**Gesture Based UI Development**

**A Unity “Space Invaders” style game.**

**Developed for use with the Myo Armband.**

Aaron Healy

Keith Higgins

**Contents**

**Title Page**

Purpose of the application 3

Gesture Research 4-5

Gesture Technology Research 5-11

Myo Armband 5

Leap Motion Controller 6

Kinect V2 7

HoloLens 8

Durovis Dive 9

Architecture for the Solution 10-15

Main Menu 11-12

Gameplay 13-15

Conclusion & Recommendation 16-17

**Purpose of the application**

Our application was developed with the aim of bringing the classic “Space Invaders” style game up to date with the latest technology. We wanted to create a game that was based on the same ideas as the classic scrolling shooter style games that we all played as children. Our aim was to incorporate a technology into our game that would involve gesture-based controls.

Gesture based games are becoming more and more popular as the technology available becomes more advanced. These styles of games offer the player the chance to really be involved in the game by using body movements or their voice to perform actions in the game.

In terms of our application, we decided to focus on using gestures to control the movement of a player’s ship in a shooter style game. We felt this would be the best area to focus on as it would create a fun and enjoyable game for the player, using different movements and gestures to avoid enemy bullets and fire back at the enemy.

**Gestures Research**

As a starting point for our project we decided to investigate what sort of gestures could be used in our game. Our game is a “Space Invaders” style shooter, therefore we had two main operations that we would want to control using gestures, movement of the player’s ship and firing of bullets from the players ship.

Researching this was interesting as we found there was a huge variety of gestures that could possibly be used. These varied depending on the type of hardware that would be used in the project. The first type of gesture we looked at was voice control. We found voice control interesting as it is a big trend at the moment with all the home assistance units available now such as Amazon’s Alexa and Echo and Googles Home units. However, after researching how voice control could be incorporated we decided not to use it. We didn’t feel it was suitable for our project as our game is fast moving and the voice recognition would have to be extremely quick to keep up with the gameplay.

We then researched using hand gestures such as finger movements and moving your arm in certain directions. These gestures would be very well suited to our game as it would allow us to use different hand gestures for different parts of the game e.g. Making a fist to fire. The movement of an arm in a certain direction could also be used to control our player’s movement, for example you could move your arm left ad the ship would respond by moving left.

We also looked at full body gestures such as leg and head movements along with the arms. Although applications made using the full body as a controller can be very cool and fun to play, we felt a full body gesture style would not suit our game as we only need a few movements. After this research we decided to move ahead and choose the hand gestures as our gesture for the application. We felt it would be the most practical for use with our project and would work well with the style of our game.

**Gesture Technology Research**

After researching various gestures that could be used in our project, we decided to research the different hardware available to us for use in the project. We were informed there was a limited number of Myo Armbands, Leap Motion Controllers, Kinect V2’s, HoloLens’ and Durovis Dive available to us from the college. Below is some of the information we researched about each of these and why we felt they would or wouldn’t suit our project.

**Myo Armband**

The Myo Armband is a gesture control armband that uses proprietary EMG sensors to measures electrical activity from your muscles to detect hand gestures. There are 5 different gestures it is able to detect. It is also able to detect motion, orientation and rotation of the forearm by using a 9-axis IMU. Bluetooth is used to transmit this information to compatible devices. The Myo would work well in our project as it would allow the user to control the movement of the ship using a gesture or through detection rotation and orientation. It would also be possible to use a gesture to control the player’s shooting.



**Leap Motion Controller**

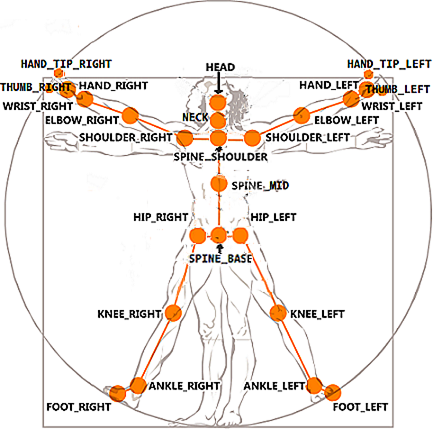
The leap motion controller is a device that plugs into a computer and can track hand movement. It consists of two cameras and three infrared LEDs. These track infrared light which have a wavelength outside the visible light spectrum. The leap motion controller features wide angle lenses. This allows the device to have a large interaction space of 8ft3. This space is in shape of an inverted pyramid.

A USB controller reads the sensor data and performs any necessary resolution adjustments. This data is then passed to the Leap Motion tracking software. The data takes the form of a grayscale stereo image of the near-infrared light spectrum, separated into the left and right cameras. Once the image data is passed to the computer, advanced algorithms are applied to the raw sensor data. Background objects are compensated for, the images are analysed to reconstruct a 3D representation of what the device sees.



In terms of our project the leap motion controller could possibly be used very successfully. The accuracy it offers could be used to move the ship small amounts to avoid enemy bullets. One problem that the leap motion controller could offer is a difficulty to pick up all hand gestures correctly in harsh light conditions such as bright sunshine.

**Kinect V2**

****The Kinect has two cameras, a depth camera and a colour camera. These cameras help track up to six skeletons at once. Each of these skeletons has 25 joints that are tracked. These are shown below.

The Kinect skeleton returns joints and not bones. It’s an important difference that can have big implications when you think about how bodies move in space. The joints are numbered from zero to twenty-four. There are eleven properties associated with each joint. Two for colour, two for depth, three for camera and four for orientation.

The Kinect has an infrared sensor which is used with the camera co-ordinates to find 3D points of each joint in space. These are used for joint positioning in 3D projects.

The depth range of the Kinect is eight meters, but the skeleton tracking range is 0.5m to 4.5m. It doesn’t usually find a skeleton closer than 1.5m so the depth will usually fall somewhere between 1.5 and 4.5. The width range can go up to about six meters and the height range will depend on the distance from the camera, but can usually find about five meters in height.

In terms of using the Kinect in our project, we felt that it would be a bit overkill for a game of our style. The advantage of the Kinect is the way it is able to track the whole skeleton and multiple people at once. This is not needed for our game so we decided it wouldn’t be the best option for us to use.

**HoloLens**

The HoloLens is a virtual reality headset with transparent lenses for an augmented reality experience.

****

HoloLens allows users to experience 3D holographic images as if they are a part of their current environment. This level of immersion enables new forms of computing in which the user’s desktop could be the living room.

The HoloLens is a powerful device which packs more processing power than many notebooks. In terms of hardware, the HoloLens features 3D spatialized sound, Wi-Fi, a camera similar to the Kinect which has a 120 degree spatial sensing system, lots of gyroscopes and accelerometers and a transparent screen for each eye.

The HoloLens would be a possible option for us to use in our project. However we felt the design of our game would not bring out the main features of the HoloLens and therefore decided against using it.

**Durovis Dive**

The Durovis Dive is a Virtual Reality headset that can take your mobile device and turn it into a Virtual Reality headset.

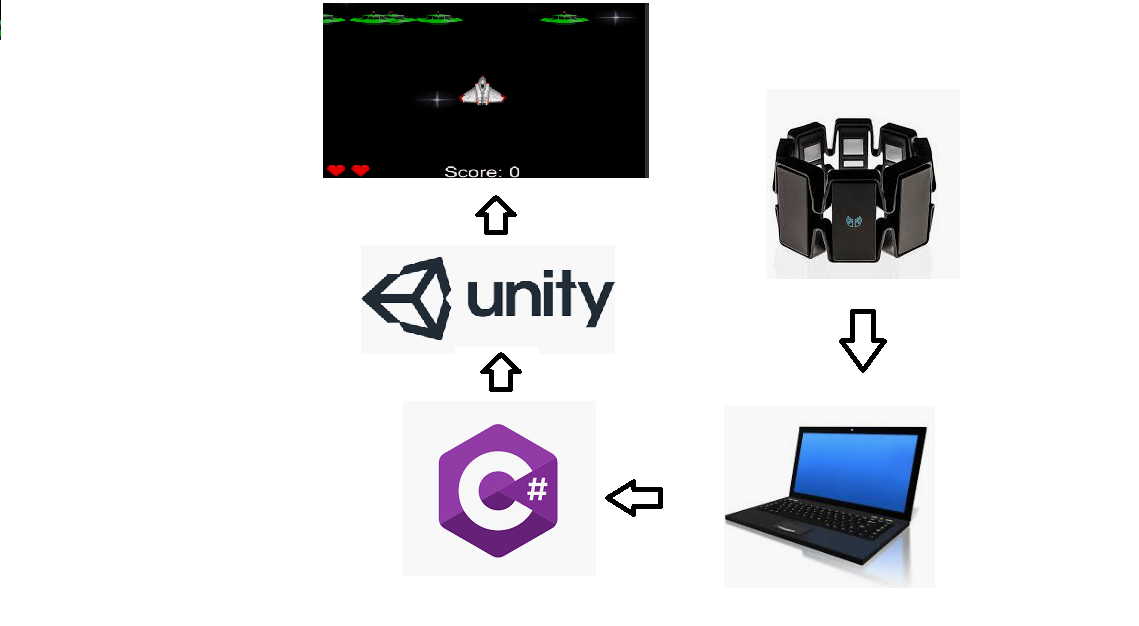


It is a rugged nylon viewer for mobile devices featuring two adjustable lenses which project an image of the virtual scene into each eye. Using it, you experience a stereoscopic view that spans most of the visual field. Combined with Dive head-tracking software, this creates a fast and responsive VR experience using the gyroscope sensors of the device.

In addition, Dive 5 offers a magnetic switch for click input which is fully compatible with both Dive and Cardboard apps. We did not feel Dive would suit our project as we were not looking to make a game that used a VR headset.

**Architecture for the solution**

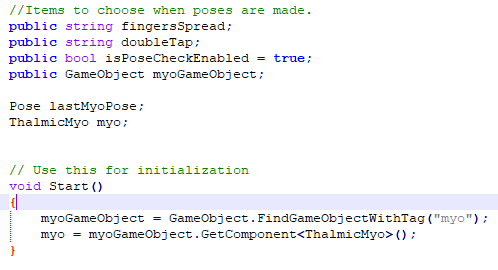
Our project was made up of essentially three parts, the unity assets, the C# scripts and the Myo armband all worked together to create our final game, A diagram of the architecture is shown below



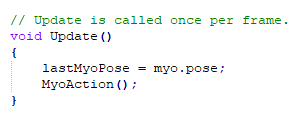
As you can see from the above image, our Myo armband was connected to the PC, we then had a number of C# scripts which were linked in with unity and its assets. All these combined came together to create the final game itself. Below I will discuss the main aspects of our game and how they are implemented.

### **Main Menu**

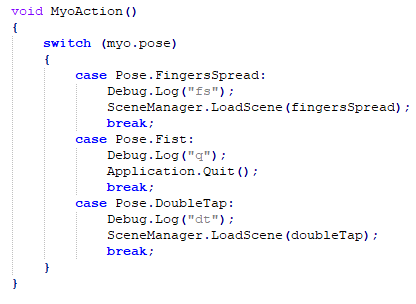
The Main Menu script in the application handles all of the logic for the menu scene. This is also the first place the Myo is referenced. We do this by creating a GameObject and linking the Myo in the scene to it using tags

. 

After we have linked our Myo in the scene to that GameObject, we are able to detect the poses being made by the user. We then feed this to our method which tells the game how to react depending on the pose the user has made.



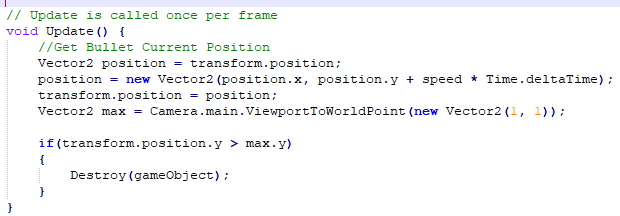
The method to deteremine what action to take based on the pose made is done using a simple switch statement. All our Myo controls throughout the game are done this way.



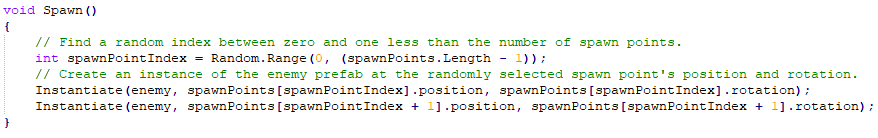
**Gameplay**

In our gameplay we have a few scripts operating. We have enemy and player bullet scripts, a spawn script, game control script, a player script and an enemy script. Ill explain what the function of these scripts are below.

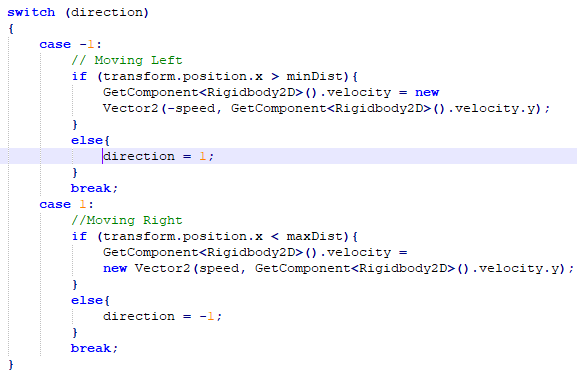
Bullet Scripts are used for controlling how the player and enemy fire their bullets. The scripts are identical apart from the speed and direction of the bullets being flipped for enemy compared to player.



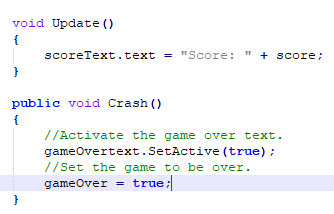
Our enemy spawn script is used to control the way an enemy spawns. This includes where it spawns and at what rate. The spawn method works by getting random points in the spawn points that have been input through unity and instantiating an enemy at these points.



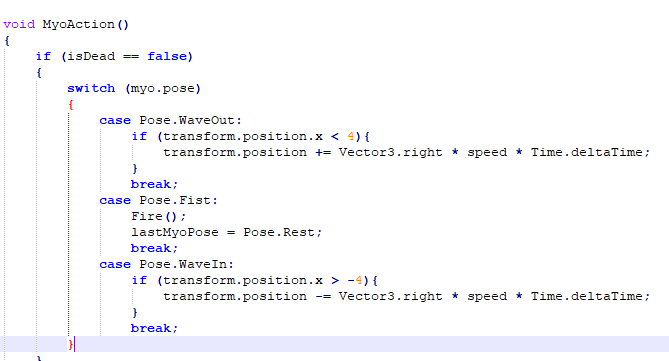
The enemy control script is used for controlling the enemy’s movement and the scenario in which the enemy is killed. The movement method is shown below. This works by moving the enemy left or right until they reach a certain point which has been input through Unity. Once they hit this point they head back in the opposite direction.



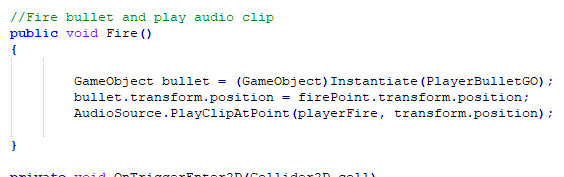
The Game Control script is used for controlling game items such as the score and whether the game is over or not. Below shows how the score is updated through this script and how if the player has died the game is declared over.



Finally our player script controls everything to do with the player, this mostly involves controlling his movement and gunfire through the Myo armband. The Myo is interacted with in the same way as it is in the main menu script. It calls a method that completes an action based on the pose being input. This method is shown below.



The Fire method controls the player gunfire. It creates a new Game Object every time it is called and instantiates a bullet to this object. It the moves the object straight up the screen from its location at a set speed. An audio clip also plays when the method is called to mimic gunfire.



**Conclusions**

**Conclusion**

From all our research, we discovered there was a lot of devices out there that can be used in gesture related applications. Some are more suited towards more precise movements and others for less precise. For example, the Kinect is far more suited towards full body movement rather than specific limb movement e.g. fingertip movement. In our case we decided that we would use the Myo armband as we felt that it would more suit our app as we just wanted to track hand movements for controlling the character in the game. Close behind our number one choice was the leap motion controller, but we felt that for player movement and controls the Myo had a wider range of pre-defined gestures that fit our app e.g. fist gesture which we mapped to shooting because for us that felt like the natural gesture.

In the end of the research we were surprised how many devices they’re that can keep track of gestures and how accurate most of them are. We found the research very interesting as me and Andy were very uneducated about gesture based devices from the beginning. I can confidently say that we have added a lot of knowledge of gesture devices to our resume.

**Recommendation**

After completing the project, Aaron and I were surprised how finicky the Myo armband was. We found that 50% of the time it didn’t pick up the right gestures and the other 50% it was impossible to connect to our laptop. We also were very disappointed with the latency issues that we found when developing our game. For example, when we did a gesture, the game responded about half a second later. Although this doesn’t seem a lot of time, this proved to be a huge problem with player controls. After some research and hoping we would find a solution, we discovered that this is a major drawback of the Myo. Although it is connected by Bluetooth to your device, the gestures are registered on your computer rather than the Myo device itself. This is the cause of the latency issue. If we had known this before we began the development we would have probably picked the leap motion controller as it is far better with latency issues than the Myo armband. Our classmates also agreed with us on this issue and they actually ended up hating the Myo armband at the end of the development.

On a more positive note, Aaron and I loved the fact of replacing a keyboard or a controller with gestures, so it was a good learning experience and something we will definitely look into in the future.