

**DEVELOPING A REAL-TIME TACTICS GAME USING
AN OPTIMIZED COST-EFFECTIVE
EYE-TRACKING DEVICE**

A Thesis

Presented to the

Department of Information Systems

and Computer Science

Ateneo de Manila University

In Partial Fulfillment

of the Requirements for the Degree

Bachelor of Science in Computer Science

by

Nicko R. Caluya

Juan Carlos G. Mapua

2013

ABSTRACT

Alternative human computer interfaces (HCIs) for games have been shown to engage the player in more meaningful play. This research aims to develop a real-time tactics game (RTT) and utilize an eye-tracking device (ETD) as the sole input device. Players are faced with a game environment with the usual features of an RTT game: controllable characters, intelligent enemies, path-finding techniques, and random event triggers. For player input, normal clicks or selections are replaced by gazes and cursor movements by saccades. However, a restricting factor in developing games for ETDs is the cost of the ETDs themselves. This was addressed with a cost-effective solution using an open-source eye-tracking software developed by the Information Technology University of Copenhagen (ITU) and a web camera. Eye detection, calibration, and tracking algorithms will be used to enhance the game experience. In the end, the game's success was measured using the following criteria: how well the ETD responds to players' eye movements, how much pleasure the players experience using the ETD as an input device, and if the cost-effective ETD solution is indeed a feasible setup for an RTT game. The game design was rated 3.83 out of 5 overall, which was consistent with the evaluations of gameplay and mechanics. Although, the ETD as interface was rated as challenging, testers still preferred the mouse better as of the moment. The development between the game and the input device is an area that needs improvement.

ACKNOWLEDGMENTS

The researchers would like to thank Mr. David Diy for mentoring, commenting, and guiding them throughout the course of this study. They would also like to thank Information Technology University of Copenhagen (ITU) for the open-source gaze tracker. Furthermore, they would like to thank the members, faculty, and staff of the Ateneo Laboratory for the Learning Sciences, for helping them borrow a laptop during the development stage. This research project was made possible in part through a grant from the Department of Science and Technology Philippine Council for Industry, Energy and Emerging Technology Research and Development (PCIEERD) entitled “Development of Affect-Sensitive Interfaces.” and the Engineering Research and Development for Technology (ERDT) program for the grant entitled, “Development of an Educational Data Mining Workbench.”

TABLE OF CONTENTS

ABSTRACT.....	ii
ACKNOWLEDGMENTS	iii
TABLE OF CONTENTS.....	iv
INTRODUCTION	1
1.1 Context of the Study	1
1.2 Research Objectives.....	1
1.3 Research Questions	1
1.4 Significance of the Study	2
1.5 Scope and Limitations.....	2
REVIEW OF RELATED LITERATURE	3
2.1 Real-Time Strategy Game.....	3
2.1.1. Real-Time Tactics	3
2.2 Midas Touch Problem.....	5
2.3 Artificial Intelligence	5
2.4 Developments in Eye-Tracking Devices (ETDs)	5
2.5 Applications of ETDs	6
2.6 Games Using ETDs.....	6
2.7 Factors Affecting Meaningful Play.....	7
2.8 Summary of Related Literature.....	8
THEORETICAL FRAMEWORK.....	9
3.1 The RTT Framework	9
3.1.1 Player Objectives	9
3.1.2 Available Resources.....	10
3.2 The ‘Snap Clutch’ Algorithm	10
3.2.1 A Moded Approach to Midas Touch	11
3.3 Artificial Intelligence and RTT Games.....	12

3.3.1 Reactive AI	12
3.3.2 Fuzzy State Machines	13
3.4 Eye Movements	13
3.4.1 Eye-Tracking Algorithms	14
3.4.2 Detection	14
3.4.3 Data Types	16
3.5 Metrics for Eye Movement Data Types	17
3.6 Metrics for Meaningful Play	18
3.6.1 Leblanc's Taxonomy of Pleasures	19
3.6.2 Schell's Elemental Tetrad	19
METHODOLOGY	21
4.1 PC Hardware Requirements for Development	21
4.2 Device Requirements for Eye-Tracking	21
4.3 External Environment Setup	22
4.4 Game Development Tools	22
4.4.1 The Unity Game Engine	22
4.4.2 Blender 3D	22
4.4.3 Programming Languages	22
4.4.4 External Libraries.....	23
4.4.5 Eye-Tracking Software	24
4.5 Survey Creation	25
RESULTS AND ANALYSIS	26
5.1 Eye Calibration Results.....	26
5.2 Survey Analysis	29
CONCLUSION.....	32
6.1 Game Design.....	32
6.2 Mapping of Eye-tracking	32

6.3 Testing and Survey	32
6.4 General Conclusions	33
BIBLIOGRAPHY	34
APPENDIX A	39
I. Appearance	39
II. Gameplay and Mechanics	39
III. Use of Eye-tracking Device	40

CHAPTER 1

INTRODUCTION

1.1 Context of the Study

The current input peripherals for video games consist of the traditional controllers for consoles, a combination of keyboard and mouse for personal computers, and touch-screen capability for some recently-manufactured handheld devices. Other modern and innovative methods for gathering input include motion-sensing devices, like Microsoft's Kinect™ [1], Sony's PlayStation® Move™ [2], and Nintendo's WiiMote™ [3]. There have also been many studies in the field of HCI on new modes of input, such as a brain-computer interface for reading brain wave activity [4] or a ring that acts as a three-dimensional "air-mouse" [5]. One particular concentration in HCI that this study is interested in is eye-tracking technology.

The currently prevailing and widespread use of the eye-tracking technology tends to lean toward commercial applications: for instance, to find out which areas of a website are visitors most likely to look or fixate their attention at, and devise advertising strategies based on that information. Eye-tracking is also studied and applied in other fields, such as health, criminology, and public safety [7], among others. But there is little material on applying eye-tracking to gaming, and there are very few games that employ this technology whether exclusively or as a feature.

1.2 Research Objectives

The main objectives of this research are as follows:

1. To develop a game that relies exclusively on the eyes as an input source, and
2. To gauge the impact and workability of using eye-tracking technology on the player's gaming experience without resorting to expensive eye-tracking devices and solutions.

1.3 Research Questions

This research attempts to answer the following questions:

1. How does one design a real-time tactics (RTT) game that utilizes an eye-tracking device?
2. Based on the metrics set by Poole and Ball [19], what can be considered an appropriate mapping of eye movements to player actions in an RTT game?
3. How does one test the design of an eye-tracking-enabled RTT game?

1.4 Significance of the Study

This research will be helpful in exploring the potential of the eyes as a very promising and efficient source of input control not just for gaming, but also for practical, everyday computing, as they are said to be the body's "fastest and most fatigue-resistant muscles" [6]. Incorporating eye-tracking input in games could potentially remove the need to physically manipulate a controller with the hands.

This study can also be used as a reference for future work on eye-tracking interfaces for people with physical disabilities or undergoing sensory rehabilitation. The importance of having an HCI like this would be most beneficial to people with hand amputation, deafness, or muteness, and can rehabilitate people suffering from vision problems.

1.5 Scope and Limitations

The output of this study will be an eye-tracking-enabled RTT game designed to run on the Microsoft Windows 7 PC platform. For the game's eye-tracking component, the open-source application ITU Gaze Tracker will be used, and will therefore require a web camera with a resolution of at least 640x480 and a frame rate of 60 MHz, as well as infrared light sources.

The focus of this study is limited to the utilization of eye-tracking methodologies and applying them to a gaming context, and its impact on the enjoyment or feasibility as an alternative or replacement for the mouse and keyboard. As such, no attempts will be made to formulate new algorithms or develop new software for eye-tracking, but merely use existing ones and modify them to suit the needs of the system and the study.

CHAPTER 2

REVIEW OF RELATED LITERATURE

2.1 Real-Time Strategy Game

A real-time strategy game is a classification of video games that employs strategy, which most of the time involves military planning, primarily set in a real-time mode. [8]. Another alternative definition comes from Geryk and his investigation on how Brett Sperry came up with the term to market “Dune II: The Building of a Dynasty” in 1992 for Westwood Studios. In order to explain the game, Sperry refused to present the game as merely ‘strategy’ or ‘wargame’ because of the fear that these terms cannot encapsulate the new idea of the game; the term ‘real-time strategy’ became justified because of its target audience back in the 1990s, having the same volume as that of Sid Meier games’ followers [9].

To approximate reality, real-time strategy (RTS) games were developed alongside and gradually replaced the popularity of the turn-based strategy games to keep players from waiting for either other players or the computer’s artificial intelligence. Since the shared-time component and simultaneousness become key elements in the game, the RTS games rely not only on strategy but also on the player’s reflexes and ability to think quickly. [10]. RTS also minimizes the complicated sequence of rules used in turn-based games [11].

2.1.1. Real-Time Tactics

To further define the genre, there are two different subgenres of the RTS games: traditional and real-time tactics (RTT). The difference between the two subgenres lies on managing resources, building bases, and developing technology within the games; the latter lacks these features, where players get fixed resources such as characters or buildings instead of acquiring them during gameplay [8].

An example of an RTT game is *World in Conflict*, published by Ubisoft and developed by Massive Entertainment for Windows. *World in Conflict*’s plot is set in 1989 and offers an alternative history to the Soviet Union’s rise to power [25]. The figure below (Fig. 2.1.) shows a screen shot of the game.



Fig. 2.1. A screen shot of “World in Conflict” [36]

Another popular example is Warhammer 40,000: Dawn of War II, made by Relic Entertainment and published by THQ for Windows, and is based on an earlier video game version [26].



Fig.2.2. A fight scene screen shot of “Warhammer 40,000: Dawn of War II” [37]

These games demand fast and precise interaction with the player, and already use artificial intelligence engines in order to challenge the player more. However, making room for processing eye-tracking as HCI can only go as far as how fast it is to translate eye movements into what should be mouse clicks.

2.2 Midas Touch Problem

The eyes' mode of input cannot be directly translated to that of a mouse, since they are constantly engaged all the time [38]. On its own, eye-tracking can only translate eye movement into cursor movement; there is no way of signalling the system the equivalent of a mouse click, let alone distinguishing between left-clicks or right-clicks. This is the premise of the "Midas Touch¹" problem: eye-gazing interfaces are also plagued with the problem of lacking the ability to toggle between active or inactive states at will for selection or de-selection purposes. To address this shortcoming, a toggling mechanism was needed to enable this transition between states, similar to a clutch for shifting gears in manual-transmission cars. According to Glenstrup and Engell-Nielsen [38], a good clutch must satisfy the following conditions:

- a. Quick and easy to operate;
- b. Not increase the cognitive load unnecessarily;
- c. Not disturb the user's gaze-pattern.

2.3 Artificial Intelligence

Artificial Intelligence (AI) is a branch of computer science that aims to simulate, as accurately and realistically as possible, the thought processes humans and animals are capable of [32]. Specific kinds of artificial intelligence included in the development of the game are the reactive AI [31] and the utilization of Fuzzy State Machines (FuSMs) [33] which will later be discussed in relation to their corresponding human simulation of tactics in the game.

2.4 Developments in Eye-Tracking Devices (ETDs)

There have been early attempts in the late 1980s to come up with a human interface using the eyes for sensing, such as the dual-Purkinje eyetracker [18]. Several recent eye-tracking applications and research utilize the Tobii™ brand of hardware [14]. However, with the price ranging from €20,000- €50,000 during the time of research

¹ This problem is associated with the myth surrounding the titular king and his vain ability to turn anything he touched into gold. He was unable to control his power.

(M. Rodrigo, personal communication, June 6, 2012), the number of efforts in utilizing commercial ETDs in academic research has been significantly low.

Recent studies have been recreating the device to find a cost-effective configuration [16] [17]. Topal et al. insists on creating cost-effective software for the device such that it requires low computations, instead of more complex algorithms [16]. On the other hand, Li et al. balanced the device and software components in order to create a cheaper ETD [17].

Also, there have been different tracking techniques that detect the eyes' fixation, saccades (the eye movement used in reading), and rolling or rotation. Apart from these, there have been different attempts to optimize tracking the eyes using different algorithms, and contrasting each of these with metrics [19]. These different movements, algorithms, and metrics (such as accuracy and sensitivity) are of great potential in the development of this study's ETD.

2.5 Applications of ETDs

Much of the research done on eye-tracking technology had the purpose of creating eye interfaces or ETDs for those who have been physically handicapped, particularly hand amputation, deafness, and muteness. Using a neural network (NN)-based texture classifier, a study has been done to improve a handicapped person's navigation [20]. Current news has also shed light onto eye-tracking as a revolutionary way of learning and recovering from disability [21] [22]. From developing a way to type using the eyes [21] to creating games for specific disabilities [22], the integration of these devices in the lives of handicapped people have been crucial for their recovery and improvement.

2.6 Games Using ETDs

Games accompanied by the use of eye-tracking devices have already been explored. As mentioned earlier, Lin et al [23] produced not only an eye-tracking application that can serve as a game, but also help in the rehabilitation and exercise for the eyes. In another study by Isokowski [24], the gaze was the highlight of the game, and those different durations and types of gazing can be used to manipulate the game and create a variety of interactions with the screen.

However, academic studies focusing on creating games have been few, primarily because these games have been produced for commercial consumption, just like the EyeAsteroid Arcade Game by Tobii, as shown in the figure below (Fig.2.6.1).



Fig.2.3. The Tobii EyeAsteroids Arcade Game [43]

There is also an eye-tracking game involves another alternative, this time, an output device such as a transparent screen monitor, with built-in webcams, with gameplay that only involves growing and moving balls on the screen [44].

Games using ETDs have continually developed over the years because of the possibility of the eyes to become an alternative to the mouse and keyboard as human interfaces for computers.

Gaze fixation, as mentioned earlier, has been explored to control the movement of virtual characters, filtering these fixations in order to execute the correct commands [19] [24].

2.7 Factors Affecting Meaningful Play

In the context of gaming, to quantify user enjoyment or pleasure has been explored through different frameworks by different researchers. Schell [41] has compiled some of these perspectives, including his own, in order to approximate the intensity of each user when immersed with gameplay. Aside from Schell's framework, this study will

also explore Leblanc's taxonomy of game pleasures, which will be discussed thoroughly in the succeeding chapter.

2.8 Summary of Related Literature

The real-time tactic genre focuses more on the character movements rather than the resource management aspect that the traditional real-time strategy game offers. In this case, the fixed number of units in the game would certainly enforce more attention from the player, hence the need for the concrete actions demanded by using tactics in a game rather than the bigger abstraction of imagining strategy. Recent RTT games mentioned earlier rely more on synergy among controlled characters, hence pressuring the player to execute decisions in the gameplay as fast as possible. The potential of the eyes in responding to this as faster stimuli would eventually keep the player rushing to initiate action.

The developments in the eye-tracking have a potential not only to facilitate behavior in alternative forms of HCIs, but also to produce applications that are most appropriate with them. With the rise of games that demand more diversity in the gameplay, choosing another HCI aside from the mouse would excite developers and users alike in creating such applications. The development of these kinds of games would also contribute to eliminating the discrimination brought about by physical disabilities, in the event that the game indicates a reasonable amount of replay and enjoyment.

Spakov and Miniotas [50], although presented clustering algorithms that can help in eye-tracking, and studies mentioned earlier provide a possibility that combination of these algorithms applied simultaneously with a gaming application, issues of resource allocation on program execution is still important to observe.

CHAPTER 3

THEORETICAL FRAMEWORK

3.1 The RTT Framework

To properly define and distinguish between RTS and RTT genres, dictionary definitions of the words strategy and tactics are provided, as they tend to be confused and used interchangeably. Strategy is “the science and art of employing the political, economic, psychological, and military forces of a nation... to afford the maximum support to adopted policies in peace or war [34],” and tactics is “the science and art of disposing and manoeuvring forces in combat [35]”. In more general gaming terms, military strategy is more of a long-term approach to resolving conflict, focusing on the many aspects of operation, and generally deals with a much larger scale of command. Military tactics, on the other hand, is more of a quick and direct short-term solution, with the intent of defeating the enemy through precise and methodical combat techniques and firepower, while effectively utilizing fixed and limited resources. Instead of the initial requirement to establish a base, gather resources, and amass an army, RTT games focus on the action and micromanagement of small numbers of units.

3.1.1 Player Objectives

Generally, the primary objective of any RTS/RTT game is to defeat the enemy by whatever means necessary. However, in an RTT game, the player should be more concerned about the survival of his units since they are generally few and fixed in numbers.

The player’s objectives in RTT games vary depending on the game itself or what type of mission is given. In RTTs like *World in Conflict* [25] where the gameplay is stealth-oriented, the player must carefully manoeuvre his units around the battlefield, target selected areas that can cause maximum impact with minimum effort, and perform a series of actions in order to accomplish the mission.

Tactics from other games include taking out enemies from a distance with a sniper, knocking enemy soldiers unconscious, tying them up and hiding away their bodies, or disarming landmines in order for reinforcements to arrive.

Players are expected to command their units in such a way as to accomplish a specific requirement or mission directive. They may use whatever weapons, skill sets, tactics, support, and other resources immediately available to them to finish the job. In certain situations, they may also be restricted in how they go about completing such objectives, such as setting up time limits, certain “essential” characters must be kept alive throughout a mission (like rescued hostages or enemy leaders as prisoners-of-war), or enemy engagement must be kept at a minimum or none at all. There are cases where objectives are issued in a specific sequence - objective A must be completed before objective B, and so on; again World in Conflict [25] does this, for instance in the first training mission, where guards must be taken out before the enemy officer arrives, then secure the supply depot, with a few more objectives, until the border can be safely opened for reinforcements. Oftentimes, certain missions can have secondary or “bonus” objectives which, when completed, will reward the player in some way (e.g., upgraded weapons, experience, new skills, etc.) to help in advancing through the game.

3.1.2 Available Resources

In traditional RTS games, the player is provided several starting units and resources, which will be used to build up production structures and primary defences. To expand one’s base, a player must procure additional resources from the surrounding terrain (e.g., ore to produce credits in most Command & Conquer games or trees for lumber and gold mines for gold in Blizzard’s WarCraft series [9]).

However, in RTT games, there is typically no base-building or resource-gathering to be done; if there were, they serve very minor roles and not really critical to gameplay. The player has, at his disposal, a fixed number of units with which he must be able to accomplish the current mission.

3.2 The ‘Snap Clutch’ Algorithm

As mentioned earlier, the Midas Touch problems brings the problem of insignificant gazes, and wrong eye movement detection that should have not been considered as player engagement with the game. As a solution, the ‘Snap Clutch’ algorithm helps in establishing a more convenient interface and navigation.

3.2.1 A Moded Approach to Midas Touch

A solution to the Midas Touch problem is, as mentioned in the previous section, to devise a “clutch” mechanism that would engage and disengage gaze control at will. Istance et al. [39] have developed a “snap clutch” system, where in addition to merely emulating mouse movement using the eyes, an innovative method of quickly toggling between states is achieved by glancing at four different directions off-screen. They have four (4) modes with which to interact with the on-screen interface, which are ‘Eye Control Off’, ‘Dwell Click’, ‘Park it Here’, and ‘Drag from Here,’ each with its own visual cues and unique effects, visualized in Figure 3.1 below.







Mode	Eye Control Off	Dwell Click	Park it Here	Drag from Here
Eye gesture to enter the mode	glance up	glance left	glance right	glance down
Pointer appearance	 green pip	 red pip	 green crosshair	 green arrow
Dwell effect and the changed pointer appearance	none	mouse click system sound (no visual change)	parks the pointer  red crosshair (+system sound)	activates drag  red arrow (+system sound)
			Dwell when the cursor is parked (red) causes a click on the pointer position Looking at the pointer picks it up again (turns pointer green)	Drag is stopped by changing the mode, or by looking at the original cursor position

Figure 3.1 Attributes of the Snap Clutch Modes [39]

The ‘Eye Control Off’ mode [39] disengages any mouse emulation but the gaze-tracking is still active, allowing users to temporarily transfer control to the mouse and rest their eyes from constantly traversing the screen; this mode can be activated at any time by glancing upwards, beyond the top border of the screen area. In ‘Dwell Click’ mode [39], mouse emulation is engaged, and any prolonged fixation or “staring” (known as “dwelling”) at a particular point would generate a click event there; this mode is activated by glancing beyond the left-hand border of the screen area. The aptly-named ‘Park it Here’ mode [39] allows the user to dwell at a certain position

and leave the mouse cursor there, and a dwell action anywhere on screen would generate a mouse down and up event at that position. The cursor can be “picked up” again simply by looking back at it, and the cycle repeats; this mode is activated by glancing beyond the right-hand border of the screen area. The final mode, ‘Drag from Here,’ [39] is similar to the ‘Park it Here’ mode in activation (dwelling on a point leaves the cursor there) but slightly differs in execution: it initiates a mouse drag action from the starting position by moving the gaze point anywhere on the screen. The dragging action can be finished by switching modes or by looking back at the starting position. This mode is activated by glancing beyond the bottom border of the screen area.

3.3 Artificial Intelligence and RTT Games

In computer games, and especially in real-time tactics games, the AI system’s primary function is to provide the illusion of ‘intelligent’ and challenging opponents, using predefined tactics and scripted commands to fulfill specific objectives and to generally hinder the player’s progress. RTTs allow for more individual-oriented gameplay and micromanagement opportunities.

In addition, player-controlled units, although designed to follow the player’s orders to the letter, still need a certain degree of autonomous behavior and should be able to make their own decisions based on the situation, or while waiting for further instructions. The same applies to enemy units, albeit with the absence of a human controller. The different tactics and strategies employed by the AI system, reflected within each unit’s individual or group actions, are very important aspects in properly defining the genre and enhancing the gameplay experience.

3.3.1 Reactive AI

Most AI techniques and behaviours utilized in games are reactive in nature. According to Champandard [31], the reactive approach, due to its simplicity and efficient implementation, is ideally very suitable in the context of computer games. Reactive AI is deterministic: a particular input will always produce a particular output. Multiple inputs can map to a single output, but never the other way around, thus removing the risk of ambiguity [31]. The biggest advantage of this approach is

that it is highly predictable: optimizations are more effective, debugging is easier, and the time complexity for obtaining results is generally constant and near instantaneous [31].

Applying reactive AI to RTTs can ease the burden on the processor by not being too computationally-intensive, especially when multiple units are constantly moving on-screen, each performing its own subset of actions. This leaves room for the processor to focus more on the complex and mathematically rigorous components of the game, such as physics simulations and collision detection. Furthermore, the player can see almost immediately the cause-and-effect relationships of their actions, and adjust one's game tactics accordingly.

3.3.2 Fuzzy State Machines

According to Schwab [33], Fuzzy State Machines (FuSMs) are a better alternative to using Finite State Machines (FSMs): mathematical models that, when given an input, will output into a change of state of that model. FuSMs build up on the structure and reproducibility of FSMs by incorporating the uncertainty factors or 'flying blind' nature of RTS AI decision-making. Even though the AI lacks information pertaining to the player's movements (when 'fog-of-war' or line of sight is important) or resources but has a clear goal (to see to the human player's demise), it can still formulate 'intelligent' decisions using the parallel nature of FuSMs in determining the amount of effort needed to execute each facet of command separately at any given time [33]. This blending of behavior produces new methods for the fulfillment of an end result, and makes for a more varied and contextual AI, rather than having reveal the entire game state to the AI system to aid in making smart decisions.

3.4 Eye Movements

In order to explore the alternative uses of the eyes to generate the same interactions as that of the hands in mouse and keyboard as interfaces, it is necessary to first take note of the common eye movements that have been involved in many studies in eye-tracking.

3.4.1 Eye-Tracking Algorithms

Most techniques for detecting eye movement primarily rely on image processing and the different clustering algorithms that come along with them. Spakov and Miniotas [50] explains the two divisions of eye-tracking algorithms essential to detecting the eyes: the mean-shift and the distance-threshold algorithm. In both solutions, there is an assumption that the fixation will be a weighted mean of just x- and y- coordinates from the common two-dimensional displays, and collectively, a weighted mean of these fixations in order to come up with polygons that constitute a gaze fixation area.

The distance algorithm, as the name suggests, clusters the fixation according to each fixation point's distance from each other in order to create an area. When a predefined distance has been set, the points outside of the area will also be calculated according to distance to either come up with a new cluster or belong to a pre-existing one.

As a modification to the distance algorithm, movement of fixation points is considered in the mean-shift algorithm. In this case, it eliminates the disadvantage of too large clusters and promotes a more specific area of interest from the user. The three-step formula below illustrates the flow of data from the $\{x,y\}$ coordinates and σ used as covariance in usually a multivariate Gaussian with zero mean, or kernel function K ,

$$K(x_i, y_i) = \exp\left(-\frac{x_i^2 + y_i^2}{\sigma}\right) \quad (1)$$

to the weighted mean $S(X)$ of the nearby points, and finally a $S_{FIX}(X)$ function accompanied by applied weights W .

$$S(X) = \frac{\sum_j K(X - X_j)X_j}{\sum_j K(X - X_j)}. \quad (2), \text{ or } S_{FIX}(X) = \frac{\sum_j W_j K(F - F_j)F_j}{\sum_j W_j K(F - F_j)} \quad (3)$$

3.4.2 Detection

There is usually an LED with infrared light as part of the webcam that scans the image of the eye. The figure below shows a rough sketch of the image.

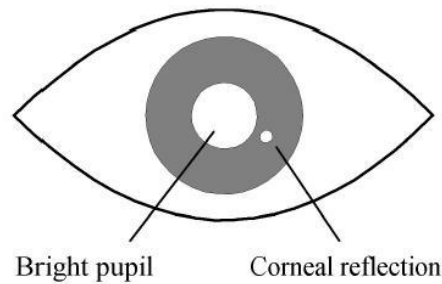


Fig. 3.2. The important areas of eye detection, image from Poole and Ball [19]

The pupil and the corneal reflection (Fig. 3.2.) have been two indications for the detection of the eye movement because of its contrast with the rest of the eyeball. In this case, the right adjustment and position can be set up so that the following detections (Fig. 3.3) can be observed:

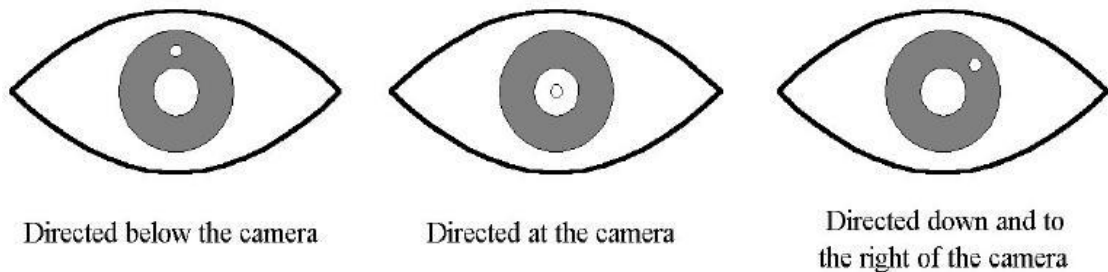


Fig. 3.3. Corneal reflection position changing according to point of regard, image from Redline and Lankford [40]

Calibration for these ETDs must then be catered to the desired point of regard. However, this will be further discussed in the methodology. The setup of the three elements (the eye, the camera, and the screen) is also a key factor in determining the best type of calibration and algorithm to be used. The figure below (Fig. 3.4) shows these different elements and the direction of the tracking, as illustrated by De Santis and Iacoviello [29]:

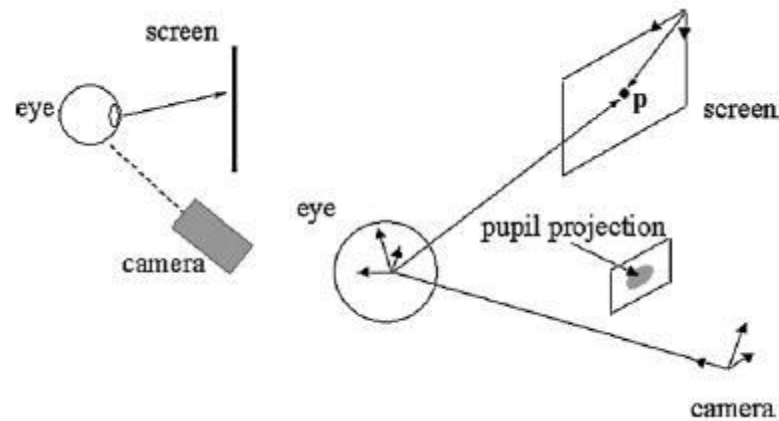


Fig. 3.4. Setup of the eye, camera, and the screen and the direction of detection [29]

In this setup, to produce near-ideal results, the person must sit upright enough to let the camera scan the eyes. Also, the level of the screen and the camera must also be at different elevations to prevent blockage.

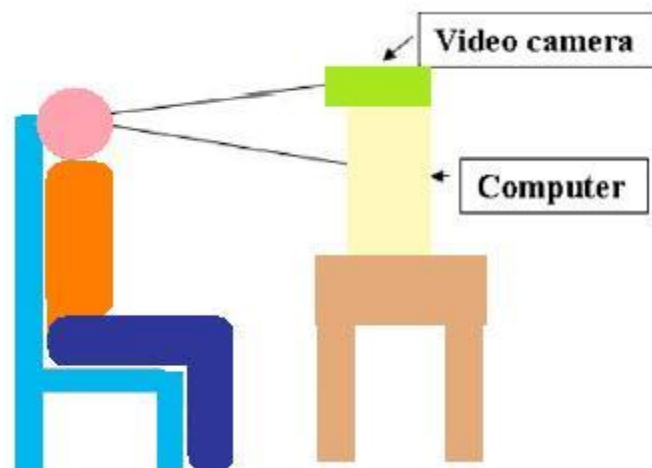


Fig. 3.4.1.4. Setup of the person, video camera, and computer for real-time eye-tracking [29]

As also illustrated by De Santis and Iacoviello [29], the setup for the person, the computer, and the camera, must maintain a balanced positioning, especially in capturing a robust real-time eye-tracking, shown above (Fig.3.5).

3.4.3 Data Types

According to Ramloll et al. [28], the typical data types of interest of eye-tracking are divided into four orders. The first one consists of x, y, and z data, pupil diameter, and blink rate; the second one refers to fixations, saccades, and pursuit eye movements;

and the third and fourth order data types relate to the complexities and variations of the scan-paths [28].

Poole and Ball [19] define these data types in their glossary of terms. Gaze fixation, also known as “dwell”, “fixation cluster”, or “fixation cycle”, where the eye concentrates on what was also defined as an area of interest. Saccades on the other hand, are 20 to 35-millisecond transitions from one gaze fixation to another; similar to the way we read or follow an image with our vision. Finally, when a sequence of these gaze fixations have been linked by saccades, there can now be a scan-path, a data type of a higher order [19]. Our study will focus more on the second-order data types. As proposed by Møllenbach et al [49] and for the purpose of determining essential features in the game, the single gaze gestures (SGGs) can be detected, as it approaches a generic form of input.

3.5 Metrics for Eye Movement Data Types

There have been numerous studies contributing to interpretations in eye movements; the most significant ones are compiled in a study by Poole and Ball [19]. Some of these metrics are of importance to the development of the eye-tracking game. Since the purpose to attend to the object in sight is evident in gaze fixations, its feasibility to become an appropriate gesture to aiming has been explored in a study by Leyba and Malcolm [27]. Also, other factors such as fixation durations and time to fixation on-target will tell whether the image presented was difficult to process, given importance, or engaging [19]. The following metrics which we will be using will come from the compilation by Poole and Ball [19] with their corresponding possible implication to the game derived from the original metric descriptions:

Table 3.1. Derived Metrics for Gazes

Gaze Metric	What It Measures
Number of fixations overall	The interface is not optimal enough for searching, being less efficient as overall fixations increase.
Fixations per area of interest	There is a more noticeable area in the interface, attention-grabbing, and important.

Fixation duration	The player may not understand the interface or may be engaged with it.
Time to first fixation on-target	The faster the player spots something on an area or object, the more attention-grabbing it is.

Table 3.2. Derived Metrics for Saccades

Saccade Metric	What It Measures
Number of saccades	The player maybe searching for more clues.
Saccade amplitude	The player searches meaningfully since the the range of sight will be longer.
Regressive saccades	The player forgot to attend to an object and has to go back and search again.

Table 3.3. Derived Metrics for Scanpaths

Scanpath Metric	What It Measures
Scanpath durations and lengths	A scan path maybe used to plot an automated map of where the character goes.
Spatial Density	Since immediate menus need more direct search, a smaller density means better menu navigation.

As these metrics have already been observed and deemed important for the study of different eye movements, certain explorations on the game design will adhere to the various results exhausted from the studies they originate from. Therefore, the application of these metrics factors in how much they can be replicated during the actual game, such as gazing to the right control in the menu, or efficiently drawing out a scan-path of attacks. As for the keyboard on-screen interface similar to Arai and Mardiyanto [30], saccades and scan-path metrics will also be taken into consideration to create a convenient and discernible layout catered for real-time gaming.

3.6 Metrics for Meaningful Play

To pursue further studies on eye-tracking games, the researchers also noted important features in gameplay particularly the metrics presented by Leblanc and Schell [41] towards an meaningful play experience. Leblanc proposes eight factors in his taxonomy of pleasures. On the other hand, Schell proposes factors to keep in mind in game development known as the elemental tetrad.

3.6.1 Leblanc's Taxonomy of Pleasures

Leblanc places sensation, fantasy, narrative, challenge, fellowship, discovery, expression, and submission as part of his Taxonomy of Game Pleasures. Though it can be very helpful to know how exactly these factors manifest in the gameplay, the researchers have only included sensation, fantasy, discovery, and submission to become the highlight of these metrics.

Sensation, as Schell [41] further explains it, can be experienced through the “toy” that the player is using. Since this relates more to external input devices such as HCs, this sensation can be measured through the player's enjoyment of the eye-tracking device, this game's “toy”.

Fantasy covers the pleasurable illusion of an imagined world, making the players immerse into avatars or characters that they are not. In this case, the effectivity of this game becoming a controller of all the units in the field would fall under fantasy, such that the player assumes tactics for every unit, or a collection of units.

The element of discovery becomes a key game pleasure every time the player finds out something new that can help in the progress and movement of the game. In the event that secret features and tactics will be revealed in the game, the discovery element can be quantified.

Finally, submission as an element deals with the player subjecting oneself into the rules and regulations of the game that in the end can benefit the gameplay through earning rewards and points.

3.6.2 Schell's Elemental Tetrad

In a book by Schell [41], he proposes a framework that shows the formation of game relies on four key elements: the mechanics, the technology, the story, and the

aesthetics. The latter two factors will become less important in the context of this study, but significant enough not to be ignored.

The mechanics of the game becomes very important in the gameplay, such that these restrictions become a necessity for a game to be a game [41]. While the mechanics aspect of the game is more visible in forming the game, the technology behind it will determine the success of the game. Aside from the PC requirements which will be discussed further in the methodology, the complexity brought about by the ETD through the software and hardware requirements can compromise the overall gameplay, and will cover the technical part of this study's first research question.

CHAPTER 4

METHODOLOGY

4.1 PC Hardware Requirements for Development

The hardware used to develop and run the eye-tracking game system (and will serve as the benchmark for the minimum system requirements of the game) was a Toshiba Satellite L505 notebook computer, with a built-in video card (Mobile Intel® 4 Series Express Chipset Family), an Intel Pentium® Dual-Core processor running at 2.00 GHz, 3 GB of RAM, and ran on a 32-bit Windows Vista Home Premium Operating System. The preliminary recommended settings came from a desktop computer on which the game was developed: powered by an Intel® Core™ i3 processor running at 2.10 GHz and an NVIDIA GeForce GTX 550 Ti dedicated graphics card, 4 GB of RAM, and running on a 64-bit Windows 7 Operating System.

4.2 Device Requirements for Eye-Tracking

The primary input device for the eye-tracking game system was a remote set-up of a USB camera or similar hardware. For the eye-tracking software to accurately and effectively track eye movement, the recommended hardware specifications for a USB camera required decent video quality, preferably of 640x480 resolution or better, and a reasonably fast frame rate, at least 30-60 frames per second. In addition, if the camera contains an IR blocking filter, it must be removed for IR illumination to work, and optionally attach a visible light blocking band pass filter to further improve eye-tracking [42].

The developers of the eye-tracking software also stress the importance of IR illumination for remote set-ups. Most cameras already come with IR light sources, commonly used for “night mode” or in environments with low lighting, but may be too weak for accurate eye-tracking. External IR sources or lamps are therefore needed, and a list of verified configurations can be found below (a more extensive list is available in the developers’ forum [47]):

- Clover Electronics IR010
- Sony HVL-IRM

- DealExtreme IR 48 LED InfraRed Illumination

4.3 External Environment Setup

As much as the limitation of this study lies on using a simple webcam instead of a commercially-released ETD, the image captured during calibration and during the game itself required adjustments in the environment where the user would play the game, especially in terms of lighting. Through the adjustments done before calibration, the user can move toward or away from the camera, until the black outline would appear in the ITU Gaze Tracker window. In other instances,

4.4 Game Development Tools

4.4.1 The Unity Game Engine

The official website of the Unity game engine describes the product as a development engine capable of creating an interactive, high-quality, high-performing 3D content across many platforms in a time-efficient and cost-effective manner [45]. These are key points that the researchers considered in choosing the game engine, as well as its rich online resources and an active community.

4.4.2 Blender 3D

Blender 3D, or simply Blender, is an open-source and cross-platform 3D content creation suite [48] that was used primarily for game asset creation such as characters and levels. It features an extensive toolset and professional-grade software technologies for a free program, and its various functionalities fit the needs for the study's game development aspect. Blender's native `.blend` file extension is also supported by Unity [12], making integration into the main development pipeline easier and faster.

4.4.3 Programming Languages

The programming languages listed below was used for the development of both the RTT game and modifications to eye-tracking software.

4.4.3.1 C#

C# is a simple, multi-purpose object-oriented programming language developed by Microsoft within its .NET initiative. It was used in the implementation of the eye-tracking software, and will be used in the Unity game engine for various scripting purposes. Both programs use the .NET Framework as a foundation, making it easier to open communications and transfer data between them.

4.4.3.2 JavaScript

JavaScript is a prototype-based scripting language that is dynamic, weakly typed, and is multi-paradigm that supports object-oriented, imperative, and functional programming models. While JavaScript is mainly used in web-based applications, it is supported in Unity for the above reasons. Also an advantage of using JavaScript in Unity is that every JavaScript automatically derives from MonoBehaviour, the base class in Unity that all scripts derive from (using C# or Boo would require an explicit derivation) [46].

4.4.4 External Libraries

4.4.4.1 A* Pathfinding Project

The A* Pathfinding Project is a powerful and easy to use pathfinding system developed for use in the Unity game engine, paving the way for smarter AI to move around obstacles, and is multithreaded to reduce its computation impact on the game's frame rate [13]. Pathfinding is a major aspect in the RTT game's AI system for large numbers of units to move through the terrain and around objects and other units in a realistic and logical fashion. The library has an extensive list of features to accomplish proper pathfinding, a few of which are as follows: various graph types (Point, Navmesh and Grid graphs, possibly more with custom scripts) that serve as guides for movement, post-processed path modifiers using smoothing algorithms to fine-tune

paths that the AI follows, and real-time graph updating to anticipate sudden changes in the game [13]. We used this A* Pathfinding Project for our game's automatic pathfinding primarily for the computer as enemy, but can also be available for the player as convenient tactics.

4.4.5 Eye-Tracking Software

This study utilized the ITU Gaze Tracker, developed by the Gaze Group, a research group at the Information Technology University of Copenhagen, Denmark, with support from the Communication by Gaze Interaction Association [16]. It is a freely-distributed, open-source eye-tracking software that seeks to deliver a low-cost alternative to commercial gaze tracking systems, and to make the technology more accessible to the global community for further research and development [47]. It is video-based, and therefore needs a camera equipped with infrared (IR) illumination to capture and process eye-tracking data.

As mentioned in the previous chapter, eye detection involves calibration, the proper set-up among the user, the screen, and the webcam. The ITU Gaze Tracker takes care of the calibration results and grades the process by displaying a result window showing significant eye movement patterns, and a 1-to-5 star scale quality equivalent. The figure shown illustrates one of the trials.

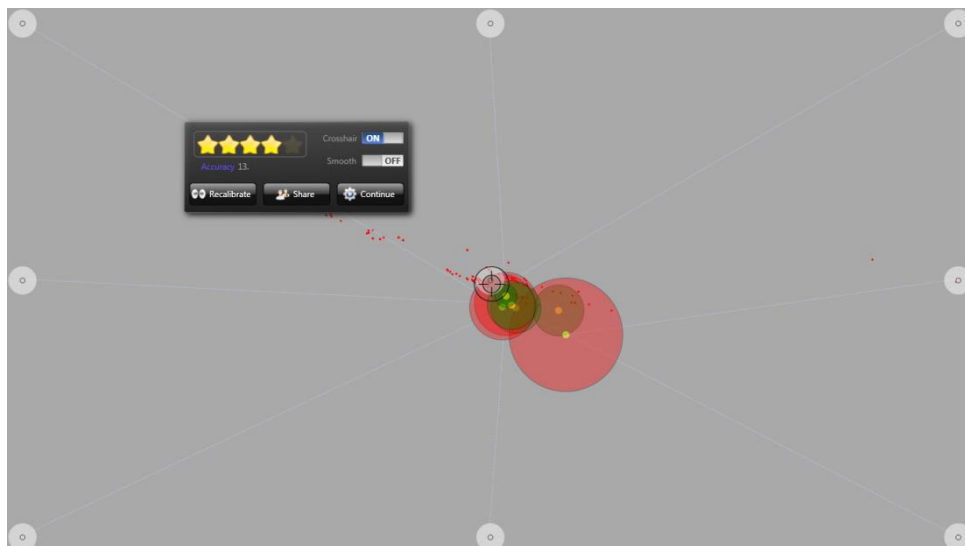


Figure 4.1. ITU Gaze Tracker's result display window, showing a 4-star calibration

Since it was also mentioned that the detection would follow a remote setup, the head-mounted option was excluded from the choices, and only the remote monocular and remote binocular detections will be taken into consideration. However, due to significant amount of delay ($>5s$) in the detection, the remote binocular option were not used in trials, leaving only remote monocular detection as the best tracking method.

4.5 Survey Creation

Three people participated in the initial testing phase of the game, where they will play the same level of the game, and in the end evaluate the game in terms using a 1 to 5 scale of quantitative judgment.

The metrics for meaningful play mentioned earlier was used for creating this survey. In the elemental tetrad by Schell [41], the study will focus most especially with real-time tactics aspect of the gameplay, and the eye-tracking technology as an interface. The story and aesthetic elements will be considered as well, but will be treated with less importance.

In the context of the game and Leblanc's taxonomy of pleasures, the meaningful play must be concentrated upon the challenge given to the players in terms of using the interface and in the finishing the levels of the game. Nevertheless, other factors such as sensation, narrative, discovery, and submission will be given a different framework of questions to ensure that these observations can be quantified.

CHAPTER 5

RESULTS AND ANALYSIS

5.1 Eye Calibration Results

For the first round of test results, the researchers relied on each tester's eye tracking calibration output and answers on the survey as data. Three testers (A, B, C) participated in this round. The methodology was revised such that the eye-tracking software was set to a remote binocular setting instead of using a remote monocular, as the camera upgrade eliminated the delay difference ($>5s$) between the two. With this, the accuracy results can now be categorized according to which eye was detected. The following images show the screenshots of the three testers.

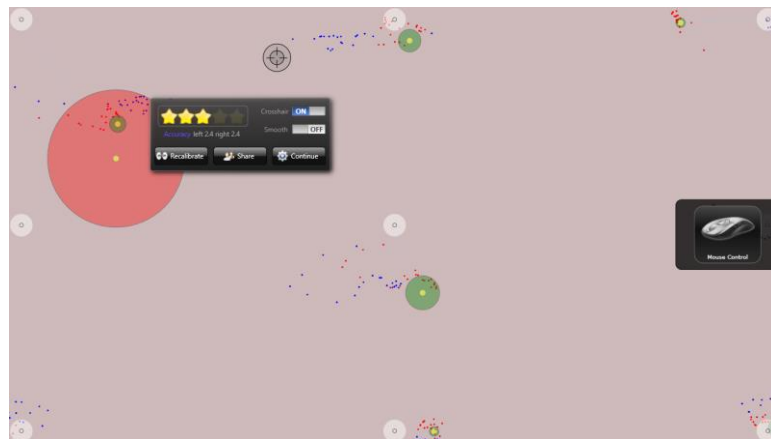


Figure 5.1 Tester A's 3-Star Calibration Result, with 2.5 Left, 2.4 Right Eye Accuracies

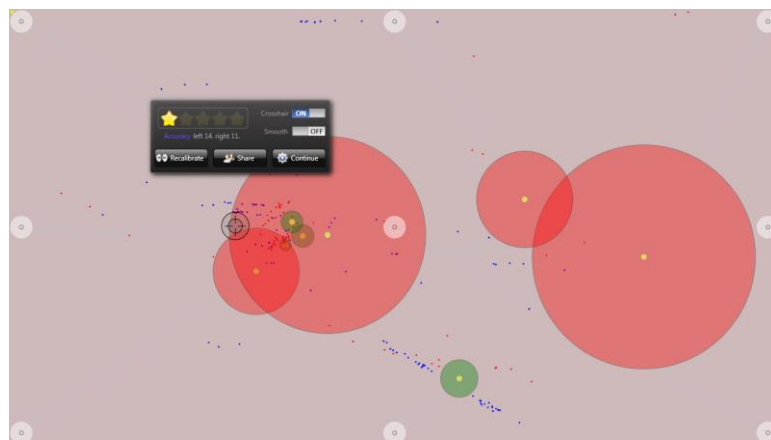


Figure 5.2 Tester B's 1-Star Calibration Result, with 14 Left, 11 Right Eye Accuracies

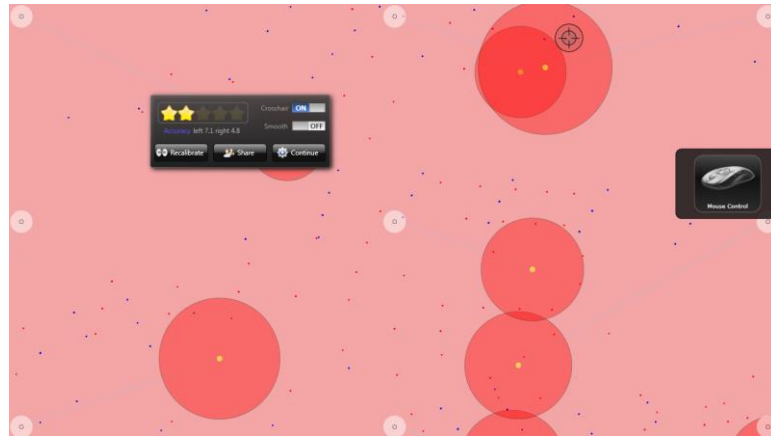


Figure 5.3 Tester C's 2-Star Calibration Result, with 7.3 Left, 4.8 Right Eye Accuracies

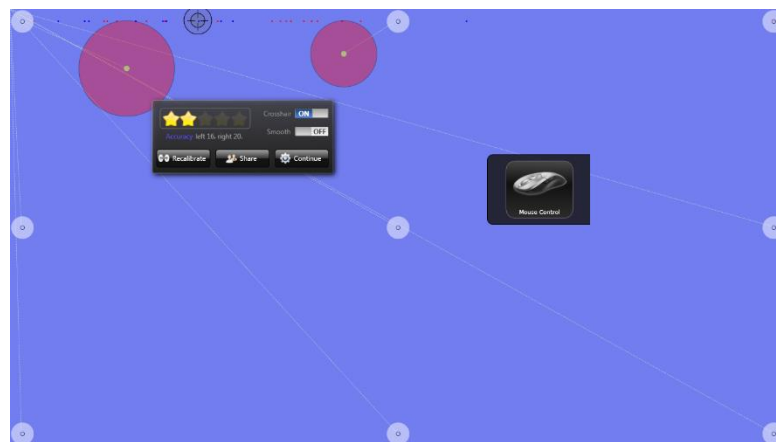


Figure 5.4 Tester D's 2-Star Calibration Result, with 16 Left, 20 Right Eye Accuracies

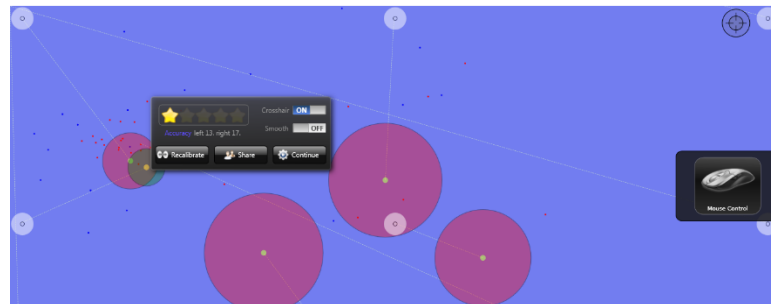


Figure 5.5 Tester E's 1-Star Calibration Result, with 13 Left, 17 Right Eye Accuracies

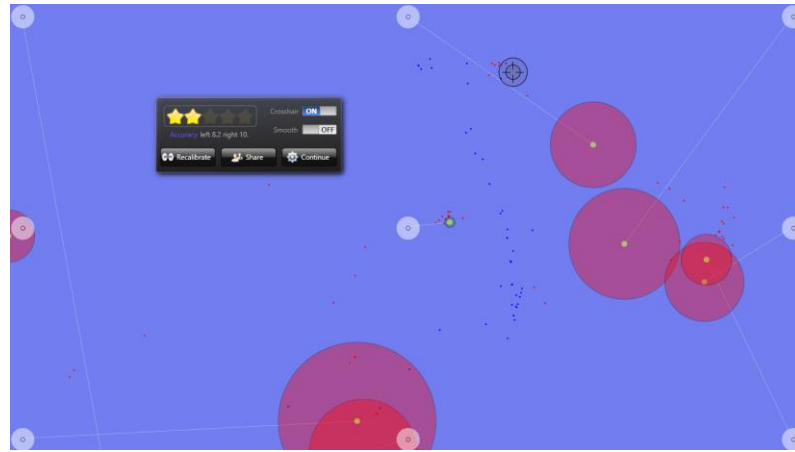


Figure 5.6 Tester F's 2-Star Calibration Result, with 8.2 Left, 10 Right Eye
Accuracies

Looking at the results of each calibration, Tester A had a consistent and accurate mapping, since the small numbers in detection meant a very high level of accuracy. In the case of Tester B, however, survey results will show that this 1-star accuracy rating is not dependent on the actual gameplay. Same goes for Tester C, which would have a difficult time moving the cursor using an ETD despite having a 2-star calibration. Another reason behind Tester C having a difficult time can be implied from the scattered offset results as shown above. This result is very different from the concentrated, almost collinear points shown by the results of Testers A and B. Testers D and E, in this case, have low accuracies because they exceed 10. For Tester F, the mapping was average compared to all previous calibrations. Using the ETD's calibration results as primary basis, then, can only be consistent if a player's facial position is consistent, as well as its detection of both eyes. More observations external to and after these calibrations will be further explained in the next chapter.

For these results to become significant, the researchers administered a survey immediately after the game to quantify the goal of gauging the impact and workability of this ETD.

5.2 Survey Analysis

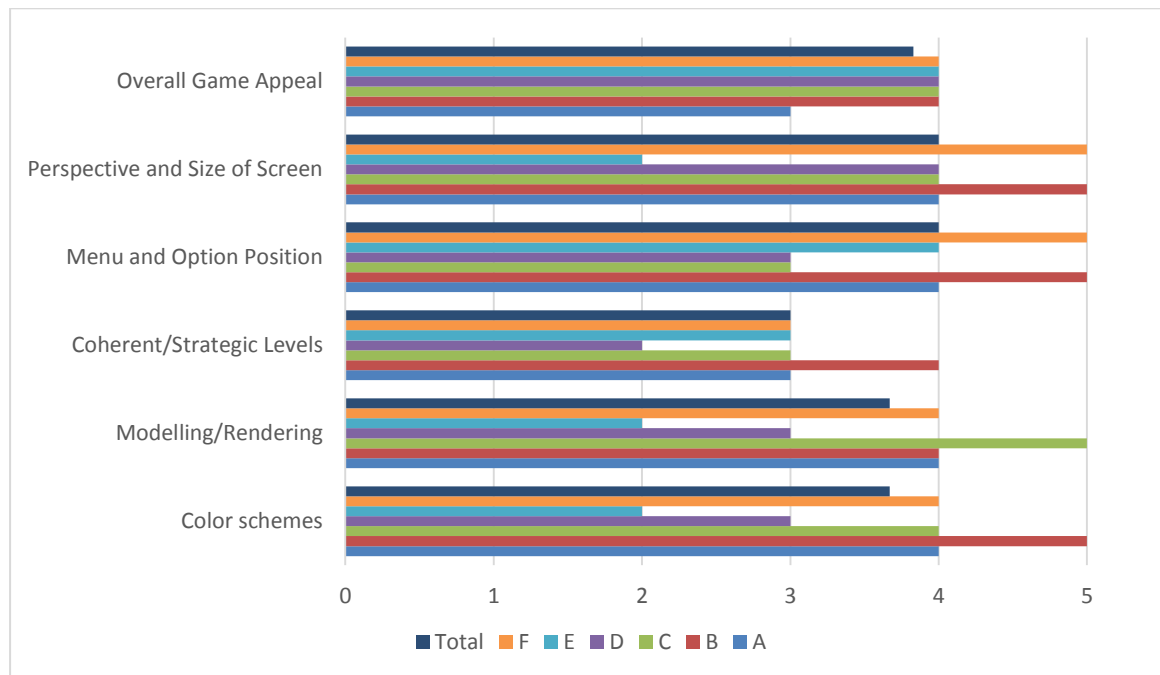


Figure 5.7 Survey Results with Appearance Criteria

Going back to the research questions, the main focus was in designing a RTT game integrated with an ETD, which have set metrics to determine the character actions in the gameplay applicable to the ETD detections, such that in the end, the output would be a quantitative analysis of the game's workability and design through a survey. As much as possible, the survey focused on eliciting results in terms of the game's appeal, its complexity through the gameplay and mechanics, and ultimately, the challenge through the use of the ETD. All three testers answered that they already have already played an RTT game, meaning they already have a preliminary background how an RTT game is to be played. The survey used a scale of 1 to 5, 5 as strongly agreeing, for all the categories. If the tester had no chance to observe, he/she has to encircle Ø, and this was not included in the computation. As far as the appearance of the game can affect eye-straining, the survey showed that the testers did not have a problem regarding the placement of the menu, options, even the color schemes. However, the testers evaluated the levels lower, since they got confused at the start of the second level, when they can control more than one character. In the end, the average overall game appearance appeal is 3.83, which is above average.



Figure 5.8 Survey Results with Gameplay and Mechanics Criteria

In terms of the gameplay and mechanics, the use of eye-tracking device greatly affected different aspects of the game, particularly character and object movement and interaction, since it requires a precise positioning of the cursor to the character being clicked. According to the suggestion for improvement, there is also an issue regarding characters getting stuck in a specific area and hence cannot interact fast enough with the eye movement. The character skill feature, like choosing the grenade option or healing option is also greatly affected because of this precise positioning. Other aspects such as replayability, ease of access to menus and options, and clear objectives scored high. Other comments ranged from adding a follow command for multiple characters, to preventing the double-click so the window will not have to exit, to the game pace being much faster with the eyes than the mouse.

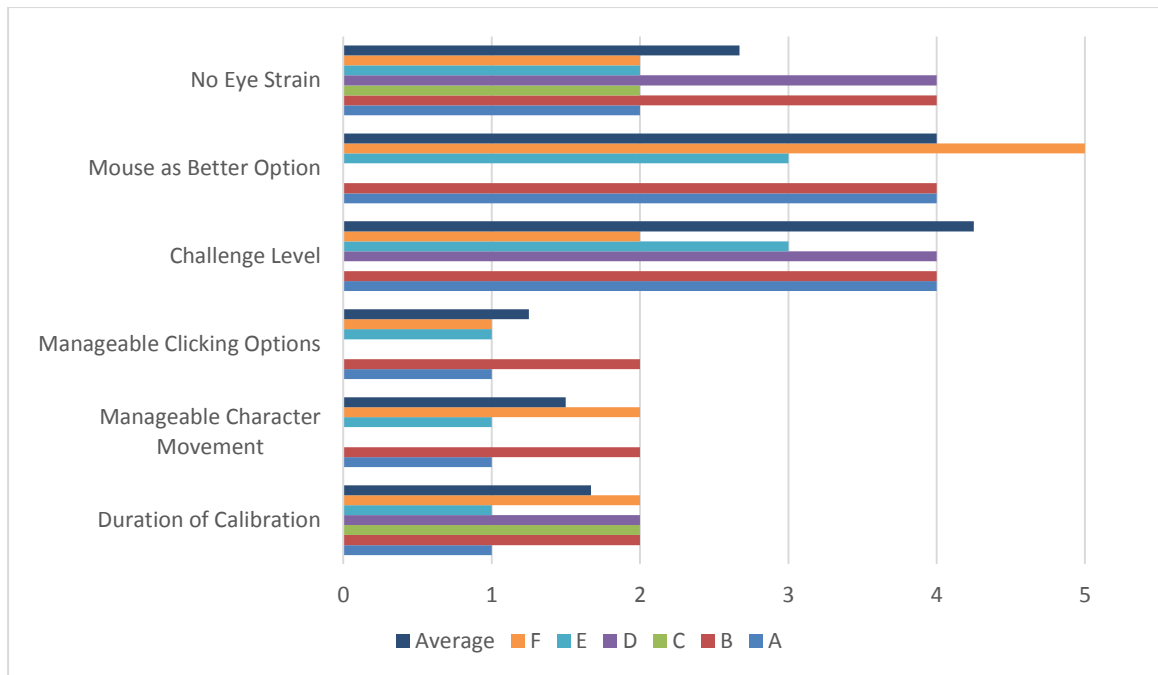


Figure 5.7 Survey Results with Use of Eye-tracking Device Criteria

Finally, the result of the aspects related to the ETD were significant, such that the contrast between their perception of using the ETD being the challenge of the game and the actual evaluation of its manageability are high. After trying out the game with the mouse instead, they still consider it a faster input device than the ETD. Tester B, even though evaluating the game as challenging, still suggests the need to improve the sensitivity to eye movement. It has also been observed that Tester B's pupils have a smaller radius, hence, it was difficult to detect the eyes.

Another aspect in considering special cases of eye detection would be Testers C and D who have deep-set eyes. They answered the last part of the quantitative survey with no chance observing most of the aspects enumerated. The eyelids, as they go down from the eyes, reduced the chance of detecting the glint and the pupil, resulting in flickering or the absence of the crosshairs and squares overlay on the webcam image before the calibration.

CHAPTER VI

CONCLUSION

6.1 Game Design

The Midas touch problem has already been addressed with different actions surrounding the character. Navigation through different parts of the map was made possible by moving the cursor to the edge of the screen. Switching characters were convenient through the options at the bottom left of the screen. With all of these taken into consideration, the game design was effective enough to yield very good results in the survey. Other factors taken into consideration such as the color schemes of the game were also noted.

The snap-clutch algorithm as another solution to the Midas touch problem was implemented through the use of the Gaze Mouse supplement of the ITU Gaze Tracker. The dwell option with a medium detection may have been the setting during the implementation, but the recommendation is inclined towards exploring the optimum setting, and an analysis on the duration of the gazes with these settings.

6.2 Mapping of Eye-tracking

As Poole and Ball [19] have already laid out, different orders of gaze patterns have certain factors to consider. For the purposes of this study, the third-order metrics of the scan-paths have been eliminated due to the A* pathfinding external library. If this had not been implemented, the player would take a longer time navigating through the game, since the shortest path to an area would not necessarily be the path taken.

Hence, the challenge for the eye-tracking metrics is to use the data regarding the gaze patterns more thoroughly, with the aim of finding new game design solutions apart from those already stated and implemented in this study. In the event that these solutions were also taken into consideration, gaming with an ETD could be affordable when the same cost-effective measure is used.

6.3 Testing and Survey

Testing the game using Schell's and Leblanc's theories may have been apt for the game design aspects, but then, different observations outside of the user-computer interaction must also be considered. The recommendation for the testing would be to find the best possible way to keep the calibration consistent, going as far as ergonomic analysis of the user, and the external environment setup. These factors may be observed in parallel with the gaming analysis.

The quality of the surveys also shows that the users are more concerned with the ETD in order to play the game well. In this light, the ETD becomes important regardless of the type of game the user plays. It is recommended, then, to try out more subgenres of computer games, and working towards the prospect of the ETD becoming a challenging yet alternative HCI. A comparative study may also be pursued in terms of the effectiveness of the ETD against the effectiveness of the mouse and/or keyboard. In this way, the time and qualitative analysis of the game would be more appropriate, but of course, keeping into mind the limitation of the ETD's affordability.

6.4 General Conclusions

The general conclusion drawn from this study focuses more on the integration between the RTT game and the cost-effective ETD. Even if the game design addresses the concerns of an RTT game or of an ETD as interface, the inconsistency of the eye tracking during the game became an important factor. The testers who did not get a chance to observe the eye-tracking element of the game cannot fully measure the effectiveness of an alternative HCI. For further development of this study, the ETD setting should be the main concern, along with other external factors that have been observed as crucial to keeping the eye-tracking consistent.

BIBLIOGRAPHY

- [1] Microsoft (2012). "Xbox 360 + Kinect". Microsoft. <http://www.xbox.com/en-US/kinect>. Retrieved 2012-05-03.
- [2] Sony (2012). "This Is How I Move™." Sony. <http://us.playstation.com/ps3/playstation-move/>. Retrieved 2012-05-03.
- [3] Nintendo (2012) "What is Wii?" <http://www.nintendo.com/wii/whatiswii>. Retrieved 2012-05-03.
- [4] Launie and Melynda Sorrels (17 November 2008). "Brain Waves are the New Input Device of the Future". ScienceRay. <http://sciencera.com/technology/applied-science/brain-waves-are-the-new-input-device-of-the-future/>. Retrieved 2012-05-03.
- [5] Loz Blain (20 May 2007). "The 3D air-mouse you wear as a ring". Gizmag. <http://www.gizmag.com/go/7293/>. Retrieved 2012-05-03.
- [6] Eye Tracking Update (12 May 2010). "Eye Controlled Video Games? Better Late Than Never". Eye Tracking Update. <http://eyetrackingupdate.com/2010/05/12/eye-controlled-video-games-late> Retrieved 2012-05-03
- [7] Eye Tracking Research and Applications (7 February 2010). "ETRA 2010 - Program - Short Papers Session 1: Eye Tracking Applications and Data Analysis". <http://etra.cs.uta.fi/program.html#short1>. Retrieved 2012-05-03.
- [8] Dan Adams (7 April 2006). "The State of the RTS". PC IGN. <http://pc.ign.com/articles/700/700747p4.html>. Retrieved 2012-04-28.
- [9] Bruce Geryk (19 May 2008). "A History of Real-Time Strategy Games: Dune II". GameSpot. http://www.gamespot.com/gamespot/features/all/real_time/index.html Retrieved 2012-04-27.
- [10] Soren Johnson (6 November 2009). "Analysis: Turn-Based Versus Real-Time". Gamasutra. http://www.gamasutra.com/php-bin/news_index.php?story=25920. Retrieved 2012-04-28.
- [11] Jakub Wojnarowicz (22 February 2001). "Editorial: What Happened to Turn-Based Games?". FiringSquad.

- <http://www.firingsquad.com/games/tbgameseditorial/default.asp> Retrieved 2012-04-27.
- [12] Unity. “How to Import Objects in Blender”.
<http://unity3d.com/support/documentation/Manual/HOWTO-ImportObjectBlender.html> Retrieved 2012-05-16.
- [13] Aron Granberg. -. “A* Pathfinding”. <http://www.arongranberg.com/unity/a-pathfinding/> Retrieved 2012-05-16.
- [14] Veronica Sundstedt. 2010. Gazing at games: using eye tracking to control virtual characters. In ACM SIGGRAPH 2010 Courses (SIGGRAPH '10). ACM, New York, NY, USA, Article 5, 160 pages.
DOI=10.1145/1837101.1837106 <http://doi.acm.org/10.1145/1837101.1837106>
- [15] Gaze Group. <http://www.gazegroup.org/> Retrieved 2012-05-16.
- [16] Cihan Topal, Ömer Nezih Gerek, and Atakan Doğan. 2008. A head-mounted sensor-based eye tracking device: eye touch system. In Proceedings of the 2008 symposium on Eye tracking research & (ETRA '08). ACM, New York, NY, USA, 87-90. DOI=10.1145/1344471.1344494
<http://doi.acm.org/10.1145/1344471.1344494>
- [17] Dongheng Li, Jason Babcock, and Derrick J. Parkhurst. 2006. openEyes: a low-cost head-mounted eye-tracking solution. In Proceedings of the 2006 symposium on Eye tracking research & applications (ETRA '06). ACM, New York, NY, USA, 95-100. DOI = 10.1145/1117309. 1117350
<http://doi.acm.org/10.1145/1117309.1117350>
- [18] Hewitt Crane and Carroll Steele. 1985. Generation-V dual-Purkinje-image eyetracker. *Applied Optics* 24 (4): 527–537. doi:10.1364/AO.24.000527
- [19] Alex Poole, Linden J. Ball. Eye Tracking in Human-Computer Interaction and Usability Research: Current Status and Future. In “Prospects”, Chapter in C. Ghaoui (Ed.): *Encyclopedia of Human-Computer Interaction*. Pennsylvania: Idea Group, Inc., 2005.
- [20] Eun Yi Kim and Se Hyun Park. 2006. “Computer interface using eye tracking for handicapped people”. In Proceedings of the 7th international conference on Intelligent Data Engineering and Automated Learning (IDEAL'06), Emilio Corchado, Hujun Yin, Vicente Botti, and Colin Fyfe (Eds.). Springer-Verlag, Berlin, Heidelberg, 562-569. DOI=10.1007/11875581_68
http://dx.doi.org/10.1007/11875581_68

- [21] Sean Coughlan (28 February 2012). "Eye-controlled computer games for disabled children.". British Broadcasting Corporation.
<http://www.bbc.co.uk/news/education-17179405>. Retrieved 2012-04-28.
- [22] Rebecca Boyle (27 November 2011). "Honduran Teen Invents Cheap, Simple Eye-Tracking Device For Disabled". <http://www.popsi.com/diy/article/2011-11/honduran-teen-invents-cheap-simple-eye-tracking-device-disabled>. Retrieved 2012-04-27.
- [23] Chern-Sheng Lin, Chia-Chin Huan, Chao-Ning Chan, Mau-Shiun Yeh, Chuang-Chien Chiu. 2004. Design of a computer game using an eye tracking device for eye's activity rehabilitation, *Optics and Lasers in Engineering*, vol. 42(1) pp.91-108 (SCI).
- [24] Poika Isokoski, Markus Joos, Oleg Spakov, and Benoit Martin. 2009. Gaze controlled games. *Univers. Access Inf. Soc.* 8, 4 (October 2009), 323-337. DOI=10.1007/s10209-009-0146-3 <http://dx.doi.org/10.1007/s10209-009-0146-3>
- [25] HeavenGames. "Overview: World in Conflict".
<http://wic.heavengames.com/gameinfo/>. Retrieved 2012-05-03.
- [26] Allen Rausch (12 December 2008). "Warhammer 40,000: Dawn of War II". GameSpy. <http://pc.gamespy.com/pc/warhammer-40000-dawn-of-war-ii/938016p1.html>. Retrieved 2012-05-03.
- [27] J. Leyba and J. Malcolm. 2004. Eye Tracking as an Aiming Device in a Computer Game. Course work (CPSC 412/612 Eye Tracking Methodology and Applications by A. Duchowski), Clemson University.
<http://andrewd.ces.clemson.edu/courses/cpsc412/fall04/teams/reports/group2.pdf>. Retrieved 2012-05-07.
- [28] Rameshsharma Ramlool, Cheryl Trepagnier, Marc Sebrechts, and Jaishree Beedasy. 2004. Gaze Data Visualization Tools: Opportunities and Challenges. In *Proceedings of the Information Visualisation, Eighth International Conference (IV '04)*. IEEE Computer Society, Washington, DC, USA, 173-180. DOI=10.1109/IV.2004.64 <http://dx.doi.org/10.1109/IV.2004.64>
- [29] Alberto De Santis and Daniela Iacoviello. 2009. Robust real time eye tracking for computer interface for disabled people. *Comput. Methods Prog. Biomed.* 96, 1 (October 2009), 1-11. DOI=10.1016/j.cmpb.2009.03.010 <http://dx.doi.org/10.1016/j.cmpb.2009.03.010>

- [30] Kohei Arai and Ronny Mardiyanto. 2011. Eye Based HCI with Moving Keyboard for Reducing Fatigue Effects. In *Proceedings of the 2011 Eighth International Conference on Information Technology: New Generations (ITNG '11)*. IEEE Computer Society, Washington, DC, USA, 417-422. DOI=10.1109/ITNG.2011.80 <http://dx.doi.org/10.1109/ITNG.2011.80>
- [31] Alex Champandard. 2003. AI Game Development: Synthetic Creatures with Learning and Reactive Behaviors. New Riders Games, Berkeley.
- [32] Ian Millington. 2006. Artificial Intelligence for Games. Elsevier, San Francisco.
- [33] Brian Schwab. 2004. AI Game Engine Programming. Charles River Media, Inc., Massachusetts. pp. 97-105.
- [34] Merriam-Webster. *Strategy*. <http://www.merriam-webster.com/dictionary/strategy>. Retrieved 2012-05-08.
- [35] Merriam-Webster. *Tactics*. <http://www.merriam-webster.com/dictionary/tactics>. Retrieved 2012-05-08.
- [36] Gamespot. 2007. "World in Conflict Screens for PC" <http://asia.gamespot.com/world-in-conflict/images/620094/>. Retrieved 2012-05-16.
- [37] Gamespot. 2009. "Warhammer 40,000: Dawn of War II Screens for PC" <http://asia.gamespot.com/warhammer-40-000-dawn-of-war/images/353205/>. Retrieved 2012-05-16.
- [38] J.P. Hansen and M. Stovring. "Udfordringer er Blikfang." Hrymfaxe 3 (1988): 28-31 in ed. Arne John Glenstrup and Theo Engell-Nielsen. Eye-Controlled Media: Present and Future State, University of Copenhagen Institute of Computer Science, 1995. <http://www.diku.dk/hjemmesider/ansatte/panic/eyegaze/node27.html>. Retrieved 2012-05-08.
- [39] Howell Istance, Richard Bates, Aulikki Hyrskykari, and Stephen Vickers. 2008. Snap clutch, a moded approach to solving the Midas touch problem. In Proceedings of the 2008 symposium on Eye tracking research & applications (ETRA '08). ACM, New York, NY, USA, 221-228. DOI=10.1145/1344471.1344523 <http://doi.acm.org/10.1145/1344471.1344523>
- [40] Cleo D. Redline and Christopher P. Lankford (2001), Eye-movement analysis: a new tool for evaluating the design of visually administered instruments. In

Proceedings of the Section on Survey Research Methods of the American Statistical Association.

- [41] Schell, Jesse. 2008. *The Art of Game Design – A Book of Lenses*. Morgan Kaufman Publishers, Burlington.
- [42] IT University of Copenhagen. 2010. ITU Gaze Tracker User Manual. http://www.gazegroup.org/software/GT_Users_Guide.pdf . Retrieved 2012-05-16.
- [43] Shane Glaun (8 November 2011). “Tobii EyeAsteroids arcade game has no joystick, uses eye control” Slash Gear. <http://www.slashgear.com/tobii-eyeaasteroids-arcade-game-has-no-joystick-uses-eye-control-08193922/>. Retrieved 2012-05-16.
- [44] Lars Marcus Vedeler and Theo Tveterås (27 March 2010). “Eye-Tracking Game: Summary”. Vimeo. <http://vimeo.com/10486276>. Retrieved 2012-05-16.
- [45] Unity <http://unity3d.com/unity/engine/programming> Retrieved 2012-05-16.
- [46] MonoDevelop. (2012). MonoDevelop 3.0. <http://monodevelop.com>. Retrieved 2012-05-16.
- [47] Gaze Group (2010). “Gaze Tracker”. <http://www.gazegroup.org/downloads/23-gazetracker>. Retrieved 2012-05-16.
- [48] Blender. “Home”. <http://www.blender.org/>. Retrieved 2012-05-16.
- [49] Emilie Møllenbach, Martin Lillholm, Alastair Gail, and John Paulin Hansen. 2010. “Single Gaze Gestures”.In *COGAIN European Network of Excellence*.
- [50] Oleg Spakov and Darius Miniotas. 2007. “Application of Eye-Tracking Algorithms in Eyes Gaze Visualizations”. In *Information Technology and Control*.

APPENDIX A

SURVEY RESULTS

I. Appearance

CRITERIA	A	B	C	D	E	F	Average
Color schemes	4	5	4	3	2	4	3.67
Modelling/Rendering	4	4	5	3	2	4	3.67
Coherent/Strategic Levels	3	4	3	2	3	3	3
Menu and Option Position	4	5	3	3	4	5	4
Perspective and Size of Screen	4	5	4	4	2	5	4
Overall Game Appeal	3	4	4	4	4	4	3.83

II. Gameplay and Mechanics

CRITERIA	A	B	C	D	E	F	Average
Understandable Menus	3	5	4	5	5	5	4.5
Character-Object Movement and Interaction	2	4	2	3	2	3	2.67
Clear and Easy Objectives	4	4	4	4	4	5	4.17
Well-paced Game (considering eye-tracking)	3	3	4	5	2	3	3.33
Helpful Skills in Level Progress	3	3	3	3	2	4	3
Replayability	4	5	4	3	4	4	4

III. Use of Eye-tracking Device

CRITERIA	A	B	C	D	E	F	Average
Duration of Calibration	1	2	2	2	1	2	1.67
Manageable Character Movement with ETD	1	2	Ø	Ø	1	2	1.5
Manageable Clicking Options with ETD	1	2	Ø	Ø	1	1	1.25
Challenge Level	4	4	Ø	4	3	2	4.25
Mouse as Better Option	4	4	Ø	Ø	3	5	4
No Eye Strain During The Game	2	4	2	4	2	2	2.67