USING A VR-BASED HEALTH MONITORING SYSTEM TO DETECT CONCUSSION

Victoria Louise Charvis 1833584

Supervised by Dr Sang-Hoon Yeo and Dr Iain Styles.

Dissertation submitted to the University of Birmingham in partial fulfilment for the degree MSc in Advanced Computer Science.

School of Computer Science, University of Birmingham

September 2018

Table of Contents

List of Figures

List of Tables

Abstract

A sports concussion can be defined as a mild traumatic brain injury (mTBI) that occurs due to a sporting accident and alters brain function (1 - 4). As there are a range of symptoms that a concussed individual may exhibit, there are also a varying range of metrics and systems that can be used to detect concussions (1 – 3). While some systems and tests may be effective as standalone tools, others are only reliably accurate when used in conjunction with additional systems (1). In a sport setting, this can be a serious problem as deciding if an athlete can safely be returned to play is a time sensitive situation, and it may not be feasible to use the required equipment on site, for example an MRI machine (3,4). Due to this limitation, there is a strong need for suitable sideline concussion detection tools. The implemented solution discussed in this paper makes use of the strong evidence for eye movements being a good indication of brain functionality (1,2,3,5,6). Through analysing user eye movements made during tasks (in particular saccades – "rapid changes in the orientation of the eye" (1) to a stimulus, and anti-saccades – saccades which mirror the direction of the stimulus) an insight can be gained into if an individual's brain is functioning as normal (2, 3). The system uses the commercially available FOVE virtual reality (VR) headset to record user eye movements in rounds of saccade and antisaccade tasks; this data is then analysed to automatically generate a report detailing initial saccade latency, speed, duration and latency of corrective saccades. This solution was able to accurately and reliably record eye movements and detect saccades, and with little user input produce informative performance data; providing a beneficial contribution to portable concussion detection. The comparison of the developed system to The ImPACT Test found that antisaccade latency was the most insightful metric and that there was a correlation between the reaction time results obtained by both systems.

Acknowledgements

I would like to thank my supervisors Dr Sang-Hoon Yeo and Dr Iain Styles for their guidance and enthusiasm throughout the project.

I would also like to thank my family and friends for their continued support, as well as members of The University of Birmingham for testing and providing feedback on the developed system.

1 Introduction

The aim of this study was to produce a portable and reliable concussion detection system appropriate for use in a sports setting. Its main objective was to accurately track user's eye movements and output a report on user's cognitive performance. This quantitative evaluation can be used to help coaches and physicians make an informed decision on if it is suitable for an athlete to return to play.

Annually, the number of head injury cases that A&E departments in the UK are estimated to receive is around 900,000 (7). As this total was obtained from emergency room data and not all concussed individuals seek medical attention or are treated elsewhere, the actual number of concussions is likely larger (1,3,7). A common cause for players not seeking medical attention is that they may not want to be removed from the current match or sporting activity (3). It is also reported in 'Traumatic brain injury and offending' that "Over a million people in this country live with the consequences of traumatic brain injury" and that this costs the economy around £15 billion each year (7). Concussions are not just a problem relating to athletes as up to 1.3 million people in the UK are estimated to be living with a TBI-related disability, and it is reportedly "the leading cause of death and disability in people aged 1-40 in the UK" (7).

Concussions are diagnosed in awake patients and visible indicators include decreased attention and balance, which can increase the risk of obtaining further injuries (1). Once concussed, players are three times more likely to get another concussion in the same season, with less force being needed to cause this (1,2,3). These impacts can change a range of neurocognitive functions including attention, memory and learning, overall cognitive function, and executive function, and can last for hours or weeks after the injury (1, 4). Concussions can result in slower recovery from the incident with some individuals being unable to recover at all, depending on the individual and the nature of the incident the subsequent after-effects may last for months or years (4). Monitoring these symptoms can help with both diagnosis and recovery and also reduce the financial burden of both the direct and indirect costs. Concussions can cause a serious health issues as even mild repeated sub-concussive impacts can lead to changes in the brains structure and functional performance, however these incidents may not be recorded (3). They can prove fatal in some cases even if classified as mild TBIs on the Glasgow Coma Scale, which is used to classify severity (1). Subsequent concussions can lead to the condition 'second-impact syndrome', and while this is rare it can cause brain swelling resulting in disability or death (1,3). Due to this it is important that concussions are quickly detected in order to prevent further injury.

Considering the points above there is clearly a need for reliable and accurate concussion detection in order to increase diagnosis and provide more help, in particular a service that is administrable during sporting activities is needed. The main focal point of many detection tests is to examine brain function and detect symptoms relating to mTBIs (4). For this reason, MRI tests have successfully been used in a variety of cases and have also demonstrated that there is an overlap between gaze and attention, with it being reported that up to 90% of concussed patients exhibit visual abnormalities (1). Ventura et al (2016) state that MRI changes in concussed patients have been found to correlate with disturbances in eye movements (3). MRI usage is not always a practical solution for sports concussions as it cannot be quickly and easily administered during a game and requires specialist training (4). An alternative technology, eye-tracking, has been used in assessing injured patients and has been found to be a reliable and practical solution for concussion detection (1,2,3,5,6).

The analysis of anti-saccades has found a way to reliably tell the difference between individuals with Alzheimer's disease, frontal lobe lesions, and schizotypal symptoms (2). More relevantly, Webb (2017) found that individuals with a sport-related concussion or mTBI/TBI produced more errors and longer anti-saccade reaction times seven years after their injury when completing tasks in comparison to their age-matched control group (2). Similarly, in the acute recovery stage anti-saccade performance was found to be affected by concussions, however saccade performance was not (2). This supports the notion that eye movements are a sensitive and valuable indicator of brain functionality. Sussman et al (2016) reported that post-concussive patients that still experienced symptoms some time after the incident were also observed to have abnormal eye movements (1). To paraphrase the aforementioned paper, eye movement abnormalities have been found to correlate with the severity of the concussive injury (1). Further support for the association between eye movements and individuals having mTBI can be found in the paper 'Ocular motor assessment in concussion', where a study reported that 30 days after the incident ocular motor function impairments were still present during complex tasks in concussed patients when compared to the control group (3).

VR can offer a controlled immersive environment that can capture the user's body movement, as well as provide better ecological validity and sensitivity than other methods (4). It is portable and can improve concussion detection through capturing participant's natural behaviour and can detecting subtle signs, such as cognitive changes, which may be missed by other assessment means (4). Studies using VR have been performed with both adults and children and in some cases included testing other factors such as balance and social problem-solving (4). These were found to give promising results and could identify residual symptoms in participants (4). The study of adolescents in 'Virtual reality as a screening tool for sports concussion in adolescents' found that concussed patients made more commission errors (errors in performing the given task) than the control group, and while the control group's commission errors rate decreased over time the concussed groups did not (4). This provides strong evidence that VR technology can be used to detect even subtle concussion effects in patients (4).

The metrics reported by the system are latency of initial saccade, duration of initial saccade, speed of initial saccade and latency of corrective saccade. These were influenced and deemed appropriate through the evaluation of many related papers. The paper 'Rapid number naming in chronic concussion' measured the performance of participants doing The King-Devick Test; this, times the user's ability to quickly read a sheet of numbers aloud (8). While this stated that there were no differences in saccade duration or amplitude, it did report a difference between saccade endpoint distances from the centres of the numbers (8). This suggests that concussed participants struggled to accurately judge where and how much to move their eyes during the task (8). Drawing inspiration from this, by plotting duration and speed (°/s) the developed system could display and report how well the participant was able to consistently aim. Further support for this was given by Ventura et al (2016), this paper also reported a range of saccade measurements affected by concussion, including larger position errors and longer latency (3). Conversely to Rizzo et al (2016) (8) this reported smaller saccade amplitudes (3), therefore by displaying saccade speed and duration the developed system could help to indicate if this is the case. This study also stated that abnormal eye movements were also evidenced in anti-saccades which had higher latencies, more directional errors and poor spatial accuracy (3). These will be able to be expressed through the developed systems metrics stated above. Longer time between saccades was also reported with deficiencies still found after 3-5 months in participants with residual symptoms (3). A similar insight can be seen in the developed system through plotting the latency of corrective saccades. This is the time between the erroneous first saccade and the corrective saccade in the right direction. The developed system also reports how many errors were recorded and the participants error rate, which is an obvious and justifiable choice of metric as it allows the assessor to quickly quantify performance. The fundamental criteria for this product was for the overall process to be quick, portable, accurate, cost-effective and require minimal staff training; and a computational solution was deemed suitable for this.

The following chapters will review existing concussion detection tools and the inherited advantages obtained from them, the development of the system specification and system design, as well as the implementation and testing of the final product. The user interface and project management will be reviewed before presenting and evaluating the final project results.

2 Review of Existing Systems

A range of paper-based screening tests are currently used in order to detect and diagnose concussion (1 - 3). The simplest is a symptom checklist in which the patient is asked if they are experiencing a range of concussion-related symptoms such as dizziness, headaches or visual disturbances (1). The symptoms on the checklist aren't exclusive to concussions and the reporting of these varies per individual, therefore concussions may not always be identified using just this method (1).
The Standardized Assessment of Concussion (SAC) is a commonly used alternative method in which the patient's orientation, concentration and memory is assessed via a cognitive test (1,3). Similar to the symptom checklist this is more easily administered in an everyday setting such as sideline screening, however again it isn't sensitive enough to be used alone (1).

Balance and gait can be used to assess the impact of a concussion and numerous sideline tests are available to evaluate this (1,2). These tests often consist of a small number of physical tasks such as The Balance Error Scoring System (BESS) in which an individual performs various stances on different surfaces, The Timed Tandem Gait Test (TGT) in which an individual walks heel-to-toe along a 3m line of tape, or using a Nintendo Wii balance board for more objective measurements (1,3). These are all administrable in a variety of settings and don't require highly specialised staff or equipment, however both can be affected by fatigue, injuries, and the individual's fitness at the time of measurement (1). Another method is to measure the individual's performance during both single and dual-tasks on a treadmill 72 hours after the incident, however similarly this isn't practical for the discussed setting and is more suited to a clinic (2).

A more popular method is The Sport Concussion Assessment Tool, 3rd edition (SCAT-3) and Child-SCAT3 which include a physical exam, symptom checklist, the Glasgow Coma Scale, as well as the SAC to assess cognition, and the BESS and/or TGT for sensorimotor assessment (1, 2). This method is favoured as it is reliable and contains a large range of tests that cover many possible indicators, however it takes around 15-20 mins to administer which limits its use in some time sensitive situations (1,2). As above, it requires baseline data and may not be practical to administer in a sport setting as it must be performed by trained medical staff (1,2,5). It is also viewed as subjective as it relies on the administrators experience, and there is some debate on if it should be used as a stand-alone tool, (1,5). Additionally, the reliability of this test decreases from 3 - 5 days onwards after the injury, meaning that it cannot be used in detecting and monitoring long-term effects (2). The King-Devick Test (KDT), as briefly mentioned above, is used for concussion screening and consists of multiple rounds of a visual test in which the participant must quickly read aloud unevenly spaced single-digit numbers (1,3). The participant reads from three different cards in order to obtain a score which is the total time taken to read all the cards (1,3). This test also supports the theory that eye movements are useful in evaluating an individual's cognition (3). As in other methods, a baseline score is required in order to measure the difference in an individual's performance after an incident, and as healthy individuals' scores often improve with practice and exercise an increased score suggests that an individual is concussed (1,3). This method is appropriate for sideline screening as it only takes around 1-2 minutes to perform and doesn't have to be administered by a medical professional (1,3), however its sensitivity (reported at 75% (1) and 86% (3)) isn't high enough to qualify it to be used as a standalone tool. Studies referenced in both 'Clinical evaluation of concussion: the evolving role of oculomotor assessments' and 'Ocular motor assessment in concussion: Current status and future directions' have found that the combination of the SAC, BESS and KDT has a 100% sensitivity rate. This is extremely good but again is impractical for quickly diagnosing sport concussions on site.

Methods that use visual tracking have also been explored. When using visual tracking, significant changes in an individual's performance were found using video-oculography after a suspected concussion incident and were correlated with a reduction in attention and working memory (1). Furthermore, the eye-movement tracking results correlated with SAC and SCAT-3 results, and further investigation found this had 88% sensitivity (1). A study using military patients also found that concussed individuals were less able to track smoothly moving targets and stepped moving targets in contrast to the control group (3). There are many commercial products available that use visual tracking and/or VR to diagnose concussions (1). King Devick Technologies Inc. have developed a concussion screening tool that uses the KDT (9). This product includes eye tracking software to monitor the patient's movement during the test and produces a range of different results including if the patient is suspected to have a concussion, statistics on past completed tests, and if professional healthcare should be sought (9). This costs $20 per athlete for sideline screening with the eye tracking system sold separately (accessed 28th Aug. 18) (9). While The King-Devick Test is being favoured as a preferable solution, Fischer et al (2016) assert that "further studies are required to establish this test as a viable stand-alone sideline assessment method to be implemented as a return-to-play protocol" (5). EYE-SYNC is an FDA approved system that provides eye-tracking utilities in order to record and analyse patient eye movement to detect any impairments (10). It provides objective results such as eye movement paths, eye movement metrics, and the individuals' scores over time (10). To test for a concussion the patient will follow a predictive target moving in a circular motion and the spatial and timing variability of their gaze will be measured, this task is often referred to as smooth pursuit (3, 10). The VR technology used is designed to be portable and aims to get results in under a minute (10). Test results are stored on a database and access to training materials is also provided with the product, along with access to other resources such as SCAT-5 and BESS (10). ImPACT Applications Inc. have also developed an FDA approved suite of tools to help with the detection of concussions (11). This consists of an approximately 25-minute test that records any symptoms the patient is experiencing and also measures cognition through various tasks (11). This data is then used to produce a medical report containing a variety of results, and different variations of the toolkit are available for different situations; for example, ImPACT paediatric and ImPACT Quick Test (for mobile applications) (11). Like the other products the concussion assessment test also relies on baseline data, but if none is available it is able to utilise normative data (11). A prevalent issue with many existing concussion detection systems is that as a baseline measurement of player ability is often needed, some individuals may memorise parts of the evaluation or have purposefully underperformed during their baseline assessment in order to reduce the chance of deficiencies being detected (3). Additionally, baseline tests can be influenced by unintentional factors such as learning and attention disorders or poor effort (12). However, participants are not always able to do this with some systems, The ImPACT Test has red flags and validity indicators to combat this that can detect when an individual is underperforming (12). The cost of this system is $999 annually, or individual tests can be purchased at retail rate for $15 - $30 (accessed 28 Aug. 18) (11).

Several studies have also been conducted that are closely related to this project. Fischer et al (2016) built a portable and cost-effective solution which was used to track eye movements during a smooth pursuit task and facilitate concussion detection (5). This paper also highlighted the need for a device that was able to be used outside of laboratories and hospitals (5). Their resultant product was suitable for pitchside use and didn't need a computer during administration (5). With a test duration of roughly 5 minutes it was able to produce results of approximately 0.87° accuracy (5). This product could be used to assess the degree of injury and enable the administrator to make a more informed decision on whether an athlete should return to the current activity (5). A medical practitioner is not needed to administer this system which is significant as one may not be available at all sporting matches, such as amateur groups (5). In 'Concussion detection using a commercially available eye tracker', Tobii eye tracking technology was used to create another cost-effective concussion detection system suitable for clinical, pitch side and battlefield use (6). This system measured the users gaze in comparison to the target stimulus in order to quantify their attention and reach a diagnosis (6). A baseline recording for each patient was used in order to assess if they were performing the task (following a circle moving in a pattern) at their normal capacity (6). Tobii eye sensors sample at 60-90 Hz and can produce eye tracking data and images for analysis (6). The system consisted of the eye tracking sensor attached to a computer running the diagnosis GUI and program, and the computer received the raw sensor data via USB (6). The product outputs both numerical and graphical results and in this, doctors will be able to see the difference between the users gaze and the target throughout the task (6).

Okdeh et al (2017) concluded that the built system was able to accurately measure eye movements and that using the score output at the end of the test, among other results, a clinician would be able to make a well-informed decision on the users state (6).

The existing systems discussed above provided insight for this study into appropriate approaches as well as technology requirements and limitations. The inherited knowledge was used to shape the design of the product, and also provided the ability to directly compare the developed system with current solutions and evaluate its performance.

3 Analysis and Specification

In order to develop the specification, an analysis of the problem and current solutions was conducted. This is reflected in the above chapters and resulted in the primary scenario of the products intended use: the sidelines of sporting activities. Currently used systems and relevant papers were investigated and evaluated, and the findings were used to develop requirements and guide aspects of the designed system.

3.1 System Overview

The solution uses the FOVE headset which retails at $599 (accessed 28 Aug. 18) (13) to obtain the raw eye movement data and an application developed in Unity to execute the tasks and record all data. The headset is used in conjunction with the provided FOVE camera to track head position. A MATLAB script is used to process the resultant data and to produce the final report, which in turn is used to assess the patient's condition from comparing it to the baseline report generated.

3.2 Product Aims

- The solution should be able to accurately track eye movements
- The output report should be detailed, useful and concise
- The overall process should take less than 30 minutes
- The product should be portable
- Minimal staff training will be required to use the product
- The solution should be cost-effective

3.3 Product Features

- The system allows input and validates user details
- The system performs prosaccade and antisaccade tasks
- The system saves user details alongside their eye movement data
- The system loads and analyses user data to produce a report
- The system automatically scales all report figures
- The system reports numerical values including errors made, non-responses and mis-recordings

3.4 Deliverables

- Software that executes the saccade and antisaccade tasks and records user data
- Software which analyses the recorded raw data and produces a report containing an evaluation of the user's performance

3.5 User Characteristics

Two users are required to use this product:

- The primary user wears the headset and performs the tasks to the best of their ability
- The secondary user to enters the primary users details, monitors their progress, and after task completion uses the software to generate a report

3.6 Constraints and Dependencies

- The software must be able to sample eye movements at a high enough rate to be accurate, this should be 120 Hz
- A windows laptop or computer with the appropriate ports (HDMI 1.4, USB 3.0, 2x USB 2.0 (13)) is needed to power the headset and run the software
- The laptop or computer used must have sufficient processing power to run the software
- The laptop or computer used must have sufficient battery life or a power source, so the software can be run completely

3.7 Non-Functional Requirements

- The system must be able to reliably record data and produce a report
- The systems performance must be testable
- The system must be useable with minimal training
- The time taken to generate a report must be under 1 minute

4 Design

The design of the software in this project drew inspiration and rationale from previous studies. The scope of influence from these resources will described in more detail in this chapter.

4.1 FOVE and Unity Design

Structurally, the software can be seen as consisting of two components. The first component is an executable application that runs the saccade and antisaccade tasks and records both user data and eye movement data. Please refer to 'Figure 1' for a visual representation of this.

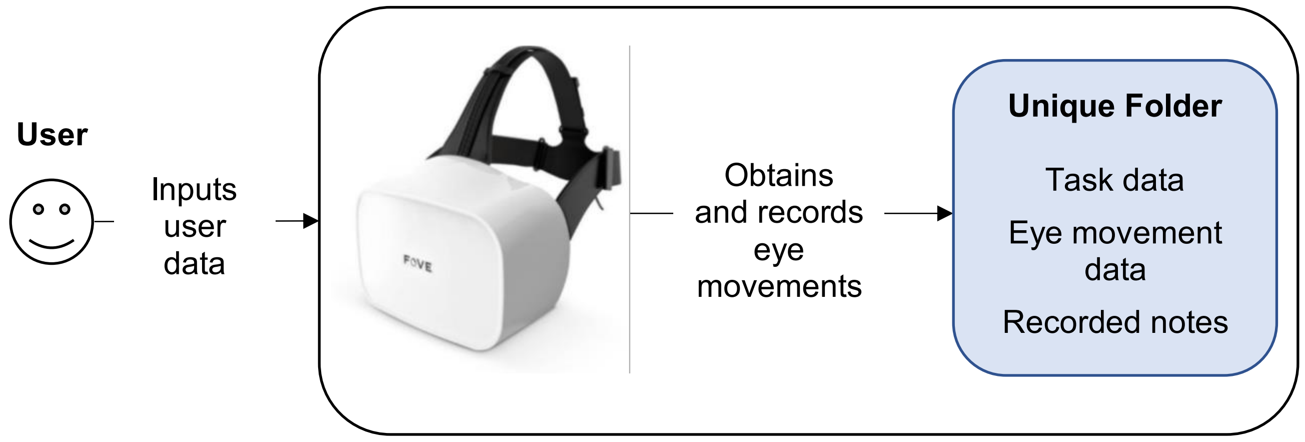


Figure 1: Summarised workflow of the first component.

Upon launching, this component makes use of the FOVE plugin in Unity and calibrates the headset to the primary user. The secondary user, for example a sports coach, then enters the users initials and any additional notes into a panel. These notes can contain information relevant to the incident, symptoms the primary user is experiencing, or anything else deemed appropriate. The input data is validated and upon being accepted by the system, task execution begins. The input initials are validated by ensuring that the text field isn't empty and only contains alphabetical characters, whereas the notes may be omitted and can therefore be left empty, and can also contain punctuation and numbers. Once validation is completed the 'while loop' waiting for correct input is exited and the antisaccade and prosaccade tasks begin. These run in the order 'prosaccade, antisaccade, antisaccade, antisaccade, prosaccade', and the number of rounds per task are passed as an argument. Within the implementation, there are 40 rounds per antisaccade task and 60 rounds per prosaccade task as they are less tiring, giving a total of 240 rounds (14). The paper 'An internationally standardised antisaccade protocol' provided framework into suitable task design for the presentation of stimulus (14). The date and time are generated by the system and are combined with the input initials, this is used to create a unique folder which will store all data generated by the test run. Data is saved to the folder at the end of each task, this means that if the test is cut short the user will still have recorded data and can therefore generate a report.

The prosaccade and antisaccade task require the same stimulus to be presented on screen, the difference between the two tasks is where the user is required to look in relation to this (1 - 3, 14 - 16). The targets are three circles placed in the centre of the screen, laid out in a horizontal line. A horizontal display was chosen as this has been found to be easier to record (14). Please refer to 'Figure 2' for a visual representation of this. Initially only the centre circle will be visible and will then disappear for 200ms, the right or left circle will then appear alone for 1 second (14). The timing of the centre stimulus' appearance is random but is constrained between 1 and 3.5 seconds, as stated by Antoniades et al (2013) (14). During the prosaccade task users must look from the centre to the appearing stimulus, whereas during the antisaccade task the user must look in the opposite direction and 'mirror' the targets appearance (1 - 3, 14 - 16). Please refer to 'Figure 3' and 'Figure 4' for a visual representation of prosaccade and antisaccade tasks.

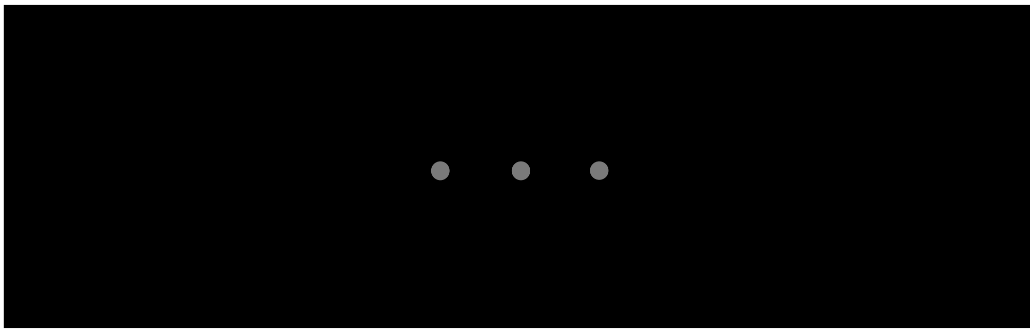


Figure 2: Representation of stimulus layout.

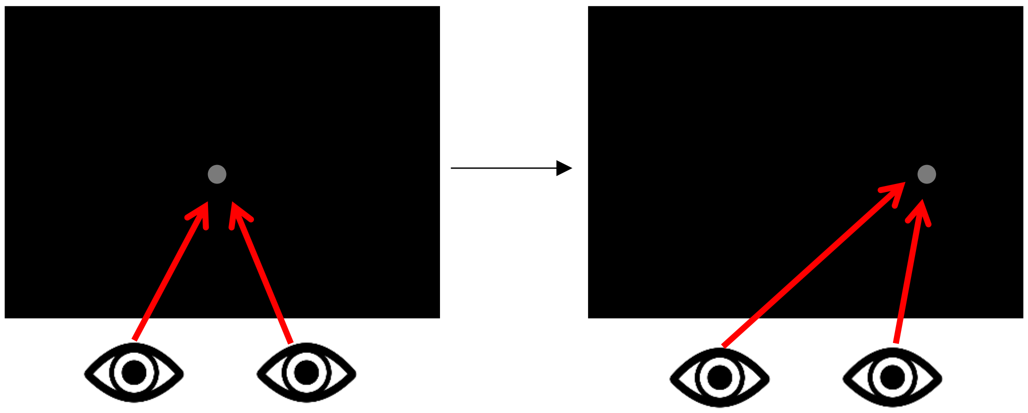


Figure 3: Representation of the prosaccade task.

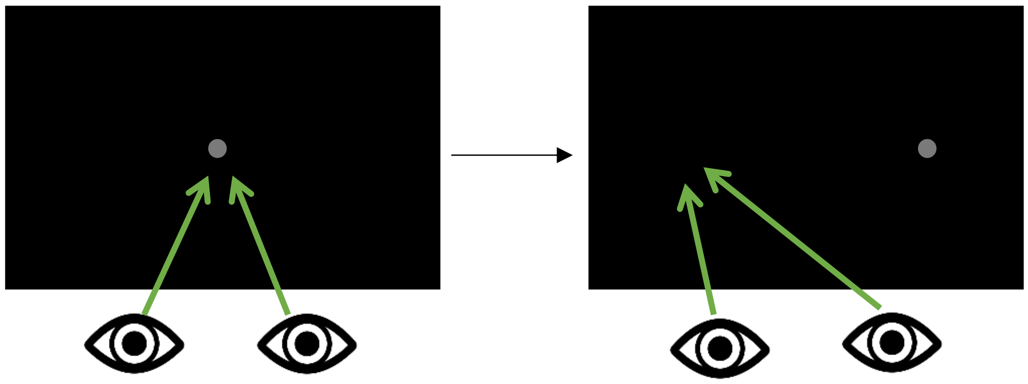


Figure 4: Representation of the antisaccade task.

Prior to task execution, text is displayed that indicates which of the two tasks is to be performed. The task execution method is passed two arguments: the number of rounds, as previously mentioned, and whether the task is prosaccade or antisaccade. The number of rounds dictates how many times the method is to be run, this is done with a ‘for loop’, and is also used to ensure the target stimulus appears on equal amounts on each side. After the completion of each round the user may have a break. Similarly to the algorithm used for inputting user details, a ‘while loop’ is entered, however in this case it is terminated when the user presses any key. This will resume program execution and move on to the next task.

Eye movement data is captured throughout the applications execution, but it is only recorded during tasks. The position of each eye is gathered separately in each frame and the framerate in Unity was altered so that it would sample at 120 Hz, instead of the default 50 Hz, to get richer data. Recorded data is put into three files which are saved in the test run folder: ‘stimulus.csv’, ‘test.csv’ and ‘notes.txt’. ‘Stimulus.csv’ is updated at the end of each task and contains the details of the time each round started, the time the stimulus appeared and whether the stimulus appeared on the right or left, and finally if each round was prosaccade or antisaccade. ‘Test.csv’ contains the time of each frame and the corresponding x and y positions of both the left and right eye. ‘Notes.txt’ consists of any additional details that are input along with the user’s initials, if no notes were entered the file is not created. Filenames are kept consistent between users as this reduces the pre-processing the analysis program has to perform, and as each test run has its own unique folder clashing filenames are not an issue. As mentioned above, the files are only created or updated at the end of each task, this allow partial completion of the test while avoiding partially completed task runs from affecting the data.

The FOVE headset was deemed the most suitable hardware choice in comparison to other VR headsets, this is largely because it has built-in eye tracking capability and a Unity API and plugin readily available (13). This meant that it could reliably record accurate data, not just as a standalone product but also during integration with the developed software. Unity was also seen as an advantageous choice as it is compatible with FOVE and is designed to facilitate the programming of visual components, for example in game development. The paper ‘Portable video-oculography device for implementation in sideline concussion assessments’ outlines key issues of portable video-oculography devices such as processing power, stability of the camera with respect to the position of the user’s eyes, and lifespan without power, however, the FOVE headset is able to tackle these issues (5). FOVE is capable of sampling at 120 Hz which is sufficient for detailed data, especially in comparison to the maximum of 90 Hz in ‘Concussion detection using a commercially available eye tracker’ (6, 13). The device is worn around the head with adjustable straps so that it can remain stable, and as the headset must be connected to a computer or laptop worries of it running out of power are not an issue, care must be taken however that the supporting device has sufficient battery life (13).

4.2 MATLAB Analysis Design

The second component of the system reads and analyses the saved data and produces a user performance report. Please refer to ‘Figure 5’ for a visual representation of the components workflow.

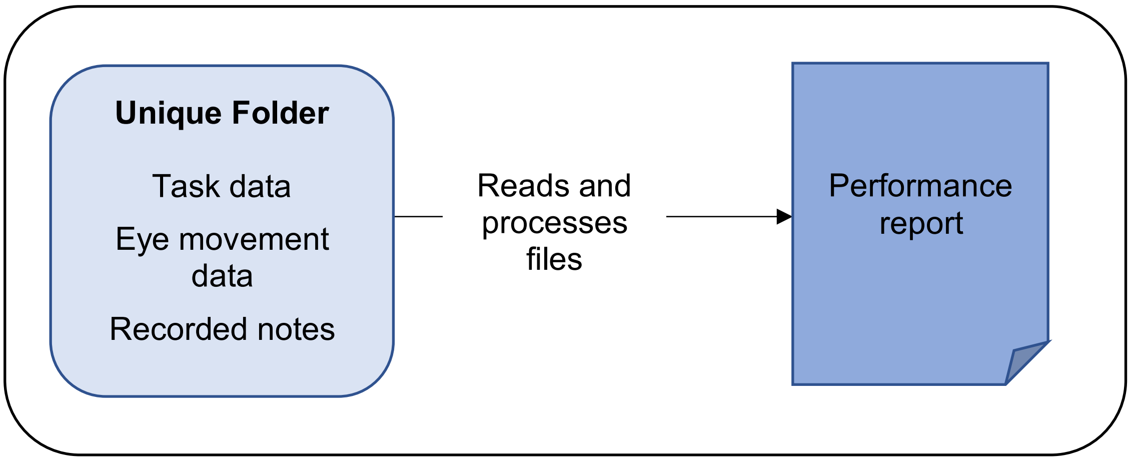


Figure 5: Summarised workflow of the second component.

MATLAB is used to analyse the data and produce a report. First the files ‘stimulus.csv’ and ‘test.csv’ are imported, and the initials of the user and the time the test was undertaken is extracted from the current directory; these details are inserted onto the front page of the generated report. The report is built using the MATLAB report generator as this provides useful functionality including contents pages, chapters, figure insertion and captions for PDF creation (17). As the developed Unity system creates files consisting of a lot of numerical data (typically around 128,500 rows for ‘test.csv’ alone), MATLAB was a suitable choice for analysis. This is because the recorded data is easily imported into matrices which is the format the language is designed to work with, and it offers fast and accurate processing, visualisation, and analysis as well as having customisable report generation. The complete eye movement paths during all tasks are plotted; these can give an initial insight into how the participant performed the tasks, for example lots of noise can mean that they blinked a lot or struggled to look steadily. Please refer to ‘Appendix A’ for examples of eye movement paths. The data is then pre-processed before analysis; the recorded x-value screen space coordinates are converted into degrees from the centre of the screen and the Butterworth filter is applied – this reduces the slight noise that is common in eye movement recordings so that the data is easier to analyse. Please refer to ‘Appendix B’ for plots that demonstrate the effect of the implemented noise reduction. The Butterworth filter was suggested by Dr Sang-Hoon Yeo and is also used in ‘Eye movement accuracy determines natural interception strategies’ (15). The processed data is then analysed and a variety of metrics are calculated, the basis of which rely on the detection of the start and end of saccades. Saccade detection was based upon the following excerpt from ‘Eye movement accuracy determines natural interception strategies’: “Saccades were detected based on a combined velocity and acceleration criterion: Five consecutive frames had to exceed a fixed velocity criterion of 50°/s; saccade on- and offsets were then determined as acceleration minima and maxima” (15).

The main design decision that was taken was to record both eyes separately, this was done to get detailed data that could be more revealing through separate plots and statistics, and to record eye position instead of where the gaze was calculated to have fallen. Through testing varying implementations, obtaining the eye position x and y values separately via ‘FoveInterface.GetLeftEyeVector().x’ and ‘FoveInterface.GetLeftEyeVector().y’ offered more precise and accurate data than when coordinates were reported together using ‘FoveInterface.GetLeftEyeVector()’, or ‘foveInterface.GetGazeRays()’. Velocity detection was another large decision, and this was chosen as it is more accurate and robust than threshold detection, which was previously implemented. This improvement will be discussed below in ‘Implementation and Testing’. As discussed above, while there is conflicting information on if prosaccades are a useful metric (3,8), both prosaccade and antisaccade tasks and their analysis was included in order to provide more information. Finally, the choice to exclude problematic data from figures such as if blinks obscured a saccade, the users’ eyes were not fully open or were closed, and non-responses, were done in an attempt to preserve the accuracy of the plots produced in the report. Please refer to ‘Appendix C’ to view a report generated from recorded data.

5 Implementation and Testing

5.1 FOVE and Unity Implementation

The main data structure used for data capture was ‘StringBuilder’, this was used to record all data for both ‘stimulus.csv’ and ‘test.csv’. Each file has its own StringBuilder and for every new data sample collected a new string is appended containing the contents of either both eye positions during a frame or the round details. This data was separated by commas and each entry begun on a new line. StringBuilder was chosen as it is designed to have content dynamically added to it and can be written directly to csv, unlike lists, for example, which would need to be processed beforehand. The generation of very long strings didn’t affect performance. The method ‘File.AppendAllText(file,string)’ was used to save all data as this creates files if they don’t exist, and can simply append successive recordings onto existing files. At the end of each task all data is saved, and the user has an unlimited break period until they press a key to continue. Because of this, as each task began the contents of the StringBuilders were reinitialised in order to prevent break data from being unnecessarily saved and processed, and to remove the data from previous tasks.

An unexpected issue that occurred during development was choosing the most suitable way to display the stimulus, which was created using 2D circle sprites. The initial program implementation used just one circle in the centre which cloned and deleted itself on the left or right depending on which side the stimulus was to appear. However, in Unity as the script that controls the program execution was attached to the centre circle, this meant that every circle that was duplicated began to run a separate instance of itself and therefore the script. This meant that the program was being continually and repeatedly executed every few seconds and that the number of sprites increased exponentially. In addition to this, it is bad practice to hardcode object locations as this makes the program harder to update. The second implementation used three separate sprites, left, right, and centre, and used ‘setActive(bool)’ to make them appear and disappear. This also caused an issue as once the centre circle was set to inactive using setActive(false), the script was suspended and program execution halted. The third option tried was to change the colour of the stimulus to match the background when it needed to disappear, however this was changed as it isn’t robust against future updates such as a change in colour scheme. The final implemented solution was to reduce the size of each circle to zero when it needed to disappear and increase it back to normal when it needed to be visible.

The initial implementation for detecting a saccade was done in Unity, each Sprite had a circular collider attached to it that spanned a slightly larger area to allow for minor calibration error. The collider would be triggered once the users gaze hit it and the current time would be recorded to indicate a saccade had been made. This was discovered to be unsuitable because if the user’s initial saccade fell short of the target it wouldn’t be recorded, skewing all metrics. Additionally, the calculations needed to perform this could take up a larger amount of processing power in comparison to what would be needed to detect a saccade after all data had been recorded. This approach was tested with both the colour change and size change visibility updates, however it was incompatible with the size change version as when the circle's diameter decreased to zero, so did the colliders diameter, making it unable to work during the antisaccade task at all.

The algorithms for the executable application were created by implementing the stimulus and its control first, then the eye movement recording, and finally the user input to create a unique saving format for each test run. After the application was fully developed and system testing with participants was scheduled to begin, a practice run application was built to help participants in understanding what was required in the tasks. This implementation uses code from the main application but excludes all data capturing and eye movement recording, and just provides a visual run through of the tasks. This was done to improve participant task comprehension and to prevent the recording of unnecessary data.

5.2 MATLAB Analysis Implementation

The eye movement analysis algorithms were implemented by first reading in all recorded data from the test run folder. This was then filtered and converted into degrees by extracting the coordinate data, calculating the eye positions in terms of screen space, and then using trigonometry to calculate the angle between the eyes (camera point in Unity) and the stimulus. In order to evaluate performance, eye movement data is separated into sections that consist of each performed round, and then the relevant saccade detection and metric calculations algorithms are executed.

Initially, threshold detection was implemented to detect saccades. Two different versions of this were used throughout the projects lifecycle until it was deemed inaccurate and velocity detection was used. Threshold detection relies on two high but realistic values being selected for maximum saccade amplitude for left and right eye movements as positive and negative numbers, respectively. Once the eye movement data on the x-axis crosses either of these thresholds, a saccade is detected and the calculation of its start and end time can begin. The initial threshold detection used fixed threshold values that were obtained by manually looking at the amplitude of saccades in all recorded rounds. Upon further analysis, this was shown to be unreliable as not all saccades would reach this height, so it would either measure a corrective saccade as an initial saccade or falsely record a none response. This resulted in uneven plots and statistics between the left and right eye in the resultant reports. Additionally, as future user data could vary to the recorded test data, using this method meant that there was no guarantee the system would be able to detect the saccades of another user once deployed. The second version of threshold detection improved slightly upon this. The algorithm obtained the position of the eyes during the frame the stimulus appeared on. The two thresholds were then set as +/- a fixed amount, such as 4 degrees, from this position. This solution was more robust and reduced the amount of falsely analysed mis-recordings, however it would still sometimes record a corrective saccade as an initial saccade. As outlined above in 'Design', saccade detection was improved by determining if the velocity of five consecutive frames exceeded 50°/s and gradient was used to calculate start and end times, the ideas of which came from Fooken et al (2016) (15). Please see 'Figure 6' for the final implementation of saccade detection. The argument 'frames' contains the five consecutive frames of eye movement to be analysed, and the argument 'times' contains the times the frames were recorded at.

This update made saccade detection more accurate and reliable and the system was no longer thrown by corrective saccades. It also increased the codes simplicity and readability as the direction of the saccade could easily be determined after it had been detected, whereas before, the code consisted of a series of ‘if-else’ statements that looked for saccades in specific directions and then ensured that the detected saccade was the first one in that round. With this update the report figures became more symmetrical between the left and right eyes, and the system became more robust against the differences in individual’s eyes and headset calibration.

```
function saccade = detectSaccade(frames,times)
   For each frame in frames
      cDeg = change in degrees from previous frame
      cTime = change in time from previous frame
      %  Calculating °/s
      factor = 1 / cTime;
      res = abs(cDeg * factor); % Scaling the results up to 1 second
      % If all 5 frames exceed ±50°/s then that is a saccade
      if res >= 50
         saccade = saccade was found
      else
        saccade = saccade was not found
        return; % Prevent calculation of subsequent frames
      end
   end
end
```

Figure 6: Pseudocode for saccade velocity detection.

Using this method the start and end times of saccades could be found, two different sets of methods were used dependant on the direction of a saccade. If a saccade is to the left, the start time is determined by backtracking along the gradient of the data until the value decreases, and the end time is detected by traversing forwards through the gradient of the data until the value decreases. Conversely, if the saccade is to the right the start time is determined by backtracking along the gradient of the data until the value increases, and the end time is detected by traversing forwards through the gradient of the data until the value also increases. Please refer to ‘Figure 7’ and ‘Figure 8’ for examples of left and right saccades along with their gradients, to illustrate how this detection would work. A key challenge of using this method was ensuring that the Butterworth filter parameters (filter order and cut-off frequency) ensured the data was smooth enough so that noise during a saccade wouldn’t be falsely detected as the start or end, whilst maintaining enough variance in the data so that it wasn’t too smooth for the actual onset and offset to be detected. This was done by repeatedly analysing multiple plots of the original data in comparison to plots of the filtered data and seeing how suitable it was, as well as running the detection algorithm to see if start and end would be calculated correctly, and generating reports to see how metrics and figures were affected.

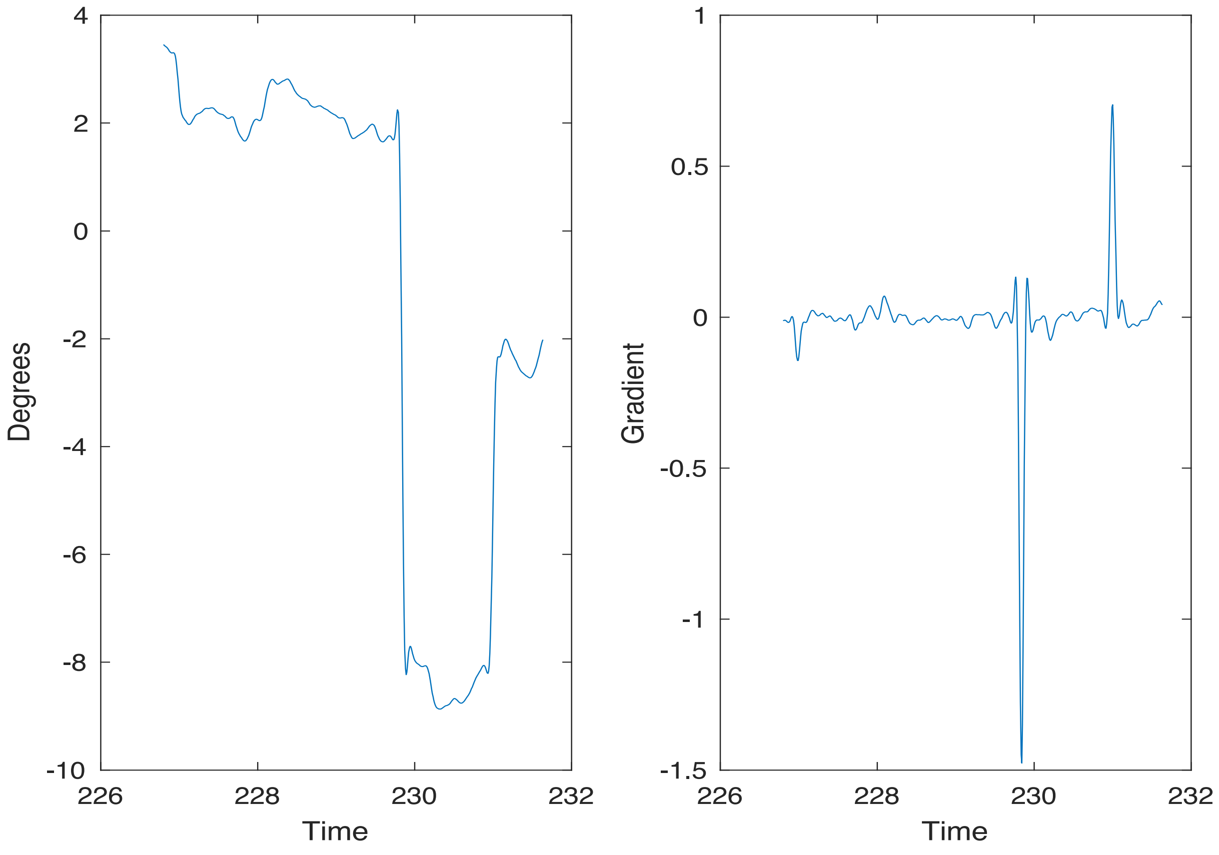


Figure 7: (Left) Raw data of a saccade to the left. (Right) gradient of a saccade to the left.

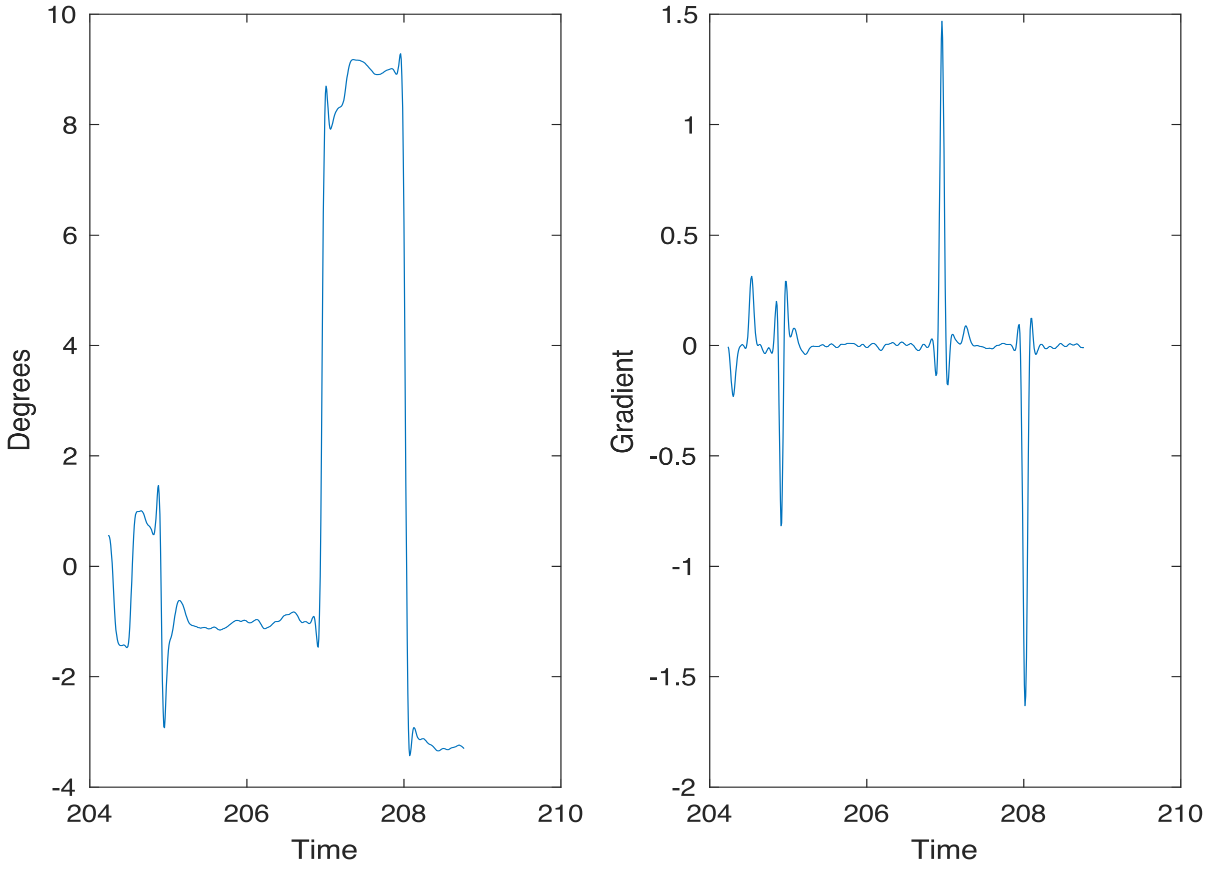


Figure 8: (Left) Raw data of a saccade to the right. (Right) gradient of a saccade to the right.

As can be seen in ‘Figure 7’ and ‘Figure 8’, the observed noise at the beginning of the saccade is due to blinking and the user focusing their gaze on the centre circle. Saccades are clearly identifiable by the quick change in amplitude. The difference in amplitude before and after the saccade is due to the fact that the centre circle is no longer visible, so the user is estimating where its position is. Using the saccade velocity detection and the onset and offset calculations described above, the metrics saccade latency, saccade speed, and saccade duration could be calculated. ‘Figure 9’ below contains the pseudocode for saccade metric calculation using the discussed methods, where ‘data’ contains the eye positions for each recorded frame and ‘time’ contains the time for each recorded frame.

```
 for each round
    % for each frame, from stimulus appearance ‘ind’
    for j = ind:length(roundTime)
        index = detectSaccade(data(j-1:j+4),time(j-1:j+4));
        if a saccade was detected
           index = j % Current frame
           break;
        end
    end
    % Getting saccade direction
    if gradient(data(index:index+1)) < 0
       direction = left
       sTime = getIncInd(index, gradient(data)); % Where gradient stops increasing
       eTime = getDecEndInd(index, gradient(data)); % Where gradient stops increasing
    else
       direction = right
       sTime = getDecInd(index, gradient(data)); % Where gradient stops decreasing
       eTime = getIncEndInd(index, gradient(data)); % Where gradient stops decreasing
    end
    % Example recorded metrics
    Latency = sTime
    Duration = time(eTime) - time(sTime)
    Speed = distance(time(eTime); time(sTime)) / (time(eTime) - time(sTime))
end
```

Figure 9: Pseudocode for metric calculation.

For each metric, a separate loop is used to process the data and plot the results directly after. While this means that the analysis script contains repeated code, it doesn’t significantly compromise on performance and the report is still generated in under a minute. This was done to make the code more readable and offers the opportunity to alter each metric individually if needed.

To give a full example of how the data is analysed after pre-processing, the pseudocode for calculating the latency of the left eye is below in ‘Figure 10’. This also demonstrates how a combination of the above algorithms were used in the system. ‘Data’ is the contents of the file ‘test.csv’, while ‘Time’ is the contents of ‘stimulus.csv’.

```
noneL = 0; % Records number of non-responses
reactL = zeros(length(Time), 2); % Records reaction times and direction looked
for each round i
   % Separate data into rounds using start and end time
   cond1 = Data(:,1) >= Time(i,1); % From start time of current round
   if (i < length(Time))
      cond2 = Data(:,1) < Time(i + 1,1); % To end time (start of next round)
      conds = cond1 & cond2;
      mData = Data(conds,2); % Data for each round
      mTime = Data(conds,1); % Time for each round
   else
      mData = Data(cond1,2); % Data for each round
      mTime = Data(cond1,1); % Time for each round
   end

   % Ignoring fluctuations before the stimulus appeared
   ind = find(mTime > Time(i,2), 1); % Index of data from after the stimulus appeared
   % Difference between vector sizes
   diff = length(mData) - length(mData(ind:end));

   % Detecting a saccade
   % For each frame from ind
   for j = ind:length(mTime)-4
      index = detectSaccade(mData(j-1:j+4),mTime(j-1:j+4));
      if ~isempty(index)
         index = j;
         break;
      end
   end

   if isempty(index)
      % No saccades were made
      noneL++;
      reactL(i,1) = dummy value
      continue onto next loop iteration
   end

   % Aligning index with mData size
   if index < ind
      index = index + diff;
   end

   % Getting saccade direction
   if gradient(mData(index:index+1)) < 0
      reactL(i,2) = left
      sTime = getIncInd(index, gradient(mData)); % Latency
   else
      reactL(i,2) = right
      sTime = getDecInd(index, gradient(mData)); % Latency
   end

   % Record reaction time
   reactL(i,1) = mTime(sTime) - Time(i,2); % Latency after stimulus
end
```

Figure 10: Pseudocode for calculating saccade latency of the left eye.

Corrective saccades were calculated differently as these required additional processing. As in ‘Figure 10’ the data is separated into rounds and the initial saccade or non-response was detected. The task type for the round, saccade or antisaccade, and direction the stimulus appeared on, left or right, is then found. Saccade direction is determined, and the end time of the initial saccade is calculated. Using this information, the algorithm checks if the initial saccade is in the correct direction according to the task type and stimulus direction. If the saccade is correct, then the round is skipped, and its data isn’t plotted. If the saccade is incorrect, the algorithm detects if a successive saccade is made in the correct direction. If a correct corrective saccade is made, the time difference between the end of the previous saccade and the start of the corrective saccade is recorded, otherwise an incorrect response is recorded.

As can be seen in ‘Figure 10’, non-responses are filtered and removed from the final recorded metrics and plotted figures. System data was further filtered during analysis to maintain report accuracy. Blinks before the saccade were omitted to prevent accidental detection as saccades, this was implemented by only processing data that followed on from the stimulus appearance. As saccades typically last up to 100ms (R.J. Leigh et al, 2015. p. 171) the probability of a blink interrupting an initial saccade was judged to be low enough to ignore. Mis-recordings were detected after analysis as metric results that were calculated as being less than zero, which is impossible. This meant that the eye movement data for that round was skewed. Mis-recordings were more frequent in participants who were tired or usually wore glasses, suggesting that it could be due to squinting or their eyes not being consistently open during the test. This is a disadvantage of recording eye movements separately instead of using ‘foveInterface.GetGazeRays()’, as if the gaze was recorded this has the capability of tracking the resultant focal point from one or both eyes.

Matrices were used to store all raw data and to perform calculations during analysis. This approach allowed for the data to be dynamically partitioned into subsections and analysed without creating unnecessary new variables. Matrices were also used to store the results of all eye movement analysis calculations; in MATLAB this reduces the amount of pre-processing that is needed as it is the correct format for many visualisation and processing operations.

The original code in the files ‘draw_mirro_hist_multi_example_script.m’ and ‘draw_mirror_hist_multi.m**’** were given to me by Dr. Sang-Hoon Yeo in order to create dual plots similar to in the paper ‘Look away: the anti-saccade task and the voluntary control of eye movement’, figure 1b (16). This code was used to create compact plots that allow a quick visual assessment of the ratio of correct and incorrect saccades made during each task, for each eye. Please refer to ‘Appendix C’ for an example of this. Some code from ‘draw_mirro_hist_multi_example_script.m’ was used in ‘plotTogether.m’ to display the results in the report, whereas the code in ‘draw_mirror_hist_multi.m**’** was modified to suit the format of the generated report. The axes on the dual plots were altered to automatically scale to the contents of the plotted histogram, this is done by plotting an unseen histogram with specific bin widths for certain metrics and gathering the maximum dimensions from this. Before the histogram is plotted and included in the report, the axes for the left and right eye metrics are calculated and paired so that they will have the same scale; as before this was implemented to make them easier and quicker to interpret. The contents of each subset of data was updated and determined based on if the round consisted of the saccade or antisaccade task, and if saccade analysis determined that the user completed it correctly.

5.3 Application User Input Testing

Within the application, once user input is accepted the entered data is recorded and the ‘data input panel’ is removed. This allows task execution to begin. In cases of the input not being accepted, the input text, if any, is highlighted and the input panel remains on the screen. The following tests were designed in accordance with the following point from the ‘Product Features’ specification: “The system allows input and validates user details”.

Test Number	Initials Input	Expected Outcome	Actual Outcome
1	No input	Input not accepted	Input not accepted
2	“ “	Input not accepted	Input not accepted
3	“vlc1”	Input not accepted	Input not accepted
4	“vlc!”	Input not accepted	Input not accepted
5	“vlc”	Input accepted	Input accepted
6	“VLC”	Input accepted	Input accepted

Table 1: User initial input test cases.

As notes recording is optional in the system, once the field is submitted the input text, if any, is recorded.

Test Number	Notes Input	Expected Outcome	Actual Outcome
1	No input	Input accepted	Input accepted
2	“Strong headache after impact.”	Input accepted	Input accepted
3	“Patient had double-vision for 5 minutes.”	Input accepted	Input accepted

Table 2: User notes input test cases.

5.4 Analysis Testing

A range of tests were performed to examine the robustness of the analysis script, with multiple runs being performed in some test cases. These tests were designed in accordance with the following points from the specification: ‘The solution should be able to accurately track eye movements' and 'The system automatically scales all report figures to the data'.

Test Number	Test	System response
1	Performed normally	Analysis executed as expected, metrics evaluated and plotted correctly
2	Heavy blinking	Large amount of noise in the 'Eye Movement Paths' plots. Slight increase in the number of mis-recordings
3	Closed eyes	Eye position is unable to be tracked and recorded at ((0,0), (0,0)). Only non-responses recorded
4	Non-responses for task performance	Eye position successfully tracked. Only non-responses recorded
5	Correct direction for task performance	Analysis executed as expected, metrics evaluated and plotted correctly
6	Incorrect direction for task performance	Analysis executed as expected, metrics evaluated and plotted correctly
7	Varying round amounts completed (1-5)	Analysis executed as expected, figures automatically scaled and metrics evaluated and plotted correctly

Table 3: Analysis script test cases.

5.5 Specification Testing

The following tests were executed to ensure that the software met the specification.

Test Number	Criteria	Tests performed	Outcome
1	The overall process should take less than 30 minutes	Measured the run time of various test runs	The test typically runs for slightly over 20 minutes
2	The product should be portable	Deployed the application for Windows	It is able to be run successfully without the device being plugged in

3	Minimal staff training will be required to use the product	Evaluated participants running the system as both primary and secondary users	In both roles, participants were easily able to use the system
4	The system is able to perform saccade and antisaccade tasks	Evaluated the systems performance in executing varying round amounts (1-5)	The system runs as expected
5	The system is able to save user details alongside their eye movement data	Evaluated the systems performance in executing varying round amounts (1-5)	The system runs as expected
6	The system is able to load and analyse user data to produce a report	Evaluated the systems performance in executing varying round amounts (1-5)	The system runs as expected
7	The system is able to report numerical values such as errors made and mis-recordings	Evaluated the systems performance in executing varying round amounts (1-5)	The system runs as expected
8	The software must be able to sample eye movements at a high enough rate to be accurate, this should be at 120 Hz	Plotted recorded eye data using '-.' in MATLAB to indicate where each sample was made and analysed the time difference between sample points	The time stamps consistently reflect 120Hz (sampled every $0.008\dot{3}$ seconds)
9	The system must be able to reliably record data and produce a report	Evaluated the systems performance in executing varying round amounts (1-5)	The system runs as expected
10	The time taken to	Evaluated the	All reports are

	produce a report must be under 1 minute	systems performance in executing varying round amounts (1-5)	generated in under 1 minute

Table 4: Specification test cases.

5.6 Comparative Testing

Dae Hyun Kim, a student completing a summer internship at the Centre for Bionics at Korea Institute of Science and Technology is currently studying the reliability of FOVE. During his research he has found that there is a discrepancy between the time recorded by Time.time(), the method being used with FOVE in the developed system, and the time recorded by the system's clock using a stopwatch. This discrepancy is a vital issue that can seriously alter the accuracy of the recorded data, however upon testing it in the developed application it only occurs when the application is minimised, or another piece of software is run. This was investigated by running the two different time methods in tandem and plotting the resultant data. While this may be an issue for some applications, the software should be kept open and running so that the test administrator can monitor the players progress, so this issue shouldn't affect the developed system.

The application was also tested on 8 participants, alongside the ImPACT baseline Test. As mentioned above in 'Existing Systems' the ImPACT Test is a well trusted and varied evaluation that aims to give a detailed insight into different aspects of a user's cognition. For user instructions given before testing, please refer to 'Appendix D'. For user conditions during testing, please refer to 'Appendix E'. In order to compare the results between the two systems, two sets of metric comparisons have been made. Firstly, analysis data is extracted from both reports; 15 metrics in total were obtained from the developed system, and 7 from The ImPACT Test. In the developed system, variables were obtained using the script 'VariableOutput.m'. This saved the mean and variance of correct saccades, correct antisaccades, and incorrect antisaccades for both latency and speed metrics to the participants test run folder. It also saved the participants error rate, and the mean and variance of any corrective saccades. From the report generated by The ImPACT Test the following metrics were manually extracted: memory composite (verbal), memory composite (visual), visual motor speed composite, reaction time composite, impulse control composite, total symptom score, and cognitive efficiency index. These two sets of variables formed the basis for the results comparison. Data was zero-centred and Principle Component Analysis was performed in 'PCAAnalysis.m' to get the three most significant variables from each system. These captured 81.83% of the variance for the developed system, and 87.76% for The ImPACT Test. For the developed system, a 2D and 3D bi-plot of all metrics was generated, this revealed that variance of the reported speed for all saccade metrics had the largest magnitude in the new space. A 2D and 3D plot of the dimensionally reduced data was also generated, and this displayed that most participant results were clustered together. For The ImPACT Test, the generated bi-plots showed that the reaction time composite and cognitive efficiency index were among some of the variables with the largest magnitude, however a noticeable amount of variables were similarly large. Some loose clustering was observed through plotting participant data in the reduced dimensionality as above, but nothing significant. After this, four plots were made by directly comparing variables from both systems to see if any grouping or correlations were present. These plots compared the error rate to the impulse control composite, and the reaction time composite to incorrect antisaccade mean latency, correct antisaccade mean latency, and correct prosaccade mean latency. No results worthy of note were found from error rate vs impulse control composite, or from reaction time composite vs correct prosaccade mean latency. The data obtained from comparing both antisaccade metrics to the reaction time composite however displayed a gradual positive trend, with the data for participant 3 being an anomaly. The results for this are shown in ‘Figure 11’ and ‘Figure 12’ below. Using this, it can be argued that antisaccade latency is a useful metric as it is shown to have a relation with similar results obtained from a trusted test, and that the developed system produced useful results for assessing participants performance. This argument is also supported by many papers that claim antisaccade metrics are particularly insightful (2,3,8,14,16).

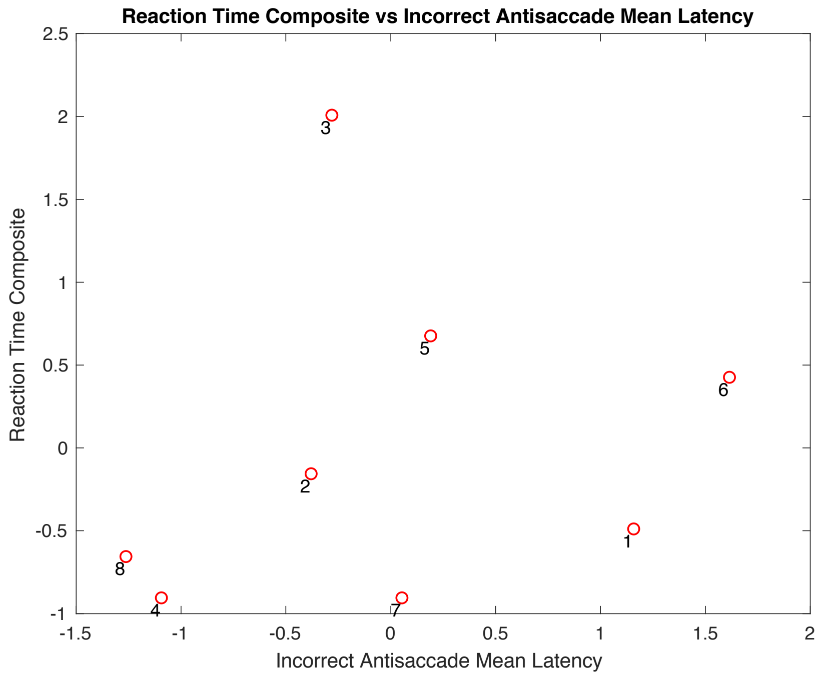


Figure 11: Reaction time composite vs incorrect antisaccade mean latency.

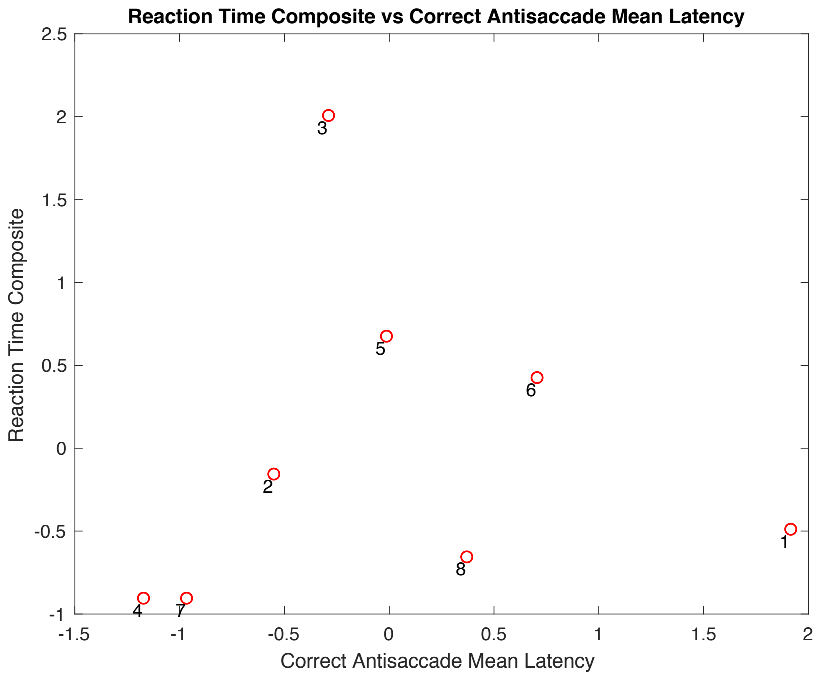


Figure 12: Reaction time composite vs correct antisaccade mean latency.

The order of which system participants tested first, the VR application or The ImPACT Test, was balanced to prevent order effects from skewing the results, and a break of five minutes was encouraged between systems. The user feedback gathered from the test sessions was that the headset was comfortable to wear and easy to use. Due to the task test before each round participants said it was easy to centre themselves to the camera and see the stimulus. As mentioned before, a complaint among two of the users that performed The ImPACT Test first was that they found it hard to focus in the headset afterwards as they felt tired. This has presented an issue as it may be uncomfortable for a concussed individual to wear the headset for 20 minutes. On the other hand, if the received injury is severe enough to cause a high level of discomfort then the system may not be needed in order to decide if it is appropriate for the individual to return to play. Additionally, existing VR systems that are currently used in a sports setting supply evidence that this isn’t a significant problem.

6 User Interface

The user interface for the executable application was designed to be minimal with little user input. This was done so that little staff and user training would be needed in order to understand how to use the system, and so that the majority of the task execution and report generation process would be automated.

After calibration, the user details screen is displayed. This consists of a panel that prompts the user to enter the relevant user and incident information. To further simplify this, it primarily consists of written instructions that prompt the user on what data to input, and a text box. Initially, both user input fields were on the same panel however this was separated and the data capture process was then split into two steps for speed and simplicity, please refer to ‘Figure 13’ for an illustration of this. The secondary user will input details on their laptop, while the primary user can use the display of the panel and instructional text to align themselves with the camera. It is possible to partially fix the presentation of all stimulus in the headset, rather than have it ‘free’ in world space, however this was seen as being disorienting to VR users. Additionally, when saccade and antisaccade tasks are typically done the stimulus is displayed on a screen or shone on the wall using lasers, so all content is designed to mimic this and simulate a ‘wall’ surface within the headset (14).

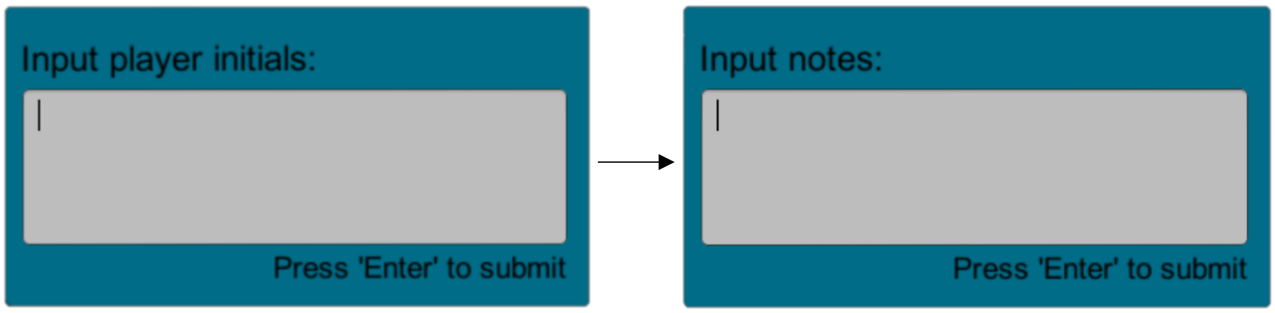


Figure 13: The text transition in the input panel.

Once valid details have been input, task execution begins. First, instructional text indicating the task type will be displayed for 3 seconds, then the task will begin and stimuli will appear. Initially, the application had a white background with black circles, as this was presented as ideal in 'An internationally standardised antisaccade protocol' (14), however in fixing the headset display to 'a wall' only a square area around the stimulus was white, whereas the surrounding world space in the environment was black. This created a problem with the validity of the test as participants could look at the stimulus in relation to its location on the displayed square, not from the centre target. Due to this the colours were inverted so the environment was black and the circles were white. Unfortunately, very bright colours against a black background emit glare in the headset. This is an issue as it is distracting to participants during testing. The white stimulus was darkened to grey in order to remove the glare; a greyscale colour scheme was used for the task stimulus to accommodate for colour-blind users. As recommended in the protocol the circles are 10 degrees apart, horizontally laid out and stimulus is presented equally on both sides (14). Performance feedback during the test is not given, as laid out in the protocol, however in contrast to the protocol the practice runs were included as a separate application (14). As mentioned in the participant instructions in 'Appendix D', practice runs are available for the user to get a feel for the test and understand what will be happening beforehand. Due to this, users are free to do a combination of antisaccades and prosaccades in any order within the practice application therefore feedback was not given during this. Additionally, in contrast to the protocol, regardless of the number of errors made in the antisaccade task a prompt is not displayed, this is because all saccade analysis is done after task completion (14). Participants are able to control the length of their break instead of it automatically being set as 1 minute, as stated in the protocol (14). This was implemented so users have more awareness and control, and it suits the varied nature of the testing environment. Finally, as discussed above the system also allows users to terminate testing early while still being able to generate a report. While this is not advised to get an in-depth report, it was done for use in extremely time sensitive situations and again to give users more control.

For ease of use, report generation was designed to use one-click analysis as this also speeds up the entire process. All data, plots and results presented in the report are clearly labelled so that it is informative and clear. The format of the saved data was done not only to make the analysis script more general, but also to provide an easily searchable format for users. The generated report will be saved into the same folder as the participants raw data, and as this is named using the date and time of the test and primary user initials, it is very accessible for the secondary user to review.

7 Project Management

The initial creation of the basic software was completed quickly, this was then refined over a number of weeks to create a polished product using incremental development. Feedback was obtained in weekly meetings with Dr Sang-Hoon Yeo, as well as from reviewing the results obtained through testing the system. During implementation, the task execution application was written first, and then the analysis script. Once these systems were integrated through writing and reading data from the same location, and the preliminary versions were completed, they were then improved in tandem. Please refer to 'Table 5' for the initial schedule from the project proposal, and if the milestones were completed on time.

Action	**Start Date**	**Due Date**	**Deliverables Expected**	**Completed**
Initial preparation: • Perform a preliminary literature review • Produce a project proposal	14/06/18	21/06/18	Literature review on concussion detection Project proposal for 'Using a VR-based health monitoring system to detect concussion'	Yes Yes
Learn Unity: • Follow introduction tutorials • Familiarise self with workspace • Investigate resources on eye tracking	22/06/18	25/06/18	Completed Unity tutorial projects	Yes
Establish project requirements: • Outline system features and requirements • Outline task(s) to be implemented	26/06/18	28/06/18	System specification	Yes
Design system software: • Plan and design what needs to be implemented	29/06/18	01/07/18	List of types of task to be implemented	Yes
Assessment deadlines: • Prepare ethics form • Prepare for inspection week Implementation and unit testing: • Implement the designed software task(s) and test independently • Develop each task • Track eye movement and record the data • Make a decision based on the	02/07/18 02/07/18	09/07/18 05/08/18	Completed ethics form Functioning project to discuss Implementation of tasks in Unity Recorded eye movement data Multiple reports/statistics to	Yes Yes Yes Yes Yes

baseline and task data • Output report / statistics			compare Detailed report	 Yes
Integration and system testing: • Combine the implemented task(s) to produce a complete system • Test the complete system	06/08/18	12/08/18	Implemented tasks are combined to form a complete system Resultant reports	Yes Yes
Presentation and demonstration week • Prepare project demonstration	20/08/18	26/08/18	Slides, relevant papers and code as presentation aids	Yes
Dissertation write up: • Compare system results to another form of concussion detection e.g. ImPACT questionnaire • Produce the dissertation, including refined existing documents.	13/08/18	11/09/18	Participant test results for both systems Analysis of results Final document 'Using a VR-based health monitoring system to detect concussion'	Yes Yes Yes

Table 5: Project proposal schedule.

Upon reflection, it would have been better if the implementation schedule had been broken down further. For example, allowing set time for the implementation of the saccade and antisaccade tasks individually may have provided a more concrete plan and increased the projects potential to include further task or analysis implementation. In the initial meeting for the project, a range of possible tasks were discussed and it was decided that saccade and antisaccade tasks would be the most appropriate to initially build, but that the project could also include others if there was time afterwards. Due to this, a limit wasn't set on how many tasks could be implemented and it provided the opportunity to build a test suite. In addition to this, the best implementation methods were not always researched beforehand. For example, threshold detection was suggested as the most suitable for the system, however if detection methods and noise filtering were independently researched more this may have saved time and reduced the number of problems encountered.

A strong positive decision for project management was the choice to improve the existing implemented tasks along with their analysis, rather than to include another task such as smooth pursuit and develop a test suite. This meant that the final developed project was more stable and robust whilst still remaining suitable for the objectives. The addition of more tasks to the project may have demonstrated an even wider range of technical ability, however the inclusion of multiple tests may be time consuming for the participant and reduce the secondary user's ability to directly compare and consistently measure participant results. If participants didn't consistently take the same tests, this would affect recovery monitoring as well as baseline measurement comparison as the reported metrics would differ. Another strength of the projects management was that time was allotted in order to test the system thoroughly, both experimentally and with participants. This provided realistic and varied data that was useful during the analysis of the system.

8 Results and Evaluation

The aims of the project were consistently considered throughout design and development, and both deliverables from the specification have been built. The system is deployable on portable devices and the technology used doesn't need a power source, meaning that the systems area of operation isn't limited to a laboratory or clinic. While Unity is able to deploy the application for a range of different platforms, FOVE is currently only compatible with Windows. It was found that running the full system on the tested devices didn't have a largely increased drain on battery, and the system can be seen as reliable as a range of testing situations, including varying round lengths and different participants, were successfully evaluated to produce satisfactory reports with correctly generated statistics and automatically scaled plots. Repeated runs of the analysis script on the same data produced identical reports, again demonstrating the stability of the system. From the above points, the system can be considered as able to accurately record and analyse data and be used to gain more insight into brain functionality. Eye movements are accurately tracked by the headset and application as evidenced by the test cases detailed above and participant test runs, however a usually small mismatch in mis-recordings and non-responses between the left and right eye is present in reports. The data for these rounds were observed, and mis-recordings were frequently due to participants making a saccade during the presentation of the stimulus, instead of focusing on the centre circle. Additionally, as displayed in Test Case 3 in 'Table 3 - Analysis Testing' the system is sensitive to closed eyes. This was also noticeable in some participants that had performed The ImPACT Test prior to testing the developed system, as they reported feeling more tired before beginning the experiment and afterwards reported that they felt their eyes were closed or unstable for periods during it. Due to this their reports contained a slightly higher number of mis-recording instances, with one participant having a significantly larger amount. While this is consistent with reports that tiredness can mimic underperformance effects similar to concussions (6), it also highlights a potential issue with using a VR headset; however further studies would have to be performed to assess the products practicality in its intended setting. For example, the developed system could be tested using participants mid-way through a sporting activity to see if tiredness from this resulted in the same feedback, or if this was due to participants sitting still in a lab for an hour. As previously mentioned, the dual plots in the report were designed to mimic the figure 1b from the paper 'Look away: the anti-saccade task and the voluntary control of eye movement' (16). As can be seen in the second chapter of the example report provided in 'Appendix C', the recorded latency results are extremely close to the results of the paper in figure 1b under the same gap condition of 200ms (16). This also provides evidence that the system is able to accurately track eye movements as it has obtained similar results to an existing study. From looking at the speed and duration dual plots present in the report supplied in ‘Appendix C’, it is clear that antisaccades have a more varied amplitude than prosaccades. This is likely due to the fact that users have to assess where the target stimulus would be and suggests that saccade amplitude can provide an insight into brain functionality. As discussed in the introduction to this paper, this supports the argument made by Ventura et al (2016) but conflicts with Rizzo et al (2016). The evaluation of saccades can also serve as a way to monitor patients’ conditions, as once symptoms recover their saccade results also return to normal (3). This means that the developed system can be used outside of the initial incident to monitor progress.

Another significant aim for the system was for the overall process to be under 30 minutes, so that participants could realistically be evaluated during a match and have the opportunity to return to play if appropriate. As the tasks take approximately 20 minutes to perform, which is consistent with the guideline laid out by Antoniades et al (2103), this would be feasible in a practical setting while still being long enough to feel that enough data has been obtained to be insightful (14). Additionally, as minimal user input is required and the system is cost effective it can be presented as a practical solution; especially for amateur events and clubs that may not have continual access to a trained medical professionals. As the system doesn’t host data online payment plans per test wouldn’t be necessary for large-scale deployment due to heavy usage, making it more accessible for a wider audience range which may not be able to get additional funding, again such as amateur events and clubs.

In all user input test cases the system accepted correctly input data, executed the tasks as intended and recorded all required user data. Moreover, eye movements have also been consistently recorded at 120 Hz and saved to the correct location. Upon running the analysis script on saved data, reports have been correctly generated in under one minute, further demonstrating the reliability and correctness of the solution. Obviously, data processing times increased with the number of tasks performed, however this is due to the volume of data being processed and the system was still able to efficiently handle it within the time limit. Input data validation response times are immediate, as is user input during tasks, for example pressing a button to continue. When unexpected data has been entered, such as incorrect user input or unusual eye movements, the test cases above illustrate that the system responds correctly and doesn’t crash. In addition to this, the system is equipped to deal with missing data; if the required eye movement analysis files don’t exist then the system will stop execution, whereas if the notes file doesn’t exist the system will simply not include any notes in the final report.

A significant advantage of digital solutions over paper-based solutions is that physical storage, which can be costly and difficult to organise for a large number of participants and tests, is not needed as all data is automatically organised by the system. Digital storage can also make it simpler to keep data protected, for example through encryption. As the generation of data and reports uses the current date, time and user initials there is no chance of a clash during saving or reading data. In addition to this, systems that utilise eye-tracking technology reduce the participant’s ability to rehearse sections of the test, such as memory-based tasks or the paper implementation of The King-Devick Test. Some basic tests, for example symptom checklists, are very portable and cost effective however results can easily be skewed by participants choosing what to report. This can be prevented by using tests which focus on a participant's ability to perform a task. Taking that into consideration, a symptom checklist is a good way to gather and record qualitative data on the incident and how the participant was affected. A symptom checklist is included in The ImPACT Test with a set amount of metrics such as headaches, dizziness and nausea, which the participant numerically rates the severity of from 0 - 6 (11). Symptoms can also be input into the notes section of the developed system but are not processed, unlike in the ImPACT test, to give a symptom score (11). The input notes are free text so the user can include detailed data, however as there are no symptom prompts this can mean symptoms may be overlooked if the participant doesn't mention them. Unconstrained input does however, provide the opportunity to input additional symptoms and information not covered in a checklist.

A notable system in sideline eye-tracking and concussion detection is The King-Devick test produced by King Devick Technologies Inc. (9). Similarly to the developed system it is portable, quick and requires a baseline test in order to make a comparative evaluation (9). A strong advantage of both systems is that they make use of an established test and measurement metrics (9). The KDT solution is however more expensive, and the sensitivity for the paper-base system is reported at 75% (1) and 86% (3). This has meant that there has been some speculation on if it is high enough for standalone diagnosis. An advantage of the developed system over the King-Devick Test is that the implemented tasks are harder to rehearse outside of the headset, and intentionally alter performance on. The EYE-SYNC system provides detailed information with a range of results such as eye movement paths, eye movement metrics, and individuals' scores over time (10). This is very similar to the developed system, which also reports eye movement paths, eye movement metrics, and past reports can be viewed to see an individual's score over time. EYE-SYNC uses the smooth pursuit task; conducted studies have found that concussed patients have greater difficulty in performing smooth pursuit accurately in comparison to the control group, giving evidence of this being an insightful test (3). The EYE-SYNC system is portable and aims to get test results in under a minute (10), this matches the time taken for the analysis script to run. While the developed system has many similarities to the EYE-SYNC system, it has undertaken a new approach through using saccade and antisaccade tasks, as opposed to smooth pursuit which was also implemented by Fisher et al (2016) and Okdeh et al (2017) (5,6). Fisher et al (2016) created a portable, custom built and cost-effective solution which was used to track eye movements during smooth pursuit and facilitate concussion detection (5). Their test takes around 5 minutes to administer and is able to produce results of approximately 0.87° accuracy (5). Fast testing is clearly beneficial in a sport setting for quick decisions, provided the data and results obtained are reliable and accurate. Through building a headset instead of using a commercially available product such as FOVE, they were able to develop a device that could be used without being connected to a computer (5). This is advantageous as although laptops are portable, the developed FOVE system in this paper relies on the computer being stable in order for the user to clearly see the stimulus. Okdeh et al (2017) used the cost-effective Tobii eye tracking technology, this measured the users gaze in comparison to the target stimulus during smooth pursuit (6). The developed system in this paper samples data at 120 Hz, which is almost double of the Tobii eye sensors, which sampled at 60-90 Hz, providing richer data (6). Both systems are similar in that they consist of an eye tracking sensor which is attached to a computer running the software, and as is common in the other mentioned systems, both systems utilise baseline measurements to analyse participant results (6). In the Tobii system the computer is running the diagnosis GUI and program, this receives the raw sensor data via USB and outputs both numerical and graphical results (6). From this, doctors will be able to see the difference between the users gaze and the target graphically throughout the task (6). The task analysis script in the developed system also displays both numerical and graphical results, however the users gaze in comparison to the target stimulus is not displayed.

The ImPACT Test records any symptoms the patient is experiencing and also measures cognition through various tasks, this takes approximately 25-minutes to complete (11). The longer test time and range of tests can provide more detailed data, however as this is the longest of all discussed tests it may not be the most favourable for quick pitchside detection. The ImPACT Test requires baseline data however it is able to utilise normative data if none is available, this is an advantage over the discussed systems as it can potentially prevent intentional underperforming of baseline measurements (11). The feedback obtained from participants testing the developed system in comparison to The ImPACT Test was that there was no discomfort in wearing the adjustable headset in comparison to using the computer screen. Participants reported finding the ImPACT test harder than the developed system, and some asked for clarification of instructions for some tasks. In contrast, participants found the saccade and antisaccade tasks were simple to perform and were clearly explained. As stimulus timing is random in the saccade and antisaccade tasks, in some rounds where it was quicker than others participants reported having to supress their head movements. This presents the possibility of the additional metric ‘head movement’ which was interpreted in ‘Virtual reality as a screening tool for sports concussion in adolescents’ as “deficits in inhibition” and this paper also reported that previous studies have “demonstrated a link between sports concussion and cognitive deficits of this type” (4). The finished product can be seen as a positive demonstration of how technology can effectively be used to aid pitchside concussion detection. As reviewed in this chapter, the results obtained establish the practical feasibility of the product for the intended setting, while highlighting that a different test to what is already commercially available is able to provide similarly beneficial results.

9 Discussion

Considering the analysis of the system individually and in comparison to existing VR systems, a key achievement are that it is able to get accurate data readings at a high frequency without compromising on performance. The system is able to produce a performance evaluation report with in-depth metrics similar to existing systems and also contains informative graphical representations of this. Both the task design and evaluation metrics created reproduce sections of the antisaccade protocol laid out by Antoniades et al (2013), as well as reflecting what alternative systems report. This provides a strong argument that chosen design is useful and justified. Considering this, throughout the project the system was designed to meet not just the protocol guidelines, but also the project aims and objectives. The results of this project have displayed the promise and potential for using VR as a sideline concussion diagnosis tool, while introducing a different task than smooth pursuit. Through participants evaluating The ImPACT Test and the developed system, participants were able to have a strong reference point of what a currently used system is like; this enabled them to provide useful and relevant feedback. Comparing and evaluating the results of both systems provided evidence that the data obtained, in particular from antisaccades, is accurate and consistent with other findings.

The main deficiency of the project in terms of planning is that more research into optimal processing strategies would have improved the initial progress. However, this provided a good learning opportunity due to the individual development and analysis of different techniques. As mentioned above, a limitation that could be further investigated is how tiredness and how open eyes are, affects the system with participants in a sports setting. Additionally, while the system checks if all used files exist, there is no protection against corrupted data. For example, if some of the data within a file is deleted the system is not equipped to handle this, therefore the system could be modified to combat this.

The development and evaluation of this project have presented several ways in which the system could be further extended. For example, a suite of tests could be developed through incorporating another task, such as smooth pursuit, and analysing the resultant eye movements. This could be beneficial if different tests are found to be more relevant to certain activities, or if a shorter test duration is needed. Through developing a suite of tests more in-depth data about the participant could be gained, however a vital responsibility is to ensure that the resultant technology is still suitable for the products intended use. An alternative to including more tasks is to keep the task implementation the same, but to measure and report additional metrics such as head movement, which has been reported to be indicative of cognition (4), and saccade amplitude, which has had debated usefulness but was shown to be promising in this system (3,8,14). An issue with measuring head movement in this setting is that if participants have to keep their head still for 20 minutes this may be unnatural and cause discomfort. A chin rest could be supplied to limit this for example, however this also raises the problem of extra equipment being needed in this setting. Another update to existing functionality would be enabling the user to define how many rounds of each task would be executed, instead of just allowing early termination. This would tackle the issue of if a user needs to perform a quicker test, while still allowing the user to perform the full variety of tasks. Another practical enhancement would be to lock or hide the files produced by the application for analysis to prevent corruption. This could be done by deleting them once the report is generated, or perhaps simply hiding them from being visible to the user.

10 Conclusion

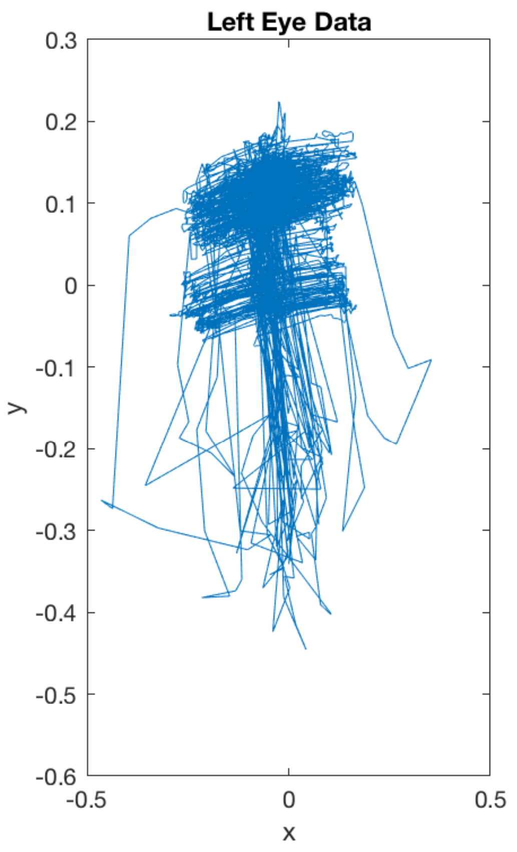
The developed software addresses the problem of providing reliable sideline concussion detection as it has been suitably designed to meet the project aims and specification, which were tailored to tackle some of the limitations of previously developed systems while including formerly unused tasks for VR sideline testing. The product was thoroughly tested, and the results from this show that is has displayed accurate eye-tracking capability, robust saccade detection, and appropriate metric calculation. The output of the system is generated in under a minute and provides a useful report consisting of both statistics and figures that can be used to quickly assess user performance. This provides a tool that can address the problem of concussions going undiagnosed and unreported, whilst quickly facilitating the return to play decision in a practical setting.

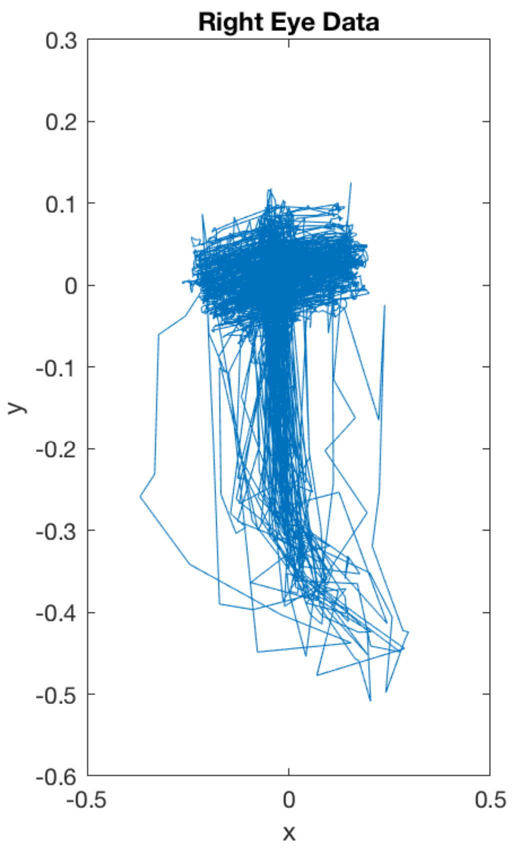
11 List of References

1. Sussman, E.S., Ho, A.L., Pendharkar, A.V. and Ghajar, J., 2016. Clinical evaluation of concussion: the evolving role of oculomotor assessments. *Neurosurgical focus*, *40*(4), p.E7.
2. Webb, B., 2017. Oculomotor Executive Dysfunction During the Early and Later Stages of Sport-Related Concussion Recovery.
3. Ventura, R.E., Balcer, L.J., Galetta, S.L. and Rucker, J.C., 2016. Ocular motor assessment in concussion: Current status and future directions. *Journal of the neurological sciences*, *361*, pp.79-86.
4. Nolin, P., Stipanicic, A., Henry, M., Joyal, C.C. and Allain, P., 2012. Virtual reality as a screening tool for sports concussion in adolescents. *Brain injury*, *26*(13-14), pp.1564-1573.
5. Fischer, J.D. and van den Heever, D.J., 2016, August. Portable video-oculography device for implementation in sideline concussion assessments: a prototype. In *Engineering in Medicine and Biology Society (EMBC), 2016 IEEE 38th Annual International Conference of the* (pp. 4361-4364). IEEE.
6. Okdeh, M.A., Hage-Diab, A., Haj-Ali, A. and Hussein, B., 2017, October. Concussion detection using a commercially available eye tracker. In *Advances in Biomedical Engineering (ICABME), 2017 Fourth International Conference on* (pp. 1-4). IEEE.
7. Parsonage, M., Traumatic brain injury and offending, 2016.
8. Rizzo, J.R., Hudson, T.E., Dai, W., Birkemeier, J., Pasculli, R.M., Selesnick, I., Balcer, L.J., Galetta, S.L. and Rucker, J.C., 2016. Rapid number naming in chronic concussion: eye movements in the King–Devick test. *Annals of clinical and translational neurology*, *3*(10), pp.801-811.
9. King-Devick Technologies, Inc. (2018). *Concussion Screening Test, Reading Solutions | King-Devick Test*. [online] Available at: https://kingdevicktest.com [Last Accessed 19 Jun. 2018].
10. Syncthink.com. (n.d.). *EYE-SYNC – SyncThink*. [online] Available at: https://syncthink.com/eye-sync/ [Accessed 19 Jun. 2018].
11. Concussion Care Management - ImPACT Applications Inc. (n.d.). *Home | Concussion Management | ImPACT Applications Inc*. [online] Available at: https://impacttest.com [Accessed 19 Jun. 2018].
12. Erdal, K., 2012. Neuropsychological testing for sports-related concussion: how athletes can sandbag their baseline testing without detection. *Archives of Clinical Neuropsychology*, *27*(5), pp.473-479.
13. FOVE. 2014. *FOVE 0.* [online] Available at: https://www.getfove.com [Accessed 1 September 2018].
14. Antoniades, C., Ettinger, U., Gaymard, B., Gilchrist, I., Kristjánsson, A., Kennard, C., Leigh, R.J., Noorani, I., Pouget, P., Smyrnis, N. and Tarnowski, A., 2013. An internationally standardised antisaccade protocol. *Vision research*, *84*, pp.1-5.
15. Fooken, J., Yeo, S.H., Pai, D.K. and Spering, M., 2016. Eye movement accuracy determines natural interception strategies. *Journal of vision*, *16*(14), pp.1-1.
16. Munoz, D.P. and Everling, S., 2004. Look away: the anti-saccade task and the voluntary control of eye movement. *Nature Reviews Neuroscience*, *5*(3), p.218.
17. MathWorks. 2017. *MATLAB Report Generator.* [online] Available at: https://uk.mathworks.com/help/rptgen/. [Accessed 1 September 2018].
18. Leigh, R.J. and Zee, D.S., 2015. *The neurology of eye movements*. 5th edn. Oxford University Press, USA.

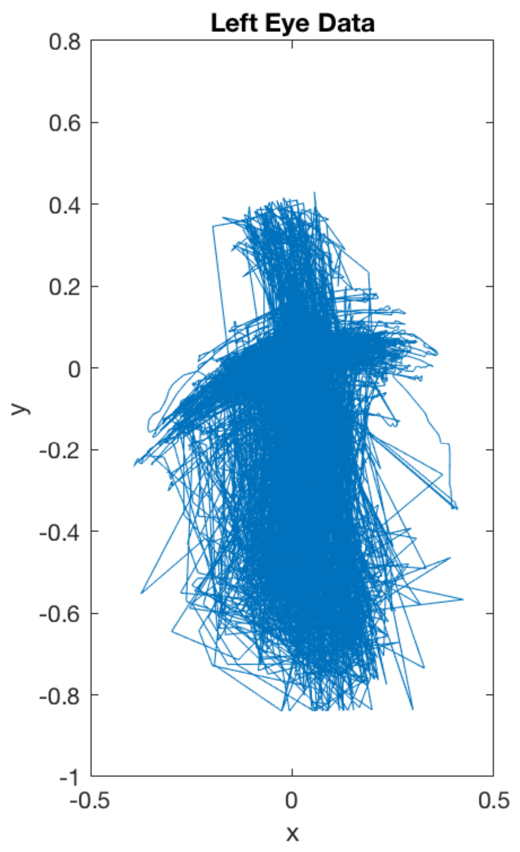
Appendix A

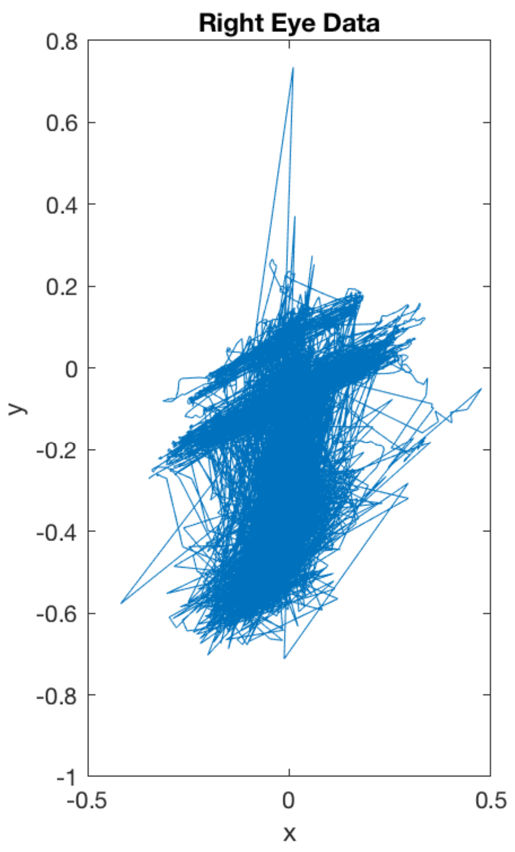
Eye movement paths with little noise:





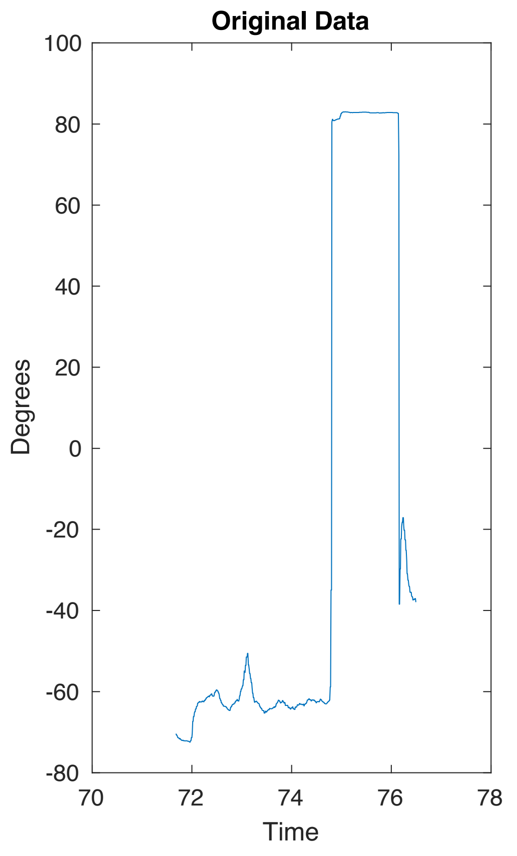
Eye movement paths with heavy noise:

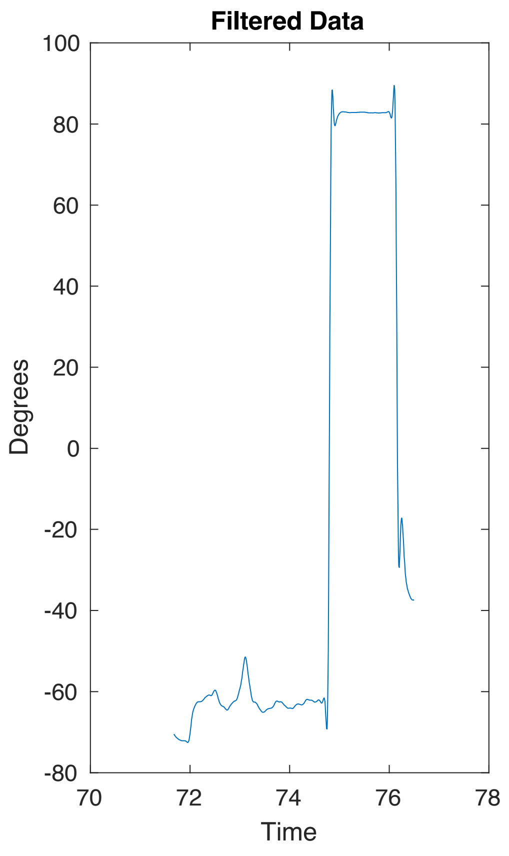




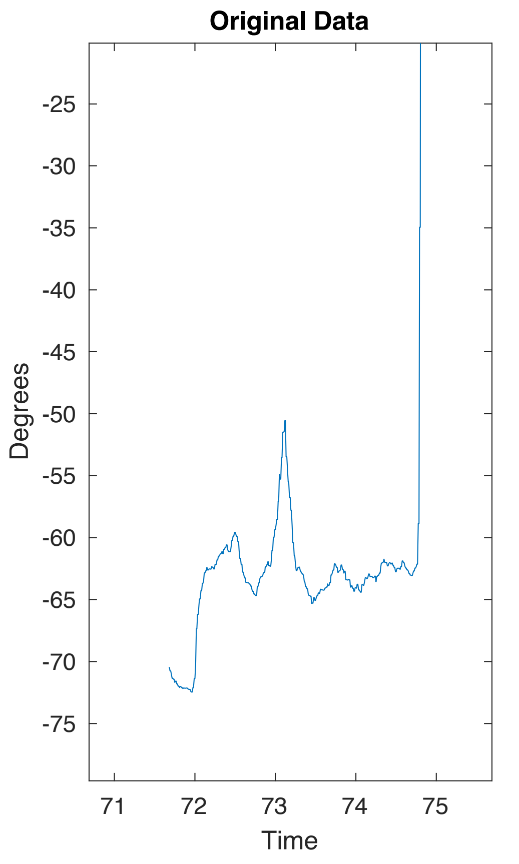
Appendix B

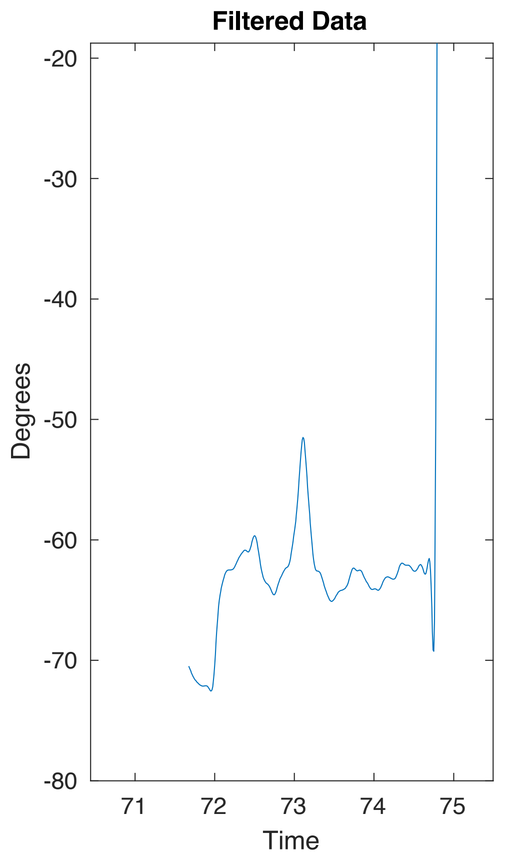
These plots show the difference between the original data for a round (left), and the filtered data (right).





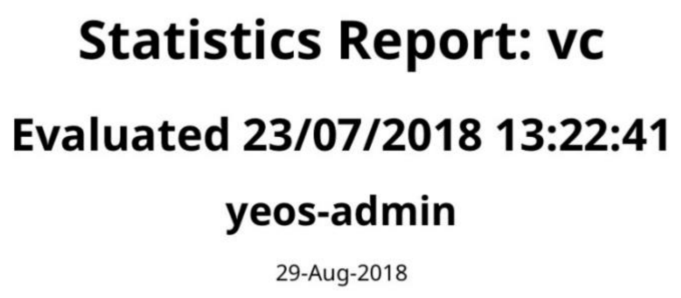
These plots show an enlarged section of the above data, to illustrate the effect of the filter.





Appendix C

A full report generated by the system (12 pages).



Statistics Report: vc

Evaluated 23/07/2018 13:22:41

yeos-admin

29-Aug-2018

Table of Contents

Chapter 1. Eye Movement Paths

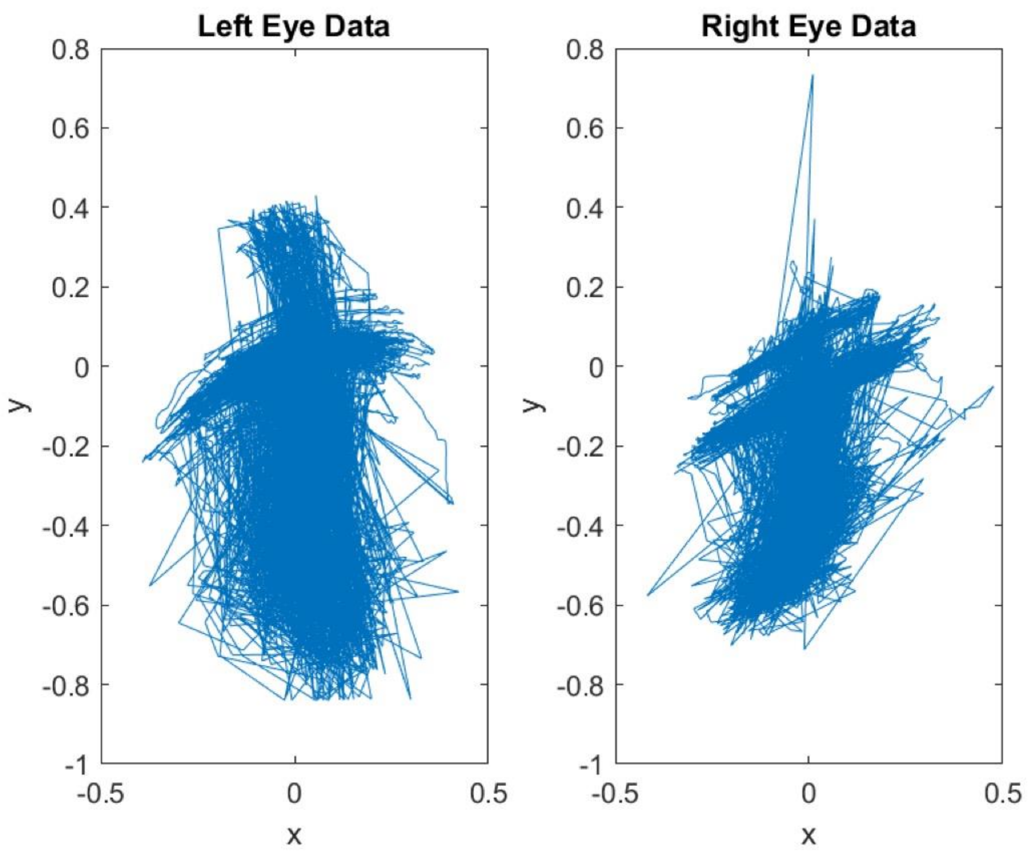


Figure 1.1. Complete Eye Movement Paths

Chapter 2. Latency of Initial Saccade

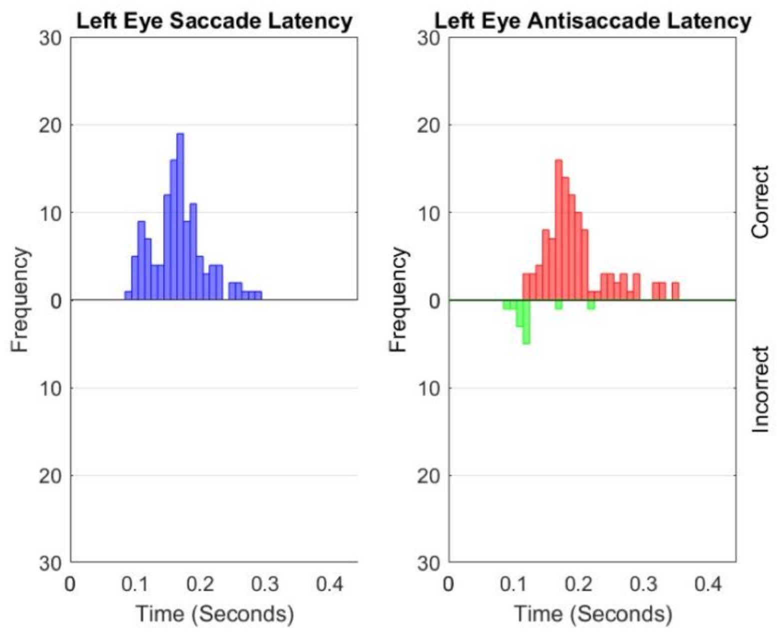


Figure 2.1. 0 non-response(s). 0 mis-recording(s)

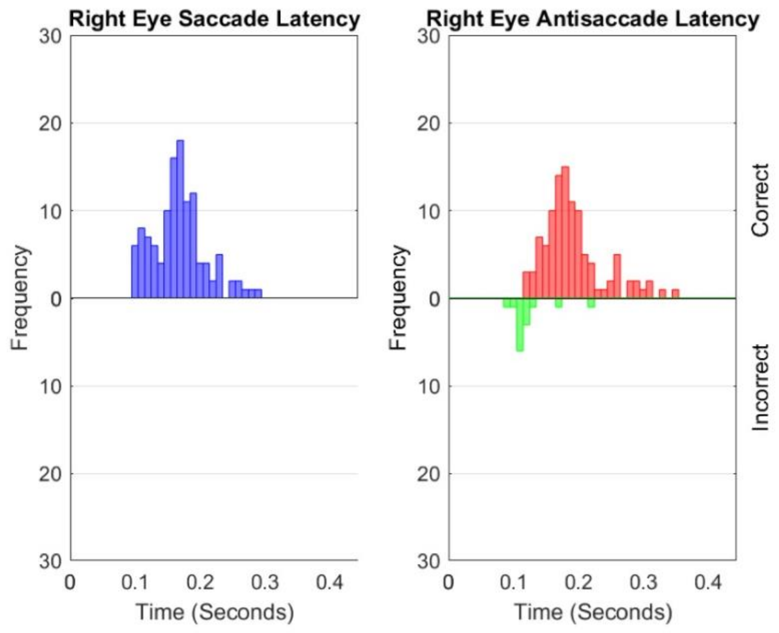


Figure 2.2. 0 non-response(s). 0 mis-recording(s)

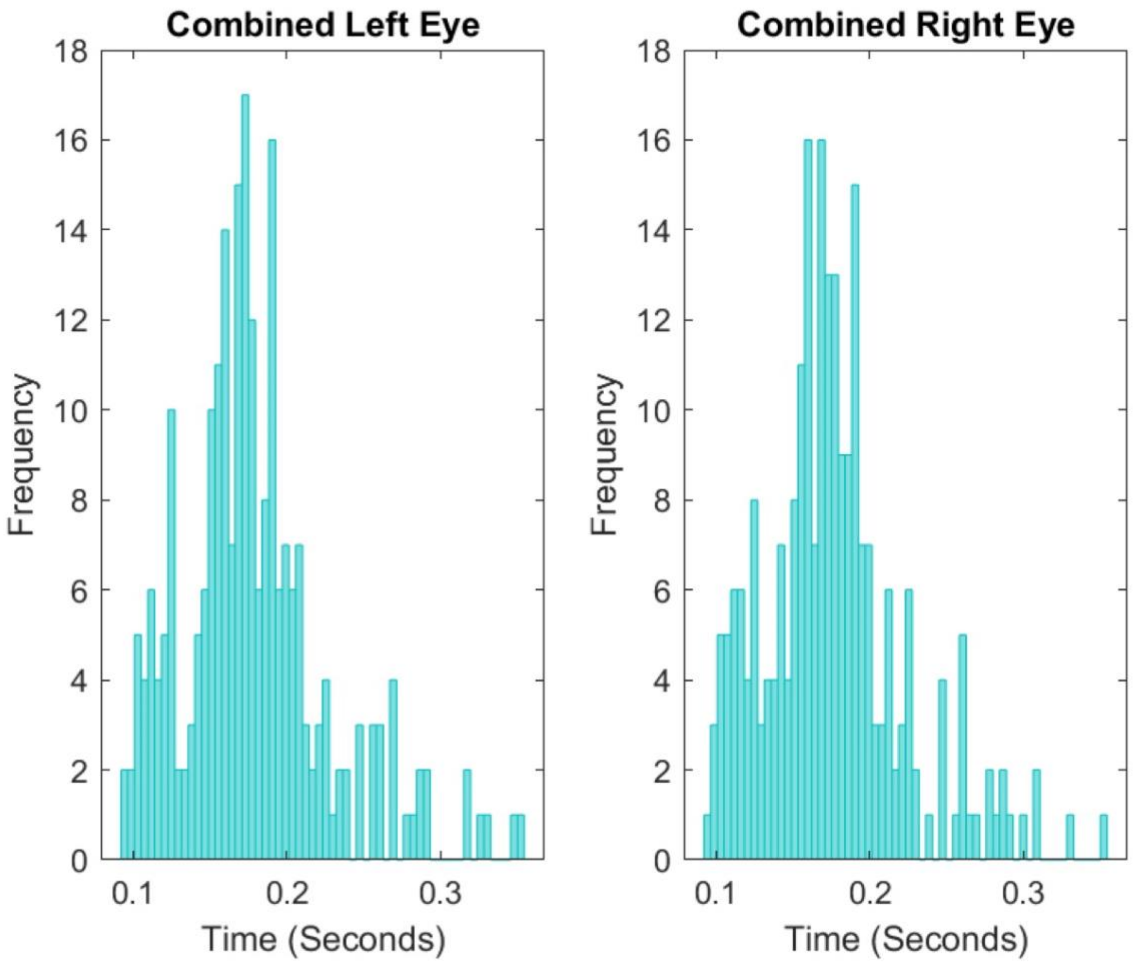


Figure 2.3. Left: 0 non-response(s). 0 mis-recording(s). Right: 0 non-response(s). 0 mis-recording(s)

Results:
454 correct. 26 incorrect. Error rate: 0.054

Chapter 3. Duration of Initial Saccade

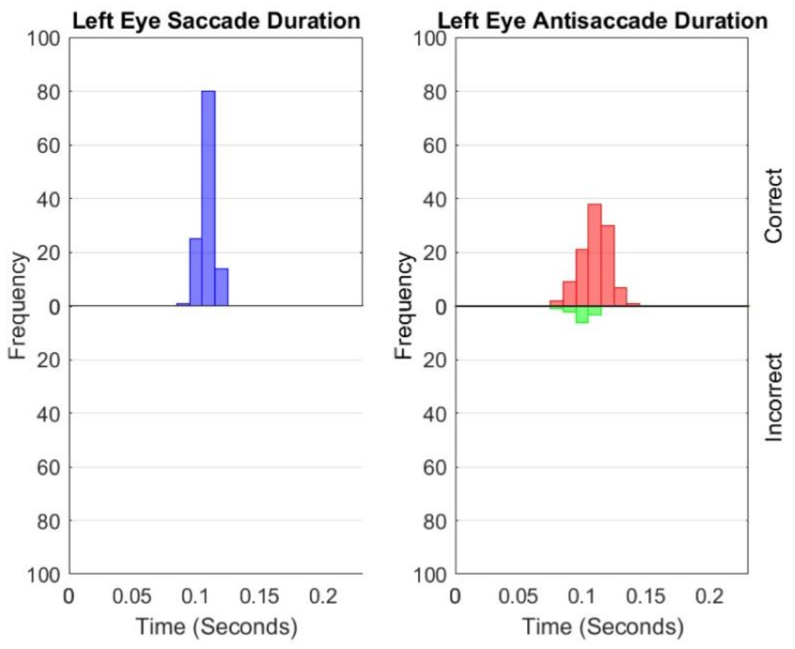


Figure 3.1. 0 non-response(s). 0 mis-recording(s)

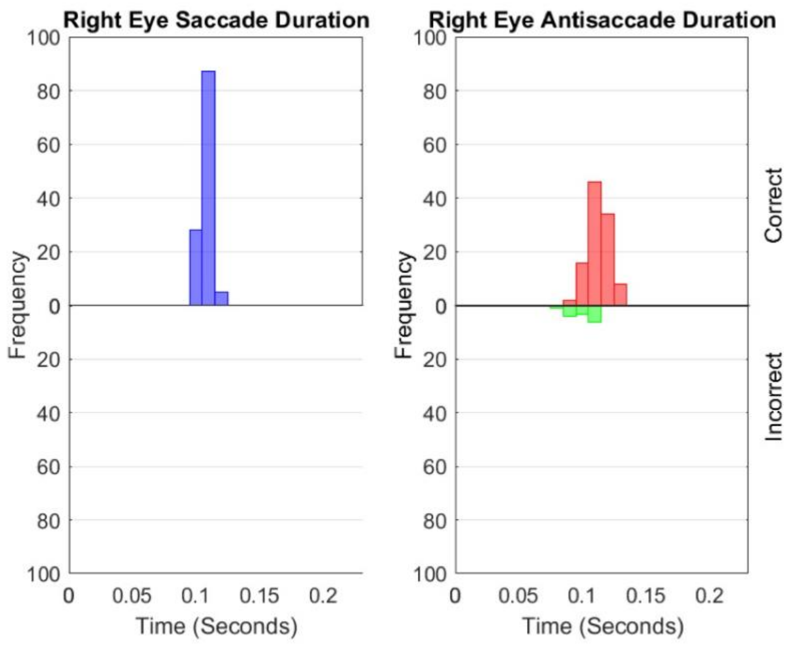


Figure 3.2. 0 non-response(s). 0 mis-recording(s)

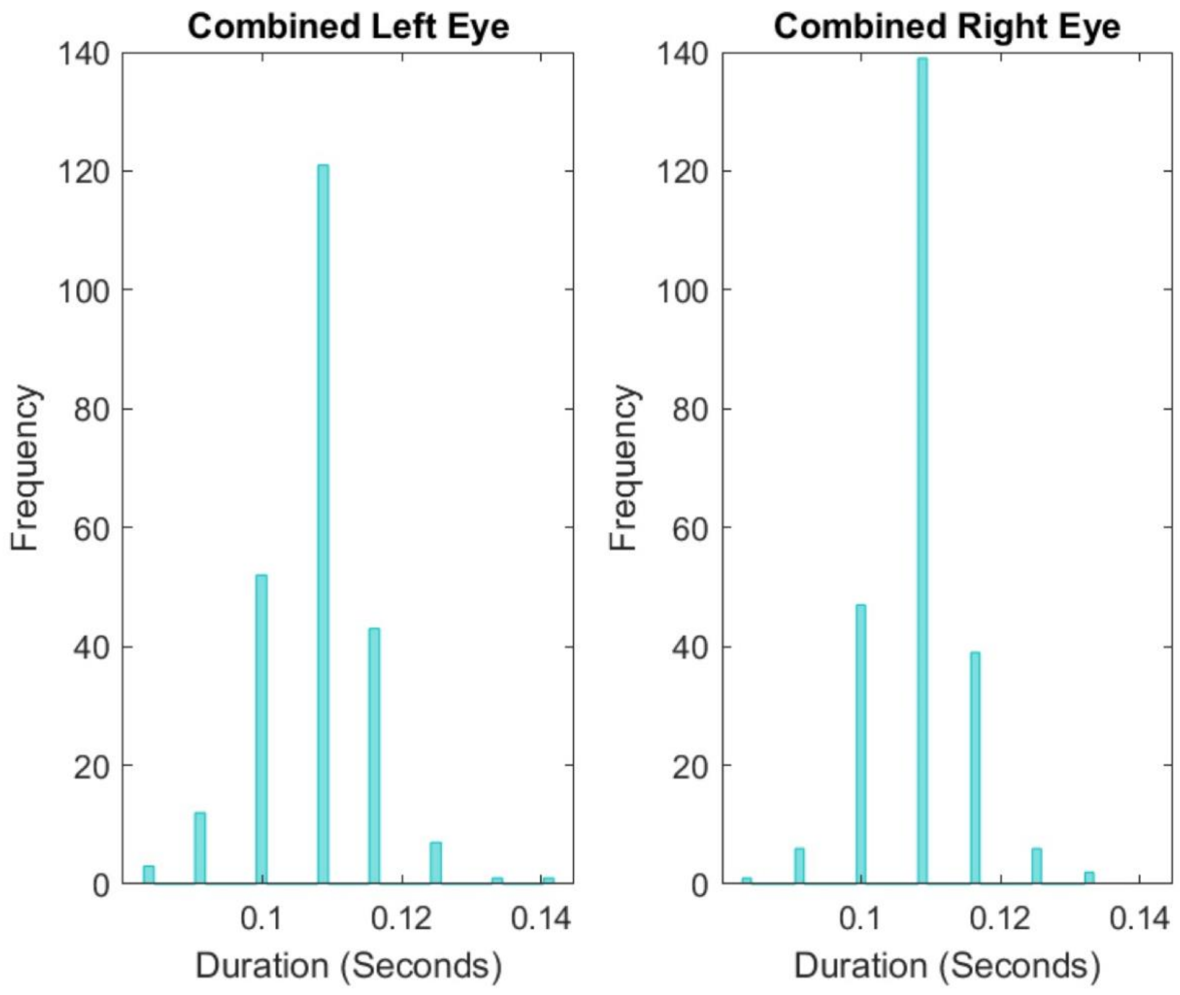


Figure 3.3. Left: 0 non-response(s). 0 mis-recording(s). Right: 0 non-response(s). 0 mis-recording(s)

Chapter 4. Speed of Initial Saccade

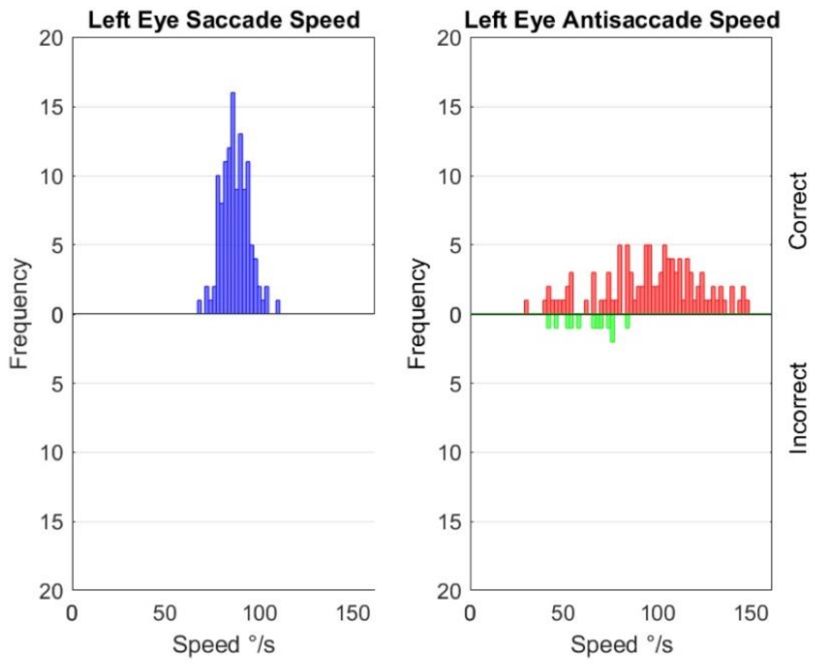


Figure 4.1. 0 non-response(s). 0 mis-recording(s)

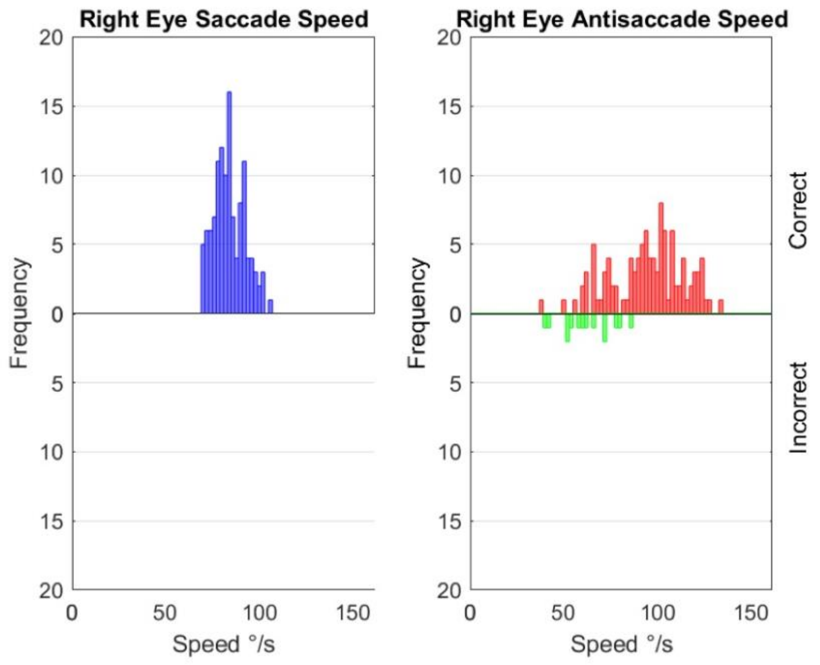


Figure 4.2. 0 non-response(s). 0 mis-recording(s)

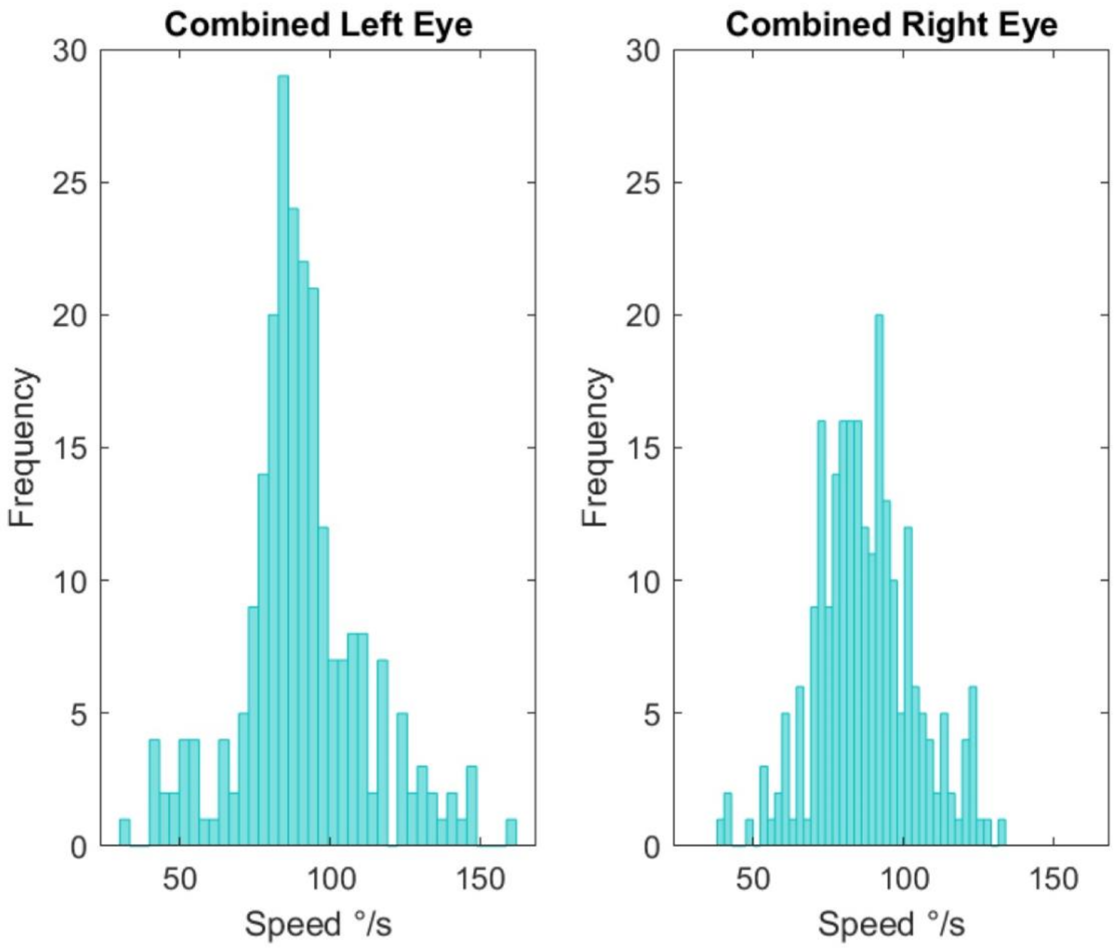


Figure 4.3. Left: 0 non-response(s). 0 mis-recording(s). Right: 0 non-response(s). 0 mis-recording(s)

Chapter 5. Corrective Latency

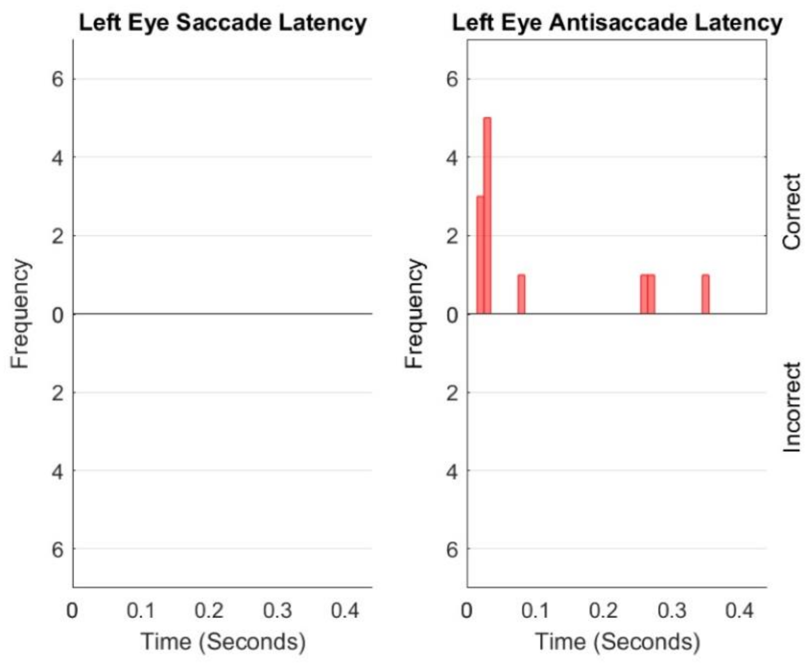


Figure 5.1. 0 not corrected. 0 mis-recording(s)

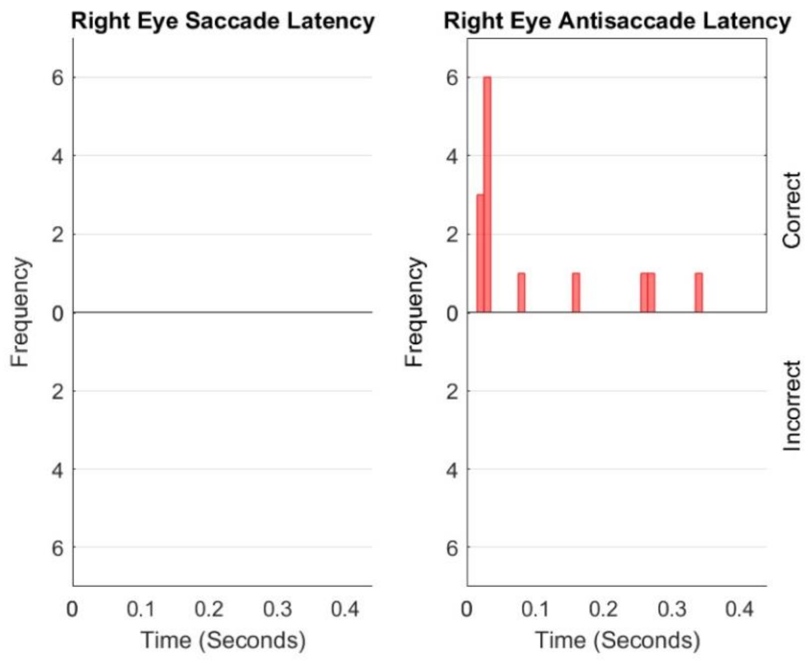


Figure 5.2. 0 not corrected. 0 mis-recording(s)

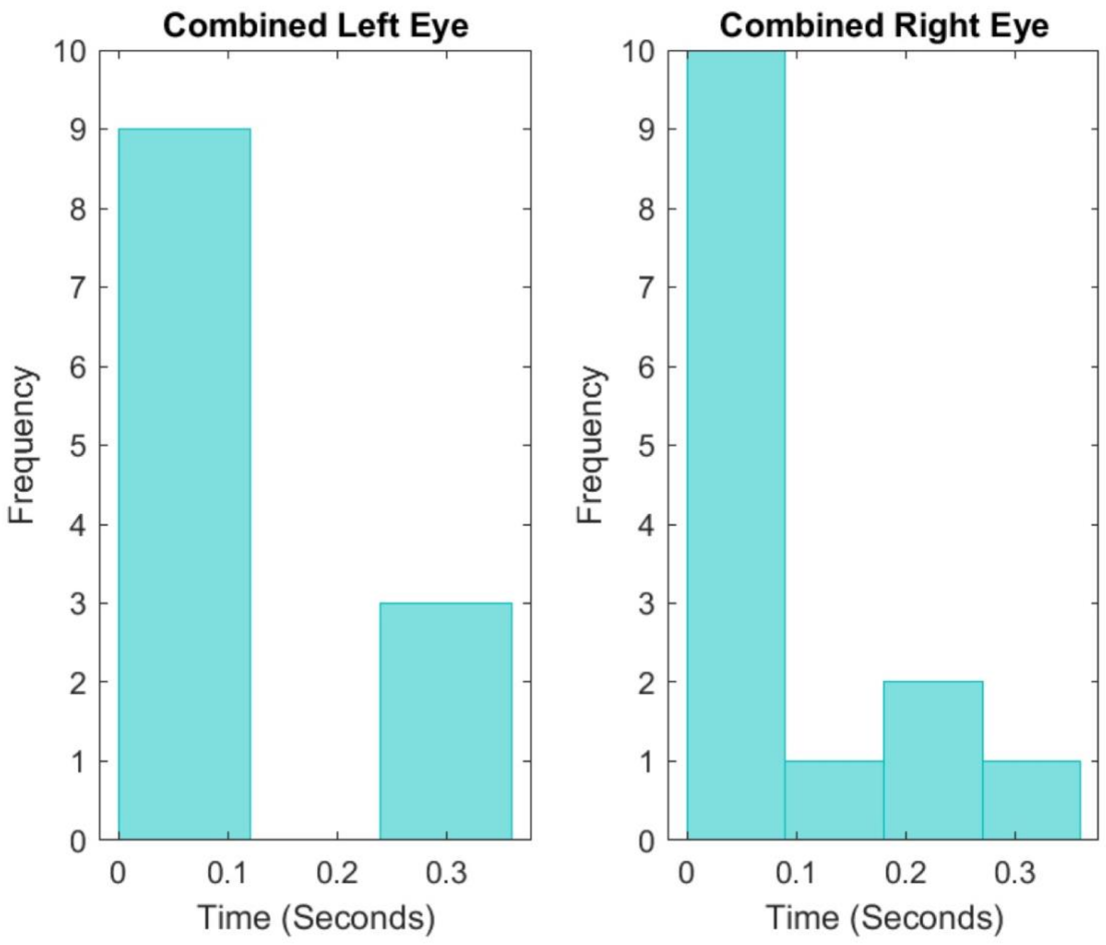


Figure 5.3. Left: 0 not corrected. 0 mis-recording(s) Right: 0 not corrected. 0 mis-recording(s)

Chapter 6. Recorded Notes

Suspected concussion due to tackle.

Appendix D

Instructions for participants

Hello, thank you for taking part in this study. You will be performing two different exercises; the Antisaccade Exercise and the ImPACT Test, which both consist of different tasks. You are able to withdraw from the study whenever you want, even after completion.

ImPACT Test

For the ImPACT Test you will be sitting at a computer and will complete several different tasks, this will take about 25 minutes. You will be given further instructions beforehand and will have the opportunity to do some practice runs.

Antisaccade Exercise

For the Antisaccade Exercise you will be wearing a VR headset and performing sets of two types of task, this will take about 20 minutes. You have the option to perform some practice runs beforehand.

Both of the tasks will have the same format; you will see a small circle displayed in the middle of the screen and it will briefly disappear and then reappear on the right or left. This will happen several times and the tasks require you to look from the centre to the left or right as quickly as possible, depending on which side the circle appears. When looking at the target, please just move your eyes and not your head.

The first of the two tasks is called the prosaccade task. In this you will need to look from the centre circle to its new position. For example if the circle appears on the right, you should look to the right directly at it as quickly and accurately as possible, and if the circle appears on the left you should look to the left directly at it as quickly and accurately as possible.

The second task is called the antisaccade task. In this you will need to look from the centre circle to the mirror of the position it’s appeared at. For example, if the circle appears on the right you should look at where the circle would have appeared on the left. If the circle appears on the left, you should look at where the circle would have appeared on the right. Please aim to be as quick and accurate as possible.

If you look in the wrong direction in any of the tasks, simply look in the correct direction as quickly and accurately as possible.

The tasks will be run in the following order: prosaccade, antisaccade, antisaccade, antisaccade, prosaccade. You don’t need to remember this order as you will be prompted before each round begins. You will be able to have a break after each round and you can choose how long to rest for. You are encouraged to look around and blink during this time.

If possible, please avoid blinking during the time between the centre circle disappearing and looking at the target circle as this can interfere with results. This time period will be roughly half a second. You are encouraged to blink a lot when waiting the centre circle to disappear and after you have looked in the correct direction.

Appendix E

The following information details the conditions the tests were held under during testing of both The ImPACT Test and the developed system.

Participants were tested individually in the VR Lab located in The School of Sport and Exercise Sciences, University of Birmingham, with only an administrator present. Participants were given instructions, see ‘Appendix D’, and then asked to sign a consent form before taking part in the study. During this time, participants were encouraged to ask questions to clarify anything they were unsure of.

Half of the participants performed The ImPACT test first, whereas the other half tested the developed system first. Participants were given a 5-minute break after they tested the first system, and before they moved on to test the second system. After both systems were tested, participants were again encouraged to ask any questions and give their feedback on their experiences and opinions of the tests. Overall, testing lasted about an hour for each participant.

Appendix F

Git repository: https://git-teaching.cs.bham.ac.uk/mod-msc-proj-2017/vlc784

How to run the application code:
The application is able run without the FOVE headset being connected however this will not record any data.
Windows: Download and unzip 'Mono', 'SaccadeTest_Data' and 'UnityPlayer' to the same folder. In order to run the test application, download and run 'SaccadeTest.exe' to the same folder and run this file. If you are using a FOVE headset and not testing in the VR lab you will need to change the string 'path' in 'Controller.cs' from 'saccadeAssets.unitypackage' and deploy the application again from Unity.

How to run the analysis code:
You will need to download all MATLAB files supplied in the repository, aside from 'PCAAnalysis', 'plotDurFig.m', 'plotFig.m' and 'VariableOutput.m', and have MATLAB version 2017b or later. Within MATLAB, navigate to the data folder created by 'SaccadeTest.exe'. If you have not used the headset and generated data, please download and unzip the folder '23072018132241vc' or '01092018144008DC'. Run the script 'ReportGenerator.m' to analyse the data generate a report.

File	**Description**
01092018144008.zip	Test data for report generation
23072018132241.zip	Test data for report generation
Controller.cs	Script for SaccadeTest.exe
Mono.zip	Supporting file for SaccadeTest.exe
PCAAnalysis	Analyses the generated data
Received_Code.zip	Unedited code received from Dr Sang-Hoon Yeo
ReportGenerator.m	Generates the user evaluation report
SaccadeTest.exe	Task execution application
SaccadeTest_Data.zip	Supporting file for SaccadeTest.exe
UnityPlayer.zip	Supporting file for SaccadeTest.exe
VariableOutput.m	Generates the user evaluation report and saves metrics to file
addToMatrix.m	Adds the supplied element to the supplied matrix
detectSaccade.m	Detects if a saccade was made
draw_mirror_hist_multi.m	Plots the dual histogram
getDecEndInd.m	Calculates the end of a left saccade
getDecInd.m	Calculates the start of a right saccade
getIncEndInd.m	Calculates the start of a right saccade
getIncInd.m	Calculates the start of a left saccade
plotDurFig.m	Plots the round data with the detected start and end of the saccade
plotFig.m	Plots the round data with the detected start of the saccade
plotTogether.m	Calculates data sub-groups and plot settings
saccadeAssets.unitypackage	Unity package for SaccadeTest.exe
seperateMetric.m	Separates the results into sub-groups

Appendix G

Code Sources:
'draw_mirro_hist_multi_example_script.m' and 'draw_mirror_hist_multi.m' were received from my supervisor Dr Sang-Hoon Yeo, and are available in a zip folder within the repository titled "Received code". Parts of 'draw_mirro_hist_multi_example_script.m' were adapted into 'plotTogether.m' to define the data subgroups and visually modify the plots. 'draw_mirror_hist_multi.m' was altered to change plot visuals and dimensions to suit the application.
'ReportGenerator.m' and 'VariableOutput.m' use a lot of resources from MATLABs report generator documentation, available at: https://uk.mathworks.com/help/rptgen/ (17), and calls the files 'plotTogether.m' and 'draw_mirror_hist_multi.m' mentioned above.
The code in 'detectSaccade.m' was written based on the saccade velocity detection concept outlined by J. Fooken et al (2016) (15).
'PCAAnalysis.m' uses the video 'How to Label a Series of Points on a Plot in MATLAB', available at https://uk.mathworks.com/videos/how-to-label-a-series-of-points-on-a-plot-in-matlab-97199.html to label the points some plots.
In 'Controller.cs', eye movement data is written to file, the code is my original work but inspiration was drawn from the following two Unity posts: https://support.getfove.com/hc/en-us/community/posts/115001063588-Eye-coordinate-data and https://support.getfove.com/hc/en-us/community/posts/115007105293-Lian

All other files and code are my original work.