Institutionen för datavetenskap
Department of Computer and Information Science

Final thesis

Clicking using the eyes, a machine learning approach.

by

Albin Stenström

LIU-IDA/LITH-EX-A--15/057--SE

2015-10-07

Linköpings universitet
SE-581 83 Linköping, Sweden

Linköpings universitet
581 83 Linköping

Linköping University
Department of Computer and Information Science

Final Thesis

Clicking using the eyes, a machine learning approach.

by

Albin Stenström

LIU-IDA/LITH-EX-A--15/057--SE

2015-10-07

Supervisor: Professor Arne Jönsson

Examiner: Associate Professor Erik Berglund

Abstract

This master thesis report describes the work of evaluating the approach of using an eye-tracker and machine learning to generate an interaction model for clicks. In the study, recordings were done from 10 participants using a quiz application, and machine learning was then applied. Models were created with varying quality from a machine learning view, although most models did not work well for interaction. One model was created that enable correct interaction 80% of the time, although the specific circumstances for success were not identified. The conclusion of the thesis is that the approach works in some cases, but that more research needs to be done to evaluate general suitability, and approaches to make it work reliably.

Acknowledgements

I would not have been able to do this alone, and there are a couple of people I would like to thank for their help.

First would I like to thank Attentec AB, Anders, and employees for giving me the opportunity to work under their roof, and take part of their expertise, interest and helpfulness. I would especially like to thank my company supervisor Mikael for his ideas and thoughts, both when things have been going well, and not.

My university supervisor Arne has given valuable perspectives on the thesis, commented on report drafts, and given good advice. I would also like to thank my test users who have taken time, and energy to provide data, and answers to my questions.

Marianne has provided language proofreading, and comments without delay, without which this report would not be the same. Last but certainly not least would I like to thank my girlfriend Anna, partly for listening, coming up with ideas, proofreading and helping me with planning, but especially for emotional support when things have seemed hopeless.

To everyone I have not mentioned, you are not forgotten, I just lack the space to mention everyone. Thank You!

Contents

Chapter 1

Introduction

This section provides an introduction to this master thesis report. First, the background and motivations for the study will be presented, then the goals. A few acronyms and a glossary are then available for a better understanding of the content of the report.

1.1 Background

Eye trackers have been present on the market for quite some time now, mainly for academic and research purposes, and at a high cost. The main use of this technology is within psychology research, interface evaluation and optimization but also as an interaction tool for people with motion disabilities. Eye trackers have in recent years become considerably simpler and cheaper and Tobii AB, one of the leading eye-tracker manufacturers, recently released a device and Software Development Kit (SDK) aimed at computer interaction for the consumer market. It is called Tobii EyeX, and using an EyeX enabled system, a user can use the gaze and an activation button to perform clicks on the screen. Another common technique in the industry for selecting things is the so called dwell time where the user needs to stare at a particular point for a selected timespan [Kandemir and Kaski, 2012].

Tobii claims that explicit monitor tasks such as dwell time or blinking to click puts strain on the eyes [Tobii AB, 2014a]. Therefore, a way works naturally with eye movement and adapts, could therefore be a good contribution to the eye tracking research.

1.2 Goal

The goal of this thesis is to evaluate the possibility of enabling a user to interact with a system using gaze selection without needing to use a preconfigured dwell time, physical button or other static behaviour. This would instead be done by training a system to recognize how the eye movements of a specific individual using the system. This corresponds to clicks or activation behaviour to make interaction more natural. Additionally, the goal is to evaluate this for a device aimed at the consumer market, making the study closer to the real world application than if done on a high end research device.

1.3 Acronyms

ANOVA Analysis of Variance. 21

AOI Area Of Interest. 2, 9, 24–28, 44, 48, *Glossary:* Area Of Interest (AOI)

GUI Graphical User Interface. 2, 24, 42, 44

PCA Principal Component Analysis. 20, 24, 31, 37–40, 46, 57

RBF Radial Basis Function. 19, 20, 31

RMS Root Mean Square. 8

SDK Software Development Kit. 1–3, 13–15, 25, 27, *Glossary:* Software Development Kit (SDK)

SVC Support Vector Classification Machine. 18, 31, 32

SVM Support Vector Machine. 18, 31, 46

SVR Support Vector Regression Machine. 18

1.4 Glossary

Area Of Interest (AOI) An area of interest is an area of a Graphical User Interface (GUI) that is of special relevance to the study. A number of different measures can be calculated from the behaviour of the eyes in relation to one or multiple AOI's. Please see 2.2.2.4 for more information. 2, 9, 24–28, 44, 48

dwell A dwell, or dwell session represents the user looking at an Area Of Interest (AOI). It is associated with a couple of eye tracking measures, such as dwell time that is the time spent in a dwell. For more information, see 2.2.2. 9, 10, 12, 13, 24, 28–30, 32, 37, 44, 45

fixation A fixation is the name of when the gaze rests on a specific feature for a certain period of time. It is also the name of an eye-tracking measure of said behaviour. Please see section 2.1.2 or 2.2.2.3 for more information. 3–6, 8–10, 12, 14, 15, 24, 28, 30, 31

ground truth The part of the classification samples that represent the known or expected class of the samples prior to classification. 22, 39, 40, 56

midas touch Activating something by just looking at it, without intending to. Comes from the story about king Midas, who turned everything he touched into gold, including friends. See 2.3.1. 12, 25, 38, 45

saccade A saccade is the name of rapid movement between fixations. It is also the name of an eye-tracking measure of said movements. Please see section 2.1.2 or 2.2.2.5 for more information. 5, 6, 9–12, 31, 48

Software Development Kit (SDK) A Software Development Kit (SDK) is a framework or library created by the creators of a device or system that enables abstractions and interfaces to the said device or system. 1, 2, 13–15, 25, 27

ZeroMQ A communication library focused on ease of use, speed and versatility. It provides communication tools for an array of different network models, both internally to a computer and between computers. 32

Chapter 2

Theory

This chapter will present the underlying theory that was needed to conduct this study and begins with delving into the behaviour of the eyes, and continues to the basis of eye tracking. Then, the characteristics of the specific eye-tracking system used in the study is explored, followed by eye-tracking study methodology. The chapter ends with a section about machine learning.

2.1 Properties of the human eye

This section will go thorough details of the properties and movements of the human eyes. This is important, because understanding of how the eyes work are an important factor for being able to understand many of the concepts related to eye tracking.

The human eye is a vastly researched topic, and a lot is known about it, although there is still disputes concerning certain topics, such as lengths of fixations (see 2.1.2). This section will present a few properties of the eyes, their movements and their impact on eye tracking.

The human eye is fast, much faster than for example moving a pointer using a mouse. Its movements are also largely involuntary and unconscious although it is possible with effort to move the eyes in a controlled way [Majaranta and Bulling, 2014, p.48]. It may therefore be beneficial to use these unconscious movements instead of controlled movements that require less effort.

A drawing of the human eye and its parts can be seen in figure 2.1, to visually place parts of the eye described in the following sections.

2.1.1 Foveal and Peripheral vision

The retina is a light sensitive area on the back of the eye that converts light to electric signals to our brain. It contains two types of light receptors called

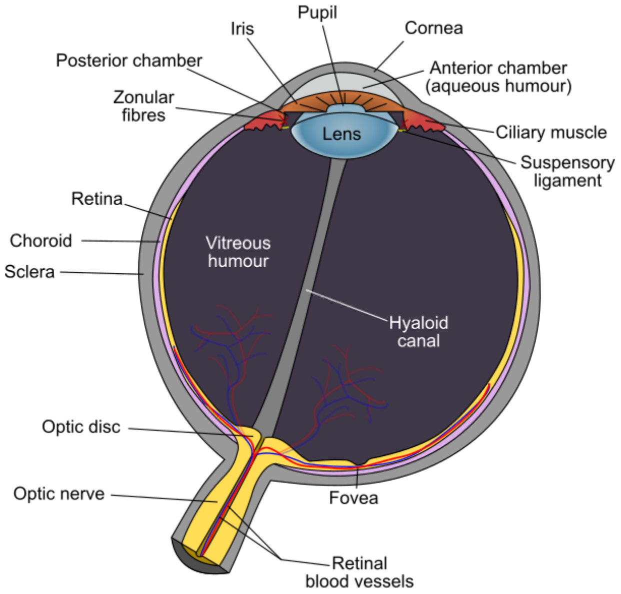


Figure 2.1: Drawing of the human eye. By Rhcastilhos [Public domain], via Wikimedia Commons

cones and rods. Cones provide visual detail and colour, and rods provide vision in the dark [Holmqvist, 2011, p.6]. A small area on the retina called the fovea has a higher density of cones, resulting in a small area of high resolution vision, the foveal vision . The size of this area depends on the distance to the focused object, but takes about 2° of the vision [Holmqvist, 2011, p.21; Nielsen and Pernice, 2010, p.6]. The foveal vision is the only area of the vision where objects can be viewed sharply. Reading for example can only be done using the foveal vision.

The peripheral vision is not as sharp as the foveal vision, but can be used to find interesting features to focus on with foveal vision [Duchowski, 2007, p.11], such as the beginning of a word or an eye of another person. Movement is even slightly better detected with the peripheral vision.

2.1.2 Fixations and Saccades

As a result of foveal and peripheral vision, a person moves the eyes around to create a sharp mental image, by focusing on items of interest. This is contrary to common belief not done in smooth movement, but in short bursts [Nielsen and Pernice, 2010, p.6]. These small bursts are called saccades and have a duration somewhere between 10 and 100 milliseconds ($30 - 80$ according to Holmqvist [2011]). This is fast enough that it effectively renders the eye blind for the duration of the saccade [Holmqvist, 2011, p.23; Nielsen and Pernice, 2010, p.7].

The time between saccades, spent focusing on a specific point is called fixations. The duration of fixations is not agreed upon by the literature. Holmqvist [2011, p.21-22] claims that a fixation is *"from some tens of mil-*

liseconds up to several seconds", Nielsen and Pernice [2010, p.7] claims that *"Fixations typically last between one-tenth and one-half second"* and Duchowski [2007, p.47] claims a duration between 150 and 600 ms. This suggests that although the role of fixations is clear, there are discrepancies regarding its definition.

Although fixations are focused on a specific point, the eye still moves slightly. The eye slowly drifts from the point, and a microsaccade brings it back [Holmqvist, 2011, p.22]. There are also small tremors in the movement of the eyes during a fixation. These movements are small, but can be quite fast regardless. Absolutely no movement in the vision would actually cause the vision to fade away within a second [Duchowski, 2007], and thus, these movements are in fact important to retain vision.

2.1.3 Smooth Pursuit

An exception to the rule that only saccades move the eyes between fixations is that if the eyes have something that moves slowly in front of them, they can follow it smoothly. This is done by matching the speed of the object [Duchowski, 2007, p.45]. There are according to Holmqvist [2011, p.178] studies that suggests this is the only exception to the rule.

2.2 Eye Tracking

This section will go through the workings of an eye-tracker as well as eye-tracking measures that are deemed relevant to this study. It ties closely into 2.1 by connecting eye movements with behaviour and important to understand parts of the method chapter concerning different eye tracking values.

2.2.1 Tracking methods

This section describes three different ways of tracking the eyes and their respective characteristics, to provide a background on positive and negative consequences of different tracking methods.

2.2.1.1 Electro-OcluoGraphy

This tracking technique consists of measuring differences in electric potential on the skin around the eyes, enabling tracking of eye movement relative to the head. A head tracker in conjunction with this technique can enable gaze measurement on a screen [Duchowski, 2007, p.57], where the head tracker measures the location and rotation of the head, and the eye tracker tracks the eyes relative the head. Advantages of this approach are that eye movement can be tracked regardless of lighting conditions, even when the eyes are closed [Majaranta and Bulling, 2014, p.45].

2.2.1.2 Scleral Contact Lens

This technique uses a special contact lens that is put on the eye. The contact lens is then connected to the measurement equipment, either mechanically, visually or magnetically to track the users eye movement. This method is intrusive, and requires care when inserting and could interfere with movement patterns of the eyes. It is on the other hand a very precise way of tracking the movement of the eyes [Duchowski, 2007, p.57].

2.2.1.3 Video-based tracking. Pupil and Corneal reflection

This type of eye tracking uses relatively simple cameras and image processing units to provide gaze point and other measures in real time [Duchowski, 2007, p.54]. A light source, usually infra-red, is used to create reflections in the cornea, also called Purkinje reflections [Holmqvist, 2011, p.21]. Figure 2.2 shows how the different purkinje reflections are created in the eye. The the different reflections are created by different parts of the light reflecting in different layers of the eye, and then angled by the layers.

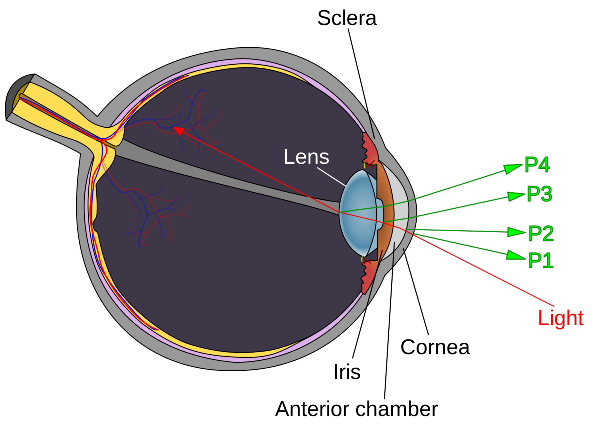


Figure 2.2: A drawing showing how the purkinje-reflections are created by refraction, and reflection in the eye. 🅭🅯🄎 Z22 @ Wikimedia Commons

The first of these reflections, and sometimes additional reflections, together with the pupil in an image of the eye can be used to calculate the position of the pupil relative to the camera and light source as well as the direction of the gaze [Majaranta and Bulling, 2014, p.44]. This is possible because the first Purkline reflection is relatively stable regardless of eye rotation [Duchowski, 2007, p.57]. A step by step breakdown of the approach is shown in figure 2.3

This technique is suitable for monitor mounted systems since the reference point is external from the user, but it is sensitive to lighting conditions since extra light in the infra-red spectrum can give extra reflections that the tracker misinterprets [Majaranta and Bulling, 2014, p.45].

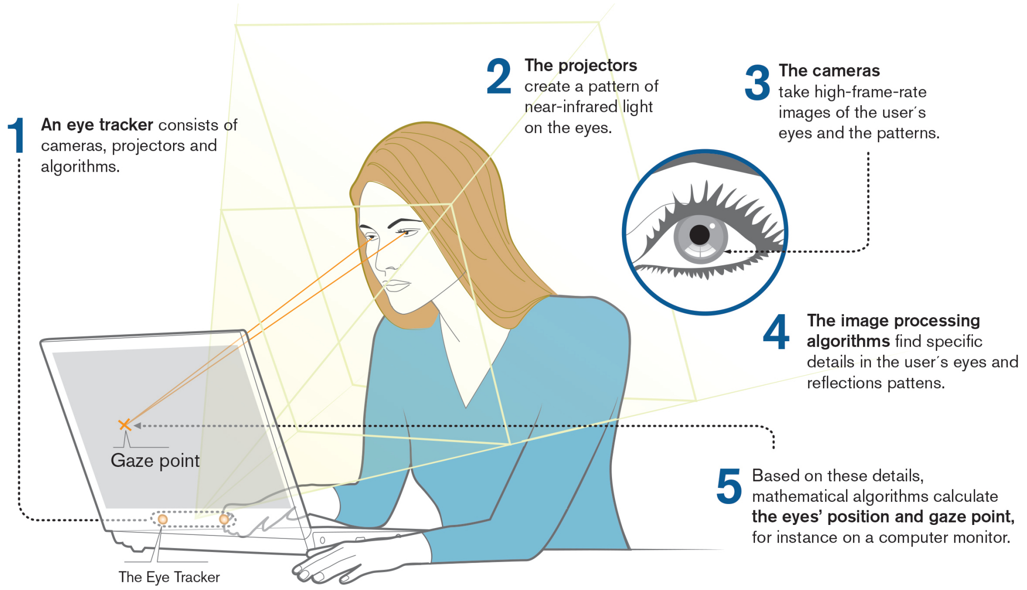

1 An eye tracker consists of cameras, projectors and algorithms.
2 The projectors create a pattern of near-infrared light on the eyes.
3 The cameras take high-frame-rate images of the user´s eyes and the patterns.
4 The image processing algorithms find specific details in the user´s eyes and reflections pattens.
5 Based on these details, mathematical algorithms calculate the eyes' position and gaze point, for instance on a computer monitor.
Gaze point
The Eye Tracker

Figure 2.3: Video-based tracking, step by step. Image reproduced with permission. ©Tobii AB

2.2.2 Metrics and Measures

This section will present different eye-tracking measures, how they are calculated, used and what they signify. In addition, concepts related with eye-tracking measures will be presented to provide the context of said measures.

2.2.2.1 Gaze

Gaze coordinates can be seen as the raw data that the eye-tracker extracts from the eye images with other measures often calculated from this data.

2.2.2.2 Position Measures and dispersion

Position of gaze and fixations varies over time and therefore, it is sometimes important to group multiple events into a fixed number of measures. Averages can give a fixed point, but the movement is then lost. Dispersion is a measure of how far from the average value the positions move [Holmqvist, 2011, p.360-362]. The most common of these are Standard deviation, variance and Root Mean Square (RMS). All three give a measure of dispersion with slightly different characteristics, but works well together with an average to group positions.

2.2.2.3 Fixations

A fixation as a measure is generally represented as a location, a start time, and a duration, but there are many different definitions, and algorithms used for detection. Researchers often speak of fixations generally, without specifying what definition and algorithm they are using. The fixation duration is perhaps the most used eye-tracking measure in research, but different definitions and processing may create variations in duration between studies [Holmqvist, 2011, p.377].

Users repeating a task generally have similar average fixation duration between repetitions, but there is a great difference in mean duration between different users for the same task. Factors such as stress, expertise, processing depth, and usability also affect the fixation duration [Holmqvist, 2011].

The number of fixations in an Area Of Interest (AOI) is a measure that primarily can be used as a comparison measure between AOI's, but sometimes also as a measure of attention to a specific AOI [Holmqvist, 2011, p.412-413]. Fixation rate is defined as a number of fixations per time period, and is roughly inversely proportional to mean fixation duration length [Holmqvist, 2011, p.416]. It has regardless been used as a measure in its own right in some research fields.

2.2.2.4 Area Of Interest

An AOI is a region that is of significance to the researchers, both regarding what areas the user looked at, and how they looked at a specific AOI. There are also a number of measures that can be calculated specifically of how a user looks at the AOI, most commonly dwells, transitions and AOI hits [Holmqvist, 2011, p.187]. An AOI hit is the fixation or raw sample that is first to enter the AOI, the time of the hit, and the number of hits at a specific AOI is often used as parts of other measures. Transitions track the order that the gaze moves between different AOI's [Holmqvist, 2011, p.189-190]. This can then be used to calculate probabilities for the gaze moving between two specific AOI's.

Dwell is a name for one visit, entry to exit in an AOI. The dwell time, or dwell duration, is the time between entry and exit, and is commonly used for interaction [Majaranta and Bulling, 2014, p.49]. Other eye-tracking events during a dwell are often part of the dwell and are often analysed as a group [Holmqvist, 2011, p.190,357], and position data may sometimes be represented relative to the AOI.

The duration of the first, and sometimes the second fixation in an AOI is sometimes used as a measure of recognition, identification and text processing [Holmqvist, 2011, p.385].

2.2.2.5 Saccades

A saccade is a representation of the fast movement between fixations. It is sometimes defined and detected as such, but can also be defined by thresholds of velocity or acceleration. Saccadic amplitude, is a commonly used measure that behaves differently between individuals, but consistently over tasks for specific individuals [Holmqvist, 2011, p.312-315]. The amplitude of a saccade is often small, but task difficulty, cognitive load, age, and text characteristics have an impact on the size.

Closely related to the saccadic amplitude, but still somewhat different is the saccadic duration. It can even be roughly calculated from the saccadic amplitude. Regardless, it is often used in neurology and pharmacological research, but rarely in human factors [Holmqvist, 2011, p.312-322].

Saccadic velocity, is often used for detecting saccades (see 2.2.2.6), but can also be used as a separate measure. The saccadic velocity is often in the form of a sharp peak, dividing measures into average velocity, peak velocity, and time between saccade onset and peak velocity [Holmqvist, 2011, p.326-329].

The velocity average gives a rather poor image of the shape of the saccadic velocity. Anticipation, task, age, and drowsiness has an impact on the saccadic velocity.

Number of saccades and saccadic rate are sometimes used, but they correlate strongly to corresponding measures for fixations (see 2.2.2.3) and are seldom useful together with said measures [Holmqvist, 2011, p.404].

2.2.2.6 Fixation and Saccade Detection

There are two common ways of identifying fixations and saccades [Duchowski, 2007, p.138]. Dwell time fixation detection and velocity-based saccade detection, both explained below. Both these techniques detect one of the types of measures and can by implication find the other in the process. There are also hybrid methods of these two.

Dwell-Time Fixation Detection Also called dispersion-based algorithms [Holmqvist, 2011, p.171], this technique revolves around averaging the gaze coordinates. A low variance or distance signifies a candidate fixation, and if the variance or distance is low continuously for a specified duration, then it signifies a fixation [Duchowski, 2007, p.38-41]. Either a threshold on maximum length of fixation, a too high movement variance, or too high distance from the fixation center can end the fixation.

Velocity-based saccade detection This technique revolves around using the velocity of the gaze to identify saccades. When the velocity of the gaze point, or rather the distance between two gaze samples, crosses a threshold is it considered a saccade, otherwise is it part of a fixation [Duchowski, 2007, p.141]. The acceleration of the gaze can also be used. This technique is sometimes divided into two, fixation detection (dwell-time based) and saccade detection (velocity based), to pinpoint differences in detecting the two, but the principles are still the same [Holmqvist, 2011, p.171-175]

2.2.2.7 Blinks

Blinks cause eye trackers to lose data, and generally the eye tracker will instead put out no data or zero(0) data for gaze point during that time. It is also common that a saccade like movement downward when eyes are closing, and upwards when eyes are opening just after and before data loss [Holmqvist, 2011, p.177].

According to Holmqvist [2011, p.177], are there few articles about blink detection, and the articles that do provide information about blink detection usually do so in their data analysis. Bonifacci et al. [2008] uses data loss for more than 96 ms as a lower threshold for blinks, while many others uses combinations of data loss duration, gaze point movement and variations in pupil diameter [Holmqvist, 2011, p.177].

Another, more reliable method of detecting blinks is to analyse the video images directly, and detect when the eyelid starts to cover the pupil [Holmqvist, 2011, p.176].

Regardless, it is used in research, and Bonifacci et al. [2008] as well as others have used intentional blinks as an interaction method. Blink duration and blink rate are both often used as measures.

2.2.2.8 Pupil dilation

Pupil dilation, or pupil diameter is a measure that can be used to study cognitive and emotional states. Mental workload, strong emotions, sexual arousal, pain and some drugs increase the pupil dilation [Holmqvist, 2011, p.393-394]. Moreover, fatigue, diabetes and age decrease the pupil dilation. However, the factor that matters most for the pupil dilation is luminance. In light environments, the pupil dilation decreases and in dark situations the pupil dilation increases [Holmqvist, 2011, p.392]. This makes it very important control the lighting of the environment if pupil dilation is used.

2.2.2.9 Tracker Frequency and Measures

The frequency of an eye-tracker defines how often a measurement of the eyes is done, and can have a great impact on performance depending on what the tracker is used for. A sampling frequency of f_s gives a time between samples of $\frac{1}{f_s}$. This results in that the absolute difference between the measured time of an event and the actual time, or simply the absolute error is uniformly distributed over $\left[0, \frac{1}{f_s}\right]$ [Andersson et al., 2010, p.4]. This is because an event between two samples cannot be detected before the time of the second sample. This also gives a mean error of $\frac{1}{2f_s}$.

Similarly, a duration measure, uses two samples and therefore gets one error term from each. The first error adds $\left[-\frac{1}{f_s}, 0\right]$ because it is not possible to know how much before the sample the real start event took place. The second error adds $\left[0, \frac{1}{f_s}\right]$ since the end event theoretically could have ended almost one sample after the recorded sample. The added error is in $\left[-\frac{1}{f_s}, \frac{1}{f_s}\right]$ and the probability distribution is in the form of a triangle. This results asymptotically in a normally distributed error with 0 mean, but with a variance of $\frac{1}{18nf_s^2}$, where n is the number of samples [Holmqvist, 2011, p.31; Andersson et al., 2010, p.4-5].

There are according to Holmqvist [2011, p.32] some disagreement regarding what sampling frequency is required to measure saccadic peak velocity. Most agree that a frequency higher than 50 Hz is needed, but some argue that quite a lot more is needed. Since a saccade is only $30 - 40$ ms a 50 Hz tracker would only be able to register a saccade using one or two samples.

2.3 Gaze Interaction

This section presents a challenge of interaction using the gaze, as well as a few examples of how others have done similar things, or things that has a implications on this study.

2.3.1 Midas Touch

Midas touch is one of the great challenges of gaze interaction, and means that the system clicks at everything the user looks at [Majaranta and Bulling, 2014, p.48]. Like king midas in the legend who could not touch anything without turning it into gold, this might seem empowering at first, but quickly becomes an obstacle [Jacob, 1990, p.12]. An example of such behaviour is if a user reads on a button, but before reading the whole text, the button is selected. The challenge consists of letting the user look around freely without action, but select something in an effective way when the user wants to [Jacob, 1990, p.13].

2.3.2 Interaction Examples

Kandemir and Kaski [2012] used eye-tracking and machine learning to create a model that could predict if a painting was liked, or relevant to the user or not. They compared the result to results using only dwell time to do the same task, as this was considered one of the most prevailing approaches. Six different eye-tracking measures were used as features for machine learning: mean and standard deviation of saccade length, fixation duration and pupil dilation respectively, and each were calculated for 3 intervals of 1 second each. This study did not discuss interaction in it's strictest sense, but it discusses related concepts.

Jacob [1990] discusses a situation where the result of selecting an item is trivially reversible. This enables a selection from a dwell time of $150 - 250$ ms to perform well. The situation used in the article is that the selection only changes text content in a window adjacent to the one where the items are selected. They also present a similar technique for scrolling using gaze, namely looking for a period on the bottom of the text. This as the earlier example is easily reversible, that is, it is possible to easily, and fast remedy the mistake.

Both Kandemir and Kaski [2012] and Jacob [1990] comment on that long dwell times are a common approach for selection, but that it forces the user to suppress the involuntary movements the eyes do naturally.

A blink with both or one eye, spoken confirmation and manual switches are also sometimes used to initiate selection on the item the user is looking at [Majaranta and Bulling, 2014, p.49]

2.4 Tobii EyeX

The eye tracker used in this thesis was a Tobii EyeX Controller, and is used together with Tobii EyeX Software Development Kit (SDK) and Tobii EyeX Interaction. These products are aimed at the consumer market as an interaction tool instead of as an academic and corporate research tool as is norm today.

2.4.1 Tobii EyeX Controller

The Tobii EyeX Controller is a development device intended for development of gaze enabled applications and systems. It is not directly intended for the consumer market, but is similar and exchangeable to the Steel Series Sentry device that is available on the market and is a collaboration between Tobii and Steel Series [Tobii AB, 2015a]. Steel Series Sentry is aimed mainly towards the gaming market, but additional devices could become available through partnerships between Tobii and other companies.

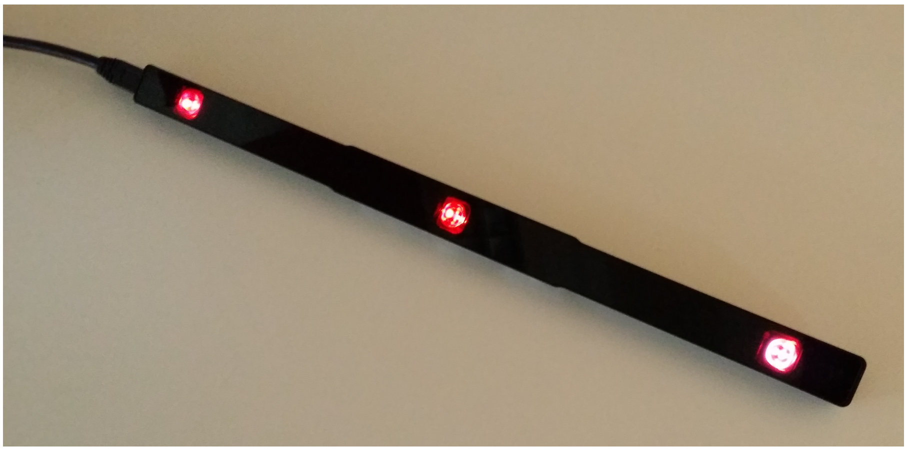


Figure 2.4: The Tobii EyeX Controller. ©Albin Stenström

The controller uses three infra-red light sources, to create images that are run through image processing to extract eye data, see figure 2.4 [Tobii

AB, 2014b]. It is connected to the computer via USB 3.0 and is mounted on the screen using magnetic mountings, see figure 2.5.

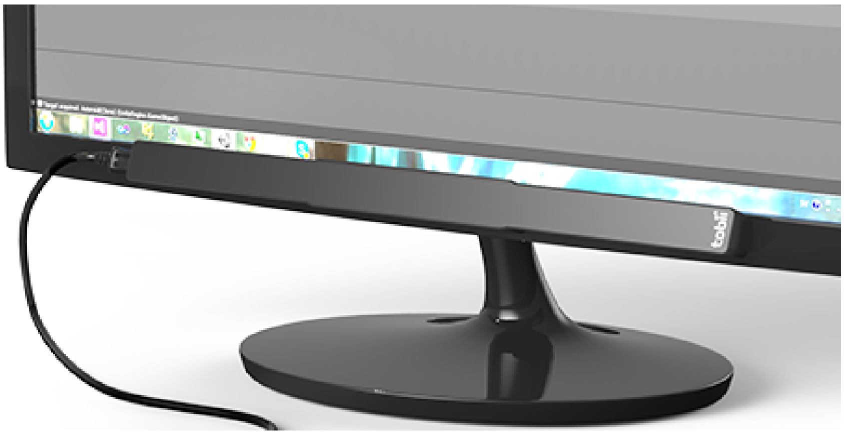


Figure 2.5: The Tobii EyeX Controller mounted on a monitor. Image reproduced with permission. ©Tobii AB

There are to my knowledge no official technical specification of the EyeX Controller. Since the controller is made for a consumer market with regards to price and power, the EyeX controller does not offer a stable sample rate, but the frequency is at least $55Hz$ at all times according to a Tobii employee [Tobii AB, 2014c]. This means that the absolute error on timestamps for events is uniformly distributed over $[0, 18]\,ms$ with a mean of $9ms$. Duration measure error is normally distributed over $N\,(0, 18)\,\mu s$ for single samples, giving a 95:th percentile of $4.2ms$. See 2.2.2.9 for the formulas.

2.4.2 Tobii EyeX SDK

The EyeX SDK provides an interface towards the eye tracker and its services, and abstractions that can hide some of the underlying complexity of the communication. The SDK is available for .NET/C#, C/C++, Unity and Unreal Engine, this section will describe only the workings of the .NET/C# version [Tobii AB, 2015b].

EyeX SDK provides a number of global data event streams that the user can use to get information about the eyes, but also the status of the device. The data streams with information about the eyes contain gaze points, eye positions and fixations respectively.

The gaze point stream provides the user with timestamped screen coordinates where the user is looking and can be configured to be unfiltered or lightly filtered to remove noise. The eye position stream provides timestamped 3D coordinates for the positions of both eyes relative to the tracker. The data can also be presented in a normalized form. The fixation data stream provides three types of events, begin, data and end each containing a screen coordinate and a time stamp. A group of one begin event, indefinite data events and one end event represent one fixation.

When the EyeX SDK is unable to record the gaze, due to absence of user, obstructions of eyes or blinking, the different event streams reacts differently. The gaze position stream dispatches no events, the eye position stream dispatches events with zero data, and the fixation stream dispatches no events.

Higher abstractions consists mainly of that a button or other clickable Windows Forms components can be made activatable. This makes it possible to look at the component and press a configurable button on the keyboard to make a click on the component. Additionally, it is possible to declare a component gaze aware, causing an event to be raised every time the gaze of the user enters the component.

It is possible to create this behaviour for other components that are not inheriting from a forms control. This involves answering queries from the framework about what components are present in a specific area, and catching activation and gaze enter events and dispatch them to the correct component. A visialisation of this can be seen in figure 2.6.

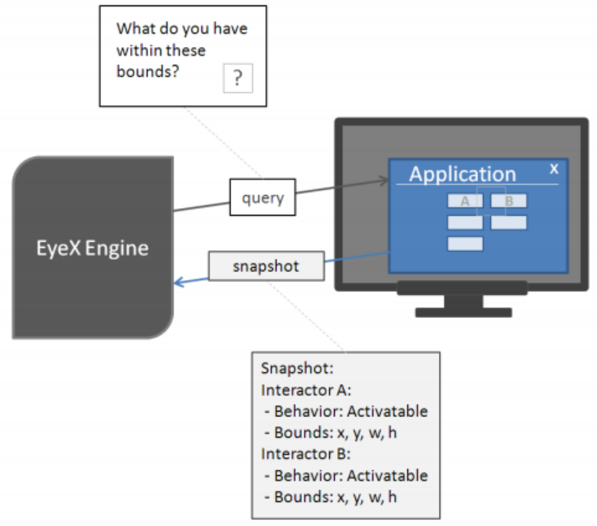


Figure 2.6: Visualisation of the interaction with the EyeX Framework.

2.5 Eye tracking user studies

This section goes through the theory and methodology of eye-tracking studies.

2.5.1 Study environment

The physical environment of a user study using eye trackers is an important factor for reliable results. The lighting of the environment is critical as a result of the eye tracker's usage of the infra-red light spectrum see 2.2.1.3.

The light of the sun consists of a large part infra-red light and should therefore not be allowed to reflect in the user's eyes or hit the tracker itself [Holmqvist, 2011, p.125]. To ensure consistent data recording, care must be taken to ascertain that the environment, the positions of the participants, and the equipment are as constant as possible during all sessions to limit external causes of errors.

The eye responds instinctively to sounds and peripheral movement (see 2.1.1) and it is therefore important to make sure that movement and sounds during an eye tracking session are minimal [Holmqvist, 2011, p.17]. This can cause trouble for some eye tracking user studies where spoken cues or think aloud strategies are used [Duchowski, 2007, p.171]. Additionally, movements on the screen are discouraged unless it is part of the study.

When using a monitor attached tracker, the monitor needs to be placed one a stable table that stands on a cement floor to make sure that no vibrations can disturb the recording. Holmqvist [2011, p.35] shows that mouse clicks on the same table as the tracker can cause vibrations that disturb accurate eye recordings.

2.5.2 Participants

Glasses and contact lenses may cause loss of accuracy for eye tracking because of reflections in the glass and air bubbles respectively [Holmqvist, 2011, p.122-125; Duchowski, 2007, p.97]. This causes some participants wearing glasses or contact lenses to be discarded during many eye tracker user studies and it is recommended to not use participants with glasses [Holmqvist, 2011, p.141].

There are some researchers [Holmqvist, 2011, p.79; Kandemir and Kaski, 2012, 88] that keep the participants ignorant of the actual purpose of the eye-tracking study until the study is done so that the users knowledge does not interfere with the study. *"If a participant knows that the researcher wants to find this result, the participant is likely to think about it and to want to help, consciously or not, in obtaining this result, thus inflating the risk of a false positive."* [Holmqvist, 2011, p.79]

2.5.3 Calibration

It is vital to make individual calibrations of the eye-tracker for each participant to ensure that the quality of the data gathered from the session is of good quality. This is vital because size and shape of the eyes vary among the population, causing the geometric calculations to fail if not calibrated successfully [Holmqvist, 2011, p.128]. Additionally, glasses alters the perceived size of the eyes. Calibration is best done by showing the user a number of dots spread over the screen and asking the user to look at them, giving the possibility to calibrate the eye model using mathematical models.

It is addition to calibration important to test how good a calibration is, with a calibration verification [Holmqvist, 2011, p.132]. This is done using the previously mentioned points and asking the participant to look at them again. The calibration can either be checked by generating a point for where the gaze lands, or calculating a closeness measure from the data. Doing a calibration validation after the session is done as a way to measure or estimate if any drift has occurred during the session.

2.5.4 Likert questionnaires

Likert questionnaires are a good way of gathering quantitative measures of the user's subjective experience of the system and is appropriate for eye tracking studies [Holmqvist, 2011, p.96; Nielsen and Pernice, 2010, p.32]. A likert questionnaire consists of a number of statements and the participant is told to mark a number between 1 and 5 where 1 means that the user do not agree with the statement at all and 5 means that the user fully agrees.

2.6 Machine Learning

This section will present the theoretical foundations of supervised machine learning for classification and other machine learning components that are used during this study.

Supervised machine learning is the task of generating a hypothesis function h from a training-set of example input-output pairs (x_j, y_j) generated from an unknown function f [Russell and Norvig, 2014, p.706]. The goal is that for every (x, y), $y = h(x) = f(x)$. If output y can take only a finite number of values this is called classification, and if only two values are allowed, binary classification. Otherwise, it is called regression.

The input x is often called a feature vector, and contains a number of values, also called features. These features are sometimes the original data, but is often calculated, or extracted from the original data. This is either done by specific algorithms, or manually.

It is common to apply normalization on each feature before learning is done to make sure that a feature with a greater range of values, or generally higher values are not given a greater impact on the results [Russell and Norvig, 2014, p.750]. It is also common to calculate more features than needed, and let a dimension reduction algorithm reduce the dimension of the feature vector in a way that should give the best classification results according to some criterion.

2.6.1 Cross Validation

Machine learning sometimes suffers from a problem called overfitting. This means that the a machine learning algorithm is able to classify the samples it is trained on well, but does not generalize well to other, similar data [Russell and Norvig, 2014, p.716,707]. The model fits the specific data used to well, and does not capture common characteristics. This may be caused by a combination of a too expressive model and algorithm parameters that make the algorithm too sensitive to specific samples.

Cross validation is a technique, where the set of samples are partitioned into one training set, and one validation set. The model is built by using the machine learning algorithm on the training set, and scored using the validation set [Russell and Norvig, 2014, p.719]. This way, the score measures how well the model generalizes to data from the same set, but that it was not trained on.

When trying to find good parameters for a machine learning algorithm, information about the validation set may still leak into the model based on what values for the parameters worked best. This can be mitigated by using a k-fold instead [Russell and Norvig, 2014, p.719-720]. A k-fold means that the samples are divided into k parts, and for each parameter set, five models are created and scored so that for each model, a different part is kept as validation set. The score can then be calculated as a mean of these. To be completely sure that the created models generalizes well, a dedicated test set should be used to score the final model.

2.6.2 Support Vector Machines

Support Vector Machines (SVMs) is a group of algorithms for supervised machine learning. They are sometimes divided into Support Vector Classification Machines (SVCs) and Support Vector Regression Machines (SVRs). This section will focus on SVCs, but the principles of SVR are basically the same.

They work by constructing a decision boundary with a maximum distance between the supplied example inputs of different classes. The distance between these example inputs of different types are called the margin of the decision boundary. This is done linearly, but in a higher dimensional feature space, where the examples may be or nearly be linearly separable, creating an non-linear separator in the original feature space [Russell and Norvig, 2014, p.755]. An example set is linearly separable if a line or hyperplane described by a linear equation can divide the samples into the correct groupings. If the samples are not linearly separable, then a soft margin can be used, meaning that samples that are on the wrong side of the boundary are assigned a penalty proportional to the distance of the sample from the boundary [Russell and Norvig, 2014, p.759].

SVMs keep a number of examples, or support vectors, which are the points that constrain the decision boundary [Russell and Norvig, 2014, p.757], this means that if a support vector is changed, then the boundary will move. The optimal solution in the original feature space is found by solving equation 2.1, where $\boldsymbol{\alpha}$ is a vector that contains the different weights α_i that is associated with corresponding sample input, and output $\boldsymbol{x}_i$ and y_i.

$$\underset{\boldsymbol{\alpha}}{\arg\max} \sum_{j} \alpha_j - \frac{1}{2} \sum_{j,k} \alpha_j \alpha_k y_j y_k (\boldsymbol{x}_j \cdot \boldsymbol{x}_j), \alpha_j \geq 0, \sum_{j} \alpha_j y_j = 0 \tag{2.1}$$

It should be noted that this will cause $\alpha_j = 0$ for all feature samples closest to the separator, $\alpha_j \neq 0$ gives support vectors, the only samples that need to be kept.

A kernel function, takes two input vectors, and calculates the dot product of them in the high dimension feature space without converting the vectors to coordinates in that feature space directly. This enables learning in high dimension feature spaces, but limit calculations to the kernel function on each pair of support vectors[Russell and Norvig, 2014, p.758]. If $F(\boldsymbol{x})$ is a function that converts a vector to the higher dimension space, then the kernel function K, corresponds to F like $K(\boldsymbol{x}_j, \boldsymbol{x}_k) = F(\boldsymbol{x}_j) \cdot F(\boldsymbol{x}_k)$. A kernel is seldom defined by the mapping function, but it is the kernel function that generates the mapping.

According to Mercer's theorem, any "reasonable" kernel function corresponds to the dot product in some feature space. By replacing the dot product in equation 2.1, learning is linearly done in the high dimension feature space, resulting in a non-linear separator in the original feature space [Russell and Norvig, 2014, p.758].

2.6.2.1 Kernels

This section explains a few common kernels, and describes their characteristics.

Linear A linear kernel is the dot product, resulting the original equation 2.1. Optionally, a constant can be added, resulting in $K(\boldsymbol{x}_j, \boldsymbol{x}_k) = \boldsymbol{x}_j \cdot \boldsymbol{x}_j + c$. A linear kernel results in no higher dimensional space, and the separator will be linear in the original space [Cesar Souza, 2010].

Polynomial A polynomial kernel is basically a linear kernel taken to the power of d, and can be seen in equation 2.2. This corresponds to a feature space with a dimension that are exponential in d. The slope α, constant term c, and the degree d needs to be adjusted to fit the problem accurately for good results [Russell and Norvig, 2014, p.758].

$$K(\boldsymbol{x}_j, \boldsymbol{x}_k) = (\alpha \boldsymbol{x}_j \cdot \boldsymbol{x}_k + c)^d \tag{2.2}$$

RBF A Radial Basis Function (RBF) kernel is a kernel that depend on the distance between the input vectors. The most popular variant is the *Gaussian Kernel*, defined as equation 2.3 [Schölkopf and Smola, 2002, p.21] or 2.4 [Scikit-learn developers, 2014].

$$K\left(\boldsymbol{x}_j, \boldsymbol{x}_k\right) = exp\left(-\frac{||\boldsymbol{x}_j - \boldsymbol{x}_k||^2}{2\sigma^2}\right) \tag{2.3}$$

$$K\left(\boldsymbol{x}_j, \boldsymbol{x}_k\right) = exp\left(-\gamma||\boldsymbol{x}_j - \boldsymbol{x}_k||^2\right) \tag{2.4}$$

The only difference between equation 2.3 and 2.4 is how sensitive the kernel is to its tuning parameter. It is possible to convert between the versions using $\gamma = \frac{1}{2\alpha^2}$ provided that both γ and σ are positive. A too big σ will result in an almost linear kernel. On the other hand, a too small σ makes the algorithm sensitive to noise. The Gaussian kernel can be proved to correspond to a feature space with infinitely many dimensions [Schölkopf and Smola, 2002, p.47].

Other RBF kernel variants are the *Exponential* and the *Laplacian* kernels (see equation 2.5 and 2.6), both leaving out the square of the distance between the kernels, and in the case of the Laplacian kernel, being less sensitive to changes in σ [Cesar Souza, 2010].

$$K\left(\boldsymbol{x}_j, \boldsymbol{x}_k\right) = exp\left(-\frac{||\boldsymbol{x}_j - \boldsymbol{x}_k||}{2\sigma^2}\right) \tag{2.5}$$

$$K\left(\boldsymbol{x}_j, \boldsymbol{x}_k\right) = exp\left(-\frac{||\boldsymbol{x}_j - \boldsymbol{x}_k||}{\sigma}\right) \tag{2.6}$$

2.6.3 Dimension Reduction

This section presents two different approaches for dimension reduction of the feature vector. This is sometimes called feature selection if a set of features are selected, or feature extraction if a new set of features are calculated from the original ones.

2.6.3.1 PCA

Principal Component Analysis (PCA) is a feature extraction algorithm that transforms a feature vector x into a base $\boldsymbol{\alpha}$, where each component z_i in the new vector z together with α_i represents a principal component of the original data. A principal component is a vector that represents as much of the variance in the original samples as possible. Each subsequent component in the new vector represents less of the variance in the original vector samples, and if the number of components are the same as the size of the samples, no data is lost [Jolliffe, 2002, p.1-2].

Although removing the components that describes the least variance reduces the retained information, the information lost is not linear to the number of components removed, making it possible to reduce the dimension at a low information loss. The first principal component basically describes the most common linear deviation from the mean values of the original vectors, and each successive component describes in the same way the variance not described by the preceding component. The mathematical formulation is defined as follows.

$$\boldsymbol{Z} = \begin{pmatrix} \boldsymbol{\alpha}_1^\top \\ \boldsymbol{\alpha}_2^\top \\ \vdots \\ \boldsymbol{\alpha}_k^\top \end{pmatrix} \cdot \boldsymbol{X} \tag{2.7}$$

$$\boldsymbol{\alpha} = \big(\boldsymbol{\alpha}_1, \boldsymbol{\alpha}_2, \ldots, \boldsymbol{\alpha}_n\big) = \mathit{eigenvectors}\left(\sum\right) \tag{2.8}$$

$$\boldsymbol{\lambda} = \big(\lambda_1, \lambda_2, \ldots, \lambda_n\big) = \mathit{eigenvalues}\left(\sum\right) \tag{2.9}$$

Where $\boldsymbol{X}$ is the sample vectors shaped $m \times n$, $\boldsymbol{Z}$ is the transformed vectors shaped $m \times k$, $k \leq n$ for m feature vectors, with n features and k retained principal components. $\sum$ is the covariance matrix of $\boldsymbol{X}$, and $\boldsymbol{\alpha}$ sorted so that for corresponding λ_i, $\lambda_i < \lambda_{i+1}$ This ensures that each principal component has the maximum variance, under the constraint that it is uncorrelated with earlier principal components, and each $\boldsymbol{\alpha}_i$ is orthogonal to each other [Jolliffe, 2002, p.2,5,6].

In equation 2.7 to 2.9, the same matrix $\boldsymbol{X}$ is used to define $\boldsymbol{\alpha}_i, i \in [1, n]$, and is then converted to the new base. It is also possible to use a sample to define the transformation, and then transform other data, although this cause a greater loss of information since the sample may not represent the variance of the population accurately.

2.6.3.2 Univariate feature selection

Univariate feature selection is a simple method of selecting the best features, by some scoring metric, from a candidate set of features. It takes one feature at a time and calculates a statistical score that measures how well the feature is predicted to be able to produce good classification results [Saeys et al., 2007]. A number of different scoring functions are common, including, but not limited to *Chi-squared*, *Analysis of Variance (ANOVA) F-value* and *Pearson Correlation*. Of these, only the ANOVA will be presented in this section.

ANOVA F-value This is a scoring function that takes a set of normally distributed populations and calculates a score based on how likely it is that they are distinct groups. It does this by comparing how much variation there is between the groups with how much variance each group contains. This means that the farther the groups are from each other, the more each group can vary without intersecting with each other. The different groups can be created by their expected output and then serve as a measure of how well it is possible to divide the data into the decided groups [Johnson and Synovec, 2002, p.229]. The formula for calculating the ANOVA F-value can be seen in equation 2.10, where n_i is the number of samples in group i, K

and N are the number of groups and samples respectively, $\overline{Y}$ and $\overline{Y}_i$ are the mean of all samples and group i respectively, and $Y_{i,j}$ is the value of sample j in group i.

$$F = \frac{\text{between groups variability}}{\text{within groups variability}} = \frac{\sum\limits_{i} n_i \left(\overline{Y}_i - \overline{Y}\right)^2 / (K-1)}{\sum\limits_{i,j} \left(Y_{i,j} - \overline{Y}_i\right)^2 / (N-K)} \tag{2.10}$$

Although the scoring function is made to work on normally distributed data, large data sets can still be used reliably due to the central limit theorem. How large a data set needs to be for this to hold depends on how nonnormal the data is [Miller Jr, 1997].

2.6.4 F1-score

F1-score is a common classification scoring function for machine learning. A classification scoring function takes the result and ground truth of a classified test set and returns a score signifying how well the data has been classified. It is calculated as seen in equation 2.11 [Yang and Liu, 1999, p.43].

$$F1 = 2 \cdot \frac{precision \cdot recall}{precision + recall} \tag{2.11}$$

Precision and recall are concepts that measure different ways of how well a class C has been classified. Precision concerns the portion of true positives among the set of samples classified to the class C by the algorithm. In practice this is means the number of correctly classified samples of class C divided by the number of samples classified to class C, see equation 2.12 [Yang and Liu, 1999, p.43]. Recall concerns the portion of the samples belonging to class C that were classified correctly, calculated as the number of true positives divided by the number of samples belonging to class C, see equation 2.13.

$$precision = \frac{\text{true positives}}{\text{positives}} \tag{2.12}$$

$$recall = \frac{\text{true positives}}{\text{true positives} + \text{false negatives}} \tag{2.13}$$

2.7 Scikit-learn

Scikit-learn is a machine learning library in python and is based on numpy and scipy [Pedregosa et al., 2011, p.2826]. It provides efficient algorithms for many different machine learning disciplines, for example classification, regression, and clustering, as well as other tools for machine learning.

Chapter 3

Method

This chapter will go through what was done during the study, as well as how it was done. First the process of the study, and how the results were achieved will be presented, closely followed with how the applications used in the study were built, and how the user study was planned.

3.1 Process

This section describes what was done, and motivates some choices regarding the process. First, the recording session is described, then two iterations of model creation.

3.1.1 Recording

First, a quiz application was developed, which let a user answer a number of pre-set general knowledge questions while recording eye behaviour, as well as mouse clicks. This application was created to be able to record eye data for learning and is described in detail in section 3.2.

Then, recording sessions were run with the participants, to record their eye movements, and clicks while using the application to get data to learn from. This is described in better detail in section 3.5. The pilot sessions that were done went well, and the data from them were included with the rest of the data as the main data set.

The study was started with the recording session to be able to develop the learning applications to match what was found among the actual data instead of predicting how the data would be found.

3.1.2 First model iteration

After the recording sessions had concluded, a learning program was developed that could take data from a recording session and produce a model of the users' eye behaviour and clicking. This model could then be used to predict clicks from eye data. This application is more thoroughly described in section 3.3.

After the model learning program was developed, and the scores of models created from test data were deemed sufficient, models were created for all participants using both PCA and univariate feature selection. Then, the quiz-application was rewritten to be able to send gathered data to a classification back end and accept click results from it. The classification back end was developed to take a model, and provide click predictions to the quiz-application after receiving eye data. It is described in section 3.4.

Interaction tests were done using models created for development testing to ensure that the system worked as intended. This data was recorded by me, without the earlier strict requirements on the environment.

3.1.3 Second model iteration

It was decided that learning should be tried on partial dwells as well as on the completed dwells and that grouping of fixation from interaction data should be moved to the learning application to enable grouping of incomplete fixations in the same way as competed ones from training data. This prompted a second iteration of model creation as these changes were done.

With these changes implemented, models were created using development testing data, and used to evaluate how well interaction worked. The models were created from the participant data.

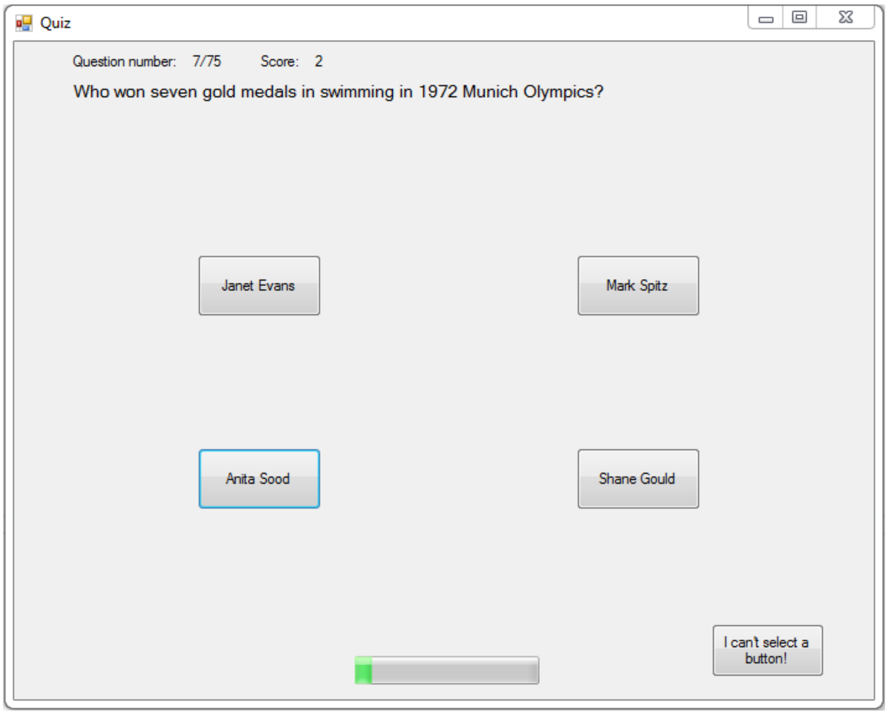
Later, additional models were created from development testing data to be able to provide statistical results regarding models created from development testing data.

3.2 Quiz Application

The application used during the user studies was a quiz application where the user was asked to answer 75 questions about various topics. It also gathered statistics about the questions, such as number of answered questions, and number of correctly answered questions as well as user eye movement data. The application could be run in 2 different modes, recording mode, and gaze-interaction mode.

3.2.1 Graphical User Interface

The primary Graphical User Interface (GUI) can be seen in figure 3.1. Each answer button was surrounded by a AOI's where eye data would be registered if the user's gaze fell within. The AOI's were not visible in the application, but can be seen in figure 3.3. After each answer, the application asked the user if the selected answer was intended to be selected or not, see figure 3.2. This was done so that the user could signal to the application



Question number: 7/75 Score: 2

Who won seven gold medals in swimming in 1972 Munich Olympics?

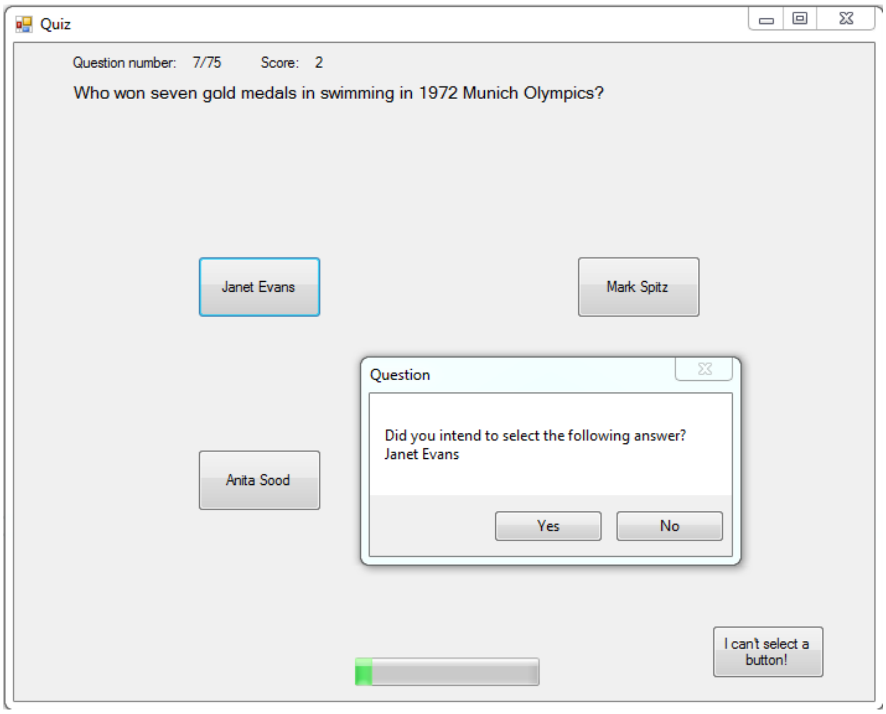
Figure 3.1: Quiz application. ©Albin Stenström

that unintended selection (midas touch) occurred, and that the selected answer was not intended. Similarly, the user could use a button to signal to the application that they could not select an answer using gaze. Both of these were important to get statistics of the success of the gaze interaction. The application also started eye-tracker calibration, and calibration testing before the task was started and calibration testing again after the task was done. Both calibration and calibration testing were done using the tools provided by Tobii in the EyeX SDK.

3.2.2 Inner Workings

The output of from the application consisted of question statistics, processed eye data and raw eye data, all to different files. The raw eye data had minimal processing, except grouping of fixation data parts into a single fixation event per fixation, more closely described later. The fixation data parts were also kept, to enable later analysis. Additionally the eye data files contained click events, AOI leave or enter events and click events. Processing of the processed data was done by filters that both filtered away data from outside the AOI's, converted coordinates from global coordinates to coordinates relative to the buttons, grouping fixation data and similar.

In recording mode, the application stored the data for later analysis and in interaction mode, the data was sent to the classification application for classification as click or not click. The answer sent from the classification application could then be used to activate the corresponding button if ap-



Quiz

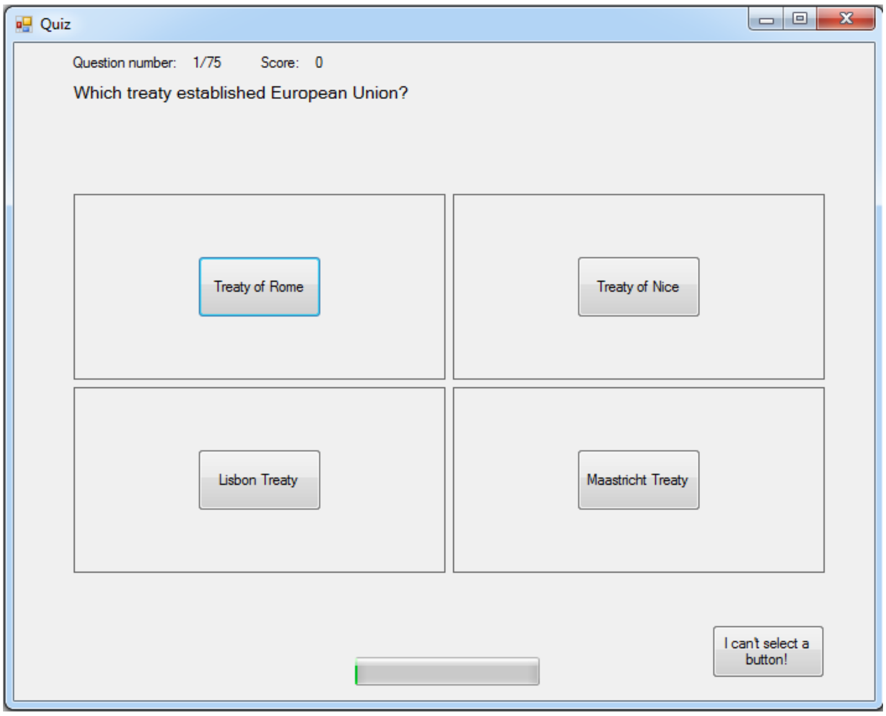
Question number: 7/75 Score: 2

Who won seven gold medals in swimming in 1972 Munich Olympics?

Question

Did you intend to select the following answer?
Janet Evans

Figure 3.2: Quiz application with confirmation window. ©Albin Stenström



Quiz

Question number: 1/75 Score: 0

Which treaty established European Union?

Figure 3.3: Quiz application with visible AOI's. ©Albin Stenström

plicable.

3.2.2.1 Filters

This section describes the different filters that were used to filter and transform the data. How the filters were used can be seen in figure 3.4

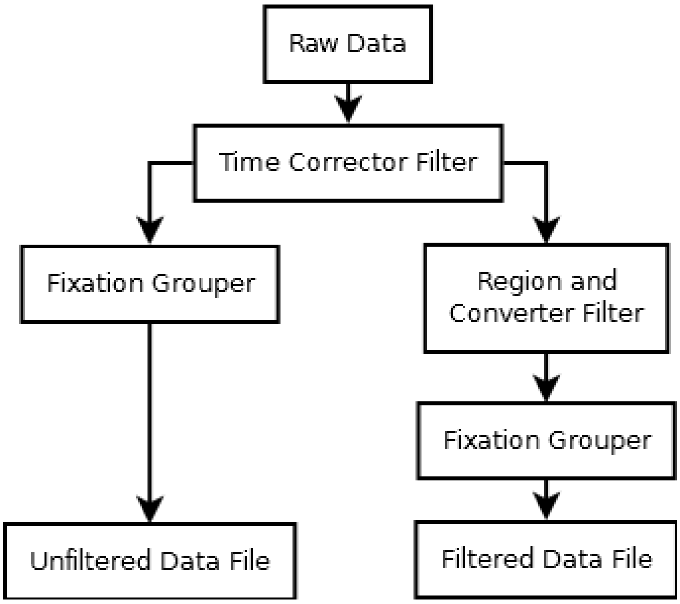


Figure 3.4: Visualization of the filters in the Quiz Application

Time corrector filter was a filter that ensured that entries that did not have a reported time was assigned a timestamp corresponding to its surroundings. An AOI entry event for example had no inherit timestamp and was assigned the time of the next entry in the stream as an approximation of when it began.

Region and conversion filter The region and conversion filter filtered away all data except clicks, while the gaze fell outside of the AOI's. This was done by looking at the entries corresponding to gaze entry and exit of AOI's and start and end for showing the confirmation window. Additionally, the coordinates of the entries going through the filter would be converted from screen coordinates to application coordinates relative the middle of the nearest button.

Fixation grouper filter This filter let all data through but in addition created fixation entries from the partial fixation entries that were generated by the SDK. The fixation was timestamped as the first part, and the duration was considered to be the difference between the first and last part timestamps. Additionally, the location was approximated by averaging the locations of the parts, and a variance in each dimension were also calculated. Care was also taken to ensure that in case that a start or end part was lost, the fixation data would still be included as one fixation entry.

3.2.2.2 Data Format

The data format was output in text format, where each line represented one data point, begin entry or end entry. The lines started with a tag, and a timestamp, followed by a comma separated list of value tags and values. The values of data points consisted of mostly coordinates, but also variances of coordinates and durations. The start and end entries had a association to a button. Example data from one dwell session is available in appendix D.

3.3 Learning Program

The learning program took data gathered during the recording session, analysed it and created a machine learning model that could predict click or no click from eye movement data. The program went through four steps, preprocessing, feature calculation, feature selection/extraction, and learning. Each of these steps will be explained below.

3.3.1 Preprocessing

The first two steps consisted of importing the data, and preprocessing it. The preprocessing step first partitioned the data into entries or sessions for each dwell session on a AOI, and marked if a click was performed during the dwell session. This was done by inspecting the data for entries for entering or leaving an AOI, and click data.

Entries were then merged into a single entry if they were within $400ms$ of each other to make sure that data noise, interference or involuntary eye movements could not end a dwell by moving outside the AOI for a short period of time. The specific value was chosen to be the maximum mean blink length according to Harvard [2013]. Additionally, entries were removed if they were shorter than $200ms$ to ensure that the data for each dwell had enough data to be relevant. This value was set because it provided a balance between the number of retained entries, and classification success. Increasing the value removed a lot of entries, while decreasing it lowered the classification accuracy a lot.

The changes introduced in the second iterations (see 3.1.3) were placed at the beginning, and at the end of the preprocessing step. Before everything else, the fixation data was grouped together into fixations, similarly to how it was done in 3.2.2. The fixations from the data file were discarded. At the end of the preprocessing step, each dwell-session was copied recursively, and from each copy, $200ms$ was cut away, at the end. Each new entry was marked as not a click. This was done to mimic the behaviour of the classification program where partial dwells are classified and should not be classified as a click before the characteristics match whole dwells.

3.3.2 Feature Calculation

The second step consisted of calculating features for each entry. The following features were calculated:

- Dwell time
- Mean fixation duration
- Standard deviation of fixation duration
- Total fixation duration
- Fixation rate
- Mean location of fixations
- Standard deviation of the fixation locations
- Euclidean distance to the mean location of fixations
- Direction of mean location of fixations
- Mean euclidean distance of fixation locations
- Standard deviation of euclidean distance of fixation locations
- Standard deviation of intra fixation locations of the last fixation
- Duration of last fixation
- Standard deviation of intra fixation locations of the first fixation
- Duration of first fixation
- Mean of gaze location
- Standard deviation of gaze location
- Mean of euclidean distance of gaze location
- Standard deviation of euclidean distance of gaze location
- Standard deviation of left eye position
- Standard deviation of right eye position
- Mean blink length
- Standard deviation of blink length

- Blink count
- Mean blink rate
- Mean saccade distance
- Standard deviation of saccade distances
- Mean duration of saccade
- Standard devation of saccade durations
- Mean of saccade average speed
- Standard deviation of saccade average speed

Blinks were approximated by data loss for longer than $96ms$ as described in section 2.2.2.7, and saccades were approximated as the movement between each pair of fixations as seen in 2.2.2.5.

A try was done to calculate features for time spans during a dwell, as done by Kandemir and Kaski [2012] as explained in 2.3.2, but it had a negative impact, on the classification results, which about halved, and were therefore not used in the final application.

The ambition was to calculate a greater number of features for a variation of measures and then use feature selection to find the features that were relevant, instead of not calculating a feature that might be relevant. Motivations for the used features will be presented below.

Dwell time is a measure that is used as a primary measure for interaction in many systems [Majaranta and Bulling, 2014, p.49], and is therefore a great candidate as a important feature.

Fixations as explained in 2.2.2.3 is a popular eye-tracking measure that measures how long the user focuses on one point, which could be important to recognize for example if the user is reading on the button or not. Additionally, Bonifacci et al. [2008] used fixation duration in their similar study with great results. Various versions of the location measures are tried to capture different characteristics, generally as mean and standard deviation. Using distance measures is a try to make each sample equally important. The first and last fixation have their own measurements as discussed in 2.2.2.3.

Gaze location was used for the same reasons as the fixation locations above, but does this for the complete dwell session to create a more complete picture of the movement.

Eye position measures the location of the eyes relative to the screen. Since the location should not matter for interaction, but head movement might, only standard deviation is used here to capture the movement during a dwell.

Blinks are another measure often used as a direct method for interaction, and was deemed likely that this would mean that it is significant for this study.

Saccades are another measure used by Bonifacci et al. [2008], and is additionally one of the more popular ones used in research. It measures the fast movements of the eyes between fixations. Peak saccade velocity is often used, but the eye tracker in this study is not fast enough to detect that, so the mean velocity is used instead. In addition to velocity, distance and duration are measured.

3.3.3 Dimension Reduction

Dimension reduction was done by passing the feature matrix through a PCA implementation and a univariate feature selection (kbest) implementation one at a time to be able to evaluate the results from using them separately. Both these implementations were provided by scikit-learn, and had a tunable parameter that decided the number of retained features. The strategy for dimension reduction was later decided on user basis based on the results of the two strategies for that specific user.

To make this possible, missing or invalid feature values were first replaced with the mean of the other instances of the same feature. Additionally, features with no variance were removed, and the data was scaled to have a zero mean, and unit variance. This was done to make sure that all features had values for all instances, and to make sure that no feature was favoured because of a greater/lesser range of values, or higher/lower mean.

3.3.4 Model creation

The algorithm used to create the estimator was scikit-learn's SVC, a SVM algorithm, and the kernel was chosen as a *gaussian RBF kernel*. A pipeline was created that contained all the components from the dimension reduction (see 3.3.3) and the estimator itself. The pipeline could then be used to transform the data according to 3.3.3 and train or classify the estimator. The accuracy was then estimated by using a 5-fold cross validation. A mean f1-score for the classes, and the fraction of correctly classified samples were calculated, as well as the fraction of correctly classified clicks, respectively no clicks. The pipeline was then saved to file using the python module pip.

SVM was chosen because of its prominent presence in literature about machine learning together with eye tracking, for example [Kandemir and Kaski, 2012]. It is in additionally an algorithms where specific domain knowledge is not needed to be successful [Russell and Norvig, 2014, p.755]. A RBF kernel was chosen because of its ability to represent a feature space of endless dimensions (see 2.6.2.1) giving, more expressive power, and because it was the kernel that by wide margin gave the best classification results, compared to linear and polynomial kernels. 5-fold cross validation was used to reduce the risk of overfitting.

3.3.5 Parameter Search

A parameter search could also be done using the learning program, to find the parameters that give the best results. This was done by calculating a score for permutations of a set of parameter values and then selecting the one that gave the highest score. The score used was an average of the f1 score of the two classes click and no click. The parameters that was searched were the target dimension of the dimension reduction, penalty parameter (c), gamma and weight of clicks for SVC, where the weight of clicks set how much incorrectly classified clicks samples should be penalised.

A dedicated test set was created when doing parameter search to verify that the models generalized well to new data.

3.4 Classification Program

The classification program took the classifier created by the learning program (see 3.3) and acted as a backend to the quiz application, classifying data as a click or not.

The program used a ZeroMQ subscriber socket to get data from the quiz application, and periodically computed features for the collected data and sent it to the estimator for classification. The estimator was imported from the file created by the learning program (see 3.3.4). The calculation of features including preprocessing was done the same way as in the learning program see 3.3.1 and 3.3.2. The only exception was that the classification program did some processing to create a simulated end of the current dwell to make the data similar to the training data where all dwells had complete data. Duration of the dwell so far was for example needed.

The estimated classification result was then sent to the quiz application through a ZeroMQ publisher socket.

3.5 User Studies

User studies were used both to gather eye tracker data for the tracker learn from and to evaluate the results of the study. The task consisted of using the quiz application presented in 3.2. During the data recording session, the participants were asked to use the mouse to click on their expected answer, and the eye tracker tracked the eye movements near the buttons as they did so.

3.5.1 Environment

The study environment consisted of a conference room for approximately 10 people. The windows faced south west and had the window blinds down and curtains closed to limit the amount of sunlight in the room. The participants sat in the middle of the room, facing the windows, with the monitor in front of them, with a table to the right (or left if needed) where the mouse was placed. A gap between the two tables made sure that no mouse movement could propagate to the monitor and eye tracker. Additionally, the floor was concrete, hindering movement through it. The back of the chair was placed 77 cm from the monitor which stood at the edge of the table. A chair was placed on the opposite side of the table with the mouse, for the session leader to sit on to be able to instruct and observe the participant. This was done to fulfil the considerations from the literature as far as possible, see 2.5.1 for specifics.

The room was not sound proof, and some sounds from the surrounding office as well as the parking lot outside were therefore able to enter the room during the sessions. Neither was the blinds and curtains flawless, and some light albeit limited was able to enter the room. Except for some small light rays, dampened by curtains, no direct sunlight entered the room. Care was taken to ensure that the light rays were not able to shine directly at the participant.

3.5.2 Data Recording

During the data recording sessions, the participants were told to use the application as if they did not know their eyes were tracked, except during calibration. Although 2.5.2 specifies that the participants should not know what the expected result of the recording are before participating, in the corresponding real world situation, a user would know why data was recorded, and as such, this was ignored. All actions in the application were done with a mouse. For more information about the application, please see 3.2. The written instructions given to the participants can bee seen in appendix A.1.

The participants were given time to explore the application prior to the session to ensure that unfamiliarity with the application did not cause any problems.

After the task was done, the participants were handed a questionnaire, to quantify the experience of using the application. See 3.5.4 for more information.

3.5.3 Pilot sessions

The data recording had an initial pilot of three participants after which the gathered data, the participants impressions and observed participant behaviour were evaluated to ensure good quality. During the data recording pilot, no issues were found, and no changes were done for later sessions. The data gathered was therefore not discarded, but added to the main study data.

3.5.4 Questionnaires

The questionnaires used after the sessions were created for two purposes. The first was to gather information about the participants regarding age, gender and sight corrections. The second was to provide insight into how the study set-up could affect the results during the pilots.

The questionnaires were done as a five step Likert questionnaire, see 2.5.4, except initial participant information. The questionnaire for the recording session is available in appendix B.1.

The standard questions was edited to use "application" instead of "system" to match the situation as argued by Bangor et al. [2008, p.576], to increase understanding of the statements.

3.5.5 Participants

In total, 10 persons were recruited as participants, where 3 participated in the pilots, and 7 participated in the main user study. The number of participants was chosen to allow for loss of some participants due to bad calibration and other issues, and still provide reliable results. The participants were recruited based on availability in the area and that they had good computer skills. Although the recommendation is to not use participants with sight corrections, see 2.5.2, it was decided to not exclude users with sight corrections to make the study apply to a more general case.

The participants in the pilot were all male, between 23 and 26 years old, with an mean of 24.7 and of standard deviation of 1.3. Two of them used glasses, one did not, and no one used contact lenses. The participants of the main user study were between 24 and 57 years old, with an mean of 37.1 and a standard deviation of 12.2, and one was female. Two of the participants used contact lenses, four used glasses and one had no sight correction. Across both groups, the average age was 33.5 years old with a standard deviation of 11.7 and a median of 27. The complete data is available in table 3.1. Data recordings were also done on myself for development testing purposes and my data will therefore be shown as participant 11.

Table 3.1: The participants of the user study. Participant 1-3 participated in the pilot and participant 4-10 participated in the main study. Participant 11 was only used for development testing.

Person	Age	Gender	Sight Corrections
1	39	male	glasses
2	23	male	glasses
3	26	male	none
Mean	24.67	-	-
St.Dev	1.25	-	-
4	25	male	glasses
5	36	male	contacts
6	24	female	contacts
7	24	male	none
8	28	male	glasses
9	52	male	glasses
10	57	male	glasses
Mean	37.14	-	-
St.Dev	12.24	-	-
Mean	33.5	-	-
St.Dev	11.75	-	-
11	26	male	none

Chapter 4

Results

This chapter will present the results of the study, starting with the results from the recording session, and continuing with the results of the model creation.

4.1 Recording session

No study-specific issues were found during the pilot or in the main study, and therefore, the pilot was deemed to be part of the main study. However a few complications regarding specific recordings were observed, and will be presented here.

Calibration accuracy Some participants had some trouble to get a good calibration even after multiple calibrations, and were told to use the slightly imprecise calibrations. This was true for participant 2 and 6, but in particular with participant number 10. Participant 10 was unable to get an even remotely accurate calibration before changing glasses. Calibrations using the new glasses were exact, but depended heavily on the angle of the head.

Calibration drift Two of the participants, namely 7 and 10 had a drift of accuracy when testing the calibration after the task was done. Participant 7 had some small drift in the middle of the screen. Participant 10 on the other hand had a quite big drift, probably related to the calibration problems.

Participant height Participant 8 was too tall for the standard configuration, putting his eyes outside of the tracking box of the eye tracker. Since hunching would have caused neck strain, the monitor was angled upwards, deviating from the standard configuration.

Miscellaneous observations Participant 4 and 5 blinked quite a lot more than the other participants, probably due to tiredness. Participant 6 was using mascara on her eyelashes which could have disturbed the recording. It was generally observed from looking at the participants eyes that many of the dwells were very short, likely around 0.5 second.

4.1.1 Questionnaire

Table 4.1 contains the statistics from the questionnaire given to the participants at the end of the recording sessions.

Table 4.1: Statistics from questionnaires. Answers marked as no opinion are not counted.

Questionnaire Statistics			
Question	Mean	Median	St.Dev
I thought the application was easy to use.	4.8	5	0.4
The questions were easy.	2.8	3	0.87
I needed the alternatives to answer most questions.	3.9	4	0.83
It was tiring to use the application.	2.5	3	0.67
The time of the task was too long for me to keep my concentration.	2	2	0.89

4.2 First iteration models

The models created during the first iteration were scored using training data and a randomized 5-fold. Feature selection for the selected models were done by univariate feature selection, since they clearly outperformed (5-20%) the models created when using PCA. For a description of the two iterations, please see 3.1.

4.2.1 Participant models

Table 4.2 provides statistics of model scores from the first iterations. The complete data is available in appendix C.1. The combined result of the models created from participant data was that 93% of the samples were classified correctly on average, with a standard deviation of 3.6%. Samples that should be classified as no click were classified correctly 97% of the time on average with standard deviation 2.1%. On the other hand, samples that should have been classified as click were classified correctly only 65% of the time on average, with standard deviation 17.4%. The variation was greatly affected by an outlier on 28%, but removing the outlier only changes the mean and standard deviation to 69% and 13.2% respectively. The highest percent of correctly classified clicks were 91%.

Table 4.2: Score statistics on the models for the participants during the first model creation iteration. The first column was calculated using a mean of 5-fold scores on training data.

	All samples	Click samples	No click samples
Mean	92.9%	64.6%	96.8%
St.Dev	3.6%	17.4%	2.1%

4.2.2 Test model

Table 4.3 provides the scores of the test model for the first iteration. The model was created from data recorded on myself for development tests and was nearly as good as the best model created for the study participants. The scores were calculated calculated as a mean of 5-fold scores on training data. The model was created using univariate feature selection.

Table 4.3: Model scores for model built on data for developer tests using univariate feature selection. Calculated by average on 5-fold of training data.

All Samples	Click samples	No click samples
96.9%	86.1%	98.9%

Tries to interact using this model were not very successful. Midas touch was a big problem and especially blinks triggered unintended clicks even though features for blinks registered no blinks at all for clicks among the training data.

It should be noted that this model was also scored against a test set from another recording of mine, with comparable results.

4.3 Second iteration models

This section presents the results of the models created during the second model creation iteration. Contrary to the first iteration, the models during the second iteration were scored using a dedicated test set. Additionally, neither PCA nor univariate features selection were clearly the best during this iteration, so both will be presented here. For a description of the two iterations, please see section 3.1.

4.3.1 Participant models

Table 4.4 provides statistics on the best models of each participants, and table 4.5 provides statistics on models created both with PCA and univariate feature selection. The complete data is available in appendix C.2.

Table 4.4: Model score statistics of the best model for each participant. Each score was calculated using a dedicated test set. Complete data available in table C.3 in appendix C.2.

	All samples	Click samples	No click samples
Mean	85.4%	15.4%	88.1%
St.Dev	10.2%	8%	10.5%

Table 4.5: Model score statistics for the models created during the second model creation iteration. Each score was calculated using a dedicated test set. Complete data available in table C.2 in appendix C.2.

Features	Univariate			PCA		
Samples	All	Click	No click	All	Click	No click
Mean	89.9%	7.7%	93.2%	86%	16.4%	88.7%
St.Dev	8.2%	9.7%	8.9%	9.5%	14.3%	10%

The scores of the models created from the participants data were much worse than during the first iteration, even including a few zero percent scores where not a single click was correctly classified. When for each participant, the best model is selected, on average, only 15.4% of the ground truth click samples were classified correctly, with a standard deviation of 8%. A decrease of 49.2 units compared to the first iteration. Ground truth no click samples were correctly classified 88.1% of the time with standard deviation 10.5%, signifying a decrease with 8.7 units. Globally, 85.4% of the samples were classified correctly and the standard deviation was 10.2%. A decrease of 7.5 units.

4.3.2 Test models

For the second iteration, models were built from four recordings of myself. For each recording, three models using PCA and three using univariate feature selection were built.

Table 4.6 provides statistics on model scores from models built from four recordings of myself. For each recording, three models were built and scored for PCA and Univariate Feature Selection each. The complete data is available in appendix C.2.

For the models created using univariate feature selection, on average 84.5% of the samples were correctly classified, with a standard deviation

Table 4.6: Statistics of model scores from data recorded for development testing purposes. Complete data available in table C.4 in appendix C.2.

Features	Univariate			PCA		
Samples	All	Click	No click	All	Click	No click
Mean	84.5%	43.7%	87.4%	79.7%	54.1%	81.3%
St.Dev	7.4%	21.8%	8.4%	11.6%	23.4%	13%

of 7.4%. Ground truth click and no click samples were classified correctly on average 43.7% and 87.4% of the time respectively, and their respective standard deviations were 21.8% and 8.4%.

The models created using PCA were classified correctly on average 79.9% of the time with standard deviation 7.4%. The corresponding values for the samples containing clicks were 54.1% and 23.4% and for samples containing no clicks 81.3% and 13%.

These models, created to provide a more statistical base of that the models created from my data give better scores, and works better for interaction did in fact not work well for interaction. A four of the best scored models were tested, and none of them even classified a single instance as a click. The data is still presented because it can be reasoned about differences compared to the first iteration, and why my personal data creates models with better score.

One model was found among data from development testing that worked fairly well. No scores were saved for the model, but memory provides that scores for both click and no click were above 80% on data set 3, a slightly better model in that regard. This model was, according to stored statistics, during a session able to classify clicks correctly 80% of the time, unintended clicks were recorded 17.3% of the time, and failure to select a button was recorded 26.7% of the time.

The impressions of using this model for interaction were that one had to take care to not focus on a location for too long or too intently, or an unintended click might be performed. Additionally, it put some strain on the eyes. However, it also felt powerful, neat and effective to be able to select the answers using only the eyes.

Chapter 5

Discussion

The goal of this thesis was to evaluate if applying machine learning to eye tracking is a plausible way to enable interaction using natural eye movements. Although it clearly is possible, this thesis cannot provide reliable methods, or statistics to support the fact that for a general situation, which raises quite a lot of questions. Interaction using models created by machine learning is possible, but the specific situations and requirements are unclear. This chapter will discuss a few such potential factors, and in addition address a few other questions that the results raised.

One thing that needs to be addressed first is the unfortunate case that no complete score is available for the working model. The oversight of not recording the scores and the assumption that the apparent reproducibility would extend to later similar recorded program versions seriously cripples the ability to make accurate speculations of Although there are three categories: participants models, my working development models and my not working models, the latter two can in this case be viewed as one since the same data was used for both. This section will discuss these differences and why they occurred. suitability of the approach. It is additionally harder to speculate about the specifics about the working version of the program since the specific version of the program was not documented due to the earlier mentioned assumption.

5.1 Questionnaire

This section will discuss a few of the results from the questionnaire done at the end of the recording session.

The questions were easy, and *I needed the alternatives to answer most questions*, with mean 2.8 and 3.9, suggests that the questions were difficult, but not too difficult. Additionally it means that the participants often looked at multiple alternatives to answer a question and therefore provide more data points than if the answer was trivial.

The time of the task was too long for me to keep my concentration had a mean of 2, suggesting that though the amount of question did not tire the participants too much. However, *It was tiring to use the application* getting an average of 2.5 and median 3 could be an indication that increasing the number of questions could in fact make the participants loose concentration.

I thought the application was easy to use with a mean of 4.8 and median 5 suggests that the application in itself was not an obstacle during the sessions. This reduces the probability that changing the GUI would improve results by very much.

These factors suggests that the specific task and application are not big contributors of errors and uncertainty to the study.

5.2 Different recordings

It is clear upon examination of the statistics in section 4.3 that the models created by my own data recordings, for some reason are quite different from the models created from the participants' data. This suggests a significant difference between these data sets.

5.2.1 Different recording programs

Although the same program has been used for all recordings, certain modifications have been made to the program during the time of the participants recordings and the last of my recordings, no changes were done to how data was recorded.

The only changes were to add the possibility for the the classification back end to classify the data that was gathered. However, these changes could have had an unpredicted impact on the recorded data. Additionally, the first two of my data recordings were done with exactly the same binary as the one used during the participants recordings, and while data set one scored worse than the other data sets, data set two scored on par with the others. In addition, review of the code changes done provides no hits that it would change anything.

Although this is still a possible source of the recording differences, it is not very probable due to the aforementioned points.

5.2.2 Environment

The recordings of the participants were done using the strict environment set-up and instructions described in section 3.5. Recordings of my own eyes however were not done in the same way.

These recordings were done at the workplace, in a small office room, with two windows to the left. The windows had blinds down, and curtains placed to block more sunlight. On the other side, glass doors and walls permitted light from the rest of the office to enter. Seating was provided by a rolling chair providing no fixed distance to the screen, and a shaky table on which both mouse and monitor were placed.

Contrary to what one might expect, this situation did not give bad results, but good ones. The expected result of reducing the variations of environment would be to get better results, see 2.5.1. This makes this difference a less likely source of the difference in data. However, the strict instructions of the user study might have caused the participants to tense up and be overly concious of the eye-tracking, loosing natural behaviour and in turn lowering the quality and relevance of the data. The relaxed requirement of the office recording on the other hand may have given more natural behaviour and more relevant results. The environment could potentially be a factor, but since the theory mostly talks about reducing variation, it seems unlikely.

5.2.3 Participants

A more promising factor is the person the recording in done on. The participants of the study had some limited time to get to know the application before the recording, and were not hugely familiar with the application when the recording started. The questions were new to them, and the underlying mechanics were hidden from them.

The participant in the development recordings on the other hand, me, was intimately familiar with the application, both working with it, and the underlying code. The questions were selected by me, and debugging and testing had enabled me learn some questions, and recognize others by answering them multiple times. The exception was the last recording, where I had chosen the questions months earlier, and had not looked at them since. Looking at the models created from the development data recordings, this seems to fit quite well. During the first recording, I was fairly new to the questions, and had not done a recording before myself. This might have reduced the quality of the recording. Recording 2 and 3 got better and better, until recording 4 where new questions were used, reducing the quality.

This still fails to explain the full difference compared to the participant models, but it is something. The missing piece could be that because of my knowledge about the program internals, and eye-tracking I unconsciously adapt to a behaviour that is more likely to succeed.

This seems to be the most probable and important reason so far to why these differences occurred, but some other aspects should also be mentioned. Eight of the ten participants used sight corrections, and I do not. This does not account for the fact that the two remaining participants still got bad models, but as discussed in 5.4, both theory and results suggests that sight corrections makes it significantly harder to classify the samples. On the other hand, this makes it unlikely that this is the only factor of producing differences, but might be a important one.

What is clear is that the individuals matter greatly, there are differences that make different approaches work well. It is this property, and problem that this thesis was aimed to take advantage of, and solve, but although the technique has been shown to work, the problem still persists.

5.3 Implementation details

The specific implementation of programs is important, in many cases, for the efficiency and success of programs. In the case of this study, it has been critical. Very slight changes to how the data is handled, even if it should only matter on decimals, can change the results greatly.

In particular a small clean-up of the code caused the programs, capable of creating a working interaction model, to fail. Small as they were, finding the exact working code was unsuccessful.

These factors suggest that this problem is an especially sensitive one, and might take great care and precision to solve successfully. Although the programs have worked for interaction, and now do not, a discussion of the different ways eye data was handled will follow, to consider how an application should best be built to succeed.

5.3.1 Continuous versus dwell session classification

This study is conducted on the premise that dividing the eye-tracker data into one entry per dwell session would provide a good model for click behaviour. It was done that way, partly because it is similar to how Kandemir and Kaski [2012] classified whole dwell sessions. On the other hand, they did continuously try to classify the dwell based on the past three seconds.

A dwell session is a well defined concept, a good delimiter of data, easy to calculate features for and easy to associate to a button. However, one weakness is that a dwell session by nature is associated with an AOI, that need to be defined. In this thesis, the AOI's were defined as quite big rectangles around the buttons, bulky through an GUI perspective, but important data still might be outside of the scope of the AOI. This means that it might be hard to design good GUIs that work well with this approach. This is true even if its size increases. This fact combined with the fact that that dwell sessions sometimes were very short during the recording sessions, as described in 4.1, could be a problem due to a small amount of recorded data per dwell.

A more continuous approach, where classification is run frequently on data gathered during a few past seconds would be a far more general approach taking care of the problem of lost data between AOI's. However, this also adds new challenges. Where the approach of this study could associate one classification result with one specific button, the continuous one cannot. Instead it needs to find patterns that associates a click to a time, location, or both. This is a great challenge, but could contribute greatly, mainly because of its generality.

5.3.2 Merging and removing dwell sessions

The dwell entries in the data were, as were explained in 3.3.1, merged together provided that they were close enough and on the same button. This was done to mitigate the problem of data noise, short distractions or blinks splitting an otherwise cohesive dwell session. However, this could also have negative consequences such as that gap of no data for the period between the dwells sometimes causing big changes in feature values.

Additionally, very short dwell sessions were removed from the data with the motivation that short sessions contains so little data that many features are undefined, and hard to classify. This could also remove important data, such as sessions with clicks just slightly shorter than the threshold containing patterns matching clicks well.

Both these functionalities of processing have been tested for different values, and found to increase the classification rate. However, this does not prove that the functionalities increase the performance for interaction, due to for example overfitting problems. It is believed to be prudent measures to increase likelihood of success, but could also be something that has implications on the interaction performance even though scores increase.

5.3.3 Splitting dwell sessions

The most important difference between the two iterations of creating models is that each dwell session was split into multiple dwell-sessions as explained in 3.3.1. This was done so that the end of the dwell entry, that actually contains the click gets more important than the beginning, that does not. Although other changes were made as well, it is likely that this change is what made it possible to create models that worked well for interaction.

Even the models that did not end up as good models for interaction differed greatly compared to the models of the first iteration. From a severe case of midas touch, with many unintended clicks to not being able to click at all. For the study participants' data this change was even greater, from a classification score of 64.6% to 15.4% in the average case. This makes this change important even though it is not fully understood exactly why different data act so different from it. This could be interpreted as that the participants data did not have what is needed to classify clicks correctly, but it is likely that the specific implementation just worked well for some data or personal characteristics, and that another implementation could fit other data better, or most data. What is quite certain is that this change reduced the problem of midas touch, either by making no clicks at all, or actually doing a majority of clicks right, even though the latter is rare.

5.3.4 Algorithms

The choice of algorithms are of utmost importance and can impact performance greatly, both in terms of results and hardware requirements. This is especially true for machine learning, where different algorithms aimed at the same task might produce very different results, depending on the characteristics of the data.

SVMs were chosen because of mentions in eye tracking research papers, and especially the most similar study that was found [Kandemir and Kaski, 2012]. It is additionally a popular choice of algorithm for classification in general. However, a lot of algorithms are mentioned in research, and might give quite different results. Researching suitability of different classification algorithms for this task and similar tasks might be a good research topic for future studies.

PCA and univariate feature selection were chosen both because of availability and coverage in literature, but none of them could clearly be predicted to outperform the other, and both were used. This decision seems to have been a good one, since both have outperformed the other for different data and different situations. The working model was created using univariate feature selection, putting it slightly above PCA in results, but this is not thought to be the deciding factor.

5.4 Glasses

It was as discussed in section 3.5.5 decided to not exclude participants with sight corrections as is recommended. This was done with the motivation that this thesis tries to apply to a more general case where sight correction does not matter. Still, this may have resulted in decreased results for most models of the participants.

Although the models created in iteration 1 did not end up to be used as interaction tools, their scores can be used to evaluate the impact of glasses and contact lenses.

The only participants in the study who used no sight corrections were participant 3 and 7. As can be seen in table C.1, in appendix C.1, the models of these two are the best models that were created.

The two participants with contact lenses, 4 and 7, scored slightly lower than the lowest participant using no sight correction, and slightly higher to much higher than the participants using glasses.

It should be noted that participant 10 had the most problems with calibration and drift, and the observation was that the glasses were the cause of this. The problems also reflect in the score, 28.5% where the mean was 64.6%.

These results makes it highly probable, that only using participants without sight corrections probably would have given better results. At this stage, non generality would not have hurt in this regard since the study and better models scores would have provided better data for interaction evaluation. On the other hand, participants requiring no sight corrections might have been harder to recruit.

5.5 Mouse versus gaze

One question that has been present from the start of this study is if using a mouse has a too big of an impact on the movement of the eyes. This would mean that there would be a too big of a difference between eye data when using a mouse and eye data when not using a mouse to be able to detect common characteristics.

In the case of the first iteration, and the not working development testing models from the second iteration, this could possibly be part of the problem. However, since models that work reasonably well have been created, this does not make the task impossible, but possibly harder.

The problem to solving this revolves around how to gather data. Lets assume that classifying natural eye movement for selecting an item is possible to do perfectly, using data from pure natural eye movement. Mouse clicks interfere with eye behaviour, as does using a separate button and even speaking. The latter of the two are also used together with eye-trackers, and might interfere less than the mouse, but they do interfere. It might in this situation be impossible since we need the gaze interaction to be able to gather data to enable gaze interaction.

The picture is not as bleak as this, it does work using mouse clicks after all. However, this does prove the challenge of finding ways to record selection without interfering with the natural eye movement. Both using the keyboard, or voice commands might be better in this regard, but they do still interfere.

Additional research might be needed here, to determine the best way of recording selection, without disturbing natural eye movement.

One way that could prove successful is to start using for example keyboard press, to create a working model, and then generate new data using this model. It might be possible to iteratively refine the models to match natural eye movement using more forced models.

5.6 The eye-tracker

The tracker in this study was a low end tracker, aimed at interaction as a complement to other interaction means for the general commercial market. As such, it needs to be cheap, in materials, technology and features. Compared to high end research trackers, that cost many times more, the tracker might be simple in features, but does it's job well compared to the price.

Although aimed at interaction, the tracker was not made for this more advanced analysis of eye movement and characteristics, but more as a pointer that could be activated by other means. It might work well for it's purpose, but still be limiting to this study.

Using a high end eye-tracker, perhaps this thesis could have taken advantage of the higher frequency, precision and more advanced features. In particular, pupil dilation and blink detection could have been a great advantage. Pupil dilation, with it's estimation of mental workload could have contributed greatly, such as it did for Kandemir and Kaski [2012]. Blink detection from video analysis, as compared by inferring from missing eye data, as discussed in 2.2.2.7, could give much more exact and reliable blink features. This includes being able to detect blinks outside of AOI's. Since blinks is common as an interaction method this proves to be invaluable. Additionally saccade detection could become much more precise and contribute to better results by using a tracker with higher frequency. In particular, this would enable saccadic peak velocity detection instead of only average saccadic velocity, but it would also mean smaller error terms for all timing features.

The motivation for using the consumer market tracker was that the study should be done with the equipment used in the situations where the study result could best be used. This point still holds true, but it might have been better to test the plausibility of the approach on a high end research tracker in the first step, and later examine how this could be applied on a consumer market tracker if successful.

5.7 Source Criticism

The referenced material in this thesis consists of a range of research articles, books (research oriented and not) and web pages where the former have been preferred before the later.

The number of citations has been a big factor in choosing research articles, and books, as a measure of how well used and respected the books are. Russell and Norvig [2014] (third edition) is for example *"The 22nd most cited computer science publication on Citeseer"* [Russell and Norvig, 2013], and although the other materials do not have the same level of citations, most are still well cited publications.

The web pages used in this study are of two types. The most common one is content from the web pages of Tobii and Scikit-learn, and are primary sources about the devices and frameworks that are used in this study. They are not about any specific research, but are still important to be able to describe the systems that are used. The second one are independent web pages that contained information that was hard to find elsewhere, although greater effort was put to validate these sites, they are still somewhat shaky in strength.

Some comments on some specific sources will follow. Harvard [2013] is from a sub domain of Harvard University's web page, and Harvard's reputation gives credence to the content together with the fact that they provide a source reference (although not accessible). Cesar Souza [2010], is well written, but has no references, nor any official affiliation with an institution of some sort and must therefore be regarded as not as reliable as most other sources. The contained information however, was not possible to find anywhere else, and thus, the source is used, but should not be trusted completely.

To ensure that each web page is available in the version that was accessed during the study, each page has been saved using `https://archive.org/web/` on the data the page was accessed.

Chapter 6

Conclusions

The goal of this study was to evaluate the suitability of using machine learning to generate clicks based on data from eye-tracking. The study shows that this is certainly possible, although the cases where it is, are not fully understood. Because of this, the results of this study does not allow solid claims about the suitability of the approach beyond that it is possible.

One thing this study has made clear is that people really are different, since different people produce such different results. This is something this study was done to capitalize on, but further study is needed to realize a system that generalizes well to all of these differences.

This study included many participants using glasses to aim for a more general case, but in hindsight, only participants without sight corrections should have been used. This could have contributed to more reliable results.

6.1 Extensions

If more time and resources were available, the first priority would have been to find the configuration that works well for the development test data, to be able to provide more solid results.

Then focus would have been put to analyse the differences between the data of the participants, and the development testing data to be able to understand the differing results better.

6.2 Further research

Since this study raises quite a lot of questions, quite a lot of different directions of research are possible.

In general, this study has shown that the approach can give acceptable results, but research needs to be done regarding in which situations it works, and how to make it work for a wider range of cases. More specifically, the following areas and points could prove to be important for further research.

The characteristics, and features of a tracker that are needed to provide good reliable results could either be a research topic on its own, or at least one point that should be considered before making additional studies. It would probably be wise to start with a more advanced tracker, and then check what features really are required.

A continuous variant of classifying is a more general approach, and might be a harder nut to crack. However, it has its own advantages compared to the approach used in this study and is a strong candidate for success within this research area.

Another direction of research is to look at different learning, and dimension reduction algorithms to compare which would work better for this type of data. Some algorithms need more domain knowledge, but can in turn prove much more effective because of this knowledge.

One of the reasons behind doing this study was that explicit interaction causes eye strain (see 1.1). In case of more successful results, an important topic of study could be to compare eye strain between different ways of interacting.

Finally, an interesting topic is how to generate data for teaching an interaction model, without disturbing the natural behaviour of the eyes. In other words, without using a mouse or other invasive ways to interact.

Appendix A

User Instructions

This appendix presents the instructions given to the participants of the user study.

A.1 Recording Session

Please follow the following instructions thoroughly:

1. Read all the instructions
2. Sit comfortably in the chair, but do not move it.
3. Follow the instructions on the screen to calibrate the eye tracker.
4. Go through calibration test as per session leader instructions.
5. Wait for the session leader ask you to start.
6. Click on Start.
7. For each question:
   - (a) Read the question.
   - (b) Consider all answers.
   - (c) Click on the button containing your guessed answer.
   - (d) Confirm your selection by clicking Yes. Clicking No is not applicable during this session but is present for continuity's sake.
8. Click Ok.
9. Fill in the questionnaire.

Please try to keep the following in mind:

- Sit comfortably and relaxedly, but do not change your seating position too much. Feel free to move your head.
- Keep the mouse on the separate table, and do not touch the table with the monitor.
- If you use Glasses or Contact lenses, please use the same the next time.
- Try to keep your eyes on the monitor during the entire session.
- If something goes wrong, or uncertainty about the session occurs, please ask the session leader.

Appendix B

Questionnaire

This appendix presents the questionnaire that was answered by the participants during the recording session. It is available as appendix B.1.

B.1 Recording session

Name: ____

Age: ____

	Male	Female	
Gender	□	□	
	None	Glasses	Contact Lenses
Sight corrections	□	□	□

Please mark the degree that you agree with the following statements, where 5 stands for "I fully agree" and 1 stands "I fully disagree". Select N.O (No opinion) if you have no opinion regarding the statement.

Statement	**1**	**2**	**3**	**4**	**5**	**N.O**
I thought the application was easy to use.	□	□	□	□	□	□
The questions were easy.	□	□	□	□	□	□
I needed the alternatives to answer most questions.	□	□	□	□	□	□
It was tiring to use the application.	□	□	□	□	□	□
The time of the task was too long for me to keep my concentration.	□	□	□	□	□	□

Appendix C

Model scores

This appendix shows the scores calculated on the models created from the recording session data for each participant.

C.1 First iteration

Table C.1 shows the scores of the models of the first iteration. The scores were calculated from using a new random 5-fold of the training data and the models were created using univariate feature selection. Correct is the percent of samples that was correctly classified. Click and No click stands for the percent of ground truth click respectively no click samples that were correctly classified.

Table C.1: Participants model scores for the first model creation iteration, calculated on training data.

Participant	Correct	Click	No click
1	94.9%	70.4%	97%
2	91.1%	50%	97.6%
3	96.4%	77%	99%
4	95%	61.6%	98.3%
5	93.5%	76.4%	97.1%
6	95.4%	73.7%	99.1%
7	96.9%	91.5%	97.9%
8	87%	46.7%	92.8%
9	92.9%	70.5%	95.7%
10	86.1%	28.5%	93.3%
Mean	92.9%	64.6%	96.8%
St.Dev	3.6%	17.4%	2.1%

C.2 Second iteration

Table C.2 shows the scores of the models of the second iteration, both for univariate feature selection and PCA as approach for feature selection. Table C.3 shows the best model scores for each participant, regardless of feature selection approach. The criterion was the click score because it is where the biggest differences lie. It should be noted that univariate feature selection was selected for participant 2 because the score on the training data for the PCA based model gave less than half the value compared to the test set and was deemed a random artefact.

Table C.2: Participants model scores for the second model creation iteration, calculated on a dedicated test set.

Features	Univariate			PCA		
Participant	Correct	Click	No click	Correct	Click	No click
1	97.6%	0%	99.1%	96.2%	1%	97.5%
2	65.8%	28.6%	67%	68.5%	57.1%	68.8%
3	94.3%	0%	100%	93.2%	13.6%	98.1%
4	87.1%	0%	89.1%	84.8%	1%	86.5%
5	91.7%	0%	97.2%	73%	11.8%	76.7%
6	94.7%	0%	100%	73.9%	27.8%	76.5%
7	88.1%	8.7%	92.2%	87.7%	13%	91.5%
8	92.6%	8.3%	94.6%	95.7%	8.3%	97.8%
9	94.2%	6.3%	98.5%	95.1%	0%	99.7%
10	92.6%	25%	94.1%	92.1%	12.5%	93.9%
Mean	89.9%	7.7%	93.2%	86%	16.4%	88.7%
St.Dev	8.2%	9.7%	8.9%	9.5%	14.3%	10%

Table C.3: Best participant model scores, selected on highest classification of Click samples. Note, univariate feature selection was selected for participant 2, see above.

Participant	Correct	Click	No click	Features
1	96.2%	10%	97.5%	PCA
2	65.8%	28.6%	67%	Univariate
3	93.2%	13.6%	98.1%	PCA
4	84.8%	10%	86.5%	PCA
5	73%	11.8%	76.7%	PCA
6	73.9%	27.8%	76.5%	PCA
7	87.7%	13%	91.5%	PCA
8	92.6%	8.3%	94.6%	Univariate
9	94.2%	6.3%	98.5%	Univariate
10	92.6%	25%	94.1%	Univariate
Mean	85.4%	15.4%	88.1%	-
St.Dev	10.2%	8%	10.5%	-

Table C.4: Model scores of models created from data gathered for development testing and analysis. Mean and standard deviation shown for each session and feature selection approach.

Features	Univariate			PCA		
Data set	Correct	Click	No click	Correct	Click	No click
1	90.0%	20%	94.4%	75.4%	52%	76.8%
1	78.9%	40%	81.4%	90.7%	12%	95.7%
1	89.7%	8%	94.9%	90%	16%	94.7%
Mean	86.2%	22.7%	90.2%	85.3%	26.7%	89.1%
St.Dev	5.1%	13.2%	6.2%	7.1%	18%	8.6%
2	84.8%	39.1%	90.6%	60.3%	65.2%	59.7%
2	67.2%	60.9%	68%	56.4%	69.6%	54.7%
2	74.5%	52.2%	77.3%	69.1%	47.8%	71.8%
Mean	75.5%	50.7%	78.6%	61.9%	60.9%	62.1%
St.Dev	7.2%	8.9%	9.3%	5.3%	9.4%	7.2%
3	87.8%	75%	88.9%	89.3%	68.8%	91.1%
3	88.3%	75%	89.4%	88.8%	68.8%	90.6%
3	91.3%	43.8%	95.6%	89.3%	75%	90.6%
Mean	89.1%	64.6%	91.3%	89.1%	70.8%	90.7%
St.Dev	1.6%	14.7%	3%	0.2%	2.9%	0.3%
4	94.1%	10.5%	98%	78.1%	73.7%	78.3%
4	85.8%	36.8%	88.1%	81.6%	78.9%	81.7%
4	81.4%	63.2%	82.2%	87%	21.1%	90.1%
Mean	87.1%	36.8%	89.5%	82.2%	57.9%	83.4%
St.Dev	5.3%	21.5%	6.5%	3.7%	26.1%	5%
Mean	84.5%	43.7%	87.4%	79.7%	54.1%	81.3%
St.Dev	7.4%	21.8%	8.4%	11.6%	23.4%	13%

Appendix D

Data Format

Appendix D.1 contains example data from one short dwell session. Each line, corresponds to one data point, and starts with a tag, and a timestamp. Then comes a comma separated list of value-tags and values. Most of the values are coordinates, but there are also durations, coordinate variances and button associations among them. Each line is terminated with a semicolon.

D.1 Example

```
GazeEnter: 3014159,714, BtnId: btnTopRight;
Gaze: 3014159,714, X: -161, Y:-98;
LeftEyePos: 3014159,714, X: -44,3271759033203, Y: 98,3838321836362, Z:
664,507837033596;
RightEyePos: 3014159,714, X: 21,7625972747803, Y: 100,012688864488, Z:
659,110020062474;
FilteredRightEyePos: 3014159,714, X: 0,453737437725067, Y: 0,472714066505432,
Z: 0,655750732421666;
FilteredLeftEyePos: 3014159,714, X: 0,588673830032349, Y: 0,479769110679626,
Z: 0,670801188151017;
Gaze: 3014173,951, X: -181, Y:-88;
LeftEyePos: 3014173,951, X: -44,4366271972656, Y: 98,5005332969691, Z:
664,194506957415;
RightEyePos: 3014173,951, X: 21,5617439270019, Y: 100,330496602769, Z:
659,101384047283;
FilteredRightEyePos: 3014173,951, X: 0,454155921936035, Y: 0,472099721431732,
Z: 0,65608601888016;
FilteredLeftEyePos: 3014173,951, X: 0,588929057121277, Y: 0,479319870471954,
Z: 0,669952799479006;
Fixation_Begin: 3014172,5023125, X: -177, Y:-90;
Gaze: 3014188,991, X: -190, Y:-86;
LeftEyePos: 3014188,991, X: -44,5446022033692, Y: 98,5287369400165, Z:
```

```
663,861166920724;
RightEyePos: 3014188,991, X: 21,4502193450928, Y: 100,575479897559, Z:
659,063330993474;
FilteredRightEyePos: 3014188,991, X: 0,454387664794922, Y: 0,471603035926819,
Z: 0,656246134439925;
FilteredLeftEyePos: 3014188,991, X: 0,589187741279602, Y: 0,479024410247803,
Z: 0,66894083658849;
Gaze: 3014204,727, X: -218, Y:-91;
Fixation_End: 3014204,26552698, X: -190, Y:-86;
Fixation: 3014172,5023125, Dur: 31,7632144801319, X:-183,5, Y:-88, VarX:42,25,VarY:4;
LeftEyePos: 3014204,727, X: -44,5809829711914, Y: 98,5451711620397, Z:
663,485409875832;
RightEyePos: 3014204,727, X: 21,4283458709717, Y: 100,72758563311, Z:
658,981855520714;
FilteredRightEyePos: 3014204,727, X: 0,454430937767029, Y: 0,471251904964447,
Z: 0,656164347330559;
FilteredLeftEyePos: 3014204,727, X: 0,589306235313416, Y: 0,478720366954803,
Z: 0,667782592773392;
GazeLeft: 3014204,727, BtnId: btnTopRight;
```

Bibliography

M. Kandemir and S. Kaski. Learning relevance from natural eye movements in pervasive interfaces. In *ICMI'12 - Proceedings of the ACM International Conference on Multimodal Interaction*, pages 85–92, 2012.

Erland Tobii AB. 4 considerations when designing ui:s for eye interaction. Online web page, Jan 2014a. URL `http://developer.tobii.com/4-considerations-designing-uis-eye-interaction/`. Accessed: 2015-06-30.

Päivi Majaranta and Andreas Bulling. Eye tracking and eye-based human–computer interaction. In *Advances in Physiological Computing*, pages 39–65. Springer, 2014.

Kenneth B. I. Holmqvist. *Eye tracking : a comprehensive guide to methods and measures.* Oxford : Oxford University Press, 2011.

Jakob Nielsen and Kara Pernice. *Eyetracking web usability.* Berkeley, CA. : New Riders, 2010.

Andrew T. Duchowski. *Eye tracking methodology. : theory and practice.* London : Springer, 2007.

Paola Bonifacci, Paola Ricciardelli, Luisa Lugli, and Antonello Pellicano. Emotional attention: effects of emotion and gaze direction on overt orienting of visual attention. *Cognitive Processing*, 9(2):127, 2008.

Richard Andersson, Marcus Nyström, and Kenneth Holmqvist. *Sampling frequency and eye-tracking measures: how speed affects durations, latencies, and more.* 2010.

Robert JK Jacob. What you look at is what you get: eye movement-based interaction techniques. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 11–18. ACM, 1990.

Anders Tobii AB. Introducing the steelseries sentry eye tracker. Online blog post, Jan 2015a. URL `http://developer.tobii.com/introducing-steelseries-sentry-eye-tracker/`. Accessed: 2015-06-29.

Jenny Tobii AB. What is eye tracking? Online web page, Jan 2014b. URL `http://developer.tobii.com/what-is-eye-tracking/`. Accessed: 2015-06-30.

Robert Tobii AB. Refresh rate and detailed specifications of the tobii rex. Online forum post, Aug 2014c. URL `http://developer.tobii.com/community/forums/topic/refresh-rate-and-detailed-specifications-of-the-tobii-rex/`.

Tobii AB. Developer's guide for .net. Online Manual, Apr 2015b. URL `http://developer-files.tobii.com/wp-content/uploads/2015/01/Developers-Guide-DotNet.pdf`. Accessed: 2015-06-29.

Stuart J Russell and Peter Norvig. *Artificial Intelligence: A Modern approach (3 d Edition)*. 2014.

Cesar Souza. Kernel functions for machine learning applications. Online blog post, Mar 2010. URL `http://crsouza.com/2010/03/kernel-functions-for-machine-learning-applications/`. Accessed: 2015-06-30.

Bernhard Schölkopf and Alexander J. Smola. *Learning with kernels. Support vector machines, regularization, optimization, and beyond.* Adaptive computation and machine learning. Cambridge, Mass. : MIT Press, c2002., 2002.

Scikit-learn developers. Support vector machines. Online web page, Apr 2014. URL `http://scikit-learn.org/stable/modules/svm.html`. Accessed: 2015-06-30.

Ian Jolliffe. *Principal component analysis*. Wiley Online Library, 2002.

Yvan Saeys, Iñaki Inza, and Pedro Larrañaga. A review of feature selection techniques in bioinformatics. *bioinformatics*, 23(19):2507–2517, 2007.

Kevin J Johnson and Robert E Synovec. Pattern recognition of jet fuels: comprehensive GCxGC with anova-based feature selection and principal component analysis. *Chemometrics and Intelligent Laboratory Systems*, 60(1-2):225 – 237, 2002. ISSN 0169-7439.

Rupert G Miller Jr. *Beyond ANOVA: basics of applied statistics*. CRC Press, 1997.

Yiming Yang and Xin Liu. A re-examination of text categorization methods. In *Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval*, pages 42–49. ACM, 1999.

F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

Bionumbers @ Harvard. Average duration of a single eye blink. Online web page, Jun 2013. URL `http://bionumbers.hms.harvard.edu/bionumber.aspx?&id=100706&ver=1`. Accessed: 2015-07-02.

A. Bangor, P. T. Kortum, and J. T. Miller. An empirical evaluation of the system usability scale. *International Journal of Human-Computer Interaction*, 24(6):574–594, 2008.

Stuart Russell and Peter Norvig. Artificial intelligence: A modern approach. Online web page, Oct 2013. URL `http://aima.cs.berkeley.edu/`. Accessed: 2015-07-02.