# **Gesture Based UI Development Project Documentation**

Github: <https://github.com/BernardWong97/Gesture-Based-UI-Project>

Team: Darren Regan & Bernard Wong

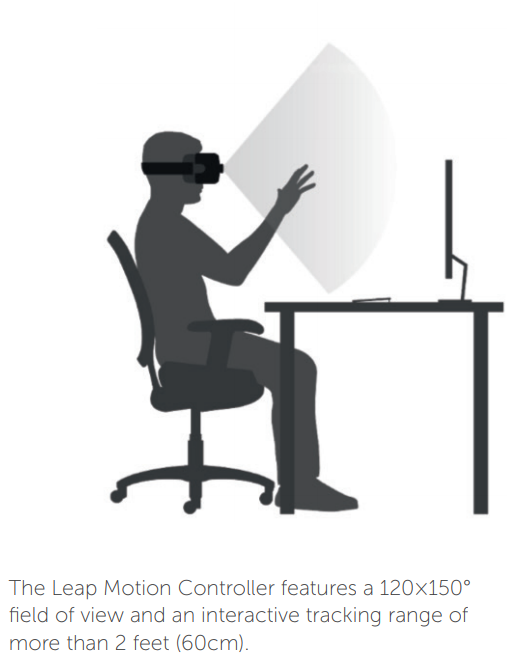
## **Introduction**

The goal of our project is to develop an application with gesture-based systems. There are many different options for technologies available to create a gesture-based application. After browsing our options, we decided to create a version of the popular game Fruit Ninja/Tap Accuarcy with the LEAP motion controller and or Mobile device.

## **What is a LEAP Motion?**

Leap Motion controller is a small USB device which is designed to be placed on a physical desk, facing upward (See figures below). It can also be mounted onto a virtual reality headset. Using two monochromatic IR cameras and three infrared LEDs, the device observes a roughly hemispherical area, to about 1 meter. The LED’s generate pattern-less IR light and the cameras generate almost 200 Frames Per Second.

The data is sent through a USB cable to the host machine, where it is analyzed by Leap motion software which uses 3D reconstruction from multiple images.

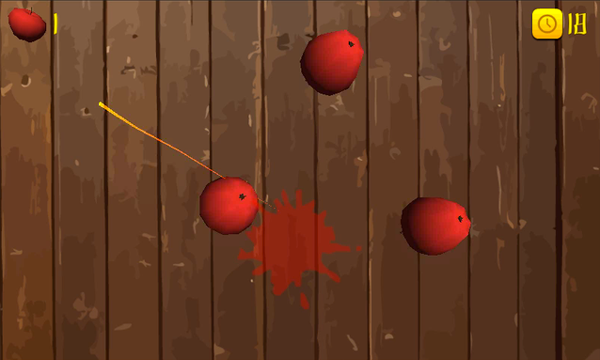


## **Purpose of the application**

The purpose of our application is to build Tap Accuracy/Fruit Ninja game with Leap motion and or Mobile device. The main gesture of our application would be thumb and or index finger, additional gestures could include voice gestures i.e. pause, start, end etc. if available/practical.

### **UI Design of Application**

The UI design of a tap accuracy and fruit slicer game is quite simple, the background commonly has a wooden texture, there is a score which increments on successful slice in the top left of the screen and a timer on the top right of the screen signifying how long the game has left before timeout. See figures below.



The cursor which is controlled by the player is often a simple circle light-ray that leaves a trail signifying a sharp cut. The trail grows with how accurate the slice is by the player which also happens to give a larger amount of points to the score. This is a good visual indicator for the player and provides feedback to the player on whether the player is doing well in the game other than the score.

Other visual feedback to the player is the timer which can decrease if the player hits object which decrease the timer. These objects are normally a generic bomb with a skull for the player to visually know that the object would have a negative result if cut by the player.

### **Gestures used for Fruit Slicer game**

The gestures that are used for fruit slicer games are normally Index finger for the Leap Motion Controller (See Intro to Leap video in Ref for example) and thumb for mobile devices.

Mobile devices and leap can also use either or, but each have a different finger that is used much more than the other.

A fruit slicer game only requires a single finger to play, while leap can use all five fingers for other gestures they are not needed for this application. Leap and a mobile phone have many similarities in gestures the only difference being a touch screen vs 3D imaging scanning with cameras.

## **Gestures identified as appropriate for this application**

As mentioned in purpose of the application the main gestures that will be incorporated into our project is Mouse Click, Thumb and Index Finger. Addition gestures can also be added like pinch thumb + index finger to pause and voice commands if available.

### **Justification**

The justification for Index finger is that the staple for a fruit slicer game is a single finger signifying a blade. The user would clench their first while pointing their index finger outwards to play the game. The user would then use a combination of moving the index finger and fist to slice fruits which is the objective of the game.

The justification for Thumb is that it provides an additional finger for the Leap motion controller if for some reason the User cannot use their index finger. The thumb is also commonly used as the main finger for a mobile phone depending on circumstance of the player. i.e. If the player is in portrait mode with a phone, the player would commonly only have one hand available to play.

If the player is in Landscape mode, the player would commonly have both hands available to play so they would use their index finger to play.

These considerations that need to be made as environment of the player could prevent them from playing our game if we don’t account for them.

### **Research**

Research for these gestures was quite simple, as well as watching videos, playing an actual fruit slicer and tap accuracy games on my mobile phone provided all the feedback I needed to correctly identify the gestures required for a tap/slicing game. The gestures are quite simple and straight forward, while playing I noticed I would somethings switch between index finger and thumb depending on my environment. This made me realize that a single finger would not be enough, and we should account for it in some way.

## **Hardware used in creating the application**

The hardware we are choosing is Leap Motion Controller or a Mobile phone device.

### **Purpose of Leap Motion Controller**

Using a Leap Controller provides an extremely accurate sensor that was established to be better and comparable with that of a computer mouse. The leap controller provides greater control of user’s hands providing a different experience for the User. The usability on the leap can be limited by the limited field of view which is a common disadvantage of the leap over a device like the Kinect.

The leap does provide a new and different experience which is its main advantage over a mobile phone.

### **Purpose of Mobile Phone**

Using a mobile phone provides a high standard of playability for the user, there is a high chance of the user already knowing how to play the game compared to the leap where the user would need to learn the controls and create the muscle memory needed to play the game correctly.

Mobile phone also provides handheld mobility which leap cannot provide as you need to be stationary on a desktop to provide a good experience. Another advantage of a mobile phone is the robust development kit around it compared to leap due to the amount of mobile phone users compared to leap users. This also makes your application available to more users which isn’t really a goal or issue for our project but is a large disadvantage in a retail version if you do not build a version that supports both systems.

## **Architecture for the solution**

Unity uses an entity/component model instead of class inheritance. So, for example a GameObject doesn’t have material or texture or renderer. However, the entity you see in the game might be built of GameObject and Renderer components (and many others), each with their own properties and methods.

For an example of this a good reference would be this article: [Unity: Now You're Thinking With Components](https://gamedevelopment.tutsplus.com/articles/unity-now-youre-thinking-with-components--gamedev-12492)

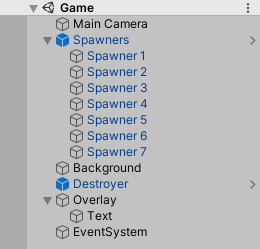
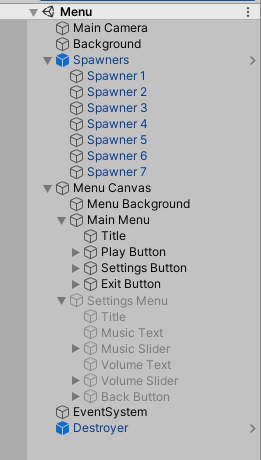
The Structure of the game is a 2D environment with a wooden texture background. There is a Main Camera object which is the point of view the player would see. Out of Camera view there is five different GameObjects, each object has a RigidBody2D which gives it gravity. Each GameObject has a frozen X-Axis, and their Y-Axis is set to be above the camera view. This is done to create the effect of items falling for users to destroy to gain points. The objects fall at a set velocity in the Y-Axis.

To get Touch inputs unity has a Script Reference [here](https://docs.unity3d.com/ScriptReference/Input.GetTouch.html) it gives a full explanation of implementation of touch inputs as well as the code involved. [Input.GetTouch details the structure of Touch Inputs](https://docs.unity3d.com/ScriptReference/Input.GetTouch.html)

[TouchPhase describes phase of finger touch](https://docs.unity3d.com/ScriptReference/TouchPhase.html)

For voice input in unity I used Microsoft Speech which has a resource for [Unity](https://docs.microsoft.com/en-us/windows/mixed-reality/voice-input-in-unity), we used this in our labs, so it might not be in the project. But was tested for simply task as saying “Play”. It worked the same as our lab we did so I didn’t add it to the project.

Example of objects for Menu Scene and Game Scene,



## References

* <https://www.ultraleap.com/>
* <https://www.ultraleap.com/product/leap-motion-controller/>
* [Leap Spec Sheet](https://www.ultraleap.com/datasheets/Leap_Motion_Controller_Datasheet.pdf)
* <https://github.com/leapmotion>
* <https://github.com/topics/leap-motion>
* [Intro to Leap Motion](https://www.youtube.com/watch?v=_d6KuiuteIA)
* [Paper on Usability of Kinect & Leap in Application for Gesture control of Biomedical Images](https://www.scss.tcd.ie/publications/theses/diss/2014/TCD-SCSS-DISSERTATION-2014-010.pdf)
* <https://www.techradar.com/news/leap-motion-hand-tracking-comes-to-smartphones>
* <https://docs.unity3d.com/Manual/TheGameObject-ComponentRelationship.html>
* <https://docs.unity3d.com/ScriptReference/Input.GetTouch.html>
* <https://docs.unity3d.com/ScriptReference/Touch.html>
* <https://docs.unity3d.com/ScriptReference/>
* <https://docs.microsoft.com/en-us/windows/mixed-reality/voice-input-in-unity>
* <https://docs.unity3d.com/ScriptReference/TouchPhase.html>