Author: Michael Edgar

Supervisor: Robert Sheehy

Second Reader: Paul Collins

An Evaluation of the use of an adaptive controller with an eye tracker to facilitate accessibility in gaming

B.Sc. (Hons) in Computing with Games Development

Table of Contents

[1 Abstract 2](#_Toc40720068)

[2 Introduction 4](#_Toc40720069)

[3 Video Games and Accessibility 5](#_Toc40720070)

[3.1 Introduction 5](#_Toc40720071)

[3.2 Video Games 5](#_Toc40720072)

[3.3 Accessibility and Technology 7](#_Toc40720073)

[4 Case Studies 8](#_Toc40720074)

[4.1 Case Study 1: Xbox Adaptive Controller 8](#_Toc40720075)

[4.2 Case Study 2: Logitech G Adaptive Gaming Kit 8](#_Toc40720076)

[4.3 Case Study 3: Tobii Eye Tracker 4C 8](#_Toc40720077)

[4.4 Case Study 4: Google Stadia 9](#_Toc40720078)

[5 Methodology 9](#_Toc40720079)

[5.1 Research Undertaken 9](#_Toc40720080)

[5.2 Research Question 10](#_Toc40720081)

[5.3 Proposed Project Implementation 10](#_Toc40720082)

[5.4 Prototype 10](#_Toc40720083)

[6 Sprints 13](#_Toc40720084)

[6.1 Voice and Eye control in Windows 13](#_Toc40720085)

[6.2 Raspberry Pi Research 13](#_Toc40720086)

[6.3 Python/C# Eye Tracker Implementation 14](#_Toc40720087)

[6.4 VoiceBot Speech Recognition 16](#_Toc40720088)

[6.5 Eye Tracker Mouse Movement 17](#_Toc40720089)

[7 References 20](#_Toc40720090)

# Abstract

Video games are a form of interactive media that involve a player interacting with an artificial world on a computer. Accessibility is a measure of the availability of some good or service to people. Accessibility can be linked with people that have disabilities. In video games, accessibility can relate to the design of the controller, the use of flashing images, the inclusion or exclusion of subtitles. Technology has provided several accessibility options to facilitate accessibility in video games. These include options such as voice to text conversions and vice versa, colour and high contrast filters and the ability to adjust the icons or text as desired.

This study evaluated several hardware devices working together with software applications in order to improve the experience of players with less motor function. The study evaluated devices specifically created to solve an accessibility issue, as well as devices that were not specifically created to solve an accessibility issue but may be used in this respect. The study also examined how hardware devices may be paired with software to solve accessibility issues, such as the use of an eye tracker with voice controls to improve the experience of players with less motor function.

The study found that the issue of accessibility in relation to video games has been an issue since for a number of decades and there has been many attempts to ensure video games were more accessible. However, the solutions were often inaccessible due to cost or lack of public knowledge. Recent solutions that have been examined in this study were found to be more accessible, with a greater focus on ensuring public knowledge in relation to the proposed solutions. The issue of cost however was found to be still prevalent, in particular in the case of hardware devices, as they were found to cost upwards of a hundred US dollars, making it inaccessible to lower income players that required these accessibility supports.

The final part of the study attempted to use existing technologies to improve the quality of video games for players with less motor function using a number of hardware devices and software applications. This part of the study outlines the steps taken in achieving this goal and outlining the challenges faced during the implementation. The implementation made use of the Tobii Eye Tracker 4C and the Xbox Adaptive Controller, along with the VoiceBot software application and C# programming scripts created by the author. To test this solution, the study attempted to implement the solution within the online video game Rocket League.

# Introduction

This study aims to outline accessibility issues in video games, with a focus on previous attempts at improving accessibility within video games and an evaluation of their success or failure. To achieve this, an explanation of both video games and accessibility will be provided. Following from this, the study will outline the use of technology to improve accessibility, providing examples of their use. This will then be followed by an inspection of accessibility as it relates to video games; with a focus on the issues that exist and attempts to resolve them. Finally, the study will examine an attempt by the author to utilise existing technologies to create a controller setup for use in video games that aims to improve the experience of players with less motor function.

# Video Games and Accessibility

## Introduction

Video games are a form of interactive media that involve a player interacting with an artificial world on a computer. Accessibility is defined as how easy something is to reach, enter, use, see, etc. (Oxford University Press, 2019). Accessibility can be linked with people that have disabilities. To ensure accessibility for as many people as possible, many countries have laws in place to prevent the exclusion of those with disabilities.

In video games, accessibility can relate to the design of the controller, the use of flashing images, the inclusion or exclusion of subtitles, etc. Accessibility is something that is often overlooked in the design of video games. An example of where this is the case is referred to by Kevin Bierre et al. They present the example of a young boy with cerebral palsy who requires a specific adapted controller to play video games, but the boy in question struggles to find suitable video games as accessibility information is difficult to find (Bierre, et al., 2019).

## Video Games

A video game can be defined by the goal(s), the rules and the environment of the game. Video games are often categorised into various genres such as puzzle, first-person shooter, couch co-op, etc.

The first known video game was implemented by Josef Kates and was called ‘Bertie the Brain’ (Smith, 2014). Kates, who at the time was a consulting engineer in Toronto 1950, made the game as an exhibit for attendees of the 1950 Canadian National Exhibit (Simmons, 1975). The game was a simple Tic Tac Toe game in which the player would attempt to beat the computer by getting three consecutive O’s in a horizontal, vertical or diagonal line. This game was created to demonstrate power of computers (Smith, 2014).

A person standing in front of a store

Description automatically generated

Figure 2.. Life magazine photo of Danny Kaye in front of Bertie the Brain at the Canadian National Exhibit in 1950 (Hoffman, 1950)

There were many games that followed Bertie the Brain, but the first video game created for entertainment purposes was ‘Tennis for Two’ (Smith, 2014). William A. Higinbotham created the game in 1958 for the Brookhaven National Laboratory (BNL) annual ‘Visitors Day’ to showcase the current research and development projects in the laboratory. Similar to Bertie the Brain, this game was purely an exhibition piece and was not released to the public. Tennis for Two ended up being dismantled after the 1959 Visitors Day (Nitray, 2015).

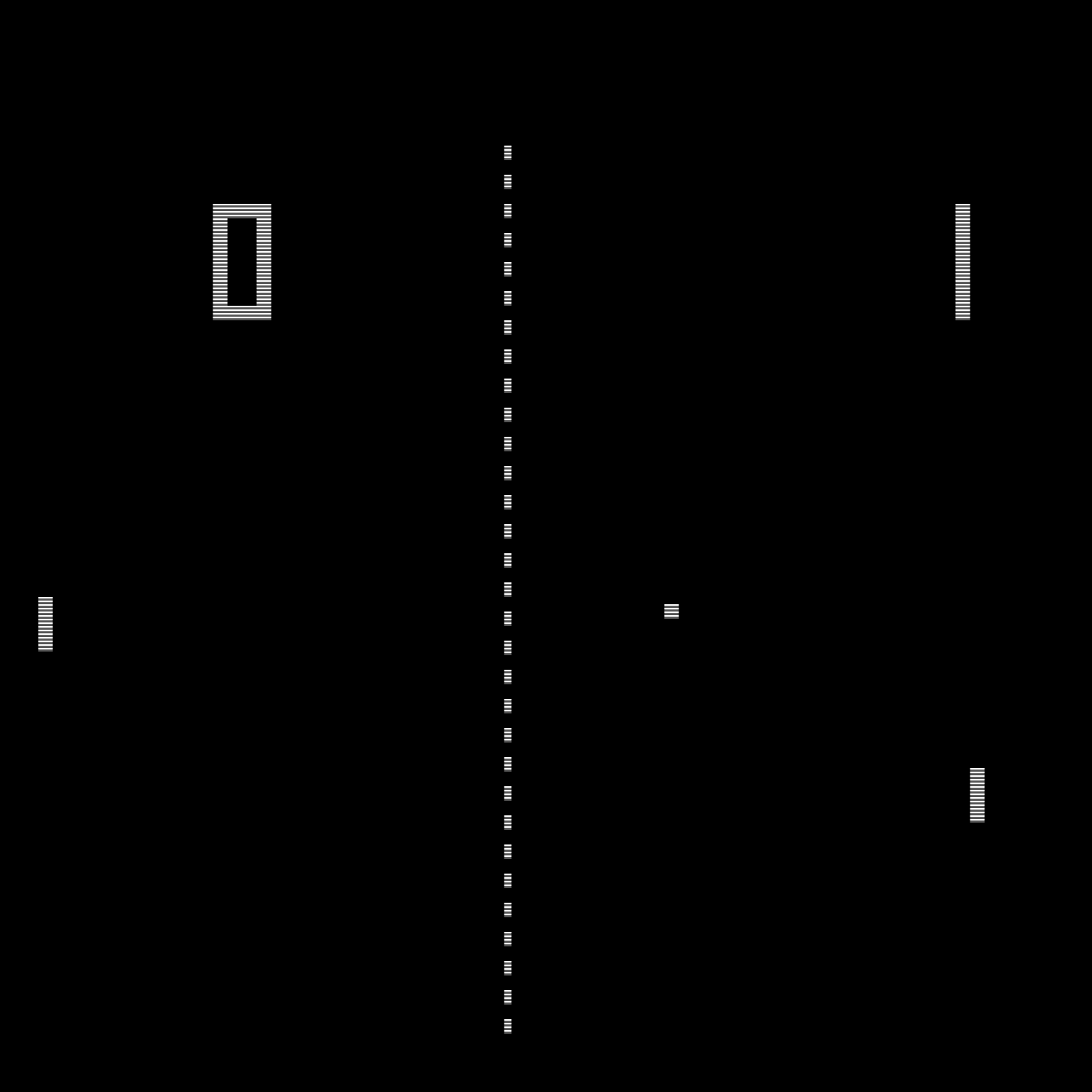
One of the first video games to be publicly released was Pong; based on the popular sport Ping Pong. The game consisted of two player-controlled panels and a ball that would travel back and forth between the two panels. The objective of the game was to make the ball object move passed the other players’ panel. It was released in 1972 for arcades (Wolf, 2007). An example of Pong is shown in figure 2.2 below.

Figure 2..: Pong Game

## Accessibility and Technology

Technology has shown a significant growth in the last thirteen years with the introduction of many “smart” devices such as the smart phone, smart television, and smart watch. With this growth, the desire to cater to accessibility needs has been highlighted. To cater to these needs, a number of accessibility options were introduced to various pieces of technology. This includes options such as voice to text conversions and vice versa, colour and high contrast filters and the ability to adjust the size of icons or text as desired. All of these options were introduced to help make technology more accessible to the general public.

The European Parliament and the Council of the European Union released a directive titled “Directive (EU) 2016/2102 of the European Parliament and of the Council of 26 October 2016 on the accessibility of the websites and mobile applications of public sector bodies”. This directive states that any website or app owned by a public body must be accessible to any person with a disability (National Disability Authority, 2017).

For as long as video games have existed, the issue of accessibility has been present for the people who play them (Wilds, 2020). Video games presented accessibility issues not only in the hardware configuration, but also in the software setup itself. One example of this would be Super Mario Bros from 1985, which lacked subtitles or audio control options; presenting issues for people who are deaf or hard of hearing. To combat these issues, solutions have been created at different levels of the development of video games. Many companies have accessibility options built into the games themselves. One of the first examples of this is the video game Real Sound: Kaze no Regret by Warp Inc. released in 1997 for the Sega Saturn video game console. What made this game unique is that it was marketed entirely towards blind players. The game relied entirely on the audio to guide players. Other companies add accessibility options as updates to existing games. This was the case for the game Days Gone by Bend Studio for the PlayStation 4 video game console, which was released in April 2019, which added multiple accessibility options in an update. This included the ability to replace repeated button presses with a button hold (Bend Studio, 2019). This option allows players that may have difficulty repeatedly pressing a button an easier alternative. Some users modify the games themselves to make the games more accessible.

# Case Studies

## Case Study 1: Xbox Adaptive Controller

The Microsoft Xbox Adaptive Controller is a hardware device released in September 2018 that can be used to control video games on both the Xbox One and Windows PC. The controller was designed to cater for people who play video games with limited mobility. The controller meets accessibility needs with two main features: its configurability and the Co-pilot mode. The controller consists of 2 large programmable buttons, a D-pad, Xbox, View, Profile and Menu buttons, 2 USB ports, a 3.5mm headset jack, USB-C and DC power ports, a connect button and 19 3.5mm jacks for external inputs. The USB ports can be used to map the left and right analog stick inputs respectively from an external device, such as a joystick. The 19 3.5mm jacks correspond to each of the button inputs on a regular Xbox One controller; not including the movement of an analog stick. However, the controller does contain 2 extra inputs that can be mapped to any input of the controller including an analog stick direction input. The Xbox Accessories application can be used to modify inputs, while allowing for multiple configurations. The other main accessibility feature of the controller is the Co-pilot mode. This mode allows a player to connect a second Xbox One controller that would work alongside the Xbox Adaptive controller. This allows a person to use any of the inputs from a regular Xbox One controller while using the Adaptive controller; as well as allowing a second person to assist someone using the Xbox Adaptive controller.

## Case Study 2: Logitech G Adaptive Gaming Kit

The Logitech G Adaptive Gaming Kit is a set of buttons and triggers; intended for use alongside the Xbox Adaptive controller. The kit contains 4 light touch buttons, 2 variable triggers, 3 small buttons and 3 large buttons. Each of these buttons and triggers can be individually connected to an Xbox Adaptive Controller to create individual controller setups that suit the user. The kit also contains a number of mats and stickers to increase ease of use and allow the user to recognise each button/trigger more easily. The Adaptive Gaming Kit was released in November 2019 (Perez, 2019).

## Case Study 3: Tobii Eye Tracker 4C

The Tobii Eye Tracker 4C is an eye tracking device released in October 2016 (Tobii Tech, 2016). The eye tracker can be used to control a number of games that support the eye tracker, as well as for general navigation in the Windows operating system. The eye tracker is marketed towards people who desire additional functionality during video games. They advertise the ability to “look around” within a game using the motion of the player’s head, along with a number of other features that aim to improve the video game experience. Alongside its ability to be used as a peripheral for video games, Microsoft has added accessibility supports that allow eye tracking to be used in Windows 10 with the Tobii Eye Tracker 4C (Microsoft Accessibility Blog, 2017).

The eye tracker connects to any PC or laptop using a USB 2.0 connection and proprietary eye tracking software, Tobii Eye Tracking. The Tobii Eye Tracking software can be used to calibrate the eye tracker and change a number of settings. Once connected and calibrated, the Tobii Eye Tracker 4C requires no further setup to be used for Windows navigation, Windows Eye Control or for use in video games.

## Case Study 4: Google Stadia

In 2019, Google released the Google Stadia, a new technology that promised a full gaming experience that one might expect from a traditional console or ‘gaming PC’, played directly from a smartphone or tv. The Stadia requires the Stadia controller and either a Google Chromecast or an Android smartphone that supported the Stadia application.

# Functional Specification

## User Stories

### Eye Tracking Game Control

**Description:**

As a player, I need the ability to have my eye movements tracked so that I can control the game.

**Priority:** Must Have

**Estimated Completion Time:** 7hr

**Acceptance Criteria:**

The eye movements of the player are received by the eye tracker.

The data containing the eye movements is sent from the eye tracker to a script.

The script receives the data containing the eye movements and sends it to the game as an input.

The game receives this input and uses it to affect the game in a meaningful and noticeable manner.

### Voice Control Game Control

**Description:**

As a player, I need the ability to use voice commands so that I can control the game.

**Priority:** Must Have

**Estimated Completion Time:** 2hr

**Acceptance Criteria:**

The microphone of the device running the game receives audio from the user.

The user audio data is sent from the microphone to the software controlling voice commands.

The software controlling voice commands checks if the user audio contains a voice command registered for the game.

The software controlling voice commands sends an input matching the given voice command to the game.

The game receives this input and uses it to affect the game in a meaningful and noticeable manner.

### Adaptive Controller Game Control

**Description:**

As a player, I need the ability to use the Xbox Adaptive Controller so that I can control the game.

**Priority:** Must Have

**Estimated Completion Time:** 1hr

**Acceptance Criteria:**

The Xbox Adaptive Controller connects to the device running the game.

The Xbox Adaptive Controller receives inputs from the player.

The player inputs are sent from the Xbox Adaptive Controller to the game.

The game receives the player inputs and uses them to affect the game in a meaningful and noticeable manner.

### Eye Tracking Cursor Control

**Description:**

As a user, I need to be able to control the device’s cursor with the eye tracker, so that it can navigate the operating system without a mouse.

**Priority:** Should Have

**Estimated Completion Time:** 4hr

**Acceptance Criteria:**

The eye movements of the player are received by the eye tracker.

The data containing the eye movements is sent from the eye tracker to a script.

The script receives the data containing the eye movements and uses it to modify the cursor’s position.

### Multiple Voice Command Profiles for Different Games

**Description:**

As a player, I need the ability to have separate profiles of voice commands for different games so that I can control multiple games with voice commands.

**Priority:** Should Have

**Estimated Completion Time:** 1hr

**Acceptance Criteria:**

The software controlling the voice commands has multiple profiles for different games, containing unique voice commands.

The voice commands from a profile, do not affect a game outside of the game’s profile.

The profiles can each send the same input for their given game.

## Use Case Diagrams

# Methodology

## Research Undertaken

The research undertaken for this project centred around accessibility issues and previous attempts to overcome these issues. Attempts around technology in general and the world of video games to overcome accessibility issues were researched; both to examine the accessibility issue posed and the success of the proposed solution. These solutions were grouped by their proposed usage: those aimed at improving general technology use and those aimed specifically for use with video games.

In relation to video game-based solutions, the source of the solution was also examined. These solutions fell into three categories in that respect: those created by a company to act as a peripheral for their existing video game console/video game, those created by a company to work in conjunction with another company’s existing video game console/video game and those created by a user to facilitate their use of a video game console/video game.

The last topic researched was the use of existing peripherals for video games as a means of solving an accessibility issue. This aimed to identify existing resources and how they can be applied to alternative situations. An example of this is the Tobii Eye Tracker 4C, a consumer eye tracking device that is marketed for use in video games, but can be used to enable Windows’ Eye Control; which allows users to control their PC or laptop with their eyes alone.

## Research Question

Can eye tracking and voice control be implemented alongside existing accessibility peripherals for use in a video gaming context. An evaluation of existing technologies, how they may work alongside each other and how beneficial they may be to players with less motor function ability.

## Proposed Project Implementation

This project aims to use a number of hardware devices working together with software applications in order to improve the experience of players with less motor function ability. This will also require software that communicates with the hardware devices and translates their inputs into inputs a video game may understand.

## Prototype

|  |  |  |
| --- | --- | --- |
| **Prototype Number** | **Start Date** | **Finish Date** |
| 1 | 04/12/2019 | 09/12/2019 |

|  |  |  |
| --- | --- | --- |
| **Task Number** | **Details** | **Status** |
| 1 | Install Unity Hub | Complete |
| 2 | Install Unity 2019.2.6f1 | Complete |
| 3 | Install Rider 2019 IDE | Complete |
| 4 | Open Unity Hub | Complete |
| 5 | Create new 3D Unity Project | Complete |
| 6 | Download Tobii SDK from Unity Asset Store | Complete |
| 7 | Import Tobii SDK from Unity Asset Store | Complete |
| 8 | In Tools click “Add Tobii SDK Demo Scenes to Build” | Complete |
| 9 | Run Sample Scenes to learn about the Tobii SDK | Complete |
| 10 | Create “Scripts” and “Prefabs” folders | Complete |
| 11 | Create “AsteroidController”, “EyeTracking”, “GameManager” & “PlayerController” Scripts | Complete |
| 12 | Read documentation at <https://tobii.github.io/UnitySDK/scripting-api> | Complete |
| 13 | Populate Scripts with code shown in Figures 4.1 – 4.6 | Complete |
| 14 | Create 3D Sphere with the properties shown in Figure 4.7 and save as a prefab named “Asteroid” | Complete |
| 15 | Make the default Directional Light a child of the default camera object. Rename the camera to Player and save as a prefab with the properties shown in figure 4.8 | Complete |
| 16 | Create a UI Text object named “Exit Text” | Complete |
| 17 | Move the Exit Text Anchor and Position to Bottom, Left and change the Text to “Press ‘Esc’ to Quit” | Complete |
| 18 | Create an UI Image object name “Current Eye Position” | Complete |
| 19 | Populate the Current Eye Position with the components shown in Figure 4.8 | Complete |
| 20 | Run the Game and test it works | Complete |

EyeTracking Class

Used to track the movement of the player's eyes

Figure 4..: EyeTracking Class

Using the Tobii Eye Tracker, along with Microsoft Services Voice, a game can be controlled without any physical input from the player, e.g. button presses. This project provided a clear use case for the eye tracker in video games; particularly how the eye tracker can be used without the need of a physical input device. The benefit of this is the ability of players without the ability to move their arms can be given the opportunity to play video games. This shaped the rationale for the rest of this project and provided the idea for the Eye Tracking Windows Control shown below.

# Implementation

## Voice and Eye control in Windows

|  |  |  |
| --- | --- | --- |
| **Sprint Number** | **Start Date** | **Finish Date** |
| 1 | 02/02/2020 | 07/02/2020 |

|  |  |  |
| --- | --- | --- |
| **Task Number** | **Details** | **Status** |
| 1 | Install Tobii Eye Tracking Application | Complete |
| 2 | Configure Eye Tracker | Complete |
| 3 | Enable Windows Eye Control | Complete |
| 4 | Experiment with Windows Eye Control to learn functions and possible uses within project | Complete |
| 5 | Enable Windows Speech Recognition | Complete |
| 6 | Experiment with Windows Speech Recognition to learn functions and possible uses within project | Complete |
| 7 | Configure Xbox Adaptive Controller | Complete |
| 8 | Connect Tobii Eye Tracker to Xbox Adaptive Controller and test functionality | See Notes |

In this sprint I was attempting to evaluate the current voice and eye tracking functionality within Windows 10, along with the functionality of the Xbox Adaptive Controller. This was to help shape where I would need to progress with the project in terms of what functionality I could incorporate and what functionality I would have to create myself. I learned that the Xbox Adaptive Controller could not directly process the inputs from the Tobii Eye Tracker. To solve this issue, I decided to process the inputs from the Tobii Eye Tracker and use an external device to feed these inputs directly to the Xbox Adaptive Controller. From discussions with my FYP supervisor, we decided to research the Raspberry Pi as a possible solution for this issue. The Raspberry Pi would process the inputs from the Tobii Eye Tracker and then feed those inputs into the relevant input within the Xbox Adaptive Controller, in a manner similar to a switchboard.

## Raspberry Pi Research

|  |  |  |
| --- | --- | --- |
| **Sprint Number** | **Start Date** | **Finish Date** |
| 2 | 11/02/2020 | 21/02/2020 |

|  |  |  |
| --- | --- | --- |
| **Task Number** | **Details** | **Status** |
| 1 | Research Raspberry Pi and potential sources within the college of obtaining one | Complete |
| 2 | Research local companies that work with Raspberry Pis | Complete |
| 3 | Contact IMaR to arrange meeting to discuss FYP and Raspberry Pi integration | Complete |
| 4 | Meet with ImaR Strand Leader of RFID & Internet of Things | Complete |

This sprint was focused on the possible implementation of a Raspberry Pi for processing the inputs from the Tobii Eye Tracker for use in a video game that wouldn’t support it natively. After many discussions, I came to the conclusion that the Raspberry Pi would only be necessary were I to be introducing this setup on an environment other than PC, e.g. Xbox One Console. For this reason, I began to research alternative implementations. From the meeting with IMaR’s Strand Leader of RFID & Internet of Things, we discovered a simple python script that would run from the command line and process the data from the Tobii Eye Tracker. This would form the basis of the next sprint.

## Python/C# Eye Tracker Implementation

|  |  |  |
| --- | --- | --- |
| **Sprint Number** | **Start Date** | **Finish Date** |
| 3 | 09/03/2020 | 21/03/2020 |

|  |  |  |
| --- | --- | --- |
| **Task Number** | **Details** | **Status** |
| 1 | Fork Tobii Eye Tracker project from GitHub (sajidbaloch, 2018) | Complete |
| 2 | Install GitHub Desktop | Complete |
| 3 | Clone Tobii Eye Tracker project | Complete |
| 4 | Run Python Script from Command Line | Completed with Errors |
| 5 | Troubleshoot Python Script Error | Completed |
| 6 | Research C# Tobii Implementation | Completed |
| 7 | Create C# Console Application in Rider | Completed |
| 8 | Import Tobii Interaction NuGet package | Completed with Errors |
| 9 | Troubleshoot NuGet import issue\* | Completed |
| 10 | Install Visual Studio 2019 | Completed |
| 11 | Create C# Console Application (.Net Core) in Visual Studio | Completed |
| 12 | Import Tobii Interaction NuGet package | Completed with Errors |
| 13 | Troubleshoot NuGet import issue (  <https://stackoverflow.com/questions/38714898/visual-studio-2015-cant-find-nuget-package-references-and-dlls-in-packages-fold>  <https://docs.microsoft.com/en-us/nuget/quickstart/install-and-use-a-package-in-visual-studio>  ) | Completed |
| 14 | Create C# Console Application (.Net Framework) in Visual Studio | Completed |
| 15 | Import Tobii Interaction NuGet package | Completed |
| 16 | Complete following tutorial for a Console Application: <https://developer.tobii.com/consumer-eye-trackers/core-sdk/getting-started/> | Completed |
| 17 | Run application to ensure it works | Completed |

A picture containing computer

Description automatically generatedThis sprint focused on the task of parsing the data from the Tobii Eye Tracker in its simplest form. This means parsing the X and Y coordinates on the screen where the Eye is looking. At first, I attempted to do this using the Python script I obtained in the previous sprint. However, this proved to be an unsuitable approach as, after troubleshooting the issue with task number 8, I found out that in order to parse the data from the eye tracker I was using, I would require a special license that I would need to pay to use ([Tobii], 2018). From this, I learned that a similar approach that would not require the license could be implemented, requiring me to use C# instead of Python. This caused a further issue when attempting to import the required NuGet package. Neither Rider nor Visual Studio seemed to recognise the package after it was imported. I learned that this was caused by the project template I had used. Originally, I was using a .NET Core Console Application when I required a .NET Framework Console Application to use NuGet packages. Once I overcame this issue, I worked through a simple tutorial to retrieve the required data from the Tobii Eye Tracker.

Figure 3.1 Tobii Eye Tracker Gaze Data Coordinates

\*Note: troubleshooting from the following websites:

<https://www.jetbrains.com/help/resharper/Finding_Exploring_and_Installing_NuGet_Packages.html>

<https://www.jetbrains.com/help/rider/Reference_Windows_NuGet.html>

<https://stackoverflow.com/questions/3304741/getting-type-or-namespace-name-could-not-be-found-but-everything-seems-ok>

## VoiceBot Speech Recognition

|  |  |  |
| --- | --- | --- |
| **Sprint Number** | **Start Date** | **Finish Date** |
| 4 | 23/03/2020 | 27/03/2020 |

|  |  |  |
| --- | --- | --- |
| **Task Number** | **Details** | **Status** |
| 1 | Go to <https://www.voicebot.net/Download/> and download VoiceBot Installer. | Complete |
| 2 | Install VoiceBot by running installer and follow steps | Completed |
| 3 | Open VoiceBot and choose download profile | Completed |
| 4 | Download Rocket League Profile | Completed |
| 5 | Test Rocket League Profile in game | Completed |
| 6 | Click Edit Profile and Add button in dialog box | Completed |
| 7 | Name command “Boost” and ensure Use name as command checkbox is set to true. | Completed |
| 8 | Click Add Group button and name group “New” | Completed |
| 9 | Click Add, then Press Mouse Button and then open the dropdown menu and choose Left Mouse Click and then OK | Completed |
| 10 | Press OK again and test new command in game | Completed |
| 11 | Ensure profile runs on correct application only | Completed |

In this sprint, I investigated the VoiceBot application for it’s potential use within this project. For licensing reasons, I had to first contact the owners of the application and get permission for its use. After this, I tested a profile for the game Rocket League. I found that, after a small adjustment period, the application was perfectly suited for this project. I could set up an individual profile for any game I played, which would hijack the controls of the game based on the voice commands I set.

## Eye Tracker Mouse Movement

|  |  |  |
| --- | --- | --- |
| **Sprint Number** | **Start Date** | **Finish Date** |
| 5 | 12/05/2020 | 21/05/2020 |

|  |  |  |
| --- | --- | --- |
| **Task Number** | **Details** | **Status** |
| 1 | Retrieve eye tracking script from previous script | Complete |
| 2 | Modify script to move cursor based on GazePoint position | Completed with Errors |
| 3 | Troubleshoot GazePoint data inaccuracy and “jumping” | Completed |
| 4 | Modify script to move cursor based on EyePosition | Completed with Errors |
| 5 | Troubleshoot EyePosition data inaccuracy | Completed |
| 6 | Modify script to move cursor based on HeadDirection | Completed with Errors |
| 7 | Trouble HeadDirection missing data | Completed |
| 8 | Modify script to move cursor based on filtered GazePoint position | Completed with Errors |
| 9 | Test script within Rocket League game | Failed |

In this sprint, I attempted to use the script created in Sprint Number 3 to move the mouse cursor. This would then allow the user to replace mouse controls in game with movements recorded by the eye tracker. I decided to keep the GazePointStream from the previous implementation and change it to override the cursor position based on the position of the gazes it recorded. However, this implementation had two main issues; firstly, it was not accurate, i.e. the position it was moving the cursor to was not the position of the gaze, as shown in figure 7.1 below. The second issue was that the cursor would jump rapidly from one position to another due to the frequency of checks by the eye tracker. While troubleshooting these issues, I came across two possible alternatives that could resolve these issues. The first was to replace the GazePointStream with an EyePositionStream and use an average of the LeftEyePosition and RightEyePosition to determine the cursor position. However, this resulted in the cursor jumping to a random point on the screen due to the EyePosition not reflecting where the user was looking. This is illustrated in Figure 7.2 below. The other solution I found was to use a HeadPoseStream to determine the cursor position by the direction of the user’s head. In terms of accuracy, this solution would fit best with the implementation seen in games that natively support the eye tracker. This implementation would allow the user to ‘look around’ in game. The issue with this implementation was that the head direction recorded was mostly NaN, which is the equivalent of null, despite movement of the user’s head position and direction, as shown in Figure 7.3. After troubleshooting, this was found to be a bug with the eye tracker not detecting the user’s head while the script was running. This was resolved by recalibrating the eye tracker; however, it was found that the head direction values returned by the eye tracker were all less than zero. This was an issue as the Cursor position was in the form of an integer value for the X and Y co-ordinates. To resolve this, the head direction value was multiplied by a factor of ten to make them integer values. This led to cursor movements being seemingly unrelated to the direction of the user’s head, e.g. tilting the user’s head upwards may cause the cursor to move towards the right or downwards. For this reason, I returned to the GazePoint method of moving the cursor. To fix the ‘jumping’ issue, I put in a check that the new position was more than 50 pixels left/right or up/down from the current cursor position. After this, I attempted a preliminary test of the eye tracking cursor movement in Rocket League. This failed due to the cursor movements having no effect in the gameplay, despite the cursor being moved by the GazePoint in menus. I tested the gameplay with a regular mouse to see if this was an issue with the game, but the mouse movements worked during gameplay. In the next sprint, I will troubleshoot this issue and attempt to make the GazePoint more accurate.

A group of people in a large body of water

Description automatically generated

Figure . Example of difference between GazePoint position from script and actual GazePoint position. Note: Cursor shows GazePoint position recorded by script while circular selection shows actual GazePoint position.

A screenshot of a computer

Description automatically generated

Figure . Example of problem with use of EyePosition for cursor movement

A screenshot of a computer

Description automatically generated

Figure . Example of HeadPoseStream not recording the user's head pose

A screenshot of a computer

Description automatically generated

Figure . Example of the HeadPoseStream after recalibration

# References

[Tobii], G., 2018. *[Solved] No Gaze data received?.* [Online]   
Available at: https://developer.tobii.com/community/forums/topic/no-gaze-data-received/  
[Accessed 12 March 2020].

Bend Studio, 2019. *Days Gone Patch 1.30 Notes Discussion.* [Online]   
Available at: https://www.reddit.com/r/DaysGone/comments/cikfip/days\_gone\_patch\_130\_notes\_discussion/  
[Accessed 15 May 2020].

Bierre, K. et al., 2019. *Game Not Over: Accessibility Issues in Video Games.* 1st ed. San Francisco: Games Accessibility Special Interest Group.

Bishop, M., 2018. *Brief History of Disability and Web Accessibility in the United States.* [Online]   
Available at: https://www.ebsco.com/blog/article/brief-history-of-disability-and-web-accessibility-in-the-united-states  
[Accessed 20 November 2019].

DougDoug, 2020. *Can you bike down GTA's Mountain, using ONLY your voice?.* [Online]   
Available at: https://www.youtube.com/watch?v=UNSPeyf\_DN0  
[Accessed 19 March 2020].

Hoffman, B., 1950. *Danny Kaye beats Bertie the Brain at Tic-Tac-Toe.* Toronto: Life Magazine.

Microsoft Accessibility Blog, 2017. *From Hack to Product, Microsoft Empowers People with Eye Control for Windows 10.* [Online]   
Available at: https://blogs.microsoft.com/accessibility/from-hack-to-product-microsoft-empowers-people-with-eye-control-for-windows-10/  
[Accessed 14 May 2020].

National Disability Authority, 2017. *EU Web Accessibility Directive.* [Online]   
Available at: http://nda.ie/Publications/Communications/EU-Web-Accessibility-Directive/  
[Accessed 4 December 2019].

Nitray, K. J., 2015. *Tennis for Two.* [Online]   
Available at: https://sexedi.ga/09174746.pdf  
[Accessed 4 December 2019].

Nitsche, M., 2008. *Video Game Spaces : Image, Play, and Structure in 3D Worlds.* 1st ed. Cambridge, Mass: The MIT Press.

Oxford University Press, 2019. *Definition of accessibility noun from the Oxford Advanced Learner's Dictionary.* [Online]   
Available at: https://www.oxfordlearnersdictionaries.com/definition/english/accessibility?q=accessibility  
[Accessed 20 November 2019].

Perez, D., 2019. *Logitech G Adaptive Gaming Kit Creates New Possibilities for Gamers with Accessibility Needs.* [Online]   
Available at: https://www.businesswire.com/news/home/20191118005308/en/Logitech-Adaptive-Gaming-Kit-Creates-New-Possibilities  
[Accessed 14 May 2020].

sajidbaloch, 2018. *Tobii-Eye-Tracker-4C.* [Online]   
Available at: https://github.com/sajidbaloch/Tobii-Eye-Tracker-4C  
[Accessed 09 March 2020].

Simmons, M., 1975. *Bertie the Brain programmer heads science council.* Ottawa: The Ottawa Citizen.

Smith, A., 2014. *Tennis Anyone?.* [Online]   
Available at: https://videogamehistorian.wordpress.com/2014/01/28/tennis-anyone/  
[Accessed 4 December 2019].

Smith, A., 2014. *The Priesthood at Play: Computer Games in the 1950s.* [Online]   
Available at: https://videogamehistorian.wordpress.com/2014/01/22/the-priesthood-at-play-computer-games-in-the-1950s/  
[Accessed 3 December 2019].

The European Parliament and the Council of the European Union, 2016. *Directive (EU) 2016/2102 of the European Parliament and of the Council,* Strasbourg: Official Journal of the European Union.

Tobii Tech, 2016. *Tobii Releases Next Generation Gaming Eye Tracker.* [Online]   
Available at: https://www.tobii.com/group/news-media/press-releases/2016/10/tobii-releases-next-generation-gaming-eye-tracker/  
[Accessed 14 May 2020].

Wilds, S., 2020. *For All The Players: A History Of Accessibility In Video Games.* [Online]   
Available at: https://www.ladbible.com/technology/gaming-for-all-the-players-a-history-of-accessibility-in-video-games-20200124  
[Accessed 29 January 2020].

Wolf, M. J. P., 2007. *The Video Game Explosion: A History from PONG to Playstation and Beyond.* illustrated ed. Westport, Connecticut: ABC-CLIO, 2008.