Noise Challenges in Mono-modal Gaze Interaction

A dissertation submitted for the award of Doctor of Philosophy (PhD).

Henrik Skovsgaard
Innovative Communication

IT University of Copenhagen

September 14, 2011

Abstract

Modern graphical user interfaces (GUIs) are designed with able-bodied users in mind. Operating these interfaces can be impossible for some users who are unable to control the conventional mouse and keyboard. An eye tracking system offers possibilities for independent use and improved quality of life via dedicated interface tools especially tailored to the users' needs (e.g., interaction, communication, e-mailing, web browsing and entertainment). Much effort has been put towards robustness, accuracy and precision of modern eye-tracking systems and there are many available on the market. Even though gaze tracking technologies have undergone dramatic improvements over the past years, the systems are still very imprecise. This thesis deals with current challenges of mono-modal gaze interaction and aims at improving access to technology and interface control for users who are limited to the eyes only. Low-cost equipment in eye tracking contributes toward improved affordability but potentially at the cost of introducing more noise in the system due to the lower quality of hardware. This implies that methods of dealing with noise and creative approaches towards getting the best out of the data stream are most wanted. The work in this thesis presents three contributions that may advance the use of low-cost mono-modal gaze tracking and research in the field:

- An assessment of a low-cost open-source gaze tracker and two eye tracking systems through an accuracy and precision test and a performance evaluation.

- Development and evaluation of a novel innovative 3D typing system with high tolerance to noise that is based on continuous panning and zooming.

- Development and evaluation of novel selection tools that compensate for noisy input during small-target selections in modern GUIs.

This thesis may be of particular interest for those working on the use of eye trackers for gaze interaction and how to deal with reduced data quality. The work in this thesis is accompanied by several software applications developed for the research projects that can be freely downloaded from the *eyeInteract* appstore[1].

[1] <http://www.eyeinteract.com/>

Acknowledgements

It is surprising how fast a PhD passes. I find it hard to believe that I did it. The process of completing this PhD has been interesting and challenging. Truth is, it would not have been possible without the help of a great deal of people whose efforts have been highly appreciated.

First, I would like to thank Associate Professor John Paulin Hansen for giving me this opportunity of doing this work in the Innovative Communication Group. A special thanks to Professor John Flach for hosting my stay at Wright State University and Julio Mateo for his efforts in editing and discussing our papers and for making my stay in Dayton as painless as possible. A big thanks to the COGAIN Network of Excellence for organizing the fruitful conference and covering my salary as a research assistant at the IT University. Also, I would like to thank to Professor Kenji Itoh and Associate Professor Hirotaka Aoki for hosting a visit to the Tokyo Institute of Technology. A special thanks to Kasper Nizam and the people at Socialt Udviklingscenter (SUS) for financing part of my PhD.

Thanks to my office mates plus neighbors over the time. Emilie Møllenbach for her inspiring personality, her helpfulness and for being my friend; Javier, my academic stepbrother, for our discussions on gaze interaction and great conference stays; Martin Tall for being my partner in crime during the beginning of the SUS project; Sune Alstrup Johansen and Fiona Mulvey for their helpfulness. I would also like to thank my old and new colleagues and fellow PhD students at the Innovative Communication group at the IT University of Copenhagen. Thanks for the weekly meetings, seminars and especially the informal discussions during the coffee and tea breaks.

Special thanks to Kasper and Morten for your friendship and unique humor; Friends from Egmont, for making my social life in Copenhagen more interesting; the guys from Enderslev, for not forgetting me and my friends from USG, for keeping me fit. Finally, I would like to thank Mutti, Jacob, Karina and the rest of my family for believing in me.

Recycling

This thesis is based on work conducted in the GazeGroup at the IT University of Copenhagen. Some parts of my thesis are based on research articles published in various places. Several parts of the articles have been rearranged and other parts have been removed for a better structure in this thesis. I am the main author in most of the papers but colleagues have contributed with valuable input throughout the process. A short overview of the chapters are given below and include a brief note on the contributions from the individual authors.

Chaptes 1-3 are specially written for this thesis and are not based on any previously published research articles.

Chapter 4 provides the background for the ongoing discussion of a standardization of eye tracking systems and different performance measures of eye tracking systems that have been presented in the literature. Parts of the text originates from the following article:

> H. Skovsgaard, J. San Agustin, S. A. Johansen, J. P. Hansen, and M. Tall. Evaluation of a remote webcam-based eye tracker. In *Proceedings of the 1st Conference on Novel Gaze-Controlled Applications*, NGCA'11, Karlskrona, Sweden, 2011. ACM

I designed and conducted the experiment and wrote the full version of the paper. The co-authors supported with proofreading and a few clarifying sentences.

Chapter 5 presents a literature study on computer control by gaze and the means for interacting with a gaze-based interface. Most of the text originates from a book chapter written for the book *Gaze Interaction and Applications of Eye Tracking: Advances in Assistive Technologies*:

> H. Skovsgaard, K. Räihä, and M. Tall. Computer control by gaze. In P. Majaranta, H. Aoki, M. Donegan, and D. W. Hansen, editors, *Gaze Interaction and Applications of Eye Tracking: Advances in Assistive Technologies.* IGI Global, July 2011.

The book chapter started as a joint collaboration between Martin Tall and I. My contributions to the book chapter involve an extensive literature re-

view and editing the manuscript throughout the writing process. Martin contributed in an early version of the manuscript with a few literature studies on the topic. Due to practical circumstances, Kari-Jouko Räihäs planned chapter on *web browsing by gaze* was merged with our book chapter. After merging the chapters, Kari-Jouko took part in the writing and editing process and added clarifying paragraphs and sentences where it was needed. Kari-Joukos unique contributions on web browsing and, later, gaze in games have been completely removed for this thesis.

Chapter 6 presents gaze communication and is specially written for this thesis and is not based on any previously published research articles.

Chapter 7 presents a case study on a novel zoom interface controlled by pan and zoom in a 3D environment. Most parts of the text in this chapter are rewritten from the following paper:

> D. W. Hansen, H. Skovsgaard, J. P. Hansen, and E. Møllenbach. Noise tolerant selection by gaze-controlled pan and zoom in 3D. In *Proceedings of the 2008 symposium on Eye tracking research & applications,* pages 205-212, Savannah, Georgia, 2008. ACM

The work originates from my master-thesis project with Dan Witzner Hansen as my supervisor. I designed and developed the experimental system during my period as master student and research assistant at ITU. I collected the experimental data for the paper and helped writing several sections together with Dan. Dan took to lead in writing and was a contributor in the design process as well of the experimental process. John Paulin Hansen and Emilie Møllenbach were the main authors of related-work section and John conducted major parts of the data analysis.

Chapter 8 presents gaze-based tools to accommodate for the limited accuracy of eye trackers. This chapter presents three experiments, which primarily originate from the following articles:

> H. Skovsgaard, J. C. Mateo, and J. P. Hansen. Evaluating gaze-based interface tools to facilitate point-and-select tasks with small targets. *Behaviour & Information Technology,* Apr. 2011.
>
> H. Skovsgaard, J. C. Mateo, J. M. Flach, and J. P. Hansen. Small-target selection with gaze alone. In *Proceedings of the 2010 Sym-*

posium on Eye-Tracking Research & Applications, pages 145-148, Austin, Texas, 2010. ACM Press.

I designed and developed the tools for the experimental systems in both papers. I was responsible for the data collection, data analysis and wrote the first draft for both papers. Julio Mateo provided valuable input during the process and his editing skills are responsible for the papers current form. John Paulin Hansen supported with proofreading and a few clarifying sentences in both papers.

Chapter 9-10 consist of the conclusion, future work and prologue. All chapters are specially written for this thesis and are not based on any previously published research articles.

Contents

List of Figures

List of Tables

Part I

Introduction, Motivation & Background

1 Introduction

In 1973 researchers at Xerox PARC invented the predecessor to the modern interface that allowed for direct manipulation of a computer. Seven years later, Merzouga Wilberts coined the term WIMP *(Window, Icon, Menu, Pointing device)* interface, later GUI (Graphical User Interface) that enabled a pointing device and rendered objects to be manipulated directly on the screen [144, 145]. Commands via the GUI are compiled into menus and actions are performed through an input device such as a mouse. In the personal computer, all the interactive elements are modeled through the desktop metaphor, to produce a simulated environment. In comparison, human-human interaction has a richer palette of interaction methods available. Consequently, several Human-Computer Interaction (HCI) research initiatives explore new devices, interfaces and interaction methods in terms of their applicability to human-computer interaction paradigms. One of several challenges in exploring these new paradigms is dealing with noise from an inaccurate input device. A recent example is the difficulties of controlling a touch-screen mobile telephone while walking. The main example investigated in this thesis is the noise from an eye tracking system that some people with disabilities are constrained to in their interaction with computer based technologies. This group are often reliant on computer based systems for everyday communication and work. Thus, any noise or error which limits their control of gaze driven interfaces is essentially a barrier to their communication with other people and their ability to work. Noisy input, in this context, becomes a quality of life issue.

In computer science, the concept of *noise* is essentially a disturbance that obscures the clarity of a signal with meaningless data. Noise is divided according to the type, source and effect, and can be regarded as *external noise*

which has its source in factors external to the receiver and *internal noise* which has its source within the receiver itself. Examples of external noise in gaze interaction include disturbing sunlight and physiological limitations of the human eye. Examples of internal noise include inaccurate sensors and poor eye and gaze estimation models. This thesis deals with noise in an HCI context. Consequently, noise is defined as signal interference from either an interaction-based, technological or physiological source.

1.1 Motivation

Interactive eye tracking systems provide the necessary technology to perform computer interaction with the eyes only. Although the eyes are excellent for pointing tasks they lack a reliable selection mechanism and often require dedicated interfaces for viable performance. The main user group consists of people with motor-disabilities and a significant proportion of potential users do not have access to this technology due to the high costs of commercial systems.

For the last two to three decades eye tracking has received a great deal of attention within the area of HCI. A lot of the effort in the eye-tracking community has been directed towards improving the technology for users who are suffering from motor disabilities and this particular group has a special need for techniques to substitute the conventional computer mouse. For the first time, users were able to communicate with their family and friends using only gaze as the input.

Bolt [14] notes in 1981 that eyes could be used used as output. He wrote: *At the user/observer interface level, interactivity is extended to incorporate where the user is looking, making the eye an output device* [14, p. 118]. Some have raised skepticism towards using the eyes as output [181]. Eyes may be well suited for pointing, as stated by Bolt, but they are always 'on' and the lack of a reliable activation mechanism may lead to *the Midas Touch problem*. This relates to the story of the mythical figure of King Midas who turned everything he touched into gold, including his own daughter. The term was first used in the context of gaze interaction by Jacob [64] and refers to an inspectional fixation being misinterpreted for a selection. The design challenge here is to reliably distinguish between the user's actions in terms

of inspections and selections.

Another challenge to use gaze as input is the relatively low precision of gaze selections in remote systems (i.e. where the eye cameras are not attached to the user's head). While it is possible to point with the traditional mouse with pixel precision, most manufacturers of remote interactive eye tracking systems can only provide an accuracy of $0.5° - 1°$[1] at best, which approximately translates to $20 - 40$ pixels measured at a 50 cm distance to the monitor. Figure 1.1 shows how difficult it is to control a standard interface with this low accuracy. For instance, placing a cursor in a text string for editing requires and accuracy as high as three pixels according to Drewes [25]. It is very challenging to perform these kinds of tasks using gaze alone, unless special tools are employed. A part of this thesis will be investigating novel tools that can be used to improve the accuracy of gaze interaction.

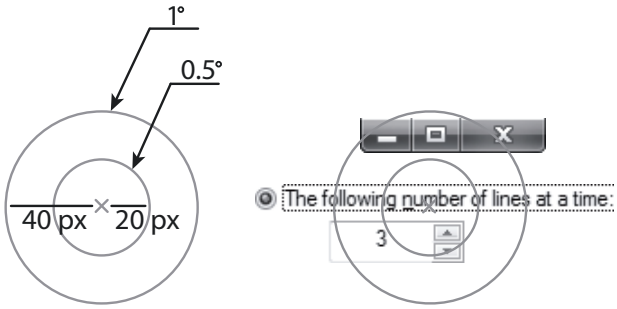


Figure 1.1: *Circles representing* $0.5°$ *and* $1°$ *taken from the reported accuracy of different eye tracking systems. On top to these circles are placed well-known objects taken from the Windows interface.*

User Case: Danish Associate Professor in theoretical physics Arne Lykke Larsen was in the age of 35 diagnosed with ALS. He has had the disease for more than ten years, which has rendered him completely paralyzed apart from a few facial expressions (e.g., moving an eyebrow) and eye movements. Arne is the author of the book 'Rather die of laughter than ALS - 99 truthful stories of what it means to live with Amyotrophic Lateral Sclerosis' [88]. Arne is using his Tobii P10 system for several hours during a day. In 2009,

[1] <http://www.cogain.org/wiki/Eye_Trackers>

during a discussion about technology, software and the use of low-cost eye tracking systems, Arne presented his views on his communication system: "My cursor is drifting near the corners of the screen and it complicates activations a lot. Tobii does not think about interfacing Windows, rather, they only think about their Tobii Communicator[2]." In other words, even a proficient user may have problems hitting targets with one of the most accurate interactive systems on the market.

In gaze interaction, where the on-screen cursor is directed by the eyes, a common approach to dealing with spatial error is to use large on-screen buttons and controls [23]. This approach limits the number of interactive elements that can be presented on the screen. A part of this thesis will investigate a novel interface paradigm that allows the user to navigate a dense information space with eye movements only (i.e., panning and zooming for object selection).

Gaze tracking systems enable people with severe motor disabilities to communicate using only their eyes. However, some of them cannot afford a commercial system, which cost between $5,000 and $30,000. Hansen et al. [43] noted how systems designed with low-cost off-the-shelf hardware would facilitate access, lower the system costs, promote research within the field of eye tracking and promote the development of new interesting gaze-driven applications.

A COGAIN[3] report by Jordansen et al. [75] estimates that about 0.6 million people in Europe (corresponding to 0.082% of the population in 2005 numbers) could benefit from gaze interaction. Obviously, not every single person of this group would choose eye tracking since there are various alternatives to the technology but as Majaranta [98, p. 32] states: *For many of them, eye control is potentially the quickest, least tiring, and most reliable form of access to technology - by far.* Extrapolating the COGAIN estimation to world population suggests that there could be more than 3 million potential users in Asia alone if the technology becomes inexpensive (See table 1.1).

[2]Tobii Communicator is a software platform that employs large gaze-enabled buttons.

[3]*COGAIN* (Communication by Gaze Interaction), is a 5 years project supported by the European Commission's IST 6th framework program that turned into an association in 2009.

Table 1.1: *Approximate number of potential users that could benefit from gaze interaction in the world (based on 2005 numbers, in millions). Source: Wikipedia*

Continent	Asia	Africa	Europe	S. America	N. America	Oceania	World
Total	3,937	921	729	557	335	34	6,512
Potential Users	3.23	0.76	0.6	0.45	0.27	0.03	**5.34**

The ITU Gaze Tracker is an inexpensive open-source gaze tracking software that can be used with low-cost and off-the-shelf hardware, such as webcams and video cameras. The software is able to track the pupil and one or two corneal reflections produced by infrared light sources. The first version of the system was introduced and evaluated by San Agustin et al. in [136]. Launching the ITU Gaze Tracker as an open-source system has resulted in code, hardware and donations from people who are interested in promoting the low-cost system and the technology in general. As of September 1. 2011, the system has been downloaded more than 18,000 times by users all over the world.

Working with low-cost equipment introduces some challenges. The quality of the hardware is often significantly lower that the quality of high-end hardware. For instance, an inexpensive camera (e.g., webcam) is often shipped with relatively low resolution and a cheap lens that causes a reduction of the image quality. Finally an inexpensive camera has lower temporal resolution (i.e, frame rate) compared to commercial systems. Finally, the very broad field of view of a standard camera makes tolerance to head movements low since the effective image of the eye is very limited. All in all, this may have a negative impact on accuracy and precision of gaze interaction. In this thesis, there will be a performance comparison between a webcam-based version of the ITU Gaze Tracker and two commercial systems. The question is whether we can produce cheap systems which provide viable input for everyday tasks and for everyday interfaces, using software solutions to hardware limitations.

1.2 Thesis Goals

The aim of the research presented in this thesis is to understand the interaction challenges that people are having when using gaze as mono-modal

input. The first step towards understanding the range of challenges experienced by disabled users is a review of previous interface designs and an evaluation of the accuracy of three gaze tracking systems. On this basis, I will present novel gaze-zoom principles designed to address the noise challenge and a novel selection technique that allows to select small targets in a standard GUI. It is my hope and ambition that my work will lead to a more widespread use of gaze interaction among motor-disabled people, since the new designs will work with an affordable low-resolution gaze tracking system.

This thesis mostly covers research for users who are motor challenged. The thesis does not focus on eye-tracking hardware or the lower levels of processing such as image analysis and gaze estimation techniques nor physical, organizational and social circumstances of the individual user.

1.3 Research Question

The research question that will be addressed in this thesis is:

> *How to counter the inherent noise in gaze control of computers by design of interface tools and applications?*

The methods and approaches that will be used to address the research question in this thesis are largely experimental. Experimental methods fall into two broad categories: qualitative methods and quantitative methods. With the exception of the qualitative case study presented in Section 6.1.2, all experiments in this thesis are based on quantitative approaches. Although this may not be common in computer science, this is appropriate in the field of HCI.

1.4 Thesis Overview

This thesis is divided into 5 major parts and is outlined as follows:

- **Part 1: Introduction, Motivation and Background**
  - **Chapter 1:** provides an introduction to the topic, the motivation for the work, and the methodology used throughout the thesis.

- **Chapter 2:** introduces the human eye and eye tracking technologies. What are the natural limitations of the human eye? Contemporary eye tracking techniques will be explained and issues related to video-based eye tracking will be discussed.

- **Part 2: Quality Factors in Interactive Eye Tracking**

  - **Chapter 3:** introduces the different factors that have an effect on the quality of interactive eye-tracking systems based on the VOG technique. This includes the limitations of the human eye, the calibration procedure and the different types of noise that contribute to the systems' spatial error. The path from eye movements to gaze-aware application is explained along with a description of the individual stages in the model.
  - **Chapter 4:** introduces the ongoing discussion of a standardization of eye tracking systems and reviews various performance measures of eye tracking systems presented in the literature. The study presents a test of interactive eye trackers employing two commercial systems and a webcam version of the ITU Gaze Tracker. This study evaluates spatial error and target acquisition under the Fitts' law framework.

- **Part 3: Gaze Interaction**

  - **Chapter 5:** presents a literature review of gaze based control of a computer and more traditional means for interacting with an interface.
  - **Chapter 6:** presents augmentative and alternative communication in relation to eye tracking. A case study of a person with ALS is used to illustrate why communication interfaces are so important for users with special needs. Examples of eye typing systems presented in the literature are discussed and classified according to input method. Finally, a review of gaze-based typing systems based on mono-modal input are presented and discussed.

- **Part 4: Novel Zoom Strategies in Gaze Interaction**

  - **Chapter 7:** presents StarGazer, a novel zoom interface that is controlled through continuous pointing gestures. StarGazer is a

noise tolerant gaze-based interface for exploring graph-based data. The layout of the interface was configured for a typing task and three different experiments were employed in the evaluation: a name typing task, a latency test, and a speed writing test.

- **Chapter 8:** presents gaze-based tools to accommodate for the limited accuracy of eye trackers. This chapter presents two studies consisting of a total of three experiments. The first two experiments evaluate a novel tool based on continuous zooming in two different interfaces where the location of the selectable objects are known and unknown. A third proof-of-concept experiment compares an optimized version of the tool to the existing tools and was found to be the most accurate gaze-based technique for small target selection.

- **Part 5: Conclusion & Future Work**

  - **Chapter 9:** is the chapter in which the overall conclusions from the experimental design carried out throughout this thesis are drawn. Furthermore, some reflections on future research within the area of gaze-based interaction are presented.

  - **Chapter 10:** is an epilogue that describes my research path in and my publications list.

2

Tracking the Human Eye

The lowest functional layer in eye tracking constitutes human eye movements. In order to be able to do research in gaze interaction and gaze-controlled interfaces, a basic understanding of the human eye and its limitations is therefore needed. Furthermore, techniques to track movements of the human eyes and process the input signal are cornerstones in gaze interaction.

This chapter gives an overview of the anatomy of the eye and how this light-sensitive organ absorbs light impulses from electromagnetic radiation of charged materials. The characteristics of the human eye are presented together with the different types of eye movements in a physiological setting. The purpose is to explain the important and relevant aspects of human eye in relation to eye tracking technologies in general. The latter part of the chapter deals with different contemporary eye tracking techniques with focus on video based eye tracking and initiatives from the open-source eye tracking community.

2.1 The Eye

Evolution has over time turned the eye into an organ that carefully detects and filters a small specter of electromagnetic energy into meaningful information that can be interpreted by the brain. Six muscles, organized in three pairs, give the eye three degrees of freedom. The different pairs of muscles control horizontal movements, vertical movements and finally, rotational movements around the direction of view. The muscles work together in order to compensate for head movements guided by the equilibrium organ in the inner ear.

A schematic view of the eye is shown in Figure 2.1. Three parts of the eye is visible from the outside: the white curved sclera, the black pupil, and the colored iris. The cornea covers the pupil and the iris with a transpar-

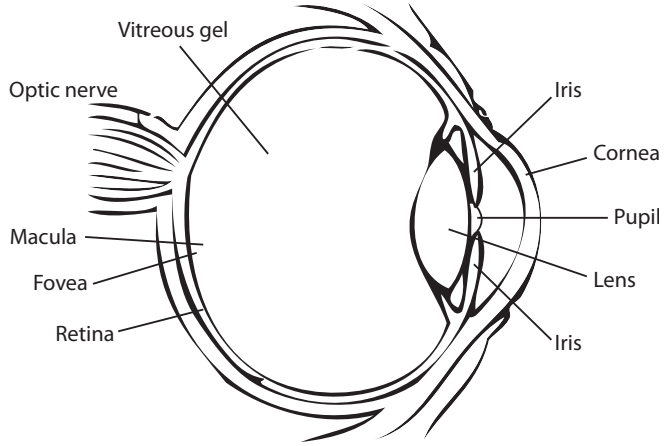


Figure 2.1: *Diagram of the human eye. Adapted from the National Eye Institute <http://www.nei.nih.gov/>.*

ent layer, which refracts light before it enters the eye. The opening at the center of the iris is the pupil that regulates the amount of light that passes through. The lens is a clear part of the eye behind the iris that helps focus the rays of light onto the retina, which is the light-sensitive tissue lining on the back on the eye. Photoreceptor cells convert the light into impulses that are transmitted to the brain through the optic nerve. The photoreceptors consist of two main groups of cells: the rods and cones. The rod cells provide monochromatic vision. They respond to varying brightness levels in the environment. The cones are less sensitive to contrast and dim light but more sensitive to color and rapid changes. Cone cells and rod cells are responsible for daylight vision and cone cells are mainly placed on the macula; an area of about 5° that is responsible for the central vision. Inside the macula area is the fovea that is responsible for the sharpest vision and covers circular area of about 1° - 2°. Around 150,000 $cones/mm^2$ together with a low number of rods is located inside the fovea, and in contrast, only 20,000 $cones/mm^2$ are located outside the fovea. This means that the visual field derived from the cells outside the macula is of low resolution and can only assist in the

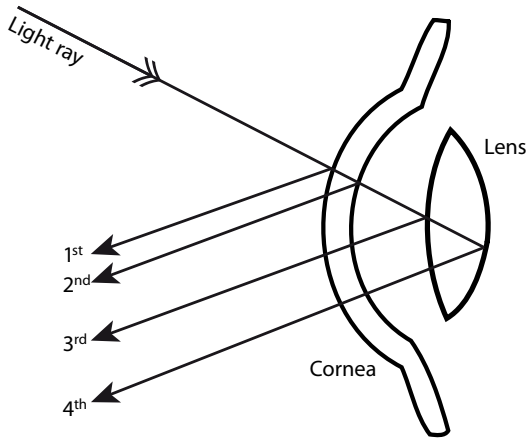


Figure 2.2: *Arrangement of the four Purkinje images. Adapted from the Fourward Technologies, Inc. <http://www.fourward.com>.*

perception of ambient motion [28]. Finally, signals from the photoreceptors are carried through the optic nerve, consisting of more than a million nerve fibers, from the retina to the brain (visual cortex).

The eye is not completely symmetric but a line connecting the different lenses within the eye can be approximated. This line is called the optical axis or *line of gaze* and should not be confused with the visual axis or *line of sight*. The visual axis is the line that connects the object currently being inspecting with the fovea. The angular offset between the visual and the optical axis differs from subject to subject with average values of about 5° and 1.5° degrees, in the horizontal and vertical directions respectively. When the eye is subjected to light it is refracted in the cornea and the lens but some of the light will be reflected back in four different reflections, known as Purkinje images (see Figure 2.2).

2.1.1 The Nature of the Eye

While awake, the eyes are constantly moving and they need to relate to the vast amount of visual information surrounding us. The eyes always work synchronously in order for humans to achieve image recognition, the eye needs a stable projection on the retina. To keep the eyes on a target, the three pairs of muscles compensate for head and body movements by moving the eyes in

the inverse direction of the movement. This is known as the vestibulo-ocular reflex. This reflex can be verified by keeping the head in a stable position then the eye-gaze direction jumps in abrupt movements called saccades and after these jumps the eyes rest for a short periods called fixations, which represents the stable projection on the retina [18].

Eye movement data, as with all signal from biological processes, is inherently noisy, as noted by Yarbus [179]. Even when subjects were asked to follow the outlines of geometrical figures as smoothly as possible Yarbus discovered here how eye movements could be viewed as a combination of fixations and saccades (see Figure 2.3).

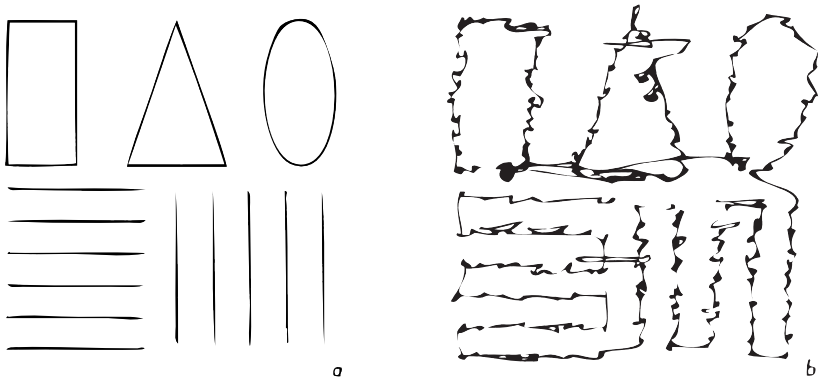


Figure 2.3: *A subject is asked to follow the lines of the figures as smoothly as possible (a) and the resulting gaze path (b). Reproduced after Yarbus [179].*

2.1.2 Eye Movements

The eyes only move in a limited number of ways and those are fundamentally the same for most people [86]. Saccades are ballistic movements that cannot be interrupted nor changed. When a saccade occurs no information is gathered for the visual system (saccadic suppression). During a saccade the eye rotates at high velocity and for large amplitudes the peak velocity is under certain circumstances able to exceed $700°/s$, making the saccades the fastest movements produced by the human body [18]. Rommelse et al. [131] list four different types of saccades:

- **Visually guided saccades** where the eyes are moved towards a visual onset or stimulus. Usually, this serves as a baseline when measuring other types of saccades.
- **Antisaccades** where the eyes movements are moved in a direction opposite to the side where a stimulus was presented.
- **Memory guided saccades** where the eyes move toward a point from memory without any visual stimulus.
- **Predictive saccades** where the eyes are performing catch up saccades while following a moving object, also known as smooth pursuit.

While measuring amplitude A and duration T for saccadic eye movements, Carpenter [18] found a linear relationship between the two variables and described it as $T = 2.2\ \text{ms}/^\circ \cdot A + 21\ \text{ms}$. Although the model does not incorporate forced changes of the muscles, it gives a good estimate of the duration of the saccade. Later, Land & Tatler [86] formalized the model into more general terms (see Equation 2.1). The time or duration D of a saccade could be expressed as the minimum duration of saccade, D_0 plus the duration increase per degrees of amplitude, d multiplied with the amplitude, A. Where the minimum duration falls somewhere between 20 to 30 ms and d is in the range of 2 to 3 ms per degree. Saccades last approximately $30 - 120$ ms and in between there are fixations, which typically lasts approximately $200 - 600$ ms [66].

$$D = D_0 + d \cdot A \tag{2.1}$$

Besides the traditional saccade there are four additional types of eye movements: smooth pursuit, vestibulo-ocular reflex, vergence and optokinetic nystagmus [127]. The movements of a smooth pursuit are slower than saccadic eye movements and has a latency of about 120 ms. *Smooth pursuit* movements are used after saccades to correct any positional errors between the eye and the target, which explains why smooth pursuit movements can only take place when tracking a visual object. As mentioned earlier, the *vestibulo-ocular reflex* occurs when the head is rotating to maintain the eye position in space. With help provided by inner ear sensors, the eyes perform a counter rotation with respect to the head in order to maintain fixation on a target of interest. *Vergence* refers to a simultaneous movement of both eyes either with

a convergent and divergent rotation, to obtain or maintain binocular vision. When inspecting a close-by object, the eyes are rotating towards each other, known as convergence, while for objects further away; the eyes are rotating away from each other, known as divergence. The maximum diverge is until the eyes are parallel when they point at infinity. Vergence movements are triggered by a change of focus of objects. The final movement is *optokinetic nystagmus*, which is an alternation between smooth pursuits and saccades. The movement resembles a "sawtooth" and is best know from fixating on different objects while sitting in a car or riding a train.

Fixations occur when information about the surroundings are gathered and even when the eyes are fixating, they are still moving. Interestingly enough, the magnitude of these fixational eye movements should be visible to the observer himself but the brain filters the visual signal, rendering them invisible. If eye movements are counteracted, the visual perception fades completely as a result of neural adaption. In other words, this means that the details obtained from a perfect fixation would fade from view [129]. Although this effect sound counterintuitive at first, it is common in sensory modalities. For example, wearing shoes for 16 hours a day and wiggle the toes to notice that the shoes are on. This is analog with releasing the eye from stabilization and the visual perception reappears [103].

It has been known for a long time that the eyes never are at rest. Early observations, such as Jurin, in 1738, *"the trembling of the eye"* and Hemholtz, in 1860, *"wandering of the gaze"*, indicate that researchers had noticed these micro movements during fixation. In modern time, researchers have agreed on three types of eye movements during fixations: *tremor*, *drifts* and *micro saccades* [179, 18]. Martinez-Conde et al. [103] summarizes these movements:

- **Tremor**, the smallest of all the eye movements with amplitude of about the diameter of a cone in the fovea. Tremor is thought to operate independently in the two eyes, which sets a physical lower boundary for the visual system during stereovision.

- **Drifts**, the slow motion of the eye that occur during the epochs between micro saccades. While drifting, the object being inspected can move across a dozen photoreceptors. Drifts seem to have a compensatory role in keeping the vision accurate in the absence of micro saccades.

- **Micro saccades**, small and fast eye movements that occur during fixation. The movements are about 25 ms in duration and are able to move from across dozens to several hundred photoreceptors. One of the roles of micro saccades is to correct displacements of the position of the eye during drifts. That is, if the drifts take the fixation target away from the fovea, micro saccades will bring the target back on the fovea. It has been suggested that the mechanism generating the large saccades and the micro saccades is the same due to a combined relationship between the velocity and amplitude for the two [186].

Figure 2.4 shows the effect of these micro movements. Look at the central black dot for about a minute, and then look at a white dot in an adjacent square to experience the effect.

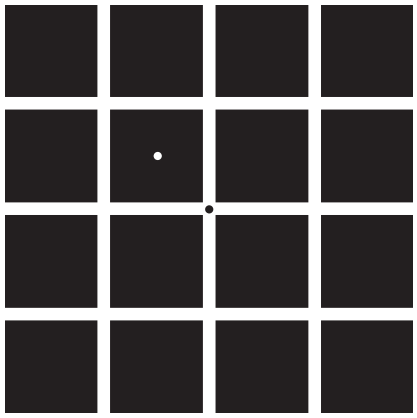


Figure 2.4: *An example of a pattern to show fixational eye movements. In order to experience it, look at the central black dot for about a minute, then look at the white dot in the adjacent dark square. The dark after-image of the white line pattern should be seen in constant motion attributed to fixational eye movements. Reproduced after Verheijen [167].*

Identifying Eye Movement

The analysis of fixations and saccades in gaze interaction require some form of mechanism to identify fixations. That is, the translation from the raw eye movement data points to fixation locations and the saccades between them.

Conducting a fixation identification helps to reduce the size and the complexity of the raw eye-movement protocol and presents the data in one logical list of ordered elements that is useful for at least two reasons: one, little to no visual processing happens during a saccade [36] and is often irrelevant for most gaze-based applications, and two, the small eye movements that occur during fixations (e.g., tremors, drifting, flicks), have little to no meaning at higher levels of gaze interaction [22]. To put the record straight, it should be noted that feature selection is by some eye tracking experts known as event detection. Although it may be desirable to reduce the feature space of the raw data protocol only to include fixation it can have a negative effect on smooth pursuit detection, which have a direct relevance for gaze interaction, and cannot be represented in the data as single x, y events. The drawback of reducing data only to fixation data, even if only fixations are of interest is that the smooth pursuit movements will be reported as a series of small fixations and saccades, or, will be entirely removed, or, will result in a massive, long fixation being reported where none actually occurred.

Smooth pursuit detection is the ability to track any moving objects. As mentioned the techniques to separate fixations from saccades is a common task in gaze tracking systems. The detection of these movements requires a moving stimulus and extends to the existing gaze movement detection algorithms. The classification of smooth pursuit movements is a difficult task since the spectrum of fixation- and smooth-pursuit velocities overlap. The physiological limitations of the visual system is limited to a range between $1°/s$ - $30°/s$ while tracking objects with the human eye [130]. The smooth pursuit movements can be divided into two parts; a motion and a saccadic component. The motion component maintains the fovea stabilized on the moving object and the saccadic component is used for minor corrections and repositioning the fovea on the moving target [134]. Robinson [130] defines the following stages in a smooth pursuit movement: the initial smooth pursuit, the correcting saccade that places the objects on the fovea and a final smooth pursuit. This is all done to match the velocity of the moving target. The visual system is able to track moving objects with the motion component only if the target velocity is below a certain saturation velocity threshold. Otherwise, if the target velocity is above the threshold the saccadic movements occur.

A poor identification algorithm may produce too many or too few fixations.

The identification algorithm can also be too sensitive to outliers (i.e., data points that are deviating extremely from the mean). This was demonstrated by Karsh and Breitenbach [77] that proved how different identification algorithms on gaze data produced vastly different interpretations on the same data protocol. Section 3.5 presents different real-time fixation detection techniques based on spatial and temporal characteristics.

2.2 Eye Tracking

Knowing the gaze direction is essential for natural communication and for interaction with computers. The following sections will present contemporary technologies thus the historical methodologies on eye tracking and gaze estimation will not be covered in detail in this thesis.

There are two main application areas in eye tracking: diagnostics and interaction. Diagnostic applications gather information on eye movements for research in many fields (e.g., psychology, health, marketing and usability). In general, diagnostic eye tracking concerns cognition and attention of users during specific tasks such as reading or driving a car. In diagnostics eye tracking, the eye is solely used to direct information on eye movements. In contrast, there are applications for interaction where the eyes are used both as input and output device. The latter area will be covered in greater detail throughout this thesis.

2.2.1 Contemporary Eye Tracking Techniques

Locating the direction of gaze successfully involves two operations: tracking of the eyes, that measure features of the eye while moving in its orbit, and gaze estimation that estimates eye gaze based on eye features. The contemporary techniques of eye tracking are usually divided into three main categories: *electro-oculography* (EOG), *contact lenses* and *video-oculography* (VOG).

EOG is a relative invasive technique that measures the potential of the retina with pairs of electrodes placed either on the left and right side of the eye or above on below the eye (see Figure 2.5). Moving the eye from the center position towards one electrode will generate a potential difference on

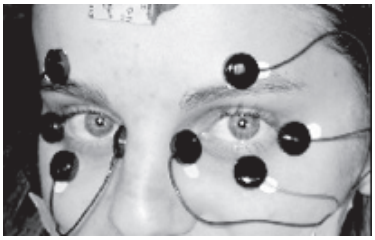


Figure 2.5: *A participant using an electro-oculography system. The electrodes are placed in pairs around the users eyes (i.e. left - right and above - below). Curtesy of MetroVision, Pérenchies, France <http://www.metrovision.fr>.*

Figure 2.6: *Scleral search coil lens mounted on the eye. Adapted from Murphy et al. [118].*

the retina between the closest electrode and the opposite electrode. If the potential on the retina is constant, the EOG method can be a good measure of eye position [28]. The big advantage of EOG is that users easily can use contact lenses or glasses without any effect of performance. Furthermore, the system allows for large head movements and is relatively inexpensive [134].

Contact Lenses are invasive, yet they the most accurate of the contemporary techniques. A small coil (known as a search coil) is embedded into a contact lens that is fit over the sclera with a slight suction to avoid drifting during eye movements (see Figure 2.6). An electro-magnetic field measures the voltage induced in the search coils to estimate the users gaze with accuracy down to 0.08° [117]. From a usability perspective, the invasive lens and the wires connecting the lens makes the system uncomfortable and limits the technique to laboratory research [134].

(a) Remote MyTobii P10 system.

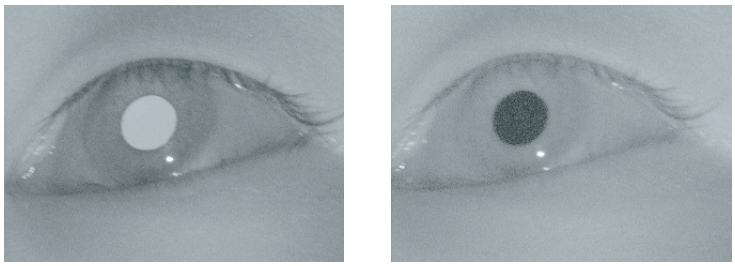
(b) Head mounted Tobii Glasses.

Figure 2.7: *The classification of gaze tracking systems can either be (a) remote or (b) head mounted. Curtesy of Tobii Technologies <http://www.tobii.com>.*

VOG employs camera technology that records the eye movements of the user in order to extract features for the gaze-direction estimation (i.e., Point-of-Regard in 2D or Line-of-Sight in 3D) [28]. A VOG system is the least intrusive of the contemporary systems and are thus the best suited for interactive applications since the users in a remote setting does not have to wear any gear and still allows for some head movement. VOG-based eye trackers are very common in experimental HCI research and are also used throughout this thesis.

A gaze tracking system can be classified as *remote* or *head mounted*. In the remote setup the camera and light sources are placed away from the user. The camera and the light sources can either be mounted on a stand or placed around a monitor (see Figure 2.7). In the remote setup, most systems provide head-pose invariance as long the users eyes are kept within the field-of-view of the camera. For better performance, a chin rest can be employed to fixate the head position during experiments. A head mounted eye tracker has the light sources and the cameras built in to a helmet or a pair of glasses thereby allowing for mobile interaction since the user is not limited to sit in front of a fixed setup.

There are several means of detecting eye movements with the VOG method: shape recognition based on geometric eye models can be constructed from eye features [92], appearance-based methods that build a model of the eye in



(a) Bright pupil effect. (b) Dark pupil effect.

Figure 2.8: *The typical techniques employing IR light sources in eye tracking are (a) bright pupil effect and (b) dark pupil effect. Image source: Morimoto et al. [115].*

real-time with a large set of training data, and finally, the hybrid methods that combines the best features from the shape recognition and the appearance features [41].

The use of infrared light (IR) is common in many eye-tracking systems for a better and more robust tracking of eye features and gaze estimation. IR light provides a stable illumination and the reflections can also be used to generate reflections on the corneal surface, which is known as corneal reflections or glints. Two techniques are typically employed when using IR light sources and depends on the lights placement in regard to the optical axis of the camera. Placing light sources close to the optical axis of the camera (i.e., on-axis light) generates a bright pupil effect, similar to the red-eye effect that occurs in standard photography with a flash. When light sources are placed away from the optical axis of the camera (i.e., off-axis light) a dark pupil effect occurs and the pupil appears darker (see Figure 2.8). Using IR lights in eye tracking is not perfect since they have a low tolerance to reflections in the eye region (e.g., contact lenses, glasses or sunlight), that mostly restrict the systems to be used indoor.

A special case of eye detection is the Eye Contact Sensor (ECS) by Dickie et al. [21]. The system employs the same parts that exist in a modern VOG eye tracking system but is only used for attention detection. Basically, the ECS is a simple sensor consisting of a camera surrounded by IR lamps to generate bright pupil effect. The ECS is capable of detecting user attention

Figure 2.9: *The Eyebox2 commercial eye contact system is capable of detecting attention up to 10 meters away. Curtesy of xuuk <http://www.xuuk.com>*

by measuring corneal reflections of nearby passing persons. If a glint falls inside the center of the pupil it is safe to assume that the given user looked directly at the camera and thus paid attention. The inventors have developed a commercial spinoff called *eyebox2* with a documented range up to 10 meters (see Figure 2.9). The main application area for such a device is to track when and who is looking at screen ads or posters in a mall for example or as an ubiquitous gaze switch.

2.2.2 Low-Cost Eye Tracking

Over the recent years there have been several open-source projects that aim at bringing down the costs of eye tracking systems. Furthermore, it allows for modifications that are normally not possible because information is kept as intellectual property by the manufacturers. For example, the EyeWriter2[1] is a low-cost eye-tracking device that allows an artist who suffers from ALS patient to paint with the eyes only. The system is head mounted and therefore not tolerant to head movements. The system does not offer a rich palette of options to the user except for painting. Development of the system has currently been inactive for more than a year. In 2004, Babcock and Pelz [4] presented a head mounted eye-tracker that uses two small cameras attached to a pair of safety glasses (see Figure 2.10). Li et al. [91] extended their work and built a similar system that worked in real time, called OpenEyes. Being headmounted, both systems are affected by head movements and are

[1]See: <http://www.eyewriter.org/>

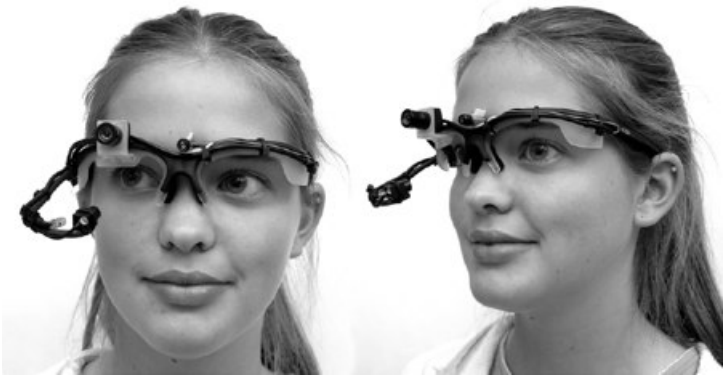


Figure 2.10: *User wearing the head mounted low-cost eye tracker by Babcock and Pelz [4]. Copyright J. Babcock. Used with permission.*

thus not suitable for use in combination with a desktop computer. Although the components used in the systems described above are inexpensive, assembling the hardware requires advanced knowledge of electronics. Zielinski's Opengazer system [185], based on a remote webcam, takes a simpler hardware approach. The gaze estimation method is not tolerant to head movements, and therefore the user needs to keep the head still after calibration. Sewell and Komogortsev [141] developed a neural-network based eye tracker able to run on a personal computer's built-in webcam under normal lightning conditions (i.e., no infrared light). The aim of their study was to employ eye tracking without any modifications to the hardware. The five participants in the study complained that even during fixations they felt a jumpy sensation of the marker, and that the marker was unstable during use.

The ITU Gaze Tracker

The ITU Gaze Tracker is a cheap open-source gaze tracking software that can be used with low-cost and off-the-shelf hardware such as webcams and video cameras. The software tracks the pupil and one or two corneal reflections produced by infrared light sources. The first version of the system was introduced and evaluated by San Agustin et al. in [136].

The core functionality has been updated to offer support for remote tracking, which that users do not have to place the camera close to their eyes. Thanks

Figure 2.11: *Two different examples of user-driven projects that employ the ITU Gaze Tracker. The first project focused on studying the male gaze in fashion design (left) and the second project focuses on gaze attention during an in-vehicle study (right) [139]. Copyright R. Dasgupta and N. Schneider. Used with permission.*

to the open-source community, parts of the code base have been updated by developers who are interested in improving the system. So far, the system has been downloaded more than 19.000 times by users all over the world. The ITU Gaze Tracker has been used by people in various projects. The systems has, for example, been employed in studies of the male gaze in fashion studies[2] and by Schneider et al. [139] for detecting drivers attention during an in-vehicle study (see Figure 2.11) At Texas State University the system has been used by the students for simple interaction applications such as picture browsers and at IT University of Copenhagen the system has been used with the OGAMA[3](Open Gaze And Mouse Analyzer) to conduct usability studies.

2.3 Summary

This chapter has demonstrated how the human eye is working and established how the eyes need to rest on a target (i.e., fixate) in order to gather information for the visual system. The anatomical features of the human eye that have an effect on eye movements have been covered. The size of

[2] <http://www.rupadasgupta.com/>
[3] <http://www.ogama.net/>

the fovea (about $1^\circ - 2^\circ$) sets a lower standard for accuracy on eye tracking systems. However, it is possible to obtain a higher accuracy ($> 0.5^\circ$) since accuracy is also decided by the precise position of the eye during calibration.

This thesis only focuses on the popular VOG technique and IR light sources. Some of the problems related to the VOG technique is that the method has low tolerance to large head movements, to natural light and to reflections in the eye region.

Part II

Quality Factors in Interactive Eye Tracking

3

Interactive Eye Tracking

There are several factors that have an effect on the quality of the raw eye data and the resulting accuracy of the calculated point-of-regard (PoR) (see Figure 3.1). The following sections present the initial limitation of the eye, the process of calibrating on a VOG system and the different types of noise or error in data from a VOG system. Any measure of a biological system will contain some error, and this error should be measured and characterized in order to consider how to deal with it for optimal interaction experience. The data flow from eye movements to application is described in two models that motivate the data processing stages applied to the system input and the consequences of these processing stages in terms of system latency. Finally, eye movement detection is discussed with respect to system latency followed by a presentation of real-time eye event detection algorithms.

3.1 Accuracy of The Eye

The manufacturers of commercial remote eye tracking systems often claim an accuracy of 0.5° of vision and a higher resolution for standard remote VOG systems is rarely reported to be above this level. Some high-end remote high-speed systems based on the VOG technique combined with chin rests have accuracies of 0.1°, since the accuracy is also based on the precise position of the eye during calibration, but without head stabilization, or some mechanism of tracking the head accurately in 3D space, 0.5° of vision seems to be the highest possible accuracy with current technology. Section 2.1 covered aspects of the human eye such as the anatomy and the inherent noise from eye movements during fixations (i.e., tremor, drift and micro saccades). The section further discussed how photoreceptor cells inside the

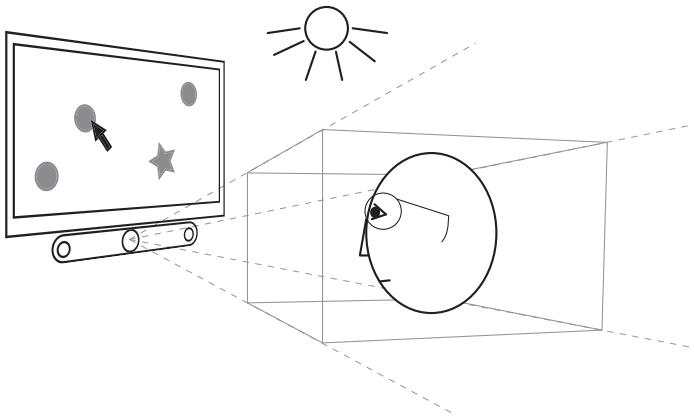


Figure 3.1: *A typical setting with a user sitting in front of an interactive eye tracker. Some of the factors that have an effect on the quality of the eye tracking include various kind of noise and system latency [49]. Some concrete examples could include: head-box size, hardware quality, inaccurate eye models, eye jitter, reflections from glasses and proprietary software.*

eye are not evenly distributed with the largest amount of photoreceptor cells located on the fovea. From the chapter it became evident that the physiological constraint on the spatial extent of high acuity vision is the size of the fovea, which covers a central area of about 1° - 2° of vision. Essentially, this means that the fovea sets the physiological lower limit for pointing with the eyes, since the eyes do not have to reposition themselves when fixating on new targets within a 1° distance from the initial fixation. Although the projection does not fall on the center of the fovea, information about the target can still be processed within the parafoveal region since it is registered at high acuity and with color within this extent.

Even if technological advances improves the accuracy of an eye tracking system down to pixel-perfect precision, it is unlikely that it will have any practical consequences in an HCI task. Whereas basic research on the relationship between the eye and the brain often requires measurement of very tiny movements, in HCI, at least when pointing, these movements are not representative of a change of intended target, but rather keep the target within central vision, and so are less critical to interaction. The human fovea seems to set the natural limitation for the eye accuracy during pointing. Drewes

[25] notes how this is analog to finger pointing where the natural pointing precision is given by the size of the fingertip. However, it is still possible to interact with smaller targets (e.g., on-screen numeric key on mobile phone) but larger keys are, by far, preferred by users when interacting for longer periods of time.

3.2 Calibration

Video-oculographic eye tracking requires a new calibration to calculate the gaze direction for each user of the system. This is required since the angular offset between the visual and the optical axis differs from subject to subject. The procedure normally consists of looking at a series of equally distributed points on the screen (usually 5, 9 or 16 points distributed over the screen) at a $50-70$ cm distance between the eye and the screen. The calibration points are shown one at a time and the subject looks at the presented points. This process is often repeated for various reasons (that will be covered later) and it is considered to be one of the most tedious and often annoying aspects of many eye-tracking systems [170]. Introducing individual user profiles which are saved by the eye tracker could to potentially solve this problem although a recalibration would be necessary if the setup changes too much. The movement of the pupils and the corneal reflections during a calibration can be seen in Figure 3.2. The series of images are recorded by the camera of the eye tracker and used to generate a model of the user's eye movements. A sequence of images are analyzed by the computer and associated with the corresponding screen coordinates and the point-of-gaze can then be measured. Figure 3.3 shows four inner targets from a 4×4 calibration on the ITU Gaze Tracker. Here, the grey circles represent the stimuli and the small dots illustrate the sampled gaze positions. The dotted line connected to the stimuli center represents mean and standard deviation. The spread of the individual samples is a result of the physiological inaccuracies, noise and error factors covered in Section 3.3.

The quality of an eye tracker is often indicated by infrequent re-calibrations and the ability to run under the different conditions in a wide range of user scenarios. Even the background color during calibration has an effect on accuracy and precision, since light levels affect the pupil size. Calibrating with a too bright (e.g., white) background on a high-end system may result

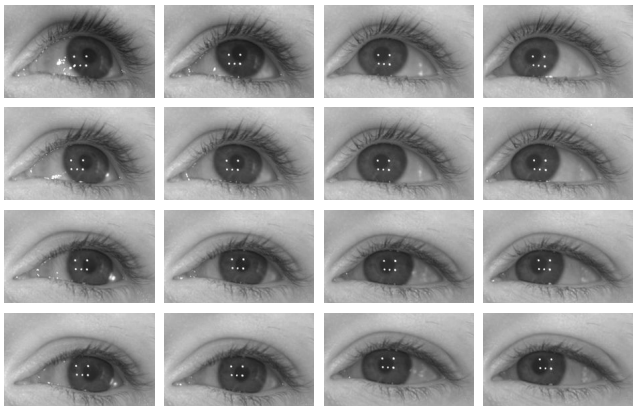


Figure 3.2: *Sequence of 16 calibration images samples taken from a* 4×4 *stimulus grid. The images illustrate one of many recorded pupil positions while a person is gazing at the individual targets. The relationship between the pupil and five corneal reflection are in this case used to generate a model of the specific user's eye movements.*

in an offset of 1.5° [27]. The success of a calibration is important for the user experience and general use of the system. Commonly, if the user, caregiver or an assistant is not satisfied with the result of the calibration the routine is repeated. This essentially means that the accuracy and precision of the system depends on a successful calibration. Some systems, like the ITU Gaze Tracker, provide a simple feedback consisting of $1 - 5$ stars to the user to indicate the quality of the calibration.

3.2.1 Offsets

In gaze interaction, a displacement between the actual point of gaze and the estimated gaze is called an offset and may occur as a result of a poor calibration. Even the best eye trackers have some offset after a calibration and if a inexperienced user expect his gaze to be aligned with the gaze-controlled cursor 'chasing' can occur [66]. This means that a user's gaze will never be able to fully look-at/acquire the mouse cursor with a corrective gaze movement since the offset will move the cursor accordingly and thus a drifting sensation will occur. However, with practice, some users are able to ignore the offset by looking a little off in order to successfully acquire objects of interest [98]. So long as any small offset is continuous and predictable, it seems

Figure 3.3: *Four inner stimuli and their resulting estimated gaze coordinates from a* 4×4 *calibration performed on the ITU Gaze Tracker with a Webcamera. The calibration was deliberately inaccurate (the user did not look directly at calibration points) and yielded an rather low accuracy of* $2.2°$.

possible to achieve a smooth interaction with some training. Inconsistent or unpredictable offset will pose more of a problem, of course.

3.3 Types of Noise

In eye tracking, there are several possible sources of noise. Holmqvist et al. [49] note how the different noise contributions in eye tracking can be seen as a combination of *system-inherent noise*, *oculomotor noise*, *environmental noise* and, finally, *optic artifacts*.

- **System-inherent noise** is the spatial variance that comes from the eye tracker itself. The most precise and reliable way of measuring system-inherent noise is with an artificial eye positioned in front of an eye tracker. This can be easily achieved for systems which use dark pupil detection, however, it is more difficult to produce an artificial eye which will 'trick' bright pupil detection systems, since this implies mimicking the reflective properties of the retina with an IR reflective surface. Unless this is achieved, the system will not accept the model as a real eye, and will not record movements from them.

- **Oculomotor noise** or Jitter, is traditionally referring to the tremor of eye movements during fixations, micro saccades and drift. During jitter, a gaze-controlled cursor may appear unsteady and distract a user. Usually, in gaze interaction, this is avoided by smoothing (i.e., low-pass filtering) the estimated gaze positions over time (for a detailed description of the fast eye movements during fixations see Section 2.1.2). Drift occurs in most eye-tracking systems over time. Usually, it becomes visible when the estimated gaze position steadily moves away from the true gaze position. Large head movements may cause severe drifts. The simplest solution to drifting is to preform a recalibration. Stampe [156] suggested a method based on heuristic filtering by measuring the drift and compensating for it in real-time. The method involves presenting a target to the user and then measuring the difference between the user's gaze and the target location and thus compensate for the amount of drift. Stampe suggest how the method should be employed between blocks or trials in order to 'dramatically' improve the stability of the system. Some systems, like the Quickglance, offer an easy way to do this by looking at a stimulus in the center on the screen whenever the user feels that the system is too inaccurate. Later, Stampe and Reingold [157], followed up the work and presented a novel method based on low-pass filtering called *dynamic re-centering*. The filter tracks the component of drifting from target fixation error and thereby helps to average variations in target fixation out over time. Some eye trackers from commercial manufacturers (such as Tobii) include a technique to prevent drifting but this will only work when target locations are known or can be assumed with high probability. Using binocular tracking they are able to use data from both eyes and average the data to decrease the effect of drifting. The effect of averaging helps to prolong a good calibration and saves the user from recalibrating.

- **Environmental noise** is variations in the position of the gaze signal that is caused by one or more external sources. Most eye tracking systems are tolerant of a limited range of head movements but for some users e.g., people with cerebral palsy, who suffer from severe involuntary head, body and or eye movements, it may be difficult to calibrate. In some cases it may be close to impossible or very inaccurate and if the calibration fails, some systems offer a default calibration [98]. Charlier et al. [19] have successfully applied special filtering techniques for eye

movement disorders. Equally, mounting an eye tracking system on a wheelchair may also introduce additional noise during locomotion. Finally, light disturbances such as daylight, ambient light and infra red light falls under this category.

- **Optic artifacts** refer to high-speed movements, often with a large amplitude that cannot originate from any human. Factors that contribute to this include glasses and contact lenses and play a role in the quality of a successful calibration. Normal glasses, glasses with progressive lenses and some contact lenses generate additional reflections from the IR light sources that may complicate or even hinder a calibration. Other factors include: mascara, eyelids, and eyelashes (that may potentially cover a part of the pupil). Finally, the contact lenses may be displaced over time, which causes a reduced quality of the tracking [98]. Many of the mentioned problems can be avoided or minimized with simple initiatives such as correct ambient light and positioning the camera to get a clear view of the user's eyes.

Removing all noise from a system seems like an impossible task since it is a high dimensional problem, which includes a lot of unknown factors. Holmqvist et al. [50] list six factors that have an effect on the quality of the eye tracking data:

- **Participants** have a large variation of the physiologies, varying neurology and physiology.

- **Operators** have different levels of skill, which have an effect on the collected data.

- **Task** may force the users to move out of the camera box or blink too much.

- **Recordings** of the eyes are affected by the environment in which they are carried out.

- **Relative positioning** between eye camera, participant and stimuli.

- **System** (the eye tracker) has a large impact on the quality of the recorded data.

For interactive eye-tracking systems a large contributor to noise in a VOG system seems to be the eye camera. The amount of information in each pixel is of importance in the feature extraction, which in turn propagates to the later stages of processing (See Figure 3.4). The gaze estimation stage also relevant since the internal models (i.e., eye model and geometric model) have a direct effect on the system's accuracy. Chapter 4 evaluates performance measures for gaze interaction and presents a pilot study where accuracy and precision is measured on three different interactive eye-tracking systems.

3.4 The Path From Eye Movements to Application

This section presents an overview of the data flow from the raw eye movements up to the point when is it made available for the gaze-controlled applications. System latency or delay is not uncommon during this process since system hardware sets a natural limitation to the processing speed. The delay is the processing time between the physical eye movement and the calculated gaze point. In real-time eye control (the main focus of this thesis), it is the point in time at which the gaze location is made available to other processes, and results in a movement of, for example, the eye-mouse. In offline processing, this is the difference between the movement of the eye and the time reported in the recorded/logged eye data. The delay can be attributed to hardware limitations, e.g. camera frame rate (i.e., temporal resolution) and image and feature processing such as processing of complex eye models on the computer. Sometimes, even a significant delay may (surprisingly) not be of any great annoyance to the user during discrete interaction, if it is consistent, although Chapter 8 discusses some possible negative consequences of a delay during interaction in a point-and-select task. Furthermore, the delay is a measure that is often not reported by the manufacturers.

All commercial eye tracking systems are shipped as fully integrated systems and understanding the different processes inside of them can be difficult since the codebase, algorithms and models are proprietary (i.e., business secrets). The eye tracker model in Figure 3.4 illustrates the basic stages in for a VOG-based system as it is found in the ITU Gaze Tracker. Although most basic functionality of a system can be inferred through measurement, other parts

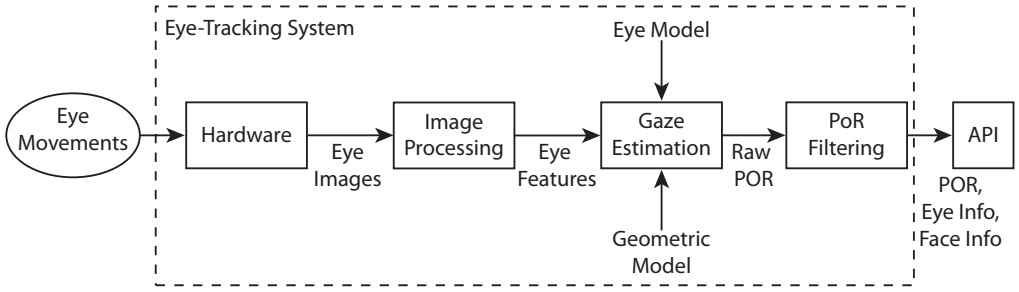


Figure 3.4: *A model of a VOG eye tracking system. Eye movements are recorded and manipulated by several stages until the point-of-regard and tracking related information are delivered to the API. Everything within the dashed line is essentially a 'black box' but the individual stages and their functionality can to some extend be guessed.*

of the system and their function are essentially a 'black box'. Furthermore, most manufactures (of interactive systems) perform initial filtering of the eye data before making the data ready for the API (application programming interface). The API constitutes the stage to which the PoR and other tracking related information is made available to the gaze applications. Processing data in the individual stages of the model takes time, which causes a delay of a system. The individual functional stages in the eye-tracking model can be summarized as follows:

- **Eye movements** constitutes the lowest functional stage in the process eye tracking and are inherently noisy (see Section 2.1).

- **Hardware** comprises the stage of sensors such as industrial cameras, light sources and casing. The temporal resolution of the cameras are often running in the range from $30Hz$ to $2000Hz$ in high end research systems, which sets a natural update limit for the later stages of eye-data processing and the camera resolution sets the limit for the spatial precision. Finally, environmental noise and optic artifacts reduce the quality of the eye image before it passes the recorded eye images to the image-processing stage. The difference in quality between industrial cameras and the cheap cameras employed in low-cost eye tracking is significant. This is shown in Figure 3.5 where a high and a low-resolution eye image are depicted. Without a modified lens the low-cost camera

needs to be placed relatively close to the user's eye to achieve viable image quality.

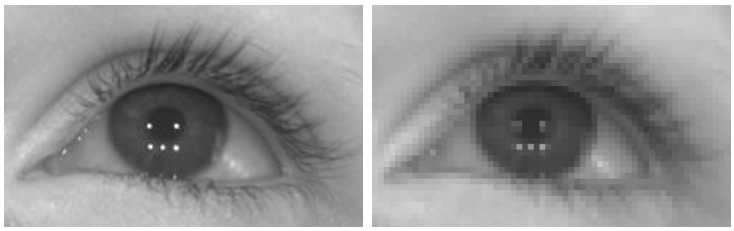


Figure 3.5: *The difference between high (left) and low (right) resolution images of an eye. A high-resolution eye image allows for a better extraction of eye features in the image processing stage.*

- **Image Processing** stage describes analyzing the eye images and locating features such as the pupil, iris and eye corners. The features are then passed to the gaze estimation model for further processing.

- **Gaze Model** represents the stage where the eye features are fitted to the eye model (constructed from the calibration) and the geometric model in order to estimate the raw PoR. In this stage, the raw PoR may be noisy as a result of the individual contributions of the prior processing stages. Furthermore, the data stream will have interruptions, for instance if the eyes have been closed during blinking.

- **Point-Of-Regard Filtering** receives the raw PoR from the gaze-model stage and prepares the data for the API. Smoothing the data is not uncommon in this stage and by averaging an optimal number of successive values it is possible to derive a good estimate of the true value (i.e., PoR) [48]. Alternatively, mathematical filters are often employed in this process and in general, an optimal filter will minimize the difference between the estimate and the true value. The filters used to measure the average can vary, but some common examples include: *boxcar filter* and *weighted average*. The boxcar filter is the easiest to implement and works by averaging n samples of detected sensor readings. The filter requires a buffer of samples and will consequently *lag* behind the true value of the changing signal. The weighted average is based on both new observations and previous estimates. The filter

requires no buffer of previous samples and is easy to implement. The drawback of the filter is that a quickly moving eye with many outliers in the recorded signal will result in erratic changes when the signal is smoothed.

Heuristic methods can be combined with a smoothing filter for 'better' performance. Since Smoothing should only occur during fixations and not during saccades, the filtering stage must identify and classify eye movements in order to apply the smoothing filter optimally (see Section 3.5 for a presentation of real-time detections techniques). The smoothing filter may additionally be combined with other strategies before presenting the PoR to the API. For example, if a new PoR coordinate is within a threshold distance to the earlier PoR coordinate the ERICA eye tracker does not report this to the API [25]. Anther strategy is to account for optic artifacts that are represented as non-human generated PoRs, for example, if high velocities between two successive PoRs are detected or if the POR falls significantly outside the screen regions. Here, the simplest solution is to pass the last 'good' PoR coordinate to the API.

Some interactive eye tracking systems allow directing the raw point-of-regard directly to the API through an identity filter. The identity filter will result in a responsive but noisy gaze-controlled cursor and may have a negative effect during an interaction task. For example, San Agustin et al. [136] detected significant noise levels after using the raw PoR from a SMI IViewX RED system during a performance evaluation. Tobii, for example, provides the raw PoR for their analysis tool but not for the developer API. However, they provide a slider in their dashboard where the user is able to adjust the cursor's responsiveness, anyhow some filtering is always enabled.

- **API** is the final stage of the eye-tracker model. This stage offers an entry point of the eye tracker that can be accessed by one or more programming languages. The type of information in this stage differs from system to system but PoR, pupil size and face location are commonly available. Furthermore, the API allows the user to control tracker specific settings such as starting/stopping a calibration, number of calibrations points, and sometimes signal smoothing. Most commer-

cial systems offer a dashboard on top of their API and users are able to control the settings of their system without any programming skills.

Once a PoR coordinate is calculated it can be accessed through the API of the eye tracker. Most interactive eye tracking systems offer dedicated software such as an Internet browser, communication tools (see Chapter 6) and a cursor control that allows a user to perform point-and-select operations (see Chapter 8). Figure 3.6 illustrates how the information provided by the eye tracker's API could be used by dedicated software. Developing applica-

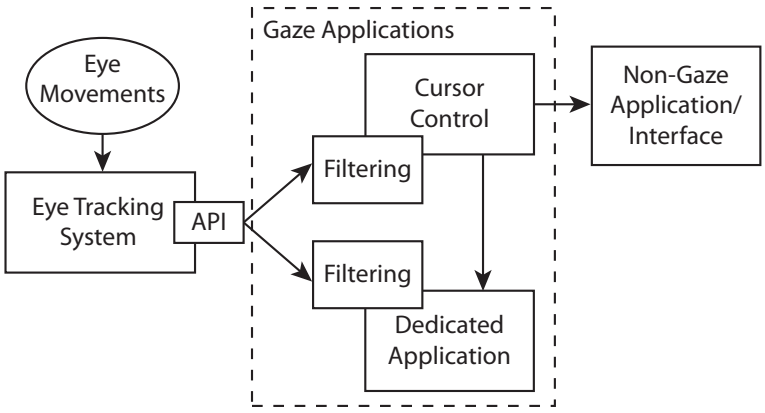


Figure 3.6: *Model of the gaze applications. Eye movements are detected by the eye-tracking system and can thus be used by gaze applications on the computer.*

tions on any API limits the software only to run on eye trackers from that specific brand. This may not always be an advantage if the software is suppose to run on multiple eye tracking systems. No standard on reporting data in the API exists so only solution to this problem is to have the individual systems direct the cursor on the screen. Developing applications on top of a PoR-directed cursor avoids this problem although additional filtering may be needed to detect the types of eye movement. This solution is widely used, for example, in generic communication tools e.g., Dasher and GazeTalk [173, 43].

The cursor control is, by far, the most critical software for the interactive eye tracker since it allows a user to interact with non-gaze applications or the operating system's interface. This issue is discussed in Chapter 8 where new tools for small-target selection based on zooming are evaluated.

3.5 Real-Time Fixation Detection Techniques

Eye movement detection is an important process for most gaze-based applications. Dwell-based systems use algorithms that divide eye movements into fixations and saccades in order to infer a user's selection intent. Fixations occur when a user looks at an object of interest and are characterized by reduced eye movement within a limited spatial region for a minimum amount of time. In contrast, saccades occur when a user is moving her eyes from object to object and are characterized by fast eye movements extending over (relatively) larger spatial regions outside the fixation threshold. Fixation-detection algorithms generally use velocity, spatial dispersion, or a combination of both to identify fixations (see Section 3.5).

Figure 3.7 shows a real-life recording of a subject looking a eight different stimuli (light grey) presented randomly on the screen and the resulting gaze signal (dark grey). A hybrid fixation-detection algorithm is applied to the gaze signal to detect the user's fixation (black). An inspection of the figure reveals that gaze lags after the stimuli. The lag shows an average difference of about 370 ms between the stimuli presentation and the corresponding gaze and is a result of the time taken to perform the eye movements plus the end-to-end delay of the current eye tracker. In the example, a minimum-fixation duration of 100 ms is applied to the hybrid fixation-detection algorithm.

Salvucci and Goldberg [133] presented a taxonomy of fixation detection algorithms. The models were based on spatial and temporal characteristics, where spatial refers to the arrangement in space and temporal refers to the measurement of time. Furthermore, they classified the algorithms according to five different criteria: velocity, dispersion, area, duration sensitivity and local adaptiveness. In the following sections, some of the fixation detection algorithms that can be implemented and used in real-time will be presented. Although Salvucci and Goldberg choose not to present any hybrid models (i.e., combining the individual models) a hybrid model will also be presented in Section 3.5. Their taxonomy also addresses some non real-time algorithms such as *hidden Markov models* and *minimum spanning trees*, but these will not be presented here.

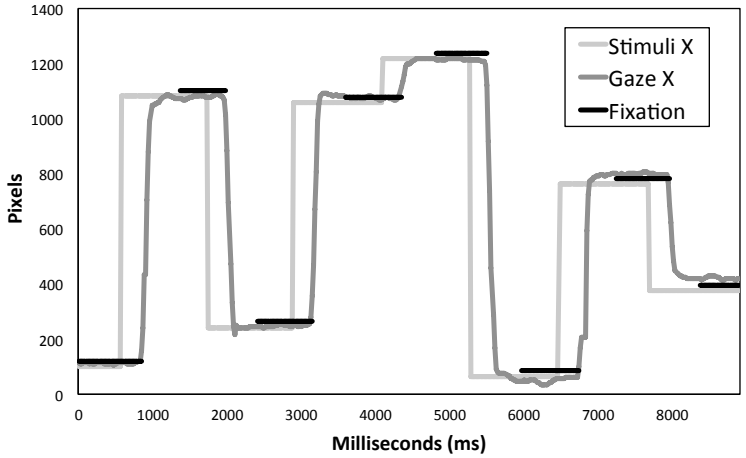


Figure 3.7: *Eight stimuli (light grey) and the resulting gaze signal (dark grey) recorded with an Eye Follower from LC Technologies running at 60 Hz for each eye, offset for 120Hz frame rate. This system allows access to the raw data. Here, corresponding fixations (black) in the gaze signal have been detected by a hybrid fixation-detection algorithm designed for investigative purposes by the author, bypassing the system's own event detection algorithm.*

Velocity-Based Algorithm. The velocity-based algorithm is considered one of the simplest to understand and implement. The velocity-based method separates fixation and saccade points based on their point-to-point velocities. Salvucci and Goldberg argue that the velocity profiles of saccadic eye movements show two distributions of velocities. First, the low velocities for fixations and second, high velocities for saccades.

The algorithm works by calculating the point-to-point velocities of the current and the previous point in the protocol. The computed velocities will then be classified based on a threshold; where a low value will be classified as a fixation point and a high value will be classified as a saccade. The process will then collapse consecutive fixation points into fixation groups and discard saccadic points. The velocity-based algorithm requires only one parameter, the velocity threshold. In gaze interaction, the point-to-point velocity can be estimated by means of the angular velocity (usually computed from an average distance between user and computer of 50 cm). In the literature, the threshold seems to be based on equipment and exploratory data analysis rather than standard values. Sen and Megaw [140] reported a threshold of $20°/sec$ in their study and Duchowsky et al. [29] used a threshold of $130°/sec$ for their 3D virtual reality study.

Dispersion-Based Algorithm. A dispersion-based algorithm utilizes the fact that spatial information from gaze points tend to cluster close together during fixation. The algorithm identifies fixations as groups of consecutive points with a least mean distance between points under some threshold [31]. Such an algorithm checks for potential fixations in the consecutive data points. The dispersion in the window is computed by summing the differences between the points' maximum and minimum x and y values. Dispersion-based algorithms require two parameters, the dispersion threshold and the duration threshold.

In gaze interaction the dispersion threshold can be set to include 0.5° to 1° of visual angle if the distance from eye to screen is known. The standard threshold reported is a minimum duration of 100ms with a least mean distance under 1° of vision. Alternatively, the dispersion threshold can be based on explorative data studies. Dependent on task and processing demands, the duration threshold is typically set to a value between 100 and 200 ms [177].

Area-Based Algorithm. The area-of-interest algorithm identifies only those fixations that occur within pre-specified target areas, in contrast to previously mentioned algorithms that are able to identify fixations at any location of the visual field.

The algorithm labels data points within a target area as a fixation point and labels data points outside the target area as saccades. Consecutive fixation points will be collapsed into fixation groups and saccade points will be discarded. If the fixation group falls above a given duration threshold the fixation group will be transformed into a fixation tuple. The tradeoff with this algorithm is that it reduces noise significantly but full information about the underlying objects is needed in order to define the regions-of-interest. This problem will be discussed in further detail in Chapter 8.

Hybrid Algorithm. A simple yet powerful hybrid algorithm is the combination of the velocity-based algorithm and the dispersion-based algorithm. Increased robustness and accuracy in the classification is obtained by combining characteristics from the spatial and temporal domain. The simple hybrid algorithm works in two phases; (1) one phase with a velocity based algorithm operating in the spatial domain that classifies the point-to-point velocities into fixation groups based on a fixed threshold and (2) a second phase that is based on a dispersion technique, which operates in the temporal domain. The second stage evaluates the group of potential fixation candidates based on a moving window. In order to be considered part of a fixation a value needs to fall below the dispersion threshold and above the duration threshold.

In a spatially-based method a series of fixation candidates will appear in the raw data stream if the point-to-point velocities are shifting around the fixed threshold. The robustness of the algorithm comes into play when combining the spatial and temporal constraints since the temporal characteristics will remove the short fixation candidates [133]. The hybrid algorithm requires that three parameters are specified; one for the spatial and two for the temporal characteristics: the fixed velocity threshold, the dispersion threshold and the duration threshold.

Smooth Pursuit Detection. Some of the previous attempts to separate smooth pursuit movements and saccades have been suggested by Sauter et

al. [138]. They used a Kalman filter to predict the states of a future sample by employing an χ^2 test on the error of the recorded and the predicted eye velocities. By evaluating the χ^2 value to a pre-specified threshold, smooth pursuit is detected if the χ^2 value falls below the threshold and saccade if above. For this specific algorithm, there is no ability to separate fixations from the smooth pursuit movements. A similar technique with a Kalman filter has been suggested by Komogortsev and Khan [81]. They also utilize a χ^2 test to classify eye movements but as a contrast to earlier experiment by [138], their algorithm was able to detect fixations with simple spatial and temporal modifications. Fixations were detected when observations fell below a fixed threshold of $0.5°/s$ for a minimum duration of 100 ms. The smooth pursuit was detected if velocities did not exceeded $140°/s$ and the algorithm did not observe any fixation.

San Agustin [134] presented a novel smooth pursuit algorithm to classify a sequence of gaze coordinates. The extended eye movement detection algorithm consisted of two stages: a detection part and a signal smoothing part. The novelty in the work by San Agustin was to detect smooth pursuits by investigating the amount of change in the distribution of angles between consecutive points. This was done by fitting lines between points in a pre-specified window. If the lines align, the distribution of the angles will be low and thus be a smooth pursuit and if they a high there is a fixation. With this novel eye movement detection method San Agustin [134] detected 77% of the fixations successfully and the relatively low detection rate was attributed to a delay between stimuli and gaze coordinates. In the case of smooth pursuit, the overall mean of correctly classified samples was 44% and the ratio of properly detected smooth pursuit decreased when as the target velocity increased.

San Agustin reported what he called a smoothing delay, which was attributed to a filtering artifact in the eye tracker (Tobii 1750 with highest cursor responsiveness). Smooth pursuit movement was typically mistaken for fixations at low velocities by the system. When the smooth pursuit movement was regarded as a series of fixations, the cursor position was smoothed and lagged behind the presented stimuli. The cursor position was smoothed until the dispersion on the Tobii system was too large and was followed by a correction, which produced saw-like shapes of the gaze path.

3.6 Summary

This chapter has presented factors that have an effect on the quality of eye tracking in interactive systems. The accuracy of the eye (in terms of what the eye is fixating on, though perhaps not in terms of how much the eye is moving) sets a natural limitation due to the size of the fovea. Even though it is possible to obtain better accuracy is it not likely that it will have any practical consequences in an HCI task, when the task is to select an onscreen target. A VOG-based system requires a calibration to calculate the PoR and several factors may have an effect on the quality of the calibration, which may result in a jittery cursor and large offsets. The different types of noise in VOG eye tracking include system-inherent noise, oculomotor noise, environmental noise and optic artifacts. Several methods exist to reduce the noise but the easiest solution is to perform a recalibration. The camera was described as the largest contributor to noise in a VOG eye tracking system and the internal models are reflected in the accuracy.

Two models were employed to describe the data flow from physical eye movements to application: one for the eye tracker and another gaze applications. The eye tracker model described the internal functioning stage of an interactive system and the model explained the task for individual stages and their contribution to system noise and lag. Most focus was on the PoR-filtering stage and the consequence of manipulating the raw PoR before passing it to the API. The gaze application model showed how an additional filtering of the PoR from API is needed before it can be used in an application. The work by Salvucci and Goldberg [133] presents several methods for detecting fixations based on spatial and temporal characteristics and one or more of these techniques are built-in to the tracker to manipulate the gaze signal.

The data flow from physical eye movement to gaze application is relatively long. The PoR coordinates are manipulated in several levels before presenting them to the user either in the API or through a dedicated application. Finally, the end-to-end delay may have a serious effect on the interaction since the PoR will suffer from a lag. This lag is simply a result of the sequential processing of the individual stages, which can be demanding for the CPU of a traditional computer.

Eye tracking manufacturers may be too conservative when designing for gaze interaction. It may be enough to perform simple interactions strategies, which may be acceptable for most users but it leaves too little freedom-to-operate for developers and researchers. This is evident in smooth pursuit detection, where the internal eye movement detection results in an unexpected behavior with an adverse effect on the detection.

The following chapter focusses on performance measures for gaze interaction and discusses a new initiative to develop a standard for measuring accuracy and precision of eye trackers. An experiment demonstrates an evaluation of three systems based on measures suggested in the literature.

4

Performance Measures for Gaze Interaction

When remote eye tracking systems first came on the market, manufacturers began to report on data quality in their systems, generally in terms of accuracy and sampling frequency. However, the work was carried out independently, without any means of comparing results across manufactures. Johnson et al. demonstrated this problem in [73] where they found disagreement in the accuracy between the measured values and the reported values from manufacturers.

In mid-2010 COGAIN initiated work to develop a standard method of measuring the accuracy and precision of eye trackers. Unfortunately, creating a new standard is a slow process. The standardization work by the COGAIN Association has not gone unnoticed by the eye tracking manufactures since several of those are now becoming involved in the process. Several of the established eye tracking manufactures whose systems are used by eye movement researchers and neurological researchers have had their own measures for accuracy and precision for many years, since manufacturers needed to report on the quality of data produced by their systems, both for users and for in-house benchmarking. In mid-2011, Tobii Industries, who are the market leaders on remote interactive eye trackers, presented a report on measuring accuracy and precision of remote eye trackers to the eye tracking community[1]. Some manufacturers report the methods they use to measure data quality, others do not. However, the ability to compare systems or even individual calibrations for accuracy and precision is: (1) an important step in ensuring

[1] <http://www.tobii.com/analysis-and-research/global/about-tobii/eye-tracking/test-method/>

a product's competitiveness, (2) a means of presenting evidence, rather than arbitrary values, to potential customers or researchers who need to know about accuracy and precision in general, and (3) an advantage in preparing in house for the new COGAIN standard when it reaches completion.

The first study on applying the Fitts' law paradigm to eye movements was in the work by Ware and Mikaelian [175] in 1987. They had interest in making the data comparable to other input devices. Although their results followed Fitts' law, they debated the applicability of the paradigm for measuring the performance of eye movements:

> *This is a theoretical construct designed to account for eye-hand coordination. We use it here only as a convenient way of summarizing the results, not because we wish to make any theoretical claims [175, p. 186].*

Several other studies cast doubt on the validity of using Fitts' paradigm for the eyes e.g., Zhai et al. [181], Siebert and Jacob [146] and Ashmore et al. [2]. Like Ware and Mikaelian, Miniotas (2000) demonstrated that Fitts' framework applied well for eye movements [108]. Using Fitts' paradigm was further supported by the study of Zhang and MacKenzie where they evaluated the eye as a pointing device following the methodology presented in the ISO 9241-9 standard [182].

This chapter introduces the ongoing discussion on the standardization of eye tracking systems and presents an evaluation of the performance of a webcam version of the ITU Gaze Tracker and two commercial eye tracking systems, based on various parameters suggested in the literature such as spatial error and a standard ISO 9241-9 evaluation. The evaluation presented in this chapter combines currently used accuracy and precision measures with a standard target-acquisition task known from the methodology described by the ISO 9241-9 standard for non-keyboard input devices [57]. The idea of measuring spatial error is not new to eye tracking, in fact, such measures have been around as long as eye tracking has been around (see e.g., [106, 53, 165, 184]). However, while the need to assess data quality is immediate and apparent for basic research in eye movements, somehow as eye trackers for gaze control entered the public domain, and left the rather small domain of academic research, industry has not always been concerned about these measures or

perhaps did not understand the relevance for interaction purposes.

Eye movement researchers and neurological researchers may be satisfied with values of spatial accuracy, precision and temporal precision, but evaluating an interactive eye tracking system involves more that these measures since it is not only used passively in attention studies, rather it is used as an input device. In gaze based interaction, the quality of the raw data is not as important as the quality of the interactive experience. These two things are not always analogous. When the designer of an interactive interface knows the limitations in terms of data quality, he can design for a good interactive experience within the constraints of accuracy and precision regarding button or area of interest size and position.

4.1 Spatial Error

The performance of a sensor is typically measured in spatial error, which is *accuracy* and *precision*. The accuracy refers to the degree to which the sensor readings represent the true value of what is measured and the precision (also known as resolution) refers to the extent to which successive readings of the same physical phenomenon agree in value.

The target analogy can be used to explain the difference between accuracy and precision of samples in a target comparison. Accuracy describes the distance between recorded samples and the actual point of regard, so that samples that fall close to the actual point of regard (typically tested using a small point presented onscreen) are more accurate. Precision is indicated by the size of the sample cluster around the point, and the tighter the samples are grouped, even if they are not near the center of the stimuli, the more precise they are considered to be. A tight cluster away from the gaze target indicates poor accuracy and high precision. This can be explored further in an example with simulated gaze data in Figure 4.1.

4.1.1 Accuracy.

Researchers working with eye movements as a clinical or behavioral measure rely on high accuracy and precision. For example, researchers in reading behavior relies on the accuracy of an eye tracker to tell them which word or

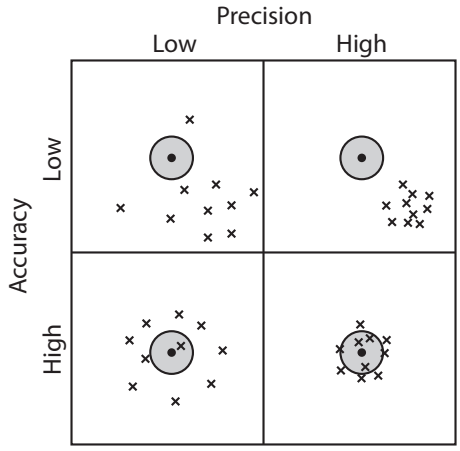


Figure 4.1: 2×2 *matrix of accuracy and precision in a "low" and "high" scenario. In each cell a the true gaze position is presented (marked as solid circle with a black outline) and ten samples of measured gaze position (depicted as crosshairs).*

even letter a person is looking at. The theoretical limit of accuracy is the size of the fovea (covers an area of about 1° - 2° of the visual field). However, it is possible to obtain a higher accuracy ($> 0.5°$) since the accuracy is also based on the precise position of the eye during calibration, which corresponds to half a centimeter at a 70 cm distance. This pertains to accuracy of the stable eye, however, regardless of the size of the fovea, accuracy can also be discussed in terms of the moving eye, so that any movement, however small, is either measured accurately or missed entirely as a consequence of the eye tracker's performance.

Accuracy A_{deg} is defined as the average angular distance, θ_i (measured in degrees of visual angle) between n fixation locations and the corresponding fixation targets (Equation 4.1).

$$\theta_{offset} = \frac{1}{n} \sum_{i=1}^{n} \theta_i \tag{4.1}$$

There seems to be some disagreement with regards to how to measure fixations and where on the screen to measure accuracy [49]. It is quite common for gaze position in the centre of the screen to be more accurate than in the periphery, possibly as a result of the fact that the person is seated op-

posite the centre of the screen and the eye model does not take account of the fact that the eye is further away from the periphery of the screen. Eye models generally assume equal distance of the eye to all parts of the screen, but in reality, this would imply a concave screen surface. This is likely to be one contribution, another is that the eye model works best when the eye is captured from the front. When looking at corners, some eye models will fail to accurately infer gaze position, because the pupil is an ellipse in the eye image. Section 3.5 presents a taxonomy of fixation detection algorithms based on spatial and temporal characteristics.

4.1.2 Precision.

The ability to measure the spatial precision of an eye tracking system is important for measuring fixations and saccades correctly, and for inferring viable target size for gaze interaction. High spatial precision is also needed for measuring small intra-fixational eye movements such as tremor, drift and microsaccades [49]. Precision of an eye tracking system is calculated from data samples recorded when the eyes fixate on a stationary target, which means that the variation of samples originating from eye movements is almost excluded.

Calculation of spatial precision can be done in two ways: dispersion, measured in standard deviation or root mean square (RMS), measured in degrees of visual angle. The dispersion measure can be calculated by computing how dispersed samples are in the x and x direction as seen in Equation 4.2.

$$s_x = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(x_i - \bar{x})^2}, s_y = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \bar{y})^2} \tag{4.2}$$

Spatial precision is suggested to be calculated as the angular distance between successive samples (x_i, y_i) to (x_{i+1}, y_{i+1}) and computed in RMS as seen in Equation 4.3.

$$\theta_{RMS} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}\theta_i^2} = \sqrt{\frac{\theta_1^2 + \theta_2^2 + ... + \theta_n^2}{n}} \tag{4.3}$$

The RMS seem to be the common choice of eye-tracking manufactures and practitioners in order to quantify the precision. Eye trackers with poor pre-

cision have RMS about 1° and some high-end eye trackers report better than 0.10°. Unfortunately, most commercial eye tracking systems used for interaction purposes do not report precision at all [49].

4.2 Performance Evaluation

Predicting and modeling human performance of interactive tools have interested researchers in HCI since it allows to match the powers of the computer to the users abilities.

4.2.1 Fitts' Law

Fitts presented his law on human psychomotor behavior in 1954 [34] and introduced the idea that the human motor system could be considered analogous to a communication channel of limited capacity when carrying out a movement task [94]. The model was originally derived from Shannon's Theorem on information transmission. Fitts called this the capacity index of performance (IP) and found that it could be calculated as the ratio between index of difficulty (ID) of a motor task and the movement time (MT) required for the given task (see Equation 4.4).

$$IP = \frac{ID}{MT} \tag{4.4}$$

The index of difficulty (ID) of a task is measured in bits and depends on the distance to the target, or amplitude (A) and the width of the target measured along the axis of movement (W). Units of bits are obtained by applying the base 2 logarithm to the ratio of A and W. MacKenzie [93] proposed an update to the original formula by Fitts that followed the original analogy by Shannon more closely (see Equation 4.5).

$$ID = \log_2\left(\frac{A}{W} + 1\right) \tag{4.5}$$

MacKenzie claims that empirical data provide a better correlation with his formula. However, the update to the ID formula is still debated by researchers. Drewes [25] argues whether a better correlation proves formula and notes how this question is left for the statisticians to show.

Calculating the throughput in the ISO 9241-9 standard is based on the effective target width W_e and the effective distance D_e, which are used to calculate the effective index of difficulty ID_e following Equation 4.6. W_e is calculated by projecting the selection points onto the task axis and then the distance x from the projection to the center is calculated. Since the underlying information theory assumes the signal is influenced by white Gaussian noise the standard deviation of x is multiplied with 4.133 in order to obtain W_e. This ensures that 96% of the hitpoints fall within the target.

$$ID_e = \log_2\left(\frac{D_e}{W_e} + 1\right), W_e = 4.133 \cdot SD_x \tag{4.6}$$

Throughput is measured in *bps* and is calculated as the relationship between effective index of difficulty ID_e and movement time MT (Equation 4.7) [182].

$$Throughput = \frac{ID_e}{MT} \tag{4.7}$$

4.2.2 Target Acquisition Task

A target acquisition task can be used to measure the performance of a pointing device. The ISO 9241-9 standard suggests an update to the old physical configuration of the Fitts' law model (termed the Fitts paradigm), which involved a one-dimensional (horizontal) movement task over a range of amplitudes and a set of amplitudes [155]. The methodology described by the ISO 9241-9 standard for non-keyboard input devices suggest a circular array of targets in 16 different locations. The first time an eye tracking device was evaluated under the Fitt's framework for gaze interaction was in 1987, in a study by Ware and Mikaelian [175]. In their study, completion time was evaluated as the only performance measure and the results were compared to the performance of a mouse. 20 years later Zhang and MacKenzie [182] did the first Fitts' law experiment on an eye tracking system under the ISO 9241-9 framework and quantified the performance through throughput and error rate.

Two different schemes can be employed in the multi-directional tapping task: serial and discrete (see Figure 4.2). In the serial task, the user taps back-and-forth between targets (see e.g., Zhang and MacKenzie [182] and San Agustin et al. [136]) and in the discrete task, the user is repeatedly activating the

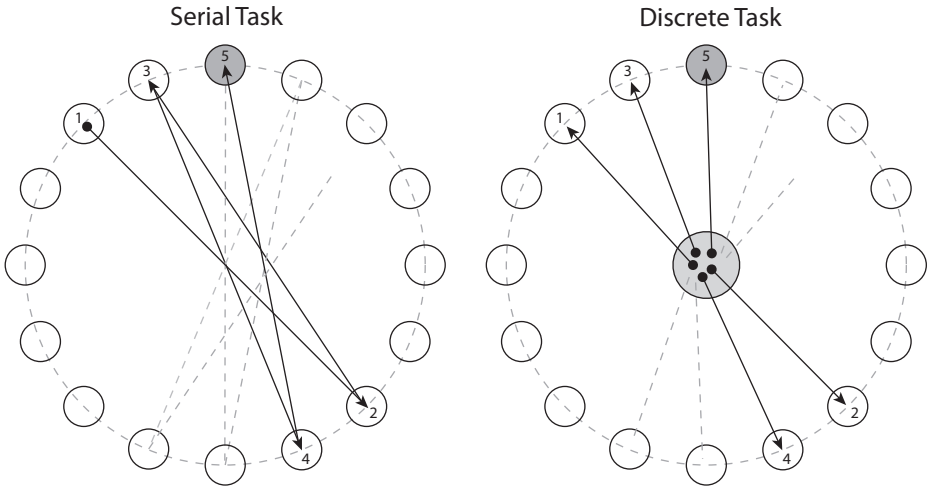


Figure 4.2: *Illustration of the process of selecting 5 targets in the ISO 9241-9 task two interface with the serial and discrete scheme. The serial task involves tapping back-and-forth between targets (left) and the discrete task involves the user to repeatedly activate the homing center and move to within the presented target (right). The current example shows four selected targets and the fifth (gray) target awaiting an selection.*

homing center and moves to the presented target (see e.g., McArthur et al. [105]). Both circular and square targets may be used and clockwise and counter-clockwise directions are allowed [155]. Target width W is given by the diameter of the target and the distances D corresponds to the distance to the new target. For a serial task this corresponds to the distance between the prior target and the new target and for a discrete task this corresponds to the distance between the "homing center" and the target. W and D should be varied during a task in order to evaluate different indexes of difficulties and conditions should be counterbalanced to avoid learning effects.

4.3 Test of Three Interactive Eye Trackers

The aim of this study is to evaluate the performance of the remote, webcam-based ITU Gaze Tracker (costing around $100) and compare it to the performance of two commercial gaze-tracking systems, a Tobii T60 ($25,000) and a Mirametrix S1 ($6,000). This is done by measuring spatial error (accuracy

and precision) and by conducting a standard target-acquisition task.

4.3.1 Method

Participants

A total of five participants, three male and two female, with ages ranging from 29 to 39 years ($M = 34$ years, $SD = 4.3$), volunteered to participate in the study. Three of the participants had no previous experience with gaze interaction. One of them used contact lenses.

Apparatus

The computer used was a 2.6 GHz Intel Dual Core processor desktop computer with 3 GB RAM running Windows XP SP3. A 17" monitor with a resolution of 1280×1024 that comes with the Tobii T60 system was used. Three gaze trackers and a Logitech optical mouse (for baseline comparison) were tested as input devices. Two of the three gaze trackers were the commercial systems Tobii T60 and Mirametrix S1. The third system was the ITU Gaze Tracker using a Sandberg Nightcam 2 webcam running at 30 fps. Due to the low resolution and broad field of view of webcams in general, the camera was fitted with a with a 16 mm lens, which allows for a good image of the eye in a remote setting. Two two Sony HVL-IRM infrared light sources was also employed in the setup with a total cost around $100. The three gaze trackers used a 9-point calibration procedure. The initial layout accuracy and precision too can be verified in Figure 4.3. The application is available for download from the *eyeinteract* website[2]. Figure 4.4 shows the experimental setup.

Design and Procedure

After calibrating the system, participants completed an accuracy test followed by a 2D target-selection task. Participants sat approximately 60 cm away from the monitor, and were asked to sit as still as possible. In order to reflect a real-life situation, no bite bar or chin rest was employed in the study. The experiment was conducted employing a within-subjects factorial design. The target-selection task had the following independent variables and levels:

[2] <http://www.eyeinteract.com/>

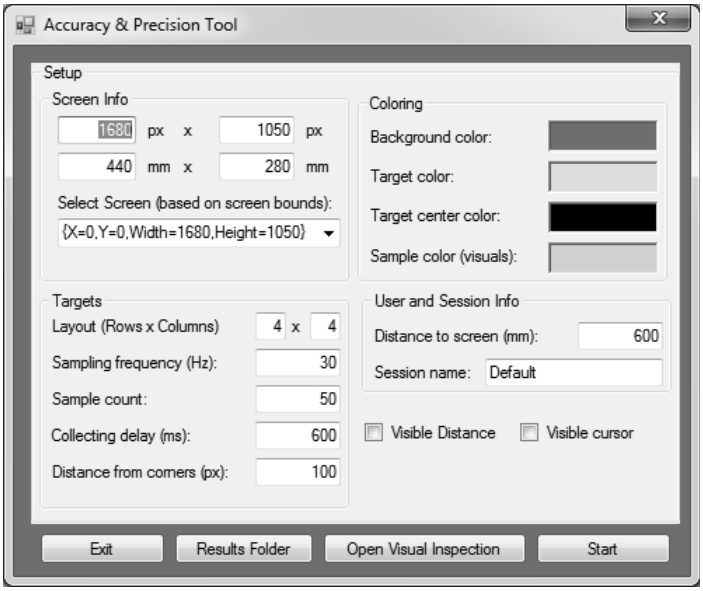


Figure 4.3: *Layout of the configuration interface for the accuracy and precision tool employed in the study.*

Figure 4.4: *Experimental setup. The participant is conducting the test using the Mirametrix system. Throughout the experiments the monitor from the Tobii T60 system was used for consistency.*

- Device (4): Mouse, Tobii T60, Mirametrix S1, Webcam
- Amplitude (2): 450, 900 pixels
- Target Width (2): 75, 100 pixels

The dependent variables in the study were accuracy (degrees), precision (degrees), throughput (bps) and error rate (%). Each participant completed 4 blocks of 1 trial (i.e., 4 trials) for the accuracy and precision test, and 16 blocks of 15 trials (i.e., 240 trials) for the target-selection task, where device, amplitude, and target width were fixed within blocks. The order of input device and task were counterbalanced across users to control for learning effects. Participants were encouraged to take a comfortable position in front of the computer and remain as still as possible during the test. The total test session lasted approximately 15 minutes.

Immediately after a successful calibration participants were instructed to gaze on a randomly appearing target in a 4×4 matrix (evenly distributed with 100 pixels to the borders of the monitor). A new target would appear when a total of 50 samples had been recorded at 30 Hz. The smooth pursuit movement and an unknown writing delay resulted in the collection of premature samples. Based on empirical tests a 600 ms delay after target onset was used before recording gaze points for a fair comparison of systems. Furthermore, samples further than $M \pm 3 \times SD$ away were considered as outliers. To prevent distractions from cursor movements, the cursor was hidden throughout the experimental blocks except, of course, for the mouse condition, which is acceptable since it is only used to construct baselines for each participant and is not used in the comparison analysis.

Once the accuracy test was completed, a serial target selection task started (cf., left-side of Figure 4.2). Participants were presented with 15 circular targets arranged in a circle in the center of the screen. Targets were highlighted one-by-one, and participants were instructed to select the highlighted target as quickly and as accurately as possible. Selections were performed with the spacebar for the gaze trackers and a left-button click for the mouse condition. Activations outside the target area were regarded as misses and were thus considered as the error rate. Every selection ended the current trial and started the next one. Based on the amplitudes and target widths, the nominal indexes of difficulty were between 2.5 and 3.7 bits.

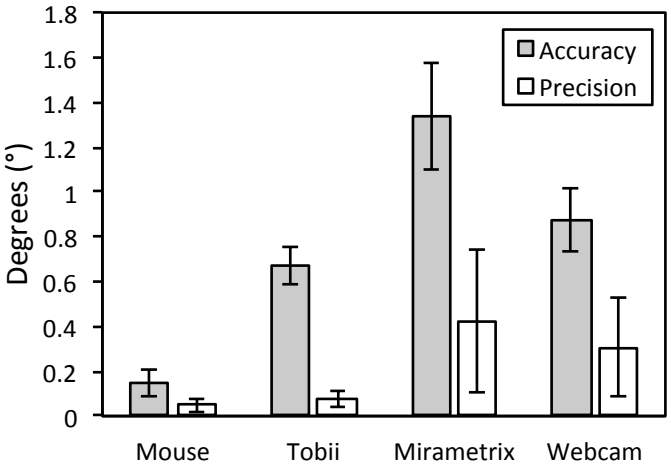


Figure 4.5: *Accuracy and Precision by device. Error bars show ± SD.*

4.3.2 Results

Spatial Error

Analysis of the accuracy and precision was performed using a one-way ANOVA, with device as independent variable. Accuracy and precision were analyzed as the dependent variables. 228 outliers of the 16,000 samples were removed from the analysis. An LSD post-hoc test was applied after the analysis. Figure 4.5 shows a plot of the average accuracy and precision per device.

Accuracy. Mean accuracy for mouse, Tobii, Mirametrix and webcam was 0.14°, 0.67°, 1.34° and 0.88°, respectively (left-side bar in Figure 4.5). The main effect of device on accuracy was statistically significant, $F(3, 12) = 16.03$, $p < 0.001$. The post-hoc test showed a significant difference between mouse and all of the gaze trackers. Tobii performed significantly better than Mirametrix, $t(4) = 3.65$, $p < 0.05$. The webcam also performed significantly better than Mirametrix, $t(4) = 4.42$, $p < 0.05$. There was no significant difference between the webcam and Tobii with $t(4) = 1.57$, $p > 0.05$. A possible reason for a missing significant difference between the Tobii and the webcam may be attributed to the limited number of participants in the study.

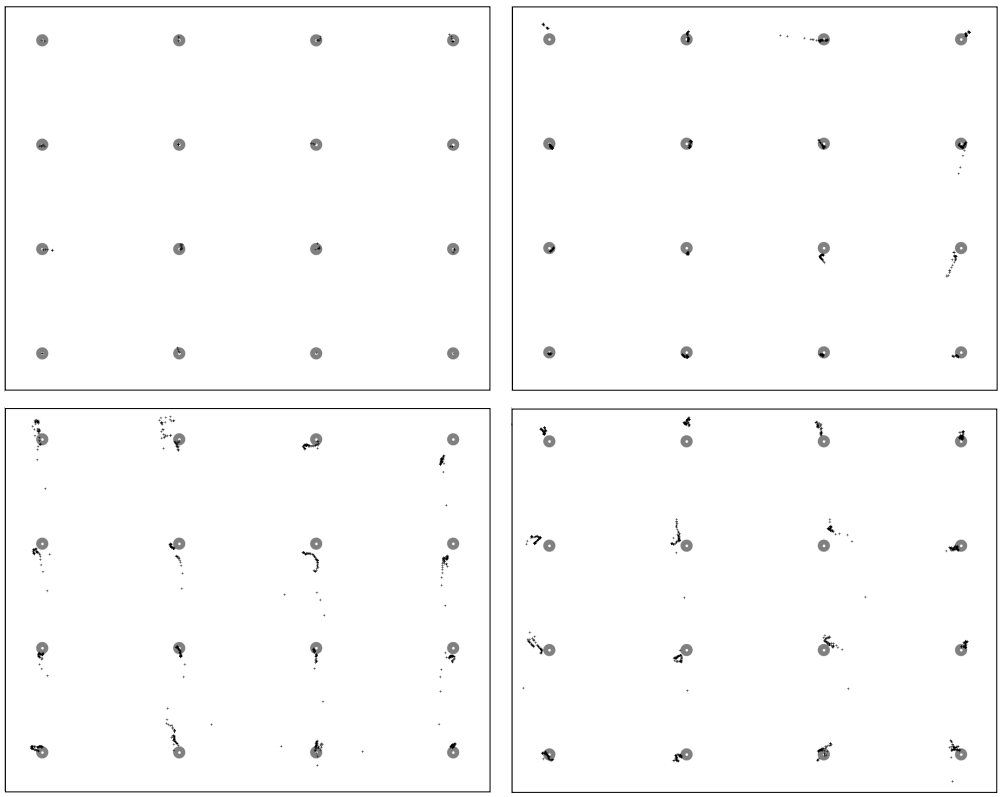


Figure 4.6: *Sample data from one of the participants in the study. Left to right (top) shows mouse and Tobii, respectively and left to right (bottom) shows Mirametrix and webcam, respectively.*

Precision. Mean precision for mouse, Tobii, Mirametrix and webcam was 0.05°, 0.08°, 0.43° and 0.31°, respectively (right-side bar in Figure 4.5). Mauchly's test indicated that the assumption of sphericity had been violated, $\chi(5) = 16.60$, $p < 0.01$, therefore degrees of freedom were corrected using Greenhouse-Geisser estimates of sphericity ($\epsilon = 0.47$). The results show that there was no significant effect on the precision of the devices, $F(1.42, 5.67) = 4.38$, $p = 0.08$.

Figure 4.6 shows an example of the mouse and the three systems employed in the study from one of the participants.

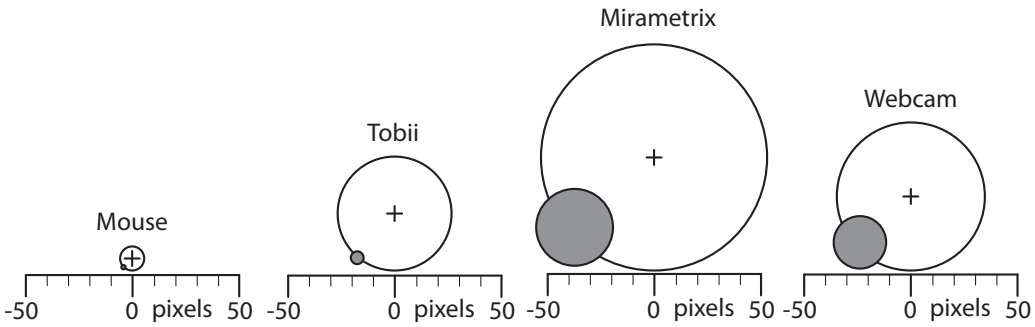


Figure 4.7: *Visual representation of spatial error for the different systems in pixels. The white circles indicate the average accuracy (i.e., offsets) between the samples and a targets (cross hairs) and the dark-gray circles show the average precision measured during the experiment. The location of the precision circles are not fixed and placed in the lower-left part of the circles for an easy overview.*

The results of spatial error can be verified in a visual representation in Figure 4.7. Even though the Tobii system is very precise compared to the two other eye tracking systems all of the systems employ large offsets, which is caused by a relatively low accuracy.

Target Selection

Analysis of the target selection task was performed using a 4×2×2 ANOVA, with device, amplitude and target width as the independent variables. For the analysis throughput and error rate were analyzed as the dependent variables. An LSD post-hoc test was applied after the analysis. All data were included.

Throughput. Mean throughput for mouse, Tobii, Mirametrix and webcam was 4.00, 2.63, 2.00 and 2.31 bps, respectively (left-side bars in Figure 4.8). The main effect of device on throughput was statistically significant, $F(3, 12) = 9.61$, $p < 0.01$. The post-hoc test showed a significant difference between mouse and all other devices. There was a main effect of amplitude $F(3, 12) = 10.73$, $p < 0.05$, with short amplitudes ($M = 2.83$ bps) having a significantly higher throughput than long amplitudes ($M = 2.62$ bps), $t(4) = 3.30$, $p < 0.05$. No significance of target width was found $F(3, 12) = 2.00$, $p = 0.23$.

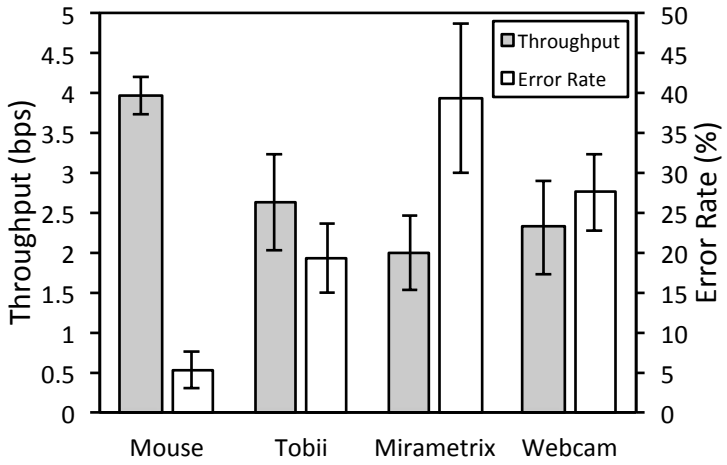


Figure 4.8: *Overall throughput and error rate by device. Error bars show ± SD.*

Error Rate. Mean error rate for Mouse, Tobii, Mirametrix and Webcam was 5.34%, 19.21%, 39.29% and 27.50%, respectively (right-side bars in Figure 4.8). The main effect of device on error rate was statistically significant, $F(3, 12) = 9.71$, $p < 0.01$. The post-hoc test showed a significant difference between mouse and all other devices. Tobii had a significantly lower error rate than the webcam, $t(4) = 4.96$, $p < 0.05$. The findings showed no effect of amplitude $F(3, 12) = 0.37$, $p = 0.58$ nor target width $F(3, 12) = 0.37$, $p = 0.58$.

4.4 Discussion

Spatial Error. The results suggest that the accuracy of the webcam-based gaze tracker (0.88°) is significantly better than the accuracy of the Mirametrix system (1.34°), while showing no significant difference to the Tobii T60 (0.67°). This indicates that the ITU Gaze Tracker can be used in software applications meant to be controlled by gaze input. However, selecting the small targets often displayed in a Windows environment might be challenging.

Although there was no significant effect of individual devices in the precision study, the data indicates that the mouse and the Tobii system had much

higher precision than the Mirametrix S1 and the webcam-based system. It must be noted that the precision is calculated after the low-pass filtering that the eye trackers perform on the data samples during fixations. This is done to smooth the signal and prevent a jittery cursor from annoying the user. The ITU Gaze Tracker gives users control over the level of smooth during fixations, a feature that many commercial systems do not provide.

Target Selection. The results obtained in the target-selection task indicate that the webcam-based eye tracker has a similar performance to the other two commercial systems in terms of throughput. The error rate of the webcam tracker was, however, significantly higher than the error rate of the Tobii T60. Throughput values were slightly lower than in previous studies [136, 182]. This can be due to the lower control over hardware setup in our experiment, as well as the low number (five) of novice users with lack of experience, who tended to be rather slow.

General Discussion. The results of spatial error analyses in the first part of the experiment confirm the observations of Johnson et al. [73]; that there is a discrepancy between the measured accuracy values and the reported values from the specifications. The manufactures state better accuracy than that measured in the experiment (see Section 1.1). This was most notable in the Mirametrix system, which had a measured average accuracy of 1.34°, compared to $0.5° - 1°$ reported in their specifications.

The Fitts' paradigm used in the ISO 9241-9 can be used to evaluate the performance of input devices. Using this methodology it allows to compare device performance form the users perspective. There are ongoing discussions about Fitts' law in general and which interpretation of the formula that should be used. For any practical purposes it is irrelevant. Although it would be desirable if researchers could agree only to use one formula for Fitts' law.

The results of the spatial error analysis represented in Figure 4.7 show how all systems produce large offsets, even the expensive Tobii system. High precision is good for interactive systems since it requires less filtering of the raw gaze samples, but high accuracy is even more desirable, since the gaze samples represents the true target better (see Section 4.1). Unfortunately,

this is not the case in this study. The Tobii system showed relatively high precision levels but none of the other systems evaluated did. In terms of interaction performance, this can be accounted for by additional filtering of the gaze signal during e.g., fixations. All of the eye trackers evaluated showed relatively large spatial offsets, which is problematic since it calls for methods that may help to accommodate for the offset during different interaction and point-and-select tasks. Later in this thesis, methods of effectively increasing target sizes will be discussed. Figure 1.1 demonstrated this issue with real targets taken from the Windows 7 environment in its default settings.

In the user study of Arne Lykke Larsen (Section 1.1), he mentions how his Tobii P10 eye tracker has difficulties tracking him in the corner of the screen. The results presented in this study consist of averaged measures across all the stimuli and not individual areas of the screen. In many systems, the corners and the top-part of a monitor are typically problematic areas and tracking point of gaze in these areas may be difficult for various reasons (see Section 3.3). Essentially, this means that spatial error differs across the screen. Such measures are essential for optimal design of interactive systems since the design should be able to maximize the usability of various areas of the screen. For example in the center of the screen, targets could be smaller and closely coupled that on the periphery of the screen.

A general problem regarding the interactive eye trackers is that the accuracy (about $0.5 - 1^{\circ}$) is not uniformly distributed over the screen. Figure 4.9 shows a surface plot of the average accuracy of the Mirametrix system superimposed on top of a Windows XP interface. An inspection of the figure confirms that the accuracy changes over the screen. Interestingly enough, the highest accuracy of the Mirametrix system was in the lower right corner and the center of the screen. The worst accuracy was located on the left side of the monitor, which is a common location for the Windows-start button, start menu and application icons.

This study presents data that is collected within a relative close time frame after a calibration. A natural question that arises is whether a good measure of spatial error immediately after a calibration is of importance compared to system robustness over time? From the target-user's perspective it is important to have a good tracking but it also of importance to maintain a good tracking over time without frequent recalibrations. Johansen et al.

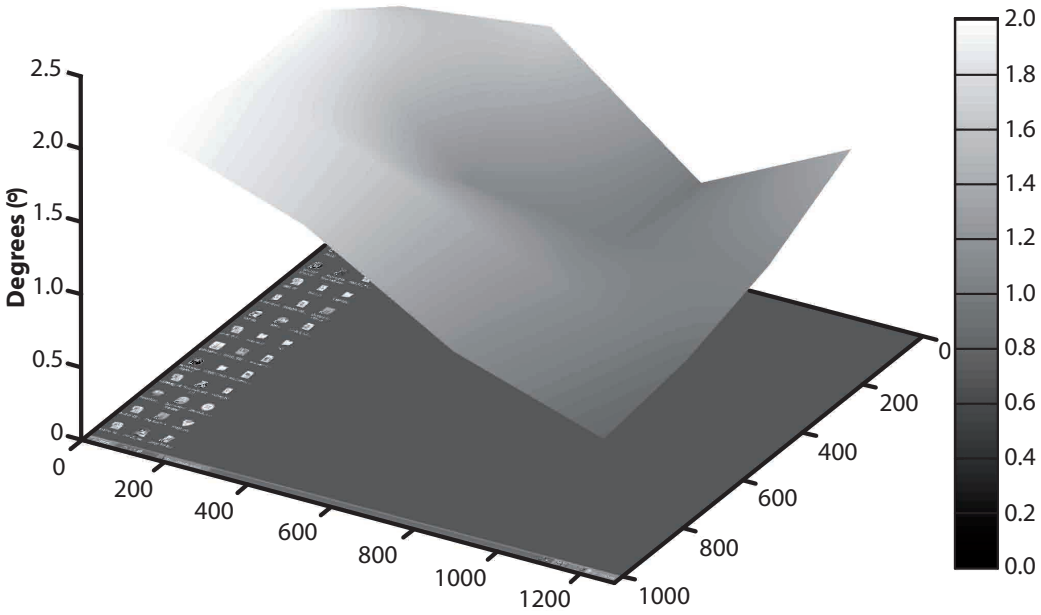


Figure 4.9: *Surface plot of the changing accuracy of the Mirametrix eye tracking system superimposed on top of a Windows XP interface.*

[72] studied robustness of two interactive eye tracking systems over time in a longitudinal study. The systems used in the study was a Tobii T60 and and a webcam version of the ITU Gazer Tracker (similar configuration as presented in Section 4.3). The accuracy and precision tool was employed in four sessions separated by two-minute intervals and users were allowed to move freely while browsing the Internet between the sessions. Figure 4.10 summarizes the accuracy of the two systems over time. The results of the study indicate that both interactive eye tracking systems had relative stable accuracy over the time. However, the accuracy of the Tobii system was better than the webcam version of the ITU Gaze Tracker.

Holmqvist et al. [50] note how spatial error directly effects interaction quality in the domain of real-time gaze control and these metrics may be of interest for end-users and interface designers. Including the methodology of ISO 9241-9 standard as a supplement to the accuracy and precision standard would be appropriate since all major users groups should be taken into account.

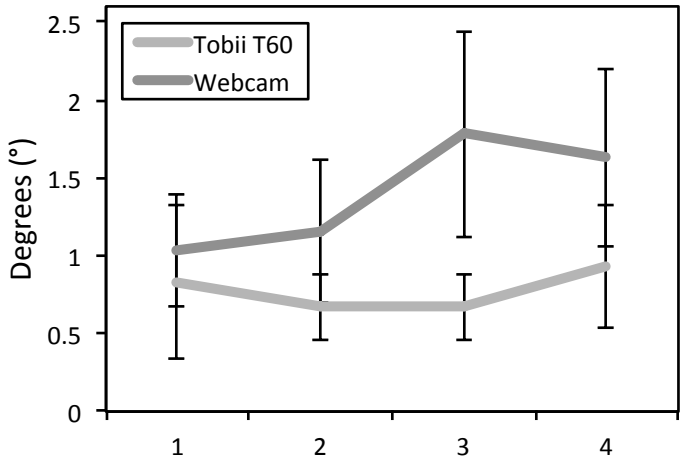


Figure 4.10: *Accuracy per sample by device in the longitudinal study. Error bars show ± SD.*

The future standard from the COGAIN organization would help manufacturers to get an objective evaluation of their individual eye-tracking systems. The evaluation should be performed by an independent lab and verify the device properties on a large user group and look at features such as spatial error, internal filters, sampling frequency, hardware specific details, calibration process, head-pose invariance, light conditions, quality of the documentation, API, ease of use, robustness and many more.

The above study is a good example why a standard of eye tracking systems is needed. Some questions raised by this study are: How robustness should be measured, which measures should be employed? How many sessions should be employed for a satisfying result and what should the interval between the measures be? Are the participants allowed to move freely between each session to simulate real-life use?

4.5 Summary

This chapter outlined important work by the COGAIN Association towards the development of a standard for the measurement of eye tracker accuracy

and precision and discussed the need to extend the measure of spatial error with a standard performance test. A method of calculating spatial error was presented and a standard performance evaluation across three remote interactive eye-tracking systems was carried out.

Results showed that a webcam version of the open-source ITU Gaze Tracker was more precise and accurate than an eye tracker costing several thousands dollars. Furthermore, the study has shown that there is a need to design dedicated interfaces to accommodate for the low accuracy of gaze input and tools to facilitate target selection need to be developed. Different viewpoints on this topic will be presented in the following Chapters. For an interaction task in a traditional WIMP-based GUI, accuracy is of greater importance (and more difficult to improve with filters) than precision and thus the following chapters will leave precision behind and focus on designing for gaze-based interaction with low accuracy in general.

Part III

Gaze Interaction

5

Computer Control by Gaze

The advantage with the conventional mouse is that virtually no noise between the physical movement of the device and the movement of the on-screen cursor exists (see Figure 4.6). This allows selection of very small targets in a windowed environment down to the finest levels, to the pixel.

Users with motor disabilities who are not able to use and control a conventional mouse need alternative input devices for performing point-and-select operations to gain access to the graphical user interface. Eye trackers are feasible input devices for users who retain control of their eye movements. Eye gaze has several desirable characteristics, such as natural and fast pointing [146]. However, most graphical user interfaces are not designed for use with these alternative input devices, which often have limited accuracy or may require unnatural selection techniques that interfere with access to mainstream GUIs.

Gaze tracking is well suited to pointing because humans naturally tend to direct their eyes in the direction of the target of interest. On the other hand, gaze tracking with no additional input modalities is not very suitable for selection, since humans tend to look at objects of interest to explore them independently of their intention to select them [64, 65]. Therefore it cannot be assumed that the user wants to perform an operation on every object that has been looked at. The speed of eye movements can, in fact, turn into a disadvantage: while it is extremely fast to turn attention to the target of interest, perceiving that item cognitively may well take enough time that the system interprets the lack of activity as an indication of expected system action. Finding a proper balance here is one of the main themes of research in this field.

The methods for computer control by gaze interaction can be divided into two main categories: either the eye tracker is simply used to control the mouse in the normal graphical user interface or a custom interface is constructed. In the first case of mouse control (the so-called eye mouse; see Bates and Istance [7]), the main problem is that there is no universal method for issuing mouse clicks. The most common method to distinguish inspections from selections is to set a time threshold (i.e., dwell time), with a click issued after the duration of the fixation exceeds a specified amount of time. Increasing the dwell time leads to slow and unnatural interaction, whereas a short dwell time leads to an increase in unintentional activations, which may cause frustration. Dwell-based activation therefore typically faces the classic speed-accuracy tradeoff: the faster the interaction, the higher the number of erroneous actions.

These problems have led to the creation of several customized interfaces that are built to accommodate the special needs of target acquisition by eye gaze. The limited accuracy of gaze trackers restricts the implementation possibilities for target selection and has resulted in a number of research projects exploring this issue. Most of these have addressed the problem by means of signal smoothing and effectively manipulating the target area with so-called zooming and distortion interfaces, which will be covered in greater detail in Chapter 8.

This chapter, explores specific techniques for gaze-based interaction intended to make target selection easier and to avoid the Midas Touch problem. Looking at techniques that do not require the use of special widgets in the interface but instead manipulate the rendering on the basis of eye gaze (so-called gaze-contingent interfaces) to facilitate the selection of small targets are investigated. Dwell-based interaction employs fixations; recent research has looked into the other option, using saccades as the basis for eye gestures. This will be investigated in this chapter. Finally, examples of how eye gaze has been used with other input modalities (blinks and winks, keyboard and mouse, facial gestures, head movements, and speech) to speed up interaction will be discussed.

5.1 Dwell-Based Selection

In dwell-based interaction, fixating for a pre-specified time threshold on a certain location will make the system issue an activation at that location. Dwelling is the most common technique for performing selections in gaze typing applications but is also applicable in other application areas, such as drawing applications [52].

Finding the optimal dwell time for issuing activations comes with a trade-off, as the interaction seems to be unnatural if waiting time increases and, on the other hand, fast interaction leads to unintentional activations [66]. Often the best solution may be to leave control of the dwell threshold to the user, as experiments with text entry have shown [99]. This allows novice users to utilize longer dwell times in the beginning and reduce the threshold as they grow more skilled and confident with the selection method.

Another possibility is to adjust the dwell time automatically, within the application, as users become more skilled. This was studied for an eye typing application by Špakov and Miniotas [171]. They found that as the users grew more experienced, they tended to leave the selected keys sooner than novice users did. As a consequence this exit time could be used as an indication of experience, and, correspondingly, the dwell time could be automatically reduced for experienced users. It was found that the algorithm was able to reduce the initial dwell time (600 ms) automatically to a level that was, in general, similar to the one chosen as most convenient by each participant when the participant was allowed to set the dwell time manually. Thus the approach was successful in this case. However, Huckauf and Urbina [55] point out that adaptive dwell time involves a lot of training (of the user, application, and algorithm) to distinguish intended from unintended selections.

Basic dwell-based interaction works well in applications specifically developed with eye gaze in mind. Then the designer can take the inaccuracy of eye trackers into account by making the selectable objects (such as keys on the soft keyboard or icons in a palette) large enough. For general access to a windowing environment, the small size of the graphical elements easily becomes an issue.

Lankford [87] explored true integration of the Windows environment with the Eye-gaze Response Interface Computer Aid (ERICA) system. Experimenting with dwell-based gaze clicking, Lankford found that many targets, such as the small buttons in the window's title bar used to minimize and maximize, were hard to select. Moreover, positioning the caret in the desired position in the text was difficult, since the density of letters was so high. In order to have the system work reliably, Lankford experimented with changing the screen resolution to the lowest possible. In addition, text on the screen was magnified 200%. These changes reduced the amount of space on the screen to a minimum, but even with this set-up the users would still need repeated attempts to hit the targets. The solution adopted by Lankford was effectively increasing target size by magnifying a small region of the Windows desktop around the detected fixation.

In summary, dwell-based methods make use of fixations and are able to accommodate for the inaccuracies of eye trackers with simple initiative such as increased screen resolution and large dwell-sensitive buttons. When the limit for reliable selecting targets is reached (i.e., targets are too small or too much noise over the system), methods based on effectively increasing the target area should be considered. Noise tolerant target selection is covered in greater detail in Part IV of this thesis.

5.1.1 Dedicated Interface Widgets

So far, it has been assumed that the interface widgets behave in a manner familiar as known mouse-based interaction. For instance, a button is pressed by clicking anywhere within its predefined area. In gaze based interaction, this does not always need to be the case. For example, the text entry technique by MacKenzie and Zhang [97] where the selection area of each soft key on the keyboard is modified dynamically, on the basis of the probability of the key (i.e., letter), given the portion of the word already typed. Characters that are more likely to follow this beginning have a larger effective selection area, although all keys are always rendered in the same size.

Another possibility is to have different viewing and selection areas for the buttons. The Quick Glance selection method [119, 120] aims to solve the Midas Touch problem by using a specific selection area next to each interface widget (see Figure 5.1); to activate the function, one must fixate on the

special selection area. There are two major advantages by doing so. First, the user can inspect the menu items without worrying about accidental activations. Second, advanced users can head for the activation area directly without even reading the menu text, just as many people know the order and location of items in the Windows Start menu, for instance. The downside of the technique is the amount of screen real estate used to display the tools. Moreover, accidental activations may still occur.

Figure 5.1: *Two-step fixation method: selection area to the right of each button. Copyright Takehiko Ohno. Used with permission.*

More recent research has used dynamically appearing interface components based on a two-step activation process. This allows for greater utilization of screen real estate and lower error rates. However, the reduction in selection errors comes at the expense of increased selection time.

Figure 5.2 shows the implementation [159] of a type of button similar to that in Figure 5.1, but now with dynamic activation. When the user dwells on the button in its initial state, the selection component appears to the right of the object of interest and the selection is confirmed with an additional dwell on its selection component.

Tall [159] experimented with various dwell time thresholds. The middle settings (300 ms for dwelling in the initial state to make the activation area visible, and another 300 ms to make the selection in the activation area)

Figure 5.2: *Dynamic two-step fixation method: selection area appears on the basis of dwell and gives feedback on activation status. Copyright Martin Tall. Used with permission.*

minimized the number of errors. In a small experiment with nine tasks, the selection time per task was slightly more than 1 sec, on average. This is not much more than the times obtained by [119] in the best case, which is an encouraging result.

In summary, dedicated interface widgets accommodate for the Midas Touch problem with the use of different viewing and selection areas. The widgets allow a user to perform really fast selections and to inspect selectable objects without any accidental activations. Methods based on dedicated interface widgets still require targets that are sufficiently large to be hit reliably with the gaze only. Furthermore, these methods seems to be limited by reserving more screen estate and a chance for introducing frustration among users from the double activation strategy per selection.

5.2 Gesture-Based Activations

Thus far the use of fixations for selection have been discussed. The use of fixations is motivated by the fact that users need to spot the target visually before selecting it; thus, target acquisition in the application can naturally follow target acquisition by the visual system. However, the nature of fixations lead them to take more time than the other basic form of eye movements, saccades. Recent research has therefore put increasing effort into studying the possibilities of gaze gestures (sequences of saccades) in target acquisition. In addition to being potentially fast, gaze gestures have the advantage that

they have the potential to avoid the Midas Touch problem, when the initial point of gaze in a gaze gesture is not significant.

Single-stroke gestures are designed to be carried out with one saccade only. Møllenbach et al. [113] studied the characteristics of single-stroke gestures in detail. The interface consisted of four peripheral initiation areas, which were immediately activated when looked at, and the centre of the screen, which was unaffected by gaze. Selections were completed by glancing from a peripheral initiation field to the opposite side of the screen in a single stroke, hence the concept of single strokes. The single stroke gestures were explored in four directions (vertical: up, down; horizontal: left, right), the effect of stroke length was examined (by varying the size of the areas), and the effect of sampling rate was also explored (by using different eye trackers). The main finding of Møllenbach et al. was that all of these parameters had a statistically significant effect on the gesture completion speed. This clearly points to the need for careful interface and interaction design when gestures are used for activation.

It would be appealing if a gaze gesture could be issued whenever and wherever the user wishes. What would the gestures then need to be like in order to be distinguishable from normal browsing of the screen? This was studied by Drewes and Schmidt [26]. Their gesture detection algorithm distinguished among horizontal, vertical, and diagonal movements within a rectangular area on the screen. In their experiment, the size of that area (and therefore the length of gestures) did not have a significant effect on the speed of gestures. They then recorded and examined normal Web surfing for half an hour and mapped the saccades to the same kind of strokes identified by their algorithm. Naturally, all short stroke sequences were found in the gaze point stream, caused by simple browsing and looking around. Therefore, they could not be used as universal gestures to invoke commands. Drewes and Schmidt did, however, find some four legged gestures (sequences of four strokes) that occurred only rarely, if at all. These could be good candidates for gestures that can be automatically distinguished from general browsing.

As an opposite example, Istance et al. [59] designed custom gaze gestures to be used with a game: World of Warcraft. In addition to the gaze gestures, they also implemented their own algorithm for detecting the gestures, instead of relying on default events produced by the eye tracker. As a result, the

Table 5.1: *Times measured for single gaze strokes (as part of longer gestures).*

Source	Stroke duration	Strokes in gesture
Møllenbach et al. [113]	79 - 270 ms	1
Istance et al. [59]	247 - 293 ms	2-3
Drewes and Schmidt [26]	557 ms	4
Heikkilä and Räihä [46]	824 - 1190 ms	2-4

gestures could be registered quickly and with good accuracy.

Dependent on the implementation, gestures can be highly noise tolerant. This was shown by Rozado [132] where the author extended an HTM (*Hierarchical Temporal Memory*) algorithm for gaze-gesture recognition. The main findings indicate that gaze gestures are very convenient for generating control commands to interface with a computer when using the appropriate recognition algorithm. In Rozado [132], the author explores the usage of gaze gestures to control a computer. Offline recognition of gaze gestures is a trivial task for traditional HTM and many other types of machine learning classifiers, with most algorithms getting robust results. However, real-time gaze gesture recognition is more challenging. This has to do with the difficulty that intentional gaze gestures have to be distinguished from standard gaze activity during normal gaze computer interaction. The algorithm is able to handle the spatio-temporal data structure of gaze gestures and properly discriminate them from other types of gaze activity and the author obtained high accuracy rates, low false positives and a positive feedback from users of this innovative interaction paradigm.

How fast are gaze gestures? This is an interesting issue, particularly since the published results vary greatly. Some results are collected in Table 5.1. The last column shows the number of strokes (legs) used in the gesture, and the second column shows the average time per stroke (duration of full gesture divided by number of strokes).

There are several explanations for such huge differences. A critical issue is what is included when one computes the duration of a gesture. Does it start from the beginning of the preceding fixation or from the first gaze point that does not belong to the preceding fixation? Similarly, does it end with

the first gaze point in a specific area (whether fixated or not), the first gaze point of the ending fixation, the gaze point that allows the fixation to be detected, or the last gaze point that belongs to the ending fixation? Another critical factor is: what is required from the gesture. Does it need to cross a line between two areas, or move between the areas (fixating on them), or follow a specific pattern? In the latter case, is the pattern visible on the screen, or does it need to be imagined by the person issuing the gesture? Such factors explain the much longer times obtained by Heikkilä and Räihä, who requested the participants in their experiment to produce images like lines, rectangles, and circles with eye gaze, motivated by the intention to use gestures in a drawing application [46]. The desire to perform the task accurately slowed down the participants in this experiment. Again, these results point to the need for and possibilities of careful design of application specific gestures, as the good performance obtained in some of the recent studies testifies. Heikkilä and Räihä [46] provide a survey of the various uses of gestures in gaze-based interfaces, in terms of both the applications and the types of gestures used.

In summary, gestures are based on saccades, which are completely different from the strategies based on fixations (i.e., dwelling). Gestures are not well-suited for target selection, since no information is gathered for the visual system, rather they are useful for issuing commands to a system similar to the computer-game control by Istance et al. [61].

5.3 Complex Interaction

The preceding sections have presented a number of techniques for activating interface elements. Life is, however, more than pointing and clicking. Fluent use of graphical user interfaces involves issuing commands such as right clicks to access a context-dependent menu, dragging items while keeping the mouse button depressed, and so on. This section looks at techniques developed for more complicated interaction than just pointing and clicking.

Pie menus are a familiar concept from mouse-based interaction that is known to perform well and is fast in practice, especially when combined with marking menus that do not even require rendering on the screen [85]. Pie menus have also been used successfully with gaze-based interaction [54, 55]. Huckauf

and Urbina [54] propose gaze-controlled pies as a universal interaction technique, demonstrating their use both for eye typing and in desktop navigation. In the latter case, for instance, their five-section pies contained desktop files and folders, and the functionality implemented included standard operations such as creating, moving, or deleting new files and folders. The items were organized in a three-level hierarchy to increase the number of selectable objects or actions that could be taken. Navigating in the pie-menu hierarchy or selecting items was set to take place after a dwell of 700 ms. An informal user study with six participants elicited positive comments and showed that all participants were able to carry out the interaction tasks in the experiment.

Later work has focused on the optimal number of slices and established six slices in a full pie as a good number, making it easy for users to distinguish the pie sectors from one another [164]. On the other hand, Kammerer, Scheiter, and Beinhauer [76] found that a half-pie with only half of the circle works better than a full pie. Their design did, however, have other differences as well: menus in sub-hierarchies did not open as new pies centered along the edge of the previous circle; instead, all hierarchies were shown with the same centre but with increasing radius of the circle.

Of particular interest is the study by Urbina et al. [164] that compared dwell-based and gesture-based selections. Instead of dwelling in a pie segment for activation, another possibility is to use simple crossing of the border of the visible circle as indication of selection. This resembles the idea of the original marking menus: experienced users could now simply glance in the direction of an item they already were familiar with, whereas novice users could inspect the pie freely before making the selection. The experiment carried out by Urbina et al. showed no speed advantage for the *selection by borders* method, but it reduced errors to half. In their paper, they point out that further improvements might be obtained by varying the location of the lower pie hierarchies; in the study, they appeared in fixed locations (centered at the border of the circle), obviously diminishing the advantage of the quick glance beyond the border. This pie menus are elaborated further in the following chapter on gaze communication.

Another setting in which diverse forms of interaction are needed is virtual worlds, such as Second Life. In that context, the users need to carry out a variety of actions for looking around, moving within the environment, and

launching events. If everything were carried out with just normal dwell, the Midas Touch problem would be prominent. Istance et al. [58] solved this in the Snap Clutch technique, using four modes, each activated by a quick glance beyond one of the borders of the screen. The modes were used for locomotion and camera control, in-world object manipulation, application control, and communication. In application control, for instance, a dwell could be used to issue a mouse-down event. Then another dwell indicated where the action was to take place, and a saccade would then generate a mouse-up event, activating the action. Such a multi-step process proved crucial where an application has controls in one part of the screen and their effect takes place in another part. In mouse-based control, the mouse cursor can be placed on the control first and the effect of the operation can be viewed when the mouse button is pressed. With gaze-based control, gaze cannot be used simultaneously for both, so breaking the sequence into its components is necessary.

A significant advantage of Snap Clutch is that the modes implemented are independent of the underlying application: they can be used in other contexts as they are. The motivation for the original work was to provide interaction techniques for users with disabilities that would make it as easy for them to interact in the virtual world as it was for able-bodied users. This goal was not quite achieved (except for locomotion control), and further work has pinned down the elements that could be improved most and suggested alternative designs [60]. These include additional feedback using a green strip at the edge of the screen to indicate the active mode, and addition of several other modes.

Porta et al. [125] suggested another solution with modes. They introduced a modified cursor control called ceCursor that could be moved in four directions by dwelling in one of the four arms pointing out of the star-shaped widget. Dwelling on one of the four arms forces the cursor to move smoothly on the desktop or jump in discrete steps from icon to icon. Selections are made with a simple dwell in the centre of the star. An interesting feature is the hot spot replicated in the activated arms. This ensures that the user will always know what is underneath the centre hot spot. Although this is not considered to be natural interaction, the ceCursor control facilitates fine-grained positioning.

This section presented combinations of gestures and dwelling (i.e., saccades

and fixations) together with joystick-like behavior. All methods seem to be prone to noise in several ways. For example, by adjusting the size of the pie menus the noise from the gaze input is able to reside within the individual selection areas. Finally, the ceCursor offers pixel-level precision through the joystick-like interface (that only seems to work for static targets). In summery, this section have presented examples of noise-tolerant methods that can be operated with unnatural gaze movements.

5.4 Enhancing Gaze: a Multi-Modal Approach

So far, interaction techniques that are suitable for use with gaze as the only modality have been described. The following turns the attention to techniques wherein the use of gaze is supported by a multi-modal approach or used to enhance traditional techniques with other input channels will be addressed. Using one or more additional modalities in conjunction with gaze may help to reduce erroneous selections and thus help to eliminate the Midas Touch problem. An additional modality can be seen as an indicator of confidence during a given task compared to interacting with the gaze-alone.

5.4.1 Blinks, Winks, and Pupil Dilation

Since the eyes are used for selection, it seems a natural idea to use them for more than just the point of gaze. Blinking (closing both eyes) or winking (closing just one eye) can be used for triggering a mouse click event. However, winking is not easy for some people, and intentional blinking would need to be separated from natural blinking. This could be done with prolonged blinks for activating events, but at the expense of speed of interaction. Moreover, as Huckauf and Urbina [55] point out, blinking may affect the vergence of the eyes. Overall, selection by blinking may prove strenuous for the eye muscles. Selection by blinking is supported by several manufacturers of eye tracking systems and has been used in research prototypes (e.g., Grauman et al. [38]). However, using blinking and winking for selections has not gained popularity comparable to that of dwell-based selections.

Another rarely used attribute of the eyes is pupil dilation. There is a good reason for this: it is hard to control pupil size voluntarily, and to distinguish voluntary changes in pupil size from involuntary changes in pupil dilation

that can be prompted by one of many factors (such as excitement and image brightness). However, Ekman et al. [30] showed that voluntary pupil size control is possible, and changes in pupil size can be detected on a statistically significant level, especially with properly designed feedback. Ekman et al. (2008) then explored the use of pupil size as a control mechanism in a game, where the interaction was carefully designed so that voluntary pupil changes were coupled with actions designed to instigate positive arousal. Thus both voluntary and involuntary pupil dilations contributed to the change detected in pupil size.

5.4.2 Keyboard and Mouse

A natural way of improving access to arbitrary graphical user interfaces is to provide a separate window that shows an enlarged image of part of the screen, similar to the Zoompad prototype by Istance et al. [62] where an area around the measured focus of the gaze is shown in a special panel. The same panel contains gaze-activated buttons for operations that correspond to those that can be issued with a mouse. A similar technique can be applied without reservation of any additional screen real estate if gaze is used in conjunction with the conventional mouse and/or keyboard.

Keyboard. Combination of gaze interaction with input from the keyboard was implemented in the multi-modal EyePoint prototype [84]. Figure 5.3 illustrates the sequence of actions being performed to click on a specific link on a web page. First, the user locates the target of interest with the gaze. Second, pressing a key on the keyboard brings up a magnified view of the region that was looked at. Third, the user locates the target in the magnified window and releases the keyboard key to issue the click. In the final step with the prototype, the magnification allows for more accurate selections since the effective width of the target is increased - a feature inspired by the two-step technique of Lankford [87].

In a controlled experiment, EyePoint was found to be slower and less accurate than a standard mouse in a pure pointing task. However, in a more realistic task that combined pointing and typing, EyePoint was slightly but

Figure 5.3: *The EyePoint selection method [84] uses a two-step process, first fixating on a target and then pressing a keyboard shortcut. This will bring up a zoomed-in view around the last fixation, which allows higher tolerance for eye tracker noise. The user fixates on the target again and releases the button, which issues a simulated mouse click in the position of the current fixation.*

significantly faster than using the mouse. In this case, the fact that participants could keep their hands on the keyboard without a need to reach for the mouse compensated for the intrinsic slowness of the two-step technique. Error rates remained considerably higher for EyePoint in comparison to the mouse.

In a study by Møllenbach et al. [114] they evaluated mouse and gaze interaction in two zoomable, multi-scaled information spaces: one large with 2000 nodes designed for a searching and browsing task and a second designed for precision zooming in a target selection task. For control mouse and gaze was used to control panning and two keys on a standard keyboard was used for zooming in and out, respectively. In their study they compared the navigation of gaze and mouse and results indicated how the performance between gaze and mouse were indistinguishable. However, in the selection task, the gaze proved 16% faster than the mouse control. Møllenbach et al. conclude how gaze-controlled pan/zoom-navigation is a viable alternative to the traditional mouse control during inspection and target exploration in multi-scaled environments.

In addition to pointing and clicking, another extremely frequent operation in a windowed interface is switching between applications and between windows. This, too, can be supported by eye gaze. Figure 5.4 shows how the EyeExposé prototype manipulates the window-switching solution in the Windows operating system. The prototype has modified the normal "press-release-mouse click" sequence and replaced it with a "press-look-release". The

potential for saving time during interaction is obvious, and accuracy should not be an issue, as the proxies of the miniaturized windows tend to be fairly large.

Figure 5.4: *The EyeExposé window selection method [82] supports fast application switching. The interaction combines the use of keyboard input and gaze pointing.*

If there are no overlapping windows (i.e., windows are placed side by side), the switching operation can be further simplified. Gazing at a specific window for a predefined time can then enlarge and put the focus on the window of interest and decrease the size of all unattended windows. This was implemented in the EyeWindow system (see Figure 5.5) by Fono and Vertegaal [35].

Figure 5.5: *EyeWindow by Fono and Vertegaal [35]. On the left, gazing at the bottom right window makes it bigger and moves the focus to that window, so that (on the right) typing can continue without removal of the hands from the keyboard.*

In their study, Fono and Vertegaal [35] found that eye tracking with automatic activation was about twice as fast as use of a mouse with special

keyboard keys. Eye tracking with key activation was more than 70% faster than manual conditions and was preferred by most participants.

Mouse. EyePoint illustrated how the use of dedicated keys in conjunction with eye gaze can support accurate selections with zooming. The earliest technique that combined the use of eye gaze and traditional input devices was MAGIC pointing, from Zhai, Morimoto, and Ihde [181]. In MAGIC pointing, the mouse pointer is automatically warped to the vicinity of a selectable object if the user's gaze is within a specified spatial threshold of the object. The final fine-grained movement of the mouse pointer is done with the physical mouse. Zhai et al. found in their experiment that the participants liked the technique and it was slightly faster than use of the mouse alone. No differences in accuracy (i.e., incorrect selections) were found.

Räihä and Špakov [128] developed a related technique to be used with multiple monitors. To overcome the long mouse movements in such settings, it has been suggested that multiple mouse pointers could be used, all controlled with a single mouse simultaneously and in synchrony [80]. Selecting with such a technique is ambiguous if several mouse pointers are present at the same time (i.e., multiple cursors on top of selectable targets). Räihä and Špakov suggested the use of eye gaze to select the active pointer. The pointer closest to the user's point of gaze was active and the underlying target received the mouse click. In an experiment with two monitors, the method improved target-acquisition times over long distance (when the target was on a different monitor than the previously active pointer). Users preferred the condition in which each monitor had one mouse pointer and eye gaze was used to indicate the active monitor. Blanch and Ortega [12] suggested a similar technique with even greater speed improvements.

5.4.3 Facial Muscles (EMG), Head and Speech

One user group that benefits from computer control by gaze is people with quadriplegia. For them, use of the other facial muscles (in addition to eye muscles) could provide additional control mechanisms. Extensive studies on use of frowning (movement of the muscles of the forehead) together with eye gaze were carried out by Surakka and his group [122, 158]. Muscle activity was measured with two electrodes attached to the forehead. In summary, their results showed that for short movement distances (6 cm on a normal

screen), interaction with the mouse was faster for able-bodied participants, but with longer distances (starting with 12 cm) gaze-based looking and frowning was not significantly slower than the mouse. In fact, with the longest distances the results indicate that the new technique could even beat the mouse. Not surprisingly, the speed of eye movements could compensate for the loss in selection speed from frowning. As usual in gaze-based interaction, the mouse is superior in selection accuracy but the accuracy of selection by gaze can be improved with sufficiently large targets.

An interesting solution for head control was suggested by Adams et al. [1], who used body movements together with eye movements. In particular, since it is natural for humans to move closer to a target that they want to inspect in greater detail, they used the screen to eye distance to control zooming of the view. The setting for their experiment was Google Earth, where participants needed to navigate to a given destination. They compared four methods of zooming: by head movement, by staring, by mouse, and by mouse but with gaze for panning. Looking at the edges of the screen activated panning; the pan region was less than 15% of the screen real estate in all directions. Zooming could be activated in the centre area by one of the four techniques. In staring-to-zoom, for instance, fixating within the centre area caused zooming in at a comfortable rate and also panning to place the fixation location at the centre. Visually scanning the centre area stopped the zooming. Glancing at the eye tracker below the screen triggered zooming out. A similar idea was proposed by Lepinski and Vertegaal [90] and implemented with a low-cost Web camera. However, only anecdotal evaluation results were published.

Combining gaze with speech is another attractive option. It was pioneered early on by Bolt [15] in his seminal Put-That-There system. It initially used pointing gestures for indicating the target and spoken commands to cause actions, but gaze was soon added as an additional modality to compensate for the inaccuracy of pointing gestures. Neither gaze nor gestures alone was accurate enough for the system, but when the two approaches were fused the overall robustness improved. Other work within the area of combining gaze and speech include: [78, 111, 83, 76]

5.5 Summary

Efficient algorithms need to be developed to determine users' intentions from spatiotemporal eye movement patterns to minimize the Midas Touch problem. In other words, software solutions need to be able to analyze the users' gaze on the monitor and react accordingly. If gaze-controlled systems are designed properly, the life of people with disabilities can be considerably improved. The use of dwell-based selection have been discussed and is the most common technique in use today. In addition, the approaches covered included specially designed interface widgets, variations of zooming, the use of gestures, and multi-modal interaction.

The examples of dedicated interfaces are mostly designed with large gaze-sensitive buttons that accommodate for a noisy input by residing the inaccuracies from the gaze input within the object of interest. Dwelling on objects for a pre-defined threshold of time helps to discriminate between inspecting fixations and selections and thus helps to eliminate the Midas Touch problem. The dedicated interface widgets are based on the same design strategy but dwell time can be lowered significantly due to a two-step selection process where a selection is confirmed by activating an additional target placed sufficiently far away from the target of interest. This allows the system to reliably detect activations from a user with few errors.

Gestures are based on saccades as opposed to dwell-based methods that are based on fixations. The advantage with gaze gestures is that they have the potential of avoiding the Midas Touch problem as they are recognized with advanced recognition algorithms that can be trained by the individual user. The naturalness of the gestures may be subjected to discussion but the technique seems prone to noise from the input.

Finally, the multi-modal approach uses the eye-gaze for pointing in the well-known point-and-select strategy and leaves the selection for an additional modality such as an EMG switch or human speech. As mentioned in the introduction, gaze is a natural way of pointing since humans tend to direct the eyes in the direction of the target of interest and signaling an activation with an additional modality is analog to using a physical limb such as a finger.

How fast can computer control by eye gaze get? Many examples illustrate how gaze can often be comparable in efficiency to control by mouse, and sometimes even faster (e.g., Dorr et al. [24]). On the downside, the biggest difference is in the interaction accuracy where selections are not always recognized correctly. They may simply be missed altogether or be associated with an incorrect object. Several techniques have been tested, and many are used commercially, but there is still room for improvement, in order to find the proper balance of operation efficiency, accuracy, and user satisfaction.

This chapter has provided an overview of some of the applications and interaction methods presented in the literature. The following chapter deals with communication applications, which are a special case of applications for gaze interaction. These applications are of importance for users who rely on eye trackers as the means for communicating with friends, family and the outside world.

6 Communicating by Gaze

This chapter introduces communication by gaze. First, a presentation to augmentative and alternative communication (AAC) is given along with a presentation of AAC in the context of using an eye tracker for communication. A case study of a late-stage ALS patient is used to illustrate why the need to communicate is a fundamental for everyone. The second part of the chapter presents different eye typing methods that can be used to produce text with gaze input only. The interaction methods are categorized, exemplified with concrete examples and each technique is discussed. Finally, a review of the gaze-typing systems that have been documented in the research literature is presented (see Table 6.1). The table summarizes overall design considerations and several system characteristics such as input method, experimental design, error rate and efficiency measured in words per minute (WPM). The chapter concludes with a discussion of the systems reviewed.

6.1 AAC

AAC is used as a supplement to or replacement of oral speech so that people with physical impairment can express their thoughts, needs and feelings. Typically, use of AAC involves selecting one of several objects on a display, either aided or unaided by employing high-level or low-level technologies. Higginbotham et al. [47] note how the technological development in interaction has dramatically increased access to AAC technologies for the individual person. In their paper, they define access to AAC as the means or opportunity to:

- Use or benefit from something (e.g., operate a communication device).
- Approach or see someone (e.g., converse with a person).

- Obtain or retrieve information from a person, the environment or an artifact (e.g., read from a communication device).
- Provide use or benefit from something or someone (e.g., assist someone in communicating using AAC technology)

One of the most fundamental goals in AAC is to obtain automaticity. Automaticity means that an impaired individual is able to perform complex tasks that require cognitive and/or motor skills under varying conditions (e.g., writing, biking, driving, singing) that they would otherwise not be able to do. Automaticity is achieved when a task is performed without any significant cognitive load and is typically learned through numerous repetitions in a well-defined task space [121]. Moats [112, p. 432] notes how automaticity can be defined as: *"fluent performance without the conscious deployment of attention"*. Automaticity is often difficult to accomplish and can, for example, be hampered by poor literacy or simply because it can be difficult to maintain focus on a display.

6.1.1 Eye Tracking and AAC

An eye tracking system may be the only means of communication for some people who are completely locked-in. Employing a simple eye gesture system such as looking up, down, left, and right may serve as a feasible method of communicating. In the human-to-human approach the "eye tracker" is an interpreter standing in front of the person. This analog solution can be optimized with aids such as a transparent plastic frame with special stickers [37]. Here, the interpreter holds the plastic frame in the hands while paying close attention to the users eye movements onto the different targets on the plastic board. Although the board is limited in terms of forming complex sentences, it can be useful for uttering short commands, it works well for outdoor scenarios and does not require any use of electricity.

Since the 1970s eye-tracking systems have been used for text production and allowed people to communicate and interact with family and friends [101]. Locked-in owners of eye-tracking systems still need assistance for the initial setup, which involves turning on the equipment, positioning the user in an appropriate distance-to the eye tracker's camera, and finally, starting a calibration. A person who is completely locked in still requires a little assistance

before use. Most modern interactive eye tracking systems are shipped with dedicated software that can be used for communication, mouse emulation, internet browsing and entertainment. Some concrete examples of applications in each category can, for example, be seen in the taxonomy by Vickers [169]. For a list of non-proprietary software, please visit the COGAIN website[1] for further information.

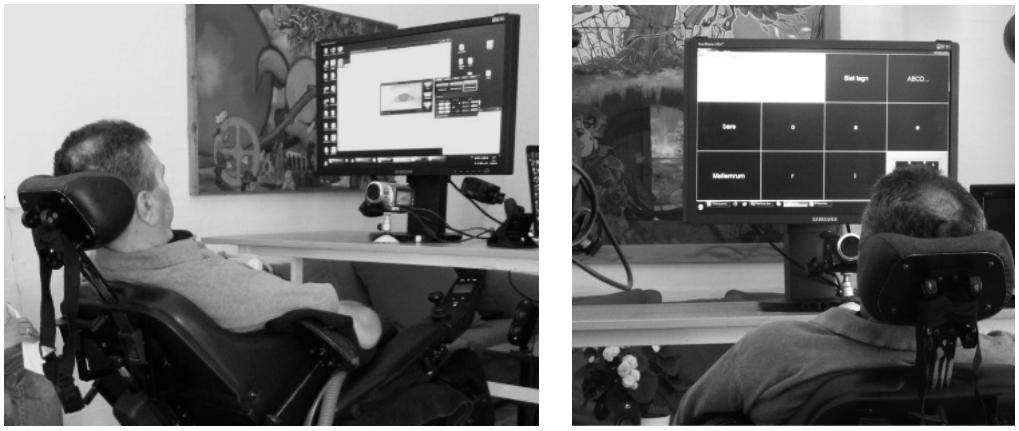
Human communication is a fundamental need and the inability to communicate may lead to loneliness and social exclusion [98]. Most communication applications allow for individual personalization such as changing the layout from QWERTY to alphabetical or choosing between dwell or step scanning for selections (see Section 6.2 for further details). Normal face-to-face speech is somewhere in the area of 100 - 250 wpm, which is significantly faster than most eye typing systems [98]. Even with the use of predictive models, gaze-based typing speeds have not exceeded this barrier, yet.

6.1.2 Case Study: Birger

Birger Jeppesen has suffered from ALS since 1996 and has used eye movements as his only means of communication since 1998. He is ventilated and needs 24 hour care. When Birger lost his ability to control the conventional mouse on his computer, he started using the QuickGlance eye tracking system for written communication. During the progression of his disease, Birger has maintained his own web site, written several articles and a book [70]. Due to the progression of his disease, he gets tired from his QuickGlance system quickly and rarely uses it now. Birger communicates with his caretakers and family based on face-to-face communication. When a person needs to communicate with Birger, she uses a memorized spelling grid with alphabetical letters grouped in rows. The spelling-grid method is analog to the step scanning and word predictions known from several switch-based systems and eye-typing systems [142, 62].

Birger has previously participated in an evaluation of the first version of the ITU Gaze Tracker[137]. Two years after the first study, the remote version of the ITU Gaze Tracker was in a state where it could be used for a user evaluation on real end users. Birger and his wife took part in the evaluation,

[1] <http://www.cogain.org/wiki/>



(a) Preparing Birger to calibrate on the remote system.

(b) Birger using the GazeTalk communication system.

Figure 6.1: *Figure (a) shows how Birger is sitting comfortably in front of the eye tracker and Figure (b) shows how Birger is communicating with the GazeTalk communication system.*

which was documented in a video[2]. In the study, a standard Sony video camera with night shot mode, auto focus and a Sony IR lamp was used. The setup is displayed Figure 6.1 and shows Birger sitting comfortably in his chair approximately one meter from a 24" monitor.

Immediately after calibrating on the system, Birger was able to answer our questions. His first sentence was: *"An eye tracker is my lifeline to the world."* Then followed by: *"It is a splendid system you have made."* Birgers wife notes: *"Being able to communicate is a human right"* and continues: *"Low-cost eye tracking system would allow any disabled person to express feelings, needs and thought and without a communication system, you take away the possibility of being a human being."* Birger follows: *"The possibility to communicate in this way means that I am able to have a private life and the freedom to be myself".*

[2]<http://vimeo.com/23375917>

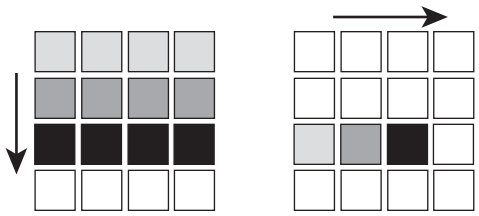


Figure 6.2: *Scanning example with sequential row and column scanning. Reproduced after Shein [143].*

6.2 Eye Typing

The first applications made for gaze interactive systems in HCI were focused on the communication needs for people with disabilities and provided them with typing interfaces [68]. Majaranta and Räihä [102] have studied these typing systems and categorized the process of producing text according to input technique. The categorization divides input techniques for gaze typing into two overall strategies: direct pointing or continuous gestures. Majaranta [98] further notes how the interaction method on most common typing systems can be divided into four categories: *Switch*, *Direct*, *Discrete Gestures (DG)* and *Continuous Pointing Gestures (CPG)*:

- **Switch**, is the preferred input for users with some level of motor control and/or who cannot remain seated in front of an eye tracker for instance due to physical condition that causes involuntary muscle movements. Most commercial eye trackers support voluntary winks that can be used to emulate a binary switch [38]. The typical layout for the switch input technique is letters arranged in a grid or matrix. Selections can be based on simple row-columns scanning technique where each selection is based on two activations. First, the rows are highlighted one-by-one until the user generates the first activation. Secondly, the letters in the selected column are highlighted one-by-one until the user completes the final activation, which leads to the selection (see Figure 6.2).

  As long as the user's eyes are within the field-of-view of the eye tracker, the switch method is highly tolerant to noise and offsets since the quality of a calibration is not used to direct a cursor or marker on the interface. The performance of switch-based systems are limited to the speed

of the scanning routine, which consists of a predefined reaction time and an activation delay. This is necessary to give the users enough time to make confident activations. There have been several attempts to improve on the scanning method with a predictive model (e.g., Evreinov and R. Raisamo [32] and Ashtiani and MacKenzie [3]). The cognitive load of learning to operate a switch-based application is at a minimum. The scanning technique is robust in the sense that gaze direction and the quality of the calibration is of no importance. In terms of efficiency, scanning speed can be increased after practice but there are at least two factors that significantly reduce efficiency: waiting time between each step and an input restricted to one bit of information (i.e. signaling an on/off signal to the system). Finally, the amount of errors generated is a tradeoff between the time interval between each scanning step and reaction time of the user. Operating a switch is a basic task that can be operated by many users.

- **Direct**, refers to the most common text-entry method used and is based on direct pointing in a dedicated interface. The layout is often a full or restricted on-screen keyboard that employs large text and big buttons. In most cases, the keyboard layout is QWERTY but alphabetical or custom setups are not unusual. Whenever gaze falls within the border of a gaze-aware object, the object may highlight to indicate that it is ready to be selected by an activation. Usually, selections are done with dwell-time activation, i.e., a prolonged fixation within the object's activation area. Often, the dwell time is indicated with an opaque animation, which acts as a feedback for the user. When the animation disappears, the fixation time has expired and the object is selected. In some cases a selection is accompanied by a sound, often a subtle 'click', which acts as a confidence indicator for the user. An example of the direct category is the GazeTalk typing system that employs a restricted keyboard with dwell-sensitive buttons (see Figure 6.3) [43]. The dynamics of a system based on the direct method make them easy to learn and understand. Dwell times usually range from 500 to 1000 ms [98] but dwell times as low as 150 ms have been reported [64]. In a recent study, Majaranta et al. [99] found that experienced users preferred an average dwell time of 378 ms when typing on a full on-screen keyboard. Interfaces based on the direct technique employ big text and large buttons to make them easier to perform activations.

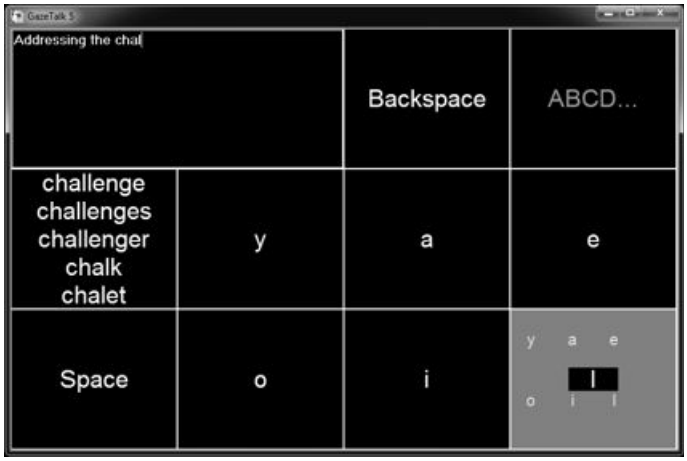


Figure 6.3: *GazeTalk employs a restricted keyboard with dwell-sensitive buttons and is an example of a system in the direct category.*

The design consideration for the direct technique is to make the targets large enough and well enough separated to make them tolerant to noise (i.e., jitter and offsets), so that noise is restricted to within the objects of interest. Studies have shown that methods based on the direct technique can reduce dwell time and, when combined with a predictive model, even increase the number of key strokes per character in a typing task. Input based on the direct technique resembles well-known pointing operations from real-life (e.g., operating a remote control), which makes the technique natural and easy to learn and use. One of the challenges related to the direct method is to choose an appropriate dwell time for a given task. As mentioned earlier in Chapter 5, a long dwell time leads to slow and unnatural interaction, whereas a short dwell time leads to an increase in unintentional activations.

- **Discrete Gestures** (DG) are performed with distinct gestures as noted by Majaranta and Räihä [102]. Selections are completed by moving gaze between regions on the screen in predefined patterns. These regions are also known as hotspots or hot zones. Hot zones can either be visible or invisible dependent on the users' skills. For example, in learning mode the hot zones are visible and function as a guide, supporting the user with an active feedback and in expert mode, where the user

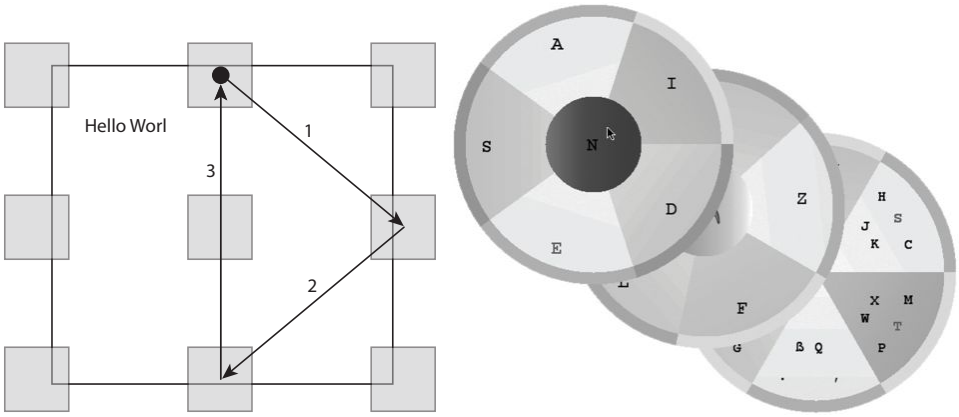


Figure 6.4: *The use of discrete gestures can for example be employed in systems based on hot spots or pie menus. Left example shows how the letter "d" can be produced with three gestures based on the Eye-S alphabet [126]. Right example shows a gaze-controlled pie menu with letters grouped into sectors that can be expanded with gaze activation. Copyright of Mario Urbina. Used with permission.*

knows the system dynamics, the zones are opaque. False activations are prevented by keeping the gaze within the hotspot areas for a short period of time (i.e., dwell time). Figure 6.4 (left) shows an example of a typing task that employs hotspots. The inspirational source for distinct gestures in gaze interaction is most likely based on the interaction from the early hand-held stylus-based devices with glyph gesture alphabets, also known as *graffiti* (see e.g., [13, 126]). Activations based on DG are known as strokes. A case of dwell-time-free DG is the pie menus e.g., [17]. These interfaces employ letters grouped into sectors and when the gaze passes over the outer border of a sector the content is presented in a sub-menu that expands as a new pie. A selection is completed by passing gaze over the outer sub-menu border. This process is demonstrated in Figure 6.4 (right). DG are based on short strokes that almost resemble the same process as hand-writing [102]. However, the human eye is not designed for distinct gestures and there is a great chance that most users find these interfaces unnatural. Although, DG may seem as an unnatural input the technique is highly tolerant to noise since the size of the hotspots and pie menus can be changed according to the quality of the calibration and eye tracker. Interaction with DG can

also be improved with predictive modeling such as the *SHARK* (Shorthand Aided Rapid Keyboarding) system by Zhai and Kristensson [180].

DG are represented in two types of interfaces: command based and dedicated. Interfaces based on hot zone can be overlayed on top of a typing application (e.g., MS Word) and pie menus are employed in dedicated interfaces that provide continuous feedback for the user. In general, training is an important factor with methods based on DG, for example, the hot zones require a user to learn the necessary patters for each letter in the alphabet to produce text. Also, this may result in high error rates in the beginning. Finally, the user needs to practice the technique on a regular basis in order to maintain skill, which may be tiring over time due high cognitive demand and unnatural eye movements.

- **Continuous Pointing Gesture** (CPG) are probably the most intuitive technique for gaze interaction since natural eye movements are used for selections. The method is analog to continuously writing on a piece of paper without lifting the hands at any time [99]. The Dasher typing system is the best example of a system based on continuous pointing gestures. Dasher is a mode-free system that employs zooming into predictive character-cells whose sizes are determined by a predictive character-based language model [173]. Typing is controlled by a continuous two-dimensional navigation in the reading direction in a column of characters that are ordered alphabetically and scaled according to their probabilities (see Figure 6.5). Users make selections by searching for the letters of interest without any unnatural dwells to break the flow. Looking in the counter-reading direction (left) changes the direction of the zoom and results in character deletions. Dasher has implemented a feature to accommodate for inaccurate inputs (e.g., offset) with statistical analysis of the accessed regions. Selections are over time evenly distributed around the center-line of the application and this information can be used to infer the quality of the calibration and can thus reduce offsets in order to prolong typing without recalibrating. In a study by Itoh et al. [63] they found that even with a deliberate miscalibration on an eye tracker, people could still type effectively on Dasher. Dasher requires training time, which is confirmed by Urbina and Huckauf [163]. They observed how novice users often felt stressed

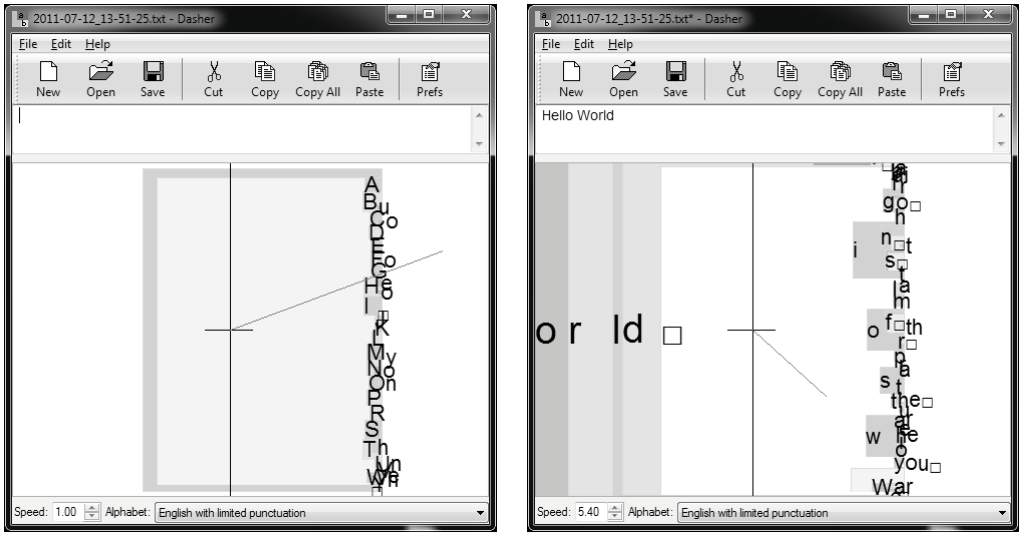


Figure 6.5: *Dasher, the mode-free typing system, by Ward et al. [173]. Letters are arranged alphabetical on the right side of the application (left) and by looking at a letters of interest, its area grows (right). The predictive model helps to grow character cells with higher probability to accommodate for easier and faster selections.*

from the constantly changing layout.

It should be reasonable to believe that continuous pointing gestures are easy to learn since they are analog to continued writing on a paper. However, systems employing this method may increase complexity and cognitive load of the users. Several studies on Dasher (e.g., [162]) show how users experience problems when learning to use the system (i.e., steep learning curve). The go-where-you-look approach is very intuitive and the limitations of Dasher seem to reside in the interface and not the interaction technique. The radical novelty of the navigational technique in Dasher is completely different from all other techniques discussed in the earlier sections. In some cases, the language model allows for multiple character selections without moving gaze, which is a result of the dynamic display. Studies show how Dasher is the fastest typing system (see Table 6.1) and smooth pursuit movements in a dynamic environment controlled by continuous pointing gestures seem to

be a very applicable approach for producing text with the eyes-only. However, the fast typing speeds are only achievable by overcoming the steep learning curve through a lot of training.

6.2.1 Predictive Models

The efficiency of gaze-based text entry may be significantly improved with character or word predictions. The time duration of a selection sets a maximum limit for the efficiency of typing but statistical models are able to improve on this constraint. Usually, a language model is designed to improve the *key strokes per character* (KSPC) metric, which has a direct effect on the word-production speed [96]. This is achieved simply by offering relevant word suggestions or word completions.

The most common techniques are based on *n-gram* models where the most popular are unigram, bigram and trigram denoted with $n = 1, n = 2, n = 3$, respectively. Estimating the probability of a word sequence, $w_1^n = w_1, ..., w_n$ can be formalized as in Equation 6.1. The n-gram probability is based on a statistical analysis of a large set of training text [20].

$$P(w_1, ..., w_m) \approx \prod_{i=1}^{m} P(w_i | w_{i-(n-1)}, ..., w_{i-1}) \tag{6.1}$$

The uses of language models in communication systems are not limited to n-grams and these can easily be combined with character-based models. GazeTalk for example, employs several n-gram-based language models and a character model. Figure 6.3, shows five word suggestions, based on the n-gram model, is presented beneath the text box. A character model is used to generate the dynamic layout of the character cells in order to promote letters of high probability. The figure shows how the character l is being selected (lower right corner) indicated by a shrinking animation (dwell time). A preview of the next layout is presented within the cell in order to keep search time as low as possible between selections.

GazeTalk, for example, employs an additional language model based on the users' own vocabulary, which is of great importance from a usability perspective. The reason for this lies in the vocabulary that is one of the individual traits that makes a person unique. This needs to be reflected in any commu-

nication system and this also explains why it can be a difficult decision to change to a new system.

6.2.2 Review of Typing Systems

Table 6.1 presents a review of the gaze-based typing systems that have been presented in the literature. The table gives an overview of the systems and system characteristics such as: input method, layout type and employed eye tracking system used. In the case of an unnamed system or missing feature, the fields are marked with an *Unnamed* or *N/A*-tag (not available), respectively. The individual characteristics used in the table are explained in detail here:

- **Appears In:** indicates where in the literature the current study appears.
- **Name:** represents the name of the system given by the designers.
- **Tracker:** indicates the eye-tracking system that has been employed in the current study.
- **Method:** is used to indicate the employed interaction method such as: *Switch Direct*, *Discrete Gestures* (DG), and *Continuous Pointing Gestures* (CPG).
- **Design:** indicates the paradigm that is used to control the typing system (e.g., QWERTY, pie menus, multitap).
- **Predictions:** (Predi.) indicates whether any type of predictive model was used. A correction mark (✓) is used to indicate true and a cross (✗) for false.
- **Audio:** indicates if the system provides any sound feedback to the user and is marked with ✓or ✗.
- **E/N:** indicates how many experts (E) and novices (N) took part of the evaluation of the current system. The use of the expert term may be misleading since some papers report eye-tracking experts and others report typing-system experts. For simplicity, experts are treated as one group. Furthermore, participants are considered novices unless it is stated that experts took part in the study.

- **Sessions:** is used to indicate the number of sessions of the experiment. Studies employing more than one session indicates average error rate (ER) and WPM for the first and the last session, respectively.
- **WPM:** is the text production speed of the current system measured in words per minute (one word equals 5 characters). If results are available both for experts and novices they are reported with a divider (E/N).
- **Error Rate (ER):** indicates the minimum string distance measured in percentage (%).

It should be noted that it is a difficult task to cover all aspects of a typing system. Features such as subjective preferences and advanced text metrics are, for example, not covered in the table. Rather, the table helps to get a clear overview of the systems presented in the literature.

A table inspection indicates that *method* does not have an effect on word-production (WPM). For example, systems based on the *Direct* method spread between $1 - 19.9$ WPM and systems based on *CPG* spread between $3.4 - 19$ WPM. A closer inspection of the table reveals that other aspects should be considered to find plausible reasons for the huge variation in WPM. Interestingly enough, typing systems that employ a predictive model are not necessary faster than other systems (spread between $2.9 - 19$ WPM). There are only two criterion that show an effect on participants' skills during an evaluation and that is E/N (experts/novices) and *sessions*. The table clearly shows how experts perform better than novice users, which may be attributed to familiarization with the experimental interface and technology. This is further supported in longitudinal studies where novice users show significant improvements over several experimental sessions. There are other factors that seem to have a general effect on the efficiency. One potential candidate is the systems' theoretical maximum WPM. The internal dynamics of a typing system will always have a theoretical maximum WPM that cannot be exceeded, even by expert users. For some systems, this limit is reached much faster than others.

Most of the systems presented in the review are experimental and not designed as full featured type-to-talk systems. Essentially, this means that

Table 6.1: *Review of gaze-based typing systems that have been presented in the literature. The survey does not include subjective ratings nor advanced text-typing metrics. Fields marked with a N/A (not available) indicate that the information was not provided by the authors. Finally, checkmarks and crosses indicate true and false, respectively.*

Appears In	Year	Name	Eye Tracker	Method	Design	Predi.	Audio	E/N	Sessions	ER(%)	WPM
Morimoto and Amir [116]	2010	Unnamed	Own (Low Cost)	DG	Context Switch	✗	N/A	0/6	8	15.3 – 6.6	6.6 – 13.6
Ashtiani and MacKenzie [3]	2010	BlinkWrite2	EyeTech TM3	Switch	Multitap	✓	✓	0/12	1	0 (almost)	4.7
Barrett et al. [5]	2009	StarGazer	ITU Gaze Tracker	CPG	Pan and Zoom	✗	✓	4/3	1	2	3.4
Majaranta et al. [99]	2009	Unnamed	Tobii 1750	Direct	QWERTY	✗	✓	0/10	10	1.3 – 0.4	6.9 – 19.9
Porta and Turina [126]	2008	Eye-S	Tobii 1750	DG	Hotspots	✗	N/A	0/8	1	N/A	6.8
MacKenzie and Zhang [97]	2008	Unnamed	ViewPoint	Direct	QWERTY	✓	✓	5/5	1	11.8	11.7
Hansen at al. [42]	2008	StarGazer	Tobii 1750	CPG	Pan and Zoom	✓	✓	0/48	1	12.6	3.5
Hansen at al. [42]	2008	StarGazer	Tobii 1750	CPG	Pan and Zoom	✓	✓	3/4	1	1.2	8.2
Wobbrock et al. [178]	2008	EyeWrite	Tobii 1750	DG	Hotspots	✗	N/A	0/10	14	24 – 7	2 – 4.9
Huckauf and Urbina [54]	2008	pEyeWrite	Eyelink2	DG	Pie Menus	✗	N/A	1/3	1	N/A	12.3/7.9
Bee and André [11]	2008	Quickwriting	iView X RED	DG	Hotspots	✗	N/A	0/3	1	N/A	5
Bee and André [11]	2008	eyeKeyboard	iView X RED	Direct	QWERTY	✗	N/A	0/3	1	N/A	8
Tuisku at al. [162]	2008	Dasher	Tobii 1750	CPG	2D Zoom	✓	N/A	0/12	10	33.1 – 4	2.5 – 17.3
Joos at al. [74]	2007	EyeGaze	EyeFollower	Direct	QWERTY	✗	N/A	0/4	5	21.5 – 7.5	2.3 – 4.3
Joos at al. [74]	2007	GazeTalk	EyeFollower	Direct	Restricted Keyboard	✗	N/A	0/4	5	36 – 30	1 – 1.5
Urbina and Huckauf [163]	2007	IWrite	Eyelink2	DG	Alphabetical	✗	✓	6/10	1	N/A	11.4/7.6
Urbina and Huckauf [163]	2007	StarWrite	Eyelink2	DG	Alphabetical	✗	✓	6/10	1	N/A	8.4/5.9
Urbina and Huckauf [163]	2007	pEYEdit	Eyelink2	DG	Pie Menus	✗	✓	6/10	1	N/A	10.9/6
Majaranta et al. [100]	2006	Unnamed	iView X RED	Direct	QWERTY	✗	✓	0/13	1	0.54	7
Hansen et al. [45]	2004	GazeTalk	QuickGlance	Direct	Restricted Keyboard	✓	✓	0/12	2	N/A	5.8 – 6.1
Miniotas et al. [109]	2003	Symbol Creator	EyeLink	DG	Latin Cursive	✗	N/A	2/4	5	14 – 3.4	5.7 – 8.6
Miniotas et al. [109]	2003	Unnamed	EyeLink	Switch	Multitap	✗	N/A	2/4	5	5 – 3.1	6.3 – 8.9
Ward and MacKay [174]	2002	Dasher	QuickGlance	CPG	2D Zoom	✓	N/A	2/2	12	23.5 – 4.5	9.1 – 19
Charlier et al. [19]	1997	Visionboard	Own	Direct	N/A (Full Alphabet)	✗	N/A	0/15	1	N/A	9[a]
Istance at al. [62]	1996	Text Keypad	N/A	Direct	Alphabetical	✗	N/A	0/5	1	N/A	1
Hutchinson et al. [56]	1989	Unnamed	ERICA	Direct	Restricted Keyboard	✓	N/A	1/0	1	N/A	2.9[b]

[a]The authors claim that the users on average typed 0.5 - 1 character per second, which corresponds to 6 - 12 WPM.
[b]The WPM calculation is based on an expert writing a full page (estimated to 250 words) in 85 minutes.

designers leave out certain features such as opportunities for editing the produced text and support for speech synthesis. Finding the optimal design for a fast gaze-based tying system is a challenging task. One key feature of a new system seems to include a high theoretical WPM (e.g., +25 WPM) in the initial design. From the review, it is clear that an evaluation should be longitudinal in order to acquire optimal learning effects. This will allow users to be familiarized with the interaction and the mapping of the interactive keys which will in turn allow for a better and more realistic mental image of the system interface.

6.3 Summary

Human communication is fundamental for everyone and gaze-based communication prevents loneliness and social exclusion. A well-designed system further allows a user to communicate with the surroundings and the outside world through internet browsing and emailing. Typing systems based on gaze interaction can be divided in into four techniques with very different properties: *Switch*, *Direct*, *Discrete Gestures (DG)* and *Continuous Pointing Gestures (CPG)*. Throughout the literature, several of these techniques have been deployed on various types of systems with very different outcomes in terms of writing speed. Although the most commonly used technique for commercial systems are based on the *direct* method, probably the most intuitive method for gaze interaction is based on continuous pointing gestures as seen in the Dasher system and StarGazer, which will be presented in the following Chapter 7.

Even though most gaze-based communication systems allow for individual personalization the usual typing speeds seldom exceeds 10 WPM, which is significantly different from the normal face-to-face speech of about $100 - 250$ WPM. Gaze-typing application are appealing to researchers for several reasons: (1) it represents a well-defined task space, (2) there is a lot of room for improvement in terms of efficiency. Based on the survey the optimal efficiency of a typing interface does not seem to rely on input technique, rather it seems to depend on a theoretical high WPM followed by multiple training sessions.

Part IV

Novel Zoom Strategies in Gaze Interaction

7 Noise Tolerant Interface

Several of the different input methods for communication applications presented in Chapter 6 does not solely apply to communication tools. In fact, these methods are used for several types of applications such as drawing, internet browsers, gaming and interface control; this can, for example, be seen in the taxonomy by Vickers [169]. Computer control by gaze was covered thoroughly in Chapter 5 where different designs and methods of compensating for inaccurate input were covered. This chapter presents a case study where a novel gaze-based interface called StarGazer is examined; it enables exploration of graph-based data, which has been specifically designed to have a high tolerance to noise. StarGazer has in this case study been adapted to work as a communication tool.

7.1 Case Study: StarGazer

The StarGazer case study is presented in the following chapter, which addresses issues related to interaction with graph structured data and applications (e.g., gaze typing systems) using low resolution eye trackers or small-size displays. The study shows that it is possible to make robust selections using a noisy tracker even with a large number of selectable targets on the screen. The study extends the knowledge of techniques based on *Continuous Pointing Gestures* (CPG) that was presented in Section 6.2. As was shown in Chapter 4, low-cost eye trackers afford lower accuracy and precision compared to the more expensive interactive eye tracking systems. The use of low-cost eye trackers on standard displays or high-end eye trackers on small displays are analog and require equivalent design considerations. The main motivation for this study is to explore a technique for gaze-based interaction

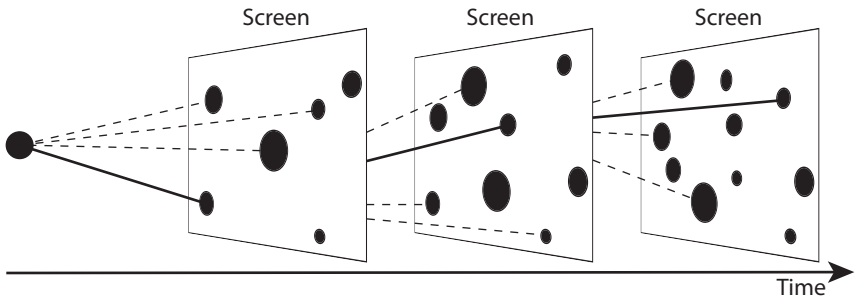


Figure 7.1: *The process of making selections on a monitor displayed as a graph over time. The dashed lines illustrate the possible paths to selectable objects on the screen and the thick lines illustrate actual selections.*

using panning and zooming in a 3D environment. Furthermore, the interaction technique should be able to reliably run on low- and high-end eye trackers and work under noisy conditions (e.g., wheelchair vibrations, drifting).

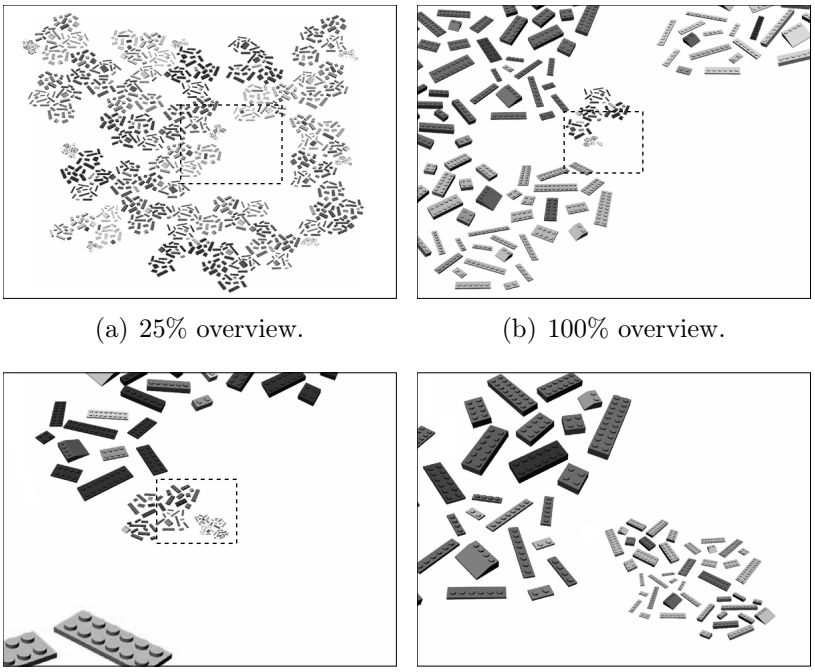
Selections based on the *direct* method (e.g., dwell) have for many years been preferred by end users. Table 6.1 indicates how eye-typing applications that employ dwell-time activation produce typing speeds anywhere between 1 to 20 words per minute (WPM) where five characters (including space) constitutes one word. During dwell time activations the user waits for the activation to complete, which is a waste of time since it impacts the number of possible selections, within a given time period.

The process of selecting objects over time in any gaze-based modern user interface can be illustrated as in Figure 7.1 where different activation areas are presented to the user depending on the state of the system. An on-screen keyboard for example, shows the selectable keys in a fixed grid and each activation area (i.e., button) can be selected. Some types of activations may even change the state of the system by revealing new selectable objects in the interface.

7.1.1 Navigating Information Spaces

The idea of using the eyes for navigational purpose is not new. In 1981, Bolt [14] presented his work on Gaze-Orchestrated Dynamic Windows. Here, the idea was to allow for dynamic interaction with multiple windows in a large display. Although at that time, the technology did not offer support for a real-time version of the system, the work became an important cornerstone for later research within the field. Fono and Vertegaal [35] presented a similar idea for gaze-based selection called *zooming windows* and their results indicated how interaction with zoom is up to 30% faster than with static windows. The flexibility and intuitiveness of zoomable interfaces to present and visualize data have been shown by Bederson and colleagues [10, 9] and Wijk and Nuij [166] and Bates and Istance [6], Møllenbach et al. [114]. For an example see Figure 7.2.

In this study, zoom is employed within the full window of the application. That is, uniform scaling of the information space will increase the spatial separation of all the selectable targets and thus exclude targets that are not in focus. The zooming process helps to facilitate an unambiguous selection of a target of interest. In case of noise, targets need more separation for reliably performing selections as seen in Figure 7.3. Additional zooming can accommodate this at the cost of additional selection time. Since uniform zooming maintains the geometric relation between the objects in the information space the non-distortive behavior helps users to maintain familiarity with the information space. Furthermore, during selection, the zooming helps to naturally indicate the level of confidence while approaching the target of interest as opposed to dwell-based selections where users have to passively wait for a selection. In other words, any selections with zooming are based on smooth pursuit tracking and selections based on dwelling consist of a traditional point-and-select task. Although zooming in full-screen may sound appealing there is at least one issue that needs to be addressed and that is the loss of contextual information during zoom. Miniotas et al. [110] share the same concern and warn against the permanent costs of spatial expansions. Because of the consequence of loosing contextual information, where objects are hidden outside the zooming region, layouts need to place objects in familiar structures (such as alphabetical letters).



(a) 25% overview. (b) 100% overview.

(c) 500% overview. (d) 2400% overview.

Figure 7.2: *The control in zoomable interfaces are intuitive and allow for dynamic interaction with objects located at different depths in the information space. The current example was designed for visual inspections in information-dense environments. Copyright E. Møllenbach. Used with permission.*

7.1.2 The StarGazer Interface

This section presents StarGazer, a noise tolerant zoomable interface designed for flexible interaction even with significant noise imposed on the input. Navigating the multi-scale interface is done through panning and zooming. This means that StarGazer offers support for browsing large tree-structured data sets.

Full-Knowledge Pan and Zoom

Interacting using pan and zoom is key in navigating StarGazer. Pan and zoom only offer control in three out of six possible degrees of freedom (forward/back, up/down, left/right and not pitch, yaw, roll), this combina-

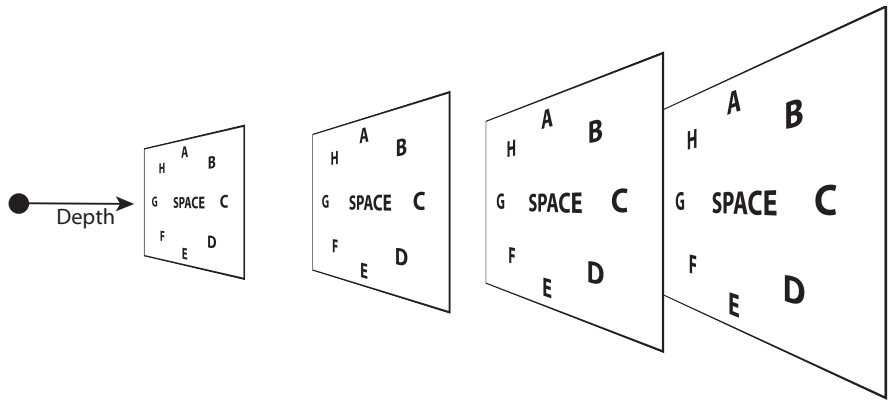


Figure 7.3: *Zoom provides uniform scaling of the information space by increasing spatial separation of selectable targets. Placing targets deeper in the informations space allow for more separation, which is useful in noisy conditions.*

tion does not provide true 3D navigation but it is adequate for this system. StarGazer is a full-knowledge system, which means that all information about the interactive elements is visible and therefore known. This information can help to compensate for the limited accuracy of gaze pointing. A force-field method similar to the technique described by Zhang et al. [183] is implemented in StarGazer and is beneficial in at least two ways: (one) the instability of the cursor is counteracted and (two) increased confidence in selections is achieved by locking-on to the object closest to the point-of-regard. However, pan and zoom may also have drawbacks, e.g., it might lock on to a false object and zoom into it, which makes it difficult to return to the object of interest.

Zooming. With zooming in gaze attentive interfaces certain regions of an interface are disregarded while focus is directed to the areas that correspond with the current point of regard. Zooming into regions of interest filters out less important information and helps to allocate more space to regions of interest. Consequently, this allows for an easy distinction and selection of objects. Figure 7.4 illustrates a full selection from the initial starting point until the moment after the selection. The example demonstrates the process of zooming in towards a letter while filtering out areas of no relevance.

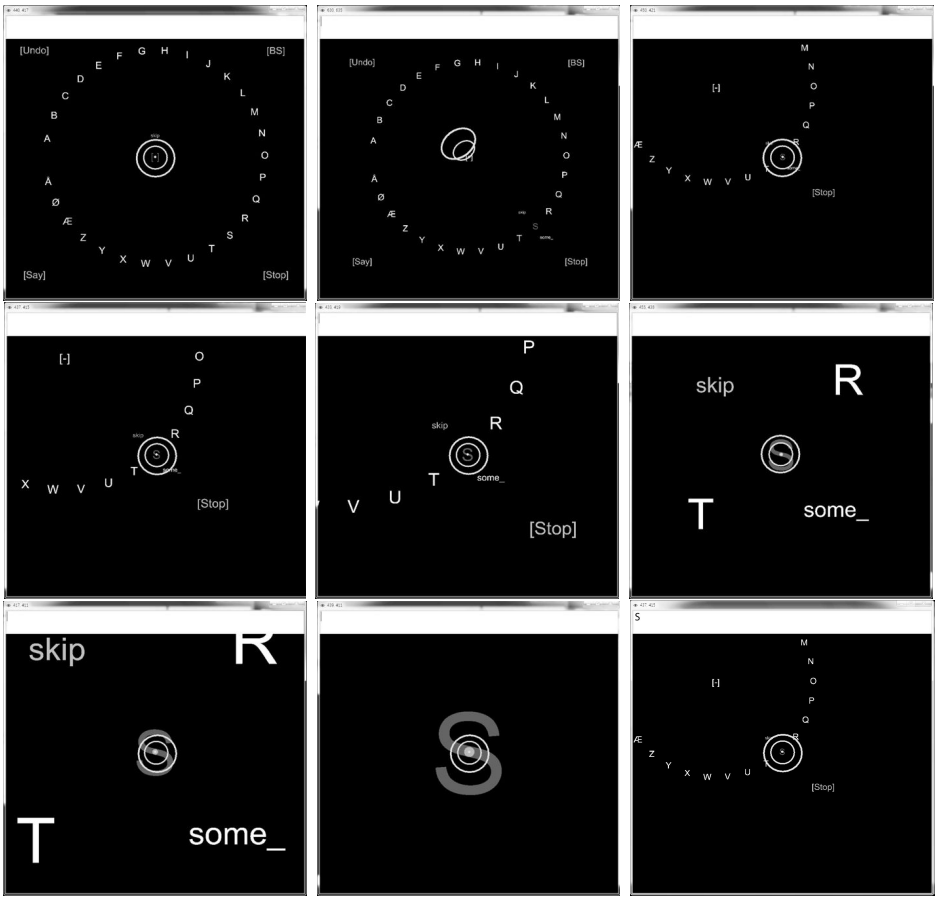


Figure 7.4: *The example illustrates a selection of the letter "S". During zoom, regions of low probability are filtered out until a unambiguous selection can be made. The final image (bottom right) indicates the new starting position for the subsequent selection.*

Zooming in StarGazer helps to separate objects over time and additionally allows the inclusion of context dependent information. This principle is well known in semantic zooming, where data relevant to a particular scale is shown [51]. Figure 7.4 shows how additional information such as word predictions have been implemented in the application and demonstrates how the zoomable interface is able to show context dependent information without any explicit mode selections. Alternatively, the additional information could have been multiple word predictions, setup functionality (e.g., speed, speech synthesis) and alternate modes (e.g., capital letters, special characters).

Panning. Panning is a planar translation that allows horizontal and vertical navigation on the same scale level. If the interaction only relies on zooming, the navigation will not allow for exploring context on the same level. The panning process helps to shift objects of interest towards the center of the screen while zooming. Figure 7.5 shows the interface of StarGazer overlaid with a scalable mask defining when zoom and pan are applied. Only panning will occur if the point of regard is located outside the zoom window. Zooming and panning is activated when the point of regard is located inside the zoom window. If navigating towards an unwanted object (e.g., letter), the user can avoid the object either by navigating towards a different object or by selecting the *skip* object (see Figure 7.4), which takes the user back to the default starting position. The potential amount of scaling and size of the zoom window depends on the amount selectable objects and the noise over the system.

The type of eye movements performed to make a selection are smooth pursuits where an object is fixated upon while it moves towards the center. This process exploits the inherent human ability to track objects. Activations are issued by the system when the object with the most confidence intersects with the activation plane (as seen in Figure 7.6).

The dynamic control of StarGazer needs to handle large coarse movements i.e., saccades and smooth pursuit movements. During the initial starting state of a selection when the user scans after the object of interest saccades occur. The system allows for fast planar translations until the target is fixed upon and moves towards the center of the screen at which time the selection process becomes a smooth pursuit and pan velocity is reduced. This

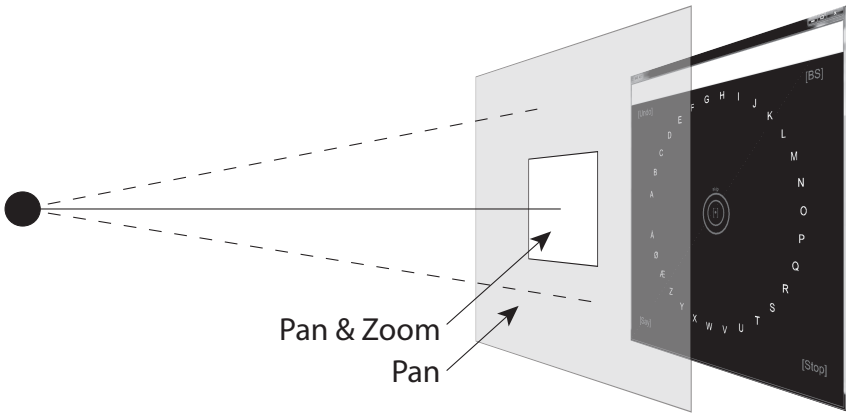


Figure 7.5: *Areas of StarGazer performing pan and zoom in the displayed window. The areas would in other applications be dependent on the density of selectable objects and the current noise levels.*

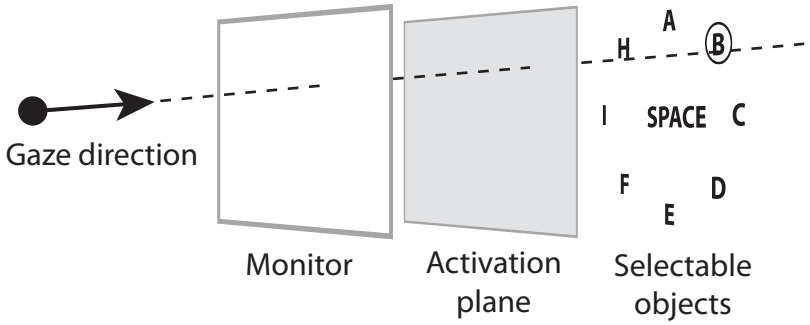


Figure 7.6: *The gaze direction is used to move the most likely target of interest into the center of the screen for selection.*

is important since the user adds confidence to a selection initially through a region, then a group of objects, and finally, to a specific object. The pan velocity, V_p is adapted to the point of regard and the distance, t from the center of the screen. The current version of StarGazer uses a logistic sigmoid function similar to Equation 7.1, where $m = 0$, $a = 50$ $n = 1$, $\tau = 0.1$ and $v_{min} = 9$. The parameters are all chosen values based on empirical testing.

$$V_p(t) = a\frac{1 + me^{-t/\tau}}{1 + ne^{-t/\tau}} + v_{min} \tag{7.1}$$

Feedback

Majaranta et al [100] emphasize the importance of feedback in gaze-based typing. StarGazer supports three types of feedback: audio, object coloring and a directional pointer. The object with the most confidence (i.e., most likely object to be selected) is colored by the system. If the user proceeds in an unchanged direction the object will be selected. The white directional pointer consists of three concentric circles placed in different depths shows the pan direction (see e.g., second image in Figure 7.4). Finally, when a selection is completed a subtle 'click' sound. The gaze input is not manipulated in a filter i.e., not smoothed during interaction since the interaction in StarGazer requires a fast and responsive input. However, the direction pointer is smoothed for stable display purposes only.

Type-to-Talk With StarGazer

StarGazer has a built-in type-to-talk functionality that offers support for an external word processor and speech synthesis. The *speech* layout provides a new functional object called *Say*. When activated it reads the produced text aloud and sends the produced text to the MS Word instance (see Figure 7.7). Although the application does not offer support for text editing, the word processor is useful for caregivers. Since StarGazer is able to run in a small display it can easily run without occupying much screen real-estate.

7.2 Evaluating StarGazer

The aim of the tests is to reveal the potentials of StarGazer in terms of its simplicity for novice users, its ease of use when applied to noisy low-cost gaze

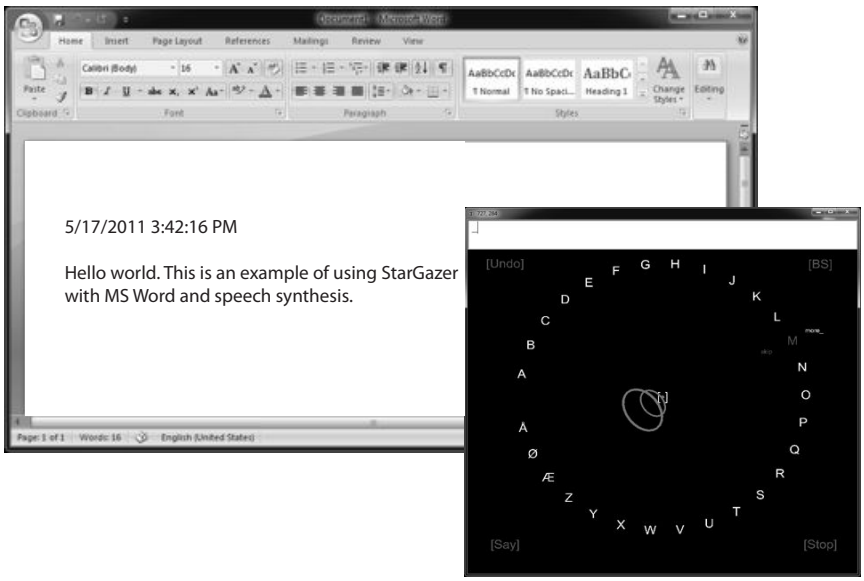


Figure 7.7: *The built-in support for MS Word and speech synthesis allows users to use StarGazer as a type-to-talk system.*

trackers or when used on small displays. The main question sought answered by this evaluation was: Is it possible for the users, when using gaze only interaction, to pan and zoom in 3D environments displaying a well-known data structure? This evaluation will be performed over three dfferent types of experiments: name writing task, latency typing task and a speed-typing task.

7.2.1 Name Writing Test

The purpose of the first test, was to examine whether the layout of StarGazer and its use of pan and zoom for navigation was intuitive enough so that people can use it after a brief introduction and without prior gaze interaction experience. The possible effect of noise and screen size were important factors to examine.

Methodology

An external program was designed to add uniform noise within a given radius to the current cursor location at a certain update rate and with a given latency. The uniform noise resembles a combination system-inherent, oculomotor and environmental noise (see Chapter 3), which reduces the precision of the estimated gaze position significantly. In the noise conditions the radius was set to 100 pixels (corresponds to 2.5° on the screen) and an update rate of 60 Hz. The noise application served as imposing more noise on an already noisy gaze tracker. StarGazer was tested on a gaze tracker in three different display sizes: small (240×320 pixels - PDA size), medium (640×640 pixels) and large (1280×994 pixels) with and without noise applied to the cursor. In total, this sums up to six different conditions.

Participants. In total, 48 subjects (32 male, 14 female) volunteered to participate in the test. All subjects had normal or corrected vision. None of the participants in this study had any experience with StarGazer and only few had tried a gaze tracker before the test. All tests were conducted with the same predefined settings (zoom and pan speed) of StarGazer.

Apparatus. A Tobii-1750 eye tracker (Tobii Industries) was used as an input device, consisting of a flat panel monitor with integrated infrared light sources and a camera. The resolution of the LCD display was 1280×1024 pixels. The eye tracker sampled at 50 Hz accuracy of 0.5 degrees of visual angle (claimed by the manufacturer). Without word predictions enabled the system uses less than 1% of the CPU time, leaving the computations for other purposes (e.g., the gaze tracker). All users performed a standard five-point calibration on the Tobii system before each experiment and performed recalibrations as needed (if the cursor was not responding to the users' eyes). Software developed in the laboratory was used to analyze the data.

Measurements. Efficiency was measured in words per minute (WPM) as is common for gaze typing and writing applications [101]. Error rate (ER) and remaining errors calculated as minimum string distance (MSD) [95] is measured. The ER is calculated as the percentage of backspace and undo selections compared to the characters produced. The MSD is defined as the number of manipulation steps that is needed to obtain the target string, which for this task is a correctly spelled name.

Design. The name-writing task employs a between-subject 2×3 factorial design. The factors are imposed noise (yes/no) and screen size (large, medium or small). Each subject only experienced one of the six combinations.

In order to promote learning by rote, all letters were placed on two concentric circles in letter groups in the 3D plane (see Figure 7.8). The space object ([-]) is located in the center of the screen. Four functional objects are placed in the corners of the display: [Undo], is used to undo the last action performed by the user, [BS] is used as the traditional back-space button on a keyboard, [Speed] enters a new menu where zooming speed can be adjusted and [Stop] is used to signal to the system when the test is over.

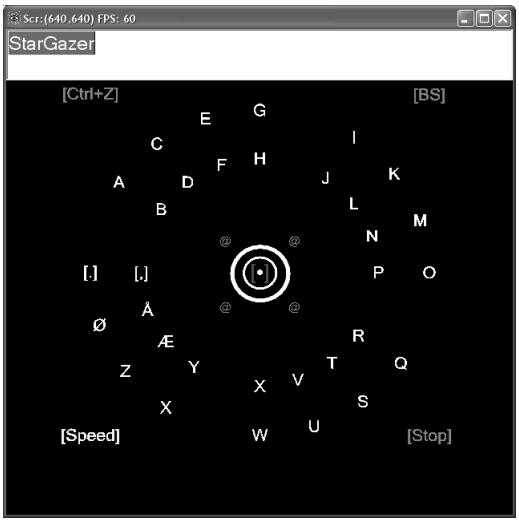


Figure 7.8: *Layout of the evaluated version of StarGazer. Letters are arranged in groups to promote rote learning.*

Task Description. The participants were seated approximately 50 cm from the screen and asked to type their name (given name and family name) into the setup window of the system with the conventional keyboard. Then, the participant rolled a normal die to decide which one of the six conditions to be tested on. After calibrating on the system, the participants were given a presentation of StarGazer followed by a two-minute period to get acquainted with gaze interaction and StarGazer. After 2 minutes, the participants were

asked to type their names as quickly and accurately as possible. The test ended simply by selecting the [Stop] object placed in the lover right corner of the display. During the experiment, the language model and word completion was disabled resulting in an optimal keystroke per character (KSPC) of 1.0 for error-free writing.

Results

The average amount of text generated by the participants were 13.2 (SD = 2.7) characters of text and the grand mean over the six conditions were 3.47 (SD = 1.42) WPM.

A two-way ANOVA showed a main effect from noise $F(1, 47) = 6.23$, $p < 0.05$ and a main effect from size of display $F(2, 47) = 20.50$, $p < 0.001$. The data shows no interaction effect between size and noise. A Bonferroni post hoc test shows a significant difference between the noise-free condition (M = 3.83 WPM) and the noisy condition (SD = 3.10 WPM, $p < 0.05$). The post hoc test also reveals that the WPM for the smallest screen size (M = 2.17 WPM) is significantly different from both the medium size (M = 3.85 WPM, $p < 0.0001$) and the large size (SD = 4.38 WPM, $p < 0.001$). The difference between the large and the medium size is not significant. Figure 7.9 summarizes the results from the name-writing experiment.

The grand mean of corrected errors (ER) is 12.6% (SD = 21.0%). The two-way ANOVA analysis reveals a main effect from display size $F(2, 47) = 3.68$, $p < 0.05$, but not from noise. The mean ER for the noise-free condition is 12.3% and the mean ER with noise added is 13.0%. The data shows no interaction effect between noise and size. The mean ER for large display size is 5.6%, which is significantly different from the mean ER for small displays (M = 23.9%, $p < 0.05$) but not from the medium display (M = 8.48%) in a Bonferroni post hoc test. The grand MSD mean is 0.18 (SD = 0.45). The ANOVA shows no effects from display size or noise on MSD.

7.2.2 Latency Writing Test

This small experiment evaluated the maximum latency (delay) that could be imposed on the input signal while having the user still be able to type on the system. The control signal was progressively delayed to simulate latency in

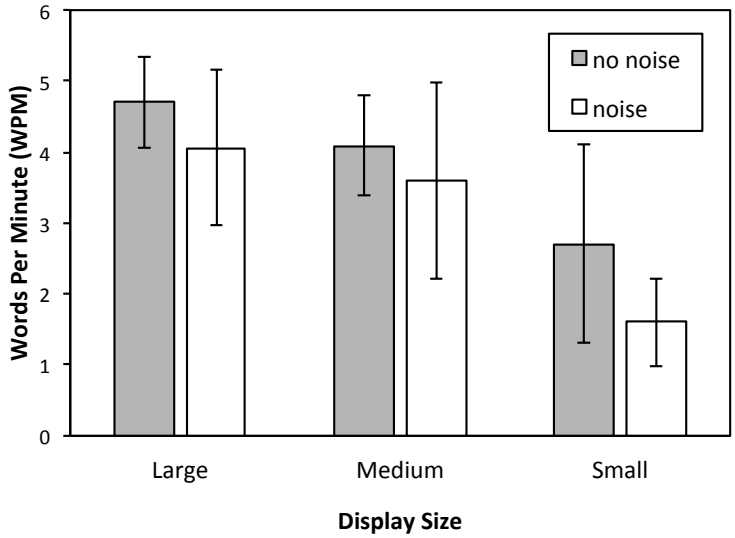


Figure 7.9: *The Figure shows the test results in WPM with and without imposed noise on the different display sizes. Error bars show standard errors of the mean.*

the gaze tracker and no additional spatial noise was added in this test. In this small experiment, three eye-tracking experts (two male and one female) participated in the study and the experiment was conducted on the Tobii-1750 eye tracker. Results were quantified in WPM.

The results can be verified in Figure 7.10. A close inspection indicates that users are able to write text on StarGazer with a latency of up to about 200 ms. A 400 ms latency seems to be the limit for producing text on the system (i.e., users were able to slowly write their names). As a passing remark, none of the participants enjoyed latencies above 200 ms and most felt nauseous with latency levels above this.

7.2.3 Speed Writing Test

In the name-writing test and the latency-writing test, users were not able to adjust the zoom speeds according to their own individual preferences. This experiment allowed users to write their names as fast as possible. The test was conducted on seven participants (six male and one female) where three has prior experience with StarGazer. The experiment was conducted on the

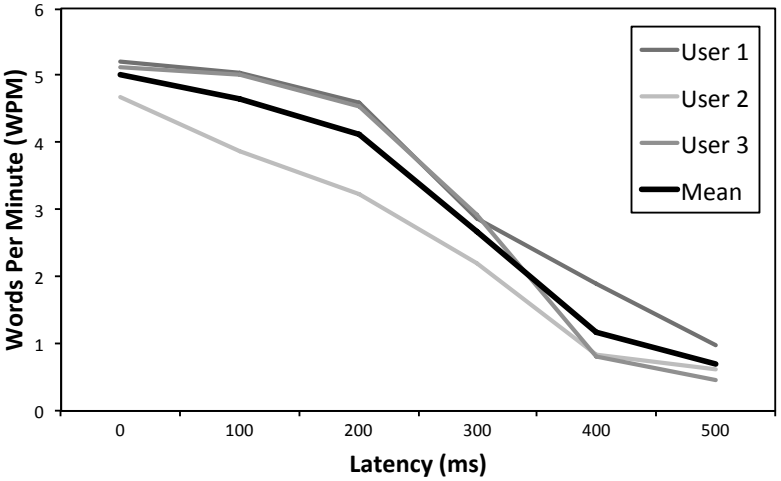


Figure 7.10: *Performance when latency is introduced. The mean performance is displayed with a solid line and the performance of each of the three participants is displayed with gray lines.*

Tobii-1750 eye tracker. The participants were instructed to write their names and allowed to adjust and experiment with the zoom on a full size display (1280×994 pixels). The participants adjusted the zoom speed to a level that subjectively gave them the best performance (optimal speed and few errors). After calibration on the eye tracker the participants were allowed five minutes of practice on StarGazer to find the optimal settings that they were comfortable with.

Results

The tests show that the subjects were able to achieve an average writing speed of 8.16 WPM ($SD = 0.98$) and a mean ER of 1.23% ($SD = 3.43$). In these test the remaining errors (MSD) are zero for all subjects.

7.2.4 Discussion

Name Writing Test. 48 participants were able to write their name with very few errors (cf. $MSD = 0.18$) and without any prior knowledge about eye tracking or StarGazer. This confirms that it is easy to understand and

learn the simple navigation principle of StarGazer. When noise was imposed, it slowed down the writing speeds since the cursor activated several objects or even forced the focus outside the point of regard window resulting longer selection times per character. However, during the test users did not loose their orientation and were in control throughout the test. This is also reflected in the error rate between the noisy and noise-free conditions with less than one percent in difference (12.6% versus 13.0%). This indicates that the overall design goal of designing a noise tolerant system was met.

A large display improves the overall feeling of being in control since the selectable objects are more separated, compared to smaller displays. This is also confirmed by the results. Compared to the smaller display, the selectable objects in the large display ensured that the amount of corrective pan movements are fewer because the object grows faster than the accuracy threshold.

The grand mean of 3.47 WPM this may seem relative low compared to other gaze-typing systems (see Table 6.1). Although the mean from the condition with a large window and no imposed noise shows 4.7 WPM, which is within an acceptable performance window for a gaze typing system. Furthermore, the test did not employ any use of character or word prediction, which in turn could have improved on the performance.

This system stands out from most other system since it employs an innovative writing interface that can be operated by novice users without any previous experience with either gaze interaction nor the interaction technique. The results of the name-writing test were useful for indicating whether the interface could be used immediately. In order to measure learning effects or optimal writing speeds, a longitudinal study needs to be employed. The large display obtained an error rate of 5.6%, which compares well to the study by Hansen et al. [45] where novice users were typing large on-screen keyboards with an error rate of 4.29%. The PDA size obtained an error rate of 23.9%, which translates into about two errors per name string. The error rate on the PDA size display is most likely not acceptable for general use. However, participants were able to achieve a basic control of the interface.

Latency and Speed Writing Tests. The participants showed that they were able to confidently type with a delay of 200 ms without any significant loss of efficiency (i.e., WPM). Even with the use of the force-field technique the interface response was characterized by severe side-, over- and under-shootings caused by the progressional increased delay of the input signal.

In the case of the speed writing test, the seven participants were able to increase the WPM levels considerably with only five minutes of practice. In fact, the average writing speed seemed to fall somewhere in the middle of all the systems presented in Table 6.1 and despite the higher zooming speeds the error rate seems to remain in the lower end of the table.

General Discussion. The evaluation of the StarGazer interface has shown that erroneous selections are at a minimum when employing panning and zooming. The novice users became sufficiently acquainted with the basic gaze-based navigation within a few minutes, which indicates that users easily construct a mental model of the system. This could be attributed to the clear feedback with easily understandable color coding and a subtle 'click' whenever selections were completed. After each selection the users were brought back to the initial state, which makes it easy for them to reorient themselves.

One clear advantage of StarGazer compared to static layouts is that the spatial separation between targets can be controlled according to the precision of the current eye tracker. This means that the spatial separation of targets can be uniformly increased and decreased for low and high precision, respectively. Also, a clear advantage of StarGazer is that increased noise levels do not increase the selection error rate; although it did slow down the activation speed. The inherent problem with zoom entails that objects are located outside the zooming region and are thus rendered beyond the visible scope of the interface. The objects could also be acquired by only using pan, but this would take a relatively long time when used on a low scale (i.e., visible targets are close to the activation plane). The only way of dealing with this problem in StarGazer is to activate the "return-to-start" object that is available next to all objects in focus (indicated by a "@" symbol), which brings the user back to the initial view.

The control system in StarGazer proved to be easy to learn and understand

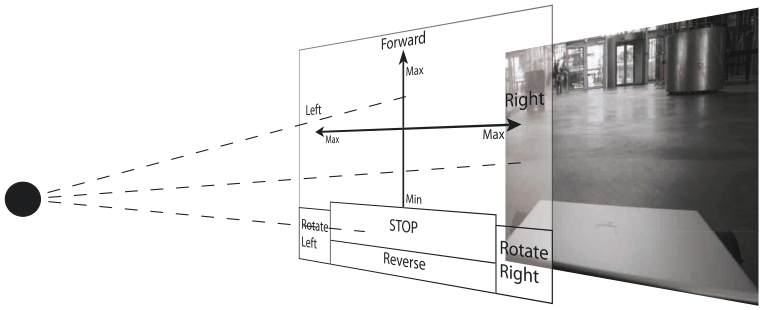


Figure 7.11: *The control functions used for the robot study as seen through the robot's video stream. Continuous pointing gestures are used to navigate the robot in the environment.*

for the participants. Although fine grained smooth pursuits seemed to be affected by the internal dynamics of the tracker's fixation detection algorithm. Consequently, the tracker seemed to artificially transform these movements into a series of fixations and saccades, making tracking tasks more difficult (see Section 8.3.3 p. 145 for further details).

Although all object locations in StarGazer are known, this may not always be the case. Continuous pointing gestures in a pan and zoom interface could also be employed in interfaces with no information about the objects in the information space. The consequence of this is that navigation may be hampered by the inaccurate input from the eye tracker. This was exemplified in a study where a robot, representing a wheelchair was controlled using a gaze controlled interface overlay placed on a video stream. The stream came from the robot camera and the navigation was achieved by a look-where-you-want-to-go metaphor in 3D space (i.e., movements are based on the gaze direction). This work was presented in [160] and was later adopted in Wästlund et al. [176]. An overlay window on top of the front-view video stream was used to control a robot. The initial design provided a direct feedback loop with no interface components displayed to the user (see Figure 7.11). The point-of-regard and the gaze position had to continuously adjust to the locomotion of the robot. The X-axis controlled the steering and the Y-axis controlled speed. The fastest and most obvious way of breaking was to look away from the screen. The type of navigation explored in this study is analog to the pan and zoom metaphor presented in the StarGaze case study. Looking in

the center and towards the top screen indicates confidence and results in the robot driving at full speed (i.e., robot moves in the Z-direction). Looking left and right respectively help to bring the robot on the right path since the point-of-regard moves gradually towards the center of the screen (i.e., panning and zooming). The additional functions help the user to make narrow turns and even put the robot in reverse (i.e., zoom out).

7.3 Summary

The StarGazer writing system allows novice users to use the system immediately and users were able to write on the system even with a significant time delay. It is safe to claim that StarGazer is able to work under noisy conditions, which was demonstrated by the results of the evaluation. The consequence of writing with a noisy input does not seem to manifested itself in a higher error rate, but it seems to slow down the selection time per character, which is a direct consequence of correcting errors caused by the noisy input.

The StarGazer case study has been focusing on the CPG paradigm. The pan and zoom interface of StarGazer is used for displaying and selecting data under conditions where the gaze tracker accuracy may be low. With gaze controlled pan and zoom selection the user is not forced to look at a button for a fixed time interval to select the object, and thus avoids the annoyance of wasting time. It should be noted that smooth pursuit selection has its own time frame that may be longer that a simple click or saccade. Furthermore, the interaction facilitates regretting before a selection is made without breaking the flow. It has been shown how it is possible to operate StarGazer with a significant latency of about 200 ms and still be able to type at acceptable rates. In the speed text the participants were able to produce text with more than eight words per minute without any use of language modeling. StarGazer is an intuitive 3D interface for gaze navigation, allowing more selectable objects to be displayed on the screen than the accuracy of the gaze trackers would otherwise permit.

This chapter has focused on the design of an interface for exploring graph-based visualization structure in a typing task. The following chapter explores tools to facilitate target selections with special focus on small targets that

normally cannot be accessed without effectively increasing the target area.

8

Tools to Access Small Targets

Gaze interaction affords hands-free control of computers. Pointing to and selecting small targets using gaze alone is difficult because of the limited accuracy of gaze pointing. This chapter presents the first experimental comparison of gaze-based interface tools for small-target (e.g., $< 12 \times 12$ pixels) point-and-select tasks.

Point-and-select operations are mandatory for interaction with modern graphical user interfaces. For example, in order to open an application in a windowed environment, users need to move the cursor to the position of the application icon and issue an activation when the cursor is on top of the icon. These point-and-select operations are relatively easy to carry out by users with good hand control using a conventional computer mouse. The mouse introduces virtually no noise between the movement of the hand and the on-screen cursor, and mouse users are able to point to and select even very small targets (i.e., $< 12 \times 12$ pixels) in a windowed environment.

Chapter 5 demonstrated how a computer could be controlled with the gaze alone and gaze in combination with other modalities. Point-and-select operations were introduced and several methods to accommodate for the Midas Touch problem such as dwell time and dedicated selection areas. The purpose of this chapter is to compare interface tools designed to facilitate point-and-select tasks with small targets when using gaze alone, and find the advantages and limitations of each tool. First, the main limitations of gaze input are presented and review some of the approaches that have been used to address these limitations. Then, two experiments comparing the performance of dwell, magnification and zoom methods in point-and-select tasks with small targets are described. Finally, the *Zoom Framework* together with

a final experiment is presented.

8.1 Designing for Gaze Interaction

Point-and-select operations can be divided into a pointing component and a selection component. During pointing, the user moves the cursor to the target area and, during selection, the user issues an activation at the location of the desired target. When using a conventional mouse, pointing is accomplished by physically moving the mouse on a mouse pad, while activations are issued by pressing the mouse button.

Gaze tracking is well suited for pointing in human-computer-interaction tasks because humans naturally tend to direct their eyes to the objects that they are manipulating. That is, it is relatively safe to assume that computer users will look at a target at which they are pointing [64]. When users look at locations or targets at which they do not want to point, these false alarms (i.e., unintentional pointing) have no serious negative impact on interaction. Another advantage of gaze pointing is that it is faster than mouse pointing (e.g., Mateo et al. [104]). In contrast, gaze tracking is not as well suited for selection because humans tend to look at objects of interest to explore them independently of their intention to select the objects [64]. Therefore, it is not safe to assume that users want to select every object at which they look. False alarms (i.e., unintentional selections) that result from making this assumption can have serious negative impact on interaction.

In summary, gaze input can be characterized as a fast and natural pointing method with two important limitations: gaze input is not well suited to infer user's selection intent and gaze pointing has limited accuracy (as previous seen earlier chapters). The next section presents approaches that have been used to address these two limitations.

8.1.1 Inferring User's Selection Intent

When a mouse is used as the input, the break between pointing and selection is unambiguous: the selection component begins when the mouse button is pressed and, while the button is not pressed, the movement is always considered part of the pointing component. Users do not move the cursor

to every object they inspect, but only to those they want to select. In contrast, when gaze is used as the input, the cursor follows the eyes both to objects users only want to inspect and to those they want to select and it is more difficult to identify the break between pointing and selection. At least three methods have been proposed to address this issue: dwell, blink, and multimodal methods. Dwell is the most commonly used method in gaze-interaction systems. For example, gaze-typing applications (e.g., Johansen et al. [71], Majaranta et al. [99]), menu selections (Tobii [161], Špakov and Miniotas [172]), and drawing programs (Hornof and Cavender [52]) employ dwell selection. Given that the interface tools studied in this chapter rely mostly on dwell-based methods, dwell will be emphasized in this section. However, it is important to note that other methods could also be used in combination with the interface tools to assist small-target selection presented below.

8.2 First Experimental Setting: Interface Tools That Address Limited Accuracy

This subsection introduces tools designed to facilitate point-and-select tasks with small targets by compensating for the limited accuracy of gaze pointing. These tools can be divided into full-knowledge and no-knowledge tools. Full-knowledge tools rely on information about the interactive world in which they act to compensate for the limited accuracy of gaze pointing. For example, these tools may use information about the location of selectable targets to infer that users are attempting to point to a certain target when they miss slightly but there are no other targets around. In contrast, no-knowledge tools do not need information about the interactive world in which they act to facilitate point-and-select tasks.

It must be noted that even though full-knowledge tools can be helpful when used with a custom-designed application, they cannot help when interacting with mainstream operating systems that do not provide this information (e.g., Windows). In these cases, no-knowledge tools are necessary. Yet, full-knowledge tools may provide important insights and inspire features to facilitate point-and-select tasks with small targets, even if they are not feasible for access to current mainstream operating systems and methods. For

example, accessing the underlying environment, variations of snapping, the act of catching objects, are often sufficient to support a user's task [8]. The bubble cursor [39] is another example of a full-knowledge tool. This tool dynamically resizes its activation area depending on the proximity of surrounding targets and only allows one target to be selected at any time. In Grossman and Balakrishnan's experiments, the bubble cursor significantly outperformed the standard cursor. Laukkanen et al. [89] later proposed two variations of the bubble cursor. The lazy bubble cursor, which used a less aggressive resizing to reduce the visual distraction resulting from the expanding cursor, and the cone cursor, which included a tail to the last enveloped target until the next target was enveloped, always leaving a target selected. In Laukkanen et al.'s experiments, the bubble cursor performed slightly faster (although with a higher error rate) than the lazy bubble and cone cursors. In the subjective ratings, users preferred the bubble and lazy bubble cursors.

Zhang et al. [183] introduced three methods for modulating the cursor trajectories to counteract the eye jitter and instability of gaze pointing: force field, speed reduction, and warping to target center. The force-field method created a force point in the target center so that, while within the target area, the cursor was attracted toward the target center. The speed-reduction method was a variant of the force-field method in which, in addition to using the target center as a force point, cursor speed was reduced while within the target area. The goal of the speed-reduction method was to prevent eye jitter from unintentionally interrupting dwell time by moving the cursor outside of the target area. The warping-to-target-center method moved the cursor to the target center as soon as the cursor entered the target area and held it there while gaze was within the target area. Zhang et al.'s evaluations showed that force field and speed reduction significantly alleviated the instability of the eye cursor, improving performance in dwell-based gaze pointing tasks. However, warping to target center did not help, most likely due to the dramatic unnatural movement of the eye cursor.

In cases where information about the underlying environment is unavailable, full-knowledge tools cannot work and no-knowledge tools are necessary. One of the most popular no-knowledge tools to deal with limited accuracy of gaze pointing is the magnification method [87]. During the first step (of this two-step method), the user looks at the region of the screen in which the target is located and, after the dwell time has elapsed, an enlarged version

of the region pops up in a new overlapping window. During the second step, the user points to the desired target and makes the selection in this enlarged region with another dwell selection. An activation in the magnified window is easily mapped back to the original target area by a simple transformation. The increased effective size of the target during the second step facilitates pointing to the desired target. Thus, point-and-select performance using the magnification method is more robust towards tracker noise or calibration offset than using dwell alone. Even though most commercial systems (e.g., Tobii and QuickGlance) have adopted this method to facilitate point-and-select tasks with small targets, no empirical evidence exists to support its effectiveness. Lankford [87] only provided anecdotal evidence to support his method. From a usability standpoint, point-and-select tasks take longer using magnification than using dwell alone (due to the double point-and-select process). Therefore, using magnification to carry out point-and-select tasks may be perceived as frustrating by users. In addition, they inherit the limitations of dwell selections (e.g., Midas Touch, unnaturalness). These usability issues provide a good argument for trying out other potentially faster methods offering similar levels of accuracy.

Zoom tools that gradually increase the workspace (or a portion of it) as if approaching the user are potential alternatives to the magnification method. Bederson and colleagues [10, 9] and Wijk and Nuij [166] showed the flexibility and intuitiveness of zoomable interfaces to present and visualize data. Bates and Istance [6] demonstrated the potential of zoom for gaze interaction. Using full-screen zoom, gaze input showed improved performance and usability when compared to the standard dwell (without zoom). In addition, gaze-input performance was comparable to head-input performance. However, full-screen zoom can lead to loss of contextual information, which can be problematic (e.g., Pook et al. [124]). An alternative to full-screen zoom is a fish-eye lens expanding targets in both display and motor space. However, the distortive nature of the fish-eye lens seemed to have an adverse effect on targeting tasks (Gutwin [40], Ashmore et al. [2]).

This chapter introduces a zoom tool that presents a window around the user's direction of gaze in which a smooth animation shows the content of the window gradually increasing in size for the duration of a predetermined zoom time. During this time, the user can make online corrections (proportional to the current level of magnification) to the cursor position. After

zoom time has elapsed, the element on the center of the zoom window is selected. An important advantage of zoom tools (over magnification tools) is that the level of magnification is not limited by the size of the window in which magnification occurs or of the display used during interaction. Hansen et al. [42] successfully used a similar method for gaze typing using a full-knowledge application with three-dimensional space. The work developed in the present study extends this method to no-knowledge situations.

In addition to extending zoom tools to no-knowledge situations, the study empirically evaluates the effectiveness of the magnification tool to facilitate point-and-select tasks with small targets. Two experiments involving point-and-select tasks with target sizes comparable to the smallest interactive elements found in window environments (or even smaller, in some cases) to compare the performance of magnification and zoom methods are presented. Before these experiments, a pre-test in which participants pointed to and selected these small targets without any interface tool (i.e., dwell alone) was conducted. This pre-test was included to ensure that the targets were indeed too small to be reliably selected using dwell alone. In addition, pre-test data provided a baseline against which the effectiveness of interface tools to facilitate point-and-select tasks with small targets could be compared.

In the first experiment, the user was presented with one target at a time, whereas in the second experiment, the user was presented with multiple targets at the same time to simulate an extreme real-world scenario in which the user had to locate a target among multiple selectable objects (i.e., distracters). Both magnification and zoom can be used in no-knowledge situations. However, these experiments were conducted in a full-knowledge environment to gain experimental control, minimize technical difficulties, and test the combination of no-knowledge tools and full-knowledge tools (in experiment 2).

8.3 Experiment 1 & 2: Evaluating Novel Zoom Tool

8.3.1 Method

Participants

Six participants (3 male, 3 female) were recruited from a Danish university campus. Participants ranged from 27 to 37 years (M = 30 years). All of them were computer users reporting from 3 to 7 hr/day (M = 6 hr/day) of computer use. None of them had prior experience with eye tracking. All participants had normal or corrected-to-normal vision.

Apparatus

The software application used to present the targets was created using C# and ran on an IBM 1.86 GHz Intel Dual Core desktop computer. A Tobii-1750 eye tracker (Tobii Industries) was used as an input device, consisting of a flat panel monitor with integrated infrared light sources and a camera. The resolution of the LCD display was 1280 $\times$ 1024 pixels. The eye tracker sampled at 50 Hz accuracy of 0.5 degrees of visual angle (claimed by the manufacturer). Roughly, this corresponds to 20 pixels sitting at the distance from the monitor at which participants sat in the experiments (i.e., 60 cm). All users performed a standard calibration before each experiment and performed re-calibrations as needed (if the cursor was not responding to the users' eyes). Software developed in the laboratory was used to analyze the data.

Initiating Selections in the Experiments

Both magnification and zoom methods are designed to be used with a fixation-detection algorithm (i.e., dwell method). Because most commercial eye trackers use proprietary algorithms, it is difficult for researchers to evaluate the impact of algorithms on their results. Given the lack of information about the algorithms used by the Tobii eye tracker and to avoid their potential impact on the results, several necessary precautions was taken. Since the system offers no support for raw gaze points, the setting with less smoothing

was used. The inferred point of gaze was therefore based on the close-to-raw sample points with no additional smoothing applied. Furthermore, the tracker's fixation detection was de-activated and a simple keyboard activation was used instead. Thus, participants in the experiments initiated the activation process by pressing the space bar. However, all pointer movement (i.e., aiming) was controlled by the eyes. The space-bar press was only used to signal the break between pointing and selection components. Although this combined method is not strictly equivalent to a pure fixation-detection algorithm, it can be regarded as a good substitute completely under the user's control and transparent for the data analysis.

Design

A full factorial within-subjects design was employed in both experiments. Independent variables manipulated in both experiments included: activation method, target size, and activation time. In addition, snap feature was manipulated only in experiment 2. To minimize asymmetric learning effects, activation method, target size, and activation time were counterbalanced using a balanced Latin Square. Dependent measures included total pointing time and hit rate.

Independent variables. The primary independent variable was activation method. For experimental purposes, this variable had two levels: magnification and zoom. However, a pre-test with dwell alone was conducted to ensure that the target sizes used were small enough to be impossible to reliably point to and select using dwell method. Results from the pre-test were used as a baseline against which was used to compared the results of magnification and zoom in experiments 1 and 2. In the pre-test, a $1.7° \times 1.7°$ (68×68 pixels) red square appeared on top of the crosshair representing the participant's point of regard when an activation was initiated (i.e., space-bar press). This square gradually decreased in size for the duration of the dwell time, until its disappearance signaled the issue of an activation at the crosshair location (see row 1 in Figure 8.1).

In the zoom condition, the initiation of an activation (i.e., space-bar press) made objects within a $7.5° \times 7.5°$ (300×300 pixel) square window surrounding participant's point of regard gradually increase in size as if approaching the user. The participant's point of regard was represented by the crosshair

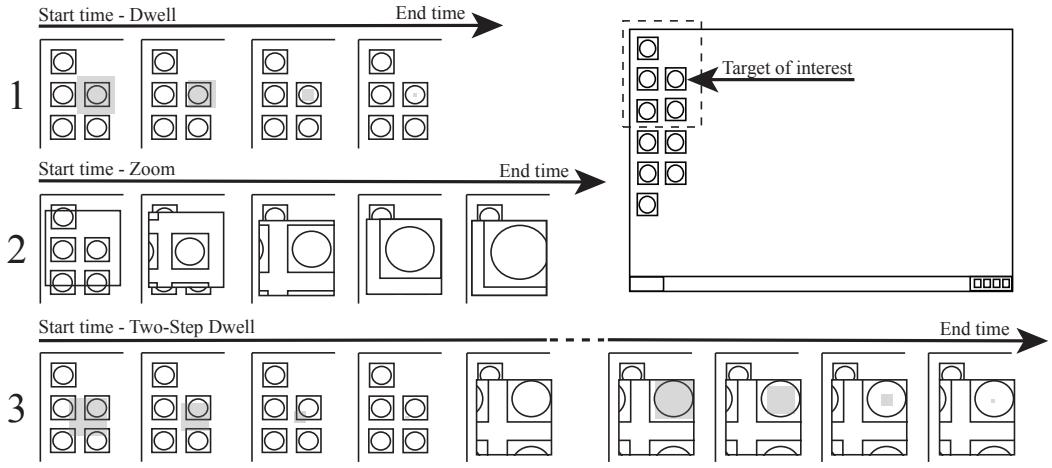


Figure 8.1: *Illustration of target selections with the three selection methods. Row 1 shows dwell activation, row 2 shows zoom activation, and row 3 shows magnification activation.*

in the center of the zooming window. The increase in size followed an s-shaped sigmoid function for the duration of zoom time. The sigmoid-shaped function was chosen based on pilot tests prior to the experiment. When modeling the size increase of objects in the zooming window as if they were approaching the user at a constant speed (i.e., linear function), object sizes increased rapidly during the last instants of zoom time. (Although this was not anticipated, this phenomenon is well known in the optic-flow literature; e.g., Jagacinski and Flach [69, Ch. 22]). The sigmoid-shaped function allowed users to interact with bigger objects for longer, giving them more opportunities to make online corrections. When zoom time elapsed, an activation was issued in the center of the zoom window which had, by then, reached a 10x magnification (see row 2 in Figure 8.1).

Although it may seem counterintuitive at first, 10x was chosen as the magnification level for zoom to facilitate the comparison between zoom and magnification. Whereas a 5x magnification level results in a 5x magnification for more than half of the trial (i.e., the entire second step), a 5x zoom magnification level would result in less than 5x magnification for the duration of the trial. Given that the effectiveness of zoom relies on the ability to make online corrections while the target is growing and that maximum magnification

level is not limited by window size, a 10x magnification was chosen to allow users to take advantage of zoom while keeping the (maximum) magnification level within a reasonable limit.

In the magnification condition, a 2.5° × 2.5° (100 × 100 pixel) window surrounded the participant's point of regard and a shrinking red square identical to the one used in the pre-test signaled the initiation of the first activation (i.e., first space-bar press). The participant's point of regard was represented by the crosshair in the center of the magnification window. When dwell time elapsed, the square disappeared and, instead of issuing an activation, a 12.5° × 12.5° (500 × 500 pixel) square window with a 5x magnification instantly appeared on that location. The same shrinking square signaled the initiation of the second activation (now on the magnified window) and, when dwell time elapsed, an activation was issued at the crosshair location (see row 3 in Figure 8.1). This is very similar to the way magnification tools are visualized in commercial gaze tracking systems.

Both magnification and zoom required a pre-specified activation time (i.e., dwell or zoom time). Because magnification involves performing two point-and-select operations (one per step), it uses two dwell times (see Dependent Measures below for more detail). In order to be able to compare magnification and zoom, zoom times were twice as long as dwell times in magnification conditions. In experiment 1, three zoom times were used (1000 ms, 1500 ms, 2000 ms) and three dwell times (500 ms, 750 ms, 1000 ms). In experiment 2, a 1000-ms zoom time and a 500-ms dwell time were used.

Target size was also manipulated in both experiments: targets were squares with 0.15°, 0.225°, or 0.30° (6-, 9-, or 12-pixels) long sides for the setup. In addition, in experiment 2, snap feature was manipulated. The snap feature assumes that interaction occurs in a full-knowledge world (i.e., the position of all interactive elements is known). Two slightly different versions of the snap feature were implemented. The first version was similar to the warping-to-center method used by Zhang et al. [183]. That is, the participant's cursor moved to the center of the closest element (i.e., target or distracter) and stayed there until the participant's gaze moved closer to a different element. Then, the cursor "jumped" to that element center. In the second version, the cursor also moved to the center of the closest element but in this case, rather than "jumping", the cursor moved in a smooth fashion to the closest element

center. Therefore, snap feature had three levels: snap without smoothing, snap with smoothing, and snap off.

Dependent measures. Two dependent measures was used in the experiments: total pointing time and hit rate. To obtain total pointing time (τ), the time to complete each trial was divided into the time to complete the pointing component (i.e., pointing time, PT) and the time to complete the selection component (i.e., selection time, ST). Determining total pointing time in dwell (pre-test) and zoom trials was straightforward: it was the time elapsed from the beginning of the trial to the space-bar press (see Equation 8.1). Selection time was equal to dwell or zoom time and, therefore, it was fixed within condition and determined by the experimenter.

$$\tau_{dwell,zoom} = PT + ST \tag{8.1}$$

Magnification trials involved performing two point-and-select operations. Therefore, in this case, there were two pointing times and two selection times (see Equation 8.2). To obtain total pointing time (τ_{mag}), the pointing time for step 1 (PT_1) and the pointing time for step 2 (PT_2) was added together. Total selection time ($ST_1 + ST_2$) was equal to two times the dwell time. Selection times were not used as dependent measures because they were completely controlled by the experimenter.

$$\tau_{mag} = (PT_1 + ST_1) + (PT_2 + ST_2) \tag{8.2}$$

Successful activations (i.e., those falling within the region of the target) were considered hits. Hit rate was the number of hits over the total number of trials. Auditory feedback informed users about the outcome of their activation: a pleasant sound signaled a hit and a warning sound signaled an unsuccessful activation.

Procedure

All participants received a demonstration of the different activation methods by the experimenter. At the beginning of each trial, a squared target of 1.7°

$\times$ 1.7° (home square) appeared in the center of the screen. As soon as participants issued an activation at the home square by pressing the space bar, a red target with a black cross appeared on a grey background at a different location in the workspace. Participants were instructed to point to and select the target as fast and accurately as possible.

In the pre-test and experiment 1, targets appeared one at a time in random order at 1 of 10 possible locations arranged in a circular layout around the home square (as described in ISO 9241-9; see Figure 8.2 and Figure 4.2 right-side). Starting each trial from the home square ensured that movements covered approximately the same distance (8.2° or 330 pixels) in all trials. Each participant completed 3 blocks of 10 trials using the dwell method. All participants completed the blocks in the same order, starting with the biggest targets (easiest) and ending with the smallest targets (most difficult).

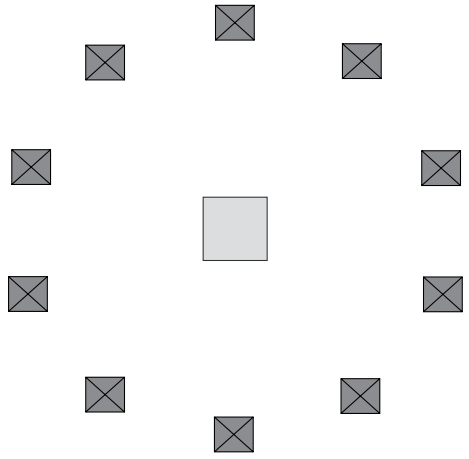


Figure 8.2: *Layout of the discrete task with 10 locations where targets could appear in the pre-test and experiment 1. Targets appeared one at a time. The home square is depicted in the center.*

Experiment 1 was conducted immediately after the pre-test. Each participant completed 18 blocks of 10 trials (i.e., 180 trials) in the same circular layout as the pre-test. Activation method (2), target size (3), and activation time (3) were fixed within blocks. Each participant completed the 18 blocks in a different order, randomly selected without replacement. Immediately af-

ter experiment 1, participants reported which of the two activation methods was perceived as fastest, most accurate, easiest to use, and most fatiguing.

In experiment 2, each trial also started from the center on the display and each participant completed 18 blocks of 10 trials (i.e., 180 trials). Activation method (2), target size (3) and cursor manipulation (3) were fixed within blocks. However, instead of appearing on an equidistant circular layout, targets appeared in a random position in a large grid located 100 pixels away from the edges of the monitor (see Figure 8.3). Although distance varied from trial to trial, average distances were not significantly different across conditions (the average distance per trial was 9.8° or 393 pixels). In addition to the target (a blue square), 1999 icons known from the Windows operating system were presented as distracters. Although the solid blue color of the target was significantly different from the distracters, the purpose was to maximize visual distractions in the process of locating and selecting the target. In each block, all the distracters and the target were of the same size (i.e., 0.15°, 0.225° or 0.30°). A subset of the interactive elements is shown in Figure 8.3. This experiment also introduced the snap feature described above.

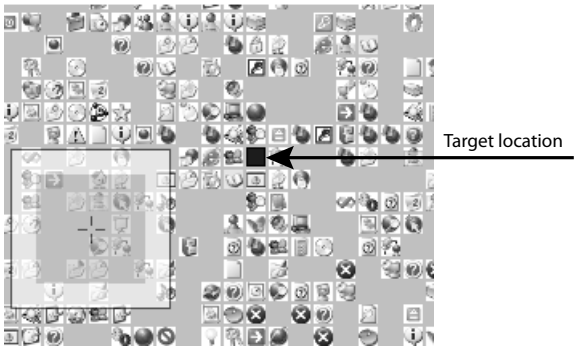


Figure 8.3: *A subset of the visual distracters presented to the user in experiment 2. The dark-square target is indicated by the pointing arrow (not shown during experiments). The transparent window shows the current location of the pointer and the inner red square shrinks gradually with dwell time.*

8.3.2 Results

Outliers Removal and Data Analysis

The total time to complete a trial was used to identify outliers. Trials with total times that were 3 standard deviations above the mean total time (for that experiment) were excluded from data analysis. This led to the removal of 5 trials in the pre-test, 19 trials in experiment 1, and 16 trials in experiment 2. The data was analyzed for each experiment using a repeated measures analysis of variance (ANOVA) and Bonferroni correction was used in the post-hoc tests.

Pre-Test

The ANOVA showed a main effect of target size on hit rate, $F(2, 10) = 4.21$, $p < 0.05$. As expected, hit rates with dwell were higher for the largest targets ($M = 0.28$, $SD = 0.16$) than for the smallest targets ($M = 0.07$, $SD = 0.12$). The low hit rates found in this pre-test (i.e., average hit rate of 0.28 for the largest targets) support the hypothesis that alternative methods are needed to reliably point to and select these targets.

Experiment 1

Hit rate. Analyses showed a main effect of activation method, $F(1, 4) = 76.24$, $p < 0.001$, and target size, $F(2, 8) = 6.92$, $p < 0.05$, on hit rate. Hit rates for magnification were higher ($M = 0.88$, $SD = 0.11$) than those for zoom ($M = 0.44$, $SD = 0.23$). In addition, hit rates were higher for the largest targets ($M = 0.78$, $SD = 0.02$) than for the smallest targets ($M = 0.58$, $SD = 0.04$). Figure 8.4 shows the hit rates for the two activation methods and the three target sizes. Target size had an effect on hit rate for both activation methods. Activation time showed no effect on hit rate, $F(2, 8) = 1.44$, $p = 0.29$. Therefore, the different activation times can be treated as repetitions.

Total pointing time. Analyses showed a main effect of activation method, $F(1, 4) = 39.58$, $p < 0.01$, and target size, $F(2, 8) = 6.04$, $p < 0.05$, on total pointing time. Total pointing time was shorter for zoom ($M = 1561$ ms, $SD = 894$ ms) than for magnification ($M = 2829$ ms, $SD = 1213$ ms). In addition, total pointing time was longer for the smallest targets ($M = 2583$

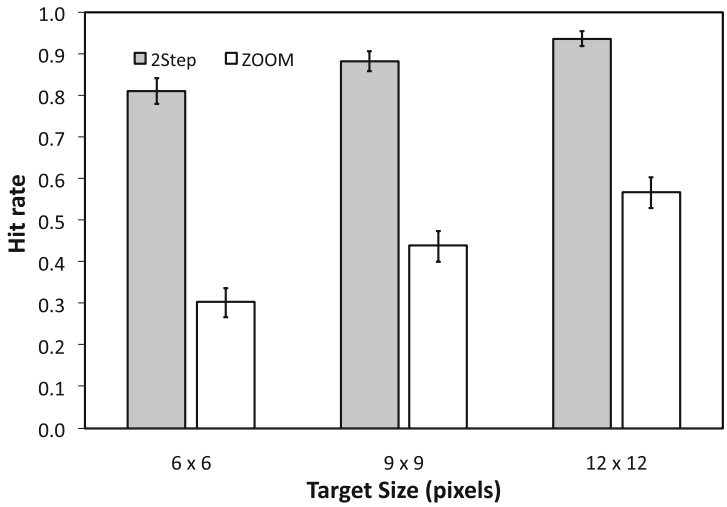


Figure 8.4: *Hit rates for the two activation methods and the three target sizes. Both magnification and larger targets resulted in higher hit rates than zoom and smaller targets, respectively. Error bars show standard errors of the mean.*

ms, $SD = 584$ ms) than for the largest targets ($M = 1891$ ms, $SD = 328$ ms). Figure 8.5 shows the total pointing time for the two activation methods and the three target sizes. Target size had an effect on total pointing time for both activation methods. Activation time showed no effect on total pointing time, $F\ (2, 8) = 0.912$, $p < 0.44$.

Subjective preference. In spite of the fact that the data indicated that the zoom method was faster than the magnification method, most participants perceived magnification method to be faster. Participants accurately perceived the magnification method as more accurate than the zoom method: the average hit rate for magnification was twice as good (i.e., 0.88 vs. 0.44) as the hit rate for zoom. The majority of the users perceived magnification method as easier to use than the zoom method, but surprisingly, magnification was also perceived as more fatiguing.

Experiment 2

An initial analysis showed that the two versions of the snap feature (i.e., with and without smoothing) were equivalent to each other both in terms of hit

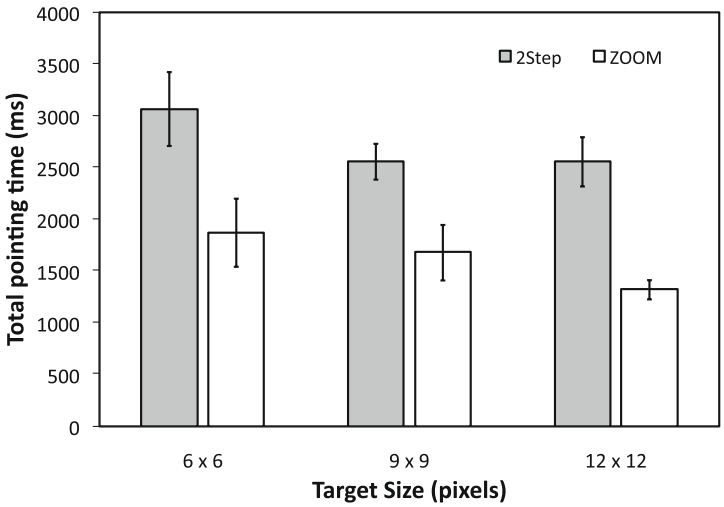


Figure 8.5: *Total pointing time (in milliseconds) for the two activation methods and the three target sizes. Both zoom and the largest targets resulted in shorter pointing times than magnification and the smallest targets, respectively. Error bars show standard errors of the mean.*

rate, $t(5) = 0.89$, $p = 1.00$, and pointing time, $t(5) = 2.17$, $p = 0.25$. In order to facilitate interpretation of the results, the two snap versions was combined into a snap-on condition and this was compared to the snap-off condition. The results below correspond to the analysis using these two levels of snap feature.

Hit rate. Analyses showed a main effect of activation method, $F(1, 5) = 98.05$, $p < 0.001$, target size, $F(2, 10) = 5.9$, $p < 0.05$, and snap feature, $F(1, 5) = 14.03$, $p < 0.05$, on hit rate. Specifically, hit rates were higher for the magnification ($M = 0.86$, $SD = 0.15$) than for zoom ($M = 0.46$, $SD = 0.1$). Hit rates were higher for the largest targets ($M = 0.68$, $SD = 0.06$) than for the smallest ones ($M = 0.61$, $SD = 0.056$). Hit rates with the snap feature on ($M = 0.72$, $SD = 0.08$) were higher than hit rates with the snap feature off ($M = 0.60$, $SD = 0.06$). Figure 8.6 shows the average hit rate for snap feature on and off and the two activation methods.

Total pointing time. Analyses showed a main effect of activation method

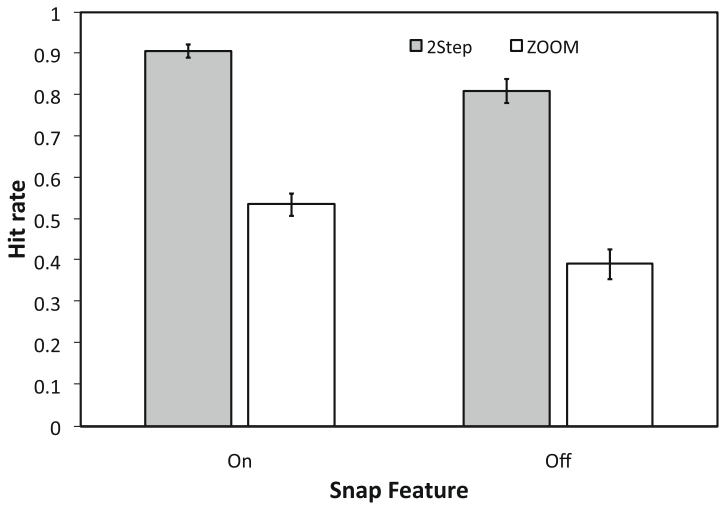


Figure 8.6: *Average hit rate for the two activation methods with snap feature on and off. Both magnification and snap on resulted in higher hit rates than zoom and snap off, respectively. Error bars show standard errors of the mean.*

on total pointing time, F (1, 5) = 41.00, p $<$ 0.001. Zoom was faster (M = 2002 s, SD = 420 s) than magnification (M = 2878 s, SD = 294 s). Neither target size, F (2, 10) = 2.18, p = 0.16, nor snap feature, F (1, 5) = 1.24, p = 0.32, showed any effect on the total pointing time.

8.3.3 Discussion of Experiment 1 and 2

The aim of this study was to compare methods designed to facilitate point-and-select tasks with small targets when using gaze input. In a pre-test, participants carried out point-and-select tasks with small targets using dwell alone. Results from this pre-test showed very low hit rates, suggesting that the targets used were too small to be reliably pointed to and selected using dwell. Experiments 1 and 2 obtained much higher hit rates using magnification and zoom, suggesting that both of these activation methods can facilitate point-and-select tasks with small targets when compared to dwell. However, there were also important differences between these two activation methods.

Overall, the zoom method was faster, but less accurate, than the magnification method. The extra time needed to perform point-and-select tasks

using the magnification method is likely to be related to the need to perform two point-and-select operations and, therefore, two pointing movements. In contrast, only one pointing movement is necessary when using the zoom method and then, during activation (zoom) time, users can perform smaller online corrections as the target is gradually increasing in size.

Even though participants took twice as long to perform point-and-select tasks with the magnification method as they did with the zoom method, they perceived the former to be faster than the latter. The source of this distortion of time perception is unclear. One possibility is that it resulted from the higher cognitive load associated with using the zoom method (see below). Another possibility is that the lower hit rate obtained with zoom may have influenced the participants' subjective report (even if the question did not relate to accuracy). Providing some type of feedback in the zoom condition about the time remaining until activation may reduce cognitive load and help estimate the duration of zoom activations.

In terms of accuracy, the magnification method may have benefited from the greater control users have in this condition when compared to the zoom method (and not due to the nature of zooming per se). During pilot testing, an attempt to implement a zoom condition in which the user was able to stop and reverse zooming using gaze patterns alone. This approach was discarded, however, because of how difficult it was for the user to control effectively. Future studies should explore ways to provide users of zoom tools levels of control comparable to those provided by the magnification method.

In spite of the apparent naturalness of the zoom method, users seemed to find continuous interactions rather problematic when compared to the discrete magnification method. That is, pointing to and selecting targets seemed easier when the target remained still (i.e., magnification) than when the target moved in reaction to the user's gaze movement (i.e., zoom). To some extent, this may have been an artifact of the zoom animation. That is, the zoom method may have transformed the point-and-select task into a tracking task in which the user had to follow a moving target, rather than pointing to a still target. Performing tracking tasks using gaze input is particularly problematic given that current algorithms only recognize fixations and saccades. Because they are not developed to deal with the smooth pursuit movements characteristic of tracking tasks, current trackers artificially transform these

movements into a series of fixations and saccades, making tracking tasks more difficult. In addition, the greater similarity of the magnification method to users' prior discrete experience in human-computer interaction may have influenced the evaluation of this method as easier to use than the zoom method. Yet, the fact that most participants perceived the magnification method as more fatiguing than the zoom method supports the notion that the increased accuracy of magnification comes at a cost.

In experiment 2, the snap feature resulted in higher accuracy (but similar speed) than no snap feature. This suggests that, given access to the underlying world, full-knowledge tools can be combined with no-knowledge tools to facilitate point-and-select tasks with small targets. It is worth pointing out that the large number of distracters in close proximity used in the study may have negatively affected the effectiveness of the snap feature. For example, there is a possibility that the snap feature may have unintentionally dragging the cursor to elements of no interest to the user. In real-world situations, there is usually more separation among elements and less distracters overall, making this "unintentional drag to undesired element" less likely to occur. In the future, it may be interesting to test how these variables affect the effectiveness of the snap feature. In addition, redefining how interactive elements relate to each other and letting these elements work in a more discrete manner may improve the effectiveness of snap feature.

In the implementation, it was not anticipated how the continuous zoom tool would change the task or how delay would affect performance. Empirical results challenged the assumption that continuous interaction would always be more natural than discrete interaction. Instead, continuous interaction seemed unnatural with delayed feedback. In fact, the manual-control literature suggests that, in the presence of delays, users naturally adopt a *move-and-wait* strategy [33]. That is, users transform the continuous task into a series of discrete components. Ironically, the attempt to make the task more natural backfired because, even though continuous interaction may be more natural in real-world situations, discrete interaction is more natural in the presence of time delays.

8.4 Second Experimental Setting: Discrete Zoom Tools

Based on the results of the first studies, a *discrete zoom tool* was designed, which is conceptually equivalent to an *n-step tool*, combining features of two-step and zoom tools (see row 3 of Figure 8.7 for an illustration). Because zooming occurs in discrete steps, this tool is expected to be more tolerant to delay than the continuous zoom tool. When compared to the two-step tool, more steps to permit greater magnification levels are expected because, after the first step, the content can be magnified further without increasing window size. Obviously, adding steps can also slow down performance. However, given that early steps require lower accuracy than the two-step tool, discrete zoom is expected to accommodate lower dwell times. Finally, the discrete zoom tool is also expected to result in more of a zooming sensation than two-step while providing users more control over zooming rate than continuous zoom. Obtaining the total pointing time for a discrete section is formalized in equation 8.3 where a series of pointing times and selection times are added together in n small steps.

$$\tau_{discrete} = \sum_{i=0}^{i<n} PT_i + ST_i \tag{8.3}$$

The zoom tool is expected to have at least four advantages over the two-step tool. First, its continuous looming appearance is expected to feel more natural to the user. Second, the user is expected to be able to make on-line corrections to the cursor position as the target increased in size. Third, target selection is expected to be faster because the user would not need to perform two separate point-and-select operations. Fourth, the maximum magnification level possible is expected to be greater than using a two-step tool with a window of similar size because the entire region around the cursor did not need to be magnified all at once.

8.4.1 The Zoom Framework

Based on the experience developing and testing tools to facilitate the selection of small targets using gaze alone, a conceptual framework to organize existing

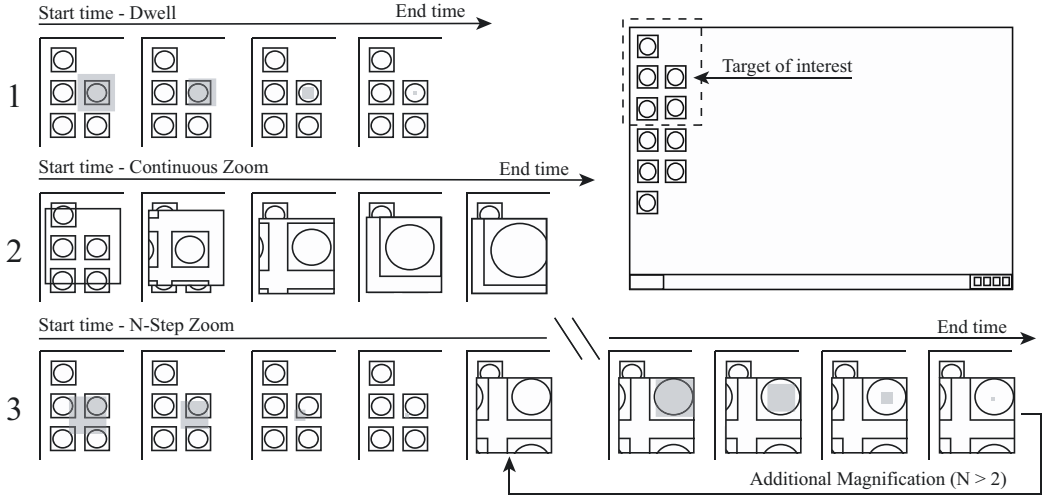


Figure 8.7: *IIllustration of the different zoom tools. Row 1 depicts a target selection with dwell (i.e., no tool). Row 2 depicts how the continuous zoom tool gradually magnifies the target area. Row 3 depicts how n-step tools work. A two-step version would end before entering the Additional Magnification loop, a three-step version would go through the loop once, and so on. The shrinking red dots in row 1 and 3 indicate dwell time.*

tools designed for small-target selection was created (see Figure 8.8). All the tools in this framework increase the effective size of targets (i.e., zoom) to facilitate small-target selection. This framework organizes tools in a discrete-to-continuous continuum. The two-step and continuous zoom tools can be placed, respectively, on the discrete and continuous ends of this continuum. The two-step tool suddenly increases target size to its maximum magnification level, whereas continuous zoom increases target size in what could be considered an infinite number of infinitely small steps.

Consistent with these two extremes, tools closer to the discrete end of the spectrum tend to have less steps of longer duration, whereas tools closer to the continuous end of the spectrum tend to have more steps of shorter duration. The theoretical shorter duration per step of tools with more steps (i.e., more continuous) is the result of shorter dwell times when compared to tools with less steps (i.e., more discrete). Tools toward the continuous end of the spectrum tend to require the user to carry out a more tracking-like task,

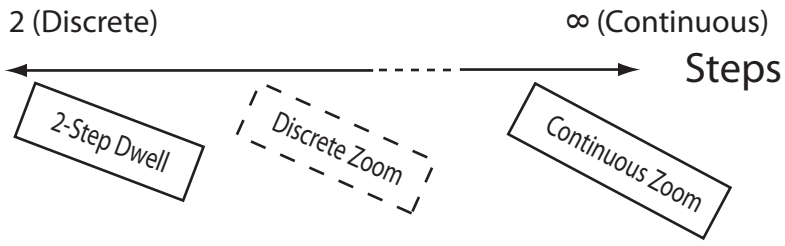


Figure 8.8: *The zoom framework.*

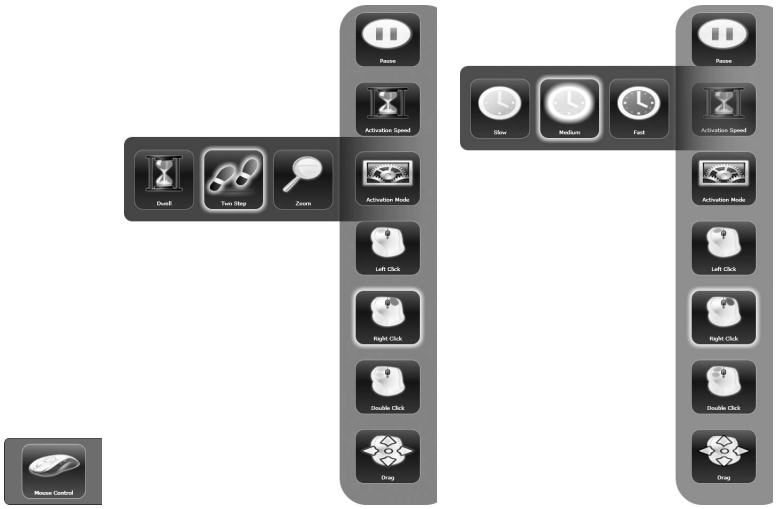
whereas tools toward the discrete end can be better characterized as a series of point-and-select operations. In addition, tools towards the continuous end of the spectrum tend to permit higher magnification levels because objects can increase in size within a window of constant size. Therefore, more continuous tools are less limited by the size of the zooming window.

In general, discrete zoom tools fall in between these two extremes. The specific three-step version tested below falls closer to the discrete end (see Figure 8.8). Even if close to two-step, it is arguable that the three-step tool can facilitate selection of very small targets and naturalness of interaction when compared to two-step magnification. It is also argued that this framework may facilitate comparisons among tools. By studying how tools vary along the continuum, this framework could provide insights into useful tool features and suggest ways in which future designs can combine these features.

8.4.2 Gaze-Controlled Mouse

In order to evaluate the zoom framework in a realistic setting the core functionality from the first experiments and the novel discrete zoom tool was implemented in an gaze-controlled interface together with a velocity-based algorithm, which is described in detail in Chapter 2 p. 41. The GUI of the gaze-controlled mouse was designed with NeoVisus interface components [159]

When the application is started, a small component button becomes visible to the user (see Figure 8.9(a)). The small gaze-aware button represents a collapsed state of the application and, when activated, the component expands to a larger structure unveiling various selection tool, options and functional-



(a) Collapsed. (b) Selection tools menu. (c) Selection time menu.

Figure 8.9: *Figure (a) shows the small collapsed component button, Figure (b) shows the different activation methods available and Figure (c) shows the different pre-specified dwell times.*

ity found in a conventional mouse (e.g., single left click). The gaze sensitive components in the application are activated after a short dwell time (100 ms). The functionality of the application include (from top to bottom): *Pause*, *Activation Time*, *Activation Mode* and standard mouse events. The pause functionality disables selections and enables the user to inspect objects for longer time periods (i.e., avoid the Midas Touch problem). *Activation time* expands a submenu that allows a user to choose between three pre-specified dwell times (i.e., slow, medium and fast)(see Figure 8.9(c)). The *activation mode* component expands a submenu where a user can choose between the different activation methods described in the experiments (see Figure 8.9(b)). The Mouse Events components show the different self-explanatory mouse activation events (e.g., right click, double click). An interesting feature is the ability to perform dragging, which enables the user to move, rearrange, and potentially select multiple targets in the Windows environment. The application is available for download from the *eyeinteract* website[1]. The application is implemented with a velocity-based technique and requires empirical test-

[1] <http://www.eyeinteract.com/>

ing for each eye-tracking system in order to configure the internal parameters of the application.

8.5 Experiment 3: Proof of Concept

8.5.1 Method

In order to study the potential of discrete zoom tools, an experiment was conducted to compare different zoom tools. Participants included 2 male expert users and 8 novices (2 males and 6 females). Novices had no previous experience with gaze interaction. An IG-30 eye tracker from Alea Technologies was used in a desktop setting. Participants were instructed to use a gaze-controlled cursor to point to the target present in the workspace as quickly and accurately as possible. Circular targets appeared one at a time at 1 of 16 possible locations equidistant (300 pixels) from the homing circle on the center. A trial started when a participant positioned the gaze cursor on the homing circle and ended as soon as the participant issued an activation using the corresponding method. A successful target selection was not required. Each participant completed 16 blocks of 16 trials, resulting in a total of 256 activations per participant. All independent variables were manipulated within participants and fixed within blocks.

Zoom tool, *target size*, and *smoothing* was manipulated. Zoom tool had 4 levels: dwell (no zoom), two-step tool, three-step tool, and optimized three-step tool. The magnification level (4x) and dwell time (600 ms) of the two-step tool were chosen based on available versions of this tool. In fact, a relatively high level of magnification was purposefully chosen and a relatively short dwell time. The three-step tool had the same magnification level and dwell time as the two-step tool, whereas the *optimized* three-step tool had twice the magnification (8x) and half the dwell time (300 ms). Achieving 8x magnification with a two-step tool is virtually impossible with a magnified window of the size used in this experiment. The 2 levels of target size were 6- and 12-pixel diameters (to represent some of the smallest targets in the environment). The 2 levels of smoothing (no smoothing and 10-sample average) were applied to the raw eye-tracker data and velocity thresholds were adjusted accordingly. The angular velocity threshold was set to 5 $pixels/s$ when smoothening was enabled and 15 $pixels/s$ with no smooth. *Hit rate*,

completion time, and *subjective ratings* was measured. Data were analyzed with a repeated measures ANOVA and LSD correction in the post-hoc tests.

The three-step tool was expected to: (a) feel more natural, (b) be more resistant to noisy input, and (c) enable reliable selection of smaller targets than the two-step tool. The discrete zoom was not expected to be faster than the two-step tool, but an optimized three-step version was expected to achieve similar speeds to the two-step tool without sacrificing accuracy. This optimized version was expected to be able to accommodate lower dwell times and greater magnification levels than current two-step tools.

8.5.2 Results

All data analyses were conducted on the data from novices. Experts were used for comparison purposes. Target size, smoothing, and subjective-rating results will not be described in detail. Suffice to say that target size affected hit rate but not completion time, whereas smoothing affected completion time but not hit rate. Hit rate was lower for smaller targets than for larger targets, $F(1, 4) = 19.90$, $p < 0.05$. Smoothing over 10 samples resulted in longer completion times than no smoothing, $F(1, 4) = 11.06$, $p < 0.05$. No evidence suggesting that no smoothing had a greater impact on the two-step than on the three-step tool was found. Therefore, this experiment did not support the hypothesis that a three-step tool is more resistant to noise than two-step. The analyses suggest that participants did not rate the three zoom tools different from each other, but some differences were apparent between dwell and all three tools (i.e., dwell was rated as faster but less accurate than zoom tools). There was found no evidence of the three-step tool being perceived as more natural than the two-step tool.

Zoom tool had a significant effect on hit rate, $F(3, 21) = 32.43$, $p < 0.05$. Mean hit rate was lowest without zoom ($M = 0.04$, $SD = 0.03$). The hit rates of the two-step ($M = 0.24$, $SD = 0.11$) and three-step tools ($M = 0.29$, $SD = 0.12$) were not significantly different from each other, $t(7) = 1.22$, $p > 0.05$. The optimized three-step tool ($M = 0.48$, $SD = 0.14$) had a higher hit rate than the three-step tool, $t(7) = 4.57$, $p < 0.05$. These results are consistent with the hypothesis that better accuracy can be achieved with a three-step than with a two-step tool. Given the difference between three-step and optimized three-step, the accuracy advantage is probably due to the lat-

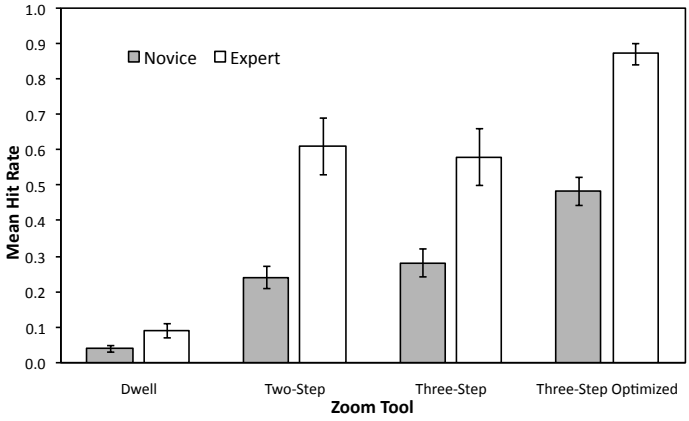


Figure 8.10: *Mean hit rates for the 8 novices and the 2 experts as a function of zoom tool.*

ter's greater magnification level. Mean hit rates across zoom tools show a similar pattern for novices and experts (Figure 8.10).

Zoom tool also had a significant effect on completion time, $F(3, 21) = 119.04$, $p < 0.05$. Completion times were shortest without zoom ($M = 1581$ ms, $SD = 192$ ms). The two-step ($M = 3193$ ms, $SD = 441$ ms) and optimized three-step tools ($M = 3152$ ms, $SD = 375$ ms) were not significantly different from each other, $t(7) = 0.39$, $p > 0.05$. The three-step tool ($M = 3905$ ms, $SD = 442$ ms) took longer than the two-step tool, $t(7) = 5.35$, $p < 0.05$. These results are consistent with the hypothesis that a three-step tool can achieve speeds comparable to a relatively fast version of the two-step tool (given shorter dwell time in the three-step tool). Again, the pattern of results was very similar for novices and experts (see Figure 8.11).

8.5.3 Discussion

There are several reasons why it is desirable to combine gaze interaction with traditional interfaces: one, there is a familiarity with the interface, two, it is costly to develop interfaces for a limited user group, and three, some interaction tasks are sometimes not an option in a traditional dedicated gaze-based interface. Overall, the results of this experiment are promising. There was found support for the possibility that discrete zoom tools can achieve similar

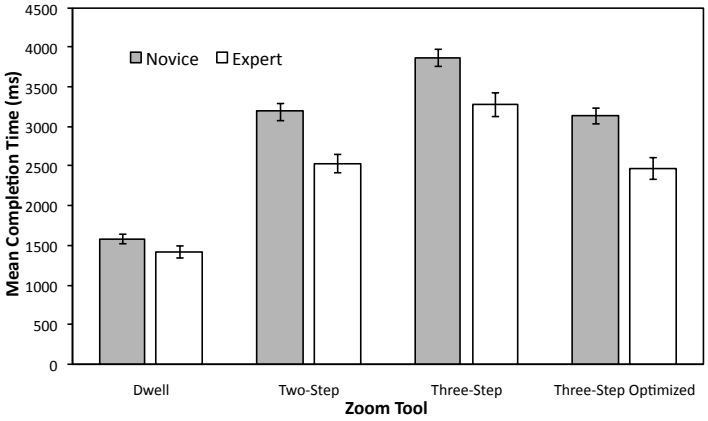


Figure 8.11: *Mean completion times for the 8 novices and the 2 experts as a function of zoom tool.*

speeds and greater accuracy than available two-step tools. Future research should explore whether this finding generalizes to situations in which distractors are present and to tasks in which successful target selection is required.

Future studies should also explore whether a two-step tool could accommodate lower dwell times and whether having different dwell times for different steps could be beneficial. The smoothing manipulation and subjective ratings did not support the hypothesis that three-step tools are more tolerant to noise and natural than two-step tools. Research with a wider range of smoothing levels and subjective ratings could help determine whether this result is due to a lack of difference between tools or to a lack of sensitivity of the measures used. Finally, even if mean values varied substantially, a similar pattern of results across a wide range of expertise levels was found. This result suggests that findings from novices may generalize to more experienced users and novice-user data may be useful to evaluate interface tools.

The primarily focus in this chapter has been on the use of the zoom metaphor as an overall design principle when designing tools for small target selections in a windowed interface. When users need to point to and select small targets using gaze alone, both magnification and zoom methods perform better than dwell alone. Point-and-select performance using magnification tends to be more accurate, but slower, than performance using zoom. Thus, magnifica-

tion is superior for the smallest targets but both methods may be helpful for small targets. Users may have perceived magnification as easier to use than zoom partly due to the high cognitive load resulting from the tracking-like nature of the zoom tool and partly due to the fact they are accustomed to discrete interaction. Future development of smooth-pursuit algorithms for gaze interaction may improve performance of zoom tools.

The experiments have shown that selecting the smallest targets in mainstream GUIs using gaze alone is not easy. Although some tools exist, there is little theoretical guidance for the development of tools to facilitate accessibility to mainstream GUIs for gaze users. Based on the previous work, a conceptual framework was proposed to categorize existing tools and guide the development of new tools. As a proof of concept, a discrete zoom tool was designed and generated hypotheses about how it would compare to other zoom tools based on this framework. An experiment in which the optimized three-step discrete zoom tool achieved better performance than a two-step tool modeled after existing tools. Results suggest that the zoom framework holds potential to guide the development of zoom tools to enhance accessibility to mainstream GUIs for gaze users. This use of discrete strategies is also confirmed in the field of control theory. Here, it is noticed how a problem arises when the eyes are used both for perception and control. They further note that humans are excellent in making strategies but there will always be a lag between visual perception and a corresponding reaction [69]. This observation enhances why an optimal interaction strategy in mono-modal eye-based interaction is optimal for tools based on discrete strategies. The time interval to reposition the gaze simply gives a user the necessary time to understand (i.e., perceive) the conveyed information and time to come up with proper reaction.

8.6 Summary

In this chapter interface tools designed to facilitate point-and-select tasks with small targets when using gaze alone have been presented. The advantages and limitations of each tool have been examined and a total of three experiments have empirically evaluated dwell, zoom, magnification and discrete zoom.

Although zoom is difficult for beginners, users likely become more accurate with training. Well-trained zoom users may take advantage of the speed of this method over magnification. It is also possible that users who are more accustomed to continuous interaction (e.g., gamers) may benefit more from zoom than traditional computer users. The discrete tool is fast and more accurate than the currently available two-step magnification tool and the zoom framework shows potential to guide the design, development, and testing of zoom tools to facilitate the accessibility of mainstream interfaces for gaze users. Based on the proof-of-concept study, the discrete tool is recommended for untrained users performing point-and-select tasks with small targets.

Part V

Conclusions & Future Work

9 Conclusions & Future Work

This chapter outlines the conclusions from the results in the dissertation and reviews the scope and implications for future research.

9.1 General Conclusions

The work in this thesis confirms previous research that has described a lower bound for the accuracy of gaze tracking and gaze interaction. This lower bound was between $0.7^\circ - 1.3^\circ$ in the studies carried out, which is somewhat higher than manufacturers claim, and far too high to the control the majority of elements in standard GUIs. Consequently, gaze-based interaction needs assistive tools to access standard GUI elements, with the implication being that interface designers should consider new approaches that provide larger targets for the user. The research in this thesis has found a way to significantly improve the two-step magnification selection tool currently implemented in many gaze tracking systems. The improvements take advantage of the zoom principle and my research with the StarGazer system shows that people are immediately able to use zoom navigation in a novel, continuous fly-where-you-look approach. The StarGazer system has a set of principle advantages that I believe can be generalized beyond the typing task that was examined. First of all, it is very tolerant to noise and it can be used on small screen sizes. Secondly, it will support navigation in any information space that can be organized in a hierarchal graph structure. Finally, it resembles the control task of steering through a 3D environment, which users are increasingly familiar with due to the growing popularity of computer games

and virtual worlds.

9.2 Future Work

Being able to access each interactive element in mainstream GUIs is a dream for many people. If an open API becomes a reality, it will potentially facilitate a complete redesign of conventional interfaces to suit the needs of the individual user. General research and the research presented in this thesis have shown how gaze-based full-knowledge tools improve the interaction between the user's action and the system's subsequent reaction. If Microsoft, for example, creates an API for their Windows interface the no-knowledge element would be taken out of the equation and the door to true gaze-based interaction would be open. Looking back at Figure 3.6 it becomes clear how non-gaze interfaces and applications, essentially, can be interfaced or redesigned as pure gaze-based applications similar to the model in Figure 9.1.

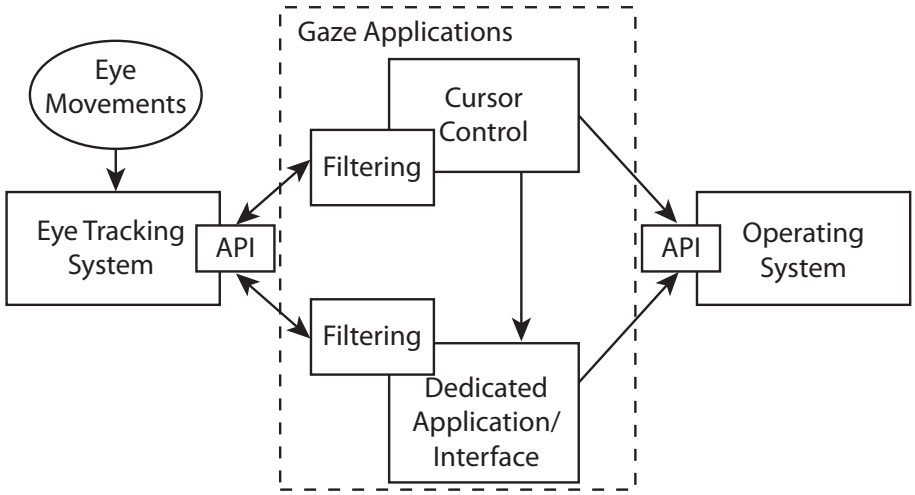


Figure 9.1: *An open API to the interactive elements of an operating system facilitates optimizing or re-designing applications and interfaces for gaze interaction or any other input device.*

One advantage of an open operating-system API would be the ability to

natively apply gaze-based cursor-guidance strategies (e.g., snapping) to all interactive elements in the interface. This would not only benefit users with special needs but also abled body users who are interested in cursor guidance. Once an API is provided, the research challenge is to design tool boxes for rapid development of gaze-based interfaces and investigate strategies for guiding a gaze-controlled cursor in an environment where all interactive elements are known.

9.2.1 Cloud-Based Services

A common trend in interface development indicates that traditional applications are in transition from the old-fashioned local installations to online cloud-based services (e.g., Microsoft Office Web Apps). Some of the advantages of cloud computing include flexible access to data, simple administration, and platform and hardware independence on the client side. Cloud-based services are normally accessed through a browser on the client side. All modern internet browsers conform to the standards from the World Wide Web Consortium (W3C) and an interesting feature with this standard is that all objects in an HTLM web page are accessible to the developer and can thus be modified with different plugins (e.g., Greasemonkey in Mozilla Firefox) to fit the needs of the individual. Advanced access to the content of some cloud-based service is offered through APIs. Some of the most popular for Web 2.0 applications include: Google Data Protocol, GoogleMaps, YouTube, Flickr, Twitter, and Facebook.

EasyTube is an an example of a cloud service designed for users with mental and/or physical challenges, which provides access to YouTube through the Google Data Protocol API[1]. Since 2005 YouTube has become an established part of Internet culture and can now be accessed from several devices such as smartphones and tablets. In spite of this huge success, no attempts have been made towards developing an accessible version of the service for people who are mentally and/or physically challenged. The amount of information on the standard YouTube interface can be overwhelming for some. More than 100 clickable objects are always available on an average YouTube page and browsing through three levels with 20 videos per level will theoretically

[1]Cloud-based version of EasyTube is developed as a master thesis by Chris Topaloudis and is located at: <http://www.easy-tube.net>

branch out to a total of 8000 video options. Consequently, people who are cognitively challenged can easily find themselves lost on irrelevant webpages. Figure 9.2 demonstrates the difference between the traditional YouTube interface and the EasyTube interface.

The EasyTube service aims to involve caregivers with relative little IT competency and have them maintain playlists for groups of users. The foundation of the service is based on a user-driven design approach, as recommended by the inclusive design methodology suggested by Keates et al. [79]. Using this methodology requires a design that offers support for a wide range of input devices, which means that EasyTube supports mouse, touch input, single-switch step-scanning and dwell activation for e.g. gaze- or head- interaction. Several visits with users, with pre-tests, video recordings of user sessions and follow-up meetings with caregivers and users in five different institutions helped to identify the design issues that informed the interaction design of the service.

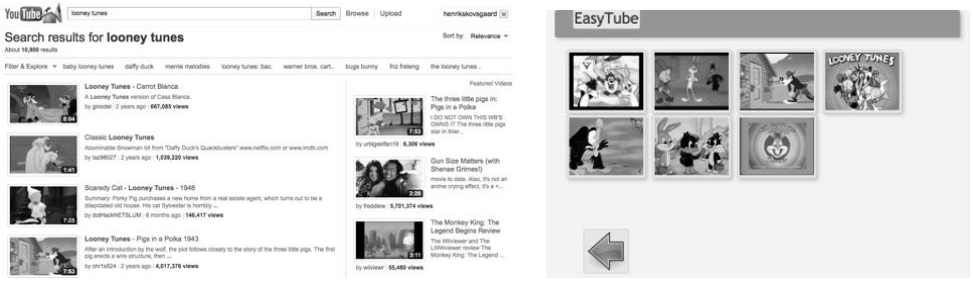


Figure 9.2: *An open API allows developers to redesign or modify interfaces for users with special needs. The example demonstrates a screenshot of a search query in a standard YouTube interface (left) and a screenshot from an EasyTube playlist (right).*

With more and more applications moving to cloud services the research challenge here is to design for inclusion and allow for input from users with special needs in the design process. Services based on HTML allow designers to modify existing services through APIs or tools. EasyTube is a good example of a dedicated interface based on an open API that has been designed to meet the needs of users with special needs and offers a wide range of support for multiple input devices.

9.2.2 Implicit Calibrations

One future direction for eye tracker development is to look at methods that do not require any form of calibration nor the need for external light sources. This will be an important step towards a totally flexible system. Currently, these goals seem out of reach, since they are requirements of the basic model of modern eye tracking systems. One step which has immediate applicability would be to develop calibration routines that are more entertaining and hence more attention-grabbing for children or for users with cognitive difficulties. The procedure can, for example, be hidden inside a small casual game where the user is observing or even controlling objects in strategical regions of the screen as seen in Figure 9.3. The user's behavior during a game sequence can be used to calibrate the system. Such methods allow eye tracking to become a valid input modality for a more variant user base, who may have difficulty following a calibration target if they do not understand why they are required to fix their gaze on an otherwise meaningless dot.

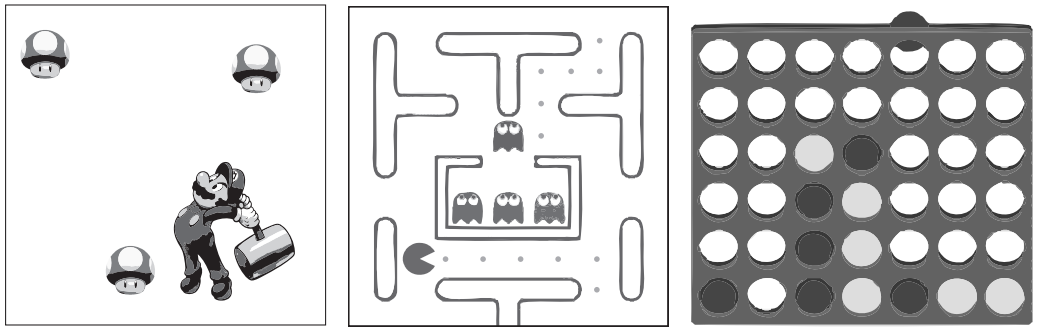


Figure 9.3: *Future methods for calibration routines leave room for the procedure to be more entertaining. The procedure can, for example, be hidden inside a small casual game where the user is observing or even controlling objects in strategical regions of the screen. The examples illustrate Super Mario smashing mushrooms (left), Pacman eating pellets (center) and 'four in a row' (right).*

There are several research challenges to the realization of implicit calibrations. First, we should identify relevant tasks that are both entertaining and intellectually engaging for the users. Games are an obvious solution. Second, we should develop a control system that is able to react to the user's gaze during the calibration sequence. Third, we should design methods for

collecting reliable information about the user's gaze during a sequence, for example, direct the user's attention to empty regions by having something engaging appear or move there.

In interfaces where no information about the the underlying elements exist, it is impossible to re-calibrate without presenting one or more stimuli on top of the display. As mentioned in Section 3.3, the Quickglance system offers partial re-calibration by presenting a calibration stimulus in the center of the screen. An interesting challenge here, is to investigate methods to fluently improve the robustness of systems by prolonging the time between calibrations while still maintaining an acceptable accuracy. That is, invent methods to continuously maintain a calibration without involving the user. As shown by Stampe and Reingold [157], a dedicated interface can be used for more robust tracking by reducing error over time. One way to achieve robust tracking is by working on the interplay between a dedicated interface and interactive eye tracker.

Gazing at any dwell-sensitive button is analog to looking at a stimulus during a calibration routine and an interesting research question is, for example, whether it is fair to assume that a user's gaze falls in the center of a dwell-sensitive button in the moment of selection in a dedicated interface? This question is fairly easy to answer and only requires logging of the PoR during selections. A dedicated interface can, potentially, record the position of the eye during a selection and use the collected data for the ongoing improvement of the internal eye model of the eye tracker for each individual user. The challenge here is to find an appropriate strategy for updating the model and invent strategies to deal with the error between the on-screen cursor and the user's actual PoR.

A requirement that must be met for this to be realized is that eye-tracking manufacturers develop a specialized API that allow external dedicated applications to update the internal model of their systems. This is illustrated in Figure 9.1 with double arrows pointing to the eye tracker's API. Since no such standard exists, this can only be integrated in open-source systems like the ITU Gaze Tracker, where the source code is available for modifications.

9.2.3 Mainstreaming Gaze Interaction

The development of eye tracking devices still remains a niche technology and the high cost of commercial systems is likely due to the low volume of sold devices. This obstacle has been characterized as the *chicken and egg problem* and was already noticed by Bolt as early as in 1985 [16] and by Jacob and Karn [68]. Since the eye tracking industry only sells a few thousand systems a year it is difficult to invest in the engineering research and development that is required for a good, inexpensive unit. Ideally, a mass market approach could lower the cost of a system to a fraction of what it is today. One way of achieving this is by including the technology in popular consumer products such as game consoles, laptops and modern smartphones. In order to lower the cost of the system, the demand from the general user base (e.g., disabled, marketing research, attentive research) should either be dramatically increased or new user groups should be found. To achieve this, developers need to generate interest via dedicated applications, and maybe one or more *killer* apps to boost the market. Recent years have seen increased interest in non-traditional input modalities, particularly in the games market. For example, the Wii game console uses whole body movement as an input. The potential for eye movements measures to increase immersion in game play is clear, and achievable. Such applications of eye tracking would greatly increase the saleability of eye trackers, particularly low cost systems and systems based on existing hardware such as webcams.

In particular, I believe there is a need to move beyond the idea of mono-modal interaction for mono-modal input and bring eye tracking into main stream products such as smartphones and games consoles, which is possible within a few years, since there are no major obstacles in terms of existing technology. The new generation of mobile phones are essentially small laptop computers with multiple processors and build-in camera(s). Miluzzo et al. [107] demonstrated the concept by implementing a simple real-time eye tracking algorithm on a Nokia N810, using the camera's built-in low-resolution camera. Owners of all types of smartphone may in the near future, for example, be able to buy a $50 cradle consisting of a built-in camera and infrared light sources as sketched in Figure 9.4. A proof-of-concept prototype and a software demo for browsing the Internet was demonstrated in mid 2011[2]. This

[2]Internet address: <http://www.senseye.mobi>

Figure 9.4: *Cradle with a built-in camera and light sources designed to fit a modern smartphone.*

affordable solution is easy to use although it may not provide an accuracy and precision that would allow for effective control in outdoor scenarios. This setup demonstrates the potential to work as a small, independent, flexible eye-tracking system or even connect to any nearby peripheral gaze-enabled device. This could, for example, be a computer with a large monitor where objects are larger. Here, the user simply needs to turn on the phone's eye-tracking software and then place the phone below the monitor. Applications for smartphones are usually designed with a minimalistic interface, similar to that of conventional dedicated gaze-based applications. Some applications on a standard smartphone may even benefit from eye control. It could be used to explore semantic information on city maps or to smoothly browse the Internet, to navigate large collections of personal contacts or music collections and to automatically turn pages when reading long texts (e.g., pdf documents).

Eye-tracking technology on a smartphone also has the potential and flexibility to interact with everyday gaze-enabled objects and change their state just by looking at them. Vertegaal [168] notes how the future of gaze aware interfaces is not limited to interactive elements on the computer, but rather to any object around us. Gaze awareness can be imbedded in the objects themselves or in the user with a portable eye tracking system. Advances in

low-cost eye tracking will soon allow for this. The research challenge here is to examine the necessary steps to integrate eye tracking into everyday objects and activities (i.e., ubiquitous gaze contingency). Among other things, this implies the design of relevant hardware, and designing intuitive interfaces and intuitive interaction for everyday objects.

The gaming industry has the potential to create an eye-tracking mass market. In the near future, remakes of well-known casual games could reappear with support for gaze control (e.g., Angry Birds, *Breakout*). For the vast majority of gamers, this novelty will be appealing and entertaining but probably tiresome after some hours of play, if limited to gaze-only control of interfaces. Gamers may have a hard time accepting the limited accuracy provided by the eye tracker. Game developers need to break away from the idea that gaze can be used for pixel-precise pointing (e.g., as a sniper in a 3D shooter) and use gaze as a passive input in combination with traditional controls as an additional dimension in the game. Gaze can be used to communicate the attention from the player to the game engine or other gamers, and thus be used strategically in the gameplay design. The concept of gaze-awareness is not new in gaze-based research but it still has to be proven in mainstream applications such as gaming. Imagine a detective game where the player is able to intimidate witnesses, annoy the bad guy or even impress a beautiful girl, just by the way he looks at the screen. Or, imagine a first person shooter game where the system's agents can be more or less good at hiding based on knowing where the player is looking. Research in attentive computer games involves designing games that add an extra dimension (i.e., visual attention) during game-play this could include pupil size variations [67] for the measurement of emotional valence. One of the challenges here, is to understand how emotions and the variation of pupil size relate to each other [123] and reliably differentiate pupil size changes due to brightness levels from pupil size changes due to the affective state of the user.

Eye tracking will continue to remain an important tool for users with special needs e.g., people with ALS. For able-bodied users, the technology has a high novelty factor and can be integrated into everyday hardware, for example, smartphones. The potential for applied eye tracking is tremendous and the question is what 'killer application' will be responsible for bringing the technology to the mass market. This remains to be seen, but my hope and ambition is to help make this happen in my future work.

10 Epilogue

My interest for gaze-based interaction was awoken by my co-supervisor Dan Witzner Hansen for a self-chosen project on interface zooming for target selection. This work resulted in the following publication: Skovsgaard [147]. While pursuing my master's degree I evaluated gaze-based navigation for a typing task in a early version of what later became the StarGazer typing system. StarGazer demonstrated a novel gaze-based approach for text production. Studies showed that novice users were able to type in the system immediately without any practice. Panning and zooming was found to be able to reside large amounts of imposed noise and even worked with a significant delay on the input signal (although the delay did not feel comfortable for the users). The average typing speeds were relative low and the system was never evaluated with the build-in language model for optimized typing speeds. The work on the StarGazer typing system has been reported in: [42, 5].

After experimenting with the StarGazer the zoom principle I wondered if this also could be a promising interaction approach for point-and-select operations on the desktop. I returned to my early work on zoom selection in a windowed environment. The work resulted in an evaluation of existing tools (i.e., dwell and two-step zoom) and a novel zoom tool based on continuous zoom. The main findings, however slightly disappointing, showed that continuous zoom in its present form was not as accurate as the traditional two-step tool. This work was published in: [149, 151]. After the presentation of Skovsgaard et al. [149], the Germany-based Alea Technologies implemented the continuous zoom technique in their IG-30 system, simply because they liked the "look and feel" of it.

In 2009, I went on a four-month research stay at Wright State University, Ohio in the Department of Psychology hosted by professor John Flach. During my stay I attended his course on control theory and it helped me to realize what was problematic with my old design. The zoom framework was then formulated to explain this difference and a study of an improved version of zoom, called discrete zoom, showed higher accuracy compared to the any other gaze-based tools. The zoom framework and experimental results was published in Skovsgaard et al. [150].

In parallel, I have been involved in writing a book chapter for the upcoming COGAIN book [152] and two different projects throughout my PhD period: The SUS project and the ITU Gaze Tracker. SUS or *Social Udviklings Center* (Social Development Center) has partly founded my PhD through the project called *Implementing Technology (in Institutions)*. This work has resulted in a report on international research on implementation processes and recommendations for system design called Implementing Technology for People with Disabilities [154]. Based on a user-centered design approach an accessible version of YouTube for users with cognitive and motor disabilities, called EasyTube, was developed. This work is to be presented in Skovsgaard et al. [148]. Furthermore, I participated in the initial development of the open-source ITU Gaze Tracker and evaluated the system in various configurations and scenarios. Studies with the ITU GazeTacker whereto I have contributed include: [136, 160, 136, 135, 137, 72, 44, 153].

10.1 Authors Publications & Presentations

10.1.1 Journal Papers

H. Skovsgaard, J. C. Mateo, and J. P. Hansen. Evaluating gaze-based interface tools to facilitate point-and-select tasks with small targets. *Behaviour & Information Technology*, Apr. 2011.

10.1.2 Book Chapters

H. Skovsgaard, K. Räihä, and M. Tall. Computer control by gaze. In P. Majaranta, H. Aoki, M. Donegan, and D. W. Hansen, editors, Gaze Interaction and Applications of Eye Tracking: Advances in Assistive Technologies. IGI Global, July 2011.

10.1.3 Conference Papers

H. Skovsgaard, J. P. Hansen, M. Tall, and K. Nizam. Providing access to YouTube for people with cognitive and motor challenges. In Interactive Technologies and Games: Education, Health and Disability, ITAG 2011, Trent University, Nottingham, Oct. 2011.

H. Skovsgaard, J. San Agustin, S. A. Johansen, J. P. Hansen, and M. Tall. Evaluation of a remote webcam-based eye tracker. In *Proceedings of the 1st Conference on Novel Gaze-Controlled Applications*, NGCA'11, Karlskrona, Sweden, 2011. ACM.

J. P. Hansen, J. San Agustin, and **H. Skovsgaard**. Gaze interaction from bed. In *Proceedings of the 1st Conference on Novel Gaze-Controlled Applications*, NGCA'11, Karlskrona, Sweden, 2011. ACM.

S. A. Johansen, J. San Agustin, **H. Skovsgaard**, J. P. Hansen, and M. Tall. Low cost vs. high-end eye tracking for usability testing. In *Proceedings of the 2011 annual conference extended abstracts on Human factors in computing systems*, CHI EA'11, page 1177-1182, Vancouver, BC, Canada, 2011. ACM.

H. Skovsgaard, J. C. Mateo, J. M. Flach, and J. P. Hansen. Small-target selection with gaze alone. In *Proceedings of the 2010 Symposium on Eye-Tracking Research & Applications*, pages 145-148, Austin, Texas, 2010. ACM Press.

J. San Agustin, **H. Skovsgaard**, E. Møllenbach, M. Barret, M. Tall, D. W. Hansen, and J. P. Hansen. Evaluation of a low-cost open-source gaze tracker. In *Proceedings of the 2010 Symposium on Eye-Tracking Research & Applications*, ETRA'10, page 77-80, Austin, Texas, 2010. ACM.

M. Barrett, **H. Skovsgaard**, and J. San Agustin. Performance evaluation of a Low-Cost gaze tracker for eye typing. In *Proceedings of the 5th Conference on Communication by Gaze Interaction*, pages 13-17, DTU, Denmark, 2009.

J. San Agustin, **H. Skovsgaard**, J. P. Hansen, and D. W. 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.

M. Tall, A. Alapetite, J. San Agustin, **H. Skovsgaard**, J. P. Hansen, D. W. Hansen,

and E. Møllenbach. Gaze-controlled driving. In *Proceedings of the 27th international conference extended abstracts on Human factors in computing systems*, page 4387-4392, New York, NY, USA, 2009. ACM.

J. San Agustin, J. P. Hansen, D. W. Hansen, and **H. Skovsgaard**. Low- cost gaze pointing and EMG clicking. In *Proceedings of the 27th international conference extended abstracts on Human factors in computing systems*, CHI EA'09, page 3247-3252, Boston, MA, USA, 2009. ACM.

H. Skovsgaard, J. Mateo, and J. P. Hansen. How can tiny buttons be hit using gaze only? In *Proceedings of the 4th Conference on Communication by Gaze Interaction*, pages 38-42, Prague, Czech Republic, 2008.

D. W. Hansen, **H. Skovsgaard**, J. P. Hansen, and E. Møllenbach. Noise tolerant selection by gaze-controlled pan and zoom in 3D. In *Proceedings of the 2008 symposium on Eye tracking research & applications*, pages 205-212, Savannah, Georgia, 2008. ACM.

H. Skovsgaard. Estimating acceptable noise-levels on gaze and mouse selection by zooming. In *Student Paper, 2008 Annual IEEE Conference*, pages 1-4, Aalborg University, Feb. 2008. IEEE.

10.1.4 Reports

H. Skovsgaard, M. Tall, and J. P. Hansen. *Implementering af teknologi til mennesker med handicaps. Internationale forskningsresultater vedrørende implementeringsprocesser og anbefalinger til design af systemer.* Technical report, Socialt Udviklingscenter (SUS), Jan. 2009.

10.1.5 Posters & Presentations

H. Skovsgaard. Interfacing YouTube and Facebook with APIs. *Technology Seminar, Hjælpemiddel Instituttet*, Ebletoft, Denmark, 2010, December 1st.

J. C. Mateo, **H. Skovsgaard**, J. M. Flach, and J. P. Hansen. Controlling computers with gaze alone: Tools to facilitate small-target selection. *Poster presented at the NSF IGERT 2010 Project Meeting.* Washington, DC, 2010, May 24th.

H. Skovsgaard, J. P. Hansen, and J. C. Mateo. Interaction with mainstream inter-

faces using gaze alone. *Presented at Scandinavian Workshop on Applied Eye Tracking*, SWAET'10. Lund, Sweden, 2010, May 6th.

J. C. Mateo, **H. Skovsgaard**, J. M. Flach, and J. P. Hansen. Selecting small targets with gaze alone: Is zooming the answer? *Poster presented at the Celebration of Research, Scholarship, and Creative Activities.* Dayton, Ohio, 2010, April 16th.

H. Skovsgaard, and J. C. Mateo. Gaze interaction Selecting small targets with gaze alone. *Presented at the Learning with Disability (LwD) Winter Workshop.* Dayton, Ohio, USA, 2009, December 3rd.

H. Skovsgaard Low-Cost Gaze Pointing and EMG Clicking. *Invited talk for the SIGCHI.dk Interaction design day.* Copenhagen, Denmark. 2009, July 17th.

H. Skovsgaard Low-cost gaze interaction with off-the-shelf components. *Invited talk for the Eye-to-IT conference.* Copenhagen Business School Copenhagen, Denmark, 2009, April 29th.

H. Skovsgaard and J. P. Paulin. Zoom selection by gaze. *Presented at the Scandinavian Workshop on Applied Eye-tracking*, SWAET'08, Lund, Sweden, page 13, 2008

Bibliography

[1] N. Adams, M. Witkowski, and R. Spence. The inspection of very large images by eye-gaze control. In *Proceedings of the working conference on Advanced visual interfaces*, pages 111–118, Napoli, Italy, 2008.

[2] M. Ashmore, A. T. Duchowski, and G. Shoemaker. Efficient eye pointing with a fisheye lens. In *Proceedings of Graphics Interface 2005*, pages 203–210, Victoria, British Columbia, 2005. Canadian Human-Computer Communications Society.

[3] B. Ashtiani and I. S. MacKenzie. BlinkWrite2: an improved text entry method using eye blinks. In *Proceedings of the 2010 Symposium on Eye-Tracking Research & Applications*, ETRA '10, pages 339–345, New York, NY, USA, 2010. ACM.

[4] J. S. Babcock and J. 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.

[5] M. Barrett, H. Skovsgaard, and J. San Agustin. Performance evaluation of a Low-Cost gaze tracker for eye typing. In *Proceedings of the 5th Conference on Communication by Gaze Interaction*, pages 13–17, DTU, Denmark, 2009.

[6] R. Bates and H. 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.

[7] R. Bates and H. Istance. Why are eye mice unpopular? a detailed comparison of head and eye controlled assistive technology pointing devices. *Universal Access in the Information Society*, 2(3):280–290, Oct. 2003.

[8] P. Baudisch, E. Cutrell, K. Hinckley, and A. Eversole. Snap-and-go: helping users align objects without the modality of traditional snapping. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, CHI '05, page 301–310, New York, NY, USA, 2005. ACM.

[9] B. B. Bederson, J. Meyer, and L. Good. Jazz: an extensible zoomable user interface graphics toolkit in java. In *Proceedings of the 13th annual ACM symposium on User interface software and technology*, UIST '00, page 171–180, New York, NY, USA, 2000. ACM.

[10] B. B. Bederson, L. Stead, and J. D. Hollan. Pad++: advances in multiscale interfaces. In *Conference companion on Human factors in computing systems*, CHI '94, page 315–316, New York, NY, USA, 1994. ACM.

[11] N. Bee and E. André. Writing with your eye: A dwell time free writing system adapted to the nature of human eye gaze. In *Perception in Multimodal Dialogue Systems*, volume 5078 of *Lecture Notes in Computer Science*, pages 111–122. Springer Berlin / Heidelberg, 2008.

[12] R. Blanch and M. Ortega. Rake cursor: improving pointing performance with concurrent input channels. In *Proceedings of the 27th international conference on Human factors in computing systems*, pages 1415–1418, Boston, USA, 2009.

[13] H. C. Blickenstorfer. Graffiti: Wow! *Pen Computing Magazine*, pages 30–31, Jan. 1995.

[14] R. A. Bolt. Gaze-orchestrated dynamic windows. In *ACM SIGGRAPH Computer Graphics*, SIGGRAPH '81, page 109–119, New York, NY, USA, 1981. ACM.

[15] R. A. Bolt. *Human Interface: Where People and Computers Meet.* John Wiley & Sons, Inc., 1984.

[16] R. A. Bolt. Conversing with computers. *Technology Review*, 88(2):35–43, 1985.

[17] J. Callahan, D. Hopkins, M. Weiser, and B. Shneiderman. An empirical comparison of pie vs. linear menus. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 95–100. ACM Press, 1988.

[18] R. H. S. Carpenter. *Movements of the eyes.* Pion Ltd., London, 2 edition, 1988.

[19] J. Charlier, C. Buquet, F. Dubus, J. P. Hugeux, and B. Degroc. VISIOBOARD: a new gaze command system for handicapped subjects. In *Medical and Biological Engineering and Computing*, volume 35, pages 461–462, 1997.

[20] P. Clarkson and R. Rosenfeld. Statistical language modeling using the Cmu-Cambridge toolkit. In *5th European Conference on Speech Communication and Technology*, pages 2707–2710, Rhodes, Greece, 1997. ISCA.

[21] C. Dickie, R. Vertegaal, C. Sohn, and D. Cheng. eyeLook: using attention to facilitate mobile media consumption. In *Proceedings of the 18th annual ACM symposium on User interface software and technology*, UIST '05, page 103–106, New York, NY, USA, 2005. ACM.

[22] R. Ditchburn. The function of small saccades. *Vision Research*, 20(3):271–272, 1980.

[23] M. Donegan, J. D. Morris, F. Corno, I. Signorile, A. Chió, V. Pasian, A. Vignola, M. Buchholz, and E. Holmqvist. Understanding users and their needs. *Universal Access in the Information Society*, 8(4):259–275, Mar. 2009.

[24] M. Dorr, L. Pomarjanschi, and E. Barth. Gaze beats mouse: A case study on a gaze-controlled breakout. *PsychNology Journal*, 7(2), 2009.

[25] H. Drewes. *Eye Gaze Tracking for Human Computer Interaction*. PhD thesis, Ludwig-Maximilians-University, Munich, 2010.

[26] H. Drewes and A. Schmidt. Interacting with the computer using gaze gestures. In *Proceedings of the 11th IFIP TC 13*, volume 2, pages 475–488, Rio de Janeiro, Brazil, 2007. Springer.

[27] J. Drewes, A. Montagnini, and G. S. Masson. Effects of pupil size on recorded gaze position: a live comparison of two eyetracking systems. VSS, 2011.

[28] A. T. Duchowski. *Eye tracking methodology*. Springer, 1 edition, 2003.

[29] A. T. Duchowski, E. Medlin, N. Cournia, A. Gramopadhye, B. Melloy, and S. 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.

[30] I. Ekman, A. Poikola, M. Mäkäräinen, T. Takala, and P. Hämäläinen. Voluntary pupil size change as control in eyes only interaction. In *Proceedings of the 2008 symposium on Eye Tracking Research & Applications*, ETRA '08, page 115–118, New York, NY, USA, 2008. ACM.

[31] S. R. Ellis and Stark. Scanpaths revisited: Cognitive models direct active looking. In *Eye Movements: Cognition and Visual Perception*, page 193–226. Lawrence Erlbaum Associates, 1981.

[32] G. Evreinov and R. Raisamo. Optimizing menu selection process for Single-Switch manipulation. In *Computers Helping People with Special Needs*, volume 3118 of *Lecture Notes in Computer Science*, pages 628–628–628. Springer Berlin / Heidelberg, 2004.

[33] W. Ferrell. Remote manipulation with transmission delay. *IEEE Transactions on Human Factors in Electronics*, 6:24–32, 1965.

[34] P. M. Fitts. The information capacity of the human motor system in controlling the amplitude of movement. *Journal of Experimental Psychology*, 47(6):381–391, 1954.

[35] D. Fono and R. Vertegaal. EyeWindows. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 151–160, Portland, Oregon, USA, 2005.

[36] A. F. Fuchs. The saccadic system. In *The Control of Eye Movement*, pages 343–362. Academic Press, New York, 1971.

[37] C. Goossens' and S. Crain. Overview of nonelectronic eye-gaze communication techniques. *Augmentative and Alternative Communication*, 3:77–89, Jan. 1987.

[38] K. Grauman, M. Betke, J. Lombardi, J. Gips, and G. R. Bradski. Communication via eye blinks and eyebrow raises: video-based human-computer interfaces. *Universal Access in the Information Society*, 2:359–373, 2003.

[39] T. Grossman and R. Balakrishnan. The bubble cursor: enhancing target acquisition by dynamic resizing of the cursor's activation area. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, CHI '05, page 281–290, New York, NY, USA, 2005. ACM.

[40] C. Gutwin. Improving focus targeting in interactive fisheye views. In *Proceedings of the SIGCHI conference on Human factors in computing systems: Changing our world, changing ourselves*, CHI '02, page 267–274, New York, NY, USA, 2002. ACM.

[41] D. Hansen and Q. Ji. In the eye of the beholder: A survey of models for eyes and gaze. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 32(3):478–500, 2010.

[42] D. W. Hansen, H. Skovsgaard, J. P. Hansen, and E. Møllenbach. Noise tolerant selection by gaze-controlled pan and zoom in 3D. In *Proceedings of the 2008 symposium on Eye tracking research & applications*, pages 205–212, Savannah, Georgia, 2008. ACM.

[43] J. P. Hansen, D. Hansen, and A. Johansen. Bringing gaze-based interaction back to basics. In *Universal Access in HCI (UAHCI): Towards an Information Society for All*, volume 3, pages 325–329, New Orleans, USA, 2001. Lawrence Erlbaum.

[44] J. P. Hansen, J. San Agustin, and H. Skovsgaard. Gaze interaction from bed. In *Proceedings of the 1st Conference on Novel Gaze-Controlled Applications*, NGCA '11, Karlskrona, Sweden, 2011. ACM.

[45] J. P. Hansen, K. Tørning, A. S. Johansen, K. Itoh, and H. Aoki. Gaze typing compared with input by head and hand. In *Proceedings of the 2004 symposium on Eye tracking research & applications*, ETRA '04, page 131–138, New York, NY, USA, 2004. ACM.

[46] H. Heikkilä and K. Räihä. Speed and accuracy of gaze gestures. *Journal of Eye Movement Research*, 3(2):1–14, 2009.

[47] D. J. Higginbotham, H. Shane, S. Russell, and K. Caves. Access to AAC: present, past, and future. *Augmentative and Alternative Communication*, 23(3):243–257, Jan. 2007.

[48] P. G. Hoel, S. C. Port, and C. J. Stone. *Introduction to Statistical Theory.* Houghton Mifflin (Boston), 1st edition, 1971.

[49] K. Holmqvist, M. Nyström, R. Andersson, R. Dewhurst, H. Jarodzka, and J. v. d. Weijer. *Eye Tracking: A Comprehensive Guide to Methods and Measures.* Oxford University Press, July 2011.

[50] K. Holmqvist, M. Nyström, and F. Mulvey. Towards a standard for measuring and reporting data quality in eye movement research (in review). *Journal of Eye Movement Research*, 2011.

[51] K. Hornbæk, B. B. Bederson, and C. Plaisant. Navigation patterns and usability of zoomable user interfaces with and without an overview. *ACM Trans. Comput.-Hum. Interact.*, 9(4):362–389, Dec. 2002.

[52] A. J. Hornof and A. Cavender. EyeDraw. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 161–170, Portland, Oregon, USA, 2005.

[53] A. J. Hornof and T. Halverson. Cleaning up systematic error in eye-tracking data by using required fixation locations. *Behavior Research Methods, Instruments, & Computers*, 34(4):592–604, Nov. 2002.

[54] A. Huckauf and M. H. Urbina. Gazing with pEYEs: towards a universal input for various applications. In *Proceedings of the 2008 symposium on Eye tracking research & applications*, ETRA '08, page 51–54, New York, NY, USA, 2008. ACM.

[55] A. Huckauf and M. H. Urbina. On object selection in gaze controlled environments. *Journal of Eye Movement Research*, 2(4):1–7, 2008.

[56] T. Hutchinson, K. White, W. Martin, K. Reichert, and L. Frey. Human-computer interaction using eye-gaze input. *Systems, Man and Cybernetics, IEEE Transactions on*, 19(6):1527–1534, 1989.

[57] ISO. Ergonomic requirements for office work with visual display terminals (VDTs) - part 9. In *Requirements for nonkeyboard input devices.* International Organization for Standardization, 2000.

[58] H. Istance, R. Bates, A. Hyrskykari, and S. Vickers. Snap clutch, a moded approach to solving the midas touch problem. In *Proceedings of*

the 2008 symposium on Eye-Tracking Research & Applications, pages 221–228, Savannah, Georgia, 2008. ACM Press.

[59] H. Istance, A. Hyrskykari, L. Immonen, S. Mansikkamaa, and S. Vickers. Designing gaze gestures for gaming. In *Proceedings of the 2010 Symposium on Eye-Tracking Research & Applications*, pages 323–330, Austin, Texas, 2010. ACM Press.

[60] H. Istance, A. Hyrskykari, S. Vickers, and N. Ali. User performance of gaze-based interaction with on-line virtual communities. In *Proceedings of the 4th Conference on Communication by Gaze Interaction*, pages 28–32, Prague, Czech Republic, 2008.

[61] H. Istance, A. Hyrskykari, S. Vickers, and T. Chaves. For your eyes only: Controlling 3D online games by Eye-Gaze. In T. Gross, J. Gulliksen, P. Kotzé, L. Oestreicher, P. Palanque, R. Prates, and M. Winckler, editors, *Human-Computer Interaction*, volume 5726 of *INTERACT 2009*, pages 314–327. Springer, 2009.

[62] H. Istance, C. Spinner, and P. A. Howarth. Enabling motor impaired users to use standard GUI software. *Proceedings of The 1st European Conference on Disability, Virtual Reality and Associated Technologies (ECDVRAT 96)*, pages 109–116, 1996.

[63] K. Itoh, H. Aoki, and J. P. Hansen. A comparative usability study of two japanese gaze typing systems. *Proceedings of the 2006 symposium on Eye tracking research & applications*, page 59–66, 2006.

[64] R. J. K. Jacob. The use of eye movements in human-computer interaction techniques: what you look at is what you get. *ACM Transactions on Information Systems*, 9(2):152–169, 1991.

[65] R. J. K. Jacob. Hot topics-eye-gaze computer interfaces: what you look at is what you get. *Computer*, 26(7):65–66, 1993.

[66] R. J. K. Jacob. Eye tracking in advanced interface design. In *Virtual environments and advanced interface design*, pages 258–288. Oxford University Press, Inc, 1995.

[67] R. J. K. Jacob. The future of input devices. *ACM Computing Surveys*, 28:138–es, Dec. 1996.

[68] R. J. K. Jacob and K. 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, J. Hyona, Radach, R., and Deubel, H., (Eds.)*, pages 573–605. Elsevier Science, Amsterdam, 2003.

[69] R. J. Jagacinski and J. M. Flach. *Control Theory for Humans: Quantitative Approaches To Modeling Performance*. CRC Press, 1 edition, Sept. 2002.

[70] B. B. Jeppesen. *Er der mon bedre i himmelen?* Dafolo, 1999.

[71] A. S. Johansen, J. P. Hansen, D. W. Hansen, K. Itoh, and S. Mashino. Language technology in a predictive, restricted on-screen keyboard with dynamic layout for severely disabled people. In *Proceedings of the 2003 EACL Workshop on Language Modeling for Text Entry Methods*, pages 59–66, Budapest, Hungary, 2003. Association for Computational Linguistics.

[72] S. A. Johansen, J. San Agustin, H. Skovsgaard, J. P. Hansen, and M. Tall. Low cost vs. high-end eye tracking for usability testing. In *Proceedings of the 2011 annual conference extended abstracts on Human factors in computing systems*, CHI EA '11, page 1177–1182, Vancouver, BC, Canada, 2011. ACM.

[73] J. S. Johnson, L. Liu, G. Thomas, and J. P. Spencer. Calibration algorithm for eyetracking with unrestricted head movement. *Behavior Research Methods*, 39(1):123–132, Feb. 2007.

[74] M. Joos, S. Malischke, S. Pannasch, A. Storch, and B. Velichkovsky. Comparing two Gaze-Interaction interfaces: A usability study with locked-in patients. In *Proceedings of the 3rd Conference on Communication by Gaze Interaction*, pages 82–88, Leicester, UK, 2007.

[75] I. Jordansen, S. Boedeker, M. Donegan, L. Oosthuizen, M. di Girolamo, and J. P. Hansen. D7.2. report on a market study and demographics of user population. Communication by gaze interaction, IST-2003-511598: deliverable 7.2., COGAIN, 2005.

[76] Y. Kammerer, K. Scheiter, and W. Beinhauer. Looking my way through the menu. In *Proceedings of the 2008 symposium on Eye-Tracking Research & Applications*, pages 213–220, Savannah, Georgia, 2008. ACM Press.

[77] R. Karsh and F. W. Breitenbach. Looking at looking: The amorphous fixation measure. In *Eye Movements and Phychological Functions: International Views*, pages 53–64. Erlbaum, Hillsdale, NJ, 1983.

[78] M. Kaur, M. Tremaine, N. Huang, J. Wilder, Z. Gacovski, F. Flippo, and C. S. Mantravadi. Where is "it"? event synchronization in gaze-speech input systems. In *Proceedings of the 5th international conference on Multimodal interfaces*, pages 151–158, Vancouver, British Columbia, Canada, 2003.

[79] S. Keates, P. J. Clarkson, L. Harrison, and P. Robinson. Towards a practical inclusive design approach. In *Proceedings on the 2000 conference on Universal Usability*, CUU '00, page 45–52, Arlington, Virginia, United States, 2000. ACM. ACM ID: 355471.

[80] M. Kobayashi and T. Igarashi. Ninja cursors: using multiple cursors to assist target acquisition on large screens. In *Proceeding of the twenty-sixth annual CHI conference on Human factors in computing systems*, pages 949–958, Florence, Italy, 2008. ACM Press.

[81] O. Komogortsev and J. Khan. Kalman filtering in the design of Eye-Gaze-Guided computer interfaces. In *Human-Computer Interaction. HCI Intelligent Multimodal Interaction Environments*, pages 679–689. SpringerLink, 2007.

[82] M. Kumar. *Gaze-enhanced user interface design.* Doctoral dissertation, Stanford University, USA, 2007.

[83] M. Kumar, J. Klingner, R. Puranik, T. Winograd, and A. Paepcke. Improving the accuracy of gaze input for interaction. In *Proceedings of the 2008 symposium on Eye-Tracking Research & Applications*, pages 65–68, Savannah, Georgia, 2008. ACM Press.

[84] M. Kumar, A. Paepcke, and T. Winograd. EyePoint: practical pointing and selection using gaze and keyboard. In *Proceedings of the SIGCHI*

conference on Human factors in computing systems, pages 421–430, San Jose, USA, 2007. ACM Press.

[85] G. Kurtenbach, A. Sellen, and W. Buxton. An empirical evaluation of some articulatory and cognitive aspects of marking menus. *Human-Computer Interaction*, 8(1):1–23, Mar. 1993.

[86] M. Land and B. Tatler. *Looking and Acting: Vision and eye movements in natural behaviour*. Oxford University Press, USA, 1 edition, Oct. 2009.

[87] C. Lankford. Effective eye-gaze input into windows. In *Proceedings of the 2000 symposium on Eye-Tracking Research & Applications*, pages 23–27, Palm Beach, USA, 2000. ACM Press.

[88] A. L. Larsen. *Hellere dø af grin end af ALS – 99 sandfærdige historier om at leve med Amyotrofisk Lateral Sclerose*. Skriveforlaget, 1 edition, 2009.

[89] J. Laukkanen, P. Isokoski, and K. Räihä. The cone and the lazy bubble: two efficient alternatives between the point cursor and the bubble cursor. In *Proceeding of the twenty-sixth annual SIGCHI conference on Human factors in computing systems*, pages 309–312, Florence, Italy, 2008. ACM.

[90] G. J. Lepinski and R. Vertegaal. Using face position for low cost input, long range and oculomotor impaired users. In *Proceedings of the 3rd Conference on Communication by Gaze Interaction*, volume 71-73, Leicester, UK, 2007.

[91] D. Li, J. Babcock, and D. J. Parkhurst. openEyes. In *Proceedings of Eye tracking research & applications*, pages 95–100, San Diego, California, 2006. ACM.

[92] Y. li Tian, T. Kanade, and J. Cohn. Dual-state parametric eye tracking. In *Automatic Face and Gesture Recognition, 2000. Proceedings. Fourth IEEE International Conference on*, pages 110–115, 2000.

[93] I. S. MacKenzie. A note on the information-theoretic basis of fitts' law. *Journal of Motor Behavior*, 21(3):323–330, Sept. 1989.

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

[95] I. S. MacKenzie and R. W. Soukoreff. A character-level error analysis technique for evaluating text entry methods. In *Proceedings of the second Nordic conference on Human-computer interaction*, NordiCHI '02, page 243–246, New York, USA, 2002. ACM.

[96] I. S. MacKenzie and K. Tanaka-Ishii. *Text Entry Systems: Mobility, Accessibility, Universality*. Morgan Kaufmann, Mar. 2007.

[97] I. S. MacKenzie and X. Zhang. Eye typing using word and letter prediction and a fixation algorithm. In *Proceedings of the 2008 symposium on Eye-Tracking Research & Applications*, pages 55–58, Savannah, USA, 2008. ACM Press.

[98] P. Majaranta. *Text Entry by Eye Gaze*. 11. Tampere University Press, Tampere University, 2009.

[99] P. Majaranta, U. Ahola, and O. 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, 2009. ACM Press.

[100] P. Majaranta, I. S. MacKenzie, A. Aula, and K. Räihä. Effects of feedback and dwell time on eye typing speed and accuracy. *Universal Access in the Information Society*, 5(2):199–208, 2006.

[101] P. Majaranta and K. 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.

[102] P. Majaranta and K. Räihä. Text entry by gaze: Utilizing eyetracking. In I. S. MacKenzie and K. Tanaka-Ishii, editors, *Text entry systems: mobility, accessibility, universality*, pages 175–187. Morgan Kaufmann, Mar. 2007.

[103] S. Martinez-Conde, S. L. Macknik, and D. H. Hubel. The role of fixational eye movements in visual perception. *Nat Rev Neurosci*, 5(3):229–240, Mar. 2004.

[104] J. C. Mateo, J. San Agustin, and J. P. Hansen. Gaze beats mouse: hands-free selection by combining gaze and emg. In *Proceeding of the twenty-sixth annual CHI conference extended abstracts on Human factors in computing systems*, pages 3039–3040, Florence, Italy, 2008. ACM Press.

[105] V. McArthur, S. J. Castellucci, and I. S. MacKenzie. An empirical comparison of "wiimote" gun attachments for pointing tasks. In *Proceedings of the 1st ACM SIGCHI symposium on Engineering interactive computing systems*, EICS '09, page 203–208, Pittsburgh, PA, USA, 2009. ACM.

[106] G. W. McConkie. Evaluating and reporting data quality in eye movement research. *Behavior Research Methods & Instrumentation*, 13(2):97–106, Jan. 1981.

[107] E. Miluzzo, T. Wang, and A. T. Campbell. EyePhone. In *Proceedings of the second ACM SIGCOMM workshop on Networking, systems, and applications on mobile handhelds*, page 15. ACM Press, 2010.

[108] D. Miniotas. Application of fitts' law to eye gaze interaction. In *CHI '00 extended abstracts on Human factors in computing systems*, CHI EA '00, page 339–340, The Hague, The Netherlands, 2000. ACM.

[109] D. Miniotas, O. Spakov, and G. Evreinov. Symbol creator: An alternative eye-based text entry technique with low demand for screen space. In *Human-computer interaction: INTERACT '03 : IFIP TC.13*, pages 137–143, 2003.

[110] D. Miniotas, O. Špakov, and I. S. MacKenzie. Eye gaze interaction with expanding targets. *CHI '04 extended abstracts on Human factors in computing systems*, page 1255–1258, 2004.

[111] D. Miniotas, O. Špakov, I. Tugoy, and I. S. MacKenzie. Speech-augmented eye gaze interaction with small closely spaced targets. In *Proceedings of the 2006 symposium on Eye-Tracking Research & Applications*, pages 67–72, San Diego, CA, 2006. ACM Press.

[112] L. C. Moats. *Speech to Print: Language Essentials for Teachers*. Brookes Publishing Company, 1 edition, July 2000.

[113] E. Møllenbach, M. Lillholm, A. Gail, and J. P. Hansen. Single gaze gestures. In *Proceedings of the 2010 Symposium on Eye-Tracking Research & Applications*, pages 177–180, Austin, Texas, 2010. ACM Press.

[114] E. Møllenbach, T. Stefansson, and J. P. Hansen. All eyes on the monitor: gaze based interaction in zoomable, multi-scaled information-spaces. *Proceedings of the 13th international conference on Intelligent user interfaces*, page 373–376, 2008.

[115] C. 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.

[116] C. H. Morimoto and A. Amir. Context switching for fast key selection in text entry applications. In *Proceedings of the 2010 Symposium on Eye-Tracking Research & Applications*, ETRA '10, page 271–274, New York, USA, 2010. ACM.

[117] C. H. Morimoto and M. R. Mimica. Eye gaze tracking techniques for interactive applications. *Computer Vision and Image Understanding*, 98(1):4–24, Apr. 2005.

[118] P. Murphy, A. Duncan, A. Glennie, and P. Knox. The effect of scleral search coil lens wear on the eye. *The British Journal of Ophthalmology*, 85(3):332–335, Mar. 2001.

[119] T. Ohno. Features of eye gaze interface for selection tasks. In *3rd Asia Pacific Computer Human Interaction*, pages 176–181, Washington, DC, 1998. IEEE Computer Society.

[120] T. Ohno and N. Mukawa. Gaze-Based interaction for anyone, anytime. In *Proceedings of HCI International 2003*, volume 4, pages 1452–1456, Mahwah, NJ, 2003. Lawrence Erlbaum.

[121] T. J. Palmeri. Automaticity. In L. Nadel, editor, *Encyclopedia of Cognitive Science*. John Wiley & Sons, Ltd, Chichester, 2006.

[122] T. Partala, A. Aula, and V. Surakka. Combined voluntary gaze direction and facial muscle activity as a new pointing technique. In *Human-computer interaction: INTERACT '01 : IFIP TC.13*. IOS Press, 2001.

[123] T. Partala and V. Surakka. Pupil size variation as an indication of affective processing. *International Journal of Human-Computer Studies*, 59(1-2):185–198, July 2003.

[124] S. Pook, E. Lecolinet, G. Vaysseix, and E. Barillot. Context and interaction in zoomable user interfaces. In *Proceedings of the working conference on Advanced visual interfaces*, AVI '00, page 227–231, New York, NY, USA, 2000. ACM.

[125] M. Porta, A. Ravarelli, and G. Spagnoli. ceCursor, a contextual eye cursor for general pointing in windows environments. In *Proceedings of the 2010 Symposium on Eye-Tracking Research & Applications*, pages 331–337, Austin, Texas, 2010.

[126] M. Porta and M. Turina. Eye-S: a full-screen input modality for pure eye-based communication. In *Proceedings of the 2008 symposium on Eye Tracking Research & Applications*, ETRA '08, page 27–34, New York, NY, USA, 2008. ACM.

[127] D. Purves, G. J. Augustine, D. Fitzpatrick, W. C. Hall, A. Lamantia, J. O. McNamara, and S. M. Williams. *Neuroscience*. Sinauer Associates, 3rd edition, 2004.

[128] K. Räihä and O. Špakov. Disambiguating ninja cursors with eye gaze. In *Proceedings of the 27th international conference on Human factors in computing systems*, pages 1411–1414, Boston, USA, 2009. ACM Press.

[129] L. Riggs and F. Ratliff. The effects of counteracting the normal movements of the eye. *Journal of the Optical Society of America*, 42:872–873, 1952.

[130] D. A. Robinson. The mechanics of human smooth pursuit eye movement. *The Journal of Physiology*, 180(3):569–591, Oct. 1965.

[131] N. N. Rommelse, S. Van der Stigchel, and J. A. Sergeant. A review on eye movement studies in childhood and adolescent psychiatry. *Brain and Cognition*, 68(3):391–414, Dec. 2008.

[132] D. Rozado. *Analysis and Extension of Hierarchical Temporal Memory for Multivariable Time Series*. PhD thesis, Universidad Autónoma de Madrid, 2011.

[133] D. D. Salvucci and J. 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.

[134] J. San Agustin. *Off-the-Shelf Gaze Interaction*. PhD thesis, IT University of Copenhagen, 2009.

[135] J. San Agustin, J. P. Hansen, D. W. Hansen, and H. Skovsgaard. Low-cost gaze pointing and EMG clicking. In *Proceedings of the 27th international conference extended abstracts on Human factors in computing systems*, page 3247–3252, Boston, MA, USA, 2009. ACM.

[136] J. San Agustin, H. Skovsgaard, J. P. Hansen, and D. W. 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.

[137] J. San Agustin, H. Skovsgaard, E. Møllenbach, M. Barret, M. Tall, D. W. Hansen, and J. P. Hansen. Evaluation of a low-cost open-source gaze tracker. In *Proceedings of the 2010 Symposium on Eye-Tracking Research & Applications*, ETRA '10, page 77–80, Austin, Texas, 2010. ACM.

[138] D. Sauter, B. Martin, N. Di Renzo, and C. Vomscheid. Analysis of eye tracking movements using innovations generated by a kalman filter. *Medical and Biological Engineering and Computing*, 29(1):63–69, Jan. 1991.

[139] N. Schneider, P. Bex, E. Barth, and M. Dorr. An open-source low-cost eye-tracking system for portable real-time and offline tracking. In *Proceedings of the 1st Conference on Novel Gaze-Controlled Applications*, NGCA '11, page 4, Karlskrona, Sweden, 2011. ACM.

[140] T. Sen and T. Megaw. The effects of task variables and prolonged performance on saccadic eye movement parameters. In *Theoretical and applied aspects of eye movement research*, pages 101–111. Elsevier, 1984.

[141] W. Sewell and O. Komogortsev. Real-time eye gaze tracking with an unmodified commodity webcam employing a neural network. In *Proceedings of the 28th of the international conference extended abstracts on Human factors in computing systems*, page 3739–3744, New York, USA, 2010. ACM.

[142] F. Shein, G. Hamann, N. Brownlow, J. Treviranus, M. Milner, and P. Parnes. WiViK: a visual keyboard for windows 3.0. In *Proceedings of the 14th Annual Conference of the Rehabilitation Engineering Society of North America*, pages 160–162, USA, 1991.

[143] G. F. Shein. *Towards task transparency in alternative computer access, selection of text through switch-based scanning.* Electronic thesis or dissertation, University of Toronto, Dept. of Industrial Engineering, 1997. grantor: University of Toronto.

[144] B. Shneiderman. Direct manipulation: A step beyond programming languages. *Computer*, 16(8):57–69, 1983.

[145] B. Shneiderman and C. Plaisant. *Designing the User Interface: Strategies for Effective Human-Computer Interaction.* Addison Wesley, 4 edition, Apr. 2004.

[146] L. E. Sibert and R. J. K. Jacob. Evaluation of eye gaze interaction. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 281–288, Hague, The Netherlands, 2000. ACM Press.

[147] H. Skovsgaard. Estimating acceptable noise-levels on gaze and mouse selection by zooming. In *Student Paper, 2008 Annual IEEE Conference*, pages 1–4, Aalborg University, Feb. 2008. IEEE.

[148] H. Skovsgaard, J. P. Hansen, M. Tall, and K. Nizam. Providing access to YouTube for people with cognitive and motor challenges. In *Interactive Technologies and Games: Education, Health and Disability*, ITAG 2011, Trent University, Nottingham, Oct. 2011.

[149] H. Skovsgaard, J. Mateo, and J. P. Hansen. How can tiny buttons be hit using gaze only? In *Proceedings of the 4th Conference on Communication by Gaze Interaction*, pages 38–42, Prague, Czech Republic, 2008.

[150] H. Skovsgaard, J. C. Mateo, J. M. Flach, and J. P. Hansen. Small-target selection with gaze alone. In *Proceedings of the 2010 Symposium on Eye-Tracking Research & Applications*, pages 145–148, Austin, Texas, 2010. ACM Press.

[151] H. Skovsgaard, J. C. Mateo, and J. P. Hansen. Evaluating gaze-based interface tools to facilitate point-and-select tasks with small targets. *Behaviour & Information Technology*, Apr. 2011.

[152] H. Skovsgaard, K. Räihä, and M. Tall. Computer control by gaze. In P. Majaranta, H. Aoki, M. Donegan, and D. W. Hansen, editors, *Gaze Interaction and Applications of Eye Tracking: Advances in Assistive Technologies*. IGI Global, July 2011.

[153] H. Skovsgaard, J. San Agustin, S. A. Johansen, J. P. Hansen, and M. Tall. Evaluation of a remote webcam-based eye tracker. In *Proceedings of the 1st Conference on Novel Gaze-Controlled Applications*, NGCA '11, Karlskrona, Sweden, 2011. ACM.

[154] H. Skovsgaard, M. Tall, and J. P. Hansen. Implementering af teknologi til mennesker med handicaps. internationale forskningsresultater vedrørende implementeringsprocesser og anbefalinger til design af systemer. Technical report, Socialt Udviklingscenter (SUS), Jan. 2009.

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

[156] D. M. Stampe. Heuristic filtering and reliable calibration methods for video-based pupil-tracking systems. *Behavior Research Methods Instruments Computers*, 25(2):137–142, 1993.

[157] D. M. Stampe, E. M. Reingold, R. Groner, and G. d'Ydewalle. Selection by looking: A novel computer interface and its application to psychological research. In *Eye Movement Research - Mechanisms, Processes, and Applications*, volume Volume 6, pages 467–478. North-Holland, 1995.

[158] V. Surakka, M. Illi, and P. Isokoski. Gazing and frowning as a new human–computer interaction technique. *ACM Trans. Appl. Percept.*, 1(1):40–56, 2004.

[159] M. Tall. NeoVisus: gaze driven interface components. In *COGAIN 2008: Communication, Environment and Mobility Control by Gaze*, pages 48–52, Prague, Czech Republic, 2008.

[160] M. Tall, A. Alapetite, J. San Agustin, H. Skovsgaard, J. P. Hansen, D. W. Hansen, and E. Møllenbach. Gaze-controlled driving. In *Proceedings of the 27th international conference extended abstracts on Human factors in computing systems*, page 4387–4392, New York, NY, USA, 2009. ACM.

[161] Tobii. User manual: MyTobii, version 2.3. *Danderyd, Sweden: Tobii Technology*, 2009.

[162] O. Tuisku, P. Majaranta, P. Isokoski, and K. 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*, ETRA '08, page 19–26, New York, NY, USA, 2008. ACM. ACM ID: 1344476.

[163] M. H. Urbina and A. Huckauf. Fast hands-free writing by gaze direction. In *Proceedings of the 3rd Conference on Communication by Gaze Interaction*, pages 65–70, Leicester, UK, 2007.

[164] M. H. Urbina, M. Lorenz, and A. Huckauf. Pies with EYEs: the limits of hierarchical pie menus in gaze control. In *Proceedings of the 2010 Symposium on Eye-Tracking Research & Applications*, pages 93–96, Austin, Texas, 2010. ACM Press.

[165] J. N. van der Geest and M. A. Frens. Recording eye movements with video-oculography and scleral search coils: a direct comparison of two methods. *Journal of Neuroscience Methods*, 114(2):185–195, Mar. 2002.

[166] J. van Wijk and W. Nuij. Smooth and efficient zooming and panning. In *IEEE Symposium on Information Visualization, 2003*, pages 15–23, 2003.

[167] F. J. Verheijen. A simple after image method demonstrating the involuntary multidirectional eye movements during fixation. *Optica Acta: International Journal of Optics*, 8(4):309–312, Oct. 1961.

[168] R. Vertegaal. Attentive user interfaces. *Communications of the ACM*, 46(3):30–33, 2003.

[169] S. Vickers. *Eye-gaze Interaction Techniques for Use in Online Games and Environments for Users with Severe Physical Disabilities*. PhD thesis, De Montfort University, Leicester, UK, 2011.

[170] A. Villanueva, R. Cabeza, and S. Porta. Eye tracking system model with easy calibration. In *Proceedings of the 2004 symposium on Eye tracking research & applications*, pages 55–55, San Antonio, Texas, 2004. ACM.

[171] O. Špakov and D. Miniotas. On-line adjustment of dwell time for target selection by gaze. In *Proceedings of the third Nordic conference on Human-computer interaction*, pages 203–206, Tampere, Finland, 2004. ACM Press.

[172] O. Špakov and D. Miniotas. Gaze-based selection of standard-size menu items. In *Proceedings of the 7th international conference on Multimodal interfaces*, pages 124–128, Torento, Italy, 2005. ACM Press.

[173] D. J. Ward, A. F. Blackwell, and D. J. C. MacKay. Dasher - a data entry interface using continuous gestures and language models. In *Proceedings of the 13th annual ACM symposium on User interface software and technology*, pages 129–137, San Diego, California, United States, 2000. ACM.

[174] D. J. Ward and D. J. C. MacKay. Artificial intelligence: Fast hands-free writing by gaze direction. *Nature*, 418(6900):838, 2002.

[175] C. Ware and H. H. Mikaelian. An evaluation of an eye tracker as a device for computer input2. *ACM SIGCHI Bulletin*, page 183–188, 1987.

[176] E. Wästlund, K. Sponseller, and O. Pettersson. What you see is where you go: testing a gaze-driven power wheelchair for individuals with severe multiple disabilities. In *Proceedings of the 2010 Symposium on*

Eye-Tracking Research & Applications, page 133–136, New York, NY, USA, 2010. ACM.

[177] H. Widdel, A. G. Gale, and F. Johnson. Operational problems in analysing eye movements. In *Theoretical and Applied Aspects of Eye Movement Research, Selected/Edited Proceedings of The Second European Conference on Eye Movements*, volume Volume 22, pages 21–29. North-Holland, 1984.

[178] J. O. Wobbrock, J. Rubinstein, M. W. Sawyer, and A. T. Duchowski. Longitudinal evaluation of discrete consecutive gaze gestures for text entry. In *Proceedings of the 2008 symposium on Eye tracking research & applications*, ETRA '08, page 11–18, New York, NY, USA, 2008. ACM.

[179] A. Yarbus. *Eye Movements and Vision*. Plenum Press, 1st edition, 1967.

[180] S. Zhai and P. Kristensson. Shorthand writing on stylus keyboard. *Proceedings of the SIGCHI conference on Human factors in computing systems*, page 97–104, 2003.

[181] S. Zhai, C. Morimoto, and S. Ihde. Manual and gaze input cascaded (MAGIC) pointing. In *Proceedings of the SIGCHI conference on Human factors in computing systems the CHI is the limit*, pages 246–253, Pittsburgh, USA, 1999.

[182] X. Zhang and I. S. MacKenzie. Evaluating eye tracking with ISO 9241 - part 9. In *Proceedings of the 12th international conference on HCI: intelligent multimodal interaction environments*, pages 779–788, Beijing, China, 2007. Springer.

[183] X. Zhang, X. Ren, and H. Zha. Improving eye cursor's stability for eye pointing tasks. In *Proceeding of the twenty-sixth annual SIGCHI conference on Human factors in computing systems*, pages 525–534, Florence, Italy, 2008. ACM.

[184] Y. Zhang and A. J. Hornof. Improving eye tracking accuracy with probable fixation locations. *Behavioral Research Methods*, 2010.

[185] P. Zielinski. Opengazer: open-source gaze tracker for ordinary webcams. http://www.inference.phy.cam.ac.uk/opengazer/, 2010.

[186] B. L. Zuber, L. Stark, and G. Cook. Microsaccades and the Velocity-Amplitude relationship for saccadic eye movements. *Science*, 150(3702):1459 –1460, Dec. 1965.