Gesture Based UI

Unity 3D + Kinect V2

JASON MC TIGUE – G00312233

Contents

[GitHub Link - 2](#_Toc478748465)

[Purpose of the application 2](#_Toc478748466)

[Gestures identified as appropriate for this application 3](#_Toc478748467)

[The gestures I chose to use 3](#_Toc478748468)

[Pause menu 5](#_Toc478748469)

[Custom Gestures. 6](#_Toc478748470)

[Hardware used in creating the application 6](#_Toc478748471)

[Myo 7](#_Toc478748472)

[Cortana 7](#_Toc478748473)

[Raspberry Pi 7](#_Toc478748474)

[Kinect 7](#_Toc478748475)

[Development 8](#_Toc478748476)

[Architecture for the solution 8](#_Toc478748477)

[Conclusions & Recommendations 9](#_Toc478748478)

[References 9](#_Toc478748479)

# GitHub Link - https://github.com/JasonMcTigue/GestureBasedUI

# Purpose of the application

The aim of this project is to develop an application to demonstrate the use of gestures.

For this project, I decided to create a unity application that uses gestures from the Kinect v2 to control on screen actions.

The game I created is like the popular arcade game brick breaker where the player must destroy all the bricks on screen before they run out of life’s. I incorporated various gestures from the Kinect v2 so the player doesn’t need to use any keyboard controls when playing.

I came up for the idea for this game when researching the different Kinect gestures and released that I would be able to use the lean left and right gestures to control the paddle and possibly use the open, close or lasso gesture to fire the ball from the paddle.



# Gestures identified as appropriate for this application

While researching gestures I found that there were several options I could use that could be appropriate for my application.

The first option was to use the built-in Kinect gestures which are different depending on which part of the body is being tracked.

When researching the Kinect v2 gestures I found out that the Kinect tracks all the bodies joints and that they were split into various categories such as:

* Body Data
* Joints Normal
* Joint Hierarchy
* Hand Data

I was most interested at this point in learning more on the hand gestures as the first thing I wanted to do was fire the ball from the paddle. There are five different hand gestures, they are

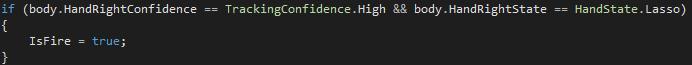
* Open
* Closed
* Lasso
* NotTracked
* Unknown

## The gestures I choose to use

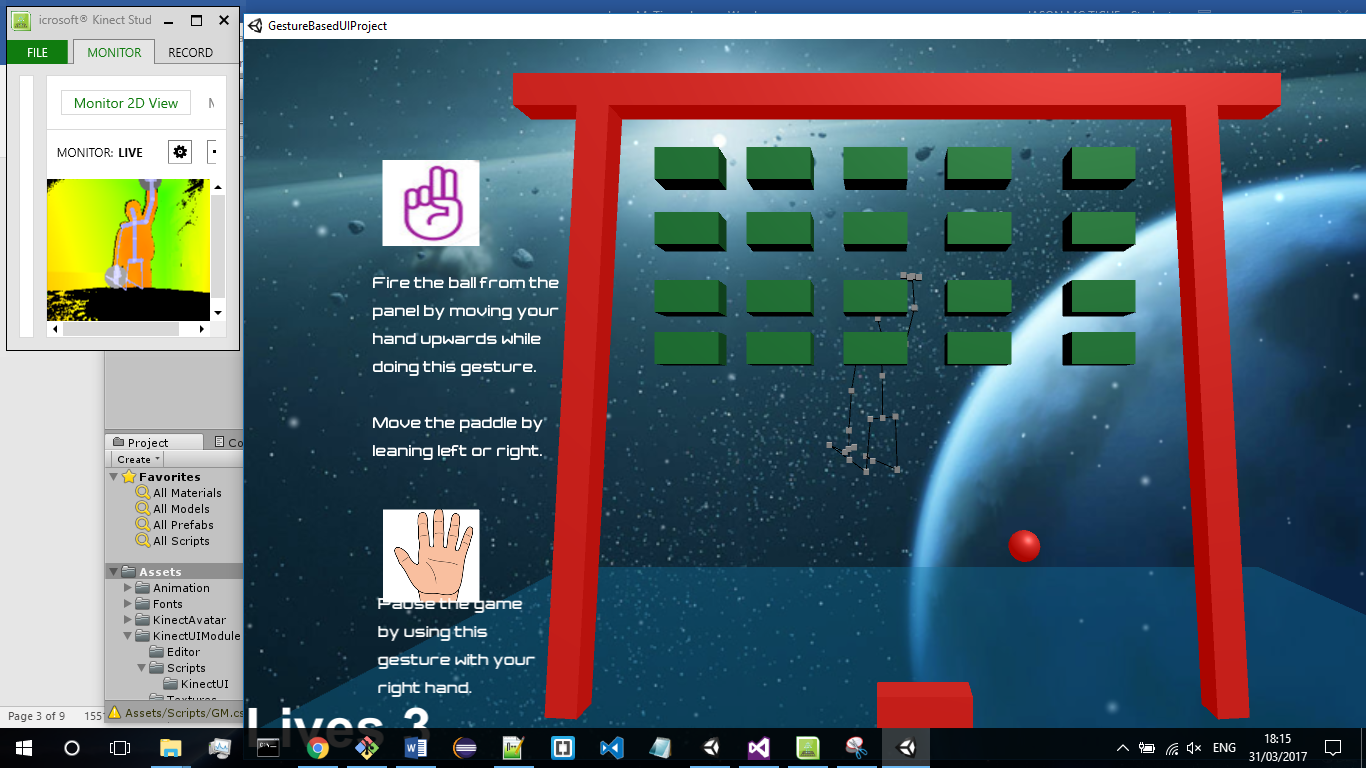
After doing some extensive research online I decided to use the lasso gesture to try and get the ball to fire. I choose this gesture because I thought it made the most sense for starting off the game as it is easy to do and understand. I opted not to use the hand Open or Closed gesture as I wanted to also create a pause menu and perhaps use these gestures for that purpose.

I then downloaded the Kinect v2 SDK which I found here [1] and imported it in to my project. The Kinect v2 SDK comes with three plugins which are basic ones that include all the built-in gestures, a face recognition plugin, and a gesture builder plugin.

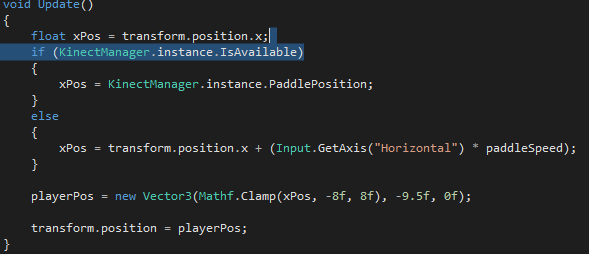
A small example of how I used the lasso gesture is demonstrated in the code here:

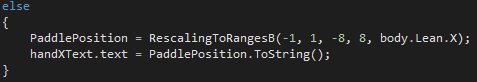


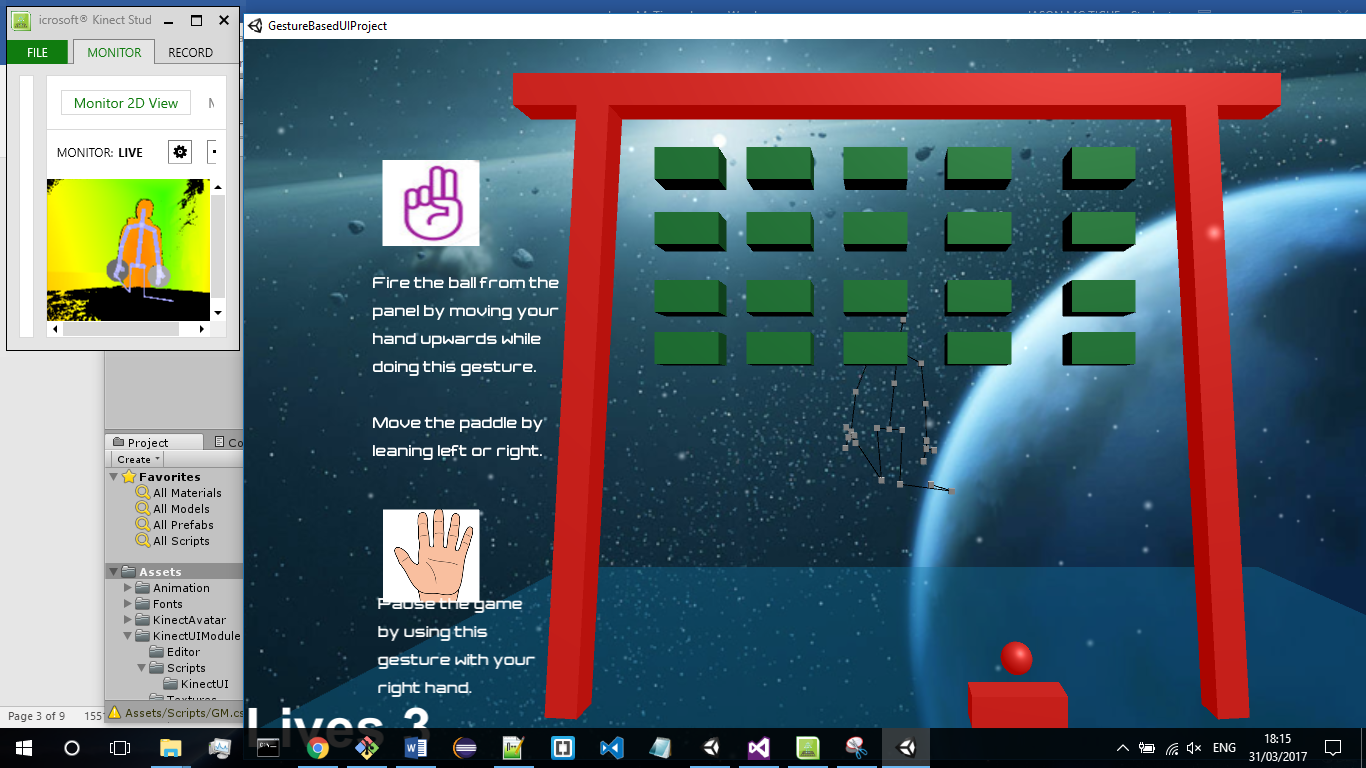
The Bool IsFire Is then set to true which in turn calls the method that fires the ball from the panel.



The second gesture I looked in to creating was one that would move the paddle either left or right depending on the player’s position. To do this I used a lean gesture which tracked the players position to the x axis of the paddle and then moved the paddle accordingly. A small example of some code I used to achieve this:







## Pause menu

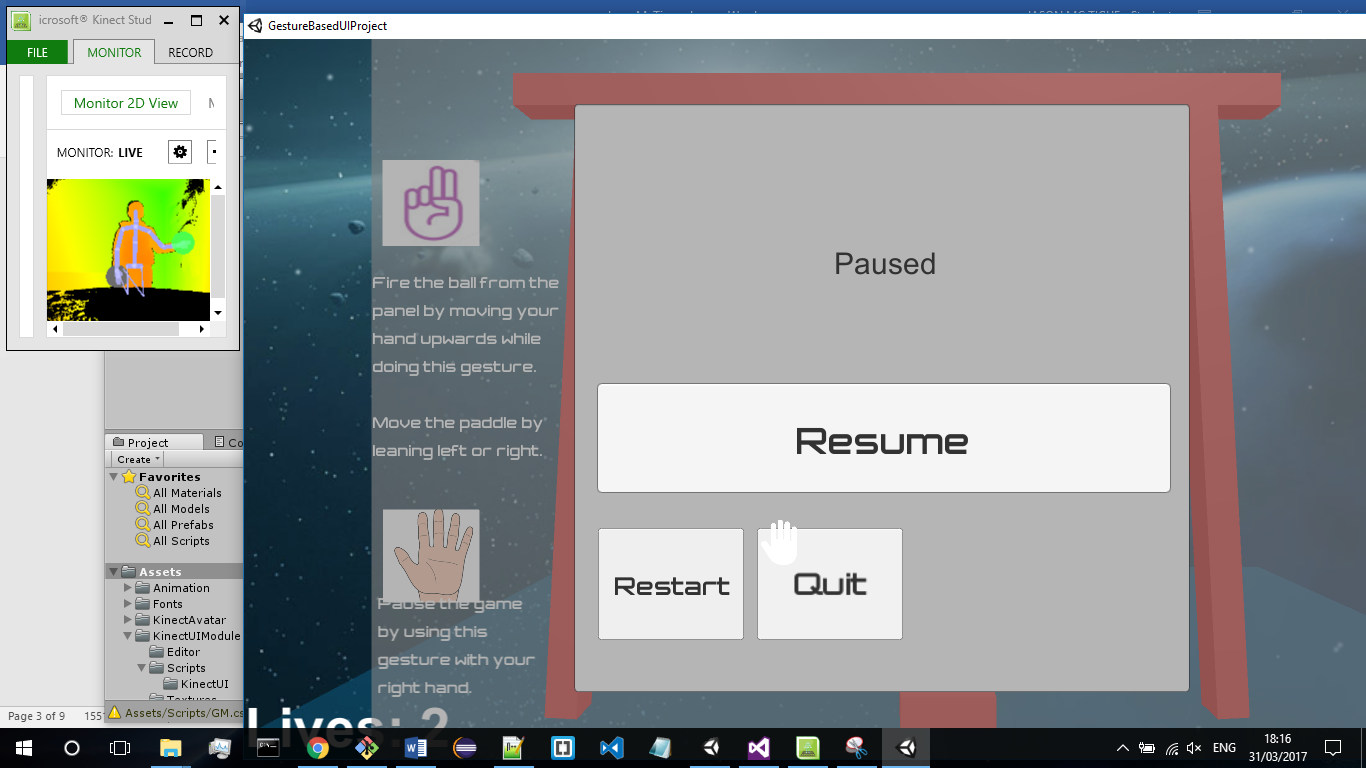
After a lot of testing and changes I eventually got the game to work using the Kinect gestures. My next step was to create a pause game menu in which I could use my hand as the mouse. The way I done this was to use a gesture that picked up when my hands became active on the screen. I found some documentation which was like what I wanted here. [2]

I followed this tutorial and picked out the parts I thought would be useful for my application. The parts I included were the body tracing, various actions like hovering and clicking and clicking options such as waiting over a button for a certain number of seconds before it opens.

I used these gestures for several reasons.

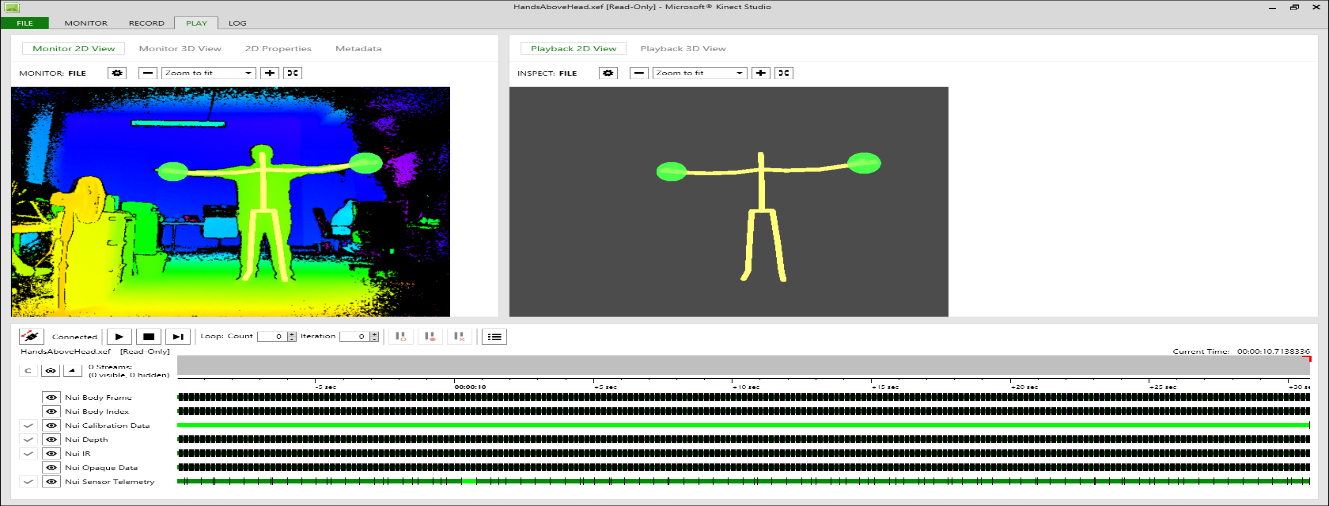
1. I wanted to be able to see the body tracking on screen so the user can tell when the Kinect is tracking their body.
2. I wanted to be able to click on buttons using Kinect gestures.
3. The ability to hover over a button and see the loading ring looked a lot better that just waiting five seconds for the button to click.

I then created a main menu screen and added similar gestures to it however to click on the how to play button I used a gesture where the user must open their palm to click. I added this to just demonstrate that I learned more one click gesture.

s

## Custom Gestures.

I also considered making custom gestures using Kinect Studio and Visual Gesture Builder but has I already had the gestures I needed I chose not to over complicate the project and didn’t peruse using any custom gestures. However, if I were to ever make a Kinect game in the future I would try and create a custom gesture as you are able to continually train the gesture until it is almost fully confident and you can also train in on various people e.g. some people might stand differently or move their hands in different motions that might stop the Kinect from recognising the gesture.



There a few bugs in the game such as the pause menu opening straight away and the hand cursor only moving a small distance on the screen.

# 

# Hardware used in creating the application

For this assignment, I had a choice of using a few several types of hardware. Some of which included:

* Myo armband
* Kinect v1/v2
* Cortana
* Raspberry Pi, Arduino

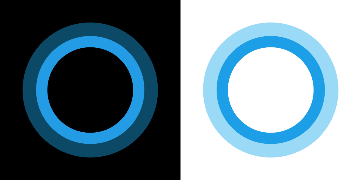
## Myo

When deciding which piece of hardware that I would use I considered the pros and cons of each one. The pros of using the myo armband were that they were easy to set up and easy to acquire via the college however after using the myo in class I found it to be unresponsive for me and I found that it was limited to only a few gestures and I didn’t feel like there was anything I could create for it that would be good enough as it already has a lot of applications on its market place.

****

## Cortana

I found Cortana to be also hard to use because it hasn’t been developed for the Irish accent quite yet so it was misunderstanding what I was saying a lot. I do think that once it is developed for the Irish accent it could be extremely useful in applications for voice control.



## Raspberry Pi

I had no expertise using the raspberry pi and I also had no access to one to I decided not to research much on it.



## Kinect

So that left me with the Kinect. I originally planned to use the Kinect V1 as I had one personally but after doing extensive research I found that it was outdated and that using the Kinect V2 was the best one to use.

The main benefits of using the Kinect v2 were that it came with an SDK and it also was straightforward to connect with unity. I found that using Kinect v2 was very enjoyable. The main negative I found was that you needed to get an adapter to connect it to your laptop.

****

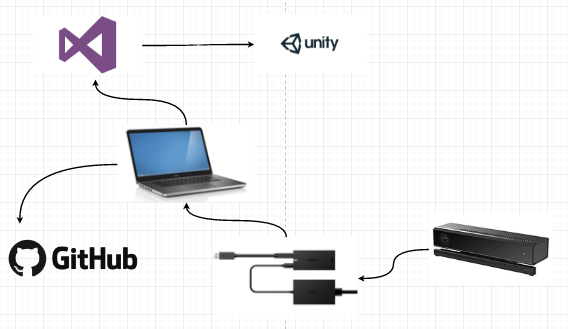
## Development

To develop the project, I used Unity and Visual Studio 2015. I found both to work well together and I had very little trouble using either. It was my first time using unity but I found that it is well documented online and easy to learn.

# Architecture for the solution

The technologies used are:

* Kinect v2
* Kinect SDK 2.0
* Unity 5
* Visual Studio 2015
* Github

****

The above diagram shows the architecture for my application. The Kinect V2 connects to a computer via an adapter which requires usb 3.0 to work.

Visual studio handles all the scripts for the project and unity is then able to handle all the import files from the Kinect SDK that are used when running the application. The whole project is then hosted on github.com.

# Conclusions & Recommendations

In conclusion, I feel like I have learned a lot from this project. It was the first time that I properly used unity and the first time I developed an application using the Kinect v2. I learned a lot regarding the different gestures that are available as I had no idea that there were so many different gestures that you could make. I also was unaware that it was relatively straight forward to connect Unity with visual studio and the Kinect v2.

Developing was a challenge as I had to learn the basics of the technologies I was using and how to integrate them in to a unity game.

The main thing that I would do differently if I were to do it again would be to spend more time on the custom gestures and try learning more about them as I feel like they could add extra aspects to the game. Another thing I would do differently is add more levels to the game and have differently things happening in the game such as boosts and rewards. I didn’t add them in to this application due to time constraints and because the main aim was to demonstrate gestures.

# References

1. <https://go.microsoft.com/fwlink/p/?LinkId=513177>
2. <https://nevzatarman.com/2015/07/13/kinect-hand-cursor-for-unity3d/>.
3. For the unity game <https://unity3d.com/learn/tutorials/modules/beginner/live-training-archive/creating-a-breakout-game>
4. For the pause menu <https://www.youtube.com/watch?v=PyEmRVRHWL8&t=7s>
5. Some gestures used <https://blogs.msdn.microsoft.com/mcsuksoldev/2011/08/08/writing-a-gesture-service-with-the-kinect-for-windows-sdk/>
6. https://msdn.microsoft.com/en-ie/library/dn799273.aspx