Graphical Processing Unit Assisted Image Processing for Accelerated Eye Tracking

Dissertation submitted by

Jean-Pierre Louis du Plessis

Student Number: 2006033415

to the

Department of Computer Science and Informatics

Faculty of Natural and Agricultural Sciences

University of the Free State, South Africa

Submitted in fulfilment of the requirements of the degree

Magister Scientiae

2 February 2015

Study Leader: Prof P.J. Blignaut

ABSTRACT

Eye tracking is a well-established tool utilised in research areas such as neuroscience, psychology and marketing. There are currently many different types of eye trackers available, the most common being video-based remote eye trackers. Many of the currently available remote eye trackers are either expensive, or provide a relatively low sampling frequency. The goal of this dissertation is to present researchers with the option of an affordable high-speed eye tracker.

The eye tracker implementation presented in this dissertation was developed to address the lack of low-cost high-speed eye trackers currently available. Traditionally, low-cost systems make use of commercial off-the-shelf components. However, the high frequency at which the developed system runs prohibits the use of such hardware. Instead, affordability of the eye tracker has been evaluated relative to existing commercial systems. To facilitate these high frequencies, the eye tracker developed in this dissertation utilised the Graphical Processing Unit, Microsoft DirectX and HLSL in an attempt to accelerate eye tracking tasks – specifically the processing of the eye video.

The final system was evaluated through experimentation to determine its performance in terms of accuracy, precision, trackability and sampling frequency. Through an experiment involving 31 participants, it was demonstrated that the developed solution is capable of sampling at frequencies of 200 Hz and higher, while allowing for head movements within an area of 10×6×10 cm. Furthermore, the system reports a pooled variance precision of approximately 0.3° and an accuracy of around 1° of visual angle for human participants. The entire system can be built for less than 700 euros, and will run on a mid-range computer system.

Through the study an alternative is presented for more accessible research in numerous application fields.

OPSOMMING

Bliknavolging is a goedgevestigde instrument wat in navorsingsareas soos neurowetenskap, psigologie en bemarking aangewend word. Daar bestaan tans heelwat verskillende tipes bliknavolgers, waarvan die algemeenste die video-gebaseerde afstandsbliknavolger is. Heelwat van die huidige beskikbare afstandsbliknavolgers is baie duur, of lewer ’n relatiewe lae proeffrekwensie. Die doel van hierdie verhandeling is om navorsers die opsie te bied van ’n bekostigbare hoëspoed bliknavolger.

Die bliknavolgerstelsels wat in hierdie verhandeling aangebied word, is ontwikkel om die gebrek aan die tans beskikbare laekoste-hoëspoed-navolgers die hoof te bied. Tradisionele laekostestelsels maak gebruik van kommersiële op-die-rak-beskikbare komponente. Die hoë frekwensie waarteen die ontwikkelde stelsel funksioneer, sluit die gebruik van sulke hardeware uit. In plaas daarvan is die bekostigbaarheid van die bliknavolger beoordeel relatief tot bestaande kommersiële stelsels. Ten einde hierdie hoë frekwensies te fasiliteer, maak die bliknavolger wat in hierdie verhandeling ontwikkel is, gebruik van die Grafiese Verwerkingseenheid, Microsoft DirectX en HLSL in ’n poging om bliknavolgingstake te versnel – spesifiek die prosessering van die oogvideo.

Die finale stelsel is geëvalueer deur eksperimentering ten einde prestasie vas te stel in terme van akkuraatheid, presisie, navolgbaarheid en proeffrekwensie. Deur middel van ‘n eksperiment waarby 31 deelnemers betrek is, is gedemonstreer dat die ontwikkelde oplossing in staat is om proeffrekwensies van 200 Hz en hoër te bemeester, en terselfdetyd hoofbewegings binne ‘n area van 10×6×10 cm toe te laat. Die sisteem behaal verder ’n presisie van naastenby 0.3° en ‘n akkuraatheid van ongeveer 1° van ‘n visuele hoek vir menslike deelnemers. Die volledige stelsel kan vir minder as 700 euros gebou word, en sal funksioneer op ’n middelslag rekenaarstelsel.

Deur middel van hierdie studie word ‘n alternatief gebied vir meer toeganklike navorsing in verskeie toepassingsvelde.

ACKNOWLEDGEMENTS

The author would like to thank the following for their input and support:

- Professor Pieter Blignaut for his invaluable advice and guidance
- Friends, family and colleagues for their advice and support throughout
- Parents for their moral support during the final stages of writing

TABLE OF CONTENTS

Chapter 1: Introduction 1

1.1 Background 1

1.1.1 Types of eye trackers 1

1.1.2 Eye tracking research with respect to remote eye trackers 2

1.1.3 Eye tracker performance 3

1.2 Objectives 4

1.3 Importance of the research 5

1.4 Methodology 6

1.5 Structure of dissertation 7

1.6 Summary 8

Chapter 2: Theory of eye tracking 9

2.1 Introduction 9

2.2 History of eye tracking 10

2.2.1 Early eye tracking 10

2.2.2 Modern eye tracking 11

2.2.3 Conclusion 11

2.3 Eye tracking as a research tool 11

2.3.1 Experimental setup 12

2.3.2 Data acquisition and analysis 12

2.3.3 Saccades and fixations 13

2.3.4 Visual representation of eye tracker data 14

2.3.5 Conclusion 15

2.4 Performance measures 16

2.4.1 Accuracy 16

2.4.2 Precision 17

2.4.3 Latency 17

2.4.4 Sampling frequency 18

2.4.5 Robustness 18

2.4.6 Conclusion 18

2.5 Eye tracking applications 19
2.5.1 Usability studies and market research 19
2.5.2 Input device 19
2.5.3 Reading and neurological research 20
2.5.4 Eye tracker application requirements 21
2.5.5 Conclusion 21
2.6 Types of eye trackers 21
2.6.1 Scleral coil 22
2.6.2 Electrooculography system 22
2.6.3 Video based eye trackers 23
2.6.4 Summary of systems 26
2.6.5 The selected eye tracker 26
2.6.6 Conclusion 27
2.7 Remote eye tracking mechanics 27
2.7.1 Hardware components of a remote eye tracker 27
2.7.2 Software components of a remote eye tracker 30
2.7.3 Gaze estimation 31
2.7.4 Calibration 31
2.7.5 A summary of the remote eye tracking process 31
2.7.6 Conclusion 32
2.8 Remote eye tracker research 32
2.9 Eye models and gaze estimation 32
2.9.1 Regression-based methods 33
2.9.2 Geometric-based gaze estimation 33
2.9.3 Conclusion 35
2.10 Feature detection methods 35
2.10.1 Shape-based approach 36
2.10.2 Feature-based approach 36
2.10.3 Appearance-based approach 37
2.10.4 Hybrid approach 37
2.10.5 Chosen method 38

2.11 Conclusion 38

Chapter 3: Discussion on eye tracking application 39

3.1 Introduction 39
3.2 Existing work 39

3.2.1 Head movement 40
3.2.2 Cost 41
3.2.3 Sampling rate 42
3.2.4 Shortcomings that will be addressed 43
3.2.5 Conclusions 43

3.3 Technical requirements 43

3.3.1 Cost 44
3.3.2 Sampling rate 44
3.3.3 Tolerance towards head movement 44
3.3.4 Precision and accuracy 45

3.4 The Graphics Processing Unit 45

3.4.1 Evolution and development of the GPU 45
3.4.2 Architecture of the modern GPU 47
3.4.3 Graphics APIs and shader languages 47
3.4.4 GPGPU applications 48

3.5 DirectX 49

3.5.1 HLSL 50
3.5.2 Utilising HLSL 50
3.5.3 Conclusion 51

3.6 Proposed solution 51

3.6.1 Overhead 52
3.6.2 Image processing functions and compatibility 52
3.6.3 Camera selection 52
3.6.4 Infrared light sources 52
3.6.5 Conclusions 53

3.7 Hardware setup 53

3.7.1 Selection of eye video size ... 54

3.8 Software development cycle ... 54

3.8.1 Simple threshold ... 54
3.8.2 Separate colour channels ... 55
3.8.3 Improved threshold functions ... 55
3.8.4 Camera adjustments ... 56
3.8.5 Centroid computation ... 56

3.9 Final software ... 57

3.9.1 Pre-process shader ... 57
3.9.2 Post-process ... 59
3.9.3 Noise removal ... 59
3.9.4 Pupil deformation ... 60
3.9.5 Feature point validation ... 60

3.10 Gaze estimation ... 61
3.11 Conclusion ... 62

Chapter 4: Experimental Design and Methodology ... 63

4.1 Introduction ... 63
4.2 Theoretical framework for research ... 64

4.2.1 Evaluation options ... 64
4.2.2 Conclusion ... 65

4.3 Research design ... 65

4.3.1 Research problem ... 65
4.3.2 Research hypothesis ... 66
4.3.3 Conclusions ... 67

4.4 Experimental design and methodology ... 67

4.4.1 Physical setup ... 67
4.4.2 Participants ... 68
4.4.3 Experimental design ... 69
4.4.4 Design motivation ... 70

4.5 Simulations ... 70

4.6 Experimental data capture and analysis 71

4.6.1 Precision 71
4.6.2 Accuracy 71
4.6.3 Sampling frequency 72
4.6.4 Trackability 72
4.6.5 Head box 72
4.6.6 Eye video size 71

4.7 Limitations 72
4.8 Summary 72

Chapter 5: Experimental Results 74

5.1 Introduction 74
5.2 Raw data preparation 74

5.2.1 Missing gaze data 74
5.2.2 Interval selection 75
5.2.3 Removal of outliers 76

5.3 Metrics 76

5.3.1 Accuracy and precision 76
5.3.2 Trackability 77
5.3.3 Selection of calibration points 77
5.3.4 Statistical analysis 77
5.3.5 Demographics 78

5.4 Results related to sampling frequency 78

5.4.1 Sampling frequency vs. precision 79
5.4.2 Sampling frequency vs. accuracy 80
5.4.3 Sampling frequency vs. trackability 81
5.4.4 Conclusion 82

5.5 Results related to head position 82

5.5.1 Head position vs. precision 83
5.5.2 Head position vs. accuracy 83
5.5.3 Head position vs. trackability 84

5.5.4 Conclusion 85

5.6 Artificial performance 85

5.6.1 Frequency vs. precision in the instance of artificial eyes 86

5.6.2 Head position vs. precision in the instance of artificial eyes 86

5.6.3 Conclusion 87

5.7 General performance remarks concerning high frequencies 88

5.7.1 Tolerance toward participants 88

5.7.2 Precision across participants 88

5.7.3 Precision as affected by location on the stimulus 88

5.8 GPU performance and overhead 89

5.9 Conclusions 91

Chapter 6: Conclusion 93

6.1 Motivation 93

6.2 Goals 93

6.3 Results 94

6.3.1 Cost of the eye tracker 94

6.3.2 Quality of reported data 94

6.3.3 Effectiveness of the GPU 95

6.3.4 Optimal head box size 96

6.4 Implications 96

6.5 Further research 97

6.6 Summary 98

References 99

Appendix A – HLSL Tracker technical manual 108

LIST OF FIGURES

Figure 2.1 - A typical eye tracker research setup 12
Figure 2.2 - A gaze plot over search results 14
Figure 2.3 - A typical heatmap 15
Figure 2.4 - An example of good accuracy with poor precision 16
Figure 2.5 - An example of poor accuracy with good precision 17
Figure 2.6 - A scleral coil system 22
Figure 2.7 - An EOG system 23
Figure 2.8 - A head-mounted eye tracker 25
Figure 2.9 - A tower-mounted eye tracker from Sensoric Motor Instruments 25
Figure 2.10 – A frame from a typical eye video with a single IR light source 28
Figure 2.11 - An image depicting the four Purkinje images 28
Figure 2.12 – Figures representing bright and dark pupil images 29
Figure 2.13 - Eye image with feature points correctly located 30
Figure 2.14 – Figure of the eye model 34
Figure 3.1 - The eye tracker setup of Noureddin et al. 40
Figure 3.2 - The eye tracker of Hennessey and Lawrence 41
Figure 3.3 - User interface of the ITU open-source tracker. 41
Figure 3.4 - The low-cost eye tracker of Böhme et al. 42
Figure 3.5 - The DirectX 9 Graphics Pipeline 49
Figure 3.6 - Sample HLSL code 50
Figure 3.7 - The UI-3360CP camera used for the eye tracker 53
Figure 3.8 - The result of the glint threshold test 58
Figure 3.9 - The results of each pass in the noise removal shader 59
Figure 3.10 - The resulting eye video after noise removal 59
Figure 3.11 – Output of steps in the image processing 61
Figure 4.1 - The laboratory setup that will be used during the experiment 68
Figure 5.1 - Spread of gaze dots over the stimulus. 77
Figure 5.2 - The overall trend of precision with respect to changes in sampling rate 79
Figure 5.3 - Average, best and worst recorded precision values 80
Figure 5.4 - Sampling frequency of the eye tracker vs. average accuracy obtained 80
Figure 5.5 - Average, best and worst accuracy obtained for each sampling frequency 81
Figure 5.6 - Trackability of the HLSL tracker for each sampling frequency 82

Figure 5.7 - Precision as affected by head position 83
Figure 5.8 - Accuracy for each head position 84
Figure 5.9 - Trackability of the HLSL tracker at various head positions 85
Figure 5.10 - Degree of precision obtained versus sampling rates for artificial eyes 86
Figure 5.11 - Precision of artificial eyes as affected by head positions 87
Figure 5.12 - Number of participants that fell within precision ranges 88
Figure 5.13 - Average precision over the various parts of the screen 89
Figure 5.14 - Maximum attainable sampling frequency for standard resolutions 90
Figure 5.15 - Reworked eye images and the resulting sampling rates attainable 91
Figure A.1 – The HLSL tracker launcher application 109
Figure A.2 – Application showing the results obtained using artificial eyes 105
Figure A.3 - The UML of the HLSL tracker core 111
Figure A.4 - The EyeVideo class where the initial tracker settings can be modified 112
Figure A.5 - The TrackerControl component 112
Figure A.6 - The Logger singleton 113
Figure A.7 - The structure containing the raw data logged by the eye tracker 113
Figure A.8 - Sample form used by the HLSL sample application 114
Figure A.9 – The CameraSettings dialog used to adjust the camera’s parameters 116
Figure A.10 - Correctly calibrated exposure, gain and gamma 117
Figure A.11 - The post-processes layer with correct camera settings 117
Figure A.12 - Pre-process image illustrating an underexposed image 117
Figure A.13 - Pre-processed eye image illustrating an overexposed image 118
Figure A.14 - Post-process image showing the result of a noisy pupil 118

LIST OF TABLES

Table 4.1 - Summary of the various combinations of experimental 70
Table 5.1 – Summary of participant demographics 78

GLOSSARY

API	An abbreviation for application program interface. An API defines the interaction between software components and contains functions that can be utilized by developers in their own applications.
Vertex	A vertex is a data structure that represents a point in space in computer graphics. Objects in computer graphics are composed of collections of vertices that together form triangles covering the surface of the object. Typically, a vertex will contain information such as the position, normal and colour of the object it forms.
GPGPU	General purpose computing on graphics processing units is the process whereby the graphics processing unit (GPU) is used to perform computations that are traditionally performed on the central processing unit (CPU).
CMOS	Short for complementary metal oxide silicon. CMOS sensors are cheap, easily produced camera sensors with a low power consumption and are capable of sampling images at very high rates.
USB	Universal Serial Bus, an industry standard cable and connectors used for connection, communication and power supply between electronic devices.

CHAPTER 1: INTRODUCTION

Understanding where we focus our attention physically has long been of interest to researchers. By studying the way in which the eyes of a human being look at things, valuable insights can be gained into how that person thinks and what captures his or her attention (Hansen & Pece, 2005; Tatler, Kirtley, Macdonald, Mitchell, & Savage, 2014). One popular tool with which this can be done is through the use of an eye tracker. The purpose of this study is to present researchers with the option of an affordable high-speed eye tracking system with which such research can be conducted. The novelty of the eye tracker lies in the use of the Graphical Processing Unit to facilitate certain of the processes that are required to perform eye tracking.

1.1 Background

Eye tracking is a useful technique through which an individual's eye movements are measured to determine his or her point of regard (Poole & Ball, 2005). As such the field of eye tracking is an important aspect of Human Computer Interaction while it also has applications in fields such as neuroscience, psychology and marketing. The specific application of an eye tracker in the above-mentioned areas can be divided into two categories, namely diagnostic or interactive (Duchowski, 2002).

As a diagnostic tool, an eye tracker provides us with quantifiable evidence of where a user's attention and gaze is directed. By measuring fixations and saccades, one can for instance determine how much attention a user is giving to specific elements on a stimulus such as a computer screen (Poole & Ball, 2005). When used as an interactive device, on the other hand, an eye tracker can be used to simulate a pointing device such as a mouse, or it can be used to determine the layout of graphical environments (Duchowski, 2002). There are several types of eye trackers available for both of these purposes, each with their own strengths and weaknesses. These range from scleral coils to video based eye trackers.

1.1.1 Types of eye trackers

Scleral coils are one of the most accurate tracking systems available (see Section 2.6.1 for more details). However, these systems are intrusive and uncomfortable to use and are also not suitable for gaze point estimations (Duchowski, 2007). Electrooculography (EOG) systems (Section 2.6.2) measure the electromagnetic variation when the dipole of the eyeball musculature moves (Holmqvist et al., 2011). EOG systems tend to score low in terms of accuracy, but do provide a very high sampling frequency.

In contrast to the above types, video-based eye trackers make use of a camera and image processing software to calculate the point of regard on a frame by frame basis, and are thus less intrusive than the previous two systems. Video-based eye trackers are discussed in detail in Section 2.7. The point of gaze is calculated using the measurement of visible features of the eye, such as the pupil and corneal reflections generated by Infrared Light Emitting Diodes (IR LEDs) (Duchowski, 2007). A further advantage of video-based eye trackers is that they can be either remote or head mounted (Li, Babcock, & Parkhurst, 2006).

Remote, video-based, eye trackers consist of a camera placed on the table in front of the user, capturing the eyes. One or more arrays of IR LEDs are also placed around a computer monitor that displays the object of interest to the researcher (the stimulus). Head-mounted eye trackers contain the same components, with two differences. Firstly, the camera and IR LEDs are mounted on the user's head, as opposed to being placed on a table in front of the user. Secondly, in addition to the camera and IR LEDs, a head-mounted eye tracker has an additional camera called a scene camera. This camera records what the user is looking at.

Remote eye trackers offer comfort and ease of use compared to other systems, allowing the user to make use of the system for longer periods of time (Morimoto & Mimica, 2005). Remote eye trackers are also more natural to use (Holmqvist et al., 2011) and are cheaper than other eye tracking solutions (Enright, 1998).

If we take into account the relative strengths and weaknesses of the eye tracking techniques discussed above, the solution developed as part of this study will need to attempt to combine both the ease of use and comfort of a remote eye tracker with a high data sampling frequency. By working with a remote eye tracker, the system will provide the relevant data needed with which to perform gaze estimation in real time, something which is not possible with scleral coil or EOG systems.

1.1.2 Eye tracking research with respect to remote eye trackers

As mentioned in the previous section, there are several different kinds of eye trackers available for use. However, the focus of this dissertation will fall specifically on remote video-based eye tracking. Two categories of research involving remote, video-based eye tracking will be discussed, namely eye localisation in the image and gaze estimation (Hansen & Ji, 2010). Eye localisation research focuses on developing better and faster methods to locate the eyes or feature points such as the pupils and corneal reflections within the eyes in the video. Gaze estimation on the other hand refers to the process whereby the user's point of gaze is calculated by analysing feature points such as pupils and corneal reflections. While both aspects of eye tracking research will be discussed, the focus in this dissertation will be on developing an eye localisation process that utilises the Graphical Processing Unit (GPU).

The eye localisation process consists of detecting the existence of the eyes and establishing their position in the image. These two tasks must also be performed on a frame by frame basis. Eye localisation can be performed on one or both eyes. When both eyes are used, reference is made of a binocular eye tracker, whereas single-eye trackers are referred to as being monocular. When compared to monocular eye trackers, binocular eye trackers can give better accuracy. This is particularly true in low cost systems (Holmqvist et al., 2011).

The eye localisation process can be time consuming depending on the technique used (Mulligan, 2012). As part of developing a high-speed remote eye tracker, this study will focus on developing a localisation method that can be used in real time and at high sampling rates. The details of this localisation procedure are discussed in Chapter 3. For the purpose of this research study, a high sampling rate will be considered to be in excess of 200 Hz. The novelty of this research lies in the use of the Graphical Processing Unit (GPU) to perform image processing, thereby facilitating the task of eye localisation. Due to the low cost of the proposed solution, the eye tracker developed in this study will also be a binocular model, and will make use of a computer system that can reasonably be regarded as standard in today's terms. Hence, it will be possible to evaluate the entire system on an unmodified laptop in the mid-price range.

1.1.3 Eye tracker performance

The performance of a remote eye tracker is influenced by several factors. Eye trackers are for instance vulnerable to users who wear spectacles or have lazy eyes (Hansen & Ji, 2010; Poole & Ball, 2005). Spectacles can generate additional reflections that are identical in appearance to corneal reflections. A lazy eye on the other hand can obscure part of the pupil and make it difficult to identify. An additional factor that plays a significant role in the eye tracking process is that of head motion. Even small head movements can have a disastrous effect on the accuracy of the gaze estimations (Morimoto & Mimica, 2005). Lastly, the condition of ambient light can also play a part. In bright outdoor conditions, the pupil can become very small and may contain many additional corneal reflections (Hansen & Ji, 2010). This is because the sun emits vast amounts of infrared light. As a result, the contrast between the iris and the sclera may not be good in these lighting conditions (Ryan, Duchowski, & Birchfield, 2004).

The eye tracker developed in this study will primarily be used in a laboratory, and the lighting problems that occur outdoors will therefore not be a concern. Factors that are of particular interest to the study are, however, the trackability, precision and accuracy attainable at different sampling frequencies and head positions. In the context of this dissertation, trackability refers to the percentage of frames that can be successfully tracked, with minimal loss of data due to incorrect feature point detection.

1.2 Objectives

Although several aspects of the goals of the study have been alluded to above, a succinct summary of the overall objectives of the research will be provided here. The primary aim is to present a low-cost eye tracking solution capable of obtaining good accuracy, precision and trackability at a high sampling rate. Further to the above, the system developed should be supported by commonly available computer systems, and should therefore make use of technologies that are widely supported by modern hardware. Moreover, the eye tracking application aims to incorporate the strengths of the GPU into the eye tracking process in an attempt to accelerate certain image processing tasks. In terms of previous work, very little has been done in involving the GPU in the eye tracking process (Duchowski, Price, Meyer, & Orero, 2012; Mulligan, 2012). By using Microsoft's High Level Shader Language and DirectX, it will be demonstrated that the eye localisation task can be performed at a high sampling rate (200 Hz). The eye tracker should also allow for head movements in the order of 10×6×10 cm (x,y,z). As an additional step, the optimal combination of image size and data acquisition rate will also be established.

In order to investigate the performance of the eye tracker and be able to judge its success, the following hypotheses have been formulated:

- $H_{0,1}$: The sampling frequency has no effect on the data quality (precision, accuracy and trackability).
- $H_{0,2}$: The position of the head within the head box has no effect on the data quality (precision, accuracy and trackability).

Each of these hypotheses can in turn be separated into three sub-hypotheses based on the individual measures of data quality (accuracy, precision and trackability). Thus, the hypotheses of the dissertation can be restated as follows:

- $H_{0,1.1}$: The sampling frequency has no effect on the precision of the eye tracker.
- $H_{0,1.2}$: The sampling frequency has no effect on the accuracy of the eye tracker.
- $H_{0,1.3}$: The sampling frequency has no effect on the trackability of the eye tracker.
- $H_{0,2.1}$: The position of the head within the head box has no effect on the precision of the eye tracker.
- $H_{0,2.2}$: The position of the head within the head box has no effect on the accuracy of the eye tracker.
- $H_{0,2.3}$: The position of the head within the head box has no effect on the trackability of the eye tracker.

The hypotheses will be tested on the basis of the results of an experiment conducted. Details of this experiment can be found in Chapter 4.

1.3 Importance of the research

The development of robust, non-intrusive eye detection methods is of paramount importance (Hansen & Ji, 2010). While eye tracking has numerous applications, the cost of commercial eye trackers renders them inaccessible to many (Kumar, 2006; Li et al., 2006). High speed eye trackers such as the Tobii TX300, SMI RED 500 and SMI Hi Speed 1000 deliver good performance, but are extremely expensive. More recently, devices such as the EyeX (http://www.tobii.com/en/eye-experience/eyex/) have become available. While affordable, the EyeX does not offer a particularly high sampling rate – experience has shown it to be in the order of 50 Hz. Thus, researchers have the option either of employing expensive high-speed eye trackers, or the more commonly available but somewhat cheaper sub-60 Hz eye trackers.

In general, it has been shown that the smaller the saccades are, the higher the required sampling frequency is (Holmqvist et al., 2011). Reading research, for example, requires eye trackers with sampling rates in excess of or equal to 500 Hz (Reichle, Pollatsek, Fisher, & Rayner, 1989). Studies have shown that saccades during reading are about 30 milliseconds in duration (Rayner, 1998). Enright (1998) demonstrated that the course sampling rate of 50-60 Hz is problematic when sampling saccades with durations smaller than 40 milliseconds and saccades smaller than 10°. Furthermore, research in neuroscience and psychology involving microsaccades requires a sampling rate of 200 Hz or higher (Holmqvist et al., 2011). In addition to various applications, high-speed eye tracking also provides the benefit of potentially better precision, as the high sampling frequency can be traded for a higher precision by averaging results within a certain time frame (Holmqvist et al., 2011) – with an additional side effect of introducing some latency into the system.

Non-commercial systems that are capable of high sampling rates do exist (Clarke, Ditterich, Drüen, Schönfeld, & Steineke, 2002; Hennessey, Noureddin, & Lawrence, 2008; Mulligan, 2012). However, not all of these systems have been evaluated in depth in terms of precision, accuracy and trackability. In this study, a novel method to high-speed eye tracking is proposed and evaluated in order to present an alternative low-cost solution to researchers requiring the use of a high-speed tracker.

1.4 Methodology

The first step in the research process covers the examination of existing work and eye tracking solutions. Current techniques for locating and tracking the eyes are identified and analysed for strengths and weaknesses. The results of the analysis are subsequently used to construct an eye tracking solution containing hardware and software components that meet the objectives as stated above.

Pilot studies involving the developed solution are conducted throughout to gauge the performance in terms of required sampling rates. The data acquired from the pilot studies are used to further refine the solution.

The finalised solution is subjected to evaluation by means of the conducting of an experiment involving 31 participants. The experiment focuses on the precision, accuracy and trackability of the system for a variety of sampling rates and head positions. An additional experiment involving artificial eyes is then used to gauge the theoretical performance of the system at a number of different frequencies while employing various head box sizes and head positions.

The results of these studies are analysed using statistical methods and conclusions are drawn from the data provided.

1.5 Structure of dissertation

This dissertation is divided into six chapters and one appendix, and is outlined as follows:

- Chapter 1 : Introduction

This chapter provides a broad overview of the theoretical background of eye tracking. Issues related to eye tracker performance are briefly discussed, as well as the various types of eye tracking systems that are available. Next, the objectives of the dissertation are stated and the importance of the research is motivated. Finally, the methodology used to develop the solution as well as test its performance is briefly described. The chapter concludes with a short summary.

- Chapter 2 : Theoretical background

Chapter 2 contains a literature overview that pertains to the current topic. A broad theoretical foundation of eye tracker terminology is provided. This is followed by a detailed examination of the types of eye tracking systems available and the process of feature point detection with relevant extracts from the literature. A suitable eye tracker type and feature detection method is selected based on the literature and objectives stated in the first chapter.

- Chapter 3: Discussion on the eye tracker application

Chapter 3 focuses on the specific details regarding the eye tracking application that is developed to provide a cost-effective high-speed eye tracker. The chapter first provides a summary of existing work, and identifies some of the shortcomings that will be addressed. The role of the GPU in the proposed solution is motivated against the relevant background knowledge of the technologies involved in incorporating the GPU. Finally, the completed eye tracker application is discussed and conclusions regarding the final solution are stated.

- Chapter 4 : Methodology

In this chapter, the methodology used to verify the performance of the proposed solution is discussed. Full details of the setup used during the testing phase are provided with regard to system setup and testing conditions. Details of the target participant demographics are provided. Additionally, the testing process is described, along with any statistical methods that will be used to analyse the results. The chapter concludes with a short summary of the test.

- Chapter 5: Results

In the penultimate chapter, the results of the test performed in the previous chapter (Chapter 4) are discussed. The eye tracker's performance is evaluated with respect to the performance objectives stated in Chapter 1. Each of these results is reported and analysed through the use of graphical representations and relevant statistical methods. The chapter concludes with a summary of the results.

- Chapter 6: Conclusions

In the final chapter, a critical analysis of the eye tracker's performance is provided in light of the results obtained in Chapter 5 and from existing work, together with suggestions for improvements and future research possibilities. The chapter ends with a summary of the entire dissertation.

- Appendix A

Appendix A contains the technical documentation of the final eye tracker application. Included in this documentation is a brief explanation of the various applications used alongside the developed tracker to perform the experiment in Chapter 4 and extract and analyse the data. For researchers wishing to extend the application, or incorporate it in other applications, the specifications and usage of the software components in the developed tracker are provided, along with relevant code samples explaining the usage of the individual components.

1.6 Summary

In this introductory chapter, a short overview of eye tracking has been provided. The objectives of the study have been stated succinctly, and the value of the research has been presented as motivation for the study. In addition hereto, a brief overview of each of the ensuing chapters has been included, along with a summary conclusion.

The following chapter (Chapter 2) provides an in-depth literature study of eye tracking, focusing specifically on aspects thereof related to video-based remote eye trackers.

CHAPTER 2: THEORY OF EYE TRACKING

In this chapter a broad overview of the field of eye tracking will be provided. Included in this overview is a discussion of eye tracking applications and those applications that are most relevant to the goals of this dissertation. A selection of different types of eye tracking systems will be discussed and concerns related to their usability will be raised in order to determine the most suitable kind of eye tracker to develop in light of the goals of the study. Finally, the chapter concludes with a discussion of the type of eye tracker chosen to be developed.

2.1 Introduction

On a day to day basis, we acquire a vast amount of information through our eyes. Many of our daily tasks require visual information (Tatler, Kirtley, Macdonald, Mitchell, & Savage, 2014). By studying what we focus on while performing these tasks, we can learn about the cognitive processes involved in these tasks, as well as gain insight into what holds our attention (Duchowski, 2007; Hansen & Pece, 2005; Poole & Ball, 2005; Tatler et al., 2014).

The field of eye tracking has existed since the late 19th century (Duchowski, 2002) and is now well established as a research tool in analysing human behaviour (Tatler et al., 2014). As a result of its versatility, eye tracking has numerous applications in fields such as neuroscience, psychology, marketing and human computer interaction (Duchowski, 2002; Morimoto & Mimica, 2005), necessitating the ongoing development of tracking systems with improved features.

To reiterate what was stated in Chapter 1, eye tracking is a technique with which an individual’s eye movements can be observed, and his or her point of regard determined. The way in which this is performed depends on the type of eye tracker being utilised, as the different types of eye trackers make use of different techniques in order to track eye movements (Morimoto & Mimica, 2005). It is also worth noting that not all eye tracking devices are suitable for determining a point of regard. For example, EOG and scleral coil systems are typically not used to determine point of regard measurements (Duchowski, 2007), and are used instead in clinical studies involving research into eye movements such as those made during sleep (Joyce, Gorodnitsky, King, & Kutas, 2002).

There are several types of eye tracking systems currently available to researchers (Duchowski, 2007; Holmqvist et al., 2011; Morimoto & Mimica, 2005) each with their own strengths and weaknesses. It is one of the goals of this study to develop an eye tracker that attempts to improve upon selected features of existing eye trackers by utilising hardware such as the Graphical Processing Unit (GPU) to develop an application that can be used across a wide range of systems. Details of the features that will be addressed are discussed in the following chapter. Chapter 2 will instead focus on providing the reader with an understanding of how eye tracking is utilised, and what a typical video-based eye tracking system is comprised of.

The structure of the rest of the chapter is as follows: Firstly, the history of eye tracking will be overviewed and a discussion provided on the way that eye tracking is utilised as a research tool. Next, the various performance criteria for eye trackers will be defined. A selection of application areas of eye tracking technology will then be reviewed, and the performance criteria of each application area highlighted. The section that follows will discuss the mechanics of remote eye trackers. Finally, the chapter will conclude with a discussion of gaze estimation and feature detection methods.

2.2 History of eye tracking

In this section, a short discussion of the history of eye tracking technology is provided. This is done to identify a trend in research directions and ensure that the work done as part of this dissertation addresses needs that are relevant.

2.2.1 Early eye tracking

The first eye tracking systems were developed in the late 19th century (Holmqvist et al., 2011; Tobii Technology, 2010) and consisted of mechanical devices that were uncomfortable to wear and very intrusive. During this era, researchers discovered basic properties of eye movements (Duchowski, 2002). One example of research from this time is the work of Javal (1879) who examined eye movements during reading. The eye tracking techniques utilised at this stage tended to be very invasive and required direct contact with the cornea (Jacob & Karn, 2003). Delabarre (1898) for example, made use of a moulded lens that was fitted onto the participant's eye. The eye required an anaesthetic before the application of the lens to minimise discomfort for the participant.

In the 20th century, researchers developed several ways with which to record the user's eye movements. Dodge and Cline (1901) proposed a new method for recording the horizontal movements of the eyes. Their technique made use of a camera and rapidly moving film. The participant's eye movements would appear as oblique lines on the film negative. Their technique is considered to be the first non-invasive method of tracking eye movements (Jacob & Karn, 2003). Later in the 20th century, Fitts, Jones and Milton (1950) made use of a camera and mirrors to observe pilots' eye movements during landing approaches. During the 1970s, eye tracking research made great advances in the technology used to track eye movements, as well as in research applications (Jacob & Karn, 2003). Much of the research during this period was focused on the links between eye movements and cognitive processes (Jacob & Karn, 2003).

2.2.2 Modern eye tracking

Today, there are several eye tracking systems available to researchers. Examples of such eye tracking systems are electromagnetic coils, electrooculography systems and video based eye trackers (Duchowski, 2007; Holmqvist et al., 2011; Poole & Ball, 2005). The features of each of these systems will be discussed in further detail in Section 2.5. Modern day eye trackers tend to be predominantly video based eye trackers, due to their ease of use and non-intrusive nature. However, EOG systems are sometimes used as they are affordable and have a high sampling frequency (Holmqvist et al., 2011). The focus in this study, however, will be on developing a video based remote eye tracker.

2.2.3 Conclusion

In this section, a short overview of the history of eye tracking technologies has been provided and the predominant type of eye tracker identified. In the following section, a typical setup during an eye tracker study will be discussed along with the various measurements that are taken.

2.3 Eye tracking as a research tool

Although eye tracking is by no means used exclusively as a research tool, all studies involving eye tracking invariably consist of certain key components. In addition to the physical setup, data is captured and analysed using a variety of measures and representations. For the eye tracker developed in this dissertation to be useful, the type of data it provides should be analysable using the common measures. It is the goal of this section to discuss the commonly used setup for eye tracking, the recording of measurements and the ways in which the results of an eye tracking study are analysed and presented.

2.3.1 Experimental setup

A typical eye tracking study will, in addition to the eye tracker, contain a stimulus on which the participant will focus his or her attention (Duchowski, 2007). The type of stimulus is determined by the research question that is posed (Holmqvist et al., 2011). One example of a common stimulus is a website or computer application displayed on a computer screen (Duchowski, 2007).

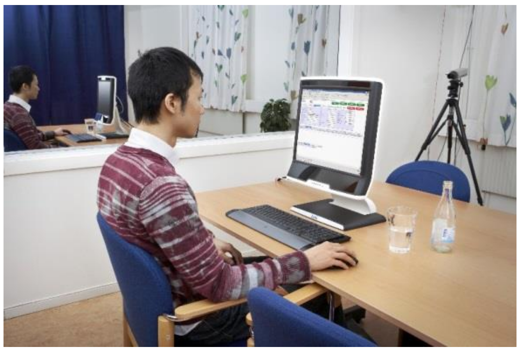


Figure 2.1 - A typical eye tracker research setup
Source: http://www.acuity-ets.com/Solutionsforusability.htm
Last Accessed: 02/02/2015

The stimulus can be divided into various regions referred to as areas of interest (AOIs) or regions of interest (ROIs). These areas contain the sections of the stimulus that the researcher is most interested in, and are also selected from the research question that is asked (Holmqvist et al., 2011). Eye movements that fall within these regions provide useful and meaningful feedback to the researcher (Poole & Ball, 2005).

2.3.2 Data acquisition and analysis

During a study, participants are required to perform a series of tasks during which a variety of measurements are taken. Typical measurements may include the time it took the participant to successfully complete the tasks or the number of steps involved (also referred to as efficiency), and the number of errors that occurred during the execution of a task (effectiveness) (Duchowski, 2007; Tullis & Albert, 2008). Additionally, the participant's eye movements are analysed (Duchowski, 2007; Holmqvist et al., 2011). Of particular interest to researchers are the fixations and saccades (discussed in the next sub-section) that occurred during the execution of the tasks (Salvucci & Goldberg, 2000). Examples of some common eye movement metrics include number of fixations, fixation duration, time to first fixation and number of saccades (Poole & Ball, 2005; Tullis & Albert, 2008). Additional eye measurements that are sometimes examined are pupil size and blink rate (Poole & Ball, 2005).

2.3.3 Saccades and fixations

Eye movements are characterised by series of quick jumps or high velocity movements that are followed by periods of time during which the eye is stabilised and remains relatively still (Blignaut & Beelders, 2009). The breaks in eye movements are referred to as fixations, whilst the rapid movements between these pauses are referred to as saccades (Salvucci & Goldberg, 2000).

Saccades are typically measure in terms of a duration in milliseconds, though it is not uncommon to express the distance or velocity of the saccade in degrees per second (Rayner, 1998). The duration of a saccade will depend on the distance covered by the saccade.

Fixations are defined by a series of gaze points that are located close to each other in terms of location and time, and consist of a certain duration measured in milliseconds (Blignaut, 2009; Hornof & Halverson, 2002). For a group of data points to qualify as a fixation, they must exist for a sufficient duration, and must be located in close proximity to each other. These values are referred to as the duration and distance threshold (Blignaut, 2009). The durations of fixations as interpreted by researchers vary, but are generally assumed to fall within the range of 200 – 400 ms (Salvucci & Goldberg, 2000). However, depending on the application, some researchers may study fixations as short as 50 ms in duration (Dalton et al., 2005).

It can be argued that all eye-tracking research invariably examines eye movements in some way, and is thus interested in fixations and saccades. Therefore, the ability to analyse these movements forms a crucial part of any eye tracker tool, and should be within the capabilities of the developed eye tracker.

As saccades form such a significant part of research involving eye tracking, it is important that they be captured as accurately as possible. Moreover, in order to capture high velocity saccades, for example velocities in excess of 300 degrees per second (Salvucci & Goldberg, 2000), one requires sampling rates higher than 50 to 60 Hz. This is particularly true when examining peak saccadic velocities in excess of 500 degrees per second (Clarke, Ditterich, Drüen, Schönfeld, & Steineke, 2002). Also, for small saccades, such as those that occur during reading, one finds that 60 Hz is insufficient (Andersson, Nyström, & Holmqvist, 2000; Enright, 1998). The eye tracker developed in this study will thus attempt to address the issue presented by 60 Hz eye trackers by developing an eye tracker that samples at 200 Hz or higher.

2.3.4 Visual representation of eye tracker data

The results of eye tracking experiments are often presented graphically. Fixations are represented with dots that are sized in proportion to the duration of the fixation. Saccades are then represented as the lines connecting these dots. An image of the stimulus with these superimposed fixations and saccades is referred to as a gaze plot.

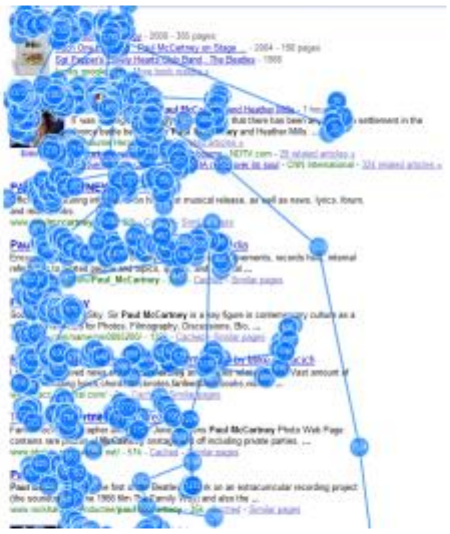


Figure 2.2 - A gaze plot over search results

An additional representation often used is a heatmap. The fixations of all participants of the study can be superimposed on a screen capture of the stimulus, with a colour indicating the weight, or number of fixations (Tobii Technology, 2010). Typically warmer colours represent areas that enjoyed more attention than areas represented by colder colours (Holmqvist et al., 2011). The figure below illustrates a typical heatmap. In this instance, the red areas represent the regions that received the most attention.

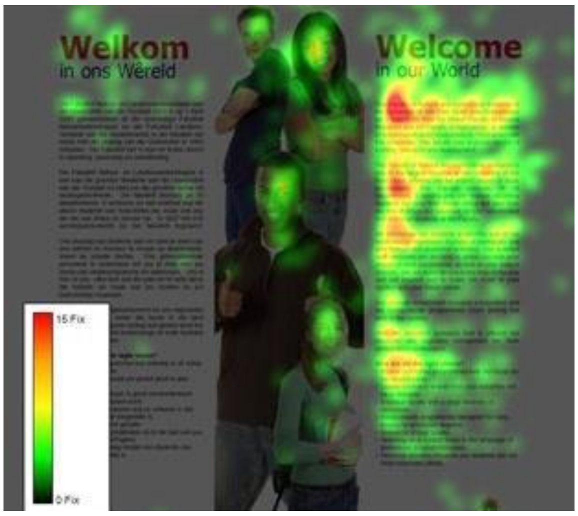


Figure 2.3 - A typical heatmap

The stimulus can be divided into regions referred to as areas of interest. Using software, researchers can explore the fixation duration, frequency and number of returns between different elements, parts or components in any visual scene or visual display (Horsley, 2014). Using data visualisations such as gaze plots and heat maps, along with area of interest analysis, researchers can visualise what users focus their attention on. It follows then that for the eye tracker developed in this study to be usable for research, it should be capable of supplying information that can ultimately be used to analyse fixations and generate visualisations such as heatmaps and gaze plots. Thus, the eye tracker should be capable of performing gaze estimation.

2.3.5 Conclusion

In this section a basic theoretical background was provided regarding the use of eye tracking in a research setting. Common metrics and measures that are used in data analysis were provided and common visualizations of eye tracker data were discussed to determine the nature of the data that an eye tracker should be capable of providing.

The following section outlines the various measures that are used to evaluate the performance of an eye tracking system, and by implication the eye tracking system to be developed in this study.

2.4 Performance measures

Each type of eye tracker has strengths and weaknesses. In order to compare various eye tracking systems and select one appropriate to the type of application that this dissertation will focus on, it is necessary to define certain metrics by which the eye tracker can be judged. It is the purpose of this section to list and define five commonly used measurements to compare the performance of eye trackers. These five factors are accuracy, precision, latency, sampling frequency and robustness, and together they form what is referred to as data quality (Holmqvist et al., 2011).

2.4.1 Accuracy

Accuracy is the difference between the reported point of gaze (PoG) and the actual point of regard (Holmqvist et al., 2011). This is also called systematic error or drift (Hornof & Halverson, 2002).

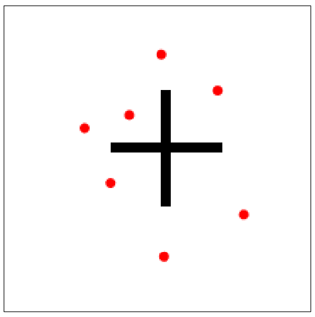


Figure 2.4 - An example of good accuracy with poor precision

Accuracy is calculated as the average angular offset in degrees of visual angle measured between the reported fixation location and the actual location, and this is also referred to as spatial accuracy (Holmqvist & Nyström, 2012). The key to good accuracy lies in a robust gaze estimation algorithm (Holmqvist et al., 2011). Head movements have an impact on accuracy. It has also been shown that accuracy suffers when working with large gaze angles (Blignaut & Wium, 2014). Accuracy can sometimes be improved manually when the fixations are superimposed on the visual stimulus and there is a consistent pattern of systematic error (Hornof & Halverson, 2002).

2.4.2 Precision

Precision can be defined as the ability of an eye tracker to reproduce a measurement reliably (Holmqvist et al., 2011). The variance encountered in accuracy is also referred to as spatial precision (Holmqvist & Nyström, 2012). The detection of gaze events such as fixations and saccades is simpler if one has an eye tracker with good precision (Nyström, Andersson, Holmqvist, & van de Weijer, 2013). Precision errors are introduced into an eye tracker system through system noise and head movements (Hennessey, Noureddin, & Lawrence, 2008). There are two common ways in which precision is measured (Holmqvist et al., 2011). The first method used to determine precision is by looking at the standard deviation of the mapped x and y coordinates combined. This measure is also referred to as pooled variance. The second method is Root Mean Square (RMS) and examines the differences between mapped coordinates on a point to point basis.

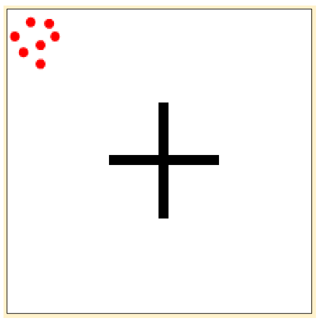


Figure 2.5 - An example of poor accuracy with good precision

Precision can be improved by averaging out the estimated gaze points within a certain time window (also referred to as smoothing). However, this introduces higher latency into the system (Hennessey et al., 2008; Jacob, 1995).

2.4.3 Latency

Latency is the delay between a recorded eye movement, and the time the eye movement is recorded (Holmqvist & Nyström, 2012; Holmqvist et al., 2011). Latency is introduced into the eye tracker as each video frame needs to be processed and the display updated (Duchowski, 2007). Variance in latency is referred to as temporal precision (Holmqvist & Nyström, 2012). One method that can be used to measure latency is having the recording computer trigger a movement in an artificial eye and then measuring the time it takes for the system to report the change (Holmqvist & Nyström, 2012; Holmqvist et al., 2011). A high-quality eye tracker should have a latency of less than 3 ms (Holmqvist et al., 2011). However, to limit the scope of this dissertation, the latency of the developed eye tracker will not be measured.

2.4.4 Sampling frequency

Sampling frequency in eye trackers is measured in Hertz (Hz), and refers to the rate at which an eye tracker captures data about eye movements (Andersson et al., 2000). The faster the eye movements involved in the particular area of research, the higher the sampling frequency that is needed (Holmqvist et al., 2011). With a higher frequency, the start and end point of saccades and fixations can be measured more accurately. High sampling frequency also affects the size of saccades that can be measured. For example, a high sampling frequency is required to measure small saccades (Andersson et al., 2000). An additional advantage that high sampling rate systems have is the potential for improved precision through the use of smoothing (see 2.4.2) which yields a more stable gaze point at the cost of a higher latency. This fact serves as an additional motivation for the higher frequency eye tracker to be developed for the purpose of this dissertation.

High speed eye trackers are considered to be 250 Hz or higher (Holmqvist et al., 2011). One consideration for high sampling frequency systems is that they produce larger files, and thus hard drive capacity can become an issue.

2.4.5 Robustness

Robustness is a measure of how well an eye tracker works for a large variety of participants. It also describes the eye tracker's tolerance toward factors such as the wearing of spectacles and mascara. An eye tracker with poor robustness may suffer from frequent data loss and poor data quality (Holmqvist et al., 2011). Robustness is also sometimes referred to as trackability (Blignaut & Wium, 2014), which is the ratio of the total number of samples recorded by an eye tracker over a time period over the number of samples that were valid. In the context of this dissertation, the eye tracker will be evaluated in terms of the trackability.

2.4.6 Conclusion

In this section, five metrics with which an eye tracker's performance can be measured have been identified and defined. However, to limit the scope of this only accuracy, precision, trackability and sampling frequency will be examined in the final evaluation of the developed system. In the following section, some typical applications of eye tracking technology are discussed.

2.5 Eye tracking applications

Eye tracking is utilised in a multitude of fields, including neuroscience, psychology, usability studies and marketing (Duchowski, 2007; Morimoto & Mimica, 2005; Tatler et al., 2014), and has gained wide acceptance as a powerful research tool. In this section, the use of eye tracking in a selection of relevant fields is expounded on. Real-world applications using eye tracking technology are discussed, with examples from existing work provided. The requirements in terms of eye tracking metrics for the various applications are stated, in order to establish performance requirements for the envisaged eye tracker

2.5.1 Usability studies and market research

A common use nowadays for eye trackers is to make use of them to evaluate the usability of interfaces (Duchowski, 2002; Tullis & Albert, 2008). By defining areas of interest around certain parts of an interface, and analysing the corresponding fixations and saccades over these parts, one can evaluate factors such as the visibility and relevance of these interface elements, as well as make changes accordingly to improve these elements (Goldberg & Kotval, 1999; Poole & Ball, 2005). Goldberg and Kotval (1999), for example, showed that interfaces that were poorly designed resulted in more fixations than well-designed ones. In other words, poorly designed interfaces produce less efficient search behaviour in terms of eye movements. Utilising the method above, eye trackers can also be utilised to help develop well designed websites, as well as locate the best possible locations for the placement of advertisements (Duchowski, 2007).

Eye tracking is also used for market research, and can provide assistance in determining how customers view advertising material over multiple mediums (Duchowski, 2002). As the retail industry is a very competitive environment, it is crucial that one gain an understanding of the target market's shopping behaviours (Harwood & Jones, 2014). This is now possible thanks to the development of mobile eye tracking technology.

2.5.2 Input device

Eye trackers can function as input devices. Using a combination of eye tracking and speech recognition, for example, one can fulfil the function of a pointing device (Beelders & Blignaut, 2014; Bulling & Gellersen, 2010; Duchowski, 2007). Using eye trackers, people with physical disabilities can interact with their environment in ways that were previously not possible. For instance, Hansen, Alapetite, MacKenzie and Møllenbach (2014) suggest that a person confined to a wheelchair can make use of an eye tracker controlled unmanned aerial vehicle to explore otherwise inaccessible areas, though this is certainly not only an option for physically disabled people.

There are several issues that must be addressed when using an eye tracker as an input device. In order to interact with technology, some way to initiate a command or selection must be developed. There are several suggestions that have been made. The basis of these techniques is the use of fixations as input. This method, however, leads to what is called the Midas touch, where everything the user looks at is activated (Jacob, 1991). To circumvent this problem, an alternative activation command could instead make use of blinks. Another adaption of the fixation method is to make use of dwell time, and this is a commonly utilised method for selection (Dybdal, Agustin, & Hansen, 2012; Majaranta & Räihä, 2002). Zhang and Mackenzie (2007) have indicated that a dwell time of 500 ms is sufficient to ensure both a reasonable response time, as well as eliminate the Midas touch problem. Experience has shown that with practice it is possible to decrease the dwell time. While blinks will work, Jacob (1991) rejects this system on the grounds that it is unnatural and forces the user to consciously focus on blinking. Another technique is to make use of gaze gestures (Dybdal et al., 2012; Majaranta & Räihä, 2007; Wobbrock, Sawyer, & Duchowski, 2008). Dybdal et al. (2012) define a gaze gesture as a certain sequence of predefined eye movements that need to be carried out successfully for a selection command to take place.

2.5.3 Reading and neurological research

In addition to usability studies, market research and use as input devices, eye trackers are also utilised in research involving reading, psychology and neuroscience. Eye trackers can provide insights about eye movements that occur while reading. For example, it has been observed that fixation time increases and saccadic length decreases as text becomes harder to read (Duchowski, 2007). Furthermore, fixation times also increase when reading aloud as opposed to reading softly (Rayner, 1998).

As a diagnostic tool, the eye tracker can be used to diagnose mental issues. Bowling and Draper (2014) suggest that by analysing saccadic eye movements during a series of tests, one can detect reduced inhibitory control caused by aging for example. The analysis of microsaccades can also be used to study covert attention (Engbert & Kliegl, 2003; Hafed & Clark, 2002).

2.5.4 Eye tracker application requirements

Each field of application has distinct requirements in terms of eye tracking performance, and finding the correct combination may be difficult (Holmqvist & Nyström, 2012).

While a high degree of precision is required for research involving small eye movements, such as with the analyses of microsaccades and reading (Holmqvist et al., 2011), accuracy is less of a factor. For input devices, however, a high degree of accuracy is required, as low accuracy can cause the incorrect element to be activated (Holmqvist & Nyström, 2012).

For sampling frequency, one must consider the speed of eye movements that are to be observed (Holmqvist et al., 2011). It has already been discussed that for reading research, 60 Hz eye trackers are insufficient. However, for usability studies or as an input device, 60 Hz may suffice. When examining microsaccades, however, one again finds that 60 Hz provides insufficient detail. It has been suggested that a minimum sampling rate of 200 Hz is required for this kind of research (Holmqvist et al., 2011). As there already exist low-cost eye tracking solutions in the order of 60 Hz, the eye tracker developed in this study will not focus on applications in usability studies or as an input device. Instead, the focus will be on applications requiring a higher sampling frequency.

2.5.5 Conclusion

A selection of eye tracking applications has been discussed to provide the reader with the relevant background for the intended use of the eye tracker that will be developed. The requirements of each application area in terms of eye tracker performance metrics have been listed to identify data quality requirements for the developed eye tracker. Based on these requirements, reading research and neurological research appear to be the most relevant application fields related to this study, as they require a high frequency sampling rate and good precision. However, as this kind of eye tracking system can also report gaze coordinates, it is conceivable that it can be used for other applications, such as usability studies. In the following section, the various types of eye tracking systems available are discussed briefly.

2.6 Types of eye trackers

This section contains a short discussion of the different types of eye trackers that are currently available. By comparing and contrasting the existing options, it will be possible to select a type appropriate to the application area identified in the previous Section (2.5). The first two eye trackers discussed are intrusive eye trackers, specifically the scleral coil and electrooculography systems. Next, video based eye trackers are discussed. Finally, the section concludes with a summary of the relative strengths and weaknesses of each type of system, motivating the choice of eye tracking system developed as part of this study.

2.6.1 Scleral coil

Scleral coil systems are among the most accurate systems available, and are capable of achieving accuracies of up to 0.08 degrees of visual angle (Morimoto & Mimica, 2005) with sampling rates of up to 10 000 Hz (Collewijn, 2001). They work by measuring the electromagnetic inductions in a contact lens placed on the participant’s eye (Duchowski, 2007; Holmqvist et al., 2011). While accurate, there are several disadvantages to scleral coils, however. Firstly, they require that lenses be modelled for each participant individually and these are uncomfortable to wear (Holmqvist et al., 2011). Thirdly, inserting the coil into the eye is a difficult task (Duchowski, 2007) requiring a great deal of practice. Finally, studies have shown that scleral coil systems have an impact on saccadic velocity, which may influence results in studies where peak saccadic velocity is of interest (Träisk, Bolzani, & Ygge, 2005).

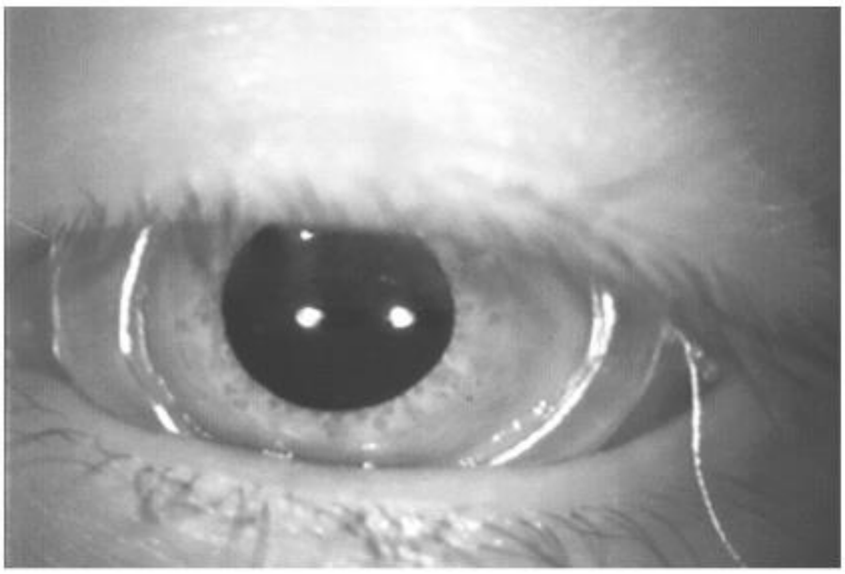


Figure 2.6 - A scleral coil system
Source: http://www.jkma.org/ArticleImage/0119JKMA/jkma-50-343-g003-l.jpg
Retrieved on: 12/06/2014

2.6.2 Electrooculography system

This type of system measures the electromagnetic variation when the dipole of the eyeball musculature moves (Duchowski, 2007; Holmqvist et al., 2011; Morimoto & Mimica, 2005).

EOG systems do have the advantage of very high sampling frequencies of potentially 1000 Hz (Lv, Wu, Li, & Zhang, 2009), but suffer from electromagnetic noise caused by movements in the surrounding muscles (Holmqvist et al., 2011; Joyce et al., 2002). Nonetheless, EOG systems are cheaper than scleral coils and easier to use. For this reason, they are used in clinical trials (Morimoto & Mimica, 2005). They can also be used in situations where the eyes are occluded (Joyce et al., 2002). It is also possible to embed EOG systems into everyday devices such as headphones (Manabe & Yagi, 2014). EOG systems, however, are not appropriate for everyday use and have poor accuracy in comparison to scleral coils – about two degrees of visual angle (Morimoto & Mimica, 2005).

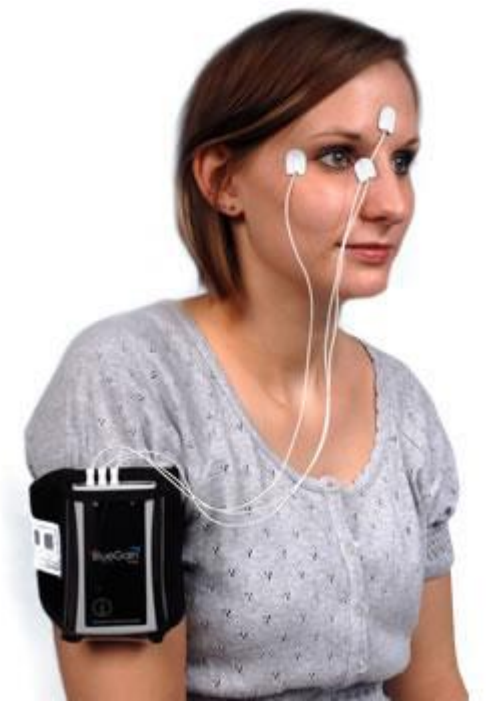


Figure 2.7 - An EOG system
Source: http://www.crsltd.com/assets/Products/BlueGain/_resampled/SetSize350350-bluegain.jpg
Retrieved on: 12/06/2014

2.6.3 Video based eye trackers

Video based eye trackers make use of a camera to measure the eye position without the need to have contact with the users (Hansen & Ji, 2010; Morimoto & Mimica, 2005). The location of the eye in the image is detected and tracked on a frame-by-frame basis. Based on the position of the pupil and other feature points, the point of gaze (PoG) can be estimated (Duchowski, 2007; Hansen & Ji, 2010).

One advantage of a video based eye tracker is that it is comfortable and easy to use in comparison to EOG and scleral coil systems (Morimoto & Mimica, 2005). Additionally, it is less invasive (Duchowski, 2007), more natural to use (Holmqvist et al., 2011), and cheaper than these other systems (Enright, 1998).

In the past, video based eye tracking systems were limited to low sampling frequencies – most of these systems were between 50 and 60 Hz (Duchowski, 2007). However, increases in the speed of computing hardware over the last few years have yielded a number of video based eye trackers that are capable of achieving comparable sampling frequencies to those of EOG and scleral coil systems. The Tobii TX300, SMI RED 500, Eye Link 1000 and SMI IVIEW X Hi-Speed are examples of video based systems that are capable of sampling eye movements at 300, 500, 1000 and 1250 Hz respectively. Mention should, however, be made of the fact that certain of these eye trackers are monocular eye trackers (the SMI IVIEW X for example, has the option of both monocular and binocular tracking). Notwithstanding this, in terms of accuracy, video based remote eye trackers have made great strides. The above mentioned systems all claim accuracies of less than one degree of visual angle (Sensoric Motor Instruments, 2015; SR Research, 2014; Tobii, 2014).

Video based eye trackers do have a few weaknesses. They do not function well in very bright conditions (Hansen & Ji, 2010). Additionally, they require an unobstructed view of the eye, and thus droopy eyelids and downward lashes that partially occlude the eyes can have a negative impact on the performance of the tracker (Holmqvist et al., 2011). Users with lazy eyes or who wear spectacles will also pose challenges (Poole & Ball, 2005).

Video based systems can be divided into two types, namely head mounted (often referred to as wearable) and static trackers (also called table mounted or remote eye trackers).

2.6.3.1 Head-mounted trackers

A wearable eye tracker is one placed on the head of the user (Lemahieu & Wyns, 2010). It contains a minimum of two cameras – one to observe the user’s eyes, and another, called the scene camera, which records the stimulus. The stimulus in this case would be anything the user is looking at (Holmqvist et al., 2011).

Though slightly intrusive, wearable eye trackers are portable and present researchers with the opportunity to conduct eye tracking research in any location (Bulling & Gellersen, 2010; Kim et al., 2014). This allows researchers to observe gaze patterns in natural environments, while participants perform a variety of tasks (Hayhoe & Ballard, 2005; Wade & Tatler, 2010).

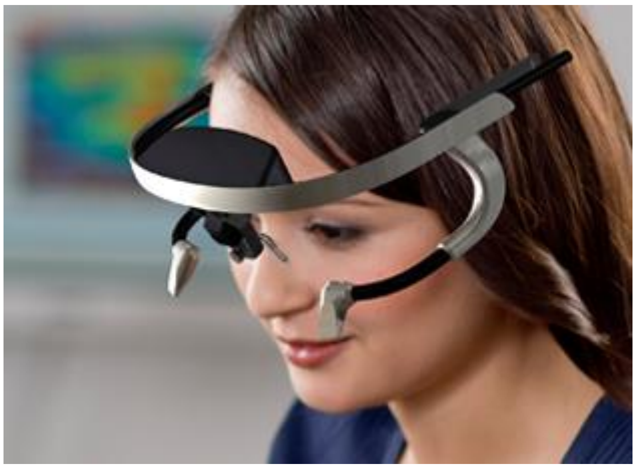


Figure 2.8 - A head-mounted eye tracker
Source: http://www.ergoneers.com//wp-content/uploads/2014/03/hw-et-hm-dikablis-5-314x229.png
Retrieved on: 12/06/2014

2.6.3.2 Static eye trackers

A static (remote) eye tracker is an eye tracker that is placed on the table in front of the participant. For this reason it is also referred to as a table mounted eye tracker (Duchowski, 2007). Static eye trackers consist of tower-mounted and remote eye trackers (Holmqvist et al., 2011). Tower-mounted systems restrict the user’s head movements for the purpose of acquiring more accurate gaze estimations, whilst remote eye trackers sacrifice that accuracy for the sake of a less intrusive setup. Remote eye trackers do not require any equipment to be attached to the user, making them the most likely candidate for general acceptance as an eye tracking interface (Hennessey & Lawrence, & Noureddin, 2006; Lemahieu & Wyns, 2010). As such they are an attractive proposition (Poole & Ball, 2005; Villanueva, Cerrolaza, & Cabeza, 2007).

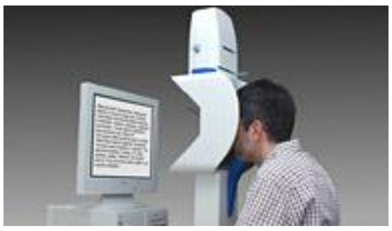


Figure 2.9 - A tower-mounted eye tracker from Sensoric Motor Instruments
Source: http://www.smivision.com/uploads/tx_templavoila/iviewx_hi_speed.jpg
Retrieved on: 12/06/2014

2.6.4 Summary of systems

Based on the sections above, one can identify three predominant types of eye tracking systems, namely scleral coil, electrooculography (EOG), and video based eye trackers. Each of these eye tracking systems has strengths and weaknesses.

Scleral coil systems offer superior accuracy and sampling rate in comparison to the other types, but require great skill to use and are highly intrusive by nature. They also require the custom fitting of lenses to each participant. Scleral coils also impact the peak saccadic velocity, and thus care should be taken when using them with saccadic research.

In contrast, EOG systems are easier and cheaper to use, but less accurate than scleral coil systems and are susceptible to noise. They also have poor accuracy in comparison. They are nonetheless useful in scenarios where the eyes may be partially or fully covered.

Video based systems are cheaper and are less intrusive than either of the above mentioned systems. Remote based systems, for example, have no contact with the user. They are also simpler to operate when compared to the previous two types of eye trackers. Furthermore, advances in mapping algorithms have led to competitive accuracies, whilst advances in computer hardware mean that higher sampling rates are being achieved. However, video-based eye trackers do not work well when exposed to very bright light conditions, and experience issues when confronted with lazy eyes, long lashes or spectacles.

2.6.5 The selected eye tracker

Based on the above mentioned facts, the eye tracker developed as part of this dissertation will fall within the video based eye tracker paradigm. It hopes to combine a high data capturing frequency of 200 Hz with the ease of use of a video based eye tracker. To ensure the eye tracker is as not intrusive, the tracker will be a remote video based eye tracker, as opposed to a head-mounted eye tracker. The non-invasive qualities of a remote eye tracker will also allow the system to be used for longer periods of time (Morimoto & Mimica, 2005), which will prove useful for studies that require lengthy tasks to be performed, as may be the case in reading research, for example.

2.6.6 Conclusion

In this section, the various types of eye trackers have been discussed. The strengths and weaknesses of each technique have been compared and contrasted to determine the most suitable type of eye tracker that should be developed for this dissertation. From the results, an eye tracker type has been selected and the choice of method that will be used has been motivated. In the following section, the components of a video based remote eye tracker will be discussed along with an overview of the eye tracking process as it relates to these components.

2.7 Remote eye tracking mechanics

As the eye tracker that will be developed will fall within the video based remote eye tracking paradigm, it is necessary to explain the inner workings of such a system. A remote eye tracking system typically contains three parts (Hansen & Hansen, 2006). Firstly, the system requires some way with which the eye(s) can be detected. Secondly, it requires a way in which the eyes can be tracked over a period of time, or in other words, a method through which feature points can be identified. Lastly, the system must contain a facility that calculates the user's point of gaze (PoG), i.e. where the user is looking. Each part involves hardware and software components that work together to enable these requirements to be met. Although there are two types of remote eye trackers, namely those that make use of active illumination in the form of infrared (IR) light and those that make use of natural light (Hansen & Ji, 2010; Hansen & Pece, 2005), attention will be devoted to the eye tracking process as it applies to remote eye trackers utilising infrared light.

2.7.1 Hardware components of a remote eye tracker

One of the three components of an eye tracker needs to provide a way in which the eyes can be detected. To do this, hardware components that capture images of the eye are required. As stated in the previous section, video based eye trackers make use of a camera to record an image of the user's eyes. In addition to the camera, remote eye trackers may contain hardware components such as IR light sources. The components involved depend on the type of eye tracking technique used.

In order to further emphasise relevant feature points, remote eye trackers may make use of infrared illumination (Figure 2.10) directed into the user's eyes (Morimoto & Mimica, 2005; Poole & Ball, 2005).

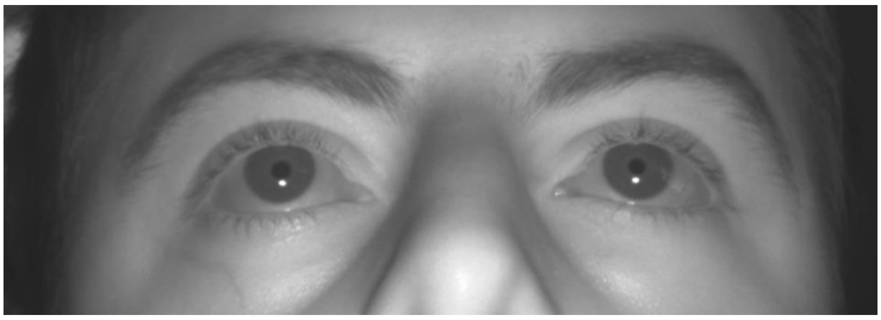


Figure 2.10 – A frame from a typical eye video with a single IR light source

Infrared (IR) light has the advantage that it is invisible to the naked human eye, and does not therefore interfere with the user's vision in any way. As IR light falls on the curved cornea of the eye some of it is reflected back in a narrow ray pointing back at the light source. As illustrated in Figure 2.11, several reflections occur on the various layers of the eye, producing what is referred to as Purkinje images. The first of these reflections is called a glint, or corneal reflection (Hansen & Ji, 2010; Holmqvist et al., 2011; Poole & Ball, 2005). A further advantage of IR is that the lighting and image exposure levels are controllable (Li & Parkhurst, 2006).

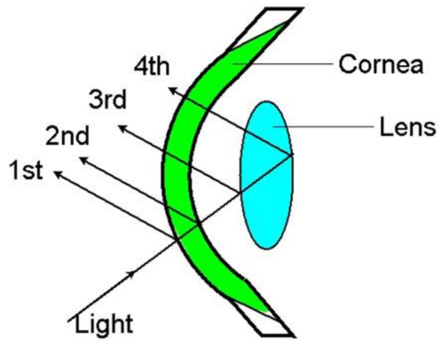


Figure 2.11 - An image depicting the four Purkinje images
(Source: Morimoto and Mimica, 2004)

Infrared light can also lead to what is referred to as the bright eye, or bright-pupil, effect when the light is positioned near the optical axis of the camera (Ebisawa, Ohtani, Sugioka, & Esaki, 1997; Hutchinson, White, Martin, Reichert & Frey, 1989). It is possible to further emphasise the pupil by alternating between bright and dark pupil images (Ebisawa, Ohtani, Sugioka, & Esaki, 1997; Hennessey et al., 2008; Zhu & Ji, 2005b). An example of this can be seen in Figure

2.12 below, where (A) represents the bright pupil image, (B) the dark pupil image, and (C) the difference image of the previous two. Note that the pupils are now the only remaining bright points in the image, thereby greatly simplifying the task of locating the pupils.

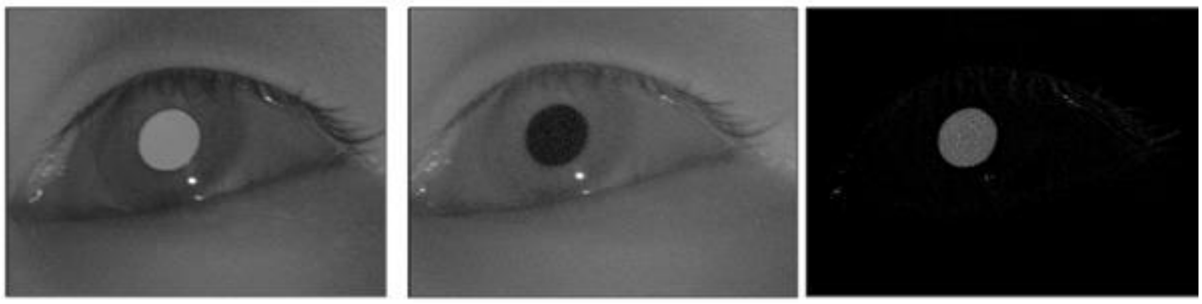


Figure 2.12 – (a) A bright pupil image (b) A dark pupil image (c) The result when (b) is subtracted from (a)
(Source: http://www.intechopen.com/source/html/6679/media/image1.png
Retrieved on: 19/01/2015)

Not all eye trackers make use of infrared illumination and instead make use of natural light. These eye trackers can have simpler hardware requirements than active illumination systems as a result. A natural eye tracker can, therefore, be constructed using a single camera such as an affordable webcam (Hansen, Hansen, Nielsen, Johansen, & Stegmann, 2002; Hansen & Mackay, 2004).

In contrast, the hardware setups of active illumination eye trackers require multiple hardware components. At the core, IR based eye trackers all contain a camera and an IR light source (Guestrin & Eizenman, 2006; Holmqvist et al., 2011). Variations of this setup include eye trackers that make use of multiple cameras and infrared lights, as well as mirrors and ultrasonic range finders (Beymer & Flickner, 2003; Coutinho & Morimoto, 2006; Yoshinobu Ebisawa, Ohtani, Sugioka, & Esaki, 1997; Noureddin, Lawrence, & Man, 2005). Examples of these variations will be discussed in Chapter 3, Section 3.2.

When designing an eye tracker, one must take into account the type of camera that is used. The slowest eye trackers generally run at 25 Hz (Holmqvist et al., 2011), so one's selected camera should be capable of at least that. Another consideration is accuracy. A high resolution camera is required to accurately estimate the PoG (Beymer & Flickner, 2003; Hennessey, Lawrence, & Noureddin, 2006). A final consideration is the size of head movements that must be tolerated.

For large head movements, it may be necessary to make use of a camera with a wide angle lens (Nagamatsu, Kamahara, & Tanaka, 2009; Noureddin et al., 2005).

2.7.2 Software components of a remote eye tracker

With the eye tracker hardware in place, there is now a way with which the eyes can be recorded. However, the positions of the feature points (see Figure 2.13) in the image still need to be determined. This is accomplished utilising an algorithm implemented as a software solution.

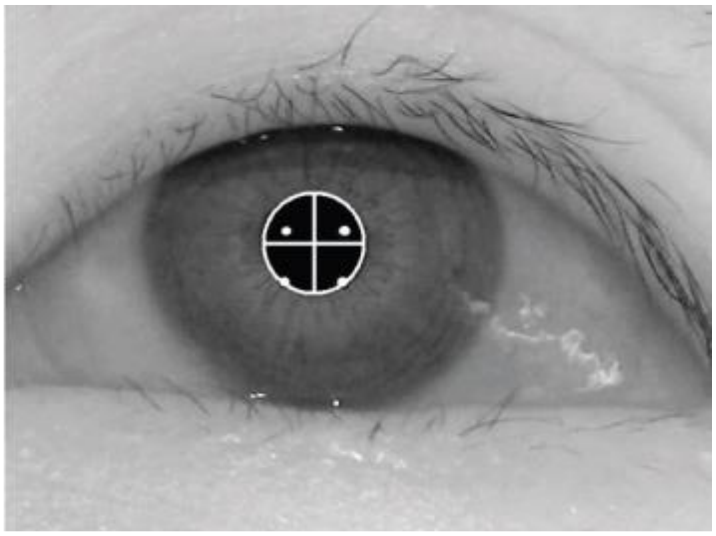


Figure 2.13 - Eye image with feature points correctly located
(Source : Gwon, Cho, Lee, Lee, & Park, 2013)

There are several algorithms available for extracting feature points such as the pupils and corneal reflections (Ebisawa, Ohtani, Sugioka, & Esaki, 1997; Lemahieu & Wyns, 2010; Li & Parkhurst, 2005; Li, Qi, & Wang, 2001; Mulligan, 1997; Ryan, Duchowski, & Birchfield, 2004; Zhu & Ji, 2005b). An overview of the various methods is given in a later section (Section 2.10). The software implementation should be able to perform the feature extraction in real time (Li, Babcock, & Parkhurst, 2006), as well as compute the PoG. The need for real-time feature extraction implies that the feature detection methods be performed as quickly as possible. One possible solution would be to make use of specialised hardware for this task. As will be discussed in the following chapter (Chapter 3), one example of such a piece of hardware is the Graphical Processing Unit.

2.7.3 Gaze estimation

Gaze estimation is the process whereby the user's point of regard is determined (Cerrolaza, Villanueva, & Cabeza, 2008) and makes up the third and final component of an eye tracker. The point of regard is also sometimes referred to as the point of gaze (PoG) (Nagamatsu, Iwamoto, Kamahara, Tanaka, & Yamamoto, 2010). Video based remote eye trackers make use of feature points such as the pupils, limbus and corneal reflections to perform gaze estimation (Cerrolaza, Villanueva, Villanueva, & Cabeza, 2012; Lemahieu & Wyns, 2010; Morimoto & Mimica, 2005). The POR is calculated relative to the eye tracker's stimulus's size in pixels (Duchowski, 2007). There are various ways in which the POR is calculated. A discussion of the various mapping techniques is presented in Section 2.10.

2.7.4 Calibration

Due to the unique characteristics of individual participants, it is necessary to calibrate each eye tracker setup for each participant (Hansen & Ji, 2010; Hornof & Halverson, 2002). A typical calibration procedure requires the user to look at a number of predefined points on the stimulus (Holmqvist et al., 2011; Lemahieu & Wyns, 2010). The number of points varies, but common values include 2, 5, 9, 13 and 16 points. The points should cover the area the stimulus represents (Holmqvist et al., 2011). The point displayed may be animated to further draw attention to it and ensure that the user looks at the point (Duchowski, 2007). During calibration, the state of the feature points in the eye video – such as the pupil – are analysed, and a mapping model constructed (Duchowski, 2007). The selected features that are analysed will depend on the model that is being used. For instance, in a regression-based gaze mapping model, the pupil-glint vector (P-CR) is examined at each calibration point to determine the coefficients of the mapping polynomial through regression. A higher number of calibration points can potentially lead to an increase in the accuracy of the gaze estimation procedure at the expense of a longer calibration routine and more complicated mapping polynomial (Cerrolaza et al., 2008).

2.7.5 A summary of the remote eye tracking process

The entire eye tracking process can be summarised as follows. A camera positioned in front of the user observes the user's eyes. The resulting image – referred to as the eye video – is processed and feature points such as the pupil and corneal reflections are extracted. Following a calibration procedure, these features are utilised to determine the PoG of the user (Lemahieu & Wyns, 2010). As the focus in this dissertation will be on developing an active illumination remote eye tracking system, the eye tracking process for a natural light eye tracker will not be discussed. Thus, from this point on remote eye tracking will refer primarily to the use of IR-based remote eye trackers.

2.7.6 Conclusion

In this section, the mechanics of a video based remote eye tracker have been discussed as they relate to a remote video-based eye tracker – the development of which is the goal of this dissertation. The various components of an eye tracker were discussed, and the role of infrared illumination and corneal reflections in the eye tracking process highlighted. Finally, the way in which the PoG is calculated was briefly described. In the proceeding section, the two aspects of remote eye tracker research (gaze estimation and feature detection) will be discussed. This dissertation will focus on feature detection. Nonetheless, gaze estimation is discussed to identify the feature points required for the eye tracker to be capable of performing gaze estimation.

2.8 Remote eye tracker research

The three eye tracking components described in the previous section can be divided into two steps (Blignaut, 2013; Hansen and Ji, 2010). The first step is to detect the eyes and relevant feature points in the eye video. The second step is to make use of those features to determine where the user is looking (calibration and gaze estimation). It follows then that researchers involved in eye tracking are interested in improving each of these two steps. Specifically, researchers are interested in finding more robust, accurate and precise mapping techniques with which to estimate the PoG, as well as finding better and faster ways to identify relevant feature points in the eye video.

2.9 Eye models and gaze estimation

There are a number of mapping models available with which the point of gaze (POG) can be computed. For the purpose of this dissertation, only models that make use of a feature-based approach will be discussed, given that the eye tracker developed will make use of this approach. However, other methods do exist. One example of such a method is an appearance-based approach illustrated by Liang, Tijus, Chahir, Jouen, and Molina (2013) which maps feature points directly to gaze coordinates.

When one considers feature-based mapping approaches, there are two categories that can be identified, namely model-based (also known as geometric) and interpolation-based (regression) methods (Hansen & Ji, 2010). The type of model that can be used will also depend on the physical eye tracker setup. For instance, the particular model may require stereo cameras or the presence of corneal reflections. These hardware components will therefore need to be present for the model to be usable.

2.9.1 Regression-based methods

Regression based methods generally consist of polynomial expressions that can be used to calculate the PoG based on the state of relevant features such as the pupil and corneal reflections observed in the eye image (Cerrolaza et al., 2008). A popular and widely used regression method is the Pupil-Corneal Reflection (P-CR) technique (Zhu & Ji, 2005a), in which the polynomial uses as input the x and y coordinates in pixels of the pupil-corneal vector observed in the eye video.

The coefficients of the polynomials in a P-CR model are determined using data obtained during the calibration procedure. Regression is performed on observed feature points observed during the calibration at the time each dot was shown. Consider the following polynomial that estimates the x component of the PoG:

$$PoG_x = D_{x0} + D_{x1}x + D_{x2}x^3 + D_{x3}y^2$$

In the polynomial shown above, D_{xn} represents the coefficients to be determined through calibration whilst x and y represent the components of the observed P-CR vector. The regression formula for this polynomial may look similar to the equation below.

$$\begin{bmatrix} D_{x0} \\ D_{x1} \\ D_{x2} \\ D_{x3} \end{bmatrix} = \begin{bmatrix} n & \sum x & \sum x^3 & y^2 \\ \sum x & \sum x^2 & \sum x^4 & \sum xy^2 \\ \sum x^3 & \sum x^4 & \sum x^6 & \sum x^3y^2 \\ \sum y^2 & \sum xy^2 & \sum x^3y^2 & \sum y^4 \end{bmatrix}^{-1} \begin{bmatrix} \sum PoG_x \\ \sum x\, PoG_x \\ \sum x^3\, PoG_x \\ \sum y^2\, PoG_x \end{bmatrix}$$

For the purpose of the regression, the PoG components are taken from the location in pixels of each dot that was displayed during calibration. The location of each of these dots is known beforehand.

The performance of the regression based method is very much dependent on the image features used, and the degree and number of terms of the polynomial expression used to calculate the PoG, as well as the calibration procedure used (Blignaut, 2013; Cerrolaza et al., 2008). One weakness of this model is that it tends to be sensitive to head movements (Zhu & Ji, 2005a). It is however, very simple to implement, and very little information is needed regarding the placement of the hardware components, as most of the necessary information can be determined during the calibration process.

2.9.2 Geometric-based gaze estimation

Geometric, or model-based gaze estimation makes use of the physical structure of the human eye (Figure 2.14) to calculate the 3D gaze direction (Hansen & Ji, 2010). These models make use of the visual and optical axis of the eye to determine the PoG (Guestrin & Eizenman, 2006; Hansen & Ji, 2010). The visual axis of the eye consists of a vector that runs through the centre of the pupil and joins with the fovea close to the centre of the cornea (Hennessey, Lawrence, & Noureddin, 2006; Villanueva, Cabeza, & Porta, 2007). The visual axis is assumed to be the true point of gaze.

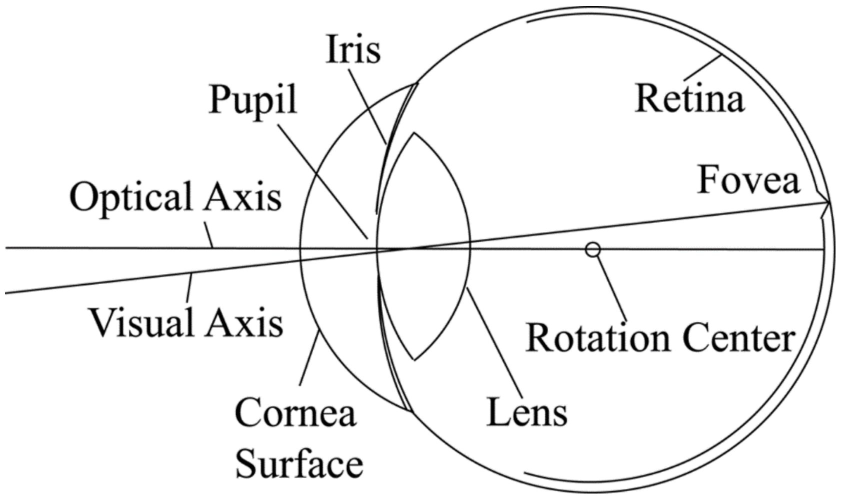


Figure 2.14 - Eye model
(Source: Ohtera, Horiuchi, & Tominaga, 2009)

As each eye is different, it is necessary to calculate the parameters of the visual axis for each person. Using the optical axis calculated from the centre of the pupil and corneal reflections, the visual axis can be constructed and the PoG calculated (Guestrin & Eizenman, 2006).

Unlike the regression-based systems, geometric models require detailed information about the positioning of the hardware components of the eye tracker (Hansen & Ji, 2010). As such, they are best suited for fixed systems that can be set up once, and left untouched thereafter. These systems are more tolerant towards head movements.

2.9.3 Conclusion

Both methods make use of feature points such as pupils and corneal reflections. Indeed, Guestrin and Eizenman (2006) state that these two features have been used for many years. With this in mind, it follows that an active system needs to be capable of accurately identifying these two feature points. While a study involving different estimation methods lies beyond the scope of this dissertation, it is worth noting that the system developed is theoretically capable of both models, as it locates the centre of the pupil and corneal reflections.

2.10 Feature detection methods

Detection of the human eye is not an easy task. Hansen and Pece (2005) state that there are numerous factors that make the detection of the eye a difficult process. For one, the pupil may be occluded by the eye lids. Furthermore, there are variations in ethnicities that need to be accounted for, such as the prominent nose features prevalent in Caucasians – a feature that is known to cast shadows over the eyes. There are two parts to eye detection. One is determining the existence of the eyes in the eye video, and the other to accurately locate the position of the eyes (Hansen & Ji, 2010).

Eye position is commonly measured using the pupil centre (Wang, Green, & Ji, 2005). However, there are a number of feature points that can be used. The three most commonly tracked features are the pupil, the iris and the sclera (Li et al., 2006). While it is possible to perform gaze estimation using only the pupil, adding additional features, such as corneal reflections, can improve the gaze estimation (Holmqvist et al., 2011). As mentioned earlier, one commonly used variation of the feature-based method is to include active infrared (IR) illumination to generate these corneal reflections.

Hansen and Ji (2010) broadly define four classes of eye detection methods. These four approaches are shape-based, feature-based, appearance-based and a hybrid approach consisting of a combination of the previous three. Each of these four approaches has strengths and weaknesses. The following sub-sections provide a short discussion of the four techniques, with a comparison of their relative strengths and weaknesses. By considering the strengths and weaknesses of each approach, a suitable technique can be selected to achieve the high sampling rate goal of the eye tracker that will be developed.

2.10.1 Shape-based approach

The first of the techniques, referred to as the shape-based approach, relies on the fact that an open eye has a distinctive shape, such as an elliptical pupil. Shape-based methods generally consist of two components, namely, a geometric model and some way of calculating the similarity between detected features and the model used (Hansen & Ji, 2010). The geometric model will decide whether the eye is modelled as an ellipse or circle, or even as two parabolas. One example is the model proposed by Yuille, Hallinan and Cohen (1992) which makes use of a deformable template. The template consists of parameters that can be adjusted to match the participant’s eyes. In this specific instance, the eye is modelled as two parabolas.

The shape-based approach does have several advantages. It allows for variations in the scale, tilt and rotation of the head, and is tolerant towards lighting conditions. One weakness of this approach, however, is that it is sensitive to an initial search point. For shape-based methods to be effective, this initial point must be located close to the eye. It is not uncommon for shape-based methods to mistake the eyebrows for the eyes. For this reason, shape-based methods may be unsuitable for eye tracking systems that require support for large head movements (Hansen & Pece, 2005). A further disadvantage of this method is that it is computationally expensive (Hansen & Ji, 2010).

2.10.2 Feature-based approach

Feature-based techniques exploit the fact that the human eye has certain characteristics. For example, there is a distinct contrast between the edge of the iris and the rest of the eye. One can also look for dark points in the image, as the pupil is typically very dark (Hansen & Ji, 2010). Systems that employ active illumination, such as IR light, can utilise bright and dark pupil images in combination with each other. As mentioned earlier in the chapter, alternating between bright and dark pupil images provides an effective method of identifying the location of the pupil in the image (Ebisawa, Ohtani, Sugioka, & Esaki, 1997; Morimoto, Koons, Amir & Flickner, 1998).

Feature-based methods, particularly those that use active illumination, are simple to implement and present little difficulty in identifying feature points such as the corneal reflections (Hansen & Ji, 2010). However, they also suffer when confronted with false positives. As with the shape-based approach, one of the pitfalls is mistaking eyebrows for eyes (Peng, Chen, & Kukharev, 2005). Consequently, it may be necessary to add some form of filtering to remove these false positives (Hansen & Ji, 2010). Feature-based methods also suffer when eyelids are partially closed, as the pupil centre may be offset (Holmqvist et al., 2011) leading to inaccuracies in the gaze mapping. Systems that rely on corneal reflections also need to be wary of the fact that the corneal reflections can be lost when viewing the eyes at extreme angles. Lastly, while feature-based systems work well indoors, they become poorer outdoors, as the pupils become smaller in bright environments, or can become masked by specular reflections caused by other light sources (Hansen & Ji, 2010; Ryan, Duchowski, & Birchfield, 2004).

2.10.3 Appearance-based approach

This technique matches the shape of the eye to a template image. The appearance of eye regions share commonalities across race, illumination and viewing angle. Thus, a large set of training examples with various poses and lighting conditions can be constructed (Hansen & Pece, 2005). These techniques can be useful for initialisation and validation of eye locations (Hansen & Pece, 2005). The method employed is independent of the actual object being tracked and can be used to track objects other than the eyes (Hansen & Ji, 2010).

2.10.4 Hybrid approach

The hybrid approach is a combination of the three previously discussed approaches. Based on existing literature, it can be concluded that a substantial number of eye tracker implementations make use of a combination of techniques, rather than just a single one. When one considers the relative strengths and weaknesses of each technique, it makes sense to attempt to mitigate weaknesses in one technique by applying a different technique in its place. One example of such an approach is the eye tracker developed by Zhu and Ji (2005b). In this example, a feature-based method is combined with template matching. A feature-based method utilising alternating bright and dark pupil images is used to identify pupil candidates. The candidates are then filtered using a Support Vector Machine (SVM) trained with a data set of eye images and false positives. Further examples of existing implementations will be discussed in more detail in the following chapter.

2.10.5 Chosen method

The solution developed as part of this dissertation will make use of a feature-based detection method. The reason for this approach is the low operational cost resulting from the simplicity of a feature-based method, which is important when developing a high-speed system. The eye tracker will also typically be located in a laboratory setting, and not outside. This means that the lighting issues that are typically associated with feature-based methods will not be a concern. Moreover, the simplicity of the implementation also lends itself to a GPU based implementation, which is the focus of the next chapter.

2.11 Conclusion

In this chapter, the reader has been familiarised with the field of eye tracking in general. The chapter has provided an overview of the history of eye tracking, to identify the direction in which the field of eye tracking is moving. Next, the way in which an eye tracker is utilised was discussed, along with the methods used to analyse eye tracking data. This was done to identify usability requirements for the eye tracker developed in this dissertation. Additionally, the data quality measures used to evaluate an eye tracker's performance were described and the measures that will be used in the evaluation of the developed eye tracker were selected. Finally, the mechanics of a video based remote eye tracker were discussed, with specific focus on the process of feature-point detection – the focus of this dissertation.

In the following chapter, an analysis of existing work in the field of low-cost eye tracking is done and the role of the Graphical Processing Unit in the feature detection process of the envisaged eye tracker will be discussed.

CHAPTER 3: DISCUSSION ON EYE TRACKING APPLICATION

The previous chapter discussed the general theory of eye tracking, and identified the type of eye tracker to be developed as a remote eye tracker. As stated in the first chapter, the goal is to utilise the Graphical Processing Unit (GPU) to develop an eye tracker that is capable of achieving a sampling rate in excess of 200 Hz, while providing acceptable values for precision, accuracy and trackability. In this chapter, the technical details of the eye tracker that was developed to meet these requirements are discussed. An overview of existing work is provided and certain shortcomings that were identified are addressed. In addition hereto, the technical requirements for the eye tracker are listed and appropriate hardware and software solutions are selected. For the system requirements, software architecture and additional source code examples demonstrating the use of the developed eye tracker, refer to Appendix A.

3.1 Introduction

Over the last decade, the academic community has been engaged in developing eye trackers that employ novel methods to detect features and perform gaze mapping. Large strides have been made in reducing the cost of eye tracking systems and improving their accuracy. Solutions have been generated using inexpensive commercial off-the-shelf components (San Agustin et al., 2010). Some of these systems are capable of achieving accuracies of around one degree (Böhme, Meyer, Martinetz, & Barth, 2006; Hennessey, Lawrence, & Noureddin, 2006). However, many of these solutions focus on gaze interaction or usability studies and as a consequence have relatively low sampling rates. The following section will discuss a selection of these eye tracking solutions. By analysing existing work, it is possible to determine shortcomings that can be addressed in this study.

3.2 Existing work

There are numerous examples of eye tracker systems developed during the past decade. A discussion of all of the existing systems, however, lies beyond the scope of this dissertation. Nonetheless, there are several examples in the literature that address the issues of cost, accuracy, head movement and sampling rate (Böhme et al., 2006; Clarke, Ditterich, Drüen, Schönfeld, & Steineke, 2002; Hennessey, Noureddin, & Lawrence, 2008; Lemahieu & Wyns, 2010; Mulligan, 1997; Noureddin, Lawrence, & Man, 2005; San Agustin et al., 2010). The purpose of this section is to provide examples of some of the systems that were developed recently, and to use these to identify shortcomings that need to be addressed in the envisaged eye tracker.

3.2.1 Head movement

Noureddin, Lawrence and Man (2005) proposed a two-camera setup with a rotating mirror (Figure 3.1). The purpose hereof was to provide tolerance for head movements. To this end, one of the cameras contained a wide-angle lens that tracked the position of the pupil and glints. A frame containing a narrow-angle camera and mirror rotated in accordance with the position of the eyes to ensure that they were always visible. The mirror reflected eye images into the narrow-angle camera. The results of this study indicated that the system was capable of tracking the pupil and glint in real time at 9 fps even in the presence of head movements. Noureddin et al. (2005) reported an average gaze accuracy of 2.9 degrees of visual angle for their system.

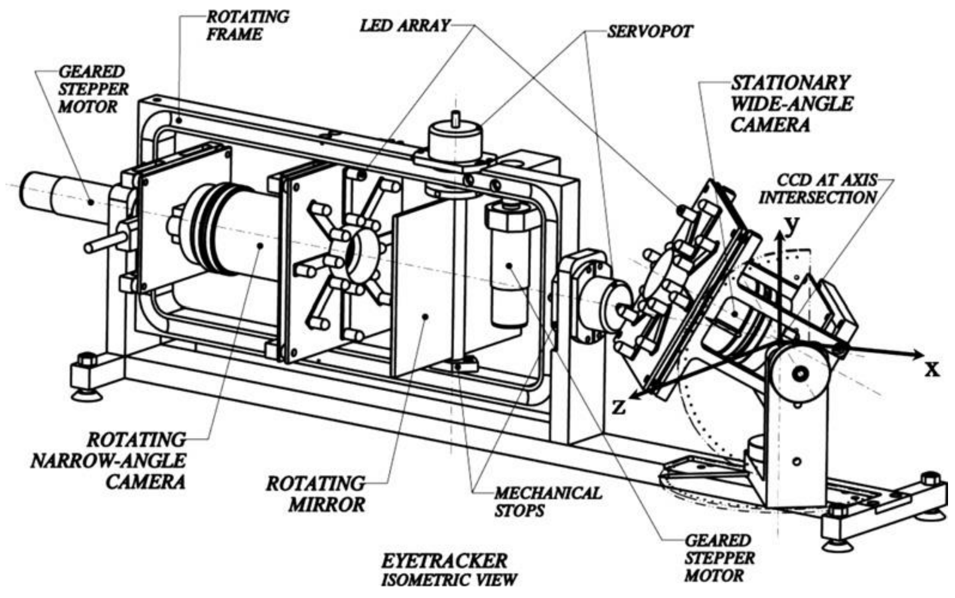


Figure 3.1 - The eye tracker setup of Noureddin et al.
Source: (Noureddin et al., 2005) pg.5

Hennessey, Lawrence and Noureddin (2006) developed an eye tracking solution with good tolerance towards head movements (Figure 3.2). Their system made use of a 3D gaze mapping technique utilising multiple glints. Unlike traditional feature-based systems utilising corneal reflections where the infrared (IR) lights are placed below the screen, the system had the lights mounted vertically on the right side of the screen. Another set of IR LEDs were mounted on the camera's optical axis to generate bright pupil images. The system allowed head movements within an area of 14×12×20 cm and was capable of achieving accuracies of less than 1° of visual angle. The study was conducted at 30 Hz, though the authors cite the camera as the reason for the low sampling rate.

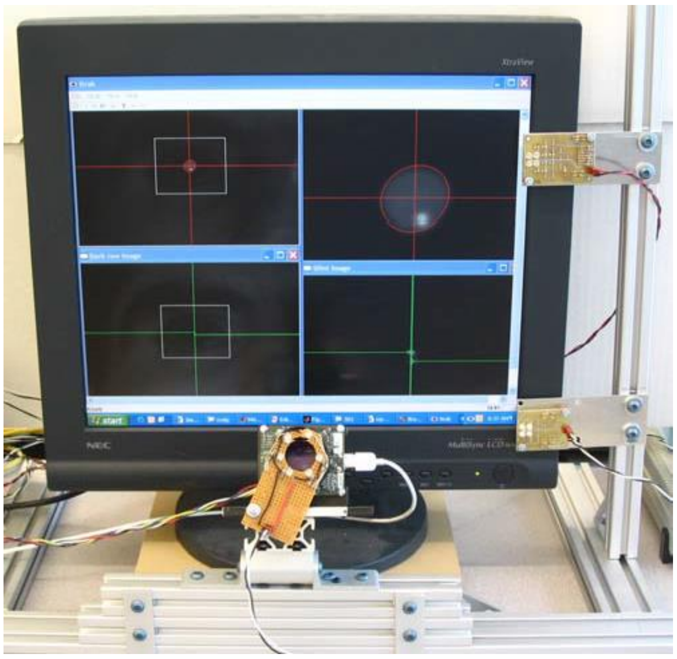


Figure 3.2 - The eye tracker of Hennessey and Lawrence.
Source: (Hennessey, Noureddin & Lawrence, 2006) pg. 6

3.2.2 Cost

San Agustin *et al.* (2010) developed the ITU gaze tracker. This open-source system allows developers to make use of cheap hardware such as a web camera and IR lights to construct an eye tracker. The open source nature of the project allows for a large amount of customization in terms of the type of eye tracker. With small modifications, one can create even a wearable eye tracker using the ITU software (San Agustin et al., 2010). An image of the ITU user interface is shown in Figure 3.3 below.

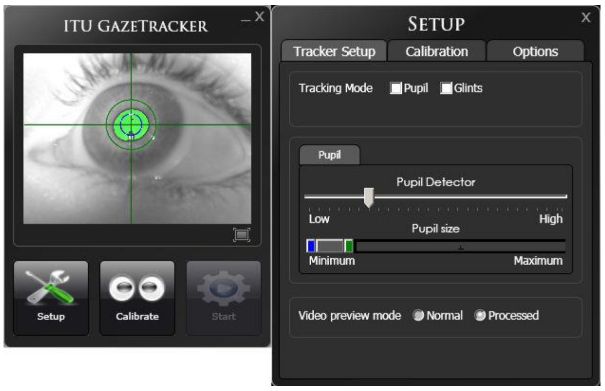


Figure 3.3 - User interface of the ITU open-source tracker.
Source: (San Agustin et al., 2010) pg. 2

Another example of a low cost solution is the eye tracker developed by Lemahieu and Wyns (2010). Their system makes use of a webcam with a resolution of 1024×768 pixels, a laptop and IR light sources. To improve the accuracy, the user's head is kept still using a simple frame. The accuracy obtained in their study was 1.5° of visual angle. Böhme et al. (2006) created a similar system (Figure 3.4) which allowed for head movements of up to 20 cm on all three axes. They obtained an average accuracy of 1.2°. Both of these systems run at under 30 Hz.

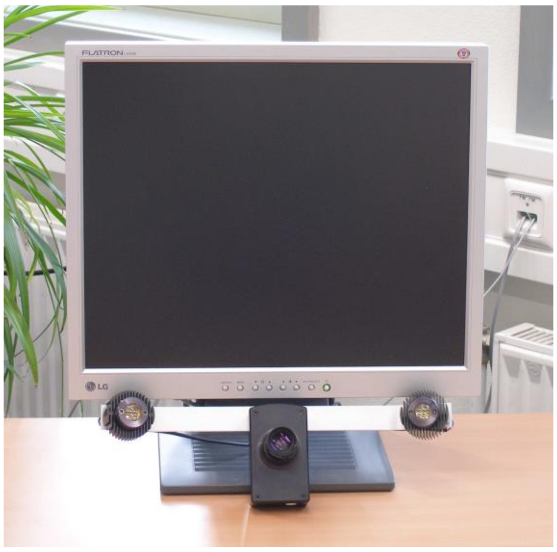


Figure 3.4 - The low-cost eye tracker of Böhme et al.
Source: (Böhme et al., 2006) pg. 2

3.2.3 Sampling rate

While the systems above generally work at 30 Hz or lower, there are systems that have reported higher sampling rates. Hennesey et al. (2008) present a technique with which sampling rates of as high as 407 Hz can be obtained. By utilising a series of hardware and software regions of interest (ROIs), they minimise the amount of data that must be processed, thereby increasing the obtainable sampling frequency. While the accuracy for their system is not reported, it does obtain an unfiltered average precision of 0.347° of visual angle.

More recently, Mulligan (2012) has shown that it is possible to obtain accuracies of 0.5° of visual angle while making use of hardware acceleration to increase the sampling rate of the system to 250 Hz for images as large as 640×480 pixels. By utilising the NVidia CUDA and the Graphics Processing Unit (GPU), it is demonstrated that certain aspects of the eye tracking process can be performed on the GPU at significantly faster speeds than on the CPU. The current study will build on the Mulligan study by expanding it to other types of GPU enabled cards, experimenting with various image sizes and testing the application extensively.

3.2.4 Shortcomings that will be addressed

The primary issue that will be addressed in the study is the relatively low sampling frequency that many of the systems discussed in the preceding section support. While it is possible that these systems are capable of higher sampling frequencies, they have not been tested at these frequencies. Thus, one of the goals of this research study is to ascertain whether or not the sampling frequency has an effect on the data quality of a P-CR (Pupil-Corneal Reflection) system, as well as report on the actual values obtained by the system at higher frequencies.

Few of the existing systems studied report the precision and trackability obtained. However, in order for a system to be effective and accepted by users, it is necessary to show that it is sufficiently accurate for its tasks (Hennessey et al., 2006). Moreover, certain tasks require both a high accuracy and precision (Holmqvist et al., 2011). The evaluation of the envisaged eye tracker should therefore report as much about data quality as is possible within the time constraints.

Finally, the system to be developed should strive to be easy to set up and use, and to be compatible with a wide range of computer systems. It should therefore be capable of running on mid-range computer systems.

3.2.5 Conclusions

Several examples of different P-CR (Pupil-Corneal Reflection) eye tracker systems developed by the academic community have been discussed. From the existing work, a number of shortcomings have been identified that will be addressed.

The following section outlines the technical requirements that must be met in order for the eye tracker to address the shortcomings.

3.3 Technical requirements

For a remote eye tracker to be effective and accessible, there are several issues that need to be attended to. To be widely accepted, the system should be accurate (Section 2.4.1), precise (Section 2.4.2), and affordable, as well as offer a fair amount of tolerance towards head movements without being intrusive (Böhme et al., 2006; Sesma-Sanchez, Villanueva, & Cabeza, 2014) . It has previously been mentioned that a remote eye tracker is the most suitable choice of eye tracking system, and will be capable of meeting with the objectives. The system to be developed will therefore require infrared light sources and one or more cameras for the eye video.

The purpose of this section is to address the issues of cost, data quality and tolerance towards head movements and explain how each of these issues will be resolved.

3.3.1 Cost

The cost of the system needs to be kept as low as possible. One way to accomplish this is to make use of hardware that is widely available, such as web cameras. However, these cameras lack the required refresh rates needed for high-speed applications. Moreover, they lack the required resolution needed for a precise system. It is clear, therefore, that it will not be possible for a high-speed system to be constructed from cheap commercial off the shelf components.

There are cameras that can still be considered affordable when compared to the cost of a commercial eye tracking system. A camera for 500 euros may not be as cheap as a webcam, but is still more affordable than a 20,000 euro eye tracker.

3.3.2 Sampling rate

If the developed eye tracker system is to move away from the low frequency (60 Hz) range, it will require hardware and software capable of running at higher frequencies. It is also the goal of this study to test the system at higher frequencies, rather than develop a theoretical solution. The hardware should therefore be capable of running at a variety of different frequencies so that the actual performance of the system at these frequencies can be observed. This is an important consideration when selecting the camera, as the bandwidth of a USB 2.0 camera may be insufficient for this task. The use of USB 3.0 may be sufficient, but will potentially drive up the overall cost of the system, particularly if a USB 3.0 expansion card is required by the host computer.

3.3.3 Tolerance towards head movement

To accommodate head movement, the use of two cameras or one camera with a wide-angle lens, is required. Making use of two cameras drives up the cost of the system, so the simplest solution would be to fit the camera with a lens that provides enough tolerance for head movements. However, this creates a further complication as the resolution of the eye video has an effect on the error obtained in the measurement of feature points (Hansen & Pece, 2005; Sesma-Sanchez et al., 2014). A balance must therefore be obtained between the size of the head box and the resolution of the eye video.

3.3.4 Precision and accuracy

It has been demonstrated that single camera systems are capable of achieving relatively good accuracies. Blignaut (2013) and Lemahieu and Wyns (2010) both demonstrate single camera setups that achieve accuracies of 1° and 1.5° of visual angle respectively. In terms of precision, Hennesey et al (2008) has shown that under the right circumstances P-CR systems can achieve relatively good precision (~0.2°) at frequencies as high as 386 Hz, with little in the way of filtering in the form of smoothing (as described in Chapter 2, Section 2.4.2) over a period of time. It stands to reason, therefore, that the proposed system can achieve similar results.

The following section discusses the Graphics Processing Unit (GPU) and technologies that can be used to utilise this powerful piece of hardware.

3.4 The Graphics Processing Unit

To meet the requirements stipulated in the previous section, suitable technology needs to be utilised. This ranges from selecting hardware to implementing a software solution. It has been discussed in the previous chapter that part of the eye tracking process involves the processing of eye images to extract relevant feature points. Mulligan (2012) has demonstrated that it is possible to accelerate aspects of the image processing phase by moving them to the GPU using NVidia’s CUDA. Algorithms such as canny edge detection have also been implemented using GLSL (OpenGL® Shader Language) and run faster on GPUs (Roodt, Visser, & Clarke, 2007). The purpose of this section is to provide background on the modern GPU and motivate its inclusion in the envisaged eye tracker system, as well as provide information on the various APIs available with which to program the modern GPU.

3.4.1 Evolution and development of the GPU

The modern GPU can trace its roots back to the 1980’s. In 1981 IBM created the first video card available for the PC (Crow, 2004). It was capable of outputting monochrome text only, and was incapable of addressing a single pixel. Later video cards added the ability to work with higher resolutions and larger amounts of colours, and were capable of manipulating individual pixels. However, these video cards were still limited as they relied on the CPU to do most of the work. The result of this dependency was that the CPU ended up doing even more work than before. To remedy this situation, IBM created a processor based video card, which contained the Intel 8088 microprocessor called the Professional Graphics Controller (IBM, 1984). This adapter performed all the video processing computations and freed up the CPU to do other work. It was this step that set the standard for all further video cards. Future video cards would all contain processors.

Later developments included the idea of a graphics pipeline (Crow, 2004; McClanahan, 2010). This pipeline resembled a manufacturing line, where results of previous steps were used as inputs, processed, and then passed to the next step. The pipeline simplified the process of programming for the graphics hardware, as it converted graphics data from a format convenient for the programmer, to that usable by the display hardware. Initially, this pipeline was not exposed to programmers. Thus, once the graphics data entered the rendering pipeline, the programmer had no control over what was rendered. This type of pipeline was referred to as a fixed function pipeline (Crow, 2004).

Modern graphics cards have shifted away from the fixed function pipeline, however, and have been replaced with a programmable pipeline (Dokken, Hagen, & Hjelmervik, 2007). In 2002, the NVIDIA GeForce FX family of graphics cards gave programmers the ability to write customized programs to operate on the vertex and pixel levels (Fernando & Kilgard, 2003). This allowed programmers to start writing creative bits of code to manipulate graphical data to achieve a number of interesting effects, such as better lighting effects and shadows. These programs, referred to as shaders (Dokken et al., 2007), could be implemented using graphics APIs such as OpenGL® or Microsoft’s Direct3D. The shaders could be written in HLSL (High Level Shader Language), GLSL (Open Graphics Library Shader Language) or NVIDIA’s Cg programming language (Fernando & Kilgard, 2003). Recent advances in GPU architecture have increasingly opened up more of the pipeline to programmers.

A typical graphics hardware pipeline consists of the following steps: vertex transformation, primitive assembly and rasterization, fragment texturing and colouring, and finally, raster operations (Dokken et al., 2007; Fernando & Kilgard, 2003). Initially, GPUs were responsible only for the final stage of this pipeline. However, as they evolved, more and more stages of the pipeline were transferred to the GPU. The advantage of moving the pipeline over to the GPU was that it freed up the CPU to do more geometry and application calculations. This in turn translated into more graphical data that could be sent to the GPU to be drawn, allowing programmers to start drawing extremely detailed scenes in real time.

3.4.2 Architecture of the modern GPU

To understand why modern GPUs have become so powerful, it is necessary to take a look at their architecture. Whereas CPUs are instruction driven, GPUs make use of streams (Dokken et al., 2007). The advantage of stream processing is that it allows simple computations to be performed very fast and in parallel. As graphics computations are generally very simple and require very little memory, the stream processing model is the ideal choice (Crow, 2004).

One can think of a stream as an ordered set of data of the same data type. In terms of graphics processing, this stream is typically a large collection of vertices. The stream in turn passes through a series of kernels. The kernels process each element in the stream and produce one or more streams as output. Each kernel performs the same set of instructions on each element in the stream. Thus, many of these instructions can be performed in parallel. This model of processing is referred to as SIMD, which stands for Single Instruction, Multiple Data (Dokken et al., 2007; Geldhof, 2007). Modern GPUs are capable of supporting thousands of stream processors.

If one considers the massive parallel potential for image processing functions thanks to the SIMD architecture, coupled with a CPU unburdened by these functions, it makes sense to try incorporating the GPU in the eye tracking process. However, to utilise the GPU, a suitable way of writing programs for it is required. As mentioned previously, APIs such as OpenGL® and Direct3D can be used. The following section will give a short overview of a few of the APIs and languages available to programmers.

3.4.3 Graphics APIs and shader languages

There are several APIs and languages available with which to interface with the GPU. In addition to DirectX and HLSL, there is OpenGL®. OpenGL® was first created by Silicon Graphics, Inc. in 1989 as an open and reproducible alternative to Iris GL, which was at the time the proprietary graphics API on Silicon Graphics workstations (OpenGL, 2015). It was similar to Iris GL in some respects, and can be thought of as a formalized definition of Iris GL. It is supported on both Windows and Linux systems, and as such has gained a wide appeal.

Open Cg is a C-like language that allows programmers to control the shape, appearance and motion on drawn objects using programmable graphics hardware . It was created in collaboration with Microsoft and is compatible with both OpenGL® and HLSL for DirectX 9.0.

OpenGL and DirectX are generally used for 3D graphics. However, the advent of programmable GPU operations has opened the door for more general computations to be performed on the GPU.

3.4.4 GPGPU applications

General Purpose calculations on Graphics Processing Units (GPGPU) is a term that refers to using the GPU for general purpose calculations instead of graphics rendering. One example of this is making use of the GPU to perform Fast Fourier Transforms (FFT) (Govindaraju, Lloyd, Dotsenko, Smith & Manferdelli, 2008) and to reconstruct images from magnetic resonance imaging (MRI) (Archirapatkave, Sumilo, See, & Achalakul, 2011; Sumanaweera & Liu, 2005). There are several APIs that can be used to perform general computations on the GPU.

Two popular APIs for GPGPU applications are NVIDIA’s CUDA (Compute Unified Device Architecture) and OpenCL (Open Compute Language). More recently, Microsoft’s DirectX 11 API has support for what is referred to as a compute shader. A compute shader (also known as DirectCompute) is a programmable shader that provides high-speed general purpose computing (Microsoft, 2015a).

The eye tracking software developed as part of this study will make use of DirectX for its implementation. DirectX has the advantage that it is supported by most modern GPUs, whereas CUDA is restricted to NVIDIA cards only (NVIDIA, 2015). A disadvantage of DirectX is that it runs on Windows platforms only. However, OpenGL offers similar functionality and it is therefore conceivable that the solution could be ported to OpenGL, thereby providing support for other operating systems, such as Linux, should it be needed.

The next section discusses DirectX and its corresponding shader language, HLSL, as well as the components of these technologies that will be utilised in the eye tracker application.

3.5 DirectX

DirectX is one of the commonly used APIs that provides access to the capabilities of the computer’s video and audio cards, which allows for the development of hardware-accelerated applications. For the purpose of the developed eye tracker, DirectX 9 for C# has been selected as the API (also referred to as Managed DirectX). Compared to later versions, DirectX 9, specifically Direct3D 9, has the advantage that it is supported by all current hardware (Microsoft, 2015b). The DirectX framework consists of numerous components that allow access to input, audio and display devices. The core of the proposed solution is powered by Direct3D, which facilitates 3-D graphics programming. Figure 3.5 below illustrates the DirectX graphics pipeline.

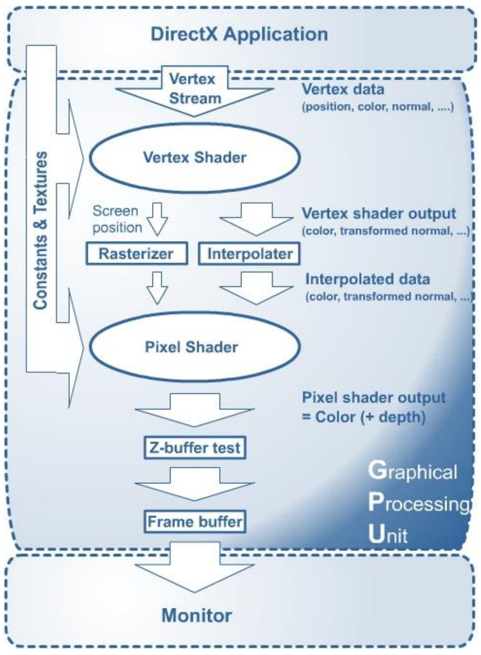


Figure 3.5 - The DirectX 9 Graphics Pipeline

Source: http://www.riemers.net/eng/Tutorials/DirectX/Csharp/Series3/Vertex_Shader.php

Last accessed: 02/02/2015

The vertex shader performs per-vertex processing, such as transforming the vertex coordinates from world coordinates to screen coordinates. At the very least, the vertex shader must output vertex positions in 2D coordinates representing the corresponding screen coordinates of the 3D positions. Additionally, the vertex shader can output texture coordinates, vertex colours, lighting factors and so on. As the vertices provided in the proposed solution are in no need of transformation, this section of the shader can be omitted in the implementation of the eye tracker software.

The pixel shader in turn performs manipulations on a per-pixel basis. This includes lighting computations and texture lookups. The pixel shader works in tandem with the vertex shader, where the outputs of the vertex shader are used as input for the pixel shader. It is at this stage of the pipeline that the eye tracking image processing functions will be implemented using shaders written in High Level Shader Language (HLSL) (see next section).

3.5.1 HLSL

HLSL is a High Level Shader Language for DirectX that allows one to create C-like programmable shaders for the Direct3D pipeline. It was first introduced in DirectX 9 (Peeper & Mitchell, 2004), and allows developers to take control of exactly what happens to graphics data on a per-vertex and per-pixel basis. The figure below provides a sample of a typical HLSL shader.

```
// Declare an output structure with a semantic binding
struct OutStruct
{
      float2 Tex2 : TEXCOORD2
};

// Declare the Tex0 out parameter as containing TEXCOORD0 data
float4 main(out float2 Tex0 : TEXCOORD0, out OutStruct Out ) : POSITION
{
      Tex0 = float2(1.0, 0.0);
      Out.Tex2 = float2(0.1, 0.2);
      return float4(0.5, 0.5, 0.5, 1);
}

// Declare the Col variable as containing the interpolated COLOR0 value
float4 mainPS( out float4 Col1 : COLOR1) : COLOR
{
      // write out to render target 1 using out parameter
      Col1 = float4(0.0, 0.0, 0.0, 0.0);

      // write to render target 0 using the declared return destination
      return float4(1.0, 0.9722, 0.3333334, 0);
}
```

Figure 3.6 - Sample HLSL code

3.5.2 Utilising HLSL

As images will be processed rather than large sets of vertices, a way of convincing the API into thinking that it is working with 3D is required. In general, the GPU's approach to 2D image processing is a subset of 3D processing (Bjorke, 2005). Thus, a quadrilateral polygon consisting of 6 vertices can be aligned to the output screen and rendered, either to the screen, or onto an off-screen buffer. As mentioned earlier, vertex positions are already in clip space and no vertex processing functions are required. Each pixel can then be manipulated by pixel shader programs to achieve the desired output. Pixel operations performed on the GPU do not lead to a loss of image quality, and are extremely fast. This allows high resolution images to be processed swiftly and without any visual artefacts. By rendering to an off-screen buffer, it will be possible to output the results of each shader pass to system memory. From here, the results of the image processing can be utilised to extract the pupils and corneal reflections.

3.5.3 Conclusion

In the section above, DirectX and HLSL have been discussed, and the relevant parts of the graphics pipeline that will be used during development have been highlighted. As the eye tracking feature detection process involves image processing, pixel shaders written in HLSL will be utilised.

The following section provides details on the proposed solution and also identifies various issues related to the development that need to be addressed.

3.6 Proposed solution

As stated earlier, one of the chief benefits of utilising the GPU is that it frees up CPU cycles that would otherwise be spent on processing the eye images. This allows the CPU to spend more time performing other tasks. For instance, the GPU can be used to assemble screen captures with video overlays along with a heat map or gaze paths in real time (Duchowski, Price, Meyer, & Orero, 2012). This could potentially allow for much faster processing of the eye video, which in turn means a higher sampling rate for the eye tracker. With this in mind, the eye tracker application developed as part of this study – henceforth referred to as the HLSL tracker – seeks to remove the image pre-processing burden from the CPU, and have it done by the GPU. This will be accomplished using Direct3D and a number of pixel shaders written in HLSL.

However, even though image processing on the GPU is theoretically faster, there are potential stumbling blocks that need to be considered.

3.6.1 Overhead

The first issue is that of the overhead incurred when converting the raw image data into the format required by Direct3D, and then rendering it using the appropriate pixel shaders. In the event that this overhead is too large, the gains made by using the GPU will in effect be cancelled out. An additional overhead will also feature when copying the image data from the GPU memory back to the system memory. It will therefore be necessary to study the rate at which the overhead grows in relation to the image size.

3.6.2 Image processing functions and compatibility

To extend compatibility to older systems, all pixel shaders will be Shader Model 2 compatible. This limits the number of instructions that can be performed in the pixel shader to 256 (Microsoft, 2015c) and will in turn affect the choice of image processing functions. HLSL is not a language developed for general computations, and one is therefore limited to simple per-pixel operations. However, thresholding, edge detection and noise removal are prevalent in eye tracking and are simple to implement in HLSL. They also contain very few instructions, and will therefore fall within the 256 instruction limit.

Not all issues are GPU related, however. Additional factors that need to be taken into account are the choice of camera, its resolution and the positioning and intensity of the IR light sources.

3.6.3 Camera selection

For accurate results, a high resolution camera will be required. As mentioned earlier in the chapter, the camera should also be capable of attaining high sampling rates. The choice of resolution is also important, as it will affect the size of the head box and in turn the amount of head movement that is tolerated by the system, as well as the overhead incurred by the transfer to and from the GPU.

3.6.4 Infrared light sources

It is important to consider the number and positioning of the infrared (IR) light sources, as well as the intensity of these light sources with regard to the number of IR LEDs each light will consist of. The positioning of the light sources has an effect on the appearance of the pupil. If positioned on the optical axis of the camera, a bright pupil image is generated (Zhu & Ji, 2005b) (Figure 2.12 in previous chapter). This is useful when making use of difference images.

However, difference images will not be used, as they effectively halve the sampling rate (Hennessey et al., 2008).

3.6.5 Conclusions

The section above highlighted the role that the GPU will play in the eye tracker. The challenges associated with this goal were identified, along with further hardware related challenges such as the choice of camera and positioning and number of light sources.

Attention will be devoted next to discussing the development of the eye tracker software and choice of hardware, as well as the various changes that were made throughout the development cycle.

3.7 Hardware setup

The final eye tracker setup consisted of a single USB 3.0 CMOS camera from IDS (Model UI-3360CP-NIR-GL) (https://en.ids-imaging.com/store/ui-3360cp.html). At the time of writing, the price of the camera was around 600 euros. The native resolution of this camera is 2048x1088 pixels and can be modified using a hardware area of interest (AOI). This allows for great flexibility when deciding on a suitable eye video size. At the native resolution, the camera is capable of around 70 FPS (https://en.ids-imaging.com/IDS/spec_pdf.php?sku=AB00475). In addition to hardware AOIs, the camera also allows the frame rate to be adjusted. The camera is fitted with a 16 mm lens along with a daylight filter that eliminates most natural light, with the exception of the IR band. This filter should yield a much cleaner image and minimise the chance of secondary corneal reflections caused by other light sources in the vicinity of the eye tracker. To generate the corneal reflections, two IR LEDs were positioned equidistantly on each side of the camera.

Figure 3.7 – The UI-3360CP camera used for the eye tracker
Source: https://en.ids-imaging.com/store/ui-3360cp.html.
Last accessed: 16/04/2015

Initially, only one IR light source positioned next to the camera was used. However, it was observed that extreme head movements caused shadows on the face which made it difficult to locate the pupils. The addition of a second light and the new positioning of the lights solved this issue.

3.7.1 Selection of eye video size

The size of the eye video was not fixed, as it forms part of the evaluation of the eye tracker. Instead, multiple resolutions will be tested to evaluate different sampling frequencies and head positions.

3.8 Software development cycle

During the development of the HLSL tracker, a number of different implementations were developed and tested. In each iteration of the implementations, a shader was developed to process the image in such a way so as to minimize the amount of work that must be done by the CPU when processing the image.

Ideally the GPU should be able to perform all the processing required, i.e. manipulating the image, computing the pupil and glint centroids, as well as mapping the pupil and glint centroids to gaze coordinates. However, given the limitations of Shader Model 2, this would not be possible. Instead, the image would be processed by the GPU with the CPU handling the centroid computations along with the appropriate gaze mapping functions.

3.8.1 Simple threshold

Initially, a shader was developed using a simple threshold function to isolate the pupil and glints into a binary image. Using this threshold, the image was converted into a black and white image. The code snippet below illustrates how this was accomplished in HLSL.

```
color.rgb = 1 - color.rgb;
if (color.r < threshold)
{
      float3 originalColor = 1 - color.rgb;
      if (originalColor.r > threshold - 0.1)
        color.rgb = 1;
      else
        color.rgb = 0;
}
```

The snippet above inverts the image’s colours after which it applies the threshold test. It was observed that glints are most visible in the red colour channel. This claim is similar to that of the implementation of Lemahieu and Wyns (2010) in their low-cost eye tracker. Glint points are eliminated with the first threshold test. Next, potential pupil pixels are then made white, while all other pixels are made black.

While the simple threshold was effective at isolating the pupil in certain conditions, it suffered from two problems. Firstly, selecting the threshold had to be done manually through trial and error, and certain changes in lighting conditions prevented the threshold from working at all. Secondly, because of the fact that both the glints and the rest of the image were now black, the number of pixels that needed to be examined to locate the glints was considerably large. Moreover, there was no way of distinguishing between glints and the rest of the eye in the processed image, which meant that the original image needed to be scanned to locate the glints.

The separate processing of the original image renders the GPU implementation redundant, as there still remains a significant portion of the image processing to be done by the CPU. Thus, this technique requires improvement in two areas. Firstly, additional control over the threshold is required to make adjustments for different lighting conditions and participant characteristics. Secondly, a way to isolate the glints and pixels in the processed image is required in such a way that they are easy to locate without having to look at the original image.

3.8.2 Separate colour channels

To separate the glints and pupils from each other in the processed image, each was stored in a different colour channel. Thus, potential pupil pixels were rendered in green whilst glints were rendered in red. The decision on whether or not the pixel was a pupil or glint was based on two separate thresholds, as suggested by Lemahieu and Wyns (2010). This allowed for better control over the threshold in different lighting conditions. In addition to the separate colour channels, a number of additional processing functions were added to enhance the contrast between the pupil and the surrounding image.

3.8.3 Improved threshold functions

As mentioned above, the improved implementation makes use of two individual thresholds to isolate the pupils and glints. These two thresholds are called Bright and Dark for the glint and pupil respectively. Pixels that do not pass the threshold tests are made black, thereby greatly simplifying the task of locating the pupils and glints. The threshold function works as follows: After the initial pre-processing of the image, the shader selects pupil candidate points using the

Dark threshold and encodes these candidates into the green colour channel, thereby eliminating the problem that occurred in the previous implementation where the glints disappeared into the rest of the image. The intensity of the green colour is determined based on the original pixel's distance from the Dark threshold, as demonstrated by the snippet below.

```
color.rgb = float3(0,distance(color.rgb, Dark),0);
```

Next, glint points are isolated and coloured red. As the glint points are among the brightest pixels in the image, the Bright filter is used to isolate them. As with the pupil points, some image pre-processing is performed before the colour coding process.

```
      //Image pre-processing
      //…
      glintColor.rgb = distance(glintColor.rgb,0)
                        / distance(glintColor.rgb, 1);
      //Check if potential glint point
      if (glintColor.r > Bright)
            color.rgb = float3(1,0,0);
```

The solution above removes the need to process the original image separately from the processed image. With the glints and pupils each isolated into their respective colour channels, the CPU now needs only look through the results and compute the centroids for each feature. As a final step, the grey-scale colour value of the original image can be encoded into the blue colour channel. This provides the ability to quickly look up the original image values if needed.

3.8.4 Camera adjustments

The success of the threshold function can be improved further by making adjustments to the exposure time, gain and gamma of the camera. Informal experimentation revealed that it was unnecessary to make adjustments to the values of Dark and Bright, and that simple adjustments to the camera could compensate for the changes in light conditions or participant characteristics.

3.8.5 Centroid computation

After the image has been processed and candidate pixels separated into the green and red colour channels, the centres of the pupils and glints can be calculated by the CPU. The pupil and glint candidate points are aggregated into lists, and the AForge image processing library (https://code.google.com/p/aforge/) is used to calculate the centres of each of these collections.

Provided there are not large quantities of outliers involved, the centre is calculated fairly accurately. In this particular implementation of the HLSL tracker, no ellipse fitting was performed. However, it is certainly possible to add this step if needed at a later stage given that all the candidate feature points are already grouped into separate collections.

Unfortunately, the pupils and glints are not always the only parts of the video that pass the thresholding test. As a result, further improvements to the image processing are required, along with the centroid computations. The following section discusses the final version of the software.

3.9 Final software

The final implementation makes use of three shaders. The first is a pre-processing shader. This shader is responsible for identifying candidate pupil and glint pixels and separating them into their respective colour channels. The next shader is a post-process shader that is responsible for removing noise in the image if needed and performing edge detection on the pupil points. The final shader is used purely for feedback to the user in the form of the eye video. This shader is responsible for rendering the original eye video with markers for the detected feature points.

3.9.1 Pre-process shader

As mentioned in the previous section, one of the issues with thresholding is that non-feature points often pass the test. This problem can be solved if the last location of the pupil or glint is known. Points that lie outside a certain radius from the known location can then be discarded, i.e. coloured black. However, the challenge remains to find the initial location of the pupils and glints. To tackle this challenge, a number of further improvements to the threshold functions were made. These changes were incorporated into the pre-process shader.

3.9.1.1 Glint isolation

To locate the eyes initially, it was determined that the glints could be used reliably. Thus, once the glints have been located, the area around the glints can be searched for the pupil points. To make the glints easier to locate, the following lines of code were added to the pre-process shader.

```
//Left-right
float c1 = tex2D(TextureSampler, float2(uv.x + 1 / width, uv.y));
float c2 = tex2D(TextureSampler, float2(uv.x - 1 / width, uv.y));
//Update down
float c3 = tex2D(TextureSampler, float2(uv.x, uv.y - 1 / height));
float c4 = tex2D(TextureSampler, float2(uv.x, uv.y + 1 / height));
float3 n = (abs(c1 - c2) + abs(c3 - c4)) / 2;
if (n.r > GlintThresh)
      color.rb = float2(1, 0);
```

To isolate the glints, the surrounding opposing pixels are subtracted from each other. The average of these differences is then taken and tested with a threshold. Pixels that pass the test are glint candidates and are coded into the red colour channel. The figure below shows the results of the above code snippet.

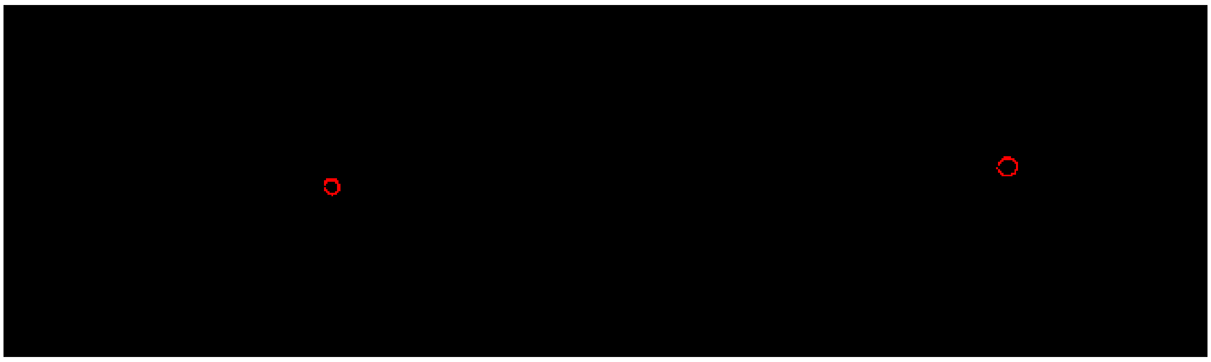


Figure 3.8 - The result of the glint threshold test

This threshold has proved to be very reliable once the camera's gain, gamma and exposure time has been adjusted for a participant, and works for large eye videos (1024×768). As with many IR systems, however, it does still suffer when confronted with other reflective surfaces such as spectacles or particularly bright areas in the background of the image.

3.9.1.2 Pupil isolation

To isolate the pupil pixels, a second threshold is used. If the location of the glints is not known, any pixel that passes the threshold test is coded into the green colour channel. However, if the location of the glints is known, some additional processing is performed instead. The distance between the closest glint and the current pixel is determined. The pixel's green colour value is determined using the distance and a radius. Hence, pixels that lie too far away from the glints are made black. Pixels inside the radius will have various shades of green.

```
float d = 1 - min(distance(screen, LeftGlint),  distance(screen,
RightGlint)) / radius;
color.g = d;
```

By doing this, certain types of noise, such as that caused by eyebrows and eyelashes, can be eliminated before processing the image, thereby simplifying the next step greatly. As a final step, an additional test on the pixels that passed the initial test converts the green gradient into a solid colour.

3.9.2 Post-process

The post process shader takes the output of the pre-process shader and removes outliers from the image. Additionally, it performs edge detection to further decrease the number of pixels that must be processed by the CPU.

3.9.3 Noise removal

In addition to the post-process shader, a noise removal shader was created that passes over the same image multiple times in order to solidify the pupil regions. This is useful where lighting conditions are poor and ensures a stable pupil point. Finally, the output image is processed by the CPU to determine the coordinates of the pupils and glints. The figures below show the resulting images. Figure 3.8 represents the original noisy pupil image, with the results of each pass of the noise removal filter. Figure 3.9 shows the resulting eye video with the feature points located.

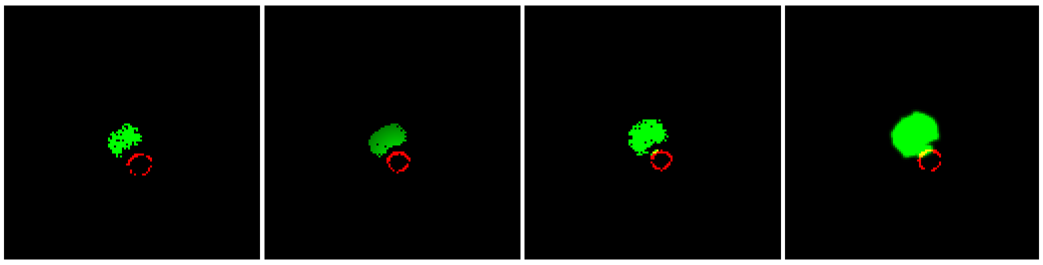


Figure 3.9 - The results of each pass in the noise removal shader

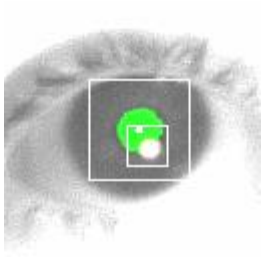


Figure 3.10 - The resulting eye video after noise removal

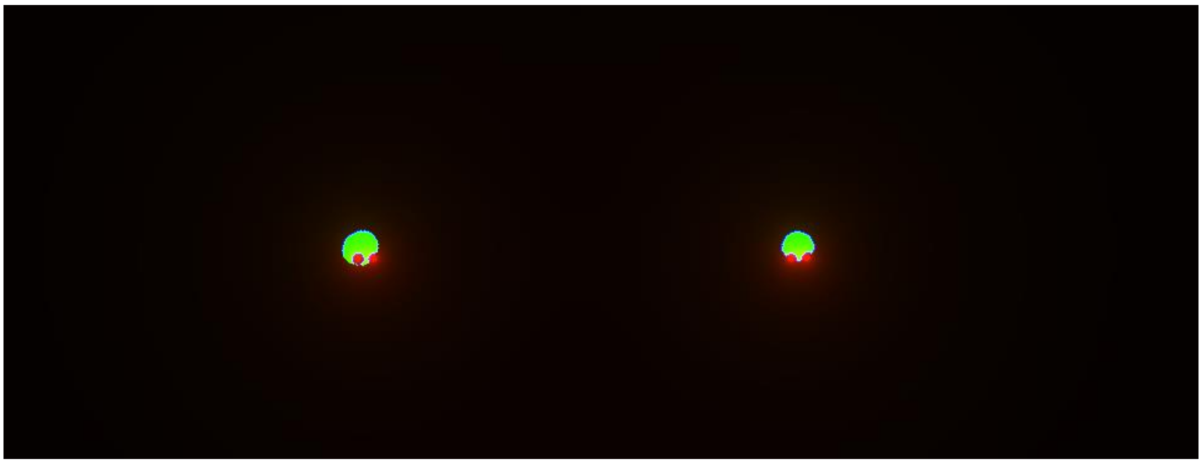
3.9.4 Pupil deformation

An additional issue that arises in the pupil centre computation process is when the glint and the pupil intersect. This intersection causes the pupil to lose its circular shape, which in turn offsets the computed pupil centre. To counter this, glint pixels inside the previously known pupil radius are coloured as pupil and glint pixels. In other words, they are made yellow.

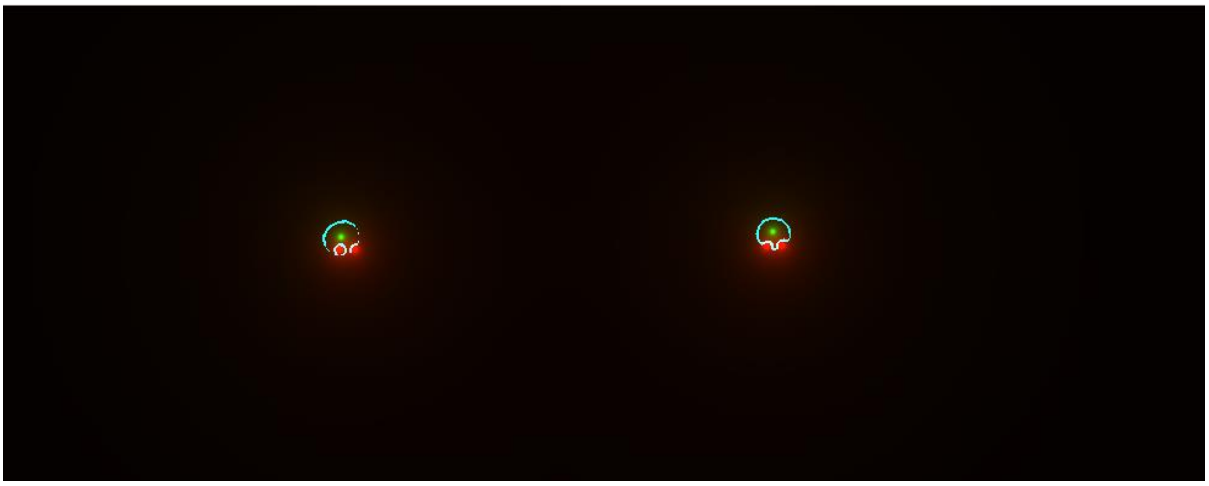
3.9.5 Feature point validation

As a final improvement, the feature point validation was improved by means of two tests. Firstly, glints were required to be a certain distance from each other. This distance was estimated based on the distance of the camera from the participant, and the lens fitted to the camera. The second test involved the values of the glints. Values that were zero or NaN (Not a Number) triggered a reacquisition flag, and resulted in the software ROIs being reset to the entire image. Pupil validation was performed using the locations of the glints. Candidate points that lay outside a certain radius from the closest glint were discarded from the pupil centre computations.

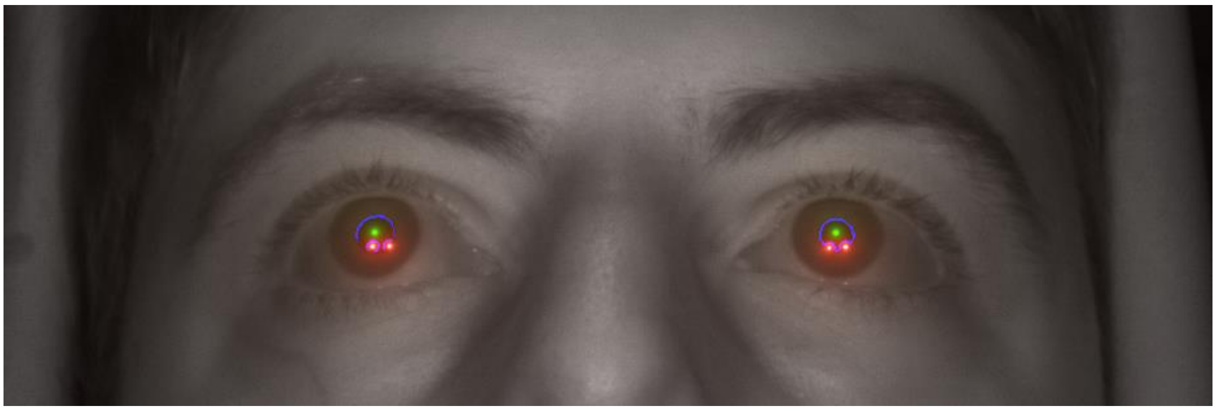
These validation tests helped to ensure that thick lashes, makeup or dark eyebrows did not interfere with the detection of the actual feature points. The images below show the outputs of each shader involved. As mentioned earlier, the final implementation makes use of two IR light sources, rather than one.



(a)



(b)



(c)

Figure 3.11 - Output of each step: (a) pre-processing, (b) post-processing, and (c) resulting eye video

3.10 Gaze estimation

In Chapter 2, the two gaze estimation models were discussed. To evaluate the accuracy and precision of the HLSL tracker described in this chapter, it is necessary to perform gaze estimation on the data generated during the experimental described in Chapter 4. The choice of gaze estimation will ultimately affect the accuracy that can be obtained (see Chapter 2, Section 2.9.1). As a complete evaluation of different mapping techniques falls beyond the scope of the dissertation, only a single technique, namely regression-based gaze estimation, will be used during the analysis. The following two polynomials will be used to compute the point of gaze (Blignaut, 2013):

$$PoG_x = a_{x0} + a_{x1}\hat{x} + a_{x2}\hat{x}^2 + a_{x3}\hat{x}^3 + a_{x4}\hat{y} + a_{x5}\hat{x}\hat{y} + a_{x6}\hat{x}^2\hat{y} + a_{x7}\hat{x}^3\hat{y}$$
$$PoG_y = a_{y0} + a_{y1}\hat{x} + a_{y2}\hat{x}^2 + a_{y3}\hat{y} + a_{y4}\hat{y}^2 + a_{y5}\hat{x}\hat{y} + a_{y6}\hat{x}^2\hat{y}$$

where $(\hat{x}, \hat{y})$ are the coordinates of the Pupil-Corneal Reflection vector, and $a_{x,y}$ the coefficients to be determined using regression based on calibration data obtained.

3.11 Conclusion

The purpose of this chapter was to provide examples of existing eye tracker implementations that address issues of cost, sampling rates and accuracy and precision. Next, relevant hardware and software technologies were identified to develop an eye tracking solution that attempted to improve upon the existing work. Of special note was the inclusion of the Graphics Processing Unit (GPU) which was utilised in the HLSL tracker to facilitate image processing functions involved in the eye tracking process. The chapter concluded with details of the eye tracking implementation and the gaze estimation that will be utilised for the purpose of evaluation.

The following chapter will discuss the methodology used to evaluate the performance of the eye tracker.

CHAPTER 4: EXPERIMENTAL DESIGN AND METHODOLOGY

In the previous chapter, the implementation of the eye tracker was discussed. The goal of Chapter 4 is to discuss how the eye tracker will be evaluated. Included in this chapter is a discussion on the various methods that can be utilised to perform research of this nature, as well as motivation for the selected method. Furthermore, the design of the research method will be provided and discussed with appropriate motivation from the literature consulted.

4.1 Introduction

There are several methods available with which research can be conducted (Olivier, 2009). Examples of common methods include experiments as well as design and creation (Oates, 2006). Up to this point, the goals of the study have been described in a general fashion. To review from Chapter 1, the main research objective is to develop an eye tracker that conforms to the following criteria:

- Captures eye movements at a high sampling frequency
- Has a high level of tolerance toward head movements and eye characteristics
- Provides data with a high level of precision, accuracy and trackability
- Is affordable (low-cost) when compared to commercial systems

As discussed in Chapter 3, the eye tracker consists of both hardware and software components. The novelty of the HLSL tracker lies in the software component and the use of the GPU for image processing. The software component must therefore be evaluated to determine the effectiveness of the solution.

In Chapter 2, five of the measures used to judge eye trackers were discussed. These measures were accuracy, precision, trackability, latency (no included in the study) and sampling frequency. It then follows that in order to effectively evaluate the developed eye tracker, it is necessary to measure its performance in terms of these five measures, since any research involving a software artefact needs to be evaluated using relevant criteria (Oates, 2006). For this to occur, the exact performance criteria need to be clearly defined so that a frame of comparison can be established.

An additional concern is the interaction that the above-mentioned factors have with one another. It is therefore important that each of these factors is examined in isolation, in addition to an overall evaluation of the eye tracker system. With the individual evaluation of each of these performance criteria, it may be possible to find the best possible combination of sampling frequency and head box size.

The following section will discuss the theoretical framework for the research, specifically focussing on a method by which the software artefact can be evaluated.

4.2 Theoretical framework for research

Research can be defined as an investigation that is carried out systematically in order to establish facts (Olivier, 2009). There are several ways in which research can be conducted. For example, existing knowledge can be used and integrated, a problem can be solved for which no apparent solution exists or a better way of doing something can be found (Olivier, 2009). Once an appropriate research method has been selected, a researcher needs to address the purpose of the research and describe the research process in sufficient detail to satisfy the academic community of its validity (Oates, 2006).

The purpose of the current research study has already been discussed in great detail in the previous chapters. What remains now is to evaluate the eye tracker using an appropriate technique, and to provide sufficient details of this technique so as to establish the validity of the results obtained.

4.2.1 Evaluation options

As alluded to above, the eye tracker must be evaluated in terms of the common eye tracker performance measurements. Furthermore, care must be taken to isolate these measurements from one another. With this in mind, one possible way to evaluate the system would be through an experiment in a laboratory. Experiments can be used to test a hypothesis or to observe the effect that variables have on one another (Hofstee, 2013). During an experiment, conditions can be manipulated and the impact thereof measured. Experimental research is ideally suited to scenarios where cause and effect relationships need to be evaluated (Mahmud, 2008). Experiments are typically conducted in the field or in a laboratory – the latter offering a higher level of control over variables than a field experiment would (Hofstee, 2013; Mahmud, 2008). Experiments in a laboratory setting are also commonly used in eye tracker evaluations. Blignaut and Wium (2014), for example, made use of a laboratory setup to determine the effect of ethnicity on eye tracker data quality.

These facts indicate that an experiment would be an ideal method to evaluate an eye tracker, as there are several variables – such as sampling frequency and head position – that need to be studied in isolation. Consequently, by testing the eye tracker in a laboratory, the various factors involved in an eye tracker's performance can be controlled and the effect of each one on the overall precision of the system measured.

One potential issue with the conducting of an experiment is that the results may not be representative of the general populace. This is due to the fact that experiments generally only make use of a small number of participants (Mahmud, 2008). It is therefore important that the selection of participants accounts for this, and that the selection be as representative as possible.

4.2.2 Conclusion

In this section, the theoretical framework for the evaluation of a software artefact has been discussed, and a method for evaluation chosen. The chosen method will consist of an experiment to be performed in a laboratory setting. This is due to the fact that the interaction between data quality measures such as precision, accuracy and trackability needs to be isolated from the sampling frequency and the position of the head within the head box. Moreover, when selecting participants, it is important that they be representative of the general populace. The following section discusses the details of the research setup, as well as how the results of the experiment will be analysed and interpreted.

4.3 Research design

The goal of this study is to measure the performance of the eye tracking software in terms of the common data quality measures of accuracy, precision and trackability for various users, at different head positions and sampling rates.

4.3.1 Research problem

To reiterate from Chapter 1, commercial high-speed eye trackers are expensive and thus inaccessible to many. There exist low cost alternatives, such as the Tobii EyeX, available for $99 (Tobii Technologies, 2014). However experience has shown that the EyeX provides sampling rates of 60 Hz at most. In Chapter 2, the various applications of eye trackers were discussed, and it was shown that these low frequency eye trackers are not sufficient for certain research fields, such as research involving eye movements during reading. To this end, this study has aimed to present a low-cost eye tracking system – presented in Chapter 3 – that is capable of achieving the required sampling rates, level of precision, accuracy and trackability needed to conduct research in these fields.

4.3.2 Research hypothesis

The developed eye tracker needs to be evaluated to determine the effectiveness of the designed solution. It was mentioned in the previous section that, for the purposes of a complete evaluation, the eye tracker needs to be evaluated in terms of the five eye tracker metrics. This study is primarily concerned with evaluating the accuracy, precision and trackability and sampling frequency of the developed eye tracker. Formally stated, the hypotheses of the study as defined in Chapter 1 are as follows:

$H_{0,1}$: The sampling frequency has no effect on data quality (precision, accuracy and trackability).
$H_{0,2}$: The position of the head within the head box has no effect on the data quality (precision, accuracy and trackability).

Each of these hypotheses is divided into three sub-hypotheses. Thus, the hypotheses presented in this dissertation are summarised as follows:

- $H_{0,1.1}$: The sampling frequency has no effect on the precision of the eye tracker.
- $H_{0,1.2}$: The sampling frequency has no effect on the accuracy of the eye tracker.
- $H_{0,1.3}$: The sampling frequency has no effect on the trackability of the eye tracker.
- $H_{0,2.1}$: The position of the head within the head box has no effect on the precision of the eye tracker.
- $H_{0,2.2}$: The position of the head within the head box has no effect on the accuracy of the eye tracker.
- $H_{0,2.3}$: The position of the head within the head box has no effect on the trackability of the eye tracker.

As a secondary goal, the study also aims to determine the best possible combination of eye video size and sampling frequency in order to achieve the largest possible head movements tolerable, coupled with the highest possible sampling rate. Taking into account these goals, the following research questions are of interest:

- What is the maximum obtainable sampling frequency that the solution can achieve?
- What percentage of participants can be tracked?

- What percentage of frames will need to be discarded due to tracking error, i.e. what is the trackability of the system?
- What is the precision and accuracy that is obtainable?
- What is the best possible combination of sampling frequency and head box size?

To answer the questions above, it is necessary to evaluate each of the components in isolation and to consider the possible interactions between each of the components. Using the results of these evaluations, the best possible theoretical combination can then be found.

Answers to these questions will provide two research results. Firstly, the performance of the eye tracker in terms of the common data quality measures used in eye tracking (sampling rate, precision, accuracy and trackability) will have been established, yielding a general impression of how well the eye tracker can perform. This will determine whether or not the eye tracking solution is feasible, and can be used to perform research in the fields for which it has been developed. Secondly, the best possible combination of the various factors will allow for an optimal setup of the system for researchers who intend to make use of the system, should it meet the required performance metrics.

4.3.3 Conclusions

The preceding section has identified the hypotheses and research questions that will be investigated in the experiment. In the following section, details of the experimental setup will be discussed.

4.4 Experimental design and methodology

The purpose of this section is to provide details of the physical setup of the experiment. Included in the discussion are details on the resolution of the eye video, sampling frequency of the tracker and various eye tracker positions that will be used to evaluate the performance of the eye tracker.

4.4.1 Physical setup

The study will make use of the eye tracking software discussed in the previous chapter. The entire study will be conducted in a laboratory (Figure 4.1). The laboratory contains four adjustable lights in each corner that will be adjusted for an overall brightness of 300 lux. Also contained within the laboratory is a table on which the eye tracker and computer monitor will be positioned. The table is equipped with an adjustable platform mounted on railings that allow it to be moved left, right, forward and backward. The table also contains a motorised servo that allows it to be raised or lowered. It is therefore possible to make precise adjustments to the position of the eye tracker in relation to the participant. Also mounted on the table is a head rest that will be used to keep the participant's head steady during the testing process.

The eye tracker setup consists of two infrared light sources and a CMOS camera positioned 65 cm from the participant. The stimulus is a computer monitor with a resolution of 1360×768 positioned 70 cm from the participant.

The computer running the eye tracking and data capture software is an HP Pavilion G7 laptop, running Windows 8 64 bit. It is equipped with an Intel Core i5-3230M running at 2.60 GHz with 4 GB RAM and an Intel HD4000 integrated graphics card in addition to a Radeon HD 7670M. The laptop allows the user to select which of the two adapters to use for a specific application. For the purpose of this study, the Intel HD4000 display adapter will be used.

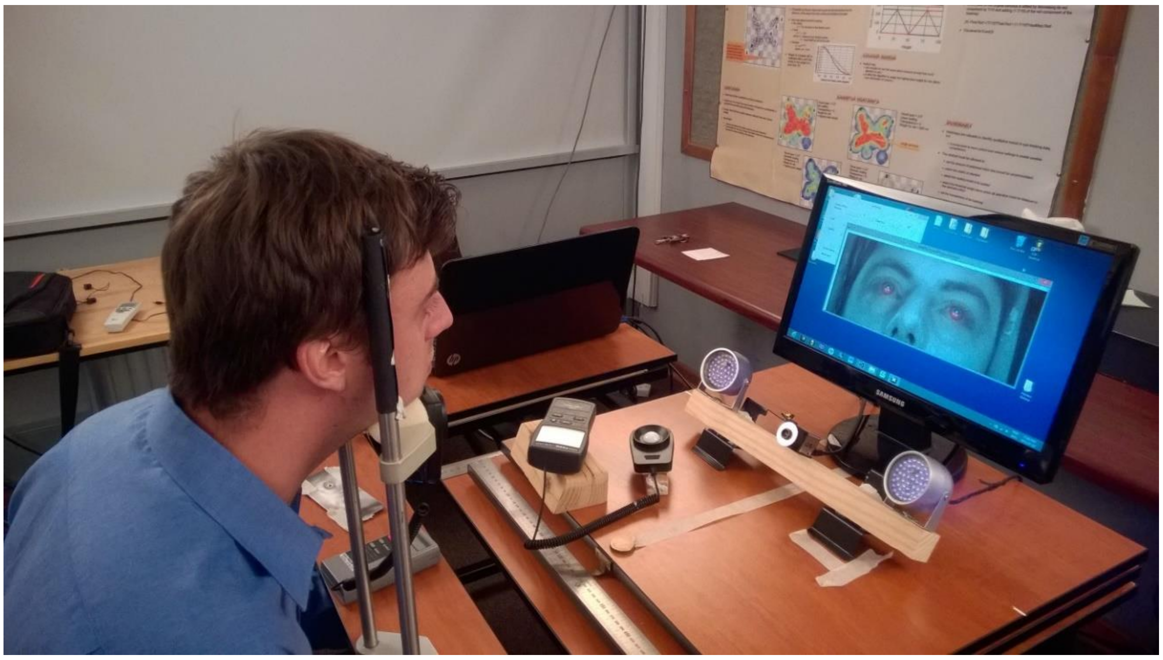


Figure 4.1 - The laboratory setup that will be used during the experiment

4.4.2 Participants

The study should involve thirty participants. To ensure the best possible results, none of the participants will wear spectacles. Participants will be recruited from among staff and students on the campus. Participants will be selected such that they are representative of the general population. Participant details such as the wearing of thick mascara, contact lenses or facial piercing will be recorded. In the event that a participant cannot be tracked, these details will be used to determine the cause of failure.

4.4.3 Experimental design

The experiment will require the participants to focus on a grid of forty dots evenly spaced on the stimulus. The stimulus will have a black background colour, whilst the dots will be coloured red. The dots will be displayed one by one in a random order for a period of two seconds each. The random order in which the dots are displayed will prevent the participant from pre-emptively glancing towards the next dot. Displaying the dot for two seconds will also generate a large portion of data from which a suitable selection can be made for analysis.

There will be two parts to the experiment. The first part will involve human participants, while the second part will consist of simulations performed on artificial eyes. The human part of the experiment consists of two phases. In the first phase, the sampling rate will be modified to evaluate the effect of higher sampling rates on the trackability, robustness and precision of the eye tracker. During this phase, resolution of the eye video will be set to 600×150 pixels. The sampling rates that will be tested range from 50 Hz to 300 Hz in incrementing steps of 50 Hz.

In the second phase of the human part of the experiment, both the sampling rate and resolution of the eye video will be held constant. The position of the eye tracker in relation to the participant will be changed on three axes. The resolution of the eye video will be set to 1200×400 pixels to accommodate the various offsets that will be tested, whilst the sampling rate will be held constant at 100 Hz. Table 4.1 summarises the physical layouts to be used during the two phases of the experiment.

Table 4.1 - Summary of the various combinations of settings that will be used during the experiment

Sampling Frequency (Hz)	Camera Resolution (Pixels)	X Position (cm)	Y Position (cm)	Z Position (cm)
Phase 1				
50	600x150	0	0	65
100	600x150	0	0	65
150	600x150	0	0	65
200	600x150	0	0	65
250	600x150	0	0	65
300	600x150	0	0	65
Phase 2				
100	1200x400	+5	0	65
100	1200x400	-5	0	65
100	1200x400	0	+3	65
100	1200x400	0	-3	65
100	1200x400	0	0	70
100	1200x400	0	0	60

During each stage of the experiment, it will be possible for the test administrator to adjust the gain, gamma and exposure time of the camera. This is necessary as the changes in sampling rates between experiments have an effect on the amount of illumination present in the eye video. The camera also allows for adjustments in the image offset. Between experimental phases, the administrator will make use of this feature to centre the participant’s eyes within the eye video before beginning with that particular phase.

4.4.4 Design motivation

The choice of eye video resolutions will ensure that the required sampling rates and head box size can be obtained. In terms of the offsets, [-3, -3] on the y axis have been chosen as it can be argued that one does not require a large vertical head box. This is due to the fact that the participant’s vertical position in relation to the eye tracker’s position will not change significantly during the use of a computer.

4.5 Simulations

In addition to the laboratory experiment, the theoretical performance of the eye tracker will also be examined by running simulations on artificial eyes. These simulations serve two purposes. Firstly, the experiment will yield precision values that can be benchmarked against other available systems. Secondly, the simulations will provide data indicating the maximum obtainable sampling frequency for a variety of eye video resolutions. Using this data and the results of the laboratory study, the best theoretical combination of sampling rate and head box size can be selected. The following resolutions will be examined: 640×480, 800×600, 1024×768, 1200×400, 1000×200,800×400, 800×200 and 600×200.

4.6 Experimental data capture and analysis

During the experiment phase, the following raw data will be recorded:

- The location in pixels of the pupils and corneal reflections in the eye video for each captured frame
- The corresponding location of the dots on the stimulus in pixels
- The time stamps of individual frames

The data will be captured using an application developed for this purpose. All resulting data will be exported into a database, and the results will be analysed in Statistica 12 to answer the questions stated in Section 4.3.2 above. All data analysis will be performed offline. This will allow for later changes to the type of polynomial used to perform the gaze estimation, should further exploration of such decision be required in later research. The details additional software used in the data analysis are described in further detail in Appendix A.

4.6.1 Precision

Precision will be calculated using the polynomials described in Chapter 3 Section 3.10 and calibrated using fourteen of the forty data points. Two results will then be validated using all forty of the calibration points. To accommodate saccades that occur between points, a period of data will be selected where it is determined that the actual fixation occurred. Details of the process through which this data is selected is discussed in more detail in Chapter 5, Sections 5.2.1 and 5.2.2.

4.6.2 Accuracy

To determine the accuracy of the system, the same mapping polynomial used to determine precision will be used. As both the accuracy and precision make use of the same polynomial, it makes sense to utilise the same period of data that was selected for the precision analysis. Accuracy will be calculated by looking at the difference between the reported point of gaze and the position of each dot displayed.

4.6.3 Sampling frequency

Time stamps will be analysed to determine the average sampling frequency obtained per participant. This will be used to verify that the eye tracker was sampling at the required rate. In the simulation phase, the time stamps will be analysed to determine the maximum obtainable sampling rate.

4.6.4 Trackability

The status of the eye data will be combined with that of the frame number to determine what percentage of data is invalid, and how often the eyes were correctly detected. This will be evaluated over different sampling frequencies and for different head positions.

4.6.5 Head box

The size of the head box is defined to be 10×6×10 cm. The effect of the position of the head in the head box will be evaluated by examining the precision, accuracy and trackability of the system at a fixed sampling frequency (100 Hz).

4.6.6 Eye video size

During the theoretical part of the experiment, the eye video size will be adjusted so as to determine the changes in overall processing time, and by implication the maximum attainable sampling rate and head box size.

4.7 Limitations

There are several limitations imposed by the restrictions of the experiment. As mentioned earlier, the HLSL tracker should be evaluated using the five performance metrics of eye trackers (accuracy, precision, sampling rate, trackability and latency). However, as latency was not evaluated, it leaves the overall performance of the system in question. An additional limitation is that the experiment is conducted in a laboratory setting, and the results may not be applicable in a real-life environment. Furthermore, by limiting the study to participants that do not wear spectacles, the experiment potentially excludes a significant portion of the general populace.

4.8 Summary

The theoretical framework for the evaluation of the developed eye tracker has been discussed, and the method of evaluation explained. Furthermore, the performance metrics that will be examined have been identified and comparison values have been selected. Finally, details on the setup of each phase of the experiment have been provided.

In the following chapter, the data resulting from the experiment will be analysed and the results discussed.

Chapter 5: Experimental Results

The previous chapter described the way in which the HLSL eye tracker should be evaluated. It is the purpose of this chapter to present the results of the various experiments, and to provide an interpretation of these results where necessary.

5.1 Introduction

The quality of sampled data can be influenced by several factors, including the characteristics of the participants, features of the equipment, or the setup of the experimentation (Blignaut & Wium, 2014). The first section in this chapter will explain the process by which errors relating to participant characteristics and experimental setup failures were identified and removed from the final data set. This was necessary in order for the performance of the eye tracker itself to be evaluated, and to eliminate the influence of experimental design faults. The second section of this chapter will discuss the methods through which the resulting data were analysed and present the results of the experiment.

5.2 Raw data preparation

Examination of the raw data revealed several issues that needed to be addressed before an analysis could be conducted. First among these issues was the effect that missing gaze data had on the average precision and accuracy obtained. As the missing gaze data would be used to calculate trackability, it was necessary to filter these points when working with accuracy and precision. An additional problem was the identification of a suitable time interval during which the fixation on the displayed dot occurred, as the presence of saccades between dots as well as the occasional glance away from the dot created instabilities in the data that were not system related. These instabilities needed to be removed to evaluate the actual performance of the system. A further consideration was the presence of outliers in the data set. In this section, the data preparation process that addresses these issues is discussed.

5.2.1 Missing gaze data

Missing gaze data points can be attributed to the occurrence of blinks and glances away from the screen. Data entries sampled during these two cases were identified by examining the quality of the reported feature points in respect to the location in pixels. Several cases were identified which were deemed to be invalid. The first case consists of feature points with coordinates where both the x and y components were zero. Due to the fact that the eye image was centred on the participants' eyes before the start of each experiment, zero coordinates could not possibly be obtained and were therefore flagged as invalid. A second case, similar to the first, was when feature point coordinates were reported as nonsensical values, such as negatives, infinity or extremely large values exceeding the resolution of the eye video. Lastly, feature points that were deemed to be too close to each other were also flagged as invalid. An example of such a case occurred when one of the participant's two eyes was occluded, and both the left and right eye yielded the same coordinates as a result.

5.2.2 Interval selection

Due to the fact that each dot was rendered directly after the previous dot, the initial data points sampled during the duration of the new dot contained feature point data from the previous dot. Moreover, the time it took each participant to switch focus from the old point to the new varied for each participant. During the switch, a saccade occurred. The duration of this saccade varied for each participant. Saccades may also have occurred when the participant glanced away from the dot during a blink. To compensate for these saccades, a suitable period of data needed to be selected during which the actual fixation on the dot occurred.

To address the issue of identifying the actual fixation, a filter was created to select a period of data with the highest likelihood of the fixation occurring. To ensure that a sufficient quantity of data was available for analysis, the period of data was set to at least 250 ms in duration. As data was sampled at a variety of different rates (see Table 4.1 in Chapter 4), a target number of data lines (n) was calculated by using the sampling frequency and the required interval of 250 ms.

$$n = \frac{Frequency \times 250}{1000}$$

The data for each participant was then partitioned into groups of data where the feature points of each entry in the database (i.e. every sample point) differed by no more than a single pixel from the previous entry. Finally, the group that contained n entries or more was selected. If none of the groups contained a sufficient number of entries, the largest period was selected.

5.2.3 Removal of outliers

Before performing statistical analysis on the mapped data outliers were removed. Outliers consisted of any values that lay outside of three standard deviations from the mean accuracy and mean precision. The decision to remove outliers was motivated by the need to determine the actual capabilities of the eye tracker when the participants were successfully tracked, as the invalid gaze estimates resulting from incorrect feature point detection would be included in the trackability measures.

5.3 Metrics

For the purpose of this dissertation, four metrics were examined, namely precision, accuracy, trackability and sampling frequency. In the first two phases of the experiment, the precision, accuracy and trackability of the system were measured for human participants over a number of different sampling frequencies and head positions (Table 4.1). The theoretical phase of the experiment was concerned with the precision obtainable with artificial eyes, and sampling rate obtainable for various eye video resolutions.

5.3.1 Accuracy and precision

The accuracy of the eye tracker was determined by calculating the error in degrees between the actual point of gaze (represented by the dot on the screen) and the reported point of gaze. This was done using the regression-based mapping procedure described in Chapter 2, Section 2.9.1 coupled with the polynomials described in Chapter 3 Section 3.10.

It is common practice to report precision using the Root Mean Square (RMS) measure. However, this dissertation will make use of pooled variance, as RMS is affected by the sampling rate of the eye tracker (Blignaut & Beelders, 2012). Given that the effect that various sampling rates have on the precision is one of the subjects under investigation, it would be unwise to report precision using RMS. Pooled variance is calculated with the following formula (Blignaut & Beelders, 2012):

$$SD(P) = \sqrt{(\sigma_x^2 + \sigma_y^2)/2}$$

where $\sigma_x^2 = \frac{1}{N}\sum_{i=1}^{N}(x_i - \bar{x})^2$ and $\sigma_y^2 = \frac{1}{N}\sum_{i=1}^{N}(y_i - \bar{y})^2$.

5.3.2 Trackability

To reiterate from Chapter 2, trackability refers to the reported number of gaze samples as a percentage of the expected number of valid samples. Whereas the precision and accuracy were determined utilising the stable 250 ms of data, trackability was evaluated using a 500 ms period of data in the middle of the two second period. This was done to ensure an equal number of samples for each participant, as the filtered data lengths varied between participants. Moreover, by using the middle of the two second period there is a high likelihood that the participant was looking at the dot. Trackability was calculated using the following formula:

$$Trackability = Sample\ Count / \frac{Frequency}{2} \times 100$$

5.3.3 Selection of calibration points

For the purpose of this study, a selection of 14 of the 40 points was used as "calibration" points on which the regression was based. The remaining points were used to validate the results. Figure 5.1 below shows the spread of points on the stimulus. The calibration points are represented by the blue dots.

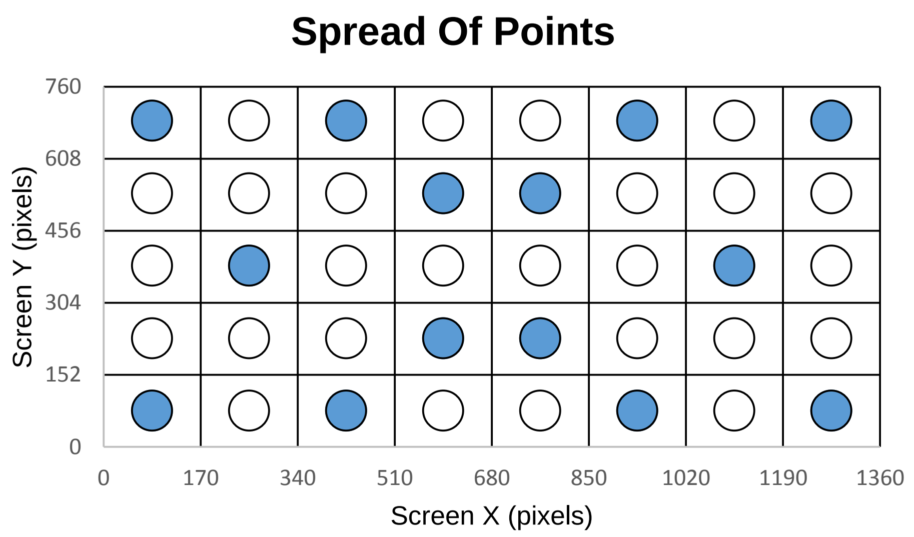


Figure 5.1 - Spread of gaze dots over the stimulus. The points represent the dots where participants were required to fixate. Blue dots represent the dots that were used for regression.

5.3.4 Statistical analysis

The raw data files resulting from the experiment were extracted and collated in a MySQL database. From there, the actual fixation data sets were extracted and analysed using a statistical software package, namely Statistica 12. ANOVA tests were used to determine the significance ($\alpha = .05$) of all results. Tukey's Unequal HSD (honest significant difference) was used to determine the significance (α = .05) of differences between individual combinations of sampling frequencies and head positions.

5.3.5 Demographics

A total of thirty-one participants were tested during the first phase of the experiment (Table 5.1). During the second phase, thirty of the thirty-one participants from the first phase where tested. Out of the thirty, a single set of data was discarded due to an error in the test procedure. In terms of gender representation, eighteen of the thirty-one were male and thirteen female. Two ethnic groups were tested, namely African and Caucasian. The number of participants in each group was 19 and 12 respectively. Only two of the participants wore contact lenses, and three participants wore mascara.

Table 5.1 – Summary of participant demographics

	Male	Female	Contact lenses	Mascara
African	12	7	1	1
Caucasian	6	6	1	2

The following sections report the results of each test. The results are reflected in terms of the three respective phases of the experiment. The first series of results are related to the effect of sampling frequency on <u>precision</u>, <u>accuracy</u> and <u>trackability</u>. A summary of the various frequencies can be found in Chapter 4, Table 4.1. The second section is concerned with the effect of the head position on <u>precision</u>, <u>accuracy</u> and <u>trackability</u>. Finally, the results of the theoretical performance using artificial eyes, are discussed.

5.4 Results related to sampling frequency

This section of results is concerned with the effect that the sampling rate had on the data quality of the eye tracker. The formally stated research hypothesis is as follows:

- $H_{0,1}$: The sampling frequency has no effect on data quality (precision, accuracy and trackability).

The frequency was adjusted between each test, increasing by 50 Hz per step. The values ranged from 50 Hz to 300 Hz. The position of the eye tracker was kept constant in relation to each participant.

5.4.1 Sampling frequency vs. precision

From the research hypothesis regarding sampling frequency, the first of the sub-hypotheses states that the sampling frequency has no effect on the precision obtained. As can be seen from Figure 5.2 below, there was a difference in precision between sampling rates. To verify whether or not this difference was significant, a one-way ANOVA test was conducted.

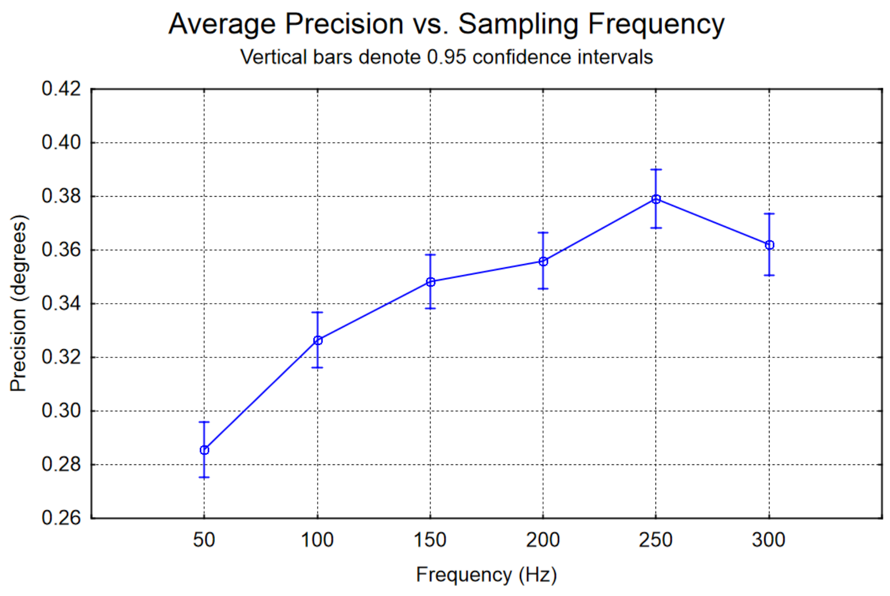


Figure 5.2 - The overall trend of precision with respect to changes in sampling rate

The results of the one-way ANOVA indicate that the sampling frequency did indeed have a significant effect on precision ($\alpha = .05$) and thus $H_{0,\ 1.1}$ is rejected. Tukey's post-hoc analysis confirmed significant differences in precision for all pairs of frequencies, with the exception of 100 Hz to 150 Hz and 250 Hz to 300 Hz, where there was no significant difference in precision. Also evident in Figure 5.2 is that higher sampling frequencies (200 Hz and above) yielded precision between 0.34° and 0.38°, whilst the best precision was obtained at lower sampling rates – in this case 50 Hz (~0.29°).

The graph (Figure 5.3) details the average precision over all participants, as well as the best and worst recorded precision values of the participants for each frequency. From the graph, it can be seen that the system is capable of achieving very good precision (~0.3°) under the right conditions. Conversely, certain participants yielded very poor results.

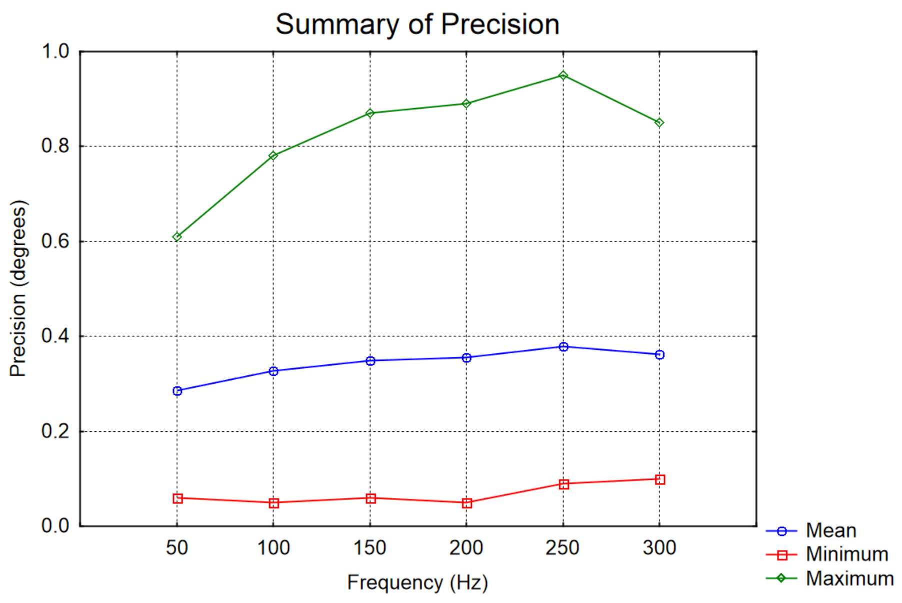


Figure 5.3 - Average, best and worst recorded precision values

5.4.2 Sampling frequency vs. accuracy

The second part of the hypothesis involves the effect that sampling frequency has on the accuracy of the HLSL tracker. Again, an ANOVA test was conducted to determine whether the sampling frequency had a significant effect on the accuracy of the system (Figure 5.4) from which it can be concluded that $H_{0,\,1.2}$ should be rejected. The results indicate that the sampling rate does indeed have an effect on the accuracy (α = .05). However, the post-hoc Tukey test indicates that the effect is only significant between 50 and 100 Hz.

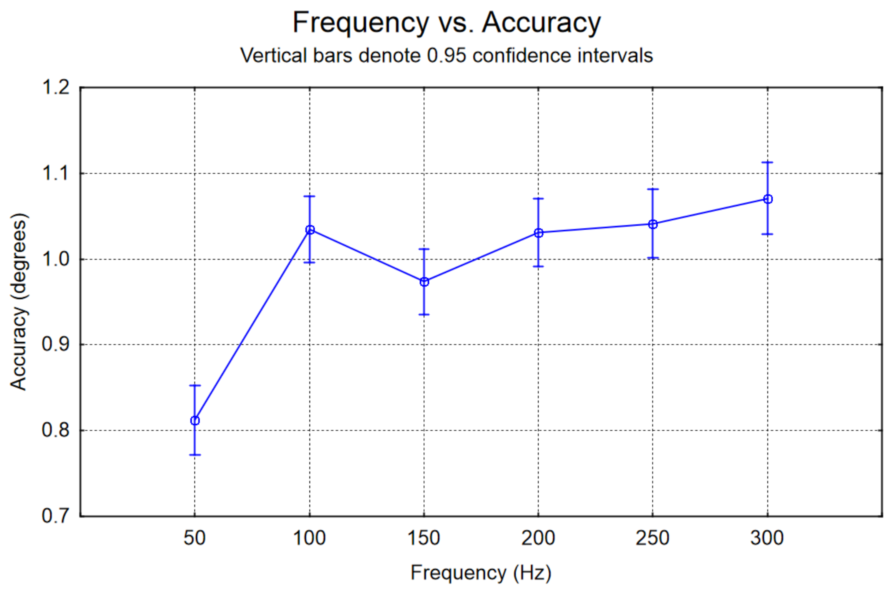


Figure 5.4 - Sampling frequency of the eye tracker measured against the average accuracy obtained

The overall average accuracy of the system was ~1° of visual angle. As was the case for precision, the best value for accuracy was obtained at 50 Hz, and the worst at frequencies above 200 Hz. In contrast to precision, however, all other frequencies yielded comparable accuracies.

As illustrated by Figure 5.5 below, it is possible to achieve accuracies of around ~0.5° for certain participants. Conversely, for certain participants the HLSL tracker reported very poor accuracy (~3°).

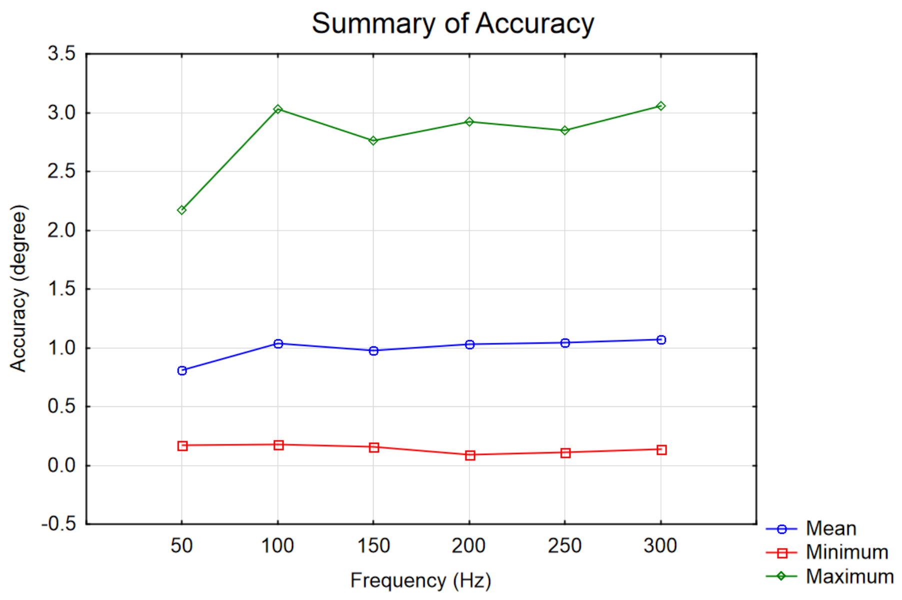


Figure 5.5 - Average, best and worst accuracy obtained for each sampling frequency

5.4.3 Sampling frequency vs. trackability

The final section concerning the effect of the sampling frequency on data quality focused on the trackability of the HLSL tracker over various sampling frequencies. The primary interest of the trackability measurements was to establish what percentage of sampled frames yielded a valid gaze coordinate. The trackability analysis was not concerned with the quality of the gaze coordinates, but only with the validity. Hence, any valid pupil glint vectors were counted, as they would yield a gaze mapping. As is evidenced by the results in Figure 5.6, all frequencies yielded similar values for trackability (~98%). Again, an ANOVA was conducted to evaluate whether or not the differences were significant.

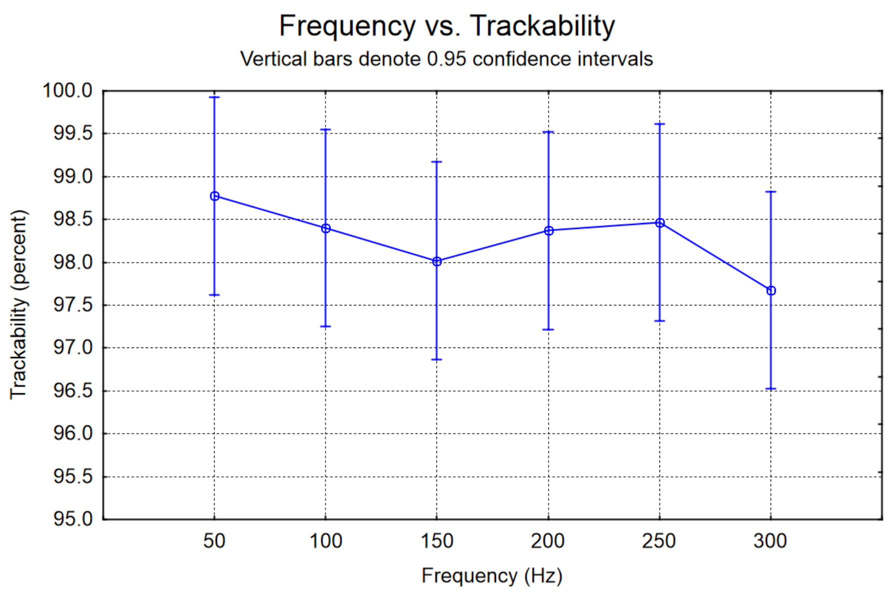


Figure 5.6 - Trackability of the HLSL tracker for each sampling frequency

In contrast to the previous two results, the results of the ANOVA indicated that the sampling frequency had no significant effect on the trackability, and therefore $H_{0,\ 1.3}$ cannot be rejected. This was confirmed by the Tukey post-hoc analysis for all frequencies.

5.4.4 Conclusion

In summary, in this section it was determined that the sampling frequency had a significant effect on both the obtainable precision and accuracy leading to the rejection of $H_{0,\ 1.1}$ and $H_{0,\ 1.2}$. However, there was no significant effect on the trackability.

In the following section, the results of the tests involving different head positions in relation to the eye tracker are provided.

5.5 Results related to head position

This section details the effect that various head positions have on the precision, accuracy and trackability of the system. The assumption made in the hypothesis is that the head position has no effect on the precision, accuracy and trackability of the HLSL tracker. To review from Chapter 4, the head positions were simulated by adjusting the position of the eye tracker relative to the participant (Chapter 4, Table 4.1). The eye tracker was adjusted by 5 cm to the left and right (x axis), 3 cm up and down (y axis), and 5 cm backward and forward (z axis). For the duration of this experiment, the sampling frequency was kept at 100 Hz, and the size of the eye video at 1000×400 pixels.

5.5.1 Head position vs. precision

The first investigation in this section is to evaluate the effect that different head positions have on the precision of the system. The assumption made in the hypothesis was that the head position had no effect on the precision. As is evidenced by the results in Figure 5.7, however, it appears that the precision does change in response to different head positions and that $H_{0,\ 2.1}$ should therefore be rejected. Precision appeared to improve with horizontal head movements, and worsen with vertical head movements. From the figure it also appears that precision increases as the eye tracker is moved closer to the participant.

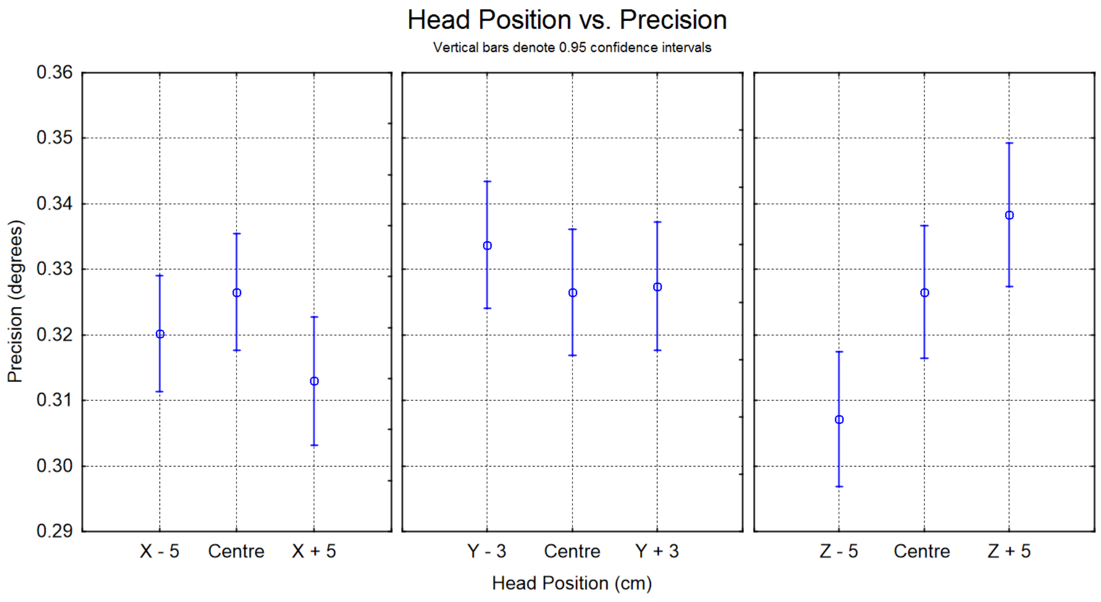


Figure 5.7 - Precision as affected by head position

To determine whether or not these changes were significant, an ANOVA was conducted. The results of the ANOVA test indicate that the overall effect of head position on precision was significant ($\alpha = .05$) thus leading to the rejection of $H_{0,\ 2.1}$. Tukey's post-hoc test indicated, however, that none of the individual head movements had a significant effect on the precision when compared to the central position. This indicates that the image processing algorithms are fairly robust toward head movements, as precision is related to the stability of the detected feature points in the eye video.

5.5.2 Head position vs. accuracy

The second assumption made under the hypothesis was that the head position had no effect on the accuracy of the eye tracker. From Figure 5.8 below it is apparent that the accuracy differed for each head position, as is expected from regression-based gaze estimation (see Chapter 2, Section 2.10.1). In contrast to precision, the accuracy decreased as the eye tracker was moved closer to the participant. This was confirmed by a one-way ANOVA test which revealed that the head position had a significant effect ($\alpha = .05$) on the average accuracy that was obtained, and thus $H_{0,\,2.2}$ is be rejected.

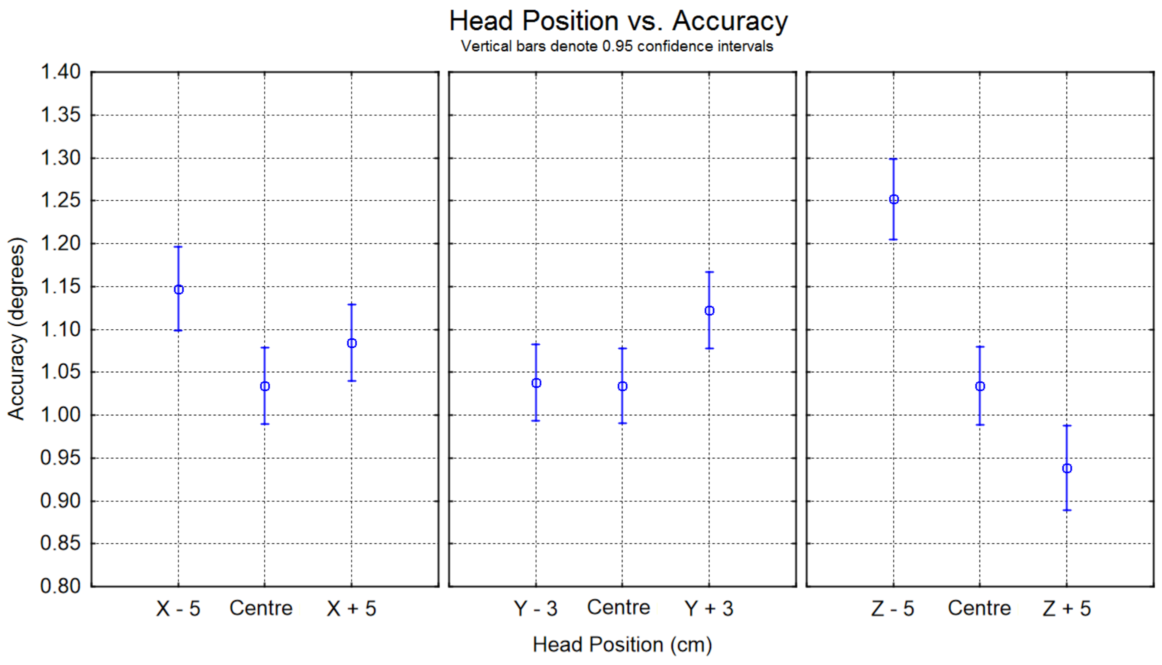


Figure 5.8 - Accuracy for each head position

A Tukey Unequal HSD post-hoc test was used to determine whether the effect of the new head position was significant in relation to the central head position. The results of this test indicate that only head movements to the left (X-5) and forward (Z-5) were significant. The lack of symmetry in the results for the axes can be attributed to an offset in the corneal reflections due to the presence of secondary reflections generated by the light sources present in the testing facility during this phase of the experiment, as the position of the table on which the eye tracker was mounted was not positioned in the centre of the two light sources used to regulate the ambient light.

5.5.3 Head position vs. trackability

The final investigation in this section examines the trackability of the system in response to head movements. The ANOVA test revealed that the head position had no significant effect on the trackability of the system (Figure 5.9).

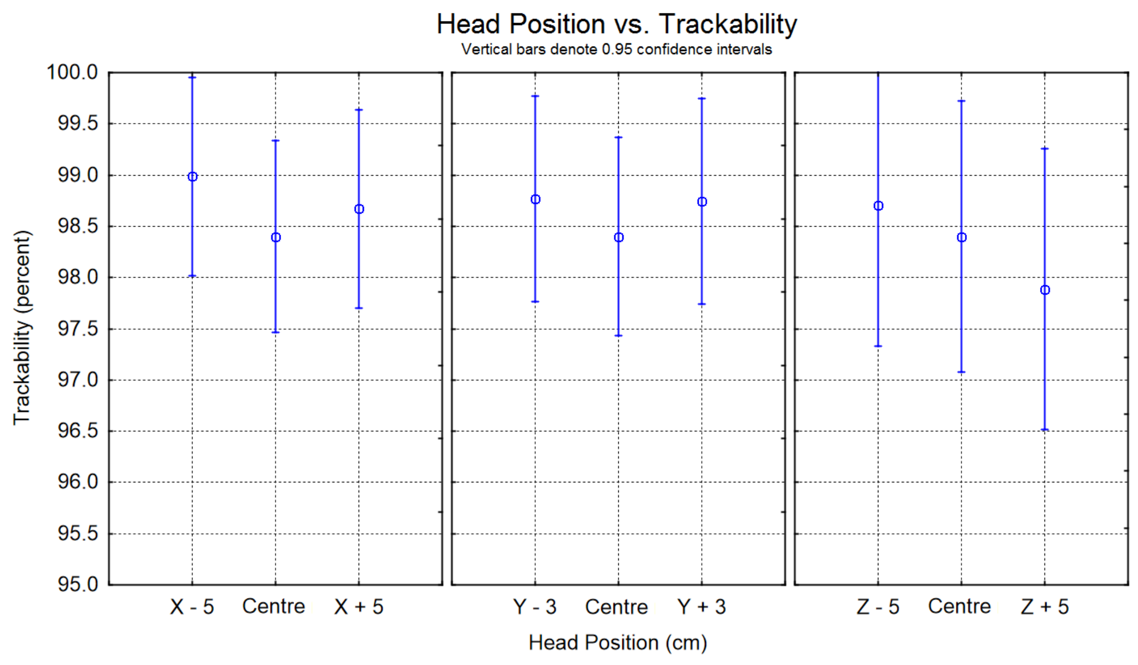


Figure 5.9 - Trackability of the HLSL tracker at various head positions

As with the previous two cases, a Tukey post-hoc verified the significance of trackability for head positions in relation to the central position. The results showed that there was no significant difference on any of the axes and from this it is concluded that $H_{0,\,2.3}$ cannot be rejected.

5.5.4 Conclusion

In this section, it was determined that the position of the head generally had a significant effect on both the precision and the accuracy leading to the rejection of $H_{0,\,2.1}$ and $H_{0,\,2.2}$. The head position had no significant effect on the trackability, however.

In the following section, the results of the experiment using artificial eyes will be reported.

5.6 Artificial performance

In theory, artificial eyes should yield precision values of 0°, as they remain completely still. However, eye tracking systems contain noise. The purpose of the artificial eyes test was to investigate the precision obtainable if the head movements and eye tremors present in human participants were completely eliminated, thereby determining the amount of noise produced by the eye tracking system. Furthermore, the theoretical performance of the system at higher sampling rates was also investigated. Hence, an additional sampling rate of 500 Hz was used. With the exception of 500 Hz, all sampling rates used were the same as in the first phase of the experiment (50 Hz -300 Hz). The calibration data from one of the human participants was selected to perform regression in order to compute the precision values using the mapping polynomials described in Chapter 3 Section 3.10.

5.6.1 Frequency vs. precision in the instance of artificial eyes

The first simulation involved the effect of higher sampling rates on precision with artificial eyes. From Figure 5.10 it is clear that the precision worsens from 250 Hz. To verify the significance of this decrease in precision, an ANOVA was conducted. As with the test involving human participants, the difference in precision was significant. The Tukey post-hoc confirmed that the difference in precision was significant between all pairs of sampling rates, with the exception of 300 Hz to 500 Hz, where no significant difference was found. The precision obtained at 500 Hz was roughly 0.15°.

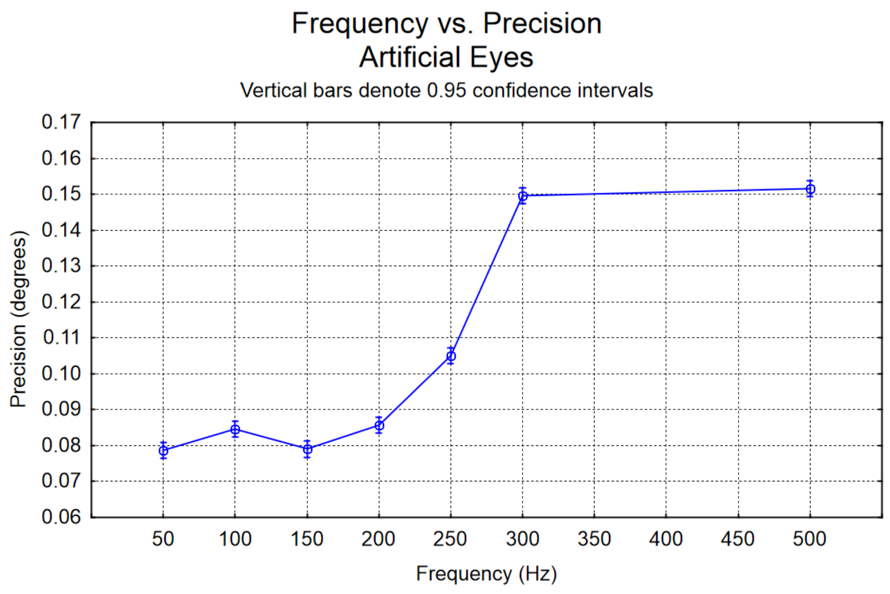


Figure 5.10 - Degree of precision obtained versus sampling rates for artificial eyes

5.6.2 Head position vs. precision in the instance of artificial eyes

The final test involving artificial eyes was to determine the effect that the head position had on the precision (Figure 5.11). The test was performed in the same way as the human participant equivalent, with the table being moved on its axes in order to simulate the various head positions. A one-way ANOVA and Tukey post-hoc were used to evaluate the results.

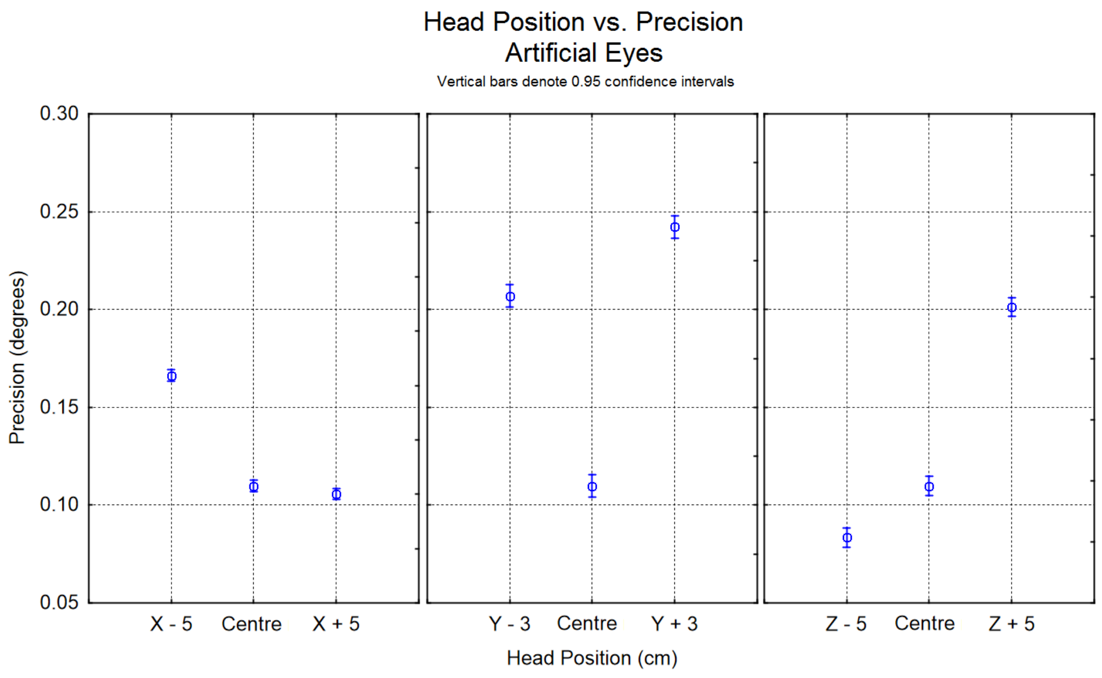


Figure 5.11 - Precision of artificial eyes as affected by head positions

From Figure 5.11 it can be observed that there is a change in the precision at different head positions. The best precision was observed when the eye tracker was positioned closer to the artificial eyes (~0.08°). The significance of the different precision values was tested with an ANOVA, which revealed that the different values were significant ($\alpha = .5$). This was confirmed by the Tukey post-hoc test, which revealed that the changes in precision were significant when compared to the centre head position. The exception to this was the X + 5 head position, where the difference in precision was not significant. These results are in stark contrast to those produced by the human participants, where there was no significant difference in precision for different head positions

5.6.3 Conclusion

It was shown that both the sampling rate and head position of the artificial eyes had an effect on the precision obtained. An additional sampling rate of 500 Hz showed similar precision to the 300 Hz precision, though the decline in precision from 250 Hz suggests that the overall precision of the system would continue to worsen as the sampling frequency increased.

The following section discusses general performance in terms of participants and data quality across the stimulus.

5.7 General performance remarks concerning high frequencies

As this dissertation is focused on high frequency eye tracking utilising the GPU for image processing, an additional discussion is provided regarding the specific performance values for frequencies exceeding 200 Hz.

5.7.1 Tolerance toward participants

For frequencies from 200 Hz to 300 Hz the features of all participants were successfully localised. This was true even in cases where participants wore mascara or contact lenses.

5.7.2 Precision across participants

Figure 5.12 below shows the distribution of precision values across participants for frequencies of 200, 250 and 300 Hz. Note that participants whose results fell outside the ranges were excluded from the graphs. The graph suggests that the majority of participants yielded precision values between 0.3° and 0.4°.

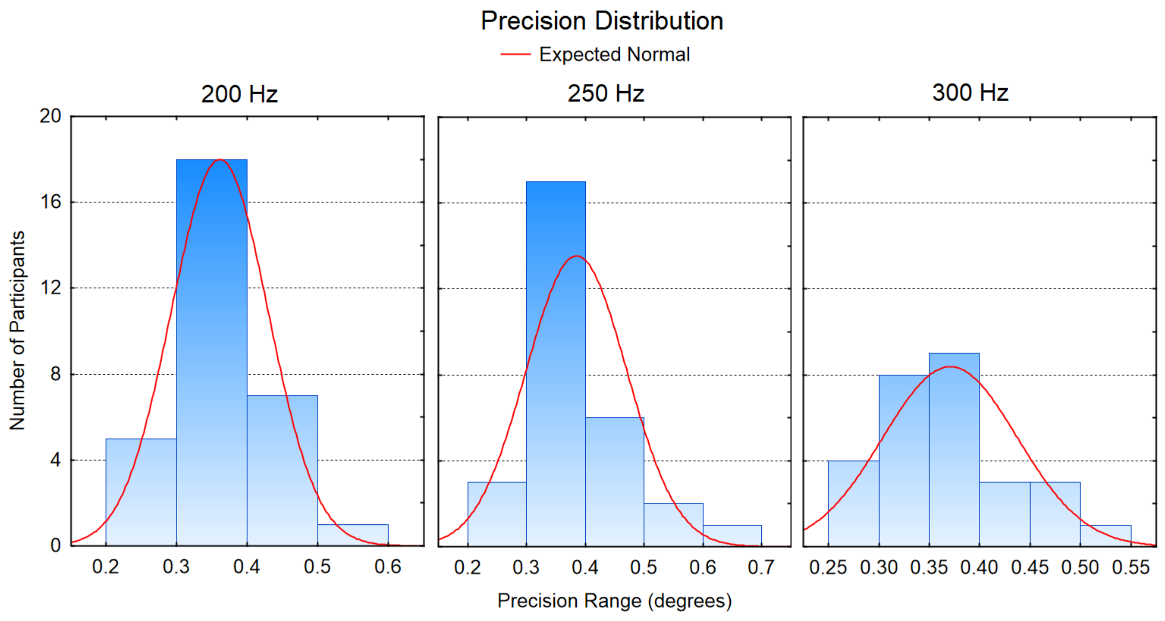


Figure 5.12 - Number of participants that fell within precision ranges

5.7.3 Precision as affected by location on the stimulus

An ANOVA test was conducted to determine the significance of precision values as affected by their corresponding location on the stimulus in terms of sampling frequency. Accuracy was not included in this discussion, as it is well established that accuracy tends to decrease as one moves toward the edges of the screen (Holmqvist et al., 2011). Figure 5.13 below shows the distribution of precision across all participants for frequencies of 200 Hz, 250 Hz and 300 Hz respectively. The graphs suggest that the sides of the screen produced the best precision.

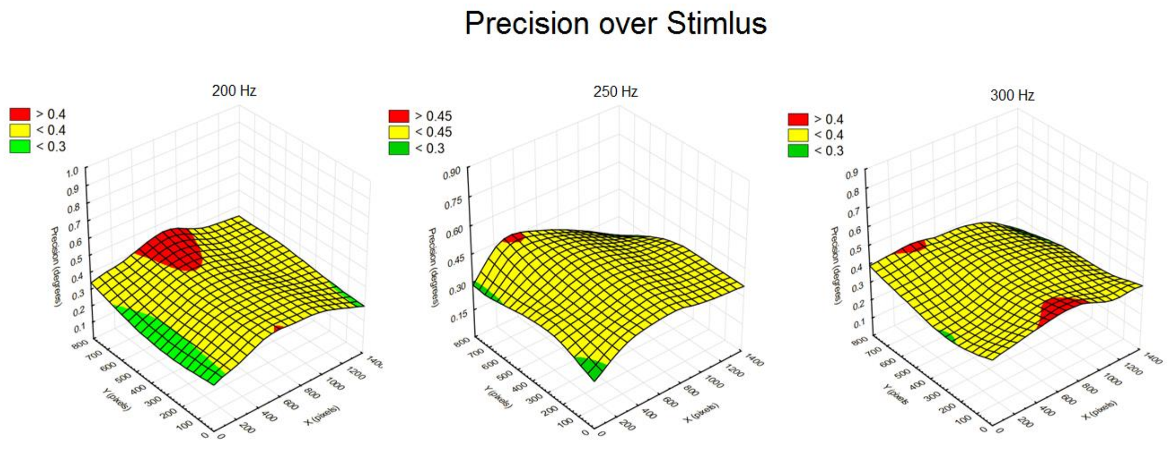


Figure 5.13 - Average precision over the various parts of the screen grouped by sampling frequency

The ANOVA test revealed that the location on the screen had no significant effect on the precision obtained. However, Tukey’s post-hoc analysis revealed that the effect was significant ($\alpha = .05$) when moving towards the centre of the screen.

5.8 GPU performance and overhead

In addition to the various eye tracker performance metrics that were examined, the performance of the GPU was also evaluated by looking at the maximum attainable sampling rate for a variety of eye video sizes. This was done to gauge the amount of overhead that is incurred as the size of the image is increased. The figure below shows the maximum attainable frame rate of standard resolution (4:3 aspect ratio) images. The entire image is processed on the GPU using the pre- and post-process shaders described in Chapter 3.

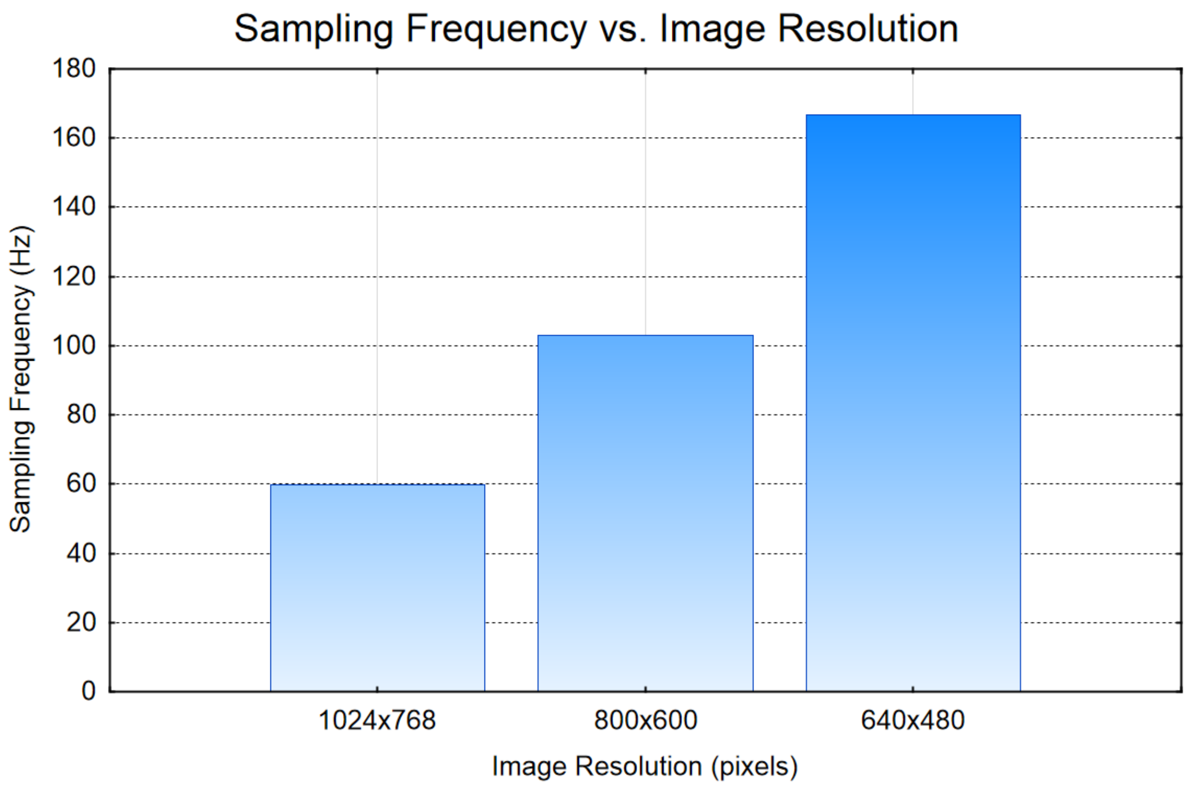


Figure 5.14 - Maximum attainable sampling frequency for standard resolutions

In Chapter 3 it was mentioned that one of the pitfalls of utilising the GPU was the overhead incurred by the copy between system memory and that of the GPU. As evidenced in Figure 5.14, the overhead causes a significant decrease in the attainable sampling rate. For a standard resolution of 1024×768, the system is capable of only 60 Hz. However, the graph does imply that even small changes in the resolution of the image can greatly increase the sampling rate. The entire eye localisation process is performed on a 640×480 image at over 160 Hz. Given that much of the y component of the eye video is wasted, as the participant's vertical head movements are unlikely to be as large as the horizontal ones, it is logical to assume that the y component of the image can be decreased without forfeiting a significant part of tolerance toward head movements. With this in mind, the simulation also examined a number non-standard resolutions, optimised to maximise the sampling rate while maintaining some semblance of support for head movements. These resolutions are shown in Figure 5.15 below.

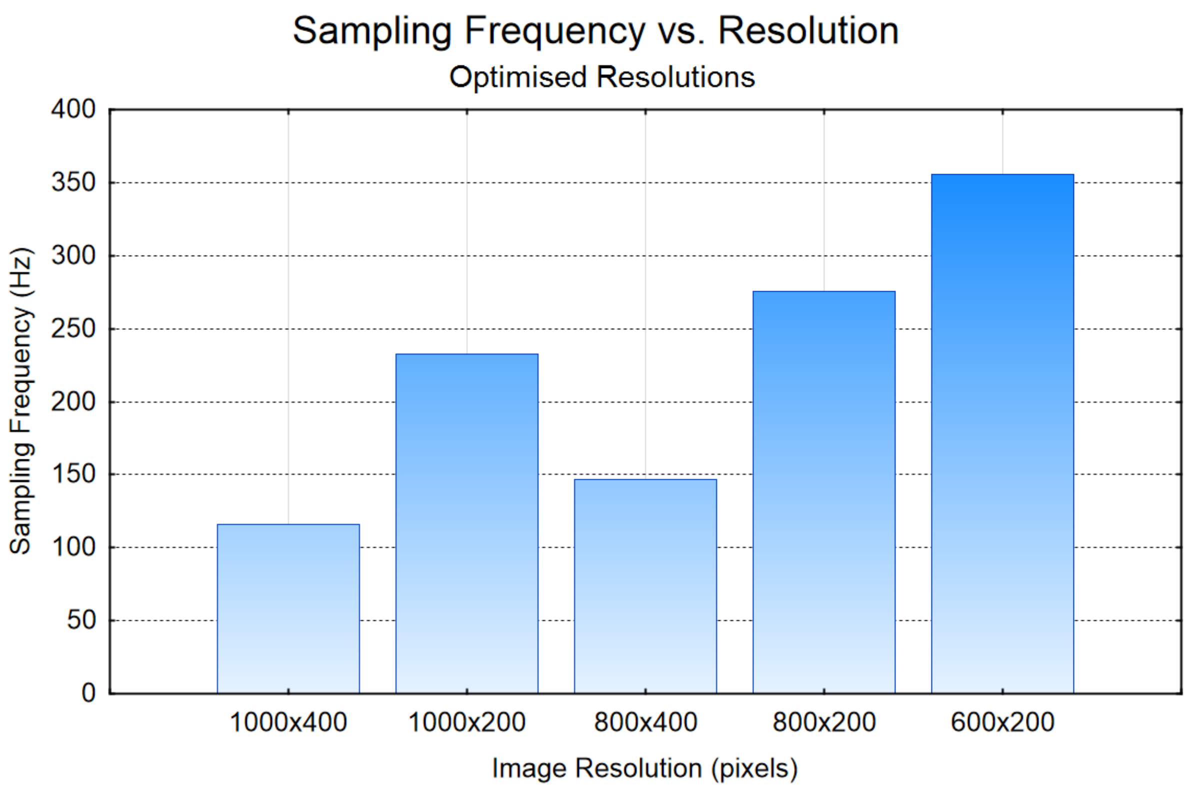


Figure 5.15 - Reworked eye images and the resulting sampling rates attainable

As is evident from the figure above, there is a marked increase in the attainable sampling rate. With a resolution of 600×200, it is possible to sample at rates in excess of 300 Hz (500 Hz was achieved with 650×150 pixels). In practical terms, the resolution of 1000×400 pixels offers a similar amount of tolerance toward head movement as the 1024×768 resolution in Figure 5.14 does, but with a frame rate (120 Hz) roughly twice that of the latter (60 Hz). From the graph above, it is reasonable to assume that the resolution of 1000×200 pixels is the logical choice, as it offers the largest amount of horizontal and vertical head space, with a sampling frequency of close to 250 Hz.

5.9 Conclusions

In this chapter, the results of the experiments described in Chapter 4 have been provided. The effects of sampling frequency and head position on data quality measures have been determined. The theoretical performance of the eye tracker has been tested using artificial eyes and simulations on various image sizes using the GPU have shown the maximum attainable sampling in practical terms.

In the ensuing chapter (Chapter 6), the final conclusions regarding the HLSL tracker that has been developed and evaluated are made, and a summary provided of the preceding chapters.

CHAPTER 6: CONCLUSION

The objective of this concluding chapter is to provide a synopsis of the entire study and to re-examine the validity of the formulated hypotheses on the basis of the statistical analysis of data presented in the previous chapter. Also included is a critical evaluation of the role that the Graphical Processing Unit (GPU) played in the HLSL tracker that was developed for the purpose of the research. The chapter concludes with a number of recommendations for future improvements to the eye tracker.

6.1 Motivation

The motivation for the HLSL tracker stemmed from two needs that were identified in Chapter 1. Firstly, eye tracking has numerous applications in various fields such as psychology, neuroscience and linguistics. However, certain of these application areas require high sampling rates and the use of expensive commercial high-speed trackers. This provided the impetus to develop a low-cost alternative high-speed eye tracker. Secondly, although low-cost alternative trackers are available, many of these eye trackers have low sampling frequencies making them unsuitable for use in some fields. The challenge was thus to generate a solution that was both affordable and that could provide high sampling rates.

6.2 Goals

From the requirements identified, the specific goals of the eye tracker were defined. The primary aim was to develop a low-cost eye tracker capable of attaining high sampling rates while maintaining a good measure of data quality (accuracy, precision and trackability). In order to meet the objective of keeping the cost of the overall system as low as possible, the eye tracker should be able to run on a variety of mid-range computer systems. In addition hereto, the solution designed should offer a degree of support towards head movements while simultaneously providing ease of use and comfort. A related goal would be to identify a localisation method that could be used in real time and at high sampling rates. In view of the lack of research on employing the GPU in eye tracking, a secondary goal of the study was aimed at incorporating the strengths of the GPU into the eye tracking process with a view to accelerating some of the image processing tasks.

6.3 Results

Each of the goals had research questions associated with them. These questions were investigated using an experiment conducted in a laboratory setting. Since the results of the experiment were discussed in detail in the previous chapter, the research questions will now be re-examined in light of the results, and conclusions will be drawn about the effectiveness of the HLSL tracker.

6.3.1 Cost of the eye tracker

One of the goals stipulated was that the system should be affordable, i.e. present a low-cost alternative. As stated in Chapter 3, the required sampling frequency prohibits the use of low-cost hardware such as web cameras. However, relative to the cost of commercial systems, the hardware components used in the HLSL tracker are still affordable, amounting to a cost of less than 700 euros in total. Moreover, with the exception of the requirement of support for USB 3.0, the system requirements for the HLSL tracker can be considered mid-range. Given that the HLSL tracker was tested with a mid-range laptop using an integrated display adapter (Intel HD 4000) and was capable of running the application without issues, one can therefore conclude that the goals in terms of the overall cost of the system have been met.

6.3.2 Quality of reported data

In addition to cost, the goals of the eye tracker also stipulated that it should be able to sample data at frequencies in excess of 200 Hz, and that the system should report a good precision and accuracy. The eye tracker should also provide tolerance towards head movements.

6.3.2.1 Sampling rate

From the results in Chapter 5, it was determined that the eye tracker could run at a variety of different sampling rates, with the maximum sampling rate tested with human participants being 300 Hz. While accuracy and precision were affected by the sampling rate, the trackability remained unaffected, rendering invalid the two sub-hypotheses that sampling frequency would have no effect respectively on the precision and accuracy of the eye tracker. The results indicate that the eye tracker could attain precision and accuracy of around 0.36° and 1.1° respectively at the 300 Hz sampling rate. Of the three highest sampling frequencies, it is apparent that the accuracy and precision is best when sampling at 200 Hz (0.36° and 1.1° respectively). The test on artificial eyes supports this claim, with 200 Hz yielding a precision of about 0.1°, and higher sampling rates providing significantly worse precision. From this one can conclude that the optimal sampling rate of the HLSL eye tracker is 200 Hz.

6.3.2.2 Head position

The HLSL tracker's tolerance towards head movements was also evaluated, as well as the effect that head movements had on the precision, accuracy and trackability of the eye tracker. The results showed that the head position in relation to the centre had no effect on the precision, indicating that the system did offer some tolerance towards head movements within a head box of 10×6×10 cm. The same held true for trackability. Accuracy was affected by the head position, however, as was expected given that a regression-based mapping estimation model was used.

6.3.3 Effectiveness of the GPU

The novelty of the HLSL tracker lies in the use of the GPU and DirectX to perform certain image processing tasks. However, as mentioned previously in Chapter 3, and from the results in Chapter 5, it is apparent that there is a significant overhead involved in utilising the GPU. This raises questions about the purpose of using the GPU. Figure 5.14 in Chapter 5 demonstrates that using the GPU with large images (1024×768) yields a maximum sampling frequency of only 60 Hz. However, this is comparable to other current low-cost alternatives suggesting that there is very little downside to using the GPU in the eye tracking process.

The results do indicate that decreasing the vertical resolution of the eye images has an effect on the maximum obtainable sampling rate. As a large vertical resolution is not required to tolerate moderate vertical head movements, this can be done safely without decreasing the size of the head box by a significant amount. Figure 5.15 in chapter 5 shows that halving the vertical resolution to 400 pixels effectively doubles the maximum sampling frequency to 120 Hz. Decreasing the vertical resolution further to 200 pixels yields a maximum sampling rate of close to 240 Hz. With this in mind, it can be concluded that with the correct selection of eye video resolution, the GPU can be used in the eye tracking process and a high sampling frequency still be obtained despite the overhead.

6.3.4 Optimal head box size

It has already been determined that the optimal sampling frequency of the HLSL tracker is 200 Hz. To determine the eye video size, it is necessary to consider the head box size. From the experiment involving head positions, it was shown that a resolution of 1000×400 pixels gives a head box of 10×6×10 cm. However, two issues arise from the selection of this resolution. Firstly, the performance of the system was evaluated at 100 Hz for these head positions, which means that increasing the sampling rate may have an additional effect on the accuracy at different head positions. The results involving different sampling frequencies revealed that from 100 Hz and higher, the difference in accuracy was no longer significant. From this it can be concluded that increasing the frequency to 200 Hz will not have an effect on the accuracy within the head box. The accuracy obtainable then is determined by the choice of gaze estimation model. The second issue with this head box size is that the GPU simulations reveal that the maximum sampling rate of this resolution is 120 Hz. The next closest resolution tested was 1000×200 pixels, which has a maximum sampling rate of 240 Hz. The vertical resolution of this image size does mean that there is less tolerance towards vertical head movements, however. From this it can then be concluded that the optimal eye video resolution lies between 1000×400 and 1000×200 pixels, i.e. potentially at 1000×300 pixels.

In summary, based on the results of the experiments and simulations, the optimal setup of the HLSL tracker is a maximum sampling frequency of 200 Hz, with an eye video size of around 1000×300 pixels, and head box size of approximately 10×6×10 cm.

6.4 Implications

The implications of these results and conclusions to be reached are that at a cost of under 700 euros it is possible for researchers to perform eye tracking at a sampling rate of around 200 Hz with a tolerance towards head movement within an area of 10×6×10 cm. This is provided that the computer system running the HLSL tracking software has comparable system specifications to that of the laptop used during the experiment, and has a USB 3.0 port and a DirectX 9 compatible display adapter. Performance figures of ~0.3° precision can be expected at this rate, unless smoothing is done at the cost of higher latency. As raw data points are reported by the eye tracker, researchers are free to use whichever gaze estimation method is preferred. The gaze estimation method (i.e. regression based versus geometric) will ultimately have the largest effect on the accuracy obtained, and is therefore not relevant to this discussion.

Additionally, researchers who are less interested in allowing head movements can decrease the size of the eye video in return for a higher sampling rate. At a resolution of 600×200 pixels a frequency of 350 Hz is attainable. It is also worth mentioning that using a more powerful computer can also have an effect on the maximum sampling rate. When run on an Alienware 17[1] laptop, for example, the resolution of 600×200 pixels yielded a sampling rate of close to 700 Hz. Given the gradual decrease in precision at higher frequencies, however, it may be unwise to utilise the HLSL tracker at frequencies higher than the optimal rate of 200 Hz.

In terms of the inclusion of the GPU in the eye tracking process, it has already been argued that there is little downside to its inclusion, as even the larger image sizes yield sampling frequencies that are usable for certain eye tracking studies. However, for high frequency eye tracking the role of the GPU is less certain. While the actual image processing functions are very fast, the large amount of overhead incurred when copying images to and from the graphics card imposes severe performance limitations on the size of eye video that can be processed while maintaining a high sampling rate. To justify this overhead, the image processing functions performed should be of such a nature that they are significantly slower when performed on the CPU only. As this study did not compare the CPU and GPU image processing functions, it is difficult to draw conclusions on the actual performance of the GPU. Given that the image processing functions performed in the HLSL tracker were fairly straightforward – a requirement imposed by the use of HLSL – it cannot be concluded that the GPU had any benefit in the eye tracking process in this case. However, the conclusions of Mulligan (2012) suggest that the GPU itself may be beneficial to the eye tracking process.

When considering the performance of the CUDA-based implementation of Mulligan (640×480 pixels at 250 Hz), it is reasonable to suggest that the issue with the HLSL tracker is not the use of the GPU, but rather the choice of implementation language, namely DirectX and HLSL. CUDA is a language developed specifically for general purpose computations on the GPU, whereas DirectX was developed primarily with games and 3D graphics in mind.

6.5 Further research

There are several aspects of the HLSL tracker that can be improved. Arguably, the most obvious area for improvement is the way in which data is copied to and from the GPU. If only

[1] Intel i7-4710MQ CPU running at 2.5 GHz with 16 GB DDR3 1600 MHz memory.

areas of the image that are necessary are processed, it may be possible to utilise the GPU using HLSL with little to no overhead, as the results suggest that the overhead decreases substantially as the image gets smaller. However, given the increase in computing power that takes place each year, it is reasonable to assume that even decreasing this overhead may not be enough to warrant the inclusion of the GPU using HLSL in the eye tracking process.

Further improvements could be made to increase the precision obtainable. Research suggests that this can be done by increasing the resolution of the eye video (Holmqvist et al., 2011). However, because of the fact that higher resolutions will increase the overhead incurred by the GPU, it may be preferable to increase the magnification of the eye by using a different lens on the camera while keeping the 1300×200 pixel resolution. In this way, only the head movement is traded, while precision is potentially increased at no cost to sampling frequency.

An additional aspect to research further is the hardware components that have an influence on the time it takes to transfer data to and from the GPU. The test with the Alienware laptop suggests that the use of faster memory or the use of a more powerful display adapter may play a role in this process.

Lastly, the effects that extremely high sampling frequencies have on precision and accuracy can be examined further in light of the results obtained in this study.

6.6 Summary

The study has presented an eye tracking solution that utilises the GPU to perform image processing at high sampling frequencies. Utilising Microsoft’s DirectX and HLSL, an eye tracking application was created based on shortcomings identified in the literature. The resulting eye tracking application – referred to as the HLSL tracker – was evaluated through an experiment. While the performance of the eye tracker is not superior to that of Mulligan (2012), the results show that it is possible to obtain sampling rates in excess of 200 Hz using mid-range computing power and HLSL.

REFERENCES

Andersson, R., Nyström, M., & Holmqvist, K. (2000). Sampling frequency and eye-tracking measures : how speed affects durations , latencies , and more. *Journal of Eye Movement Research, 3*(3), 1–12.

Archirapatkave, V., Sumilo, H., See, S. C. W., & Achalakul, T. (2011). GPGPU acceleration algorithm for medical image reconstruction. In *IEEE International Symposium on Parallel and Distributed Processing with Applications* (pp. 41–46). Busan , 2011.IEEE.

Beelders, T. R., & Blignaut, P. J. (2014). Gaze and speech: pointing device and text entry modality. In M. Horsley, M. Eliot, B. A. Knight, & R. Reilly (Eds.), *Current Trends in Eye Tracking Research* (pp. 51–75). Cham: Springer International Publishing.

Beymer, D., & Flickner, M. (2003). Eye gaze tracking using an active stereo head. In *Proceedings of the 2003 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'03)*. Madison, Wisconsin, June 16-22, 2003. IEEE.

Bjorke, K. (2005, May). *Introduction to image processing on the GPU*. (NVIDIA Development). Last accessed January 30, 2015, from NVIDIA Developer Site: http://developer.download.nvidia.com/shaderlibrary/docs/GPU-Img-Proc-Intro.pdf

Blignaut, P. (2009). Fixation identification : The optimum threshold for a dispersion algorithm. *Attention, Perception, & Psychophysics, 71*(4), 881–895.

Blignaut, P. (2013). Mapping the pupil-glint vector to gaze coordinates in a simple video-based eye tracker, *Journal of Eye Movement Research , 7*(1), 1–11.

Blignaut, P., & Beelders, T.R (2009). The effect of fixational eye movements on fixation identification with a dispersion-based fixation detection algorithm. *Journal of eye movement research, 2*(5), 1-14.

Blignaut, P., & Beelders, T.R (2012). The precision of eye-trackers : A case for a new measure. In *Proceedings of the Symposium on Eye Tracking Research and Applications.* (pp. 289–292). Santa Barbara, California, 2012. ACM: New York.

Blignaut, P., & Wium, D. (2014). Eye-tracking data quality as affected by ethnicity and experimental design. *Behavior Research Methods, 46*(1), 67–80.

Böhme, M., Meyer, A., Martinetz, T., & Barth, E. (2006). Remote eye tracking : State of the art and directions for future development. In *The 2nd Conference on Communication by Gaze Interaction : Gazing into the Future* (pp. 12–17). Turin, Italy, September 4 – 5, 2006.

Bowling, A., & Draper, A. (2014). Using saccadic eye movements to assess cognitive decline with ageing. In M. Horsley, M. Eliot, B. A. Knight, & R. Reilly (Eds.), *Current Trends in Eye Tracking Research* (pp. 237–244). Cham: Springer International Publishing.

Bulling, A., & Gellersen, H. (2010). Toward mobile eye-based human-computer interaction. *IEEE Pervasive Computing*, *9*(4), 8–12.

Cerrolaza, J. J., Villanueva, A., & Cabeza, R. (2008). Taxonomic study of polynomial regressions applied to the calibration of video-oculographic systems, In *Proceedings of the Symposium on Eye Tracking Research and Applications* (pp. 259 – 266). Savannah, Georgia, 2008. New York:ACM.

Cerrolaza, J. J., Villanueva, A., Villanueva, M., & Cabeza, R. (2012). Error characterization and compensation in eye tracking systems, In *Proceedings of the Symposium on Eye Tracking Reseach and Applications* (pp. 205–208). Santa Barbara, California , 2012. New York:ACM.

Clarke, A.H., Ditterich, J., Drüen, K., Schönfeld, U., & Steineke, C. (2002). Using high frame rate CMOS sensors for three-dimensional eye tracking. *Behavior Research Methods, Instruments, & Computers : A Journal of the Psychonomic Society, Inc*, *34*(4), 549–560.

Coutinho, F.L., & Morimoto, C.H. (2006). Free head motion eye gaze tracking using a single camera and multiple light sources. *Brazilian Symposium on Computer Graphics and Image Processing* (pp. 171–178) . Manaus, Amazonas, October 8-11, 2006. IEEE.

Crow, T.S. (2004). *Evolution of the Graphical Processing Unit*. Master's Dissertation, University of Nevada, Reno.

Dalton, K.M., Nacewicz, B.M., Johnstone, T., Schaefer, H.S., Gernsbacher, M.A., Goldsmith, H.H., Alexander, A.L, Davidson, R. J. (2005). Gaze fixation and the neural circuitry of face processing in autism. *Nature Neuroscience, 8*(4), 519–26.

Delabarre, E. . (1898). A Method of Recording Eye-Movements. *The American Journal of Psychology, 9*(4), 572–574.

Dodge, R., & Cline, T. S. (1901). The angle velocity of eye movements. *Psychological Review*, *8*(2), 145.

Collewijn, H. (2001). Interocular timing differences in the horizontal components of human saccades, *41*, 3413–3423.

Dokken, T., Hagen, T.R., & Hjelmervik, J.M. (2007). An introduction to general-purpose computing on programmable graphics hardware. In G. Hasle, K.-A. Lie, & E. Quak (Eds.), *Geometric Modelling, Numerical Simulation, and Optimization* (pp. 123–161). Berlin, Heidelberg, New York: Springer.

Duchowski, A.T. (2002). A breadth-first survey of eye-tracking applications, *Behaviour Research Methods, Instruments, & Computers, 34*(4), 455–470.

Duchowski, A.T. (2007). *Eye Tracking Methodology (2nd ed.)*. London: Springer.

Duchowski, A.T., & Price, M., Meyer M., Orero, P. (2012). Aggregate gaze visualization with real-time heatmaps, In *Proceedings of the Symposium on Eye Tracking and Applications* (pp. 13–20). Santa Barbara, California, 2012. New York:ACM.

Dybdal, M.L., Agustin, J.S., & Hansen, J.P. (2012). Gaze input for mobile devices by dwell and gestures. In *Proceedings of the Symposium on Eye Tracking Research and Applications* (pp. 225–228). Santa Barbara, California, 2012. New York:ACM.

Ebisawa, Y., Ohtani, M., Sugioka, A., & Esaki, S. (1997). Single mirror tracking system for free-head video-based eye-gaze detection method. In *Engineering in Medicine and Biology Society* (pp. 1448–1451). Proceedings of the 19th Annual International Conference of the IEEE; Chicago, Illinois, October 30 – November 2, 1997. IEEE.

Engbert, R., & Kliegl, R. (2003). Microsaccades uncover the orientation of covert attention. *Vision Research*, *43*(9), 1035–1045.

Enright, J.T. (1998). Estimating peak velocity of rapid eye movements, *Behaviour Research Methods, Instruments, & Computers*, *30*(2), 349–353.

Fernando, R., & Kilgard, M.J. (2003). *The Cg Tutorial: The Definitive Guide to Programmable Real-Time Graphics*. Boston: Addison-Wesley Longman Publishing Co.

Fitts, P.M., Jones, R.E., & Milton, J.L. (1950). Eye Movements of Aircraft Pilots During Instrument-Landing Approaches. *Aeronautical Engineering Review*, *9*(2), 24–29.

Geldof, P. (2007). *Generic Computing on a Graphics Processing Unit*, M.Sc dissertation, University of Amsterdam, The Netherlands.

Goldberg, J.H., & Kotval, X.P. (1999). Computer interface evaluation using eye movements: methods and constructs. *International Journal of Industrial Ergonomics*, *24*(6), 631–645.

Govindaraju, N. K., Lloyd, B., Dotsenko, Y., Smith, B., & Manferdelli, J. (2008). High performance discrete fourier transforms on graphics processors. In *Proceedings of the 2008 ACM/IEEE conference on Supercomputing* (pp.2:1-12). Austin, Texas, 2008. Piscataway: IEEE Press

Guestrin, E.D., & Eizenman, M. (2006). General theory of remote gaze estimation using the pupil center and corneal reflections, *IEEE Transactions on Biomedical Engineering, 53*(6), 1124–1133.

Hafed, Z.M., & Clark, J.J. (2002). Microsaccades as an overt measure of covert attention shifts. *Vision Research*, *42*(22), 2533–2545.

Hansen, D.W., & Hansen, J.P. (2006). Robustifying eye interaction. In *Proceedings of the 2006 Conference on Computer Vision and Pattern Recognition Workshop (CVPRW'06)* (pp. 152). June 17-22, 2006. IEEE.

Hansen, D.W., Hansen, J.P., Nielsen, M., Johansen, A.S., & Stegmann, M.B. (2002). Eye typing using markov and active appearance models. In *Proceedings Sixth IEEE Workshop on Applications of Computer Vision* (pp. 132–136). IEEE.

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

Hansen, D.W., & Mackay, D. (2004). Eye tracking off the shelf, *Proceedings of the Symposium on Eye Tracking Research and Applications* (pp. 58 – 58). San Antonio, Texas, 2004. New York, NY: ACM.

Hansen, D.W., & Pece, A.E.C. (2005). Eye tracking in the wild. *Computer Vision and Image Understanding, 98*(1), 155–181.

Hansen, J.P., Alapetite, A., MacKenzie, I.S., & Møllenbach, E. (2014). The use of gaze to control drones. *Proceedings of the Symposium on Eye Tracking Research and Applications* (pp. 27–34). Safety Harbor, Florida, 2014. New York, NY: ACM

Harwood, T., & Jones, M. (2014). Mobile eye-tracking in retail research. In M. Horsley, M. Eliot, B.A. Knight, & R. Reilly (Eds.), *Current Trends in Eye Tracking Research*. Cham: Springer International Publishing.

Hayhoe, M., & Ballard, D. (2005). Eye movements in natural behavior, *Trends in Cognitive Sciences, 9*(4).

Hennessey, C., Lawrence, P., & Noureddin, B. (2006). A single camera eye-gaze tracking system with free head motion, *Proceedings of the Symposium on Eye Tracking Research and Applications* ,(pp. 87–94). San Diego, California, March 27 - 29, 2006. New York, NY: ACM.

Hennessey, C., Noureddin, B., & Lawrence, P. (2008). Fixation precision in high-speed noncontact eye-gaze tracking, *IEEE Transaction on Systems, Man and Cybernetics, 38*(2), 289–298.

Hofstee, E. (2013). *Constructing a Good Dissertation: A Practical Guide to Finishing a Masters*. Johannesburg: EPE.

Holmqvist, K., & Nyström, M. (2012). Eye tracker data quality : What it is and how to measure it, *Proceedings of the Symposium on Eye Tracking Research and Applications* (pp. 45–52). Santa Barbara, California, 2012. New York, NY: ACM.

Holmqvist, K., Nyström, M., Andersson, R., Dewhurst, R., Jarodzka, H., & Van de Weijer, J. (2011). *Eye Tracking : A Comprehensive Guide to Methods and Measures* (1st ed). New York: Oxford University Press.

Hornof, A.J., & Halverson, T. (2002). Cleaning up systematic error in eye-tracking data by using required fixation locations. *Behavior Research Methods, Instruments, & Computers : A Journal of the Psychonomic Society, 34*(4), 592–604.

Horsley, M. (2014). Eye tracking as a research method in social and marketing applications. In M. Horsley, M. Eliot, B. A. Knight, & R. Reilly (Eds.), *Current Trends in Eye Tracking Research* (pp. 179–182). Cham: Springer International Publishing.

Hutchinson, T.E., White, K.P., Martin, W.N., Reichert, K.C., & Frey, L.A. (1989). Human-computer interaction using eyegGaze input, *IEEE Transactions on System, Man and Cybernetics, 19*(6), 1527–1534.

IBM. (1984). *IBM Personal Computer Professional Graphics Controller Technical Reference*. IBM Corporation.

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

Jacob, R.J. ., & Karn, K.S. (2003). Eye tracking in human – computer interaction and usability research : Ready to deliver the promises. In Hyönä, J., Radach, R. & Deube, H. (Eds), *The Mind's Eye* (pp. 573-605). Amsterdam: Elsevier.

Jacob, R. J. K. (1995). Eye tracking in advanced interface design. In W. Barfield & T. A. Furness (Eds.), *Virtual Environments and Advanced Interface Design* (pp. 258–288). Oxford University Press.

Javal, E. (1879). Essai sur la physiologie de la lecture. Annales D'Oculistique. 82, 242–253

Joyce, C.A., Gorodnitsky, I.F., King, J.W., & Kutas, M. (2002). Tracking eye fixations with electroocular and electroencephalographic recordings. *Psychopysysiology, 39(5)*, 607–618.

Kim, E.S.,Naples, A., Geary, G.V., Wang, Q., Wallace, S., Wall, C., Kowitt, J., Perlmutter, M., Friedlaender, L., Reichow, B., Volkmar, F., Shic, F. (2014). Development of an untethered, mobile, low-cost head-mounted eye tracker.In *Proceedings of the Symposium on Eye Tracking Research and Applications* (pp. 247–250). Safety Harbor, Florida, 2014. New York, NY: ACM.

Kumar, M. (2006). *Reducing the cost of eye tracking systems*. Last accessed January 31, 2015 from Stanford University HCI Group: http://hci.stanford.edu/research/GUIDe/publications/Stanford%20CSTR%202006-08%20-%20Reducing%20the%20Cost%20of%20Eye-Gaze%20Tracking%20Systems.pdf.

Lemahieu, W., & Wyns, B. (2010). Low cost eye tracking for human-machine interfacing, *Journal of Eyetracking, Visual Cognition and Emotion, 0*(1), 1–12.

Li, D., Babcock, J., & Parkhurst, D.J. (2006). openEyes : a low-cost head-mounted eye-tracking solution, In *Proceedings of the Symposium on Eye Tracking Research and Applications* (pp. 95-100). San Diego, California, March 27 - 29, 2006. New York, NY: ACM.

Li, D., & Parkhurst, D. J. (2005). Starburst : A robust algorithm for video-based eye tracking, *Elsevier Science.* Last accessed 2015/02/01 at http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.88.8248&rep=rep1&type=pdf

Li, D., & Parkhurst, D.J (2006). Open-Source Software for Real-Time Visible- Spectrum Eye Tracking. In *The 2nd Conference on Communication by Gaze Interaction : Gazing into the Future* (pp. 4–6). Turin, Italy, September 4 – 5, 2006.

Li, Y., Qi, X., & Wang, Y. (2001). Eye detection by using fuzzy template matching and feature-parameter-based judgement. *Pattern Recognition Letters*, *22*(10), 1111–1124.

Liang, K., Tijus, C., Chahir, Y., Jouen, F., & Molina, M. (2013). Appearance-based gaze tracking with spectral clustering and semi-supervised gaussian process regression. In *Proceedings of Eye Tracking South Africa* (Vol. 1, pp. 29–31). Cape Town, South Africa, 2013. New York, NY: ACM.

Lv, Z., Wu, X., Li, M., & Zhang, D. (2009). Development of a human computer Interface system using EOG. *Health*, 0(1), 39–46.

Mahmud, Z. (2008). *Handbook of Research Methods : A Simplified Version* (pp. 138). Mara: University of Technology.

Majaranta, P., & Räihä, K.-J. (2002). Twenty years of eye typing : Systems and design issues. *Proceedings of Eye Tracking Research and Applications* (pp 15–22). New Orleans, Louisiana, March 25 -27, 2002. ACM.

Majaranta, P., & Räihä, K.-J. (2007). Text Entry by Gaze : Utilizing eye-tracking. In I. . MacKenzie & K. Tanaka-Ishii (Eds.), *Text Entry Systems: Mobility, Accessibility, Universality* (pp. 175–187). San Francisco : Morgan Kaufmann Publishers.

Manabe, H., & Yagi, T. (2014). EOG-based eye gesture input with audio staging. *Proceedings of the Symposium on Eye Tracking Research and Applications* (pp. 381–382). Safety Harbor, Florida, 2014. New York, NY: ACM

McClanahan, C. (2010). History and evolution of GPU architecture, *A Paper* Survey. Last accessed 31/01/2015 from: http://mcclanahoochie.com/blog/wp-content/uploads/2011/03/gpu-hist-paper.pdf.

Microsoft. (2015a). Compute Shader Overview. Retrieved 27/01/2015 from http://msdn.microsoft.com/en-us/library/windows/desktop/ff476331(v=vs.85).aspx

Microsoft. (2015b). Getting Started with Direct3D. Retrieved 09/01/2015 from http://msdn.microsoft.com/en-us/library/windows/desktop/hh769064(v=vs.85).aspx#Which_D3D_version

Microsoft. (2015c). Shader Model 2. Retrieved 08/01/2015 from http://msdn.microsoft.com/en-us/library/windows/desktop/bb509655(v=vs.85).aspx

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

Morimoto, C., Koons, D., Amir, A., & Flickner, M. (1998). Real-time detection of eyes and faces. In *Proceedings of 1998 Workshop on Perceptual User Interfaces (*pp. 117-120).

Mulligan, J.B. (1997). Image processing for improved eye-tracking accuracy. *Behavior Research Methods, Instruments, & Computers : A Journal of the Psychonomic Society, Inc*, *29*(1), 54–65.

Mulligan, J.B. (2012). A GPU-accelerated software eye tracking system. In *Proceedings of the Symposium on Eye Tracking Research and Applications* (pp. 265–268).Santa Barbara, California, March 28 – 30, 2012. New York, NY : ACM.

Nagamatsu, T., Iwamoto, Y., Kamahara, J., Tanaka, N., & Yamamoto, M. (2010). Gaze estimation method based on an aspherical model of the cornea : Surface of revolution about the optical axis of the eye. In *Proceedings of the Symposium on Eye Tracking Research and Applications* (Vol. 1, pp. 255–258). Austin, Texas: ACM.

Nagamatsu, T., Kamahara, J., & Tanaka, N. (2009). In *CHI'09 Extended Abstracts on Human Factors in Computing Systems* (pp. 3613-3618). ACM.

Noureddin, B., Lawrence, P.D., & Man, C.F. (2005). A non-contact device for tracking gaze in a human computer interface, *Computer Vision and Image Understanding, 98*(1), 52–82.

NVIDIA (2015). *CUDA ZONE.* Retrieved 09/01/2015, from https://developer.nvidia.com/cuda-faq

Nyström, M., Andersson, R., Holmqvist, K., & van de Weijer, J. (2013). The influence of calibration method and eye physiology on eyetracking data quality. *Behavior Research Methods*, *45*(1), 272–88.

Oates, B.J. (2006). *Researching Information Systems and Computing* (pp. 341). London: SAGE Publications.

Ohtera, R., Horiuchi, T., & Tominaga, S. (2009). Iris extraction using parametric template matching for eyegaze tracking. *Optical Engineering*, *48*(4), 047204-047212.

Olivier, M.S. (2009). *Information Technology Research*. (B. Janari, Ed.) (3rd ed.). Pretoria: Van Schaik Publishers.

OpenGL. (2015). *History of OpenGL*. Last accessed 02/02/2015 from https://www.opengl.org/wiki/History_of_OpenGL.

Peeper, C., & Mitchell, J.L. (2004). ShaderX: Introductions & tutorials with DirectX 9. In W. F. Engel (Ed.), *ShaderX2 : Introductions & Tutorials with DirectX 9* (pp. 393). Plano: Wordware Publishing.

Peng, K., Chen, L., & Kukharev, G. (2005). A robust algorithm for eye detection on gray intensity face without spectacles, *Journal of Computer Science & Technology, 5*(3), 127–132.

Poole, A., & Ball, L.J. (2005). Eye tracking in human-computer interaction and usability research : Current status and future prospects. In Ghaoui, C. (Eds.), *Encyclopedia of Human-Computer Interaction*. Pennsylvania: Idea Group.

Rayner, K. (1998). Eye movements in reading and information processing : 20 Years of research, *Psychological bulletin, 124*(3), 372–422.

Reichle, E.D., Pollatsek, A., Fisher, D.L., & Rayner, K. (1998). Toward a model of eye movement control in reading, *Psychological review*, *105*(1), 125-157.

Roodt, Y., Visser, W., & Clarke, W.A. (2007). Image processing on the GPU : Implementing the canny edge detection algorithm. In *Proceedings of the Eighteenth Annual Symposium of the Pattern Recognition Association of South Africa* (pp. 45-50). Pietermaritzburg, November 28 – 30, 2007. University of Kwazulu-Natal Press.

Ryan, W.J., Duchowski, A.T., & Birchfield, S.T. (2004). Limbus / pupil switching for wearable eye tracking under variable lighting conditions, In *Proceedings of the Symposium on Eye Tracking Research and Applications* (pp. 61 – 64). Savannah, Georgia, 2008. New York:ACM.

Salvucci, D. D., & Goldberg, J. H. (2000). Identifying fixations and saccades in eye-tracking protocols, In *Proceedings of the Symposium on Eye Tracking Research and Applications* (pp. 71–78). Palm Beach Gardens, Florida, November 6-8, 2000. ACM.

San Agustin, J., Skovsgaard, H., Mollenbach, E., Barret, M., Tall, M., Hansen, D.W., & Hansen, J.P. (2010). Evaluation of a low-cost open-source gaze tracker, *Proceedings of the Symposium on Eye Tracking Research and Applications* (pp. 77–80). Austin, Texas, 2010. ACM.

Sensoric Motor Instruments (2015). *RED / RED250 / RED500*. Last accessed 2015/01/31 from http://www.smivision.com/en/gaze-and-eye-tracking-systems/products/red-red250-red-500.html.

Sesma-Sanchez, L., Villanueva, A., & Cabeza, R. (2014). Design issues of remote eye tracking systems with large range of movement. *Proceedings of the Symposium on Eye Tracking Research and Applications* (pp. 243–246). Safety Harbor, Florida, 2014. New York, NY: ACM.

SR Research. (2014). *EyeLink 1000*. Retrieved May 19, 2014, from http://www.sr-research.com/EL_1000.html

Sumanaweera, T., & Liu, D. (2005). Medical Image Reconstruction with the FFT. In M. Pharr & R. Fernando (Eds.), *GPU Gems 2* (2nd ed., pp. 765–769). Addison-Wesley.

Tatler, B.W., Kirtley, C., Macdonald, R.G., Mitchell, M.M.A., & Savage, S.W. (2014). The active eye: Perspectives on eye movement research. In M. Horsley, M. Eliot, B.A. Knight, & R. Reilly (Eds.), *Current Trends in Eye Tracking Research* (1st ed., pp. 3–16). Cham: Springer.

Tobii Technology. (2010). *An introduction to eye tracking and Tobii Eye Trackers*. Last accessed 2015/01/31 from http://www.tobii.com/eye-tracking-research/global/library/white-papers/tobii-eye-tracking-white-paper/.

Tobii Technology. (2014). *tobii eyeX*. Last accessed 2015/01/31 from http://www.tobii.com/en/eye-experience/eyex/.

Träisk, F., Bolzani, R., & Ygge, J. (2005). A comparison between the magnetic scleral search coil and infrared reflection methods for saccadic eye movement analysis. *Graefe's Archive for Clinical and Experimental Ophthalmology*, *243*(8), 791–7.

Tullis, T., & Albert, B. (2008). *Measuring the User Experience* (p. 317). Burlington: Morgan Kaufmann.

Villanueva, A., Cabeza, R., & Porta, S. (2007). Gaze tracking system model based on physical parameters. *International Journal of Pattern Recognition and Artificial Intelligence*, *21*(5), 855–877.

Villanueva, A., Cerrolaza, J. J., & Cabeza, R. (2007). Geometry Issues of Gaze Estimation. In *UAHCI'07 Proceedings of the 4th international conference on Universal access in human-computer interaction: ambient interaction*.

Wade, N. J., & Tatler, B. W. (2010). *The Moving Tablet of the Eye : The origins of modern eye research* (pp. 312). New York: Oxford University Press.

Wang, P., Green, M.B., & Ji, Q. (2005). Automatic eye detection and its validation. In *Conference on Computer Vision and Pattern Recognition* (pp. 164). San Diego, CA, USA, June 25, 2005. IEEE.

Wobbrock, J.O., Sawyer, M.W., & Duchowski, A.T. (2008). Longitudinal evaluation of discrete consecutive gaze gestures for text entry, In *Proceedings of the Symposium on Eye Tracking Research and Applications* (pp. 11 – 18). Savannah, Georgia, 2008. New York:ACM.

Yuille, A.L., Hallinan, P.W., & Cohen, D.S. (1992). Feature extraction from faces using deformable templates. *International Journal of Computer Vision*, *8*(2), 99–111.

Zhang, X., & Mackenzie, I.S. (2007). Evaluating eye tracking with ISO 9241. In J. A. Jacko (Ed.), *HCI'07 Proceedings of the 12th international conference on Human-computer interaction: intelligent multimodal interaction environments* (pp. 779–788). Berlin, Heidelberg: Springer International Publishing.

Zhu, Z., & Ji, Q. (2005a). Eye gaze tracking under natural head movements. *In Conference on Computer Vision and Pattern Recognition* (pp. 918-923). June 20-25, 2005. IEEE.

Zhu, Z., & Ji, Q. (2005b). Robust real-time eye detection and tracking under variable lighting conditions and various face orientations, *Computer Vision and Image Understanding, 98(1)*, 124–154.

APPENDIX A – HLSL TRACKER TECHNICAL MANUAL

The HLSL tracker is an eye tracking mechanism designed to provide the locations of feature points within an eye video. The purpose of this appendix is to provide an overview of the HLSL eye tracker and API. Included in this document are the system requirements of the eye tracker software and description of the various tools used during the data capturing and analysis of the experiment described in Chapter 4. Developers wishing to utilise the HLSL API will also find instructions on how to create a basic eye tracking application, with explanations of the various components that are available.

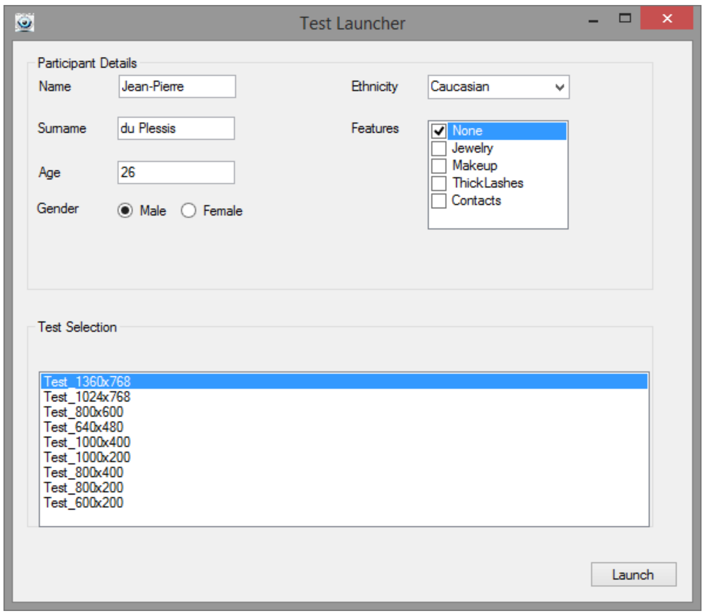
1.1 Hardware and software requirements

The performance of the HLSL tracker will depend on the hardware present in the host machine. To run the HLSL tracker, there are a number of software requirements. The host machine should run Windows XP or a later version with the latest version of Direct X 9, as well as the Dot.Net 3.5 framework (or higher). Due to the use of DirectX 9, the host machine will require a compatible display adapter. Additional hardware requirements may apply depending on the choice of IDS camera used.

The HLSL tracker was developed to work specifically alongside IDS (https://en.ids-imaging.com/) cameras utilising the uEye drivers and API (https://en.ids-imaging.com/download-ueye.html). Thus, at the time of writing the dissertation the use of other camera types is not supported.

1.2 Data capturing application

The HLSL tracker used in the experiment described in Chapter 4 consisted of two applications. The first application was a launcher program (Figure A.1) responsible for capturing the details of the participant, and setting up the test. The test consisted of the participant's details, as well as the eye tracker parameters required for the test (sampling rate and eye video size). The launcher application also launched the eye tracker application that would display the gaze dots and record the data for the selected test.


Test Launcher

Participant Details

Name: Jean-Pierre

Surname: du Plessis

Age: 26

Gender: [✓ Male] Female

Ethnicity: Caucasian

Features: [✓ None] Jewelry Makeup ThickLashes Contacts

Test Selection

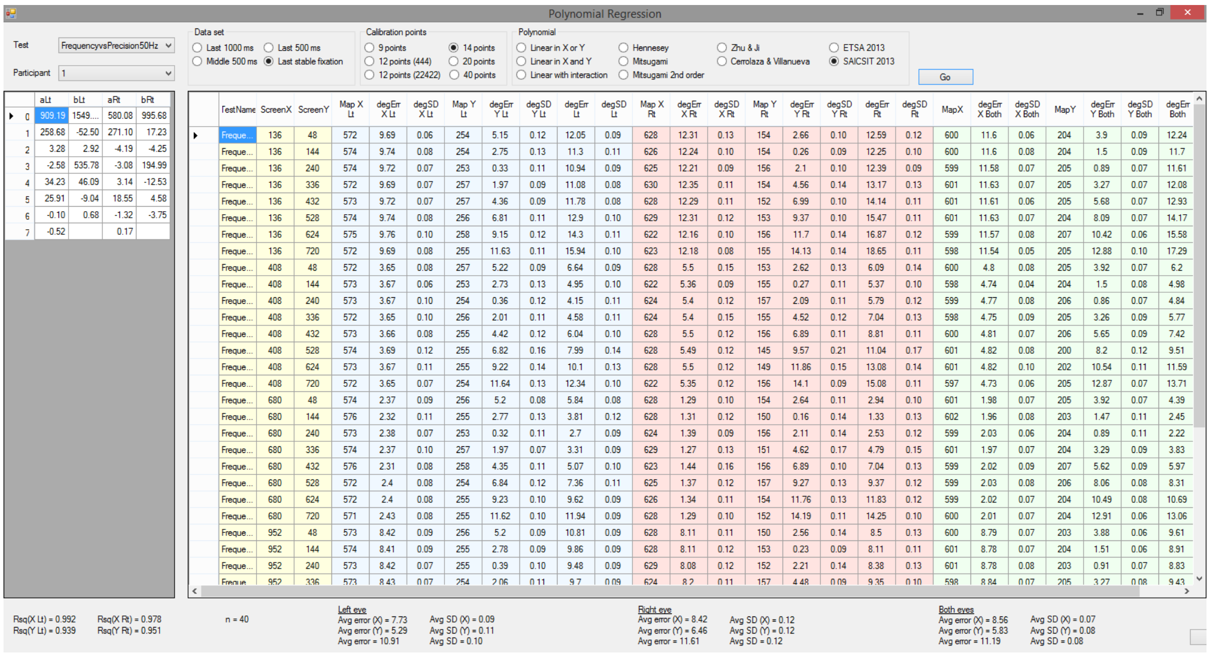
[✓ Test_1360x768]
Test_1024x768
Test_800x600
Test_640x480
Test_1000x400
Test_1000x200
Test_800x400
Test_800x200
Test_600x200

Figure A.1 – The HLSL tracker launcher application. This application records participant details and launches the eye tracker with the required parameters based on the test selected.

The second application was responsible for displaying the forty dots and saving all the data recorded during the session to a CSV file. The output file contained the participant details along with each individual sample. The samples utilised the data structure shown in Figure A.7 in Section 1.4.3 in addition to storing the x and y coordinates of the currently displayed dot.

1.3 Data processing applications

The data from the experiment was collated into a MySQL database using an application developed for this purpose. The application merged the data from each individual test for each participant into a single table in the database. A second application filtered the data and extracted the 250 ms of data used to calculate the accuracy and precision. A third and final application (Figure A.2) was used to compute the accuracy and precision, and store the results in the database.



Polynomial Regression

Test: FrequencyvsPrecision50Hz
Participant: 1

Data set: ( ) Last 1000 ms ( ) Last 500 ms ( ) Middle 500 ms [✓ Last stable fixation]

Calibration points: ( ) 9 points [✓ 14 points] ( ) 12 points (444) ( ) 20 points ( ) 12 points (22422) ( ) 40 points

Polynomial: ( ) Linear in X or Y ( ) Hennesey ( ) Zhu & Ji ( ) ETSA 2013 ( ) Linear in X and Y ( ) Mitsugami ( ) Cerrolaza & Villanueva [✓ SAICSIT 2013] ( ) Linear with interaction ( ) Mitsugami 2nd order

Go

	aLt	bLt	aRt	bRt
0	909.19	1549....	580.08	995.68
1	258.68	-52.50	271.10	17.23
2	3.28	2.92	-4.19	-4.25
3	-2.58	535.78	-3.08	194.99
4	34.23	46.09	3.14	-12.53
5	25.91	-9.04	18.55	4.58
6	-0.10	0.68	-1.32	-3.75
7	-0.52		0.17	

TestName	ScreenX	ScreenY	Map X Lt	degErr X Lt	degSD X Lt	Map Y Lt	degErr Y Lt	degSD Y Lt	degErr Lt	degSD Lt	Map X Rt	degErr X Rt	degSD X Rt	Map Y Rt	degErr Y Rt	degSD Y Rt	degErr Rt	degSD Rt	MapX	degErr X Both	degSD X Both	MapY	degErr Y Both	degSD Y Both	degErr Both
Freque...	136	48	572	9.69	0.06	254	5.15	0.12	12.05	0.09	628	12.31	0.13	154	2.66	0.10	12.59	0.12	600	11.6	0.06	204	3.9	0.09	12.24
Freque...	136	144	574	9.74	0.08	254	2.75	0.13	11.3	0.11	626	12.24	0.10	154	0.26	0.09	12.25	0.10	600	11.6	0.08	204	1.5	0.09	11.7
Freque...	136	240	574	9.72	0.07	253	0.33	0.11	10.94	0.09	625	12.21	0.09	156	2.1	0.10	12.39	0.09	599	11.58	0.07	205	0.89	0.07	11.61
Freque...	136	336	572	9.69	0.07	257	1.97	0.09	11.08	0.08	630	12.35	0.11	154	4.56	0.14	13.17	0.13	601	11.63	0.07	205	3.27	0.07	12.08
Freque...	136	432	573	9.72	0.07	257	4.36	0.09	11.78	0.08	628	12.29	0.11	152	6.99	0.10	14.14	0.11	601	11.61	0.06	205	5.68	0.07	12.93
Freque...	136	528	574	9.74	0.08	256	6.81	0.11	12.9	0.10	629	12.31	0.12	153	9.37	0.10	15.47	0.11	601	11.63	0.07	204	8.09	0.07	14.17
Freque...	136	624	575	9.76	0.10	258	9.15	0.12	14.3	0.11	622	12.16	0.10	156	11.7	0.14	16.87	0.12	599	11.57	0.08	207	10.42	0.06	15.58
Freque...	136	720	572	9.69	0.08	255	11.63	0.11	15.94	0.10	623	12.18	0.08	155	14.13	0.14	18.65	0.11	598	11.54	0.05	205	12.88	0.10	17.29
Freque...	408	48	572	3.65	0.08	257	5.22	0.09	6.64	0.09	628	5.5	0.15	153	2.62	0.13	6.09	0.14	600	4.8	0.08	205	3.92	0.07	6.2
Freque...	408	144	573	3.67	0.06	253	2.73	0.13	4.95	0.10	622	5.36	0.09	155	0.27	0.11	5.37	0.10	598	4.74	0.04	204	1.5	0.08	4.98
Freque...	408	240	573	3.67	0.10	254	0.36	0.12	4.15	0.11	624	5.4	0.12	157	2.09	0.11	5.79	0.12	599	4.77	0.08	206	0.86	0.07	4.84
Freque...	408	336	572	3.65	0.10	256	2.01	0.11	4.58	0.11	624	5.4	0.15	155	4.52	0.12	7.04	0.13	598	4.75	0.09	205	3.26	0.09	5.77
Freque...	408	432	573	3.66	0.08	255	4.42	0.12	6.04	0.10	628	5.5	0.12	156	6.89	0.11	8.81	0.11	600	4.81	0.07	206	5.65	0.09	7.42
Freque...	408	528	574	3.69	0.12	255	6.82	0.16	7.99	0.14	628	5.49	0.12	145	9.57	0.21	11.04	0.17	601	4.82	0.08	200	8.2	0.12	9.51
Freque...	408	624	573	3.67	0.11	255	9.22	0.14	10.1	0.13	628	5.5	0.12	149	11.86	0.15	13.08	0.14	601	4.82	0.10	202	10.54	0.11	11.59
Freque...	408	720	572	3.65	0.07	254	11.64	0.13	12.34	0.10	622	5.35	0.12	156	14.1	0.09	15.08	0.11	597	4.73	0.06	205	12.87	0.07	13.71
Freque...	680	48	574	2.37	0.09	256	5.2	0.08	5.84	0.08	628	1.29	0.10	154	2.64	0.11	2.94	0.10	601	1.98	0.07	205	3.92	0.07	4.39
Freque...	680	144	576	2.32	0.11	255	2.77	0.13	3.81	0.12	628	1.31	0.12	150	0.16	0.14	1.33	0.13	602	1.96	0.08	203	1.47	0.11	2.45
Freque...	680	240	573	2.38	0.07	253	0.32	0.11	2.7	0.09	624	1.39	0.09	156	2.11	0.14	2.53	0.12	599	2.03	0.06	204	0.89	0.11	2.22
Freque...	680	336	574	2.37	0.10	257	1.97	0.07	3.31	0.09	629	1.27	0.13	151	4.62	0.17	4.79	0.15	601	1.97	0.07	204	3.29	0.09	3.83
Freque...	680	432	576	2.31	0.08	258	4.35	0.11	5.07	0.10	623	1.44	0.16	156	6.89	0.10	7.04	0.13	599	2.02	0.09	207	5.62	0.09	5.97
Freque...	680	528	572	2.4	0.08	254	6.84	0.12	7.36	0.11	625	1.37	0.12	157	9.27	0.13	9.37	0.12	599	2.03	0.08	206	8.06	0.08	8.31
Freque...	680	624	572	2.4	0.08	255	9.23	0.10	9.62	0.09	626	1.34	0.11	154	11.76	0.13	11.83	0.12	599	2.02	0.07	204	10.49	0.08	10.69
Freque...	680	720	571	2.43	0.08	255	11.62	0.10	11.94	0.09	628	1.29	0.10	152	14.19	0.11	14.25	0.10	600	2.01	0.07	204	12.91	0.06	13.06
Freque...	952	48	573	8.42	0.09	256	5.2	0.09	10.81	0.09	628	8.11	0.11	150	2.56	0.14	8.5	0.13	600	8.79	0.07	203	3.88	0.06	9.61
Freque...	952	144	574	8.41	0.09	255	2.78	0.09	9.86	0.09	628	8.11	0.12	153	0.23	0.09	8.11	0.11	601	8.78	0.07	204	1.51	0.06	8.91
Freque...	952	240	573	8.42	0.07	255	0.39	0.10	9.48	0.09	629	8.08	0.12	152	2.21	0.14	8.38	0.13	601	8.78	0.08	203	0.91	0.07	8.83
Freque	952	336	573	8.43	0.07	254	2.06	0.11	9.7	0.09	624	8.2	0.11	157	4.48	0.09	9.35	0.10	598	8.84	0.07	205	3.27	0.08	9.43

Rsq(X Lt) = 0.992 Rsq(X Rt) = 0.978
Rsq(Y Lt) = 0.939 Rsq(Y Rt) = 0.951

n = 40

Left eye
Avg error (X) = 7.73 Avg SD (X) = 0.09
Avg error (Y) = 5.29 Avg SD (Y) = 0.11
Avg error = 10.91 Avg SD = 0.10

Right eye
Avg error (X) = 8.42 Avg SD (X) = 0.12
Avg error (Y) = 6.46 Avg SD (Y) = 0.12
Avg error = 11.61 Avg SD = 0.12

Both eyes
Avg error (X) = 8.56 Avg SD (X) = 0.07
Avg error (Y) = 5.83 Avg SD (Y) = 0.08
Avg error = 11.19 Avg SD = 0.08

Figure A.2 - Precision and accuracy application showing the results of the precision obtained using artificial eyes

This application allowed a number of different mapping polynomials and calibration point selections to be experimented with for each participant, before finally selecting a combination to use for the final results. The data generated by this mapping application was then analysed using Statistica 12.

1.4 HLSL tracker API

The HLSL tracker software also contains an API that can be utilised to develop custom eye tracking applications. The API contains three core components (Figure A.3) that are of interest to the developer. Each of these components is explained in further depth in the following sections. The HLSL tracker components work together to process eye images and locate the pupils and corneal reflections. These coordinates can be accessed by developers (see 1.4.3) at any time while the application is running. Note that no gaze estimation is performed, and that only the raw data in the form of feature point locations and time stamps of individual frames (Figure A.7) is reported by the system.

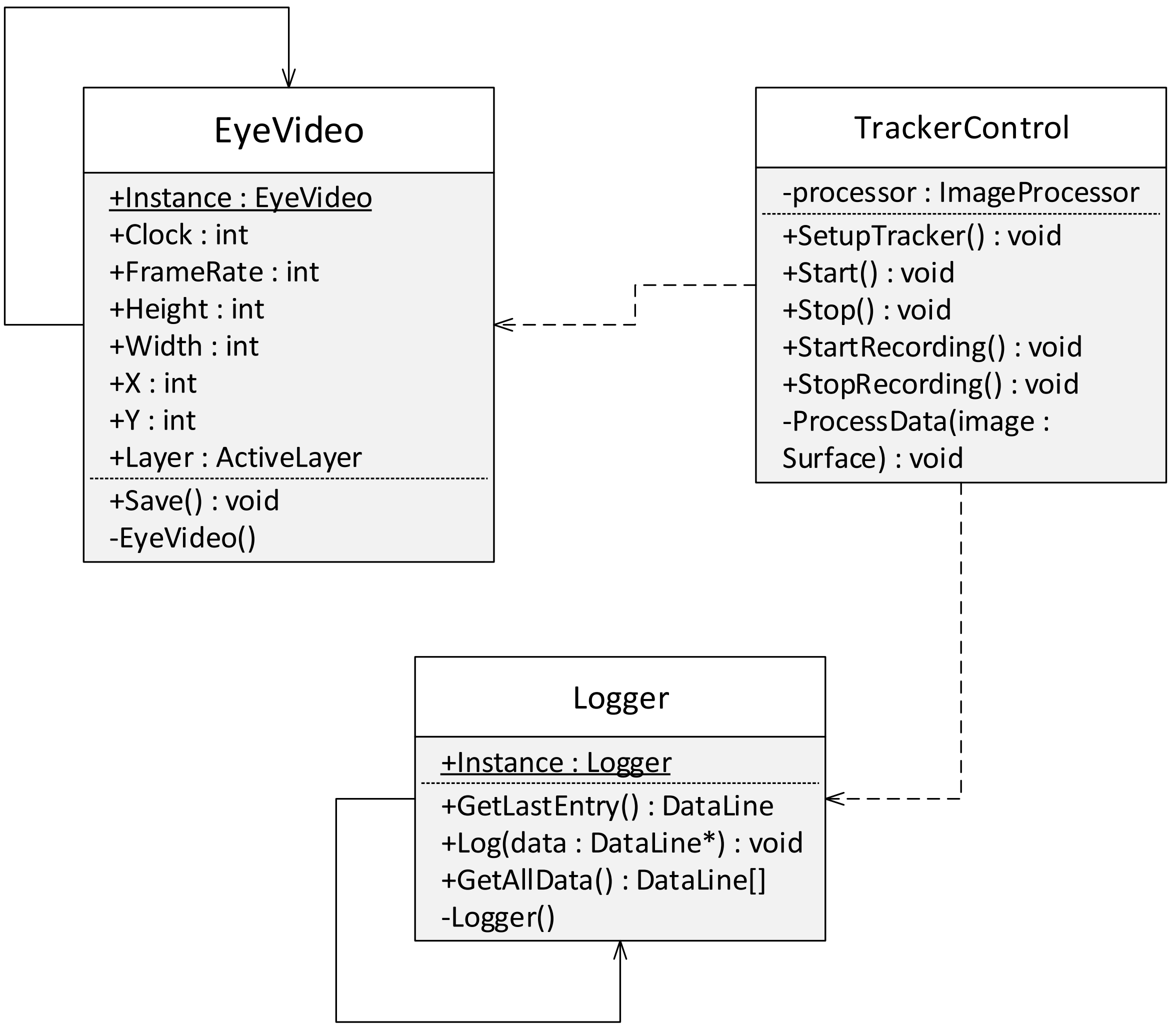


Figure A.3 - The UML of the HLSL tracker core

1.4.1 Configuring the eye tracker

Both the initial size of the eye video and sampling rate of the eye tracker can be set by the programmer using the **EyeVideo** instance. Note that adjustments made to the size of the eye video will require the eye tracker to be reinitialised. However, the frame rate of the eye tracker can be modified at runtime (Section 1.6). The eye video size and position can be set by modifying the **Width**, **Height**, **X** and **Y** properties in the **EyeVideo** instance (see Figure A.4), whereas the sampling rate is set by changing the **Clock** and **FrameRate** properties.

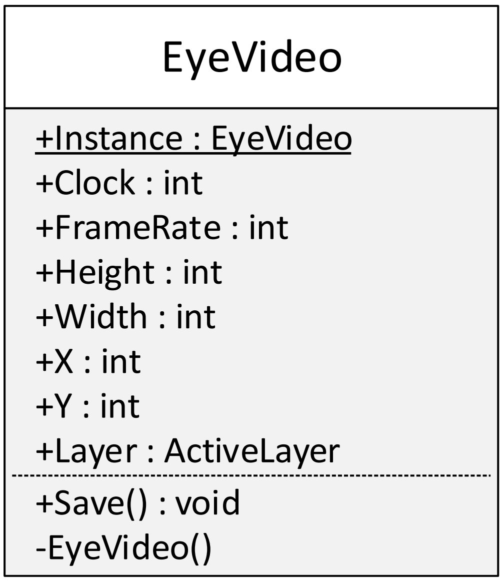


Figure A.4 - The EyeVideo class where the initial tracker settings can be modified

1.4.2 The TrackerControl component

The **TrackerControl** (Figure A.5) control forms the heart of the HLSL tracker. It is responsible for communicating with the IDS camera, processing the eye images and displaying the results of the processing on the eye video. The **TrackerControl** component has five methods that are of use to the programmer. The **SetupTracker** method will initialise the tracking software based on the values that are present in the **EyeVideo** singleton. **Start** and **Stop** will start and stop the tracking process. Finally, the **StartRecording** and **StopRecording** will toggle data capturing on or off.

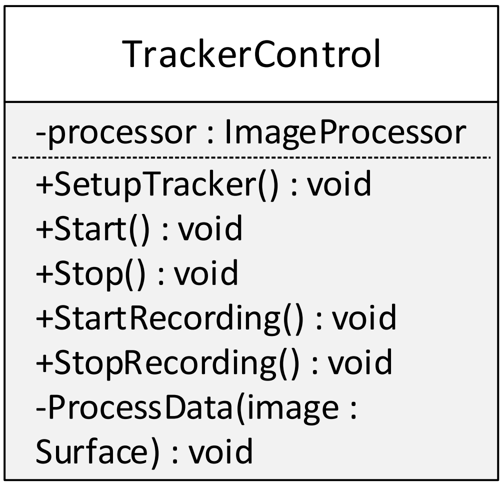


Figure A.5 - The TrackerControl component

1.4.3 Accessing the eye tracker data

All data captured by the **TrackerControl** is automatically logged by the **Logger** (Figure A.6) instance, provided that data capturing has been turned on. The data can be accessed by using the **GetLastEntry** method, which will return the last set of data captured. Programmers can also access all data captured in a single session by using the **GetAllData** method.

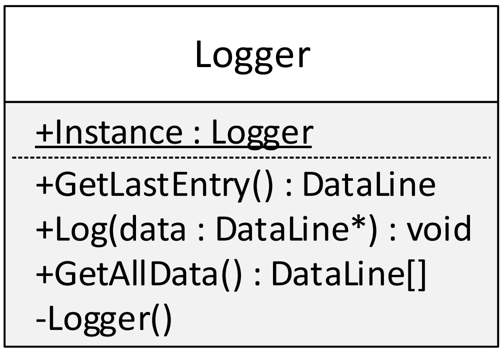

Logger
+Instance : Logger
+GetLastEntry() : DataLine
+Log(data : DataLine*) : void
+GetAllData() : DataLine[]
-Logger()

Figure A.6 – The Logger singleton that can be used to access the feature point data sampled by the tracker on a frame-by-frame basis

The data from the **Logger** is stored in the **DataLine** structure (Figure A.7) in system memory to minimise the impact on performance that writing to the hard drive directly may have. Developers can implement their own storage methods to store the data located in memory.

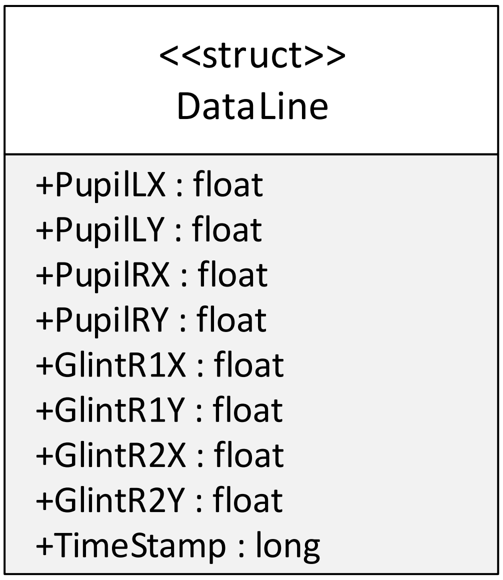


Figure A.7 – The structure containing the raw data logged by the eye tracker for each frame

1.5 Sample application

This section will provide a step-by-step walkthrough in creating a simple application using the HLSL tracking API in Visual Studio 2012, utilising the Dot.Net framework and C#

programming language. This demonstration assumes that the reader is already familiar with programming in C#.

1.5.1 Creating the form

The HLSL sample application utilises a form resembling the following image (Figure A.8).

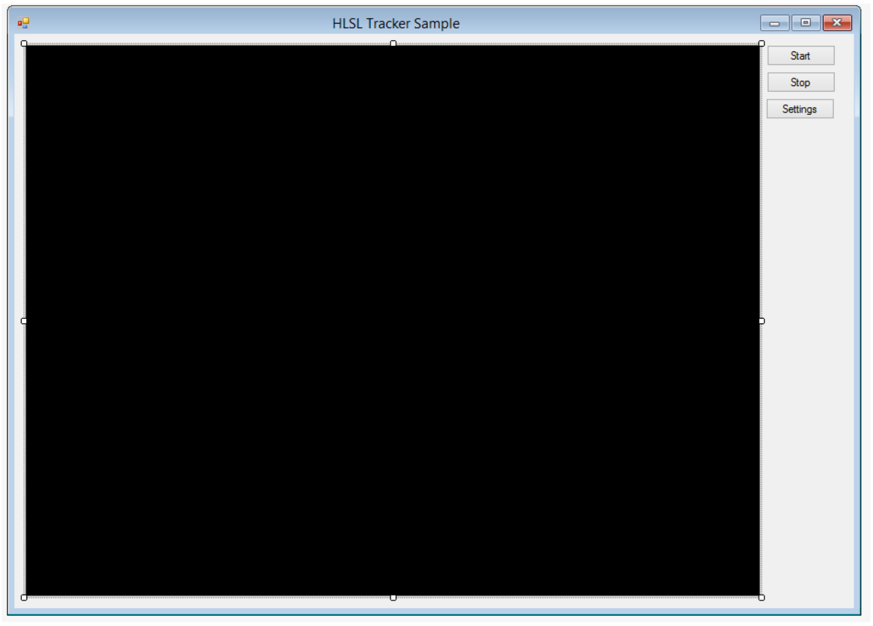


Figure A.8 – Sample form used by the HLSL sample application

The form contains the **TrackerControl** component (in black), and three buttons to start, stop and adjust camera settings.

1.5.2 Tracker parameters and initialisation

The initial eye image size and sampling rate of the tracker are set using the `EyeVideo` class. Once the initial eye tracker parameters are set, the tracker is initialised. The following code snippet illustrates how this is done:

```
protected override void OnLoad(EventArgs e)
{

            EyeVideo.Instance.Width = 800;
            EyeVideo.Instance.Height = 600;
            EyeVideo.Instance.Clock = 40;
            EyeVideo.Instance.FrameRate = 30;

            tracker.SetupTracker();
            base.OnLoad€;
}
```

Note that the **TrackerControl** will resize to these dimensions once the program is run. This should be taken into account when laying out other controls on the form. In practice it is recommended that the size of the **TrackerControl** be set to the required eye video size in design time to avoid potential layout issues.

1.5.3 Starting and stopping the eye tracker

Once the eye tracker has been initialised, the tracking process can be started. During this process the feature points will be located in the eye video, and reported using the **Logger** and the output eye video. The start and stop commands are placed in the click events of the **Start** and **Stop** button on the form. To enable or disable data capturing, the **StartRecording** or **StopRecording** method is called. The following code snippet illustrates how this is done:

```
private void startTracker(object sender, EventArgs e)
{
     tracker.Start();
     tracker.StartRecording();
}

private void stopTracker(object sender, EventArgs e)
{
     tracker.StopRecording();
     tracker.Stop();
}
```

1.5.4 Retrieving data

The captured data of an entire session can be retrieved using the following snippet:

```
DataLine[] data = Logger.Instance.GetAllData();
```

The can be done at any time, though at high frequencies it may be advisable to request the data only at the end of a session to avoid potential spikes in the sampling rate. To request the data of the last frame captured, use the **GetLastEntry** method instead.

1.5.5 Notifications of captured frames

The **TrackerControl** also contains an event called **FrameCaptured** that can be used to perform actions each time a new frame is processed by the image processor. The code below will request the data extracted from the last frame, and insert its timestamp into the title of the form.

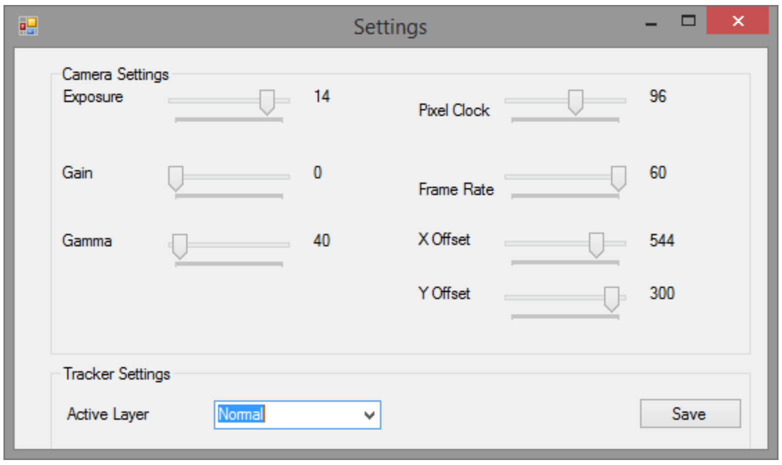
```
void tracker_FrameCaptured(object sender, EventArgs e)
{
     this.Text = Logger.Instance.GetLastEntry().TimeStamp.ToString();
}
```

1.6 Adjusting the camera

Depending on the participant and lighting conditions, the camera may require adjustments to ensure that the pupils are tracked correctly. The HLSL tracker API has a **CameraSetting** dialog (Figure A.9) that can be used to make these adjustments, provided a supported IDS camera is attached. The dialog can be opened with the following code snippet placed in the click event of the **Settings** button:

```
CameraSettings settings = new CameraSettings();
settings.Show();
```

Using this dialog, the sampling rate of the eye tracker can also be adjusted. However, the sampling rate achievable will also depend on the specifications of the system on which the software is running. The **Logger** can be used to determine the actual sampling frequency of the eye tracker by examining the timestamps.


Settings

Camera Settings
Exposure 14
Pixel Clock 96
Gain 0
Frame Rate 60
Gamma 40
X Offset 544
Y Offset 300

Tracker Settings
Active Layer: Normal

Figure A.9 – The CameraSettings dialog used to adjust the camera’s parameters

The frame rate is modified by making changes to the pixel clock and frame rate of the camera. Note that changes to the pixel clock may affect the maximum exposure time of the camera. The X and Y offset can be used to move the eye video left or right. The amount of movement that is permitted is dependent on the camera model and eye video resolution.

1.6.1 Fine-tuning feature points

As mentioned earlier, some adjustment to the camera is required to ensure that feature points are correctly and accurately tracked. This is done by modifying the exposure, gain and gamma of the camera. Figure A.10 below illustrates a correctly calibrated camera.

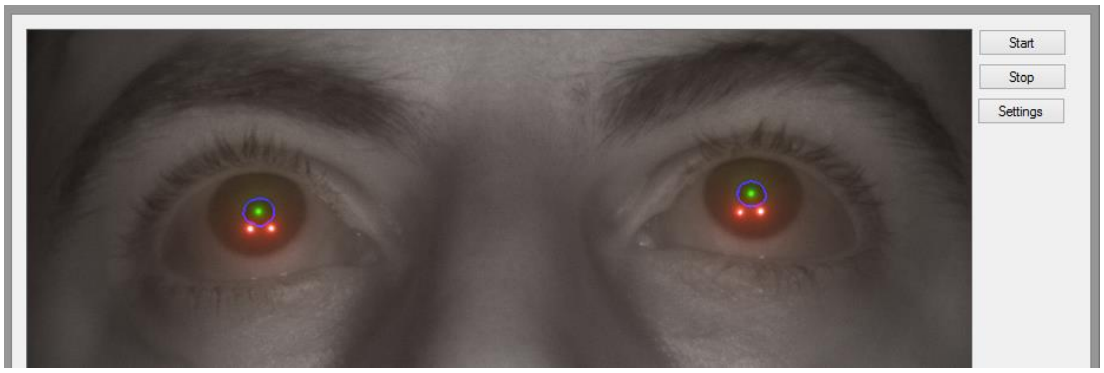


Figure A.10 - Correctly calibrated exposure, gain and gamma

It is sometimes simpler to adjust the camera by looking at the image as the image pre- or post-processor sees it (Figure A.11). These two layers can be toggled by switching the active layer to the “pre-” or “post-” layer.

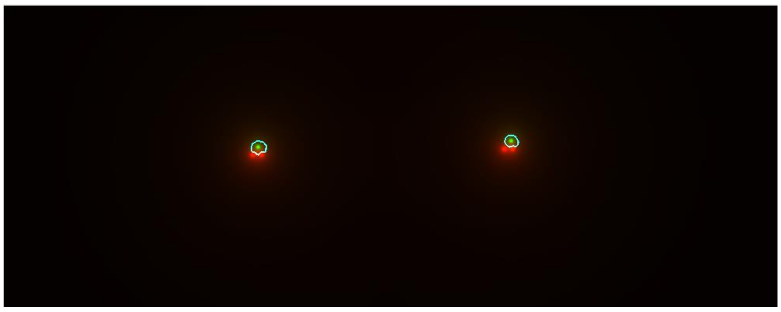


Figure A.11 - The post-processes layer, with correctly calibrated exposure, gain and gamma

The following series of images illustrates typical eye videos with incorrect camera settings.

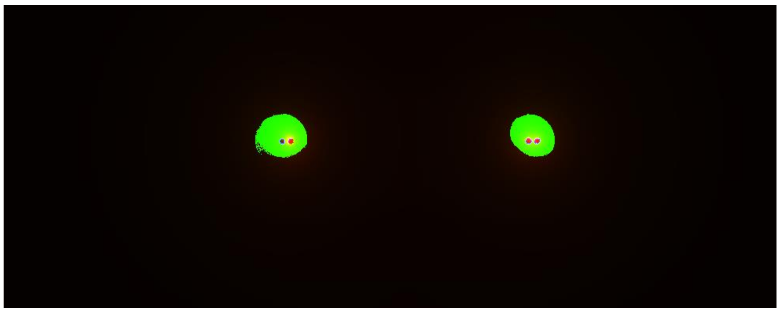


Figure A.12 - Pre-process image illustrating an underexposed image

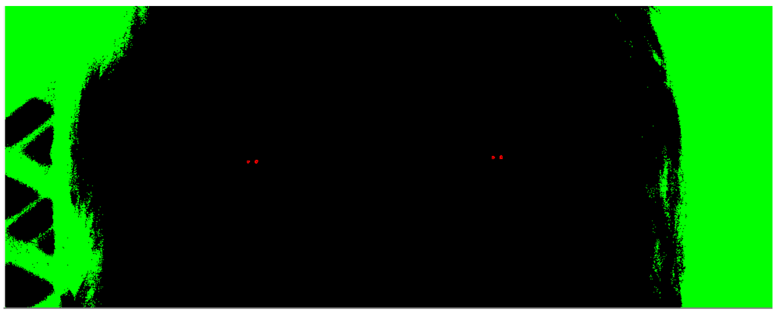


Figure A.13 - Pre-processed eye image illustrating an overexposed image

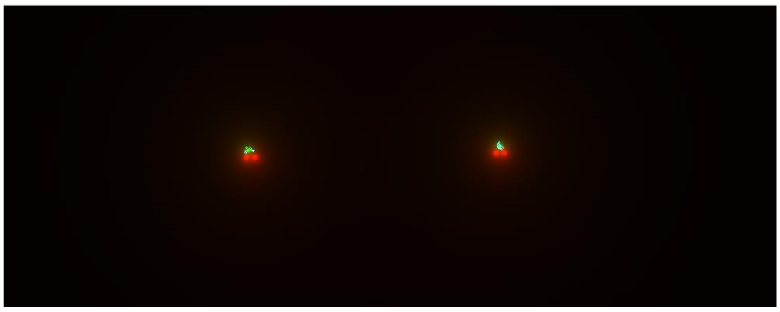


Figure A.14 - Post-process image showing the result of a noisy pupil that requires adjustments to the gamma

1.7 Conclusion

This appendix provided an overview of the various applications utilised alongside the HLSL tracker API to gather data and evaluate its performance. The requirements in terms of hardware and software were provided. Moreover, sample code demonstrating a simple eye tracking application was shown, along with an explanation of the components available to developers in the API. Finally, illustrations depicting the proper adjustment of the camera's exposure time, gain and gamma were displayed.