Evaluative Report – “The A-Maze-Ing Zombie Game”

The Leap Motion Controller can track all 10 of your fingers simultaneously to within a hundredth of a millimetre (Gorman, 2013), this is an astonishing feat considering the size and simplicity of the device. The simplicity doesn’t stop there for the Leap Motion, the setup for this device is as easy as it should be for any software; Plug in USB device, follow link to download software, start. It’s integration with unity follows suit, the package containing the prefabs and sample projects can be found easily on the Leap Motion website. Using the technology can be somewhat harder, when writing the code for the software it can be difficult to test different gestures as it will not always pick up the specific gesture you are trying to make. As with any project where you are developing software there is always bumps in the road, with the leap motion having to declare which hand was which was something that I often forgot to do meaning the action would automatically fall to the hand that is left in the area for the longest time, I eventually changed it so that the logic worked around the ‘Foremost Hand’.

Comparing the Leap Motion to traditional forms of input (keyboard and mouse), it holds its ground in terms of entertainment, although there are a few bugs when it comes to hand tracking, overall it is a good, innovative, tool which brings another dynamic to the gaming industry. It is less useful as a tool to actually replace a mouse for general computer usage, I found it more difficult to navigate through the menus of the game when the leap motion was in control of the input. One of the strengths and also one of the weaknesses of the leap motion is that it uses cameras, this is a strength because it can physically track more of the hand compared to an infrared image alone, this gives rise to the hundredth of a millimetre accuracy. This is, however, a negative as this leads to the performance of the controller more susceptible to the lighting conditions of the room and as it is a camera the lens will need cleaning regularly to stop smudging. One aspect of using the controller which can be quite frustrating is that sometimes naturally your hand will get closer to the desk and in turn the controller, if one hand goes too close to the technology you may end up blocking out the other hand meaning all functionality will stop on the hand as it is no longer detected.

The core mechanics for this game include the movement of the character, which is now controlled by the rotation of the hand that is first picked up by the leap motion, if you want the character to move forward, formerly “W”, you must have a flat palm and move your hand so that palm is moving closer to the screen and over the leap motion controller. I removed the ability to move backwards, left and right as I felt this was getting overly confusing for the user and one movement of the wrist in the wrong direction could send them down the wrong path. I made these mechanics these actions as I feel these are the most natural way of controlling movement with just hand movements – although this isn’t very intuitive for the movement of people the same principles apply as if you were controlling a plane.

Another mechanic for this game I am now controlling with leap motion is the shooting and aiming of a gun, this is now controlled by the pointing of the index finger of the hand that is in front when the user wants to look around, but behind when the user is walking forward, this is replacing the traditional input of the mouse cursor. The shooting is controlled by the presence of the thumb on this hand that controls the looking, if the thumb is present the gun will not shoot, if the thumb is tucked under the palm or closed in towards the index finger the gun will shoot. I decided to make the gun shoot like this because this feels like the most authentic way of replicating the pulling of a trigger, I had tried it so that when the thumb was present it shot but this didn’t feel as real as removing the thumb to shoot.

The mouse cursor has also been removed on the main menu for this project, it has been replaced by the tracking of a hand, similar to the aiming of the gun. I looked at three options to implement the menu with leap motion. One of which included the use of a raycast from the end of an index finger and when the raycast hits a button, activate it. Another method was to implement the hand model into the scene which the player controls and when the hand collides with a button, activate it. The final method, which I decided to use, was controlling a 3D object with your hand, acting as a cursor, when the cursor is over a button, it will activate. I decided to use this method because I found the raycasts to be inaccurate and it was difficult to track the cursor across the screen. The use of a hand model into the scene didn’t look natural in my menu as there was only two dimensions to my menu scene, meaning that, again, the hand was hard to track in the scene. The use of a cursor was the most natural method to navigate the menu scenes, as the object was always in the scene it was easy to track and pick up if the user took their hands out of the scope for the controller. Changing weapons has changed from pressing “1”,”2”,”3” etc. to the number of fingers that are being displayed on the hand that is used to control movement. I chose for this to be the method of changing weapons as this felt more comfortable for player. In the design document, I intended to have the changing weapons on the right hand, this was difficult to track for the leap motion, when I tested the changing of weapons on the right hand it caused some discomfort when trying to hold out three fingers and go between shooting and not shooting

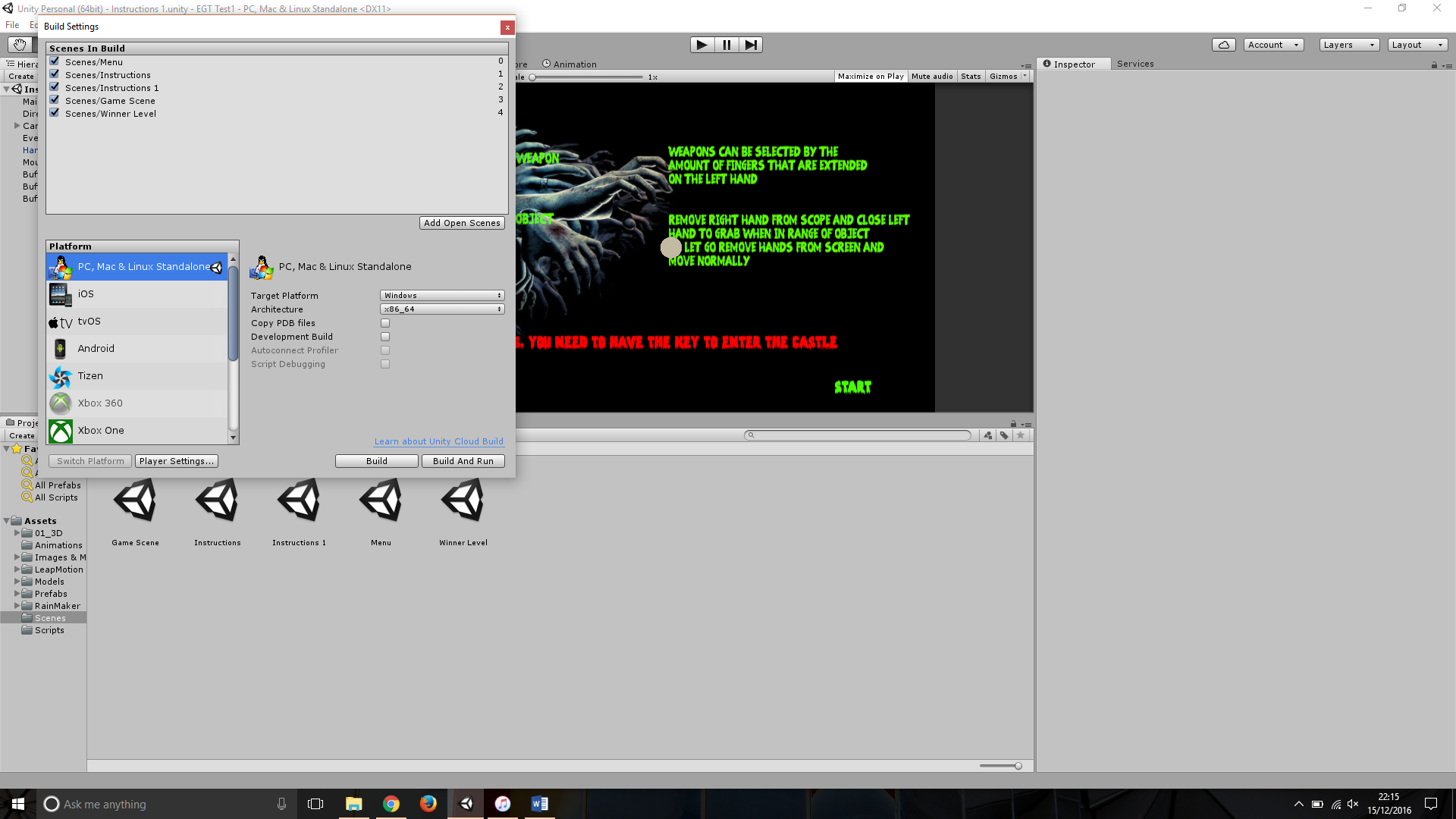
Another feature I have implemented with the leap motion is the ability to pick up and move an object, this is implemented by the closing of the left hand, this will bring the grabbable object towards the player, after the object is attached to the player open the hand again and continue to use the game in the same manner as originally intended. This works correctly until the user takes both hands out of the scope, after returning from unpausing the player can no longer move. To make the player to move again you must ‘grab’ the key and then ‘ungrab’ it. After you have done this you should be able to move again.

Looking at the performance of the game as a whole, most of what I set out to do does work, there was a slight change in theory when deciding what hand does what, originally, I planned for specific hands to perform specific tasks, this has now changed so the user can decide which hand they want to control movement and which hand they want to control shooting. I changed this mechanic to act like this as it gives the player more freedom and the ability to find what is more comfortable for them. The use of the hand to control the menus and aim the gun is a mechanic that isn’t perfect, it is difficult for the user to see what they are shooting at in game and the clicking of the button is sometimes not picked up by the leap motion. This is the same for the changing of weapons, the logic is in place correctly but the tracking of the fingers sometimes gets lost so the weapon will change without the player wanting them too. Another small problem with this application is that the game doesn’t like to have too many commands given to it at once, this means that moving and shooting can sometimes get locked into a set position.

Future iterations of this project could be improved by extensive testing on the most effective ways of performing these commands, I chose these methods as I thought they would be the best way. It would be more effective if I had the time to assess varying different methods of executing the same tasks.

Some of the problems I faced during this project include the confusion when importing the leap motion assets into the complete, non-leap motion, game. The obstacle that I encountered during this task was when one of the scripts that I had already made matched a script name of one of the imported scripts, this meant other imported scripts were trying to access variables from my ‘Controller’ script that weren’t in there. To fix this problem I changed the name of my ‘Controller’ script to ‘Controller Player’. Another problem with the leap motion software was; towards the end of the project the controller wouldn’t start capturing any movements unless you uninstalled and reinstalled the software to your computer. This was very time consuming and made it difficult to made small adjustments as you would have to wait for the uninstall and install process to complete before doing any work with it.

Appendix

The .exe file will not run straight from the file. This is a problem with the compatibility between the leap motion and unity. To run a successful build go into build settings in unity, make sure the architecture is set to ‘x86\_64’ and then select ‘Build & Run’, shown below

# References

Gorman, M. (2013, July 07). *Leap Motion Controller Review*. Retrieved from Engadget: https://www.engadget.com/2013/07/22/leap-motion-controller-review/