VRTRIX™ Digital Glove Manual

Ⅰ. Plugin Setup Steps

1. Go to the root directory where your UE4 Project located.

   If you already had a "**Plugins**" folder under this directory, just drag the "**DigitalGloves**" folder into this folder.

   Otherwise, create a new folder named "Plugins", drag the "**DigitalGloves**" folder into this "**Plugins**"

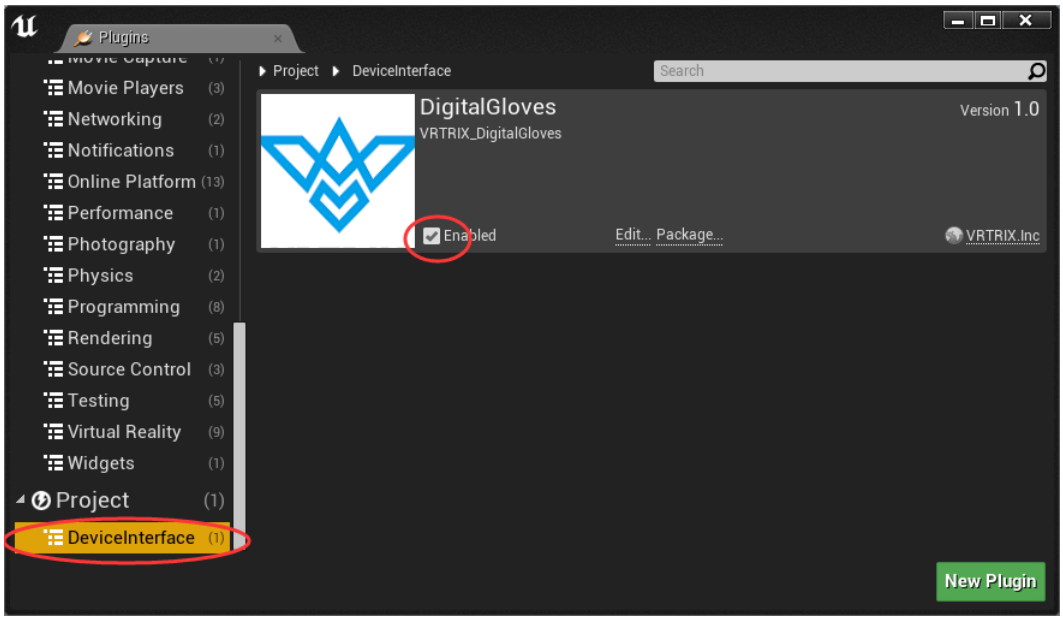
2. Export the "**VRTRIX**" folder from our demo project to your "**Content**" folder under your Project root directory.

   Note: please make sure you don't have C++ Class Named,

   "**ComportDateParseAsyncTask**" "**UMyHandComponent**" "**Hand**"

   If you do have, please modify your local class name to avoid name confliction.

3. Open you Project. Go to **Edit-> Plugins**, Scroll down to the bottom. Under the **Project** Category, there this a Plugin named **DeviceInterface**, enable it by clicking the **Enabled** box.



After steps 1,2,3 finished successfully, you are all set to explore all the features of **VRTRIX™** Digital Gloves.

If you encountered any issues while setting up the UE4 Plugin, please contact us by email or Wechat.

Technical Support Email: **info@vrtrix.com**

Wechat Offical Accounts ID: **VRTRIX**

We highly recommend you read and understand the simple blueprints in our example before you start developing your own application on our gloves.

II. Project Example && Features Explain

Project Example:

Main Resources included in this Project Example.

1. Demo Map

2. Two Pawn Blueprints represent two hands

3. An Actor Blueprints

4. Meshes and Materials

What we can do in this example?

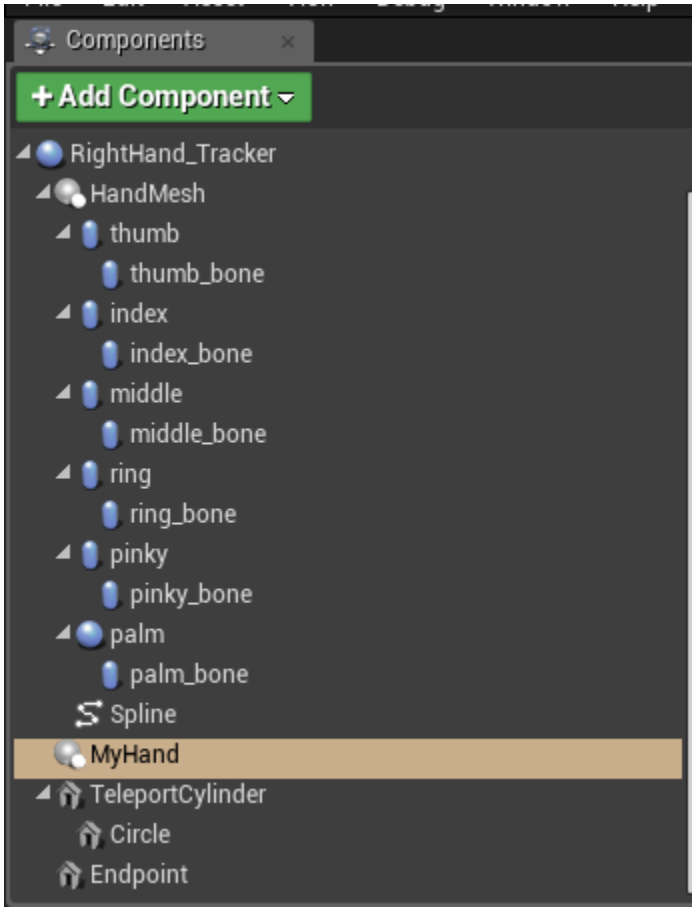
1. Customize your own gestures for each hand
2. Teleporting with your hand
3. Physical Interact (grab, hit, etc.) with moveable game objects with your hand
4. Fire Magic with your hand

Features Explain:

Pawn Blueprints **Left_Hand**, **Right_Hand**:

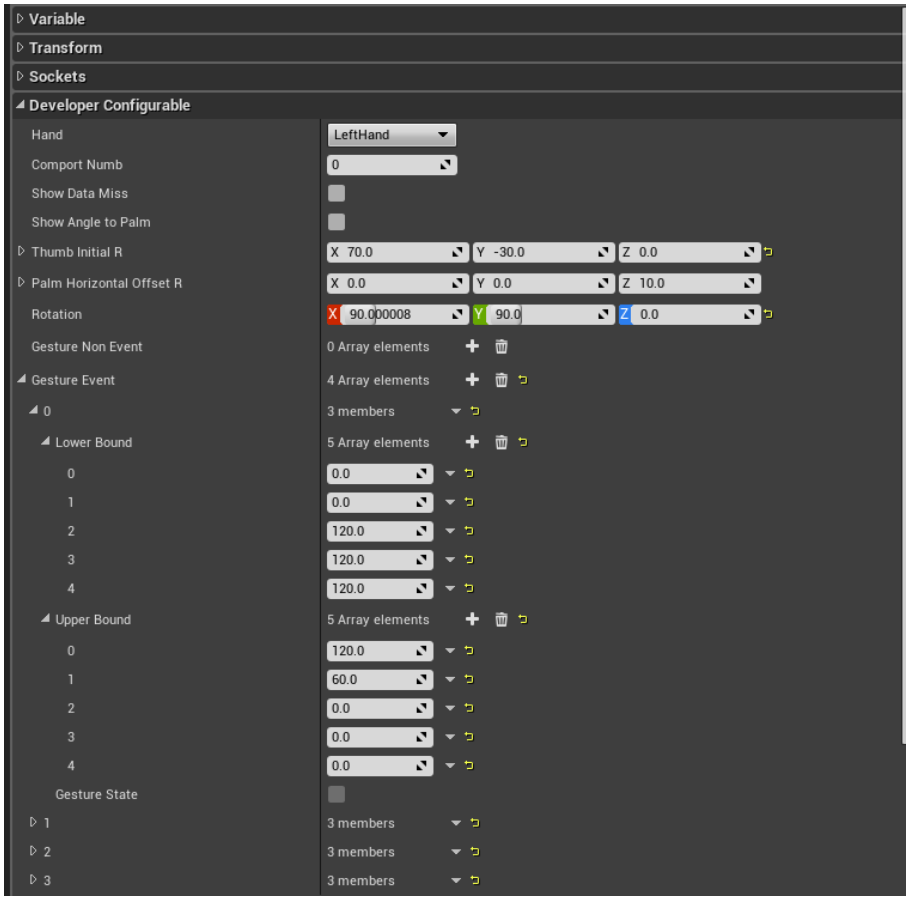
Easy to use, create a reference component player character then spawns the hand and attach it to a reference component of your player character.

Open the Blueprints of **Left_Hand** you can see all the components. The essential component is **MyHand** component, which is a Custom **SceneComponent** Class in our Plugin and will support most features of our glove.



In the Detail Panel for **MyHand**, you can see a category name Developer Configurable.

This is the most import features that a UE4 Developer will deal with.



If you want to use our hand mesh, you don't need change most parts of this category.

The main area you will modify is **Gesture_Non_Event** and **Gesture_Event** arrays.

These arrays contain the custom define gesture.

Gesture_Non_Event: If defined, **Gesture_State** will be recalculated every frame, when

hand pose fall into the area gesture defined, **Gesture_State** will be set **TRUE**, otherwise, it will be set **FALSE**. Only **Gesture_State** accessible in blueprint, no event will be generated.

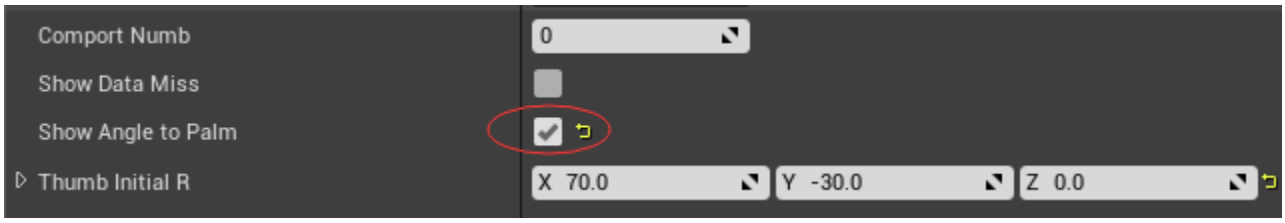
Gesture_Event: Most feature same as **Gesture_Non_Event**, but gesture defined in this array will also generate Trigger and Release Event when hand **Gesture_State** Flips.

How to define gestures?

Each gesture contains two 5-elements boundary arrays, which denote the angle boundary of finger to palm.0 to 4 correspond to thumb to pinky in sequence. Only when all the 5 fingers' angles to palm are inside the corresponding (low_bound, upper_bound) area, the gesture state will be set TRUE, otherwise it will be set FALSE.

Note: 0 in the array means the boundary is don't care for the corresponding finger. Which means the bounds could be (A, B), (A, +∞), (-∞, B) or (-∞, +∞).

You can click the **Show_Angle_To_Palm** box to print out all the angles in output log window every frame, which could be your reference when you are defining your own gesture or debug the gestures.

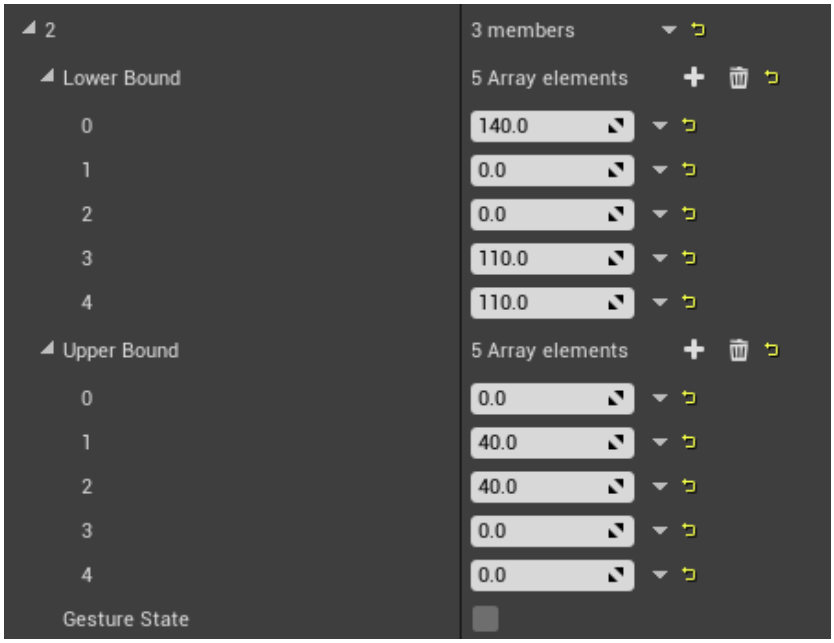


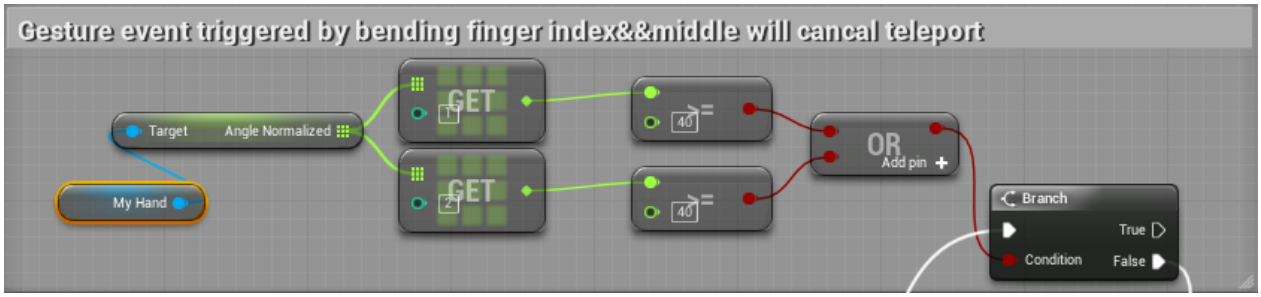
You can also access the angles to palm array in blueprint. Which can provide the information which boundary our hand is crossing when **Gesture_State** flipped. Then we

can do different operation on a same event.

For example: we can define a gesture to fire a magic. When the gesture is triggered, we start to accumulate power. While, for gesture release event, we can do differently by accessing the angles to palm data. If hand **Gesture_State** flipped when cross Upper_bound, we can cancel the magic; if hand **Gesture_State** flipped when cross Lower_bound, we can fire the magic.

Like our teleport execution:





For other parts (variables, functions, etc.) of our example. We also have explaining

comments for all of them. Please refer to:

Plugins\Runtime\DigitalGloves\Source\DigitalGloves\Public\ MyHandComponent.h

The comments are very straight forward and easy to understand.

If you have any problem while dealing with glove or our UE4 sdk , please contact us by email or Wechat at any time.

Technical Support Email: **info@vrtrix.com**

Wechat Offical Accounts ID: **VRTRIX**