Eye Tracking as an Additional Input Method in Video Games

Using Player Gaze to Improve Player Immersion and Performance

Niclas Ejdemyr

Faculty of Computing
Blekinge Institute of Technology
SE-371 79 Karlskrona, Sweden
This thesis is submitted to the Faculty of Computing at Blekinge Institute of Technology in partial fulfillment of the requirements for the degree of Bachelor of Science in Digital Game Development. The thesis is equivalent to 10 weeks of full-time studies.

Contact Information:
Author(s):
Niclas Ejdemo
E-mail: niej11@student.bth.se

University advisor:
MSc. Diego Navarro
Dept. Creative Technologies

Faculty of Computing
Blekinge Institute of Technology
SE–371 79 Karlskrona, Sweden
Internet :www.bth.se
Phone :+46 455 38 50 00
Fax :+46 455 38 50 57
Abstract

Context. Gaze based interaction in video games is still a developing field, and is mostly used as an off-line evaluation tool or a replacement for traditional input methods. This thesis will look closer at the prospect of using eye tracking as an additional input to be used alongside the traditional methods of input to improve the immersion and performance of the player.

Objectives. To implement a gaze based interaction method into a first person adventure in a way to improve player performance and immersion.

Methods. Using the Tobii REX eye tracker, 18 volunteers participated in an experiment. They played two versions of a game in a controlled environment. The versions had the same mechanics and game elements but only one of them had eye tracking implemented. After the experiment the participants answered nine questions about which prototype they preferred.

Results. All participants' scores were in all cases but one, lower when using the eye tracking input method, compared to the traditional one. The time it took for the participants to complete the game was longer for everybody. 16 out of 18 players also felt more immersed in the game while using eye tracking compared to playing with the traditional input method.

Conclusions. The results from the experiments provided evidence that the interaction method designed for this thesis did not improve player performance. The results also showed that the interaction method did improve immersion for most players.

Keywords: Eye tracking, gaze aware, immersion, player performance.
# Contents

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Abstract</td>
<td>i</td>
</tr>
<tr>
<td>1 Introduction</td>
<td>1</td>
</tr>
<tr>
<td>1.1 Background</td>
<td>1</td>
</tr>
<tr>
<td>1.2 Objective and Gap</td>
<td>2</td>
</tr>
<tr>
<td>1.3 Research Question</td>
<td>2</td>
</tr>
<tr>
<td>1.4 Thesis Layout</td>
<td>2</td>
</tr>
<tr>
<td>2 Related Work</td>
<td>3</td>
</tr>
<tr>
<td>3 Implementation</td>
<td>5</td>
</tr>
<tr>
<td>3.1 Game Design</td>
<td>5</td>
</tr>
<tr>
<td>3.1.1 Objective</td>
<td>5</td>
</tr>
<tr>
<td>3.1.2 Mechanics</td>
<td>5</td>
</tr>
<tr>
<td>3.1.3 Game Elements</td>
<td>6</td>
</tr>
<tr>
<td>3.2 Development</td>
<td>9</td>
</tr>
<tr>
<td>3.2.1 Interaction Method</td>
<td>9</td>
</tr>
<tr>
<td>3.2.2 Game Objects</td>
<td>9</td>
</tr>
<tr>
<td>3.2.3 EyeX Engine</td>
<td>12</td>
</tr>
<tr>
<td>4 Method</td>
<td>13</td>
</tr>
<tr>
<td>4.1 Experimental Setup</td>
<td>13</td>
</tr>
<tr>
<td>4.2 Testing</td>
<td>14</td>
</tr>
<tr>
<td>5 Results</td>
<td>16</td>
</tr>
<tr>
<td>6 Analysis</td>
<td>19</td>
</tr>
<tr>
<td>7 Conclusions and Future Work</td>
<td>21</td>
</tr>
<tr>
<td>References</td>
<td>22</td>
</tr>
</tbody>
</table>
List of Figures

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.1</td>
<td>The key model</td>
<td>6</td>
</tr>
<tr>
<td>3.2</td>
<td>Overview of the map in Unity 5.0.1</td>
<td>7</td>
</tr>
<tr>
<td>3.3</td>
<td>Floating enemy model</td>
<td>8</td>
</tr>
<tr>
<td>3.4</td>
<td>Fixed enemy model</td>
<td>8</td>
</tr>
<tr>
<td>3.5</td>
<td>Example of the user interface. Middle square is the reticule, bottom center square tells the player how many keys are left, left bottom square shows the time and score.</td>
<td>9</td>
</tr>
<tr>
<td>4.1</td>
<td>Equipment Set Up</td>
<td>13</td>
</tr>
<tr>
<td>5.2</td>
<td>Score for both prototype versions per participant</td>
<td>17</td>
</tr>
</tbody>
</table>
List of Tables

5.1 The results from the questionnaire and results chi-square test per question .......................... 18
Chapter 1

Introduction

Chapter 1 will give some insight in what Eye tracking technology is, how it works and what it is used for in video games.

1.1 Background

Eye tracking is a technology that monitors where the user’s gaze is located. This enables the user to interact with a computer using their eyes. The user’s gaze can be measured in two different ways. The first method is called corneal reflection eye tracking. It works by projecting infra-red light at the user. A set of cameras captures the reflections from the user’s cornea and pupil. This is then used to calculate the gaze of the user. The second method is called electro-oculography, which works by placing electrodes around the eye of the user to measure the electrical charge produced by the eye movements to calculate the gaze position of the user. [5]

Eye tracking have two major functions. The first is to give the user an alternative input method to the keyboard and mouse interfaces. The second function is as an offline evaluation tool, to study where the user’s gaze is located in time and space. This information can be used for market research, usability testing, sports research, etc. [1] [12].

Eye tracking is also used in different kind of games like Dota 2 and Starcraft as an analysis tool. The eye tracker records the user’s gaze and fixation points during the game, to be reviewed later, to help the player to improve performance in the future. Some new games have come to adopt eye tracking as an additional input method alongside the traditional input methods. To name a few, Assassins Creed: Rogue uses eye tracking as a way to control the camera if the player does not use another method to override the eye tracker control. Son of Nor by Still Alive Studios uses the eye tracking for more complex interactions such as terraforming and telekinesis. The player’s gaze point will be the position where

[2] sonofnor.com
the player character’s powers will origin from, instead of the reticule in the center of the screen. Both games are still playable without eye tracking.

This thesis will look closer at the eye tracker as an additional input method implemented in a first person adventure game using the Tobii REX eye tracker.

1.2 Objective and Gap

As presented before, eye tracking in games is used as a analysis tool or as an alternative for other input methods, not as an addition to the standard input methods. This thesis proposes to combine eye tracking technology with a traditional input method, keyboard and mouse, with the intent of improving player’s immersion and performance.

1.3 Research Question

The question this thesis will be address is:

1. How can a gaze based interaction method be implemented into a first person game to improve immersion and player performance, as an additional input method along side the traditional input methods?

**Immersion** is the perception of being present in a non-physical world. Immersion can be divided into 4 groups, sensory-motoric immersion, cognitive immersion, emotional immersion and spatial immersion. The relevant group in this thesis will be sensory-motoric immersion, which is experienced when the player performs an tactile, visual or other sensory action that gives a reaction, e.g in a game.[2]

**Player performance** measure how well the player complete the objective of the game. Player performance will be measured in both the time it takes the player to complete the objective and his/her final score.

1.4 Thesis Layout

This thesis contains 6 chapters. Chapter 2 will contain results and information from other studies related to the topic. Chapter 3 details how the experiment was set up and the method used during testing. How the game prototypes were designed and implemented is detailed in chapter 4. Chapter 5 contains the results from the experiment. Chapter 6 will discuss the result and ideas for future work.
Chapter 2

Related Work

In this chapter information about other studies and articles involving eye tracking and eye tracking in games will be presented.

One of the first to explore the field of human-computer interaction using gaze based technology was Jacob. He showed how eye tracking could be used as an effective way to navigate user interfaces. His report also showed that replacing the mouse with eye tracker is an overly simplified way to implement the eye tracker due to the eye's natural behaviour of rapid movement. [7]

Smith and Graham presented a study that shows the use of an eye tracker in three different games and found that the games were more enjoyable to a majority of the participants. The most similar test by Smith and Graham to the experiment conducted in this study is their Quake 2 test. It used keyboard input to move the player and mouse input to tilt the camera on the Y-axis while the player's gaze controlled the camera movement on the X-axis. The study also showed that the games takes longer and are harder to control with the eye tracker compared to the keyboard and mouse input method. At the same time the participants in this study felt more immersed in the game while playing. [11]

In an study by Isokoski et al. it was presented that the eye tracking technology is not accurate enough to be used as an alternative to the traditional input method of a keyboard and mouse in most games. They also mention the Midas touch. The Midas touch is used to describe that if the eye tracker has control of interacting with objects, it will always do this if the user's gaze is located over the object. This gives unwanted interactions to the player at all times. To address this issue, a gaze dwell timer was implemented, but it was not useful in high paced games where the user was constantly moving their gaze as it reduced performance and accuracy. [6]

The use of the eye tracker as a secondary input method has been done before but requires further research. Few have attempted to implement eye tracking for other purposes than control. One contribution to this area is Chan et al. who constructed a game designed around the eye tracker instead of implementing the
eye tracker as an replacement for other input methods in an existing game. The game consists of the player looking at elements in order to receive a video call, the more focus the player have on the game elements the better the call will be and if the player loses focus on the game elements the quality of the call will be lower. [3]

Navarro explores the eye tracking as an assistive tool, to help the player to complete a game faster. The eye tracker collected the player’s gaze position at all times, which was then used to determine if the player had missed an objective. If the player had an objective inside a firing range and hadn’t fixated his/her gaze on the objective it grew in size, when the player saw it, it would shrink down to its normal size again. The result from this study showed that all participants had a shorter time while using the eye tracker compared to playing the game without the assistance of the eye tracker. [3]

In an article by Odonovan et al. [10] based on the master thesis by Odonovan [9] a gaze based and voice controlled input method is compared to keyboard and mouse input. The test consisted of a first person game in which the player would navigate a maze-like structure while dispatching enemies and collection gold coins. The player’s performance and enjoyment are measured by collecting data from the game such as time, speed, number of coins collected, etc. In order to measure the player’s enjoyment a questionnaire where the player would rate their experience of numerous aspects of the game on a 1 to 7 scale. The results from this study showed that players would perform worse and that the players thought the game was harder to control but still think that the game was more immersive.
3.1 Game Design

The game is a prototype of a first person adventure game. The game takes place in a maze-like structure with a number of doors that leads to numerous corridors and rooms. The game prototype will be separated into two versions, one that uses the traditional input methods (TI), the other uses an eye tracker implementation (ETI). Other than this the versions are identical.

3.1.1 Objective

The objective of the game is to collect all keys in the map while avoiding the enemies. Each key is worth 100 points and there are six of them placed around the map. Picking up the last key will trigger the "Game Over" message and the final score will be displayed on the screen. The player is also tasked with avoiding the enemies as they will lower the player’s score. The maximum amount of points that are available is 600.

3.1.2 Mechanics

The main objective of the game is to collect all keys on the map as fast as possible. A secondary objective is to avoid the enemies to keep the score high.

- **Movement**. The player is able to move along the horizontal plane of the game world, forward, backwards, left and right using the W, A, S and D keys on the keyboard. The player is also able to look around in the three dimensional room using the computer mouse.
Chapter 3. Implementation

- **Gaze awareness.** The game is tracking the gaze of the player while playing the ETI version. The gaze point is used to interact with the two enemy types in the game. When the TI version is played, a reticule in the center of the screen is used as a substitute for the player's gaze point.

### 3.1.3 Game Elements

There are a number of different elements in the game that the player can interact with, or use to make his/her way to the keys. There is no sound in the game due to time restraints. Instead visual cues were exaggerated communicate if the player was in danger or not, such as the enemies changed color from bright green to bright red when the player was in danger.

![Figure 3.1: The key model](image)

#### 3D Elements

All the 3D models are made by the author using a student license of Autodesk Maya 2015. All of the models except for the torch, are untextured. The particle effects used are made by G.E.TeamDev and is a free download from Unity's asset store[1].

- **Floating enemy.** The floating enemy (Figure 3.3) has a radius that if the player enters, the enemy will start to chase the player. To stop the enemy the player have to look at it, either with the reticule in the center of the screen or using the eye tracking, depending on what version of the game is being played. If the enemy makes contact with the player some points will be lost.

• **Fixed enemy.** The fixed enemy (Figure 3.4) is attached to a wall in the map and has a collision box in front of it, which functions as the line of sight for the enemy. If the player are in the line of sight and is looking at the enemy model some points will be lost, the amount depending on how long the player is maintaining eye contact.

• **Obstacles.** There are 4 different type of obstacles in the game. Neither the player nor the enemy can pass through the obstacles, therefore they can be used as an aid to escape the enemy if the obstacle is between the player and the enemy.
Chapter 3. Implementation

User Interface elements

The user interface is simple, it displays the current score, time and the numbers of keys the player have to pick up to complete the level. When the game is over the user interface displays a game over text to tell the player that they are finished. The user interface uses the default font for Unity, Arial. The red boxes in Figure 3.5 below highlight the following:

- Bottom left: Time and Score.
- Bottom center: Number of keys left in the map.
- Center: The reticule that is used as the player’s aim in the TI version.
Figure 3.5: Example of the user interface. Middle square is the reticule, bottom center square tells the player how many keys are left, left bottom square shows the time and score.

3.2 Development

The development of the game prototypes took place over a time period of about two weeks. Unity 5.0.1 Personal Edition was used for this as it is compatible with the EyeX engine used for the eye tracker.

3.2.1 Interaction Method

The main interaction method that the player would use to avoid the enemies needed to be simple and easy to convert between the 2 different prototypes. To avoid the Midas touch problem, the method should not be able to activate objects in the game world like buttons and user interface elements. The method that was chosen to be the main interaction method to interact with the enemies was just to either look at them or not look at them.

3.2.2 Game Objects

The Unity GameObject class functions as a basic object in the game world that components and scripts can be attached to it to create more complex objects. Depending on what components are attached to the GameObject, it can become several different game elements such as lights, enemies, doors, keys, etc.
Chapter 3. Implementation

Player

- **Components**: capsule collider, box collider, player controller script, player controller with eye tracking script.

- **Function**: The player object is the object that holds all the movement for the player. The colliders allow for the other objects in the game to interact with the player. The scripts hold the code for the movement and for the aim. The aim functions returns a number between zero and two, which tells the other objects that uses this function if the player is looking at an enemy or not.

The two scripts functions in mostly the same way the only difference is that the eye tracker script gets a variable from the eye tracker controller that is the player’s gaze, while the other script uses the center of the screen.

Floating enemy

- **Components**: enemy mesh, box collider, navigation mesh agent, enemy movement script, enemy movement ETI script, default shader.

- **Function**: The floating enemy object represents all the moving enemies in the game. Seen in Figure 3.3. The collider functions as a target for the player’s aim. Without the collider, Unity can’t return the game object that the player is looking at. It also makes it so that the enemy pushes the player instead of floating to the precise location of the player and clips through the camera. All enemies have a navigation mesh agent that allow them to move around the map without getting stuck in walls or obstacles.

The enemy movement script handles the navigation mesh agent, it checks if the player are inside of a specified radius, if this is true the navigation mesh agent will be activated and the target position will be the player’s position. To check if the enemy is within range for a collision, another radius, that is just outside the enemy mesh will check if the player is inside of the inner radius and if so will count as a collision.

To check if the player is looking at the enemy a public variable from the player controller script is checked, if the player is looking at the enemy the navigation mesh agent will be temporarily disabled. To clearly show to the player if the enemy is active or not its color is connected to the navigation mesh agent. If it is active the enemy turns red, if it is not the enemy turns green.
The ETI enemy work in the same way except for what script from the player object it checks for the aim of the player.

**Fixed enemy**

- **Components**: box trigger collider, eye controller script, eye controller ETI script, eye mesh, box collider, change color script, default shader.

- **Function**: The fixed enemies (Figure 3.4) are located at about eye height on the walls of the game. The box trigger collider works as a line of sight for the enemy if the player enters the box the enemy sees the player, the opposite is true if the player leaves the box. When the player is within the line of sight and the aim of the player is on the enemy some points will be lost. To signal to the player that points are being subtracted the enemy will turn red with help of the color change script.

Like the floating enemy, the only difference between the ETI version and the traditional version is the script that is checked for the player’s aim.

**Door**

- **Components**: door mesh, box collider, navigation mesh obstacle, door memory script, default shader.

- **Function**: The doors function as dividers between the hallways and rooms. As the enemy can’t open the doors, doors can also be used to lure the enemy and then trap them in a room or hallway.

The doors check if the player is within range and if the player are the door will open and stay open for a little while and then they will begin to close.

**Obstacles & Key**

- **Components**: mesh depending on obstacle or key(Figure 3.1), box collider

- **Function**: The obstacles colliders prevents the player and enemies from passing though them.

  The keys have trigger colliders instead and will delete themselves and award points to the player if he/she collides with them, to simulate that the player picked up the key.
Chapter 3. Implementation

Game controller

- **Components**: game controller script, parent to gui text objects.
- **Function**: The game controller keeps the GUI updated and adds score when keys are picked up and decreases when the enemies take it away.

Eye tracker controller

- **Components**: eye tracker script.
- **Function**: The eye tracker script sets up the eye tracker host this is required for Unity to create an instance of the eye tracker to get the data that the eye tracker is collecting. The script also returns the gaze point of the user so that it can be used in the player controller script for the aim function.

3.2.3 EyeX Engine

The EyeX Engine is the software that was used to get the data from the Tobii REX, together with a framework for Unity, which contains C# scripts that gives functionally from the eye tracker such as eye-gaze point, fixation point and fixation time.

This chapter will provide further information about the experiment set up in section 3.1 and how the testing was conducted in section 3.2.

4.1 Experimental Setup

An experiment was done with a set of game prototypes, under a controlled environment. The test device was an Acer Aspire 5742G with the Tobii REX placed on the area between the screen and the keyboard and held in place with magnets. A Logitech MX500 mouse and the built in keyboard on the laptop were used as input devices. The setup can be seen in Figure 3.1.

The Acer Aspire 5742G laptop has the following specifications:
Chapter 4. Method

- Intel core i5 480M, 2.67 GHz.
- 4GB of RAM.
- 2GB Nvidia graphics card.

The Tobii REX has the following specifications:

- Sample rate of >55 Hz.
- Optimal working area is 40-90 cm from the screen with a 50 x 36 cm area where the eyes of the user will be tracked.
- Supports screens up to 27".
- USB 2.0 connection.

4.2 Testing

A group of participants were chosen to be part of the controlled experiment. The participants were selected from a sample of people that volunteered and met a set of requirements. They had to:

- Have previous experience playing games that use the first person perspective camera.
- Have previous experience using the keyboard and mouse as an input method for games.

Other parameters such as age, gender and occupation were not considered. Most participants were male, age 18-30.

When a volunteer came to the testing room he/she was asked to sit in the chair in front of the computer. The participant was told that they may choose to stop the test at any time and that they were going to be anonymous in the results. To eliminate as many distractions as possible, the blinds were pulled down over the windows, the door was locked and cellphones were turned off. To ensure a more reliable result the version of the game which the participant would begin with were changed between tests. This is to try to remove the version bias, and to balance testing.

If player A started with the traditional input(TI) version, player B would start with the eye tracker input(ETI) version.

Chapter 4. Method

The author was present the entire test and observed. If the participant had any questions during the tutorial the author would answer them. During the test itself, the author answered if the question did not affect the result in any way. The test took between 10 and 15 minutes per participant. Once the test was completed the participant was asked a series of questions about what method he/she preferred. The questions were the same as reported by Smith and Graham due to the similarity of the study. The questions were:

1. Which did you enjoy playing with more?
2. Which was easier to learn?
3. Which was easier to use?
4. With which did you feel more immersed in the gaming world?
5. For which did the controls feel more natural?
6. Which would you prefer to use in the future?

The result of the test were saved in a spreadsheet and the result from the enquiry were saved in another spreadsheet. A chi-square test will be made on the results from the questionnaire to validate a statistical significance. Also a t-test will be made using the performance results to validate a statistical significance in the different results from the two prototype versions. These tests will be done on the internet using pre-made test calculators. Quantpsy.org will be used for the chi-square test and the t-test will be made using graphpad.com.

http://www.quantpsy.org/chisq/chisq.htm
http://graphpad.com/quickcalcs/ttest1.cfm
Chapter 5

Results

Chapter 6 will present the data collected from the experiment. Also the data for each prototype and each participant will be compared. Data from the questionnaire will also be presented.

The results presented in this chapter are gathered from the experiment described in chapter 3. The volunteers consisted of 17 men and 1 woman. The data represented in Figure 5.1 are the time of all participants from both, the TI and the ETI. From the results collected a few observations can be made. Figure 5.2 shows the score difference between the two test per player, the value in the table indicate how much slower and how many more points the player lost in the ETI version. Table 5.3 show how the participants answered the questionnaire.

By analysing at the data some observations can be made:

- Average completion time for the TI version is 57.83 seconds. While the ETI version have a average completion time of around 79.50 seconds.

- The longest completion time for the TI version was 79 seconds (1 minute 19 seconds). The longest completion time for the ETI version is 145 seconds (2 minutes 25 seconds).

- The quickest time to complete the TI version was 42 seconds. The same participant also completed the ETI version with the shortest time of 48 seconds.

- All but two participants have the same or lower score while playing the ETI version of the game.

The t-test showed for the time data that the probability value(p) < 0.0001. The t-test for the score data showed that p < 0.0009. The data contain a statistical significant difference if p < 0.05. This indicates that the performance data collected from the test is different and consistent enough to not be just chance.

The results from the chi-square test shows that most of the result carries some
statistical significant value. The exception from this is the first and fifth question, enjoyment and more natural controls as can be seen in Table 5.1.
Chapter 5. Results

Figure 5.2: Score for both prototype versions per participant

Table 5.1: The results from the questionnaire and results chi-square test per question

<table>
<thead>
<tr>
<th>Question</th>
<th>TI</th>
<th>ETI</th>
<th>p-value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Which did you enjoy playing with more?</td>
<td>5(28%)</td>
<td>13(72%)</td>
<td>p = 0.0593</td>
</tr>
<tr>
<td>Which was easier to learn?</td>
<td>17(95%)</td>
<td>1(5%)</td>
<td>p = 0.0001</td>
</tr>
<tr>
<td>Which was easier to use?</td>
<td>15(83%)</td>
<td>3(17%)</td>
<td>p = 0.0046</td>
</tr>
<tr>
<td>With which did you feel more immersed in the gaming world?</td>
<td>2(11%)</td>
<td>16(89%)</td>
<td>p = 0.0009</td>
</tr>
<tr>
<td>For which did the controls feel more natural?</td>
<td>11(61%)</td>
<td>7(39%)</td>
<td>p = 0.3457</td>
</tr>
<tr>
<td>Which would you prefer to use in the future?</td>
<td>1(5%)</td>
<td>17(95%)</td>
<td>p = 0.0001</td>
</tr>
</tbody>
</table>
Chapter 6

Analysis

Using the result from the TI version test as a baseline for this experiment, as it represents the standard way of giving input while playing games in this genre, the results shows that all participants performed better using the TI version compared to the ETI version. This is supported by the answers from the post test questionnaire, Table 5.1, that shows that 83% of the participants thought the TI version was easier to play compared to the ETI version. Most of the participants also found that the ETI version was harder to learn and control compared to the TI version. This is most likely due to personal experience with the input methods, as all participants had previous experience with playing games using keyboard and mouse controls. The previous experience using TI is also a possible reason to why the participants performed overall much better on the TI version compared to the ETI version.

One other possible reason for the overall lower performance in the ETI version is the lower frame rate while the game prototype is played from the Unity editor. The reason to why the ETI version was played from the Unity editor was due to an unforeseen error with the eye tracker. The compiled version did not receive any valid data from the eye tracker which led to the game prototype being unplayable. The decision was made to play the ETI version from the editor to get any test results. A better way to do this would probably been to play the TI version from the Unity editor as well, to have the same conditions during the test to get an more even result. The frame rate in the Unity editor was 15 frames per second(fps) compared to the complied version’s 60 fps. According to K.Claypool and M.Claypool’s [4] study of the performance difference between different frame rates in a first person game, 15fps received a score of 3.5 of a maximum of 5 on a game playability scale while 60fps received a score of 4. The result from this study also shows that player movement performance follows approximately the same curve as the game playability.

The post test questionnaire shows that 16 out of the 18 participants felt more immersed in the game while using the eye tracker. A reason for this can be that the player has to be much more focused on a single element of the game at a time...
to keep his/her gaze at the object for much longer compared to the TI version. One other reason for this result is the possibility that the input method is something that is not used in games in this way, that the player stops the enemy with just his/her gaze gives a sense of more direct interaction with the game. This direct interaction with game elements can give player a higher sense of immersion.

Both the performance results and the questionnaire show similar results to the study by Smith and Graham [11]. The participants performed worse while using eye tracking compared to the the version which used the mouse and keyboard. The result from the Quake 2 test in the study showed that the participants average time using their eyes were slower compared to using the mouse and keyboard. Both studies also show that the participants felt more immersed while playing with the eye tracker compared to using only the keyboard and mouse. These results also follow the same pattern as the thesis by Odonovan [9], with worse performance results and better immersion for the player when using the eye tracker as an input method. An important distinction between this study and the two others is that the eye tracker is not used to give any movement input to the game, how this effects the results is difficult to discuss without a larger test group to get a more significant results from the questionnaire and experiment.
Chapter 7
Conclusions and Future Work

This study shows that in a controlled environment the use of an eye tracker as an additional input method gives a decrease in player performance. At the same time the enquiry after the experiment shows that a majority of the players felt more immersed in the game and would prefer to play with the eye tracker as an addition to the traditional input methods.

To answer the research question "how can a gaze based interaction method be implemented into a first person game to improve immersion and player performance, as an additional input method along side the traditional input methods?". Based on the performance data presented in Figure 5.1 and Figure 5.2 it can be seen that the interaction method designed for this study does not improve player performance in this experiment. All participants performed worse using the ETI version, both time and score were for almost all participants higher when playing the TI version. The interaction method works well to enhance the immersion of the player as shown by the answers in the post test questionnaire as shown in Table 5.1. The interaction method designed and tested in this thesis is one way to make an interaction method that works to improve immersion, but it did not improve player performance during this study.

Future work in gaze based interactions in games would include better ways to let either the player interact with the game world, or to let the game world adapt to the player’s gaze to improve player immersion and offer new game play mechanics. An example of this would be by having elements of the game either avoid or be attracted to, the player’s gaze. To improve player immersion the eye tracker can be used to control an element of a game, for example a flash light in a horror game while the keyboard and mouse controls the player movement and camera control.
References


