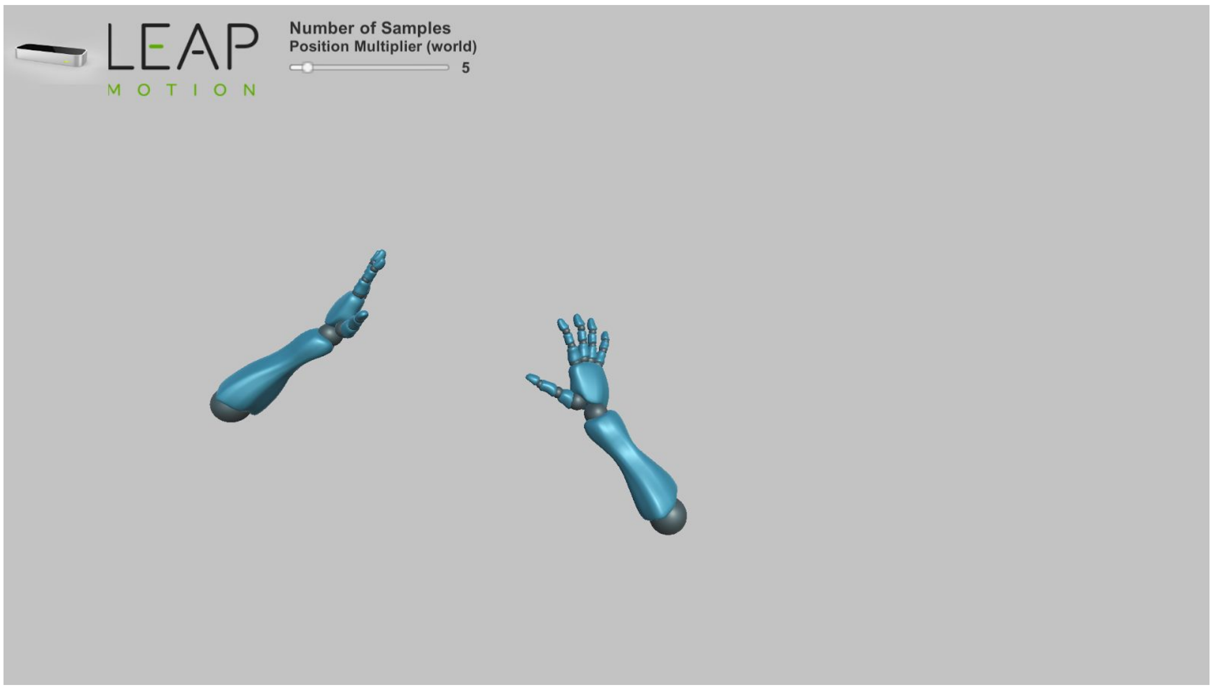
Leapmotion Demo

Description

Any question contact: mobile.neurorehablab@gmail.com
The leapmotion Demo package is a package to be used in Unity3D. This demo allows you to fully integrate your leapmotion with unity3D as long as you use the Reh@panel protocol to send its information. There is already a tool (Reh@panel) that sends the leapmotion information using this protocol.



User Manual

Importing the package

1. Create a new project
2. Import the package LeapmotionDemo.unitypackage
    a. Assets -> Import package -> Custom Package

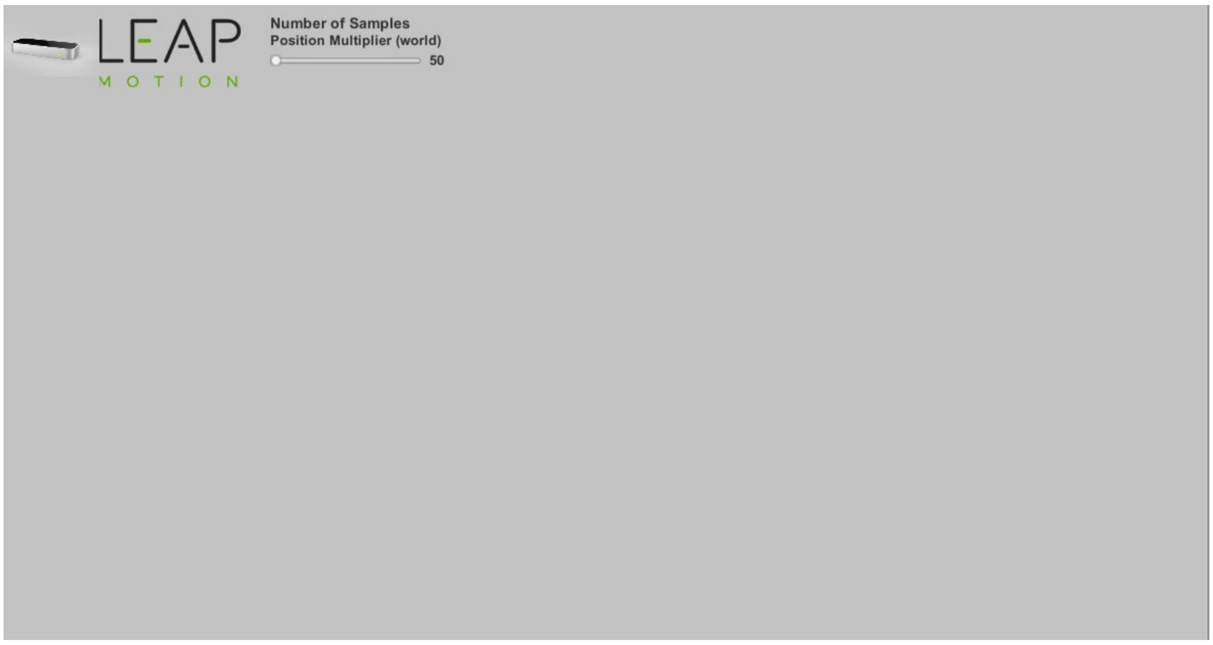
Requirements to use the package

1. Make sure you have the Leapmotion SDK installed - Download Link here
2. Plug in your leapmotion to the PC
3. Make sure that it is sending information by UDP to port 1202 (To change this port see below).
    a. You can use the Reh@panel to do this

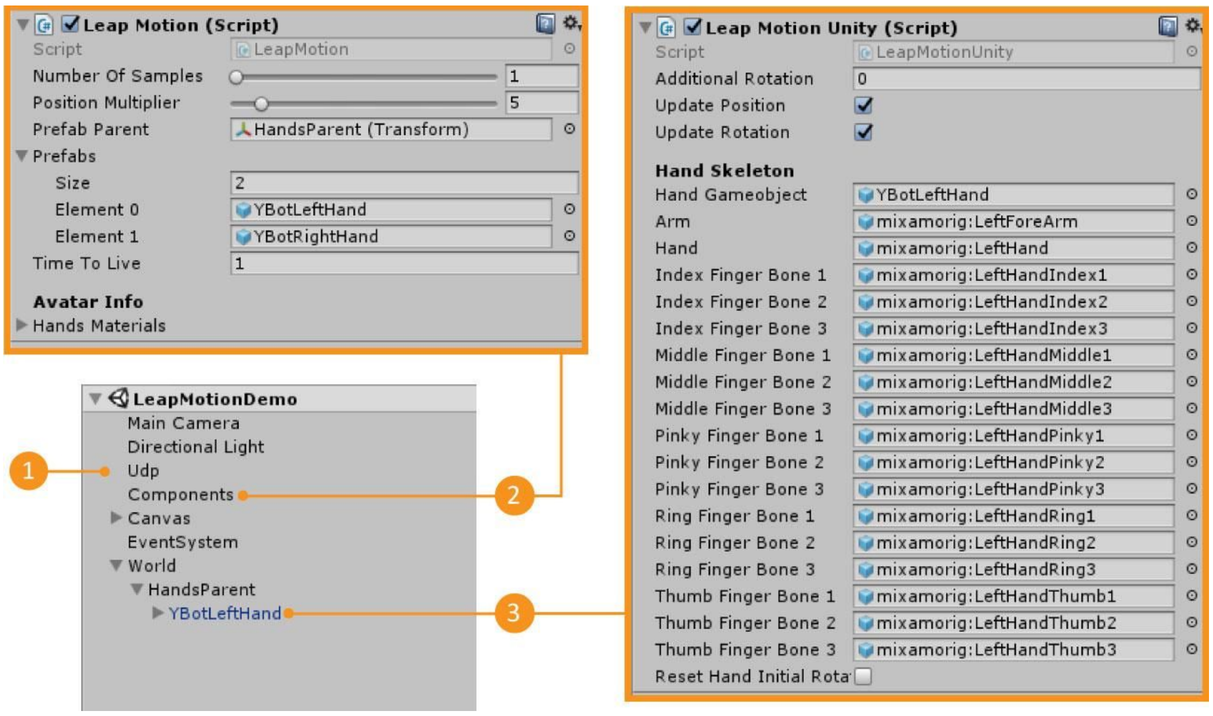
Testing the scene

1. Open the Scene
    a. Neurehab -> Demo Leapmotion-> Scenes -> LeapmotionDemo

2. Make sure you fulfill all the requirements
3. Press Play



Understanding the Demo Scene



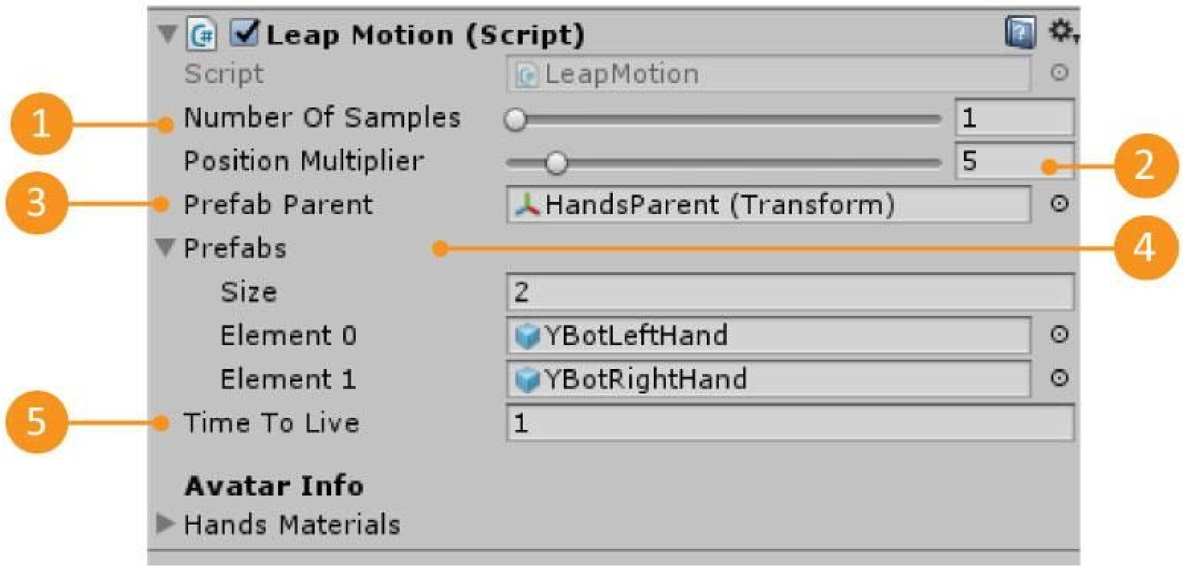
Leap Motion (Script)
Script: LeapMotion
Number Of Samples: 1
Position Multiplier: 5
Prefab Parent: HandsParent (Transform)
Prefabs
Size: 2
Element 0: YBotLeftHand
Element 1: YBotRightHand
Time To Live: 1
Avatar Info
Hands Materials

LeapMotionDemo
Main Camera
Directional Light
Udp
Components
Canvas
EventSystem
World
HandsParent
YBotLeftHand

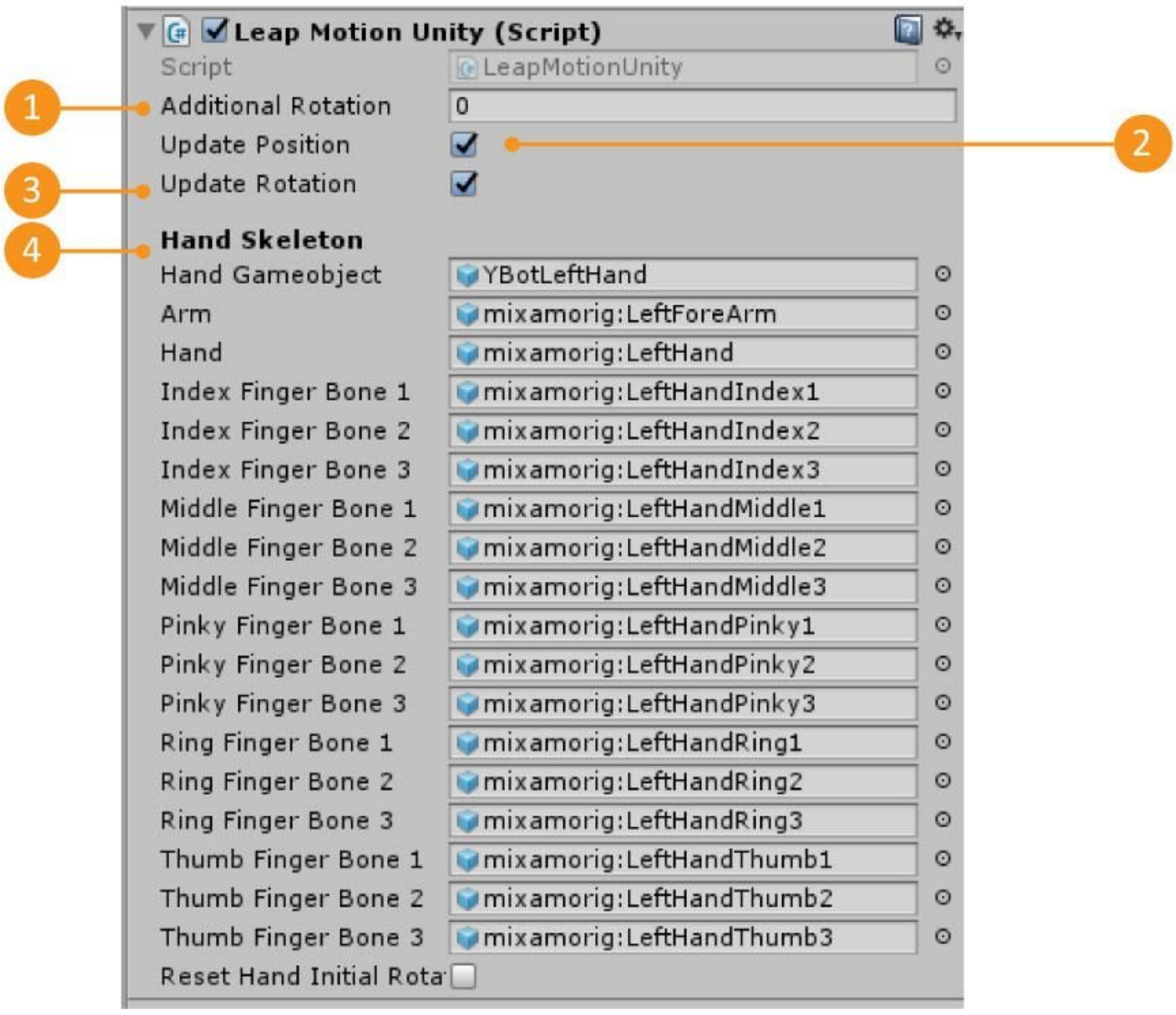
Leap Motion Unity (Script)
Script: LeapMotionUnity
Additional Rotation: 0
Update Position: ✓
Update Rotation: ✓
Hand Skeleton
Hand Gameobject: YBotLeftHand
Arm: mixamorig:LeftForeArm
Hand: mixamorig:LeftHand
Index Finger Bone 1: mixamorig:LeftHandIndex1
Index Finger Bone 2: mixamorig:LeftHandIndex2
Index Finger Bone 3: mixamorig:LeftHandIndex3
Middle Finger Bone 1: mixamorig:LeftHandMiddle1
Middle Finger Bone 2: mixamorig:LeftHandMiddle2
Middle Finger Bone 3: mixamorig:LeftHandMiddle3
Pinky Finger Bone 1: mixamorig:LeftHandPinky1
Pinky Finger Bone 2: mixamorig:LeftHandPinky2
Pinky Finger Bone 3: mixamorig:LeftHandPinky3
Ring Finger Bone 1: mixamorig:LeftHandRing1
Ring Finger Bone 2: mixamorig:LeftHandRing2
Ring Finger Bone 3: mixamorig:LeftHandRing3
Thumb Finger Bone 1: mixamorig:LeftHandThumb1
Thumb Finger Bone 2: mixamorig:LeftHandThumb2
Thumb Finger Bone 3: mixamorig:LeftHandThumb3
Reset Hand Initial Rota

1 - UDP	Gameobject that has the UDP connection Information. You can change the UDP port here
2 - Components	Gameobject that has the Leapmotion.cs script

3 - Leapmotion Prefab	Prefab that has the LeapmotionUnity.cs script. This prefab will only appear in the scene when you have pressed Play


Leap Motion (Script)
Script LeapMotion
Number Of Samples 1
Position Multiplier 5
Prefab Parent HandsParent (Transform)
Prefabs
Size 2
Element 0 YBotLeftHand
Element 1 YBotRightHand
Time To Live 1
Avatar Info
Hands Materials

1 - Number Of Samples	The number of data samples to save for each data input. In the end returns the average of that data input.
2 - Position Multiplier	Multiplies the position that is receiving for this value. This is useful when you have a game world scale that differs from the real world.
3 - Prefab parent	The Gameobject parent where the device prefabs are going to be instantiated Each instantiated prefab represents a Leapmotion that is sending data by UDP
4 - Prefabs	The list of prefabs that can be instantiated.
5 - Time to Live	The maximum time in seconds that a prefab can wait for new data before it is destroyed



Leap Motion Unity (Script)

Script	LeapMotionUnity
Additional Rotation	0
Update Position	✓
Update Rotation	✓

Hand Skeleton

Hand Gameobject	YBotLeftHand
Arm	mixamorig:LeftForeArm
Hand	mixamorig:LeftHand
Index Finger Bone 1	mixamorig:LeftHandIndex1
Index Finger Bone 2	mixamorig:LeftHandIndex2
Index Finger Bone 3	mixamorig:LeftHandIndex3
Middle Finger Bone 1	mixamorig:LeftHandMiddle1
Middle Finger Bone 2	mixamorig:LeftHandMiddle2
Middle Finger Bone 3	mixamorig:LeftHandMiddle3
Pinky Finger Bone 1	mixamorig:LeftHandPinky1
Pinky Finger Bone 2	mixamorig:LeftHandPinky2
Pinky Finger Bone 3	mixamorig:LeftHandPinky3
Ring Finger Bone 1	mixamorig:LeftHandRing1
Ring Finger Bone 2	mixamorig:LeftHandRing2
Ring Finger Bone 3	mixamorig:LeftHandRing3
Thumb Finger Bone 1	mixamorig:LeftHandThumb1
Thumb Finger Bone 2	mixamorig:LeftHandThumb2
Thumb Finger Bone 3	mixamorig:LeftHandThumb3
Reset Hand Initial Rota	☐

1 - Additional Rotation	Additional rotation to add to the bones
2 - Update Position	True if you want to update the object position according to what it is being received
3 -Update Rotation	True if you want to update the object rotation according to what it is being received
4 - Available Data	All the data that we are receiving from a Leapmotion. You will notice that leapmotion sends data from the joints and not from the bones themselves. For example, the palm is the articulation (wrist) but the bone that is referenced is the hand. If you want to add a new avatar, you should always take this into account.

How to access the device data information

The data is accessed through the LeapmotionUnity.cs script. If you open this script, in the UpdateHandRotation and UpdateHandPosition functions you will notice that the Leapmotion prefab rotation and position values are updated there.

```
public void UpdateHandRotation()
{
   if (Arm != null)
   {
        Arm.transform.rotation =
           HandGameobject.transform.rotation * //reference object gameobject (in this case the hand)
           GenericDeviceData.GetRotation(LeapMotionBones.forearm.ToString()) * //rotation that comes
from leapmotion (world rotation)
           _initialRotations[(int) LeapMotionBones.forearm]; //initial rotation of the forearm
inside unity

....
}
```

Accessing the values

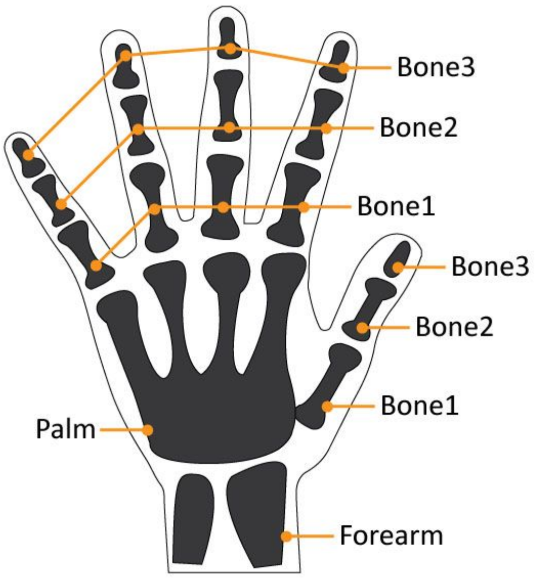
To be able to access any values data that the Leapmotion is sending, you will need to know two things: the label of the information you want to access and the type of that information. Then, using the LeapmotionUnity.cs script, you can find all this information by accessing the GenericDeviceData Dictionaries as shown below:

`GenericDeviceData.Get`**`[INFORMATION_TYPE]`**`(“`**`[INFORMATION_LABEL]`**`”)`

For example, to access the Leapmotion hand rotation value which is labeled as ‘palm’ and is of the type rotation, you can:
`GenericDeviceData.Get`**`Rotation`**`(“`**`palm`**`”)`

Current Leapmotion protocol values

Information Label	Information Type
forearm	position
	rotation
palm	position
	rotation
thumb_bone1	position
	rotation
thumb_bone2	position



	rotation
thumb_bone3	position
	rotation
index_bone1	position
	rotation
index_bone2	position
	rotation
index_bone3	position
	rotation
middle_bone1	position
	rotation
middle_bone2	position
	rotation
middle_bone3	position
	rotation
ring_bone1	position
	rotation
ring_bone2	position
	rotation
ring_bone3	position
	rotation
pinky_bone1	position
	rotation
pinky_bone2	position
	rotation
pinky_bone3	position
	rotation

Accessing the parameters

In the Rehapanel protocol, parameters are things that always present no matter if it is sending a position, rotation, value or bool. For more information on the Reh@Panel read here.
To be able to access any parameters data that the Leapmotion is sending, you will need to know the parameter name. Then, using the LeapmotionUnity.cs script, you can find all this information by accessing the GenericDeviceData.GetParameters("parameterNameHere") as shown below:

```
GenericDeviceData.GetParameters("[PARAMETER_NAME]")
```

For example, to access the Leapmotion hand side parameter which is labeled as "side", you can do:

```
var handSide = GenericDeviceData.GetParameter("side");
```

Current Leapmotion parameters

Name	**Description**
Id	The GenericDeviceData ID.
Side	Which hand the GenericDeviceData belogs to. Can be **left** or **right**