LEAP MOTION SIMPLE CONTROL PACKAGE

The package includes Leap Motion Core Assets v3 Orion (LeapMotion folder) (link here). In case you would like to upgrade Leap Motion Core Asset, just replace this folder.

This asset aims to provide predefine some hand gestures based Leap Motion API for further integration within larger project. It also include some code example for the application of these gestures, such as menu navigation, throwing ball, etc.

Please be aware this package depends on the library which is support from the Leap Motion team, then if you download new library from the Leap site and have trouble to use, please email me to notify. Very appreciated!

For more information, please don't hesitate to contact me via email dttngan91@gmail.com or udrawr@gmail.com

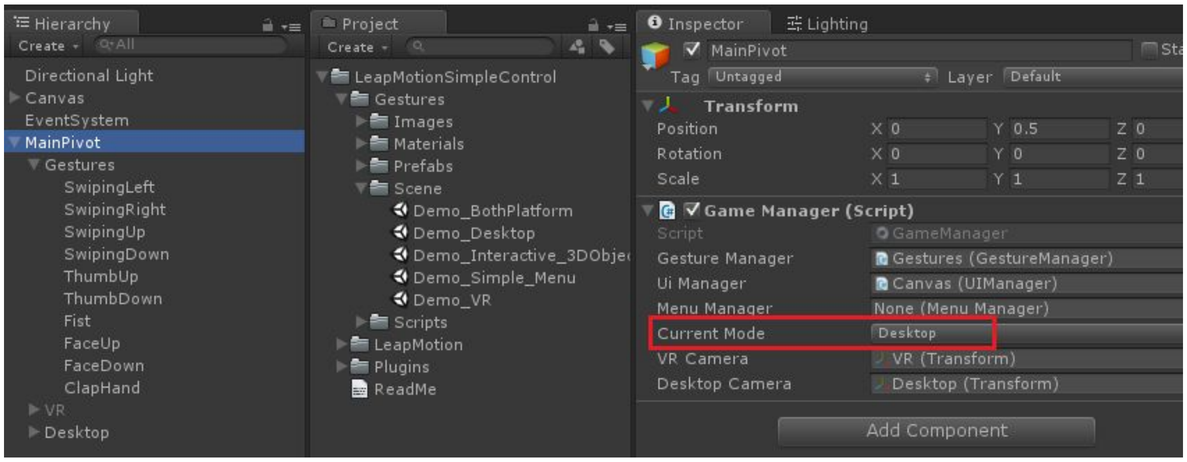
Demo: https://youtu.be/nFpJF41h9Zo

1. Scenes

- Demo_Beginner
- Demo_BothPlatform
- Demo_Desktop
- Demo_Rotate3D
- Demo_VR_Movement
- Demo_ThrowBall
- Demo_Simple_Menu

Demo_Beginner: A simple scene to demonstrate how to integrate our library as plugin. (see section 5 for more information)

Demo_BothPlatform: The system is designed for both Desktop and VR Leap Motion mode. You can easily switch between these two platforms by gameobject MainPivot in the Demo scene. Within the GameManager component, there is an option of current mode including VR mode and Desktop mode. You need to set this in the Unity Editor before running.


Hierarchy
Create All

- Directional Light
- Canvas
- EventSystem
- MainPivot
  - Gestures
    - SwipingLeft
    - SwipingRight
    - SwipingUp
    - SwipingDown
    - ThumbUp
    - ThumbDown
    - Fist
    - FaceUp
    - FaceDown
    - ClapHand
  - VR
  - Desktop

Project
Create

- LeapMotionSimpleControl
  - Gestures
    - Images
    - Materials
    - Prefabs
    - Scene
      - Demo_BothPlatform
      - Demo_Desktop
      - Demo_Interactive_3DObje
      - Demo_Simple_Menu
      - Demo_VR
    - Scripts
  - LeapMotion
  - Plugins
  - ReadMe

Inspector | Lighting

MainPivot
Tag Untagged Layer Default

Transform

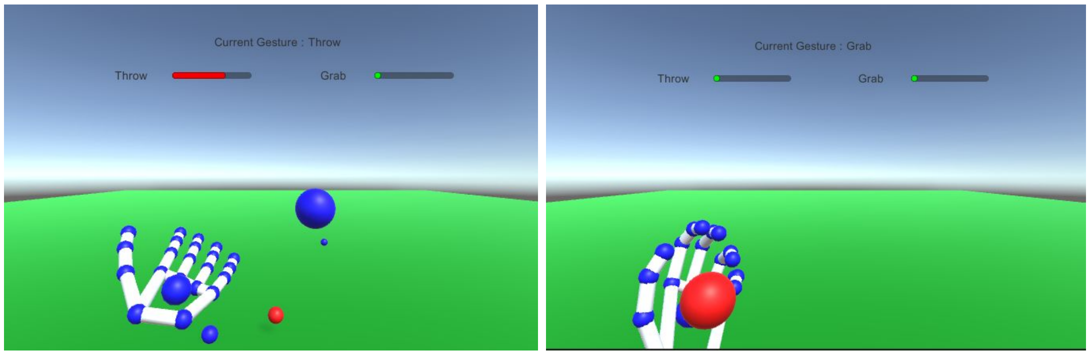
Position	X 0	Y 0.5	Z 0
Rotation	X 0	Y 0	Z 0
Scale	X 1	Y 1	Z 1

Game Manager (Script)

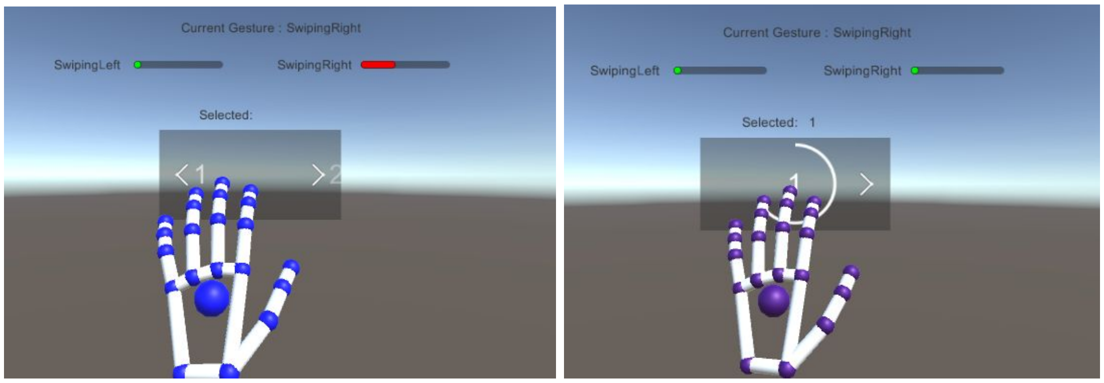
Script	GameManager
Gesture Manager	Gestures (GestureManager)
Ui Manager	Canvas (UIManager)
Menu Manager	None (Menu Manager)
Current Mode	Desktop
VR Camera	VR (Transform)
Desktop Camera	Desktop (Transform)

Add Component

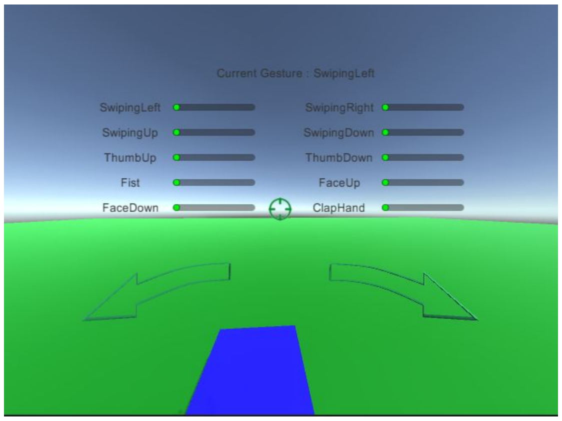
Demo_Desktop: specific package for each platform using Leap Motion
***Demo_ThrowBall**:* a demo for ball interactive via *Throw* a ball gesture and *Grab* gesture.

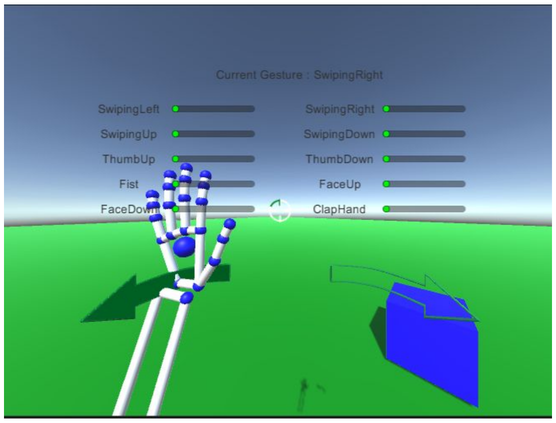


***Demo_Simple_Menu**:* a demo for simple menu navigation using Swiping Left/Right as well as Touch and Hold a 2D menu item to select.

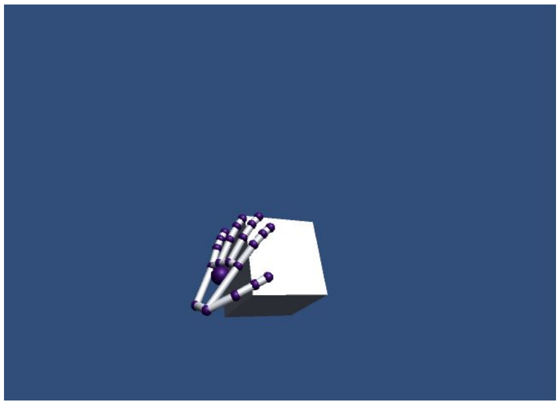
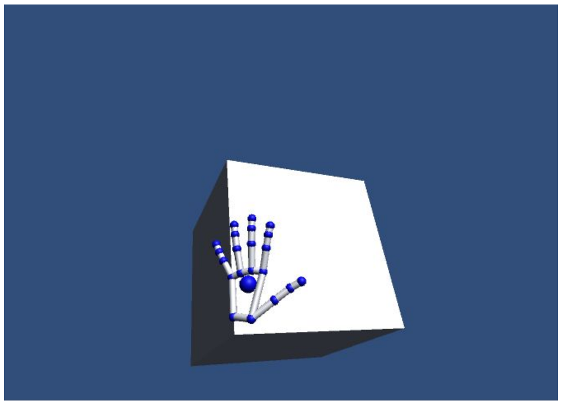


Demo_VR_Movement: a demo not only combines all gestures to use in the virtual reality environment but also propose a way to use gesture interaction to move around this 3D world.



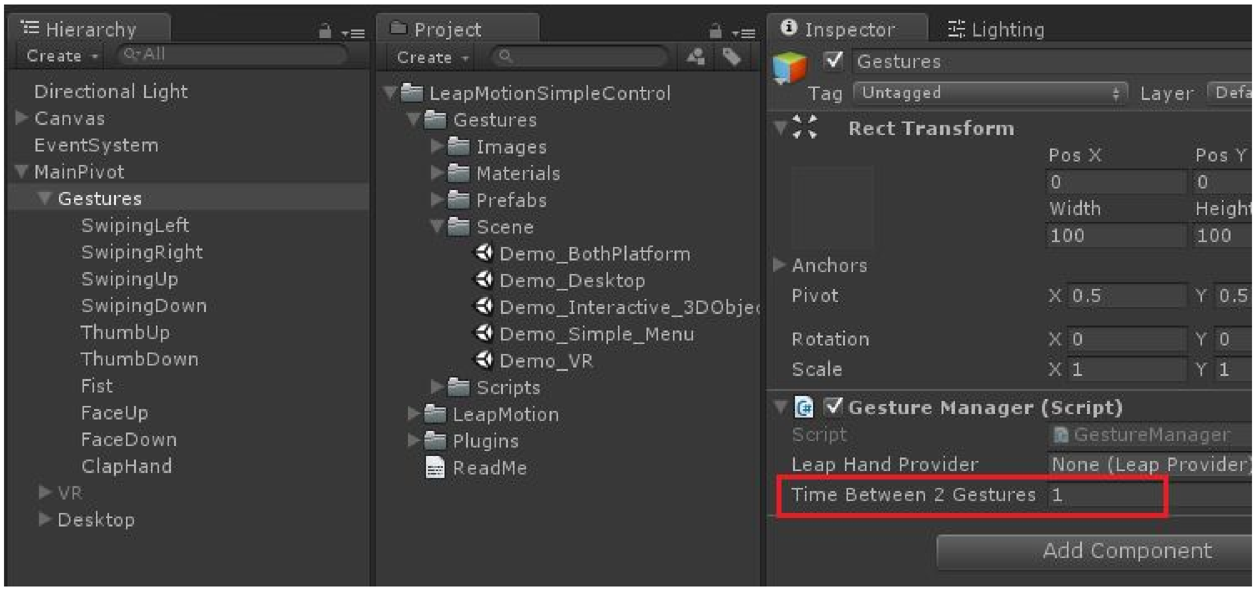


***Demo_Rotate3D**:* a demo shows how to use rotate a 3D objects as well as zoom in/out camera based on swiping left/right gesture and others gesture combination.

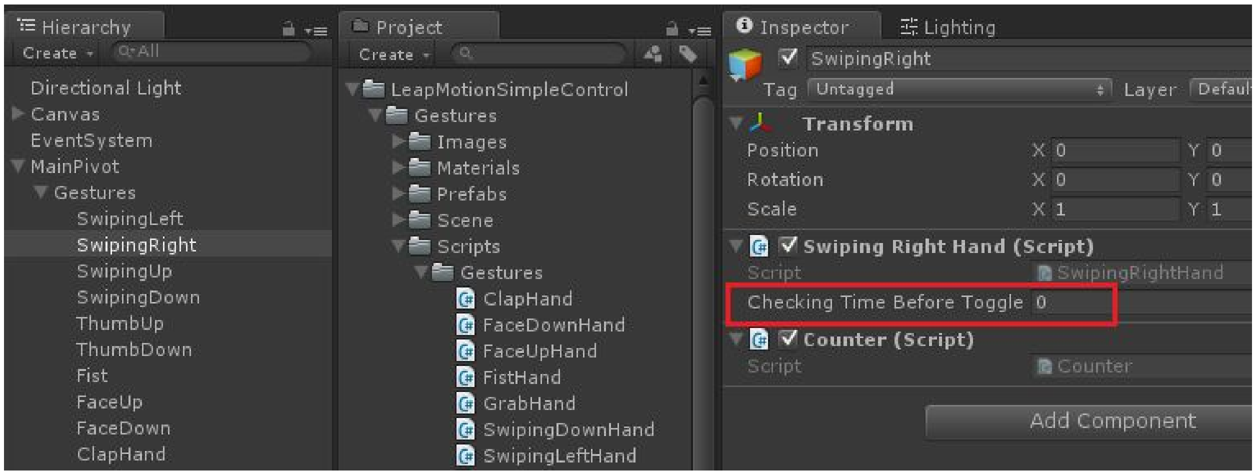
2. Adjustable parameters related to gesture system

-**TimeBetween2Gestures:** In gameobject MainPivot>Gestures>GestureManager script, there is a param called *TimeBetween2Gestures*, which allows to set up the delay transition time in seconds to perform a new gesture.



Gestures
Tag Untagged Layer Defa
Rect Transform
Pos X 0 Pos Y 0
Width 100 Heigh 100
Anchors
Pivot X 0.5 Y 0.5
Rotation X 0 Y 0
Scale X 1 Y 1
Gesture Manager (Script)
Script GestureManager
Leap Hand Provider None (Leap Provider)
Time Between 2 Gestures 1

-CheckingTimeBeforeToggle: Each specific gesture script inherited a variable called *CheckingTimeBeforeToggle* which allows to check timer to detect this gesture. For example, faceup gesture require your hand must be stay stationary within the *CheckingTimeBeforeToggle* before toggle the recognition system.



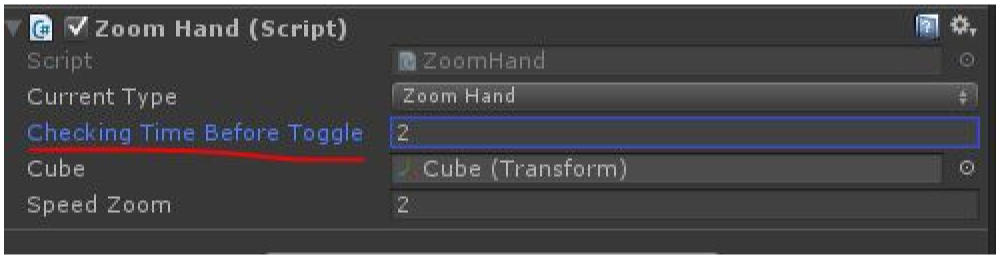
SwipingRight
Tag Untagged Layer Defaul
Transform
Position X 0 Y 0
Rotation X 0 Y 0
Scale X 1 Y 1
Swiping Right Hand (Script)
Script SwipingRightHand
Checking Time Before Toggle 0
Counter (Script)
Script Counter

3. Predefine hand gestures

Each gesture is implemented in separate class within its name and inherited the base class *BehaviourHand:*

- SwipingLeft
- SwipingRight
- SwipingUp
- SwipingDown
- ThumbUp
- ThumbDown
- Fist
- FaceUp
- FaceDown
- ClapHand
- Grab
- Throw (*)
- RotateHand
- ZoomHand (**)

(*)Throw, Grab, RotateHand, ZoomHand gesture may overlap other gestures, thus, it need to disable/enable gameobject separately.
(**) ZoomHand is a customized and enhanced from the real time hand grab strength data to scale cube in the demo scene “Demo_Rotate3D”. To trigger, user have to open full hand and stay for 2 seconds and open/close the hand slowly to adjust the scale of 3D object.

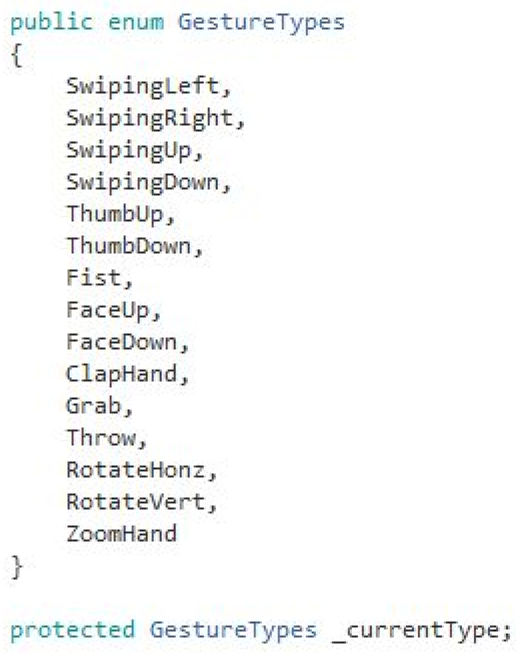


Zoom Hand (Script)

Script	ZoomHand
Current Type	Zoom Hand
Checking Time Before Toggle	2
Cube	Cube (Transform)
Speed Zoom	2

4. Add new gesture type

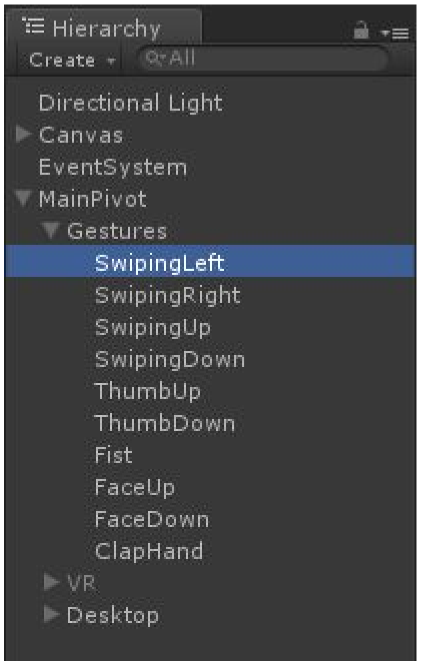
- **Step1**: Add new GestureType enum in *GestureManager* script.



```
public enum GestureTypes
{
    SwipingLeft,
    SwipingRight,
    SwipingUp,
    SwipingDown,
    ThumbUp,
    ThumbDown,
    Fist,
    FaceUp,
    FaceDown,
    ClapHand,
    Grab,
    Throw,
    RotateHonz,
    RotateVert,
    ZoomHand
}

protected GestureTypes _currentType;
```

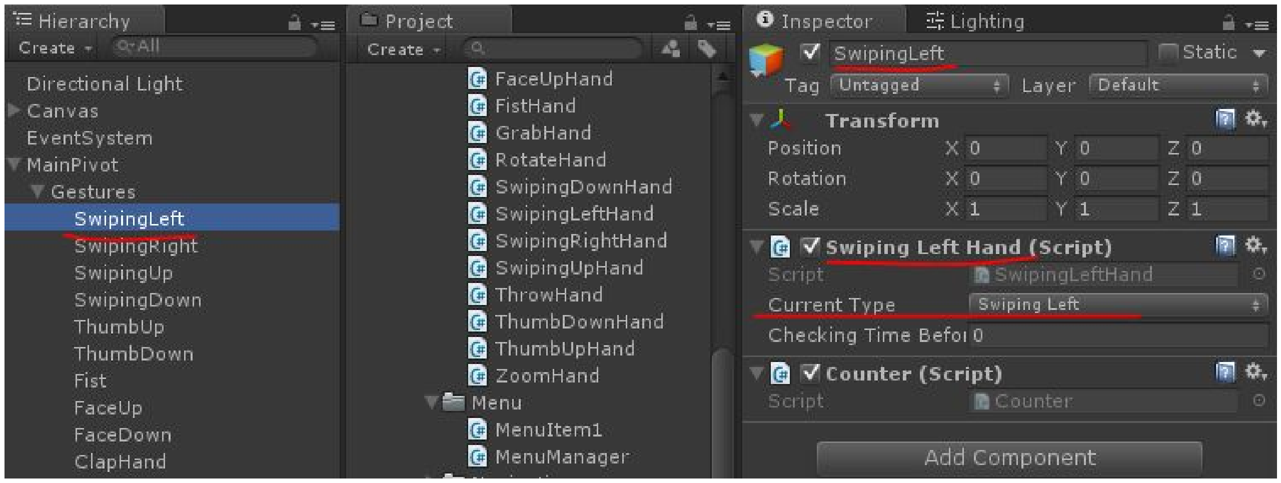
- **Step2:** Add new child gameobject of *MainPivot>Gestures* with the same name as GestureType



```
Hierarchy
Create ▾  All

  Directional Light
▶ Canvas
  EventSystem
▼ MainPivot
    ▼ Gestures
        SwipingLeft
        SwipingRight
        SwipingUp
        SwipingDown
        ThumbUp
        ThumbDown
        Fist
        FaceUp
        FaceDown
        ClapHand
    ▶ VR
    ▶ Desktop
```

- **Step3:** Create new script to implement a specific toggle condition method, then attach this script with the new gameobject which was created at Step2. This new script must inherit the *BehaviorHand*



```
Directional Light
Canvas
EventSystem
MainPivot
  Gestures
    SwipingLeft
    SwipingRight
    SwipingUp
    SwipingDown
    ThumbUp
    ThumbDown
    Fist
    FaceUp
    FaceDown
    ClapHand
```

```
FaceUpHand
FistHand
GrabHand
RotateHand
SwipingDownHand
SwipingLeftHand
SwipingRightHand
SwipingUpHand
ThrowHand
ThumbDownHand
ThumbUpHand
ZoomHand
Menu
  MenuItem1
  MenuManager
```

```
SwipingLeft
Tag Untagged    Layer Default
Transform
Position  X 0  Y 0  Z 0
Rotation  X 0  Y 0  Z 0
Scale     X 1  Y 1  Z 1
Swiping Left Hand (Script)
Script          SwipingLeftHand
Current Type    Swiping Left
Checking Time Befoi 0
Counter (Script)
Script          Counter
```

- **Step4:** Implement the override method *CheckConditionGesture* in the new script with your specific condition. Remember to implement the *_currentType* mapping to your new *GestureType* or change the *CurrentType* in the Inspector.

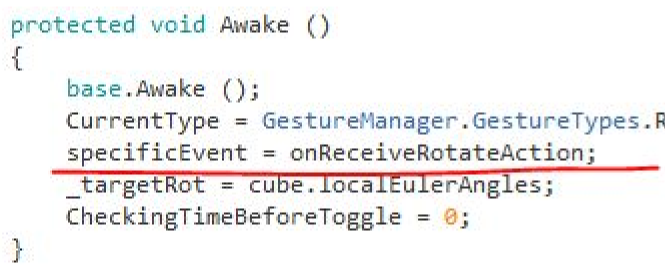
```
// Use this for initialization
void Start ()
{
    _currentType = GestureManager.GestureTypes.SwipingLeft;
}

// Update is called once per frame
void Update ()
{

}

protected override bool checkConditionGesture ()
{
    Hand hand = GetCurrent1Hand ();
    if (hand != null) {
        if (isOpenFullHand (hand) && isMoveLeft (hand)) {
            return true;
        }
    }
    return false;
}
```

- **Step5:** (Optional) If you would like to add your own event trigger to the current gesture, please add the to the *specificEvent* like RotateHand script.



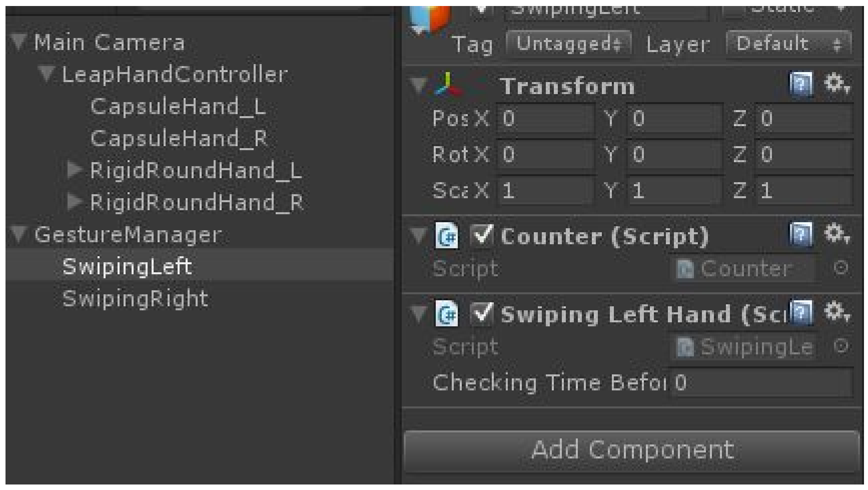
```
protected void Awake ()
{
    base.Awake ();
    CurrentType = GestureManager.GestureTypes.RotateVert;
    specificEvent = onReceiveRotateAction;
    _targetRot = cube.localEulerAngles;
    CheckingTimeBeforeToggle = 0;
}
```

○

5. How to use pre-defined gesture in your own project

Step1: Create new scene, add LeapHandController prefab from the Leap Motion unity package as children of your main camera

Step2: Create new gameobject contain script “GestureManager.cs”. For each specific gesture your would like to trigger in your project, please add the corresponding gesture checking script in a child gameobject of GestureManager. Please consider to adjust the “CheckingTimeBeforeToggle” variable for each script depending on your needs.



```
▼ Main Camera
  ▼ LeapHandController
      CapsuleHand_L
      CapsuleHand_R
    ▶ RigidRoundHand_L
    ▶ RigidRoundHand_R
▼ GestureManager
    SwipingLeft
    SwipingRight
```

Tag: Untagged — Layer: Default

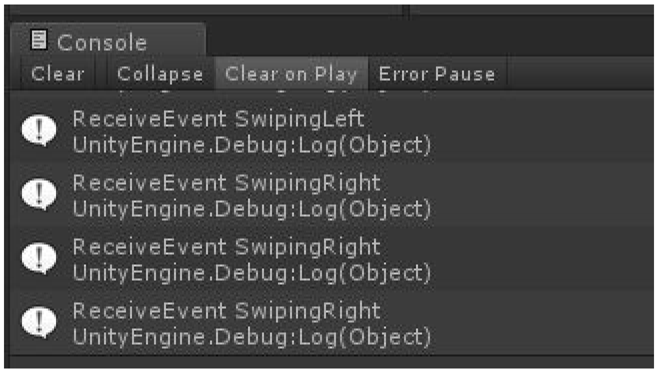
Transform
Pos X 0 Y 0 Z 0
Rot X 0 Y 0 Z 0
Sca X 1 Y 1 Z 1

Counter (Script)
Script: Counter

Swiping Left Hand (Sc
Script: SwipingLe
Checking Time Befo: 0

Add Component

That’s all! Just simple 2 steps to made a simple recognition, no need prefab as the gesture recognition is simple based on script. You will observe in the Console when your gesture triggered.

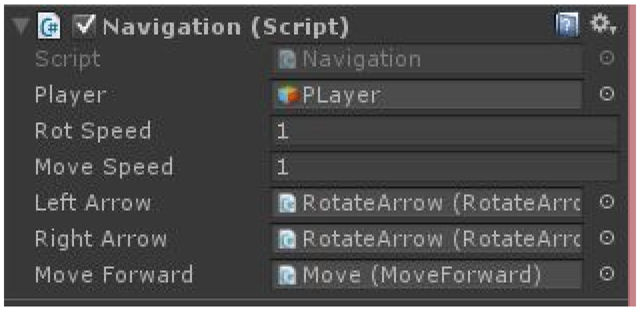


```
ReceiveEvent SwipingLeft
UnityEngine.Debug:Log(Object)
ReceiveEvent SwipingRight
UnityEngine.Debug:Log(Object)
ReceiveEvent SwipingRight
UnityEngine.Debug:Log(Object)
ReceiveEvent SwipingRight
UnityEngine.Debug:Log(Object)
```

Please see the Demo_Beginner scene for more information.

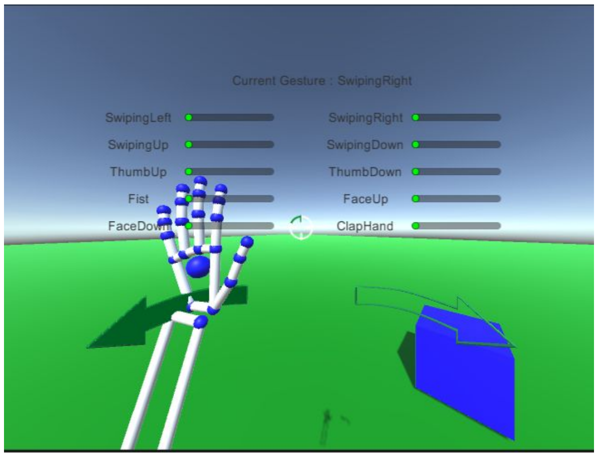
6. VR Player Control

VR player control allow to navigation in 3D space using leap hand to control the rotation and the facing direction to move forward. You can freely adjust the speed of move forward or rotation in Unity editor for better experience.



Script	Navigation
Player	PLayer
Rot Speed	1
Move Speed	1
Left Arrow	RotateArrow (RotateArrc
Right Arrow	RotateArrow (RotateArrc
Move Forward	Move (MoveForward)

The navigation system has two parts including Move and Rotation. The Move part contains a loading/indicator bar to be fulfilled if the player keep looking direction steadily for 2 seconds. When it is fulfilled, the player will move forward following its facing direction. Otherwise, the Rotation part contains two arrow which allow the leap hand to touch in 3D space. Left or right rotation is based on the touching left or right arrow.



The system is designed for specific VR player controller, when you putting on a headset, the only way to interact with the synthetic environment is facing direction and leap hand controller.