxed, the detected signal has a low amplitude. When the muscle is contracted, the electric potential increases, resulting in a peak in the measured signal.

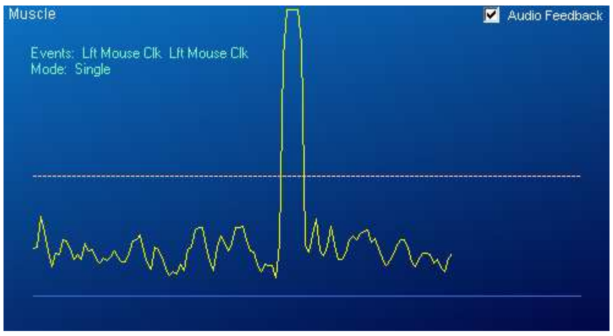


Figure 13.2: Muscle activity recorded with the NIA system, showing an activation produced by a muscular contraction and detected when the measured electric potential goes above the threshold.

A threshold can be established to classify a given sample into a relaxed state or a contracted state. If the measured electric potential takes a value above the threshold, the sample will be classified as a contracted muscle state. In Figure 13.2, the muscle is contracted and the value of the electric potential goes above the threshold, producing an activation.

EMG signals can be used in HCI to issue click events by contracting specific muscles. Being the user group in need for alternative input modalities, severely disabled users can benefit from a fast and reliable selection technique. Even individuals in late stages of ALS retain some muscle activity that can enable them to use an EMG switch [36].

Using the Cyberlink[2], Nelson et al. [57] found indications that EMG clicking could be up to 20% faster than finger-button clicking in a pure reaction task. A few studies have evaluated the potential of combining gaze pointing and EMG clicking. Partala et al. [60] explored the benefit of a combination of gaze pointing and facial-muscle EMG clicking compared to mouse input in target-acquisition tasks. Task completion times were found to be shorter for the alternative input technique for long distances (above 100 pixels) and after removing the trials in which selection occurred outside the target. A very high error rate (34%) was observed for the gaze-EMG combination.

Surakka et al. [76] extended the previous study with a more in-depth Fitts' Law analysis, and compared the index of performance of the gaze-EMG combination to the mouse. They found the gaze-EMG combination to have a higher index of performance for long distances and error-free data. For short distances the mouse was more effective. The data did not show any speed advantage of gaze-EMG over the mouse, but the authors hypothesized to find a difference for long distances.

13.2 Performance Evaluation of Gaze Pointing and EMG Clicking

An experiment was conducted to study the potential of combining gaze pointing with facial-muscle activation and compare its performance with mouse pointing and mouse-button activation in target-acquisition tasks.

13.2.1 Participants

A total of five male volunteers, ranging from 25 to 30 years old, participated in this study. All five of them were regular mouse users, four had previous experience with gaze tracking and two had tried an EMG system before.

13.2.2 Apparatus

Figure 13.3 shows all the equipment used in this study. Targets were presented by software programmed in C# that ran at 60 Hz on a Pentium IV. The display was a 17" monitor with a resolution of 1024×768 pixels. The optic mouse (Acer) was set to an intermediate speed. Muscle activity was measured with a Cyberlink system [57][2]. Participants wore a headband on the forehead that measured electrical signals from

[2]The Cyberlink system is the predecessor to the Neural Impulse Actuator system introduced above. In most respects, both systems are identical except for some physical design improvements on the NIA.

facial muscles. The Cyberlink sent off a click command to the computer via the RS 232 channel each time participants slightly frowned or tightened their jaw.

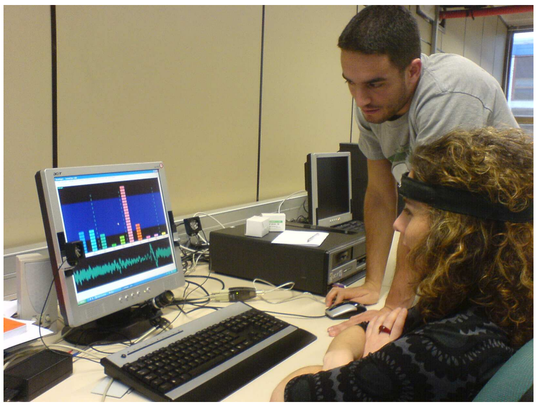


Figure 13.3: Experimental setup, showing the gaze tracking system, the mouse and the Cyberlink headband.

A gaze tracking system developed at the Public University of Navarra was used as the pointing device. It has an infrared light source on each side of the screen and uses an interpolation-based technique to estimate gaze. The measured accuracy is $<0.5°$ (around 16 pixels in our configuration), sampling at 30 Hz.

13.2.3 Design and Procedure

The experiment was conducted using a $2\times2\times3\times3$ within-subjects full factorial design. Factors were *pointing method* (mouse or gaze), *selection method* (mouse button or EMG switch), *target width* (100, 125 or 150 pixels), and *distance to the target* (200, 250 or 300 pixels). The indexes of difficulty, calculated using Equation 5.3, were between 1.2 and 2 bits. In each trial, completion time and unsuccessful activations (i.e., clicks outside the target) were measured. Participants also completed a questionnaire rating the speed, accuracy, ease of use, and fatigue perceived in association with each input combination.

Each participant completed a block of trials for each input combination. The order of these four blocks was chosen to counterbalance the effects of order and practice across participants. At the beginning of each trial, participants had to point at a crosshair shown in the center of the workspace (i.e., home position). Shortly after, a circular target appeared in the workspace. Participants were instructed to move the cursor and click on the target as quickly and accurately as possible. A successful click (i.e., inside the target) ended a trial. Then, the target disappeared and the participant moved the cursor back to the home position to release a new target.

In each block, 16 data points were collected for each width and distance combination, one for each of 16 possible directions of movement, as specified in the ISO 9241 - Part 9 standard (see Chapter 5). The resulting 144 trials (16 directions $\times$ 3 widths $\times$ 3 distances) were presented in a random order in each block. Participants could take breaks at any time by not moving the cursor back to the home position after the end of a trial. After each block, participants rated the input combination used during the block. At the end of the fourth block, they compared the four input combinations to each other.

13.2.4 Results and Discussion

Data analysis was performed using three 2×2×3×3 within-subjects full factorial ANOVAs, with pointing method (mouse or gaze), selection method (mouse button or EMG switch), target size (100, 125 or 150 pixels), and target distance (200, 250 or 300 pixels) as the independent variables. Completion time, throughput, and error rate were analyzed as the dependent variables. Our task required a successful activation to complete each trial. Unsuccessful activations resulted in longer completion times. To avoid the effect of unsuccessful activations on the speed measures, erroneous trials were removed from the data used for the ANOVAs of completion time and throughput. However, completion time data before and after removing erroneous trials was compared in the Fitts' Law analysis described below. Error rate was defined as the proportion of erroneous trials (i.e., with one or multiple unsuccessful activations) in each condition.

Fitts' Law Analysis

The mean completion times for each combination of target size and target distance were used to analyze how well the data fitted Fitts' Law. As the index of difficulty (*ID*) increases, Fitts' Law predicts a linear increase in completion time. Using Equation 5.4, the regression lines for the four input combinations were calculated and plotted in Figure 13.4, together with the corresponding equations. The linear fits for all four input combinations show positive slopes, indicating that a positive correlation exists between *ID* and completion time, in accordance to Fitts' Law. The gaze-EMG combination had the shallowest slope of the four input combinations.

The theoretical behavior of the four input combinations in case of perfect pointing performance was also studied. This analysis was carried out by removing erroneous trials, i.e. those trials where the selection was performed when the pointer was outside the target. In the case of mouse pointing 21.45% of the trials were removed; 23.05% in the case of gaze pointing (see below for an analysis on error rate). The regression lines and corresponding equations are shown in Figure 13.5. When looking at these error-free data, input combinations in which the mouse was used for pointing present positive slopes, whereas combinations in which gaze was used for pointing present a virtually flat slope. This is in accordance with the findings by Partala et al. [60]. This

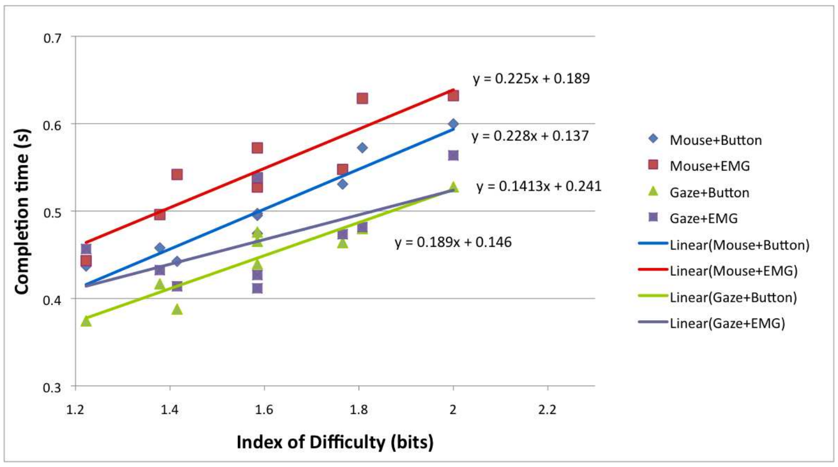


Figure 13.4: Completion time as a function of the index of difficulty for the four input combinations.

shallow slope suggests that, in cases where the accuracy is high enough to acquire the targets without errors, an increase in index of difficulty does not affect completion time. Since Fitts' Law predicts such correlation, a reformulation of the law might be necessary for gaze pointing.

Overall mean throughput was 3.03 bits/s. Mean throughput was higher using gaze pointing (M = 3.31 bits/s, SD = 0.78) than using mouse pointing (M = 2.76 bits/s, SD = 0.65), F(1, 4) = 7.98, $p < 0.05$. Mean throughput was not significantly different between using mouse selection (M = 3.10 bits/s, SD = 0.69) and EMG selection (M = 2.97 bits/s, SD = 0.84), F(1, 4) = 1.52, $p > 0.05$. Figure 2 shows the mean throughput obtained for each input combination.

Throughput values are lower than in previous studies presented in the literature. Specifically, it is common for mouse to obtain a throughput value of around 5 bits/s. The low value obtained in the present study is due to the high error rate (see explanation for this high error rate below). Gaze pointing showed a better performance than mouse pointing, with a throughput value similar to the value obtained in Chapter 6 and in other studies such as [95].

Overall mean completion time was 393 ms. Mean completion time was lower using gaze pointing (M = 354 ms, SD = 46) than using mouse pointing (M = 433 ms, SD = 43), F(1, 4) = 29.91, $p < 0.05$. Mean completion time for mouse selection (M = 394 ms, SD = 57) and EMG selection (M = 393 ms, SD = 62) were not significantly different, F(1, 4) = 0.004, $p > 0.05$.

Overall mean error rate was 22.25%. Neither pointing method, F(1, 4) = 0.64, $p > 0.05$, nor selection method, F(1, 4) = 1.35, $p > 0.05$, had a significant effect on error rate. Mean error rate was 21.45% for mouse pointing and 23.05% for gaze pointing.

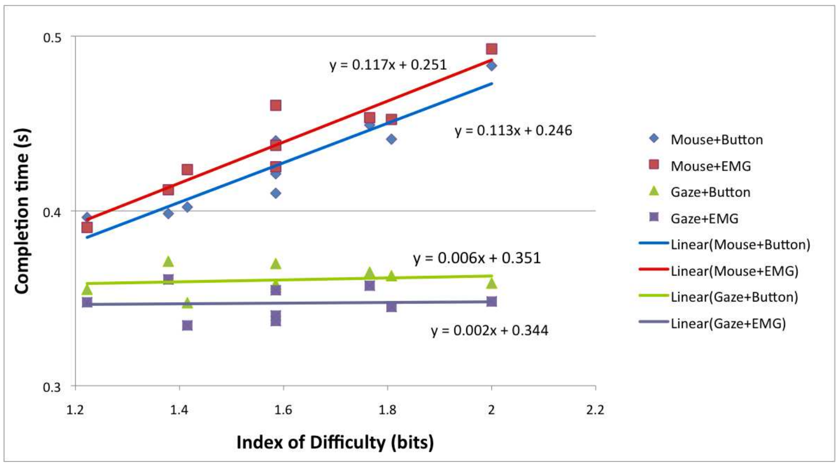


Figure 13.5: Completion time as a function of the index of difficulty for the four input combinations after removing erroneous trials.

In the case of selection method, mean error rate was 20.69% for mouse button click and 23.82% for EMG switch.

Figure 13.7 shows the mean completion time vs the error rate for each input combination. Combinations that used gaze pointing are faster, while error rate is similar for all combinations.

The error rate was notably high for mouse pointing. Usual error rates for mouse are around 5%. In the present study, the objective was to find a difference in speed between the gaze-EMG combination and the mouse. In order to find that difference, participants were asked to perform as fast as they could. They traded speed for accuracy: the completion times were low compared to the values in previous studies ([95], [60], [76]), but the error rates were significantly higher. The loss in accuracy did not compensate for the gain in speed, and the throughput decreased.

Even though EMG selection was not faster than mouse selection as Nelson found in his study [57], the combination of gaze pointing and EMG selection performed at least as well as the mouse while maintaining the hands-free advantage. Surakka et al. [76] were not able to find a speed advantage of the gaze-EMG combination over the mouse and they suggested that this advantage would become apparent if longer distances were used. In the present study, a speed advantage was found even though shorter distances were used. Longer distances could be expected to provide an even bigger advantage of gaze over mouse.

Subjective Ratings

Participants perceived gaze pointing as faster, but less accurate, than mouse pointing. Most of them reported that the gaze-EMG combination was natural to use, but needed

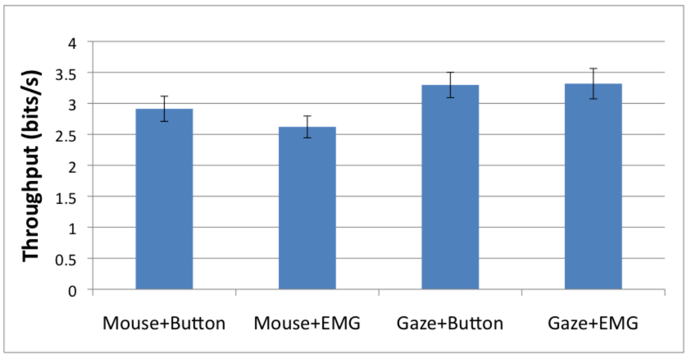


Figure 13.6: Mean throughput for each input combination. Error bars show the standard error of the mean.

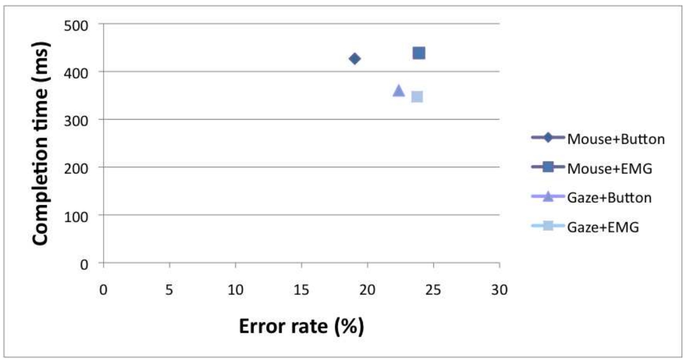


Figure 13.7: Completion vs error rate for each input combination.

more practice to use it to its full potential. Gaze was also rated as fatiguing, in part because of the need of keeping the head rather still for long periods of time. One participant even suggested using a chinrest. However, the external validity of the experiment would have significantly decreased by using a chinrest.

13.3 Dealing with False Activations in a Gaze-EMG Combination

In an everyday situation, users typically perform two kinds of eye movement tasks: visual tasks such as reading or browsing, and selection tasks to activate a menu item or a link. During the visual task no activations should occur, since they could lead to an undesired selection. An EMG switch, due to noise in the system and in the environment, is more likely to perform false activations than a conventional key or mouse-button activation.

The study presented in the previous section, along with most previous investigations on the combination of gaze pointing and EMG clicking, focuses on investigating the speed advantage of the gaze-EMG combination over the mouse in an ideal desktop scenario. However, noisy environments, such as walking or using a wheelchair, can potentially introduce a high number of undesired muscle activations, and therefore a high number of click events will take place. In these scenarios, the robustness against false activations rather than speed should be emphasized.

In this section, a system that tightly integrates gaze information and muscle information is described. The hypothesis is that voluntary activations are only produced during a fixation. Therefore, it is possible to decrease the number of undesired activations by introducing a requirement for a fixation in the gaze tracker before issuing a click event when a muscle activation is detected by the EMG switch. No EMG activation can occur when the gaze is not fixated. In order to achieve this, the EMG system needs to have access to information on fixations and saccades, which must be provided by the gaze tracking system.

13.3.1 Integrated Multimodal Interface

The system proposed is composed of two subsystems: the gaze tracker and the EMG switch. Both have been developed at the Public University of Navarra. The gaze tracker, described in the previous section, is used to control the mouse cursor, while the EMG switch is used to perform selections. The EMG system uses three surface electrodes, two placed on the user's forehead and one on the wrist (see Figures 13.1 and Figure 13.9). The electrodes are connected to a bio-signal amplifier. The detection of muscle activity is performed in time domain by an AGLR Ramp algorithm (see [1] for further details).

Two versions of the integrated gaze tracker-EMG system have been implemented and compared to each other: the System With No Communication (SWNC) and the System With Communication (SWC). In the SWNC the gaze tracker and the EMG switch work independently of each other: whenever the EMG detects a muscle activation, a click event is issued. This is the normal implementation of a gaze tracker-EMG combination. In the SWC, on the other hand, the EMG system uses information about the eye movements. When the gaze tracker system detects a fixation, the EMG is notified. If a muscle activation occurs during a no-fixation state, the activation is rejected; if it occurs during a fixation, the EMG sends a click event to the operating system. *Activation time* is defined as the minimum duration a fixation needs to be maintained, and is equivalent to the minimum duration time of the area-based fixation detection algorithms presented presented in Chapter 11.

Figure 13.8 illustrates the behavior of each implementation. Three EMG activations take place: the first one occurs when the user is not performing a fixation. In the second one the user is fixating on a point, but the time elapsed since the beginning of the fixation is lower than the activation time. In both cases there is no fixation detected by the gaze tracking system system. The third activation of the EMG occurs during a fixation, and the gaze tracking system raises the "fixation detected" flag, notifying the EMG system that a fixation is taking place. The SWNC does not take into account information about fixations, and therefore issues a click event on the three EMG activations. The SWC, on the other hand, checks the fixation state on every EMG activation detected. Since the first and second activations occur during a no-fixation state, both are rejected.

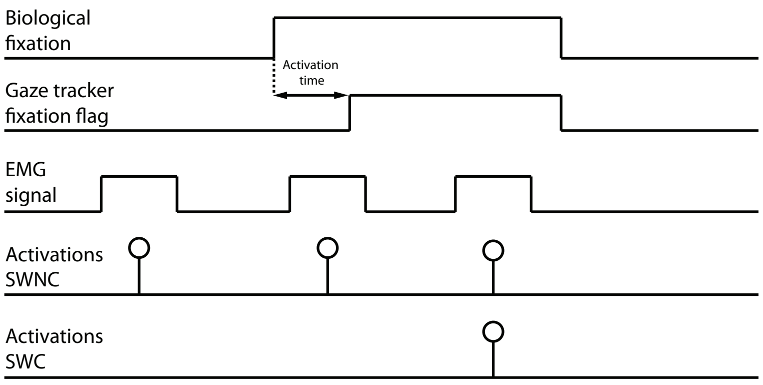


Figure 13.8: Time sequence of three EMG activations, two occurring during a no-fixation state and another occurring during a fixation state.

Undesired muscle activations can occur due to involuntary user actions or due to system noise. In both cases, these activations can be modeled by introducing noise to the system.

13.3.2 Participants and Apparatus

A total of 10 volunteers, ranging from 24 to 41 years old, participated in the experiment. They were randomly divided into two groups: one group tested the SWNC while the other tested the SWC. Two participants had previous experience with gaze tracking and EMG activation; each of them was assigned to a different group.

A 17" monitor with a resolution of 1280×1024 was used to present the target-acquisition task. The gaze tracker and EMG presented above were used for pointing and performing selections. The experimental setup is shown in Figure 13.9.

13.3.3 Design and Procedure

The experiment was conducted using a 2×3×2 mixed design. There was one between-subjects factor, *communication* (SWNC or SWC) and two within-subjects factors, *noise* (no noise NN, low noise LN or high noise HN) and *activation time* (200 ms or 400 ms).

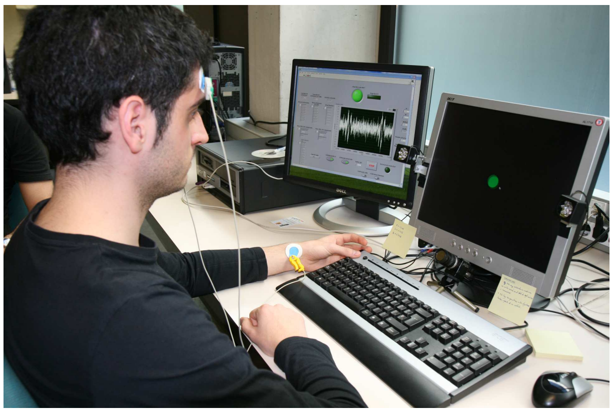


Figure 13.9: Experimental setup. The main screen is displaying the target-acquisition task, while the secondary monitor is displaying the fixation state and the EMG signal.

The task that participants carried out consisted on a target-acquisition task based on the ISO 9241 - Part 9 standard (see Chapter 5). The size of the targets was fixed to 150 pixels in diameter. Prior to starting the experiment, participants calibrated both the gaze tracker and the EMG system and ran a warm-up trial to become acquainted with the multimodal interface. Each participant completed a block of 64 trials for each combination of noise level and activation time. The order of the 6 blocks was counterbalanced to neutralize learning effects.

In order to simulate a situation where the user performs both a visual and a selection task, a moving object was added between targets. It appeared for a random time of 2 to 4 s following a lemniscate of Bernoulli curve with constant speed, and users performed a smooth pursuit movement to track it. Once the random time elapsed, a new target that the user had to select was displayed. Participants were instructed not to perform any activations while the moving object was on the screen. A noisy environment in which undesired muscle activations occur was simulated by introducing random activations to the EMG system. A homogeneous Poisson process was used to model the noisy activations. In the low noise LN condition, the average time between noisy activations was 8 seconds, while in the high noise HN condition it was 4 s.

Four metrics were measured in each trial: completion time (i.e., time elapsed since the target appears until the user selects it), error rate (i.e., the proportion of selections performed outside the target out of the total number of selections), involuntary selections (i.e., the proportion of targets selected by a noisy activation and not by an EMG activation out of the total number of targets), and clicks produced by noise (i.e., the proportion of noisy activations that issued a click event out of the total number of activations).

13.3.4 Results

The analysis was carried out using four 2×3×2 mixed-design ANOVAs, with communication being a between-subjects factor, and noise and activation time being within-subjects factors. Completion time, error rate, involuntary selections and clicks by noise were analyzed as the dependent variables. All data were included.

The type of communication between gaze tracker and EMG had a significant effect on completion time, F(1, 8) = 33.15, $p < 0.05$, and on error rate, F(1, 8) = 11.42, p < 0.05. When communication between systems existed, completion times were significantly higher and error rates were significantly lower than when no communication existed. Noise had no effect on either completion time, F(2, 16) = 1.01, $p > 0.05$, or error rate, F(2, 16) = 1.49, $p > 0.05$. Figure 13.10 shows average completion times and error rates for each combination of communication and noise.

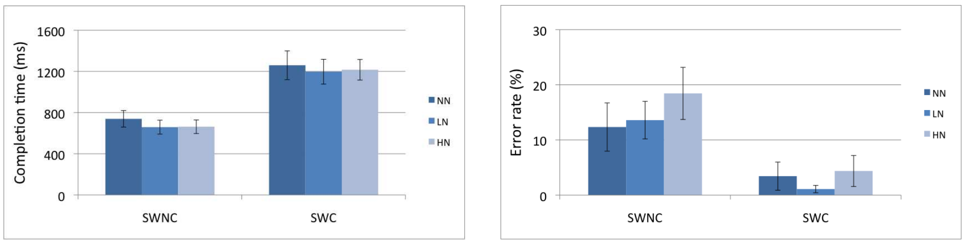


Figure 13.10: Average completion time and error rate for each combination of communication type and noise level. Error bars show the standard error of the mean.

The type of communication had a significant effect on the number of involuntary selections, F(1, 8) = 25.36, $p < 0.05$, and on the number of clicks produced by noise, F(1, 8) = 412.12, $p < 0.05$. Both were significantly reduced when communication between gaze tracker and EMG was used. Noise also had a significant effect on both involuntary selections, F(2, 16) = 55.26, $p < 0.05$, and clicks produced by noise, F(2, 16) = 233.51, $p < 0.05$. Both increased together with noise. Figure 13.11 shows average involuntary selections and noisy clicks per target for each combination of communication and noise.

The effect of activation time was studied for the SWC (it is irrelevant for the SWNC). An activation time of 400 ms did not result in significantly longer completion times (F(1, 4) = 4.165, $p > 0.05$) than an activation time of 200 ms, nor in lower error rates (F(1, 4) = 0.580, $p > 0.05$). Although no significant difference was found, the average completion times were 254 ms faster when using an activation time of 200 ms (see Figure 13.12).

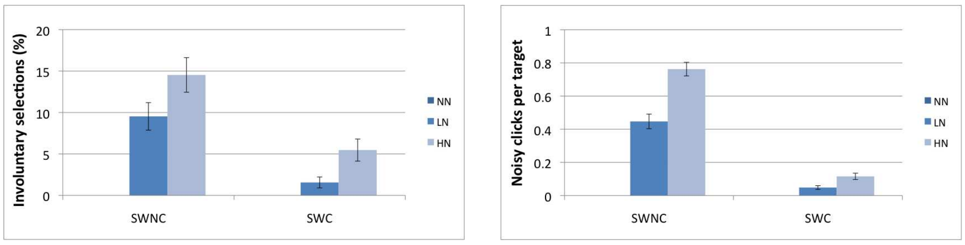


Figure 13.11: Average rate of involuntary selections and clicks produced by noise for each combination of communication type and noise level. Error bars show the standard error of the mean.

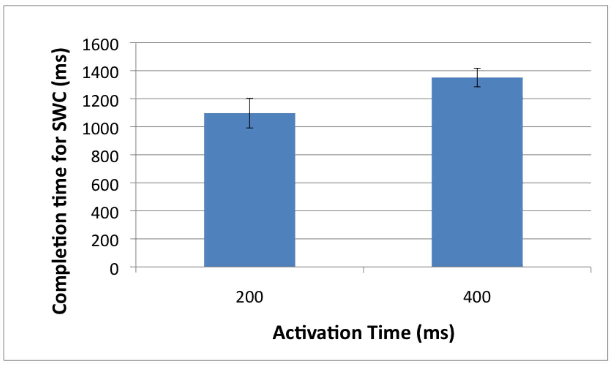


Figure 13.12: Average completion time for SWC for 200 and 400 ms activation time.

13.4 Discussion

The first study presented in this chapter indicates that the hands-free input gaze-EMG combination can perform at least as well as the mouse, while keeping the user's hands free for other tasks. In spite of the fact that the distances used were on average shorter than those used by Surakka et al. [76], a speed advantage of the gaze-EMG combination over the mouse was found in the present study. Nevertheless, contrary to the findings by Nelson et al. [57], EMG selections were not faster than mouse-button selections. This difference between both results may be partially attributed to the difference between the pure reaction-time task used by Nelson et al. and the target-acquisition task used in this investigation.

Although subjective ratings were positive when the gaze tracking was particularly accurate, participants reported certain discomfort when calibration offsets appeared and when EMG clicks were missed by the system. Furthermore, participants would often make natural movements of their head and face that led to false activations of the EMG system and resulted in undesired and erroneous selections.

The second study aimed to solve the latter issue by integrating gaze information and EMG information. The requirement that for every facial muscle activation there must be a fixation before issuing a click event decreased the error rate, but it also increased the time required to select the targets. Using this technique, most noisy activations that occur when the user is performing a visual task such as reading or browsing will not become a click event due to the lack of a fixation state[3]. In noisy environments where false activations are likely to occur, a tight integration of gaze tracker and EMG switch might be beneficial to increase the robustness and reliability of the system. This can be the case when the user is driving a wheelchair or has involuntary facial muscle movements.

13.5 Further Reading

Mateo et al. [53] and San Agustin et al. [65] provide further results regarding the first study presented in this chapter. The second study was conducted in collaboration with the Gaze Interaction Group at the Public University of Navarra and published in [1]. The author of this thesis collaborated in designing the experimental procedure and the target-acquisition application, in writing the article, and carried out the statistical analysis of the results.

[3]It must be noted that a reading task involves saccades and short fixations, and therefore the smooth pursuit task used in this experiment might not provide a good simulation of a reading task.

Part V

Conclusion and Future Work

Chapter 14

Conclusion

This chapter summarizes the work presented in this thesis and suggests possible lines of future research.

14.1 Contributions of this Thesis

The work presented in this thesis covers different aspects regarding the use of low-cost and off-the-shelf components for gaze-based interaction. A summary of the most important conclusions is given now.

Part II describes the ITU Gaze Tracker, a gaze tracking system built from off-the-shelf hardware components, and presents an evaluation of the performance of a webcam-based version of the system. Contrary to previous low-cost systems described in the literature that require hardware modifications, the system proposed in this thesis is easy to build and use. The main contributions of this part are:

- The ITU Gaze Tracker has been developed as part of this thesis and has been released as open-source software. The system is built from low-cost and off-the-shelf components, and aims to provide an inexpensive entry point for users and researchers that want to try gaze interaction. A description of the system is given in Chapter 4.
- The performance of the ITU Gaze Tracker has been evaluated and compared with two commercial systems in target-acquisition and eye-typing tasks. A similar performance is obtained for the three systems under study, demonstrating that it is possible to build a system from low-cost components that has a comparable performance to commercial gaze trackers. This evaluation is presented in Section 6.1.
- A limitation of the proposed system is the lack of head-pose invariance. The system, however, can be successfully used by people with severe motor-skill

disabilities who can only move their eyes. A pilot study with a person with ALS is described in Section 6.1, and shows the high flexibility of the system.

- The possibilities of using the ITU Gaze Tracker to interact with a head-mounted display are investigated in Section 6.2. The low performance and high error rate indicate that mobile interaction with the current design is challenging.
- As of September 13, the software has been downloaded more than 1700 times from http://www.gazegroup.org, which indicates a high interest in having a low-cost gaze tracking system that is easy to use. The community is reporting different projects where the ITU Gaze Tracker is employed, and collaborating in finding bugs and developing the next version of the software.

The use of low-cost and off-the-shelf components often increases the flexibility of the system, since the user might be able to place the camera and light sources in the most convenient location. However, the lack of control over the hardware setup might decrease the robustness of the system. A novel geometry-based gaze estimation method based on homographic mappings is proposed in Part III of this thesis. The method does not require geometry calibration, and user calibration can be done using as few as 4 points. An analysis of the effect that different assumptions made by the model have on the accuracy has been carried out using simulated data. The distance between cornea plane and pupil plane is found to constitute the main source of error in the estimation of gaze. Furthermore, the method is compared to two other geometry-based, uncalibrated methods based on the cross-ratio property of projective space. A theoretical accuracy of $1°$ is obtained for the homography method with 4 calibration points.

The main properties of the homography method proposed in this work are:

- Uncalibrated method. The camera and the light sources can be placed in any location, thereby increasing the flexibility of the setup. Compared to other geometry-based, uncalibrated methods based on the cross-ratio, the number of light sources is reduced from five to four.
- Average accuracy of $1.29°$ of visual angle using real data when four points are used for calibration. Commercial systems based on calibrated methods usually have an accuracy of $0.5°$.
- Low number of calibration points. Only four points are required to calculate the homography that maps the pupil to the point of regard. Increasing the number of points to 25 improves the accuracy by 20% in the tests conducted using real data.
- Low effect of head movements on the accuracy. The average estimation error using four calibration points slightly decreases from $1.29°$ to $1.27°$ when the user moves from calibration position and performs natural head movements.
- Off-the-shelf hardware can be employed. Although a high-end camera has been used in the tests conducted for this thesis, it is possible to use common cameras such as video cameras.

The homography method does not model the non-linearities that occur when the eye rotates. The model has been extended with an interpolation method that seeks to compensate the effect of these non-linearities by applying a third-order polynomial regression on the screen coordinates. A theoretical error of 0.03° is obtained with 16 or more calibration points. Using real data, the error is 0.91° for 16 calibration points. Although interpolation methods are greatly affected by head movements, the combination of homography and interpolation is found to be more robust to changes in head pose than a pure interpolation technique. The error increases slightly to 0.99°.

Part IV investigates ways to enhance gaze interaction and provide a more natural way of using gaze pointing. Two approaches are taken: (1) smoothing the cursor position accordingly to the eye movements to reduce noise, and (2) combine gaze pointing and facial-muscle activation to improve selection times. A novel algorithm that can detect fixations, saccades and smooth pursuit movements is presented and evaluated. The algorithm is based on the eye velocity and the eye movement pattern, and can correctly classify fixations in 77% of the cases, and smooth pursuits in 44%. Target velocity has a high impact on the ratio of properly detected smooth pursuits, which decreases as target velocity increases. The delay introduced by reaction time and by the gaze tracking system has also an impact on the ratio of correctly detected movements. Furthermore, during smooth pursuit movements the cursor lags behind the target. A Kalman filter is used for two purposes, smoothing and prediction. The Kalman filter allows to smooth the cursor coordinates without introducing lag. More importantly, it is possible to predict the next sample and update the cusor position with the prediction, effectively reducing the lag between target and cursor position. The average error between predicted and observed value during smooth pursuit movements is 0.57°, significantly better than the results obtained by Komogortsev et al. in [41] of around 3°.

One of the unsolved challenges in gaze-based interaction is the Midas Touch problem. In this thesis I have explored the potential of a multimodal input technique based on the combination of gaze pointing and EMG clicking. Although EMG clicking was expected to have a performance improvement over mouse clicking, no real advantage has been seen. However, the combination of gaze and EMG was faster than the mouse, thus providing a hands-free input technique that can be used in scenarios where the hands are used for other tasks. This technique can be of special importance for people with locked-in syndrome. Furthermore, the participants were not experienced with the gaze and EMG combination, and therefore an improvement in performance could be expected as users become more proficient with the system. Ways to deal with the noise in the EMG selections have also been investigated. On the assumption that we only select objects that we are fixating on, a tightly integrated gaze - EMG combination has been explored, in which only activations that occur during a fixation issue a click event. This allows to reduce the number of involuntary selections produced by noise.

14.2 Future Work

New research questions arose while addressing some of the challenges introduced by using low-cost components in gaze interaction. Some of them are:

- A combination of head tracking and gaze tracking can be explored in order to increase the robustness to head movements in the head-mounted version of the ITU Gaze Tracker. This would require one camera tracking the head and another tracking the eye. The built-in webcam included in many laptop computers nowadays could be employed for head-pose tracking.
- Although the performance obtained with the ITU Gaze Tracker is similar to commercial systems, the experiments have been conducted with non-disabled participants. Further and more thorough experiments are to be conducted with people with ALS in order to fully explore the potentials of the system and investigate and address its limitations.
- Outdoor mobile tracking has not been explored. Since the eye tracking methods are based on infrared light, this can lead to tracking issues in outdoor environments. Possible ways to deal with these are: (1) implement methods to robustly track and identify each corneal reflection, so that noisy reflections can be eliminated; and (2) investigate and implement tracking methods based on natural illumination (e.g. [25]).
- The homography method requires four known features on the image. Although four light sources have been employed, it is possible to use the reflection of the screen on the eye and find the corners of such reflection, thus providing head-pose tolerant, infrared-free gaze tracking.
- The evaluation of the gaze estimation methods has been carried out using a high-quality camera. Analyzing the accuracy employing a videocamera can give an indication of the difference between using off-the-shelf and non-off-the-shelf hardware.
- The homography method for gaze estimation is extended with an interpolation technique that compensates the non-linearities, thereby improving the accuracy over the pure homography method. Nevertheless, other methods that account for this error could be considered.
- The eye movement detection algorithm proposed in this thesis is based on thresholds that need to be estimated empirically depending on the characteristics of the system, and must be chosen by the user. An automatic calibration of these parameters would help simplify setting up the correct values.
- Multimodal input is explored using an EMG switch to perform selections. Other multimodal combinations could also be explored, like tongue activations [75] or brain-computer interfaces (BCIs) [79].

Appendix A

Kalman Filter

The Kalman filter is a recursive filter that allows to reconstruct and predict the state of a dynamic system from a series of noisy measurements. It works in a discretized time domain, and aims to minimize the error between the predicted system state in the current time step without using the current observation, and the updated system state when the current observation is taken into account. Being recursive, only the previous estimate of the system state and the current observation are required to compute an estimate for the system state in the next time step.

The system state in time step k is denoted as x_k, and is computed as follows:

$$x_k = A_k x_{k-1} + B_k u_k + w_k \tag{A.1}$$

where A_k is the state transition matrix that relates the system state in the previous time step *k-1* with the system state in the current time step *k*. u_k is a control vector and is governed by B_k, the control input matrix. w_k is the system noise, and is assumed to follow a multivariate Gaussian distribution with zero mean and covariance Q_k, i.e. $w_k \sim N(0, Q_k)$.

Not all variables in the system state might be observable. The true measurement of the system state x_k at time step k is denoted as z_k and given by the following equation:

$$z_k = H_k x_k + v_k \tag{A.2}$$

where H_k is the observation model that relates the system state with the actual observation, and v_k is the observation noise, which is assumed to follow a multivariate Gaussian distribution with zero mean and covariance R_k, i.e. $v_k \sim N(0, R_k)$. The system noise w_k and the observation noise v_k are assumed to be mutually independent.

There are two stages in the Kalman filter: *predict* and *update*.

Predict The predict stage uses the estimate of the system state in the previous time step *k-1* to estimate the state in the current time step *k*. Therefore, it is an *a priori* estimate, because it does not take into account the actual measurement.

$$\hat{x}_k^- = A_k \hat{x}_{k-1} + B_k u_{k-1} \tag{A.3}$$

where $\hat{x}_k^-$ denotes the *a priori* prediction of the system state, i.e. the prediction without using the measurement z_k.

We then predict the error covariance as follows:

$$P_k^- = A_k P_{k-1} A_k^T + Q_k \tag{A.4}$$

This stage can also be used to predict the system state in future time steps.

Update The update stage incorporates the measurement into the *a priori* estimate to obtain an improved *a posteriori* estimate. The Kalman gain K_k is computed first as expressed in Equation A.5.

$$K_k = P_k^- H_k^T \left(H_k P_k^- H_k^T + R_k \right)^{-1} \tag{A.5}$$

The system state estimate is updated taking into account the current measurement z_k:

$$\hat{x}_k = \hat{x}_k^- K_k \left(z_k - H_k \hat{x}_k^- \right) \tag{A.6}$$

Finally, the error covariance matrix P_k is updated:

$$P_k = \left(I - K_k H_k \right) P_k^- \tag{A.7}$$

It must be noted that A, B and C matrices are known. The covariance matrices of system noise and measurement noise, Q and R, need to be computed beforehand. Measuring the measurement covariance is usually possible, since we need to be able to measure the process anyway. Computing the system noise covariance can be more difficult, and often the value needs to be tuned before a good performance of the filter is achieved.

Bibliography

[1] Mikel Ariz, Javier Navallas, Arantxa Villanueva, Javier San Agustin, Rafael Cabeza, and Martin Tall. Optimizing the interoperability between a VOG and an EMG system. In *The 5th Conference on Communication by Gaze Interaction*, Lyngby, Denmark, 2009.

[2] Jason S. Babcock and Jeff B. Pelz. Building a lightweight eyetracking headgear. In *Proceedings of the 2004 symposium on Eye tracking research & applications*, pages 109–114, San Antonio, Texas, 2004. ACM.

[3] Richard Bates, M. Donegan, H. Istance, J. Hansen, and K.-J. Räihä. Introducing COGAIN: communication by gaze interaction. *Universal Access in the Information Society*, 6(2):159–166, 2007.

[4] Richard Bates and Howell Istance. Zooming interfaces!: enhancing the performance of eye controlled pointing devices. In *Proceedings of the fifth international ACM conference on Assistive technologies*, pages 119–126, Edinburgh, Scotland, 2002. ACM.

[5] D. Beymer and M. Flickner. Eye gaze tracking using an active stereo head. In *Computer Vision and Pattern Recognition, 2003. Proceedings. 2003 IEEE Computer Society Conference on*, volume 2, pages II–451–8 vol.2, 2003.

[6] Martin Böhme, Michael Dorr, Mathis Graw, Thomas Martinetz, and Erhardt Barth. A software framework for simulating eye trackers. In *Proceedings of the 2008 symposium on Eye tracking research & applications*, pages 251–258, Savannah, Georgia, 2008. ACM.

[7] Roger H. S. Carpenter. *Movements of the Eyes*. Pion Ltd, 2nd edition, 1977.

[8] Juan J. Cerrolaza, Arantxa Villanueva, and Rafael Cabeza. Taxonomic study of polynomial regressions applied to the calibration of video-oculographic systems. In *Proceedings of the 2008 symposium on Eye tracking research & applications*, pages 259–266, Savannah, Georgia, 2008. ACM.

[9] T. F. Cootes, C. J. Taylor, D. H. Cooper, and J. Graham. Active shape models-their training and application. *Comput. Vis. Image Underst.*, 61(1):38–59, 1995.

[10] F.L. Coutinho and C.H. Morimoto. Free head motion eye gaze tracking using a single camera and multiple light sources. In *Computer Graphics and Image Processing, 2006. SIBGRAPI '06. 19th Brazilian Symposium on*, pages 171–178, 2006.

[11] Heiko Drewes and Albrecht Schmidt. Interacting with the computer using gaze gestures. In *Human-Computer Interaction – INTERACT 2007*, pages 475–488. 2009.

[12] Andrew T Duchowski. A breadth-first survey of eye-tracking applications. *Behavior Research Methods, Instruments, & Computers*, 34:455–470, November 2002.

[13] Andrew T. Duchowski. *Eye Tracking Methodology: Theory and Practice.* Springer, 1 edition, 2003.

[14] Andrew T Duchowski, Nathan Cournia, and Hunter Murphy. Gaze-Contingent displays: A review. *Cyberpsychology & Behavior: The Impact of the Internet, Multimedia and Virtual Reality on Behavior and Society*, 7(6):621–34, December 2004. PMID: 15687796.

[15] Andrew T. Duchowski, Eric Medlin, Nathan Cournia, Anand Gramopadhye, Brian Melloy, and Santosh Nair. 3D eye movement analysis for VR visual inspection training. In *Proceedings of the 2002 symposium on Eye tracking research & applications*, pages 103–110, New Orleans, Louisiana, 2002. ACM.

[16] Andrew T. Duchowski, Vinay Shivashankaraiah, Tim Rawls, Anand K. Gramopadhye, Brian J. Melloy, and Barbara Kanki. Binocular eye tracking in virtual reality for inspection training. In *Proceedings of the 2000 symposium on Eye tracking research & applications*, pages 89–96, Palm Beach Gardens, Florida, United States, 2000. ACM.

[17] Y. Ebisawa. Improved video-based eye-gaze detection method. *Instrumentation and Measurement, IEEE Transactions on*, 47(4):948–955, 1998.

[18] Y. Ebisawa, Y. Ebisawa, S. i. Satoh, and S. i. Satoh. Effectiveness of pupil area detection technique using two light sources and image difference method. In *Engineering in Medicine and Biology Society, 1993. Proceedings of the 15th Annual International Conference of the IEEE*, pages 1268–1269, 1993.

[19] Paul M. Fitts. The information capacity of the human motor system in controlling the amplitude of movement. *Journal of Experimental Psychology: General*, 121(3):262–269, 1992.

[20] Richard Godijn and Jan Theeuwes. The relationship between exogenous and endogenous saccades and attention. In *The Mind's Eye: Cognitive and Applied Aspects of Eye Movement Research*, pages 3–26. Elsevier, Amsterdam, 2003.

[21] E.D. Guestrin and M. Eizenman. General theory of remote gaze estimation using the pupil center and corneal reflections. *Biomedical Engineering, IEEE Transactions on*, 53(6):1124–1133, 2006.

[22] Elias D. Guestrin, Moshe Eizenman, Jeffrey J. Kang, and Erez Eizenman. Analysis of subject-dependent point-of-gaze estimation bias in the cross-ratios method. In *Proceedings of the 2008 symposium on Eye tracking research & applications*, pages 237–244, Savannah, Georgia, 2008. ACM.

[23] Dan Witzner Hansen. *Committing eye tracking*. PhD thesis, IT University of Copenhagen, 2003.

[24] Dan Witzner Hansen and Qiang Ji. In the eye of the beholder: A survey of models for eyes and gaze. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 99(1), 2009.

[25] Dan Witzner Hansen, David J. C. MacKay, John Paulin Hansen, and Mads Nielsen. Eye tracking off the shelf. In *Proceedings of the 2004 symposium on Eye tracking research & applications*, pages 58–58, San Antonio, Texas, 2004. ACM.

[26] Dan Witzner Hansen and Arthur E.C. Pece. Eye tracking in the wild. *Computer Vision and Image Understanding*, 98(1):155–181, April 2005.

[27] Dan Witzner Hansen, Henrik H. T. Skovsgaard, John Paulin Hansen, and Emilie Møllenbach. Noise tolerant selection by gaze-controlled pan and zoom in 3D. In *Proceedings of the 2008 symposium on Eye tracking research & applications*, pages 205–212, Savannah, Georgia, 2008. ACM.

[28] John Paulin Hansen, Dan Witzner Hansen, and Anders Johansen. Bringing gaze-based interaction back to basics. In *Proceedings of Universal Access in Human-Computer Interaction (UAHCI, 2001)*, Lousiana, USA, 2001.

[29] John Paulin Hansen, Kristian Tørning, Anders Sewerin Johansen, Kenji Itoh, and Hirotaka Aoki. Gaze typing compared with input by head and hand. In *Proceedings of the 2004 symposium on Eye tracking research & applications*, pages 131–138, San Antonio, Texas, 2004. ACM.

[30] Richard Hartley and Andrew Zisserman. *Multiple view geometry in computer vision*. Cambridge University Press, 2000.

[31] T.E. Hutchinson, T.E. Hutchinson, K.P. White, K.P. White, W.N. Martin, K.C. Reichert, and L.A. Frey. Human-computer interaction using eye-gaze input. *Systems, Man and Cybernetics, IEEE Transactions on*, 19(6):1527–1534, 1989.

[32] Poika Isokoski. Text input methods for eye trackers using off-screen targets. In *Proceedings of the 2000 symposium on Eye tracking research & applications*, pages 15–21, Palm Beach Gardens, Florida, United States, 2000. ACM.

[33] Robert J. K. Jacob. The use of eye movements in human-computer interaction techniques: what you look at is what you get. *ACM Trans. Inf. Syst.*, 9(2):152–169, 1991.

[34] Robert J. K. Jacob and Keith S. Karn. Eye tracking in Human-Computer interaction and usability research: Ready to deliver the promises. In *The Mind's Eye: Cognitive and Applied Aspects of Eye Movement Research*, pages 573–605. Elsevier, Amsterdam, 2003.

[35] Birger Bergmann Jeppesen. *Er der mon bedre i himmelen?* Dafolo, 1999.

[36] Andrew T Junker and John Paulin Hansen. Gaze pointing and facial EMG clicking. In *Proceedings of The 2nd Conference on Communication by Gaze Interaction – COGAIN 2006: Gazing into the Future*, Italy, 2006.

[37] Marcel Adam Just and Patricia A. Carpenter. Eye fixations and cognitive processes. Technical report, 1975.

[38] J.J. Kang, M. Eizenman, E.D. Guestrin, and E. Eizenman. Investigation of the Cross-Ratios method for Point-of-Gaze estimation. *Biomedical Engineering, IEEE Transactions on*, 55(9):2293–2302, 2008.

[39] Do Hyong Koh, Sandeep A. Munikrishne Gowda, and Oleg V. Komogortsev. Input evaluation of an eye-gaze-guided interface: kalman filter vs. velocity threshold eye movement identification. In *Proceedings of the 1st ACM SIGCHI symposium on Engineering interactive computing systems*, pages 197–202, Pittsburgh, PA, USA, 2009. ACM.

[40] Oleg Komogortsev and Javed Khan. Kalman filtering in the design of Eye-Gaze-Guided computer interfaces. In *Human-Computer Interaction. HCI Intelligent Multimodal Interaction Environments*, pages 679–689. 2007.

[41] Oleg V. Komogortsev and Javed I. Khan. Eye movement prediction by kalman filter with integrated linear horizontal oculomotor plant mechanical model. In *Proceedings of the 2008 symposium on Eye tracking research & applications*, pages 229–236, Savannah, Georgia, 2008. ACM.

[42] M Land, N Mennie, and J Rusted. The roles of vision and eye movements in the control of activities of daily living. *Perception*, 28(11):1311–1328, 1999. PMID: 10755142.

[43] Chris Lankford. Effective eye-gaze input into windows. In *Proceedings of the 2000 symposium on Eye tracking research & applications*, pages 23–27, Palm Beach Gardens, Florida, United States, 2000. ACM.

[44] Dongheng Li, Jason Babcock, and Derrick J. Parkhurst. openEyes: a low-cost head-mounted eye-tracking solution. In *Proceedings of the 2006 symposium on Eye tracking research & applications*, pages 95–100, San Diego, California, 2006. ACM.

[45] Dongheng Li, David Winfield, and Derrick J. Parkhurst. Starburst: A hybrid algorithm for video-based eye tracking combining feature-based and model-based approaches. June 2005.

[46] A. Liu, G. Tharp, L. French, S. Lai, and L. Stark. Some of what one needs to know about using head-mounted displays to improve teleoperator performance. *Robotics and Automation, IEEE Transactions on*, 9(5):638–648, 1993.

[47] I S MacKenzie. A note on the information-theoretic basis of fitts' law. *Journal of Motor Behavior*, 21(3):323–330, September 1989.

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

[49] I. Scott MacKenzie, Tatu Kauppinen, and Miika Silfverberg. Accuracy measures for evaluating computer pointing devices. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 9–16, Seattle, Washington, United States, 2001. ACM.

[50] I. Scott MacKenzie and R. William Soukoreff. Phrase sets for evaluating text entry techniques. In *CHI '03 extended abstracts on Human factors in computing systems*, pages 754–755, Ft. Lauderdale, Florida, USA, 2003. ACM.

[51] Päivi Majaranta, Ulla-Kaija Ahola, and Oleg Špakov. Fast gaze typing with an adjustable dwell time. In *Proceedings of the 27th international conference on Human factors in computing systems*, pages 357–360, Boston, MA, USA, 2009. ACM.

[52] Päivi Majaranta and Kari-Jouko Räihä. Twenty years of eye typing: systems and design issues. In *Proceedings of the 2002 symposium on Eye tracking research & applications*, pages 15–22, New Orleans, Louisiana, 2002. ACM.

[53] Julio C. Mateo, Javier San Agustin, and John Paulin Hansen. Gaze beats mouse: hands-free selection by combining gaze and emg. In *CHI '08 extended abstracts on Human factors in computing systems*, pages 3039–3044, Florence, Italy, 2008. ACM.

[54] Emilie Mollenbach, John Paulin Hansen, Martin Lillholm, and Alastair G. Gale. Single stroke gaze gestures. In *Proceedings of the 27th international conference extended abstracts on Human factors in computing systems*, pages 4555–4560, Boston, MA, USA, 2009. ACM.

[55] C. H. Morimoto, D. Koons, A. Amir, and M. Flickner. Pupil detection and tracking using multiple light sources. *Image and Vision Computing*, 18(4):331–335, March 2000.

[56] C.H. Morimoto, D. Koons, A. Amit, M. Flickner, and S. Zhai. Keeping an eye for HCI. In *Computer Graphics and Image Processing, 1999. Proceedings. XII Brazilian Symposium on*, pages 171–176, 1999.

[57] W.T. Nelson, L.J. Hettinger, J.A Cunningham, M.M. Roe, L.B. Dennis, H.L. Pick, Andrew T Junker, and C. Berg. Brain-body-actuated control: Assessment of an alternative control technology for virtual environments. In *Proceedings of the IMAGE Conference*, pages 225–232. The IMAGE Society, Inc., 1996.

[58] S. Nilsson. Interaction without gesture or speech – a gaze controlled AR system. In *Artificial Reality and Telexistence, 17th International Conference on*, pages 280–281, 2007.

[59] Takehiko Ohno, Naoki Mukawa, and Atsushi Yoshikawa. FreeGaze: a gaze tracking system for everyday gaze interaction. In *Proceedings of the 2002 symposium on Eye tracking research & applications*, pages 125–132, New Orleans, Louisiana, 2002. ACM.

[60] T. Partala, Anne Aula, and Veikko Surakka. Combined voluntary gaze direction and facial muscle activity as a new pointing technique. In *Proceedings of INTERACT 2001*, Amsterdam, 2001.

[61] D A Robinson. The mechanics of human smooth pursuit eye movement. *The Journal of Physiology*, 180(3):569–591, October 1965.

[62] Wayne J. Ryan, Andrew T. Duchowski, and Stan T. Birchfield. Limbus/pupil switching for wearable eye tracking under variable lighting conditions. In *Proceedings of the 2008 symposium on Eye tracking research & applications*, pages 61–64, Savannah, Georgia, 2008. ACM.

[63] Dario D. Salvucci and Joseph H. Goldberg. Identifying fixations and saccades in eye-tracking protocols. In *Proceedings of the 2000 symposium on Eye tracking research & applications*, pages 71–78, Palm Beach Gardens, Florida, United States, 2000. ACM.

[64] Javier San Agustin, John Paulin Hansen, Dan Witzner Hansen, and Henrik Skovsgaard. Low-cost gaze pointing and EMG clicking. In *Proceedings of the 27th international conference extended abstracts on Human factors in computing systems*, pages 3247–3252, Boston, MA, USA, 2009. ACM.

[65] Javier San Agustin, Julio C. Mateo, John Paulin Hansen, and Arantxa Villanueva. Evaluation of the potential of gaze input for game interaction. *PsychNology Journal*, 7(2):213–236, August 2009.

[66] Javier San Agustin, Henrik Skovsgaard, John Paulin Hansen, and Dan Witzner Hansen. Low-cost gaze interaction: ready to deliver the promises. In *Proceedings of the 27th international conference extended abstracts on Human factors in computing systems*, pages 4453–4458, Boston, MA, USA, 2009. ACM.

[67] S. Shih, Y. Wu, and J. Liu. *A calibration-free gaze tracking technique*. 2000.

[68] Sheng-Wen Shih and Jin Liu. A novel approach to 3-D gaze tracking using stereo cameras. *Systems, Man, and Cybernetics, Part B, IEEE Transactions on*, 34(1):234–245, 2004.

[69] Linda E. Sibert and Robert J. K. Jacob. Evaluation of eye gaze interaction. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 281–288, The Hague, The Netherlands, 2000. ACM.

[70] Henrik Skovsgaard, John Paulin Hansen, and Julio C. Mateo. How to hit tiny buttons with gaze only? In *Proceedings of the 4th Conference on Communication by Gaze Interaction*, Prague, 2008.

[71] R. William Soukoreff and I. Scott MacKenzie. Measuring errors in text entry tasks: an application of the levenshtein string distance statistic. In *CHI '01 extended abstracts on Human factors in computing systems*, pages 319–320, Seattle, Washington, 2001. ACM.

[72] R. William Soukoreff and I. Scott MacKenzie. Metrics for text entry research: an evaluation of MSD and KSPC, and a new unified error metric. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 113–120, Ft. Lauderdale, Florida, USA, 2003. ACM.

[73] R. William Soukoreff and I. Scott MacKenzie. Towards a standard for pointing device evaluation, perspectives on 27 years of fitts' law research in HCI. *Int. J. Hum.-Comput. Stud.*, 61(6):751–789, 2004.

[74] India Starker and Richard A. Bolt. A gaze-responsive self-disclosing display. In *Proceedings of the SIGCHI conference on Human factors in computing systems: Empowering people*, pages 3–10, Seattle, Washington, United States, 1990. ACM.

[75] Lotte N. S. Andreasen Struijk, Romulus Lontis, Bo Bentsen, Henrik Vie Christensen, Hector Alejandro Caltenco, and Morten Enemark Lund. Fully integrated wireless inductive tongue computer interface for disabled people. In *Proceedings of the 31st Annual International Conference of the IEEE Engineering in Medicine and Biology Society*, Minneapolis, USA, 2009. IEEE conference proceedings.

[76] Veikko Surakka, Marko Illi, and Poika Isokoski. Gazing and frowning as a new human–computer interaction technique. *ACM Trans. Appl. Percept*, 1(1):40–56, 2004.

[77] Vildan Tanriverdi and Robert J. K. Jacob. Interacting with eye movements in virtual environments. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 265–272, The Hague, The Netherlands, 2000. ACM.

[78] Outi Tuisku, Päivi Majaranta, Poika Isokoski, and Kari-Jouko Räihä. Now dasher! dash away!: longitudinal study of fast text entry by eye gaze. In *Proceedings of the 2008 symposium on Eye tracking research & applications*, pages 19–26, Savannah, Georgia, 2008. ACM.

[79] Boris M. Velichkovsky and John Paulin Hansen. New technological windows into mind: there is more in eyes and brains for human-computer interaction. In *Proceedings of the SIGCHI conference on Human factors in computing systems: common ground*, pages 496–503, Vancouver, British Columbia, Canada, 1996. ACM.

[80] Roel Vertegaal. The GAZE groupware system: mediating joint attention in multiparty communication and collaboration. In *Proceedings of the SIGCHI conference on Human factors in computing systems: the CHI is the limit*, pages 294–301, Pittsburgh, Pennsylvania, United States, 1999. ACM.

[81] A. Villanueva and R. Cabeza. A novel gaze estimation system with one calibration point. *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on*, 38(4):1123–1138, 2008.

[82] Arantxa Villanueva. *Mathematical Models for Video-Oculography.* PhD thesis, Public University of Navarra, 2005.

[83] Arantxa Villanueva and Rafael Cabeza. Models for gaze tracking systems. *J. Image Video Process.*, 2007(3):1–16, 2007.

[84] Arantxa Villanueva, Rafael Cabeza, and Sonia Porta. Eye tracking: Pupil orientation geometrical modeling. *Image and Vision Computing*, 24(7):663–679, July 2006.

[85] Jian-Gang Wang and Eric Sung. Gaze determination via images of irises. *Image and Vision Computing*, 19(12):891–911, October 2001.

[86] David J. Ward, Alan F. Blackwell, and David 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*, pages 129–137, San Diego, California, United States, 2000. ACM.

[87] David J. Ward and David J. C. MacKay. Fast hands-free writing by gaze direction. *Nature*, 418(6900):838, 2002.

[88] Colin Ware and Harutune H. Mikaelian. An evaluation of an eye tracker as a device for computer input2. *SIGCHI Bull*, 17(SI):183–188, 1987.

[89] A. T. Welford. *Fundamentals of Skill.* Methuen, 1968.

[90] Jacob O. Wobbrock, James Rubinstein, Michael W. Sawyer, and Andrew T. Duchowski. Longitudinal evaluation of discrete consecutive gaze gestures for text entry. In *Proceedings of the 2008 symposium on Eye tracking research & applications*, pages 11–18, Savannah, Georgia, 2008. ACM.

[91] Alfred F. Yarbus. *Eye Movements and Vision.* Plenum Press, New York, 1967.

[92] Dong Hyun Yoo and Myung Jin Chung. A novel non-intrusive eye gaze estimation using cross-ratio under large head motion. *Computer Vision and Image Understanding*, 98(1):25–51, April 2005.

[93] Dong Hyun Yoo, Jae Heon Kim, Bang Rae Lee, and Myoung Jin Chung. Non-contact eye gaze tracking system by mapping of corneal reflections. In *Automatic Face and Gesture Recognition, 2002. Proceedings. Fifth IEEE International Conference on*, pages 94–99, 2002.

[94] Laurence R. Young and David Sheen. Survey of eye movement recording methods. *Behavior Research Methods & Instrumentation*, 7:397–429, 1975.

[95] Zhang and MacKenzie. Evaluating eye tracking with ISO 9241 - part 9. http://dx.doi.org/10.1007/978-3-540-73110-8_85, 2007.

[96] Z. Zhang. A flexible new technique for camera calibration. *Transactions on Pattern Analysis and Machine Intelligence*, 22(11):1330–1334, 2000.

[97] Zhiwei Zhu and Qiang Ji. Eye and gaze tracking for interactive graphic display. *Machine Vision and Applications*, 15(3):139–148, July 2004.

[98] Zhiwei Zhu and Qiang Ji. Robust real-time eye detection and tracking under variable lighting conditions and various face orientations. *Computer Vision and Image Understanding*, 98(1):124–154, April 2005.