Gesture Based UI Team Project

Author: Fionn Browne & Pavel Antonov

G00373152 & G00373627

Department of Computer Science and applied Physics

Atlantic Technological University Galway

https://github.com/FionnBrowne/GESTURE-BASED-URI-DEVELOPMENT

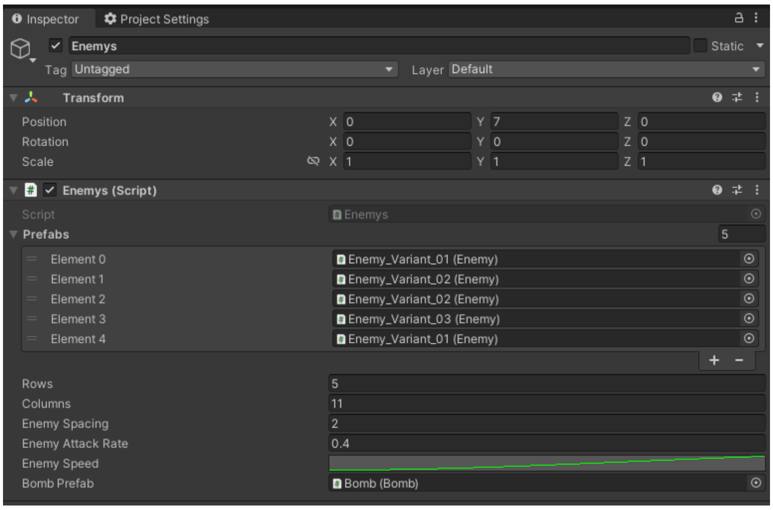
Table of Contents

1 Purpose of the application

We were required to develop an application using a natural User interface for this assignment. We had a range of deceives available to us, such as the MYO armband, the Kinect, and the Oculus VR headset. We could have used voice controls for our application as it also fits the parameters of gesture-based control, but we wanted to use real-world hardware as we have not had an opportunity like this before.

We decided to go with developing a classic game using a gesture-based interface, so we went with a clone of space invaders using the MYO armband as the player controls. This game is done in unity using C#.

The user will set up their view to a 16:8 aspect ratio for this game and land on a home menu. When they launch the game the will, we control the player and try to destroy all enemies before they hit the ground level. The player can hide behind shields to save their life or shoot through the defensive barrier to get a clean shot at the enemy. The more enemies are killed, the faster they will shoot and move down along a gradual curve which the user can set to increase the difficulty.



☑ Enemys Static
Tag Untagged Layer Default

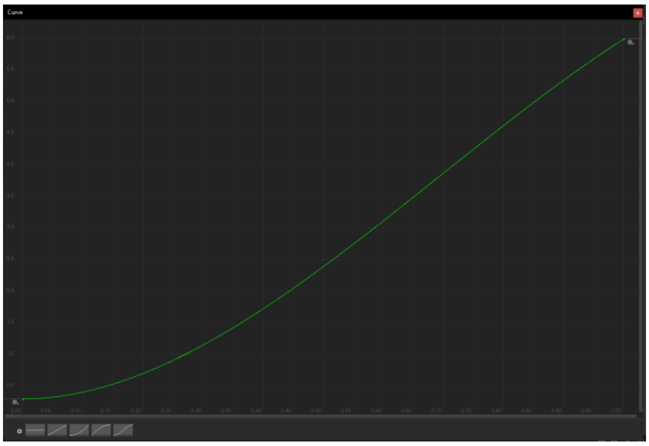
Transform

Position	X	0	Y	7	Z	0
Rotation	X	0	Y	0	Z	0
Scale	X	1	Y	1	Z	1

☑ Enemys (Script)

Script	Enemys
Prefabs	5
Element 0	Enemy_Variant_01 (Enemy)
Element 1	Enemy_Variant_02 (Enemy)
Element 2	Enemy_Variant_02 (Enemy)
Element 3	Enemy_Variant_03 (Enemy)
Element 4	Enemy_Variant_01 (Enemy)
Rows	5
Columns	11
Enemy Spacing	2
Enemy Attack Rate	0.4
Enemy Speed	
Bomb Prefab	Bomb (Bomb)

Then the user can edit or insert multiple keys in the curve to set it to a level they find appropriate.



The user then can complete the level by destroying all the enemies on the screen, if the user wants to increase or decrease the difficulty of the game, they can increase the number of rows or columns of enemies in the game or their movement speed.

2 Gestures identified as appropriate for this application

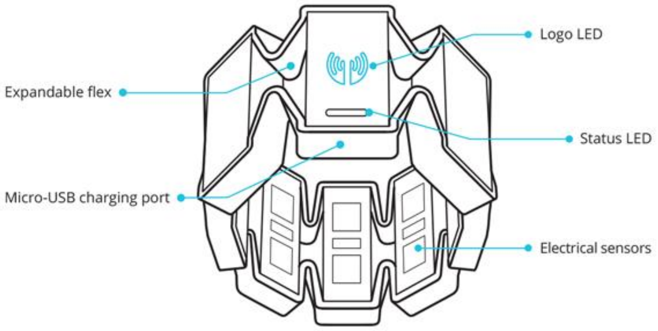
We experimented with many gestures in this game, and some we found did not synergise so well with each other, or the movement was quite stiff. We decided on these gestures for the player to use an MYO armband as it made the most sense. So, we went with the MYO armband being set on the left arm not to restrict the user's mouse hand. The gestures we used were:

- Wave in = moving right.
- Wave out = moving left.
- Fist = shoot.
- Rest (neutral gesture) = no movement
- Double tap = pause

We found that the wavering motions came naturally when trying to manoeuvre objects, so we modelled the action accordingly, which is why we went with the fist being shoot, as it does not strain the hand and makes sense to the user immediately on the play. Then for the player to stop moving, we thought a few approaches but landed on an idle hand would make the most sense, so we went with an open hand facing down, which can be easily transitioned into any of the other gestures with no strain. Then if the player needs a break, we went with a familiar gesture used for mobile devices, which is a double-tap, as most people are used to pausing or resuming applications on double taps, which provides a simple but straightforward movement for the MYO armband to detect.

3 Hardware used in creating the application

For this assignment we used the MYO armband, we went with this choice as we wanted to develop a 2d classic game and felt that the MYO would be perfect for this purpose as it's a compact and sturdy piece of hardware which provided most gestures, we had thought of implementing into the application. We had also considered the Kinect for our game, but we didn't have much room available to us and didn't want to use the Oculus as there was a limited amount of them and we wanted to make sure we received the technology we wanted.



🟦 Myo is connected via Bluetooth®

🟩 Myo is plugged in and fully charged

🟧 Myo is charging

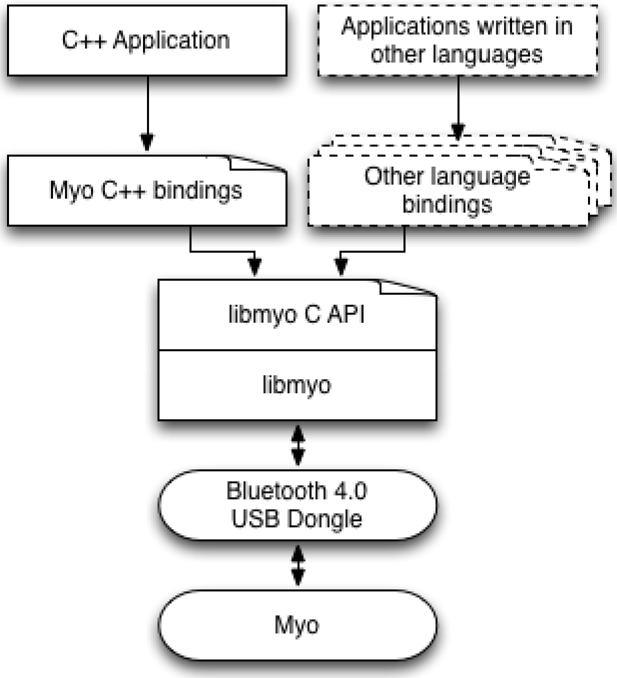
Myo LED colors and their meanings

The drawback of the MYO armband is that it was discontinued in 2018 and Thalmic labs links don't work correctly for the installation. But we were provided with the firmware and libraries required to implement the hardware into our unity project.

4 Architecture for the solution

The myoband SDK for unity was used to connect the Myo armband and project. The SDK contained an example lab with scripts. This was used to check out the Myo armband for the first time inside unity. The Myo SDK for unity contains a prefab essential for use in development. This prefab called my hub searches for a connected Myo armband on the pc without extra input from the user. The myo hub game object just needs to be in the game hierarchy and has a

myo game object as a child. This myo game object is then fed into our player script which utilises gesture controls from the Myo armband. In the player script, we check for which gesture it is and then have actions for them. All the player functionality is in the player controller script including myo gestures.



5 Conclusions & Recommendations

Developing this project was a fun and exciting experience to work with hardware and software and see other methods of developing applications or considering the controls/gestures more deeply to provide a smoother experience to the user and rapidly become second nature to them. This was particularly enjoyable as it is usually just keyboard input for a project which required consideration to the ergonomics and strain put on the user for prolonged usage. However, with the Myo armband, we had to think of all possible combinations of gestures that the user would try for each action. Which meant we could model the movements to everyday actions or gestures that people would use, which would make the gameplay much more intuitive. Since there were only one myo armband, both developers were present when testing it for the first time. We had some initial issues but using https://www.youtube.com/watch?v=3VLoGSVORjY for understanding how the Myo armband works with unity and https://developerblog.myo.com/setting-myo-package-unity/ for setting the application up allowed us to get underway. If we were to work more on this project, we would combine the MYO armband with the oculus or similar technology. The MYO has complementary features that would go well with VR for seamless and minimal wear to gaming.

6 GitHub & Requirements

For this application, it was designed to run using the MYO armband and through unity. For installation of this application a user would be required to use github and have git installed on there computer. Next clone the repository to a file which you will select through the unity hub. Once completed create a custom aspect ratio for the game of 16:8 and then select play.

https://github.com/FionnBrowne/GESTURE-BASED-URI-DEVELOPMENT