



NOTIOM MOCAP LIVE HANDBOOK

NoitomMocapLive_Unity_v1.1.0.0

Noitom Technology Co.Ltd: <http://noitom.com.cn/>
Mar, 1st, 2022

TABLE OF CONTENTS

TABLE OF CONTENTS.....	2
1. Introduction.....	3
2. Import Package.....	4
3. Structure Overview.....	5
3.1 Example Scenes.....	5
3.2 Core Scripts Overview.....	5
3.3 Public Variables of Neuronanimatorinstance.cs.....	7
4. Data streaming setup in Axis.....	7
4.1 Axis Neuron/Axis Neuron Pro Setup.....	7
4.2 Axis Studio Setup.....	8
5. Neuron Skeleton Tools.....	9
6. How to configure a character model.....	11
7. How to Apply Real-time Data from Axis.....	13
8. HMD INTEGRATION.....	17
9. Setup in AHM/Alice.....	17
9.1 AHM/Alice Data Streaming Setup.....	17
9.2 AHM/Alice Device Setup.....	18
10. How to Apply AHM/Alice Device Data.....	19
11. How to Apply FBX file Data from Axis.....	19
11.1 Humanoid Animation type.....	19
11.2 Generic Animation Type.....	23
12. APPENDIX A: SKELETON MAPPING.....	24
12.1 PN Studio skeleton structure mapping.....	24
12.2 PN/PN Pro skeleton structure mapping.....	25
13. APPENDIX B: SKELETON BONES.....	26
14. APPENDIX C: BINARY DATA SEQUENCE.....	28

1. Introduction

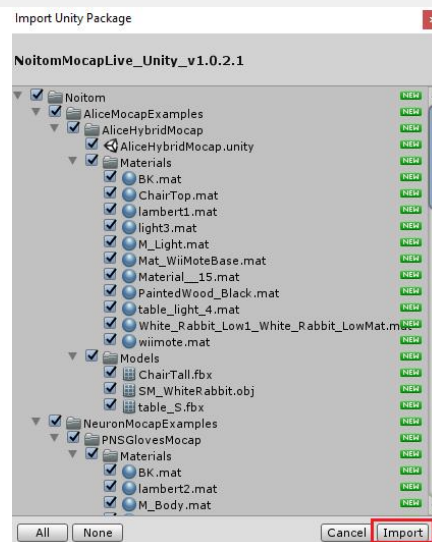
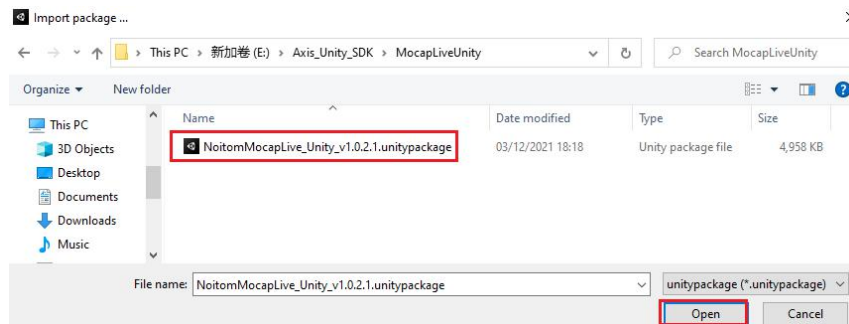
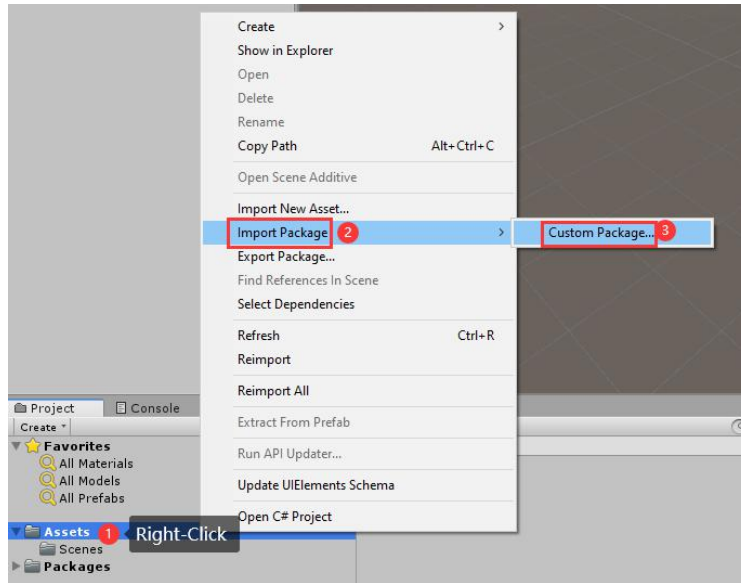
This document should help you get familiar with real-time motion capture data reading and how to use fbx file exported from Axis Neuron & Axis Studio inside the game engine Unity 3D. Axis Neuron & Axis Studio not only allows the export of motion capture data but also be able to stream motion capture data in real-time to third party applications, making the data available to drive characters and any GameObject in animation.

The data stream to the game engine is identical regardless if you use recorded or live motion data. This allows you to record certain actions and use them to test before putting on the full system and doing a live test. You can have multiple actors at the same time inside Axis and have them all streamed into Unity. The data stream is based on the BVH structure including header information and body dimension data is available from Axis via a command interface. Because motion-data is streamed over network the computer running Axis Neuron & Axis Studio doesn't necessarily have to be the same computer running Unity.

If you find bugs or need help please contact us at Noitom_service@noitom.com

2. Import Package

Choose Assets > Import Package > Custom Packages to import **NoitomMocapLiveUnity.unitypackage**.

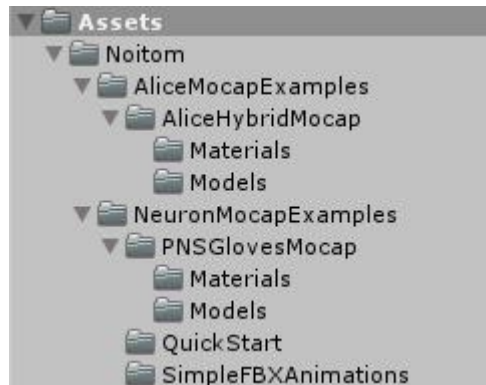


3. Structure Overview

3.1 Example Scenes

Here are some pre-made examples explained of the SDK package:

1. **QuickStart:** The simplest scene to show how to use this SDK.
Use NeuronTransformInstance.cs script to drive avatar fbx.
2. **PNSGlovesMocap:** A simple example of how to drive hands only part of avatar fbx.
3. **SimpleFBXAnimations:** A simple example of how to use exported fbx animations file from Axis Neuron/Axis Studio to drive avatar fbx.
4. **AliceHybridMocap:** A simple example of how to use hybrid props from Alice in Unity.
Use NeuronTracker.cs script to motion Props fbx



3.2 Core Scripts Overview

1. **Assets/Noitom/Scripts/Mocap/MocapApiManager.cs**
Manages connections with Axis, get motion data from MocapApi. Users does not need to pay attention to this script
2. **Assets/Noitom/Scripts/Mocap/NeuronActor.cs**
Data class to store only the most recent motion data frame, also provides methods to parse the received motion data which is received as float values from the network.
3. **Assets/Noitom/Scripts/Mocap/NeuronSourceManager**
Neuronsourcemanager plays the role of connecting data sources. Each connection needs to be added to the Neurosourcemanager instance, and hang props and characters under it as child nodes.
4. **Assets/Noitom/Scripts/Mocap/NeuronSource.cs**

NeuronSource manages instances of NeuronActor with two dictionaries called activeActors and suspendedActors. NeuronSource monitors the latest timestamp update inside NeuronActor in the OnUpdate method and use a threshold to judge if any actor is lost (number of actors in Axis Neuron has changed or the connection was lost completely). When this happens NeuronSource will add or remove actors between the two dictionaries and notify NeuronActor.

5. **Assets/Noitom/Scripts/Mocap/NeuronInstance.cs**

Base class for all kinds of instances for receiving motion data.

Inherits from UnityEngine.MonoBehaviour. NeuronInstance provides callbacks for state changes and the receiving of mocap info from a NeuronActor instance which was bound to this instance by connect or other methods. This class is not intended to be used directly but can be inherited to provide custom methods to apply motion data, handle states change and mocap info.

6. **Assets/Noitom/Scripts/Mocap/NeuronAnimatorInstance.cs**

Inherited from NeuronInstance. Provides custom methods to apply Neuron motion data to the transform components of the bones bound in the Unity animator component. Needs a humanoid skeleton setup to work properly.

7. **Assets/Noitom/Scripts/Mocap/NeuronAnimatorPhysicalReference.cs**

Data class for initialization and cleanup of a reference skeleton used for motions based upon Unity's rigidbody component. Used by NeuronAnimatorInstance if physics toggle is enabled.

8. **Assets/Noitom/Scripts/Mocap/NeuronTransformsInstance.cs**

Inherited from NeuronInstance. Provides custom methods to apply Neuron motion data directly to transform components. Use this for non-humanoid skeletons or skeletons with more bones than the default setup used in Unity.

9. **Assets/Noitom/Scripts/Mocap/NeuronTransformsPhysicalReference.cs**

Data class for initialization and cleanup of a reference skeleton used for motions based upon Unity's rigidbody component. Used by NeuronTransformsInstance if physics toggle is enabled.

10. **Assets/Noitom/Scripts/Mocap/ NeuronRigidbody.cs**

Use this script to receive hybrid mocap props data from Axis Studio

11. **Assets/Noitom/Scripts/Mocap/ NeuronRigidbodyListHelper.cs**

Use this script to show how many hybrid mocap props in Axis Studio and show their ids.

12. **Assets/Noitom/Scripts/Mocap/NeuronTracker.cs**

Use this script to receive rigidbody props data from Alice.

13. **Assets/Noitom/Scripts/Utilities/BoneLine.cs**

Utility class using a line renderer to draw bone lines.

14. **Assets/Noitom/Scripts/Utilities/BoneLines.cs**

Utility class for Neuron editor to add or remove BoneLines.

15. **Assets/Noitom/Scripts/Utilities/BoneRigidbody.cs**

Utility class for Neuron editor to add or remove Rigidbodies.

16. Assets/ Noitom /Scripts/Utilities/FPSCounter.cs

Utility class to calculate the FPS (Frames-Per-Second).

3.3 Public Variables of Neuronanimatorinstance.cs

1. **Actor ID** is character's id number. Default value is 0 which is the first character in Character panel of Axis
2. **Skeleton Type**
 - 1) Select **Perception Neuron Studio**, it means the script will use the **PN Studio** skeleton structure which includes 3 joints of spine, and 2 joints of neck.



- 2) Select **Perception Neuron**, it means the script will use the **PN/PN Pro** skeleton structure which includes 4 joints of spine and 1 joint of neck.

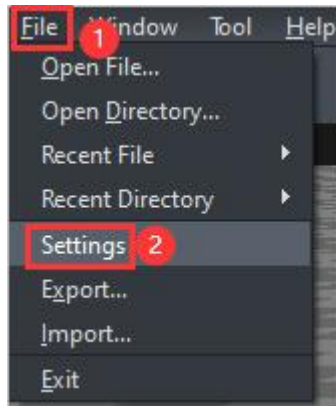


3. **Bound Animator** the Animator component this instance should use for the received motion data. You can use this if you don't want to keep the script on the same GameObject as the animator component.
4. **Motion Update Method** tells the instance if it should use rigidbody functions provided by Unity to move and rotate each bone. The default method is to apply the received float values directly to the transform components of each bone.
5. **Disable bone movement** tells the instance if it should apply avatar's bone's transform movement

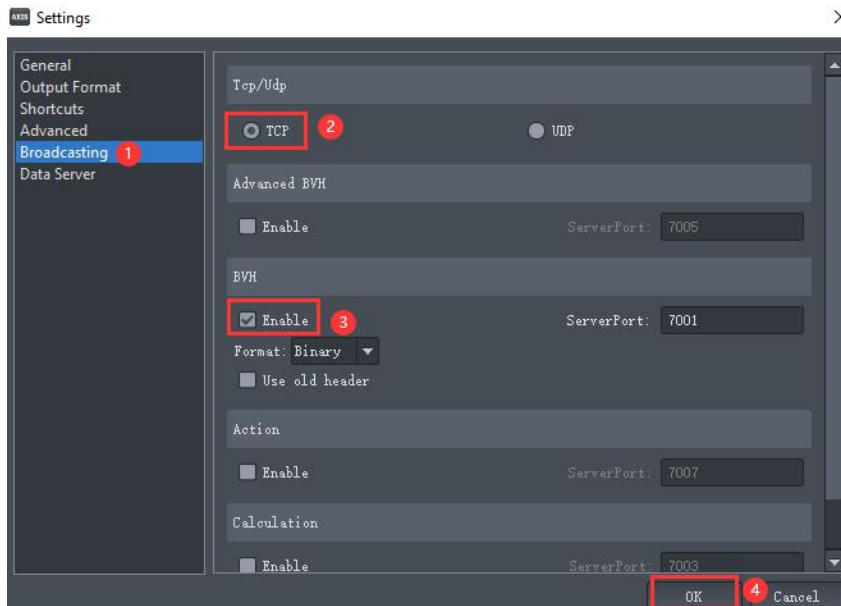
4. Data streaming setup in Axis

4.1 Axis Neuron/Axis Neuron Pro Setup

1. Run Axis Neuron/Axis Neuron Pro



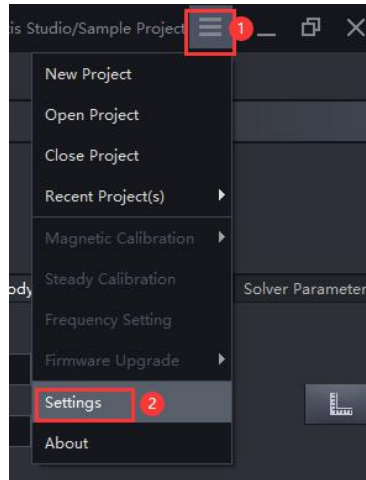
2. Click File >Settings >Broadcasting, select TCP/UDP protocol and enable BVH data streaming by checking the Enable checkbox. In this case we use TCP protocol.



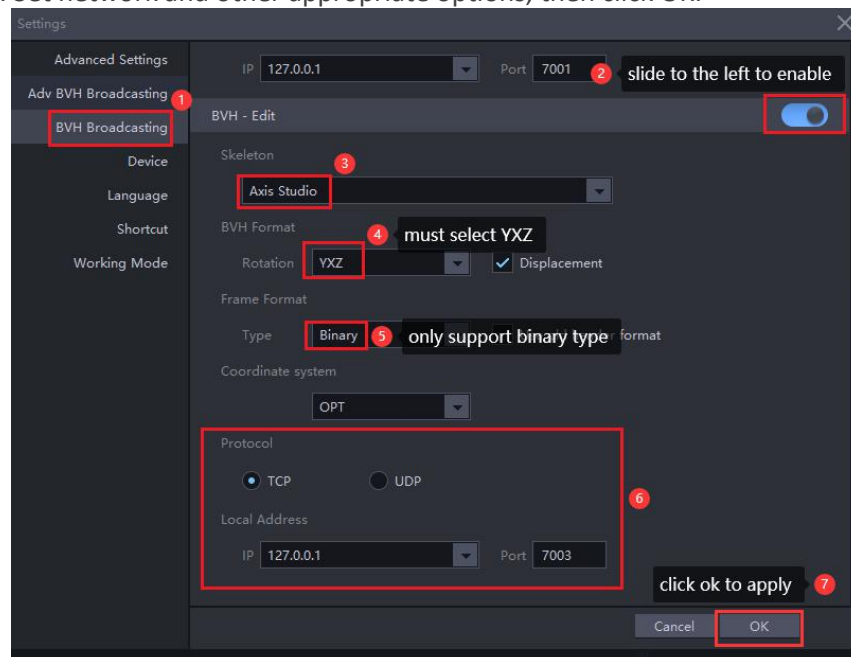
3. Playback a recorded motion capture data or connect a device

4.2 Axis Studio Setup

1. Run Axis Studio and open a project
2. Click Menu >Settings



3. Navigate to BVH Broadcasting, enable BVH-Edit (or BVH-Capture) , in this case we select BVH -Edit. Set network and other appropriate options, then click OK.

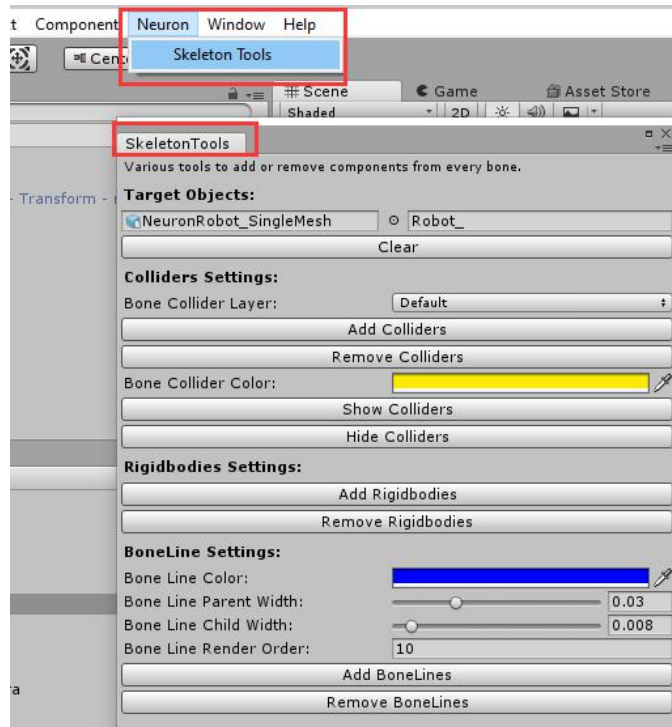


Note: Enable BVH-Capture if you stream motion capture data from Capturing viewport.

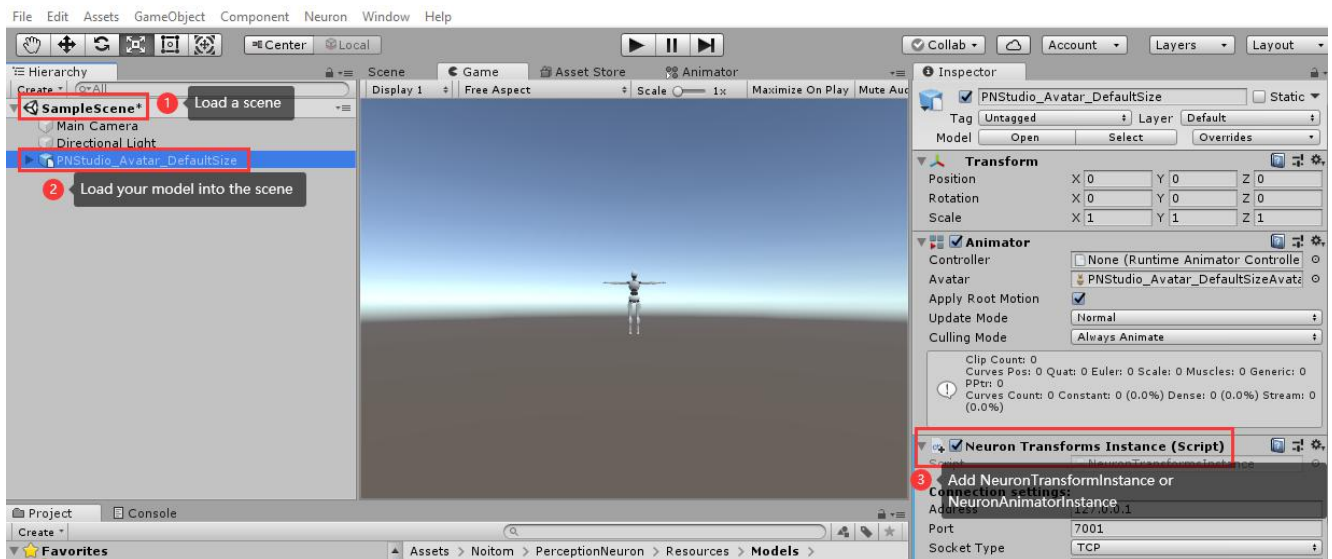
4. Playback a recorded motion capture data or connect a device

5. Neuron Skeleton Tools

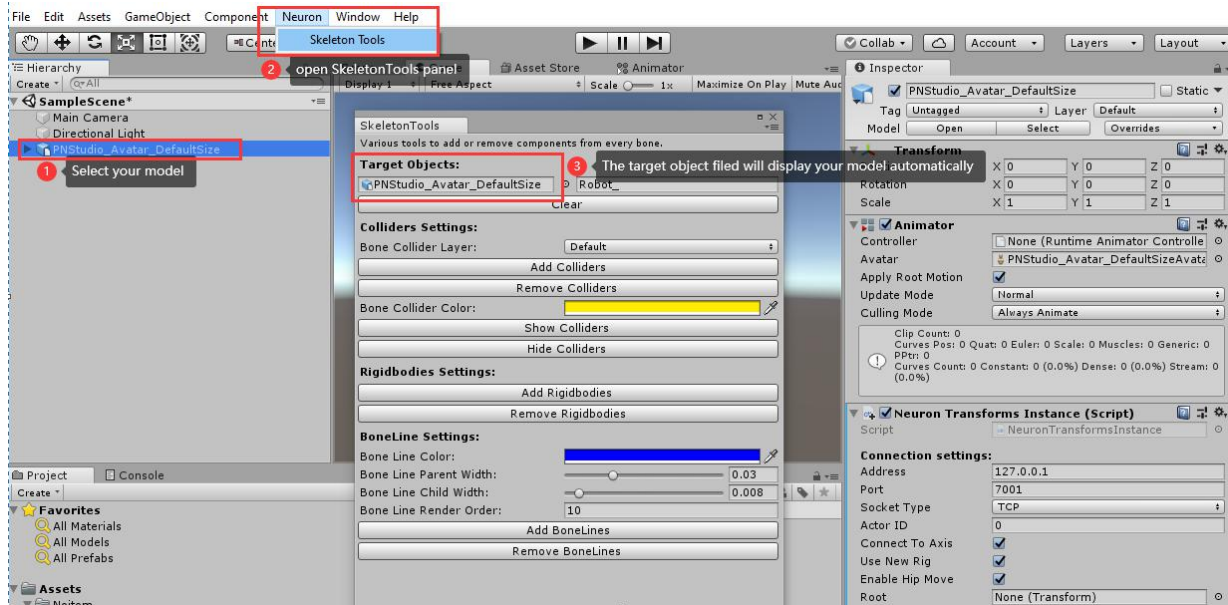
We programmed a nifty tool to help you with configuring and making changes to your avatar. Its called Skeleton Tools and can be found in the menu Neuron -> Skeleton Tools.



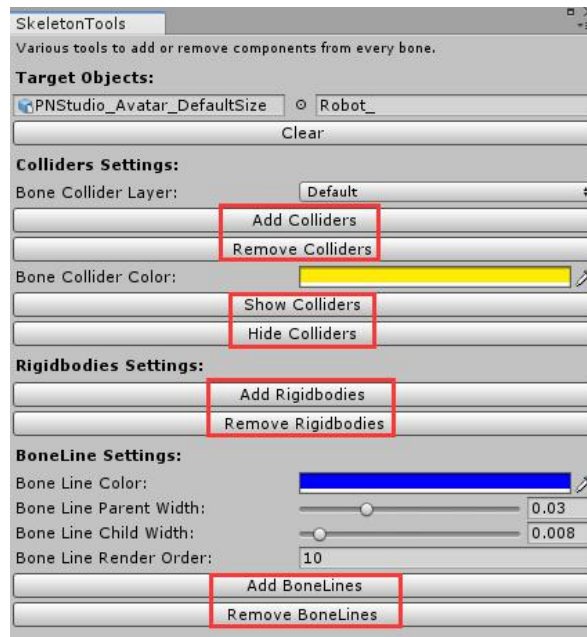
1. Select your GameObject that either has a NeuronAnimatorInstance or NeuronTransformInstance script attached to it. This will provide the tools script with the right interface to work with.



2. Keep your model selected and click Neuron > SkeletonTools to open the SkeletonTools panel.



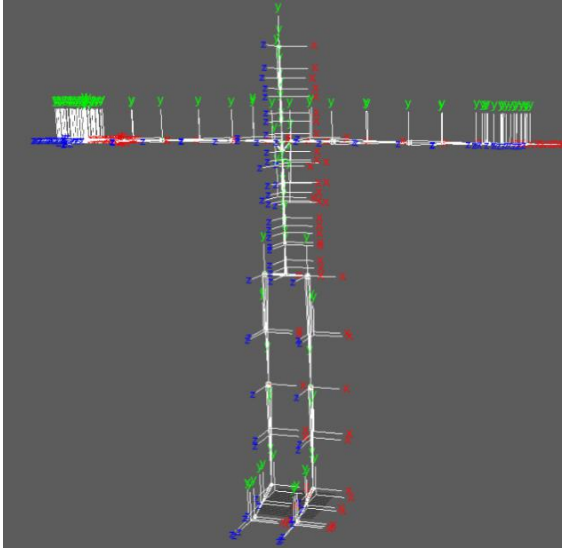
3. Now you can add/remove components for bones such as:
- 1) Adding/removing of colliders.
 - 2) Showing/hiding of colliders.
 - 3) Adding/removing of rigidbody components.
 - 4) Adding/removing of bone lines. (Bone lines are a great tool for debugging and can also be added to instantiated reference skeleton, when using physics update, during runtime).
The render order setting defines which line to render onto of each other if you have multiple bone lines.



6. How to configure a character model

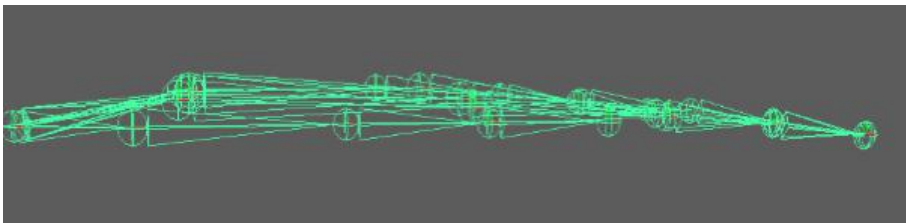
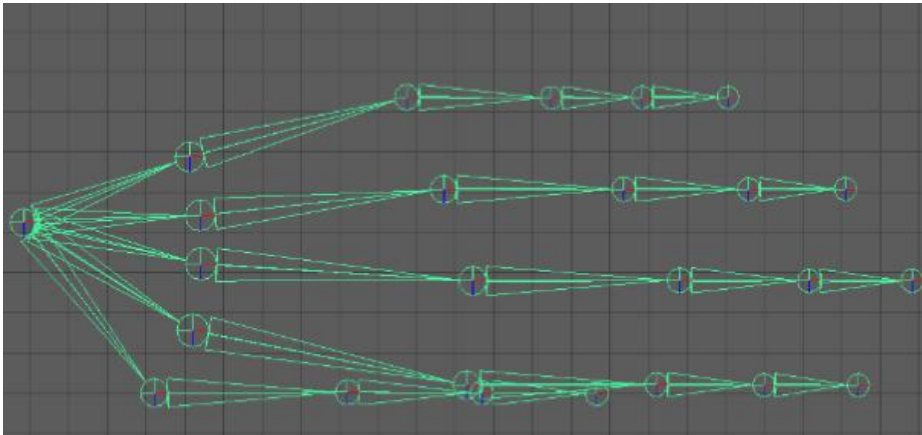
The following is a guide for how to configure a new character model to receive real-time motion data by Perception Neuron. Before we start you need to be aware of a few things:

1. Your model needs to be rigged on a humanoid skeleton
2. The bones of your rig can not have any existing rotations in them
3. Your model needs to be rigged in a T-Pose
4. The rotation value of all bones in T pose is zero
5. Each bone's local rotation axes should be the same orientation as shown in the following figure:

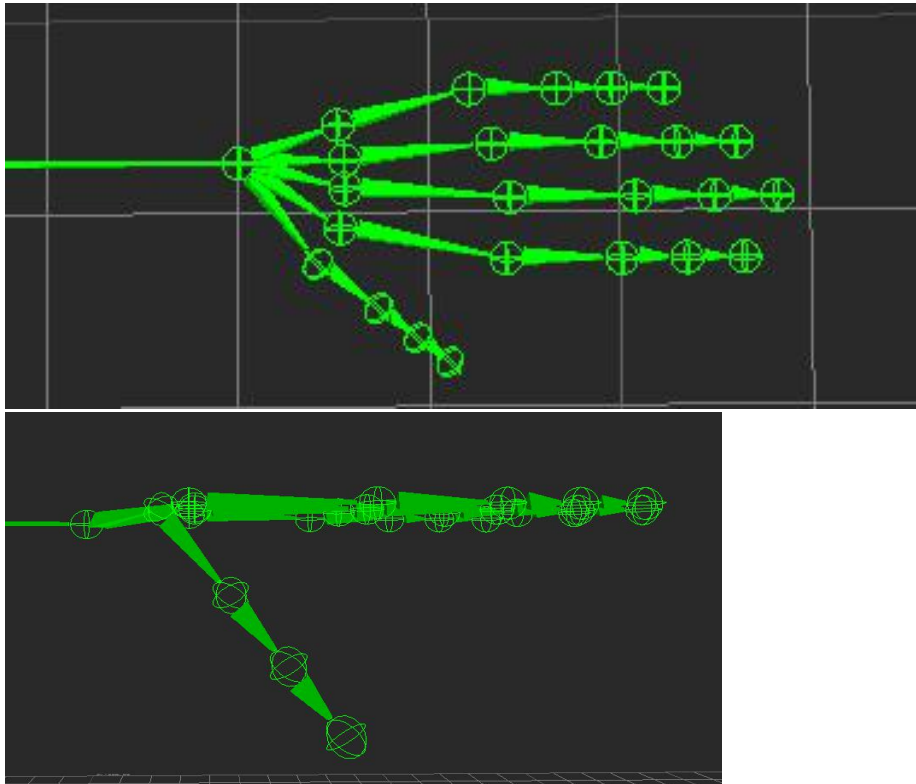


6. Fingers posture refer to the following two kinds, one is thumb inside the palm and the other one is thumb open.

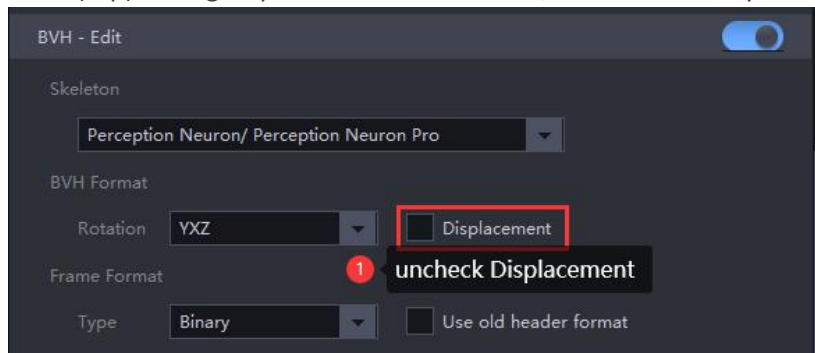
- 1) Thumb inside the palm shown as in the following figures.



- 2) Thumb open shown as in the following figures.



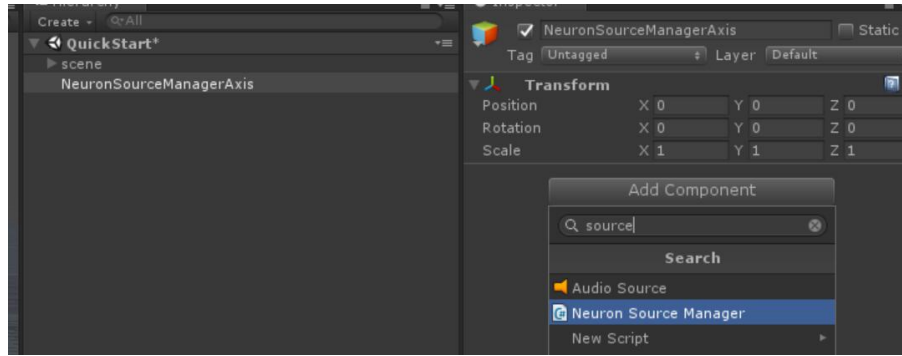
7. We support two sets of Skeleton Structure PN/PN Pro and PN Studio. Please refer to [APPENDIX B: SKELETON BONES](#)
8. Set BVH output data without displacement in Axis If you're model is not rigged with our skeleton setup or your model may be twisted. For the BVH data without displacement, except the root node (Hip) having displacement and rotation, other bones only have rotation data.



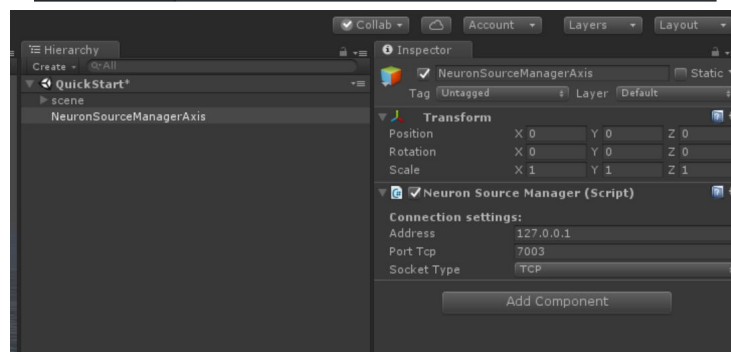
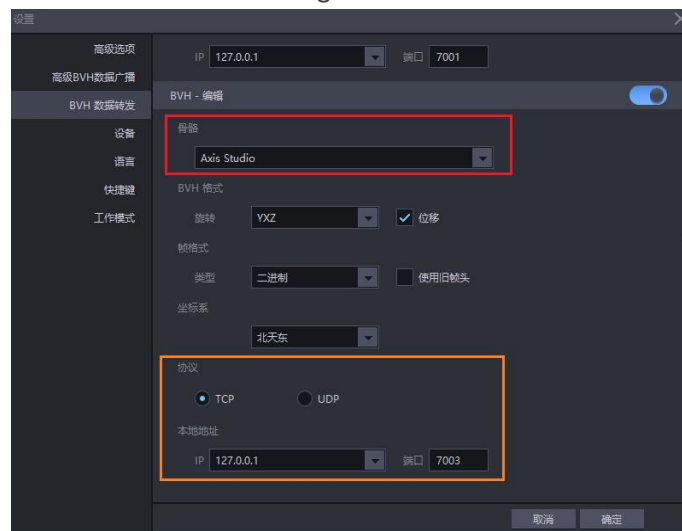
Note: If you don't consider the points mentioned above the applied motion from the integration will look wrong. Please make sure to correct those issues. Worst case suggestion is to rig your model on the included skeleton template found at: Noitom/PerceptionNeuron/Resources/Models

7. How to Apply Real-time Data from Axis

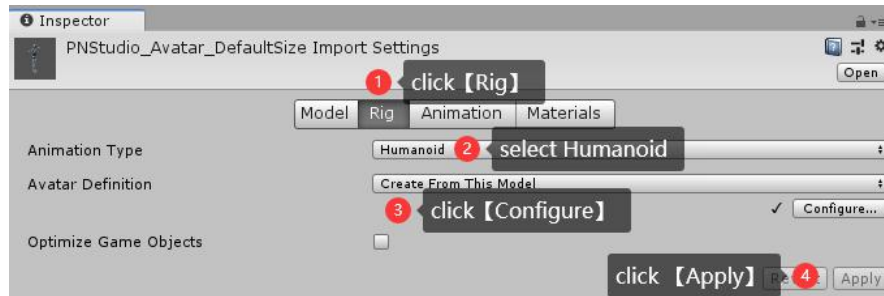
1. Import PerceptionNeuron.unitypackage
2. Add an empty GameObject. In the scene, add the neurosourcemanager component through addcomponent



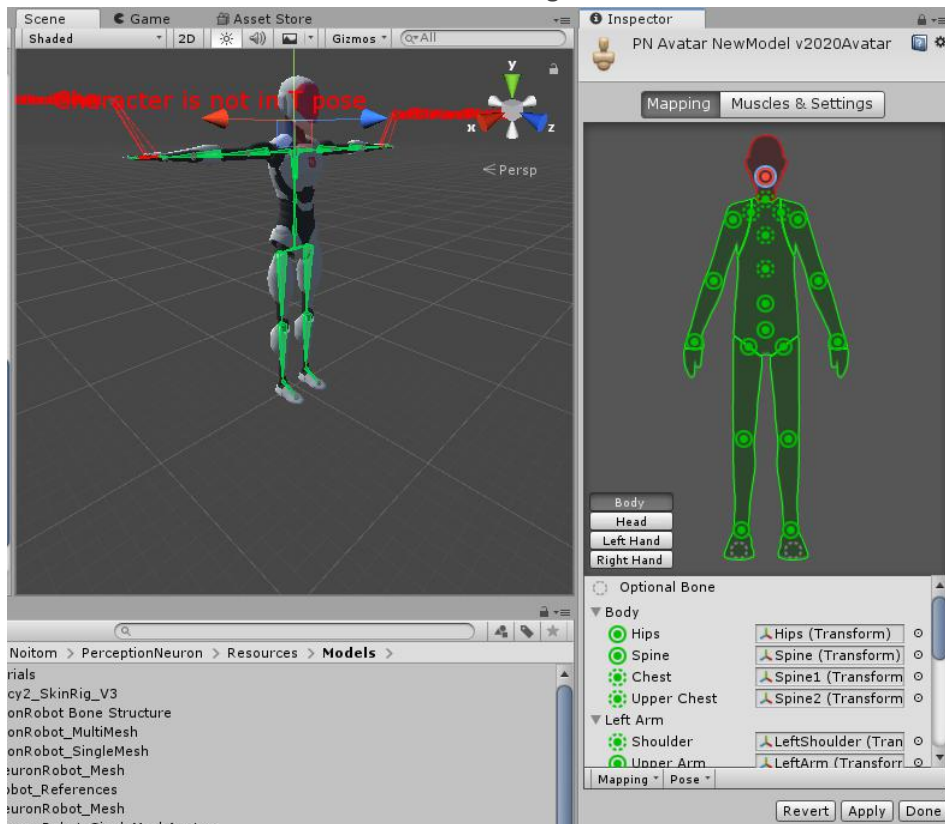
3. The connection settings are set according to the data flow parameters set in the axis software. For the data flow forwarding settings of the axis software, refer to how to set the data flow forwarding in the axis software. The data source of this example will use the axis Studio software to playback the data. It is recommended that the axis Studio software circularly broadcast and play the dynamic capture files to ensure that the data flow is not interrupted. The relevant settings of the data flow are shown in the figure:



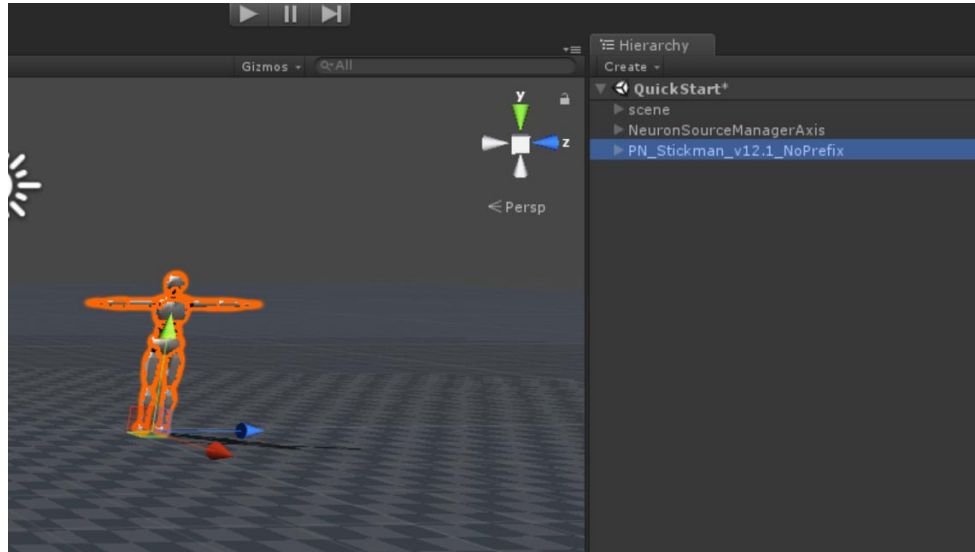
4. You need to use a humanoid human skeleton model. For the model, please refer to how to configure / select the dynamic capture model and import the model into unity. The axis Studio software sample model will be used in this example
5. In the Project tab in Unity click on your model and then click on the Rig tab in the inspector windows. Set your Animation Type to "Humanoid", then click on 【Apply】 .



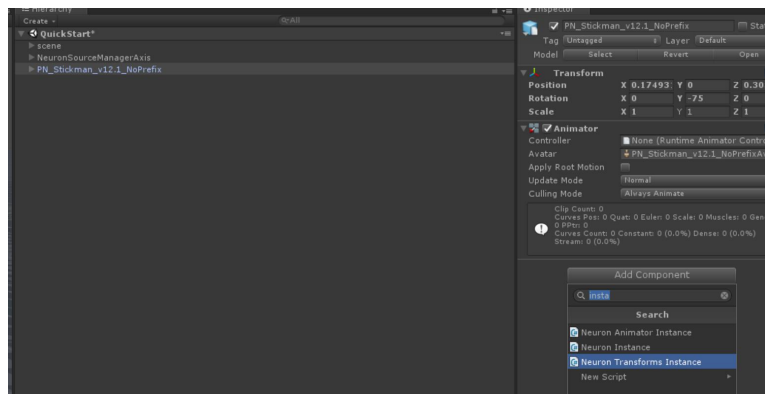
6. Next click on **【Configure】** to check the bone mapping. Usually Unity will find the right bone references for almost all the humanoid skeleton so its good check them.



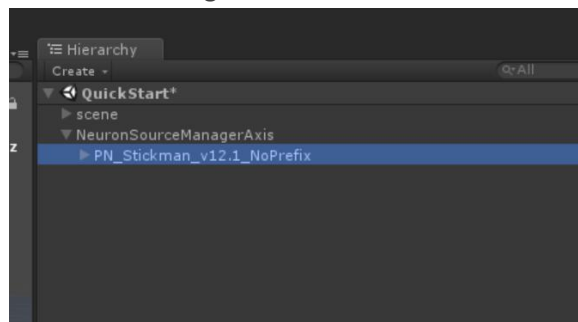
7. Click on Mapping and then click on Load to load our template file that fixes the mapping for you. IN the new window navigate to Noitom/Resources/BoneMapping and load the file PNStickMan.ht. Please refer to [APPENDIX A: SKELETON MAPPING](#)
Make sure you have all the right bones mapped to the correct body segments inside each of the tabs for Body and Click on **【Apply】** and then on **【Done】**
8. Load the scene you want to use your model in. Click and drag your model into the Hierarchy tab to load it into the scene.



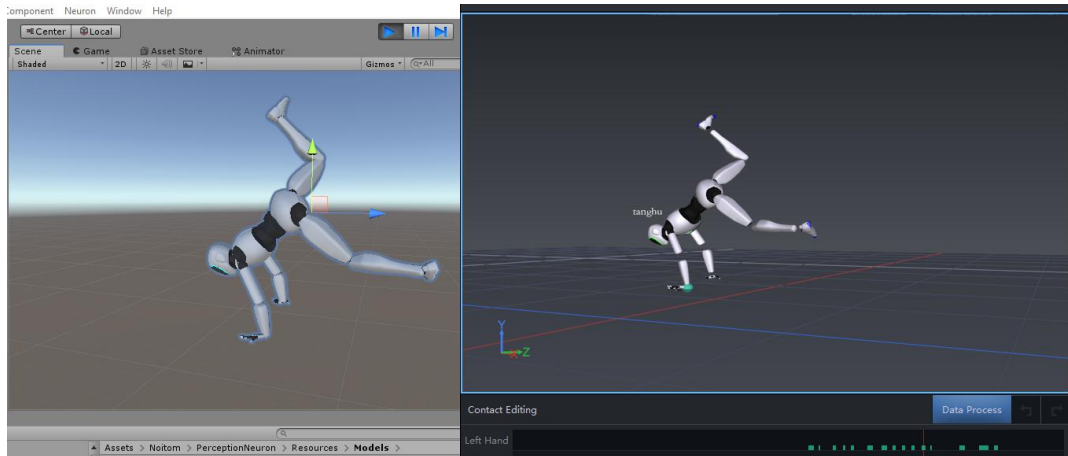
9. Select your model in the Hierarchy tab and click on **【Add Component】** in the inspector window. IN the search field type “Neuron” and select the script “NeuronTransformInstance.cs”.



10. Attach avatar to the NeuroSouceManager as a child node.



11. Click on **【Play】** in Unity to see the motion being applied to your new model.



8. HMD INTEGRATION

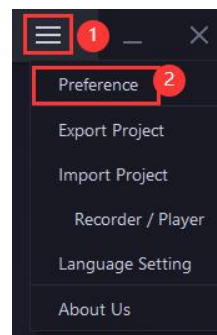
Getting any HMD synced up with the Axis integration can be a bit tricky. Here is how we got it done and what we found out to be working the best. There are three core rules worth mentioning:


1. The rotation tracker of the HMD should always have priority. Don't overwrite its rotation values with something else and don't use the head rotation values from the Axis.
2. Don't use the positional tracking of the HMD.
3. Never parent the HMD cameras or GameObject to the skeleton setup.

9. Setup in AHM/Alice

9.1 AHM/Alice Data Streaming Setup

1. Run AHM/Alice



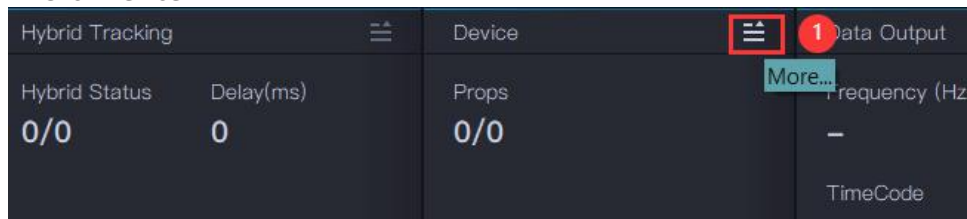
2. Click  > Preference > Data Output, enable Mocap, select TCP/UDP protocol and set the IP address and the port. In this case we use TCP protocol.



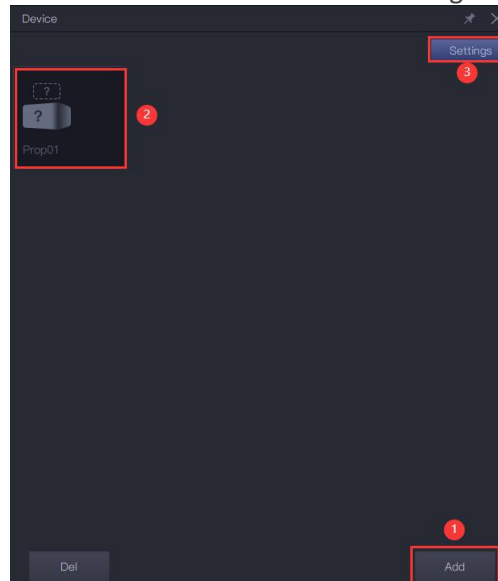
3. Click **【OK】** to apply.

9.2 AHM/Alice Device Setup

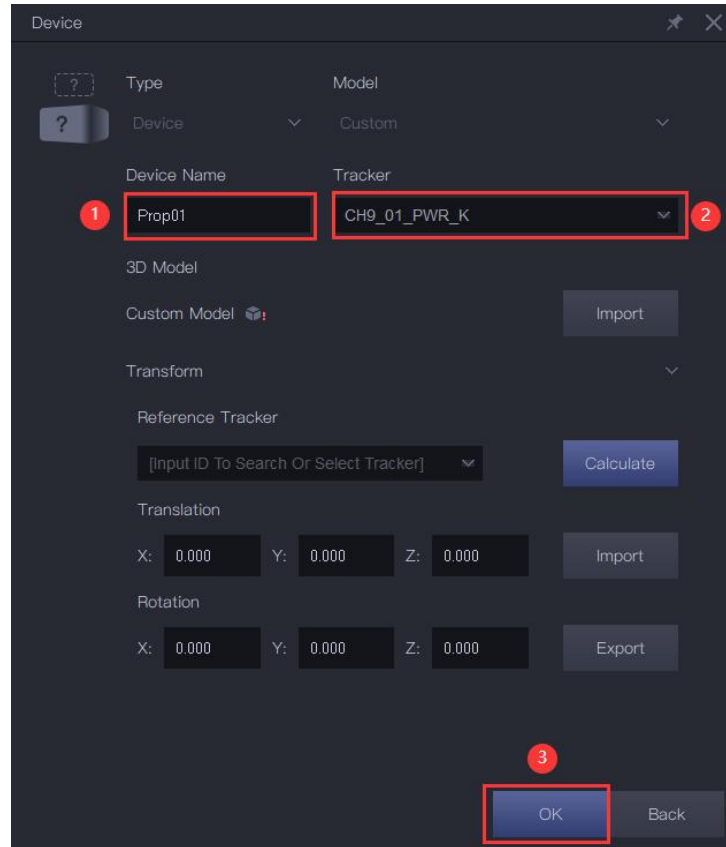
1. Run AHM/Alice
2. Click Device >



3. Click 'Add' to add new device. Select this device and click 'Settings'.



4. Input the name of Device (the name will be used in Unity to identify the specific device), and select the AHM/Alice tracker (the hybrid tracker) to tracking this device. Then click 'OK' to apply.



10. How to Apply AHM/Alice Device Data

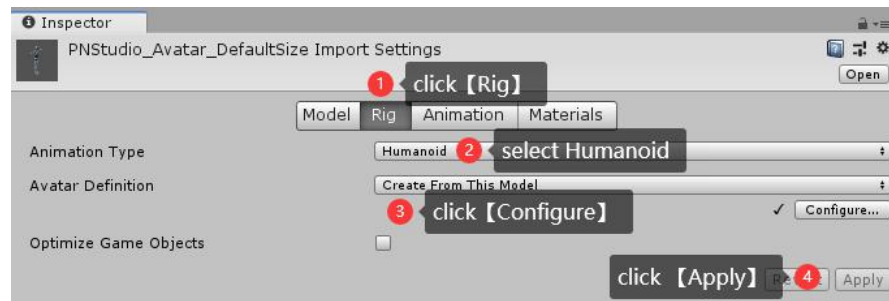
1. Create a GameObject and **【Add Component】** , select "Neuron Tracker".
 - a) Address: the IP address set in 9.1
 - b) Port: the port set in 9.1
 - c) Socket Type: TCP/UDP set in 9.1
2. Set the parameters as the settings in AHM/Alice
 - a) Device Name: the name of the device set in 9.2

11. How to Apply FBX file Data from Axis

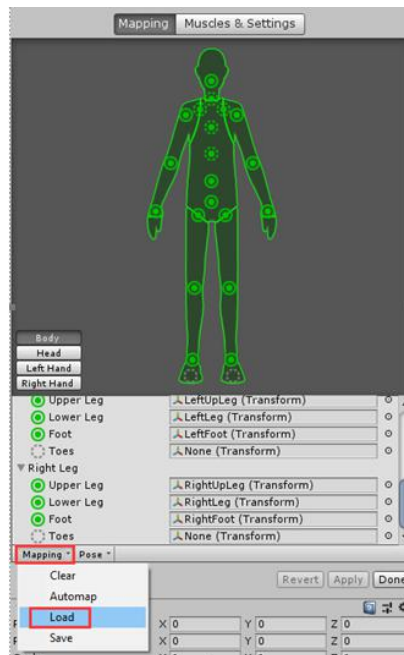
11.1 Humanoid Animation type.

3. Import your model and in the Project tab in Unity click on your model and then click on the Rig

tab in the inspector windows. Set your Animation Type to “Humanoid”, then click on 【Apply】 .



- Next click on 【Configure】 to check the bone mapping. Usually Unity will find the right bone references for almost all the humanoid skeleton so its good check them. You can load pre-make .ht mapping file or manually do the mapping.

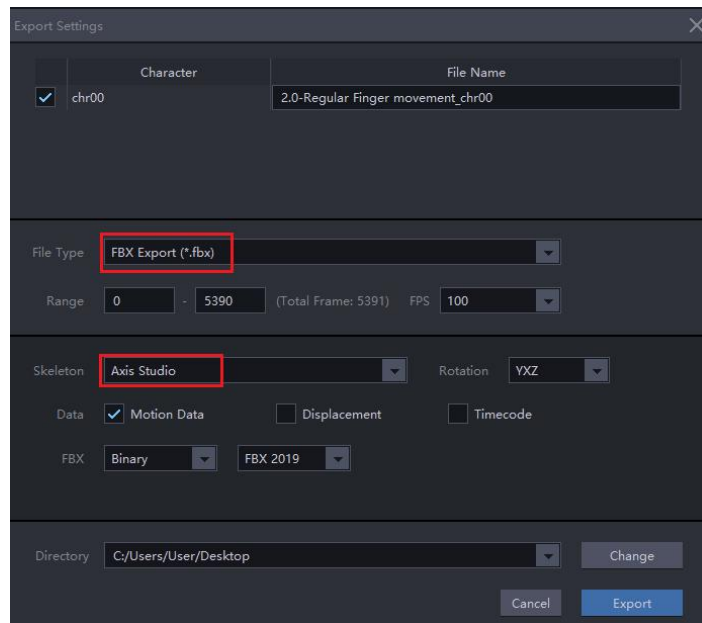


Projects > Neuron_Mocap_Live_Unity > Assets > Noitom > Resources > Models > BoneMapping			
名称	修改日期	类型	
NeuronRobot.ht	2022/3/11 10:26	HT 文件	
NeuronRobot.ht.meta	2022/3/11 10:26	META 文件	
PNStickMan.ht	2022/3/11 10:24	HT 文件	
PNStickMan.ht.meta	2022/3/11 10:24	META 文件	

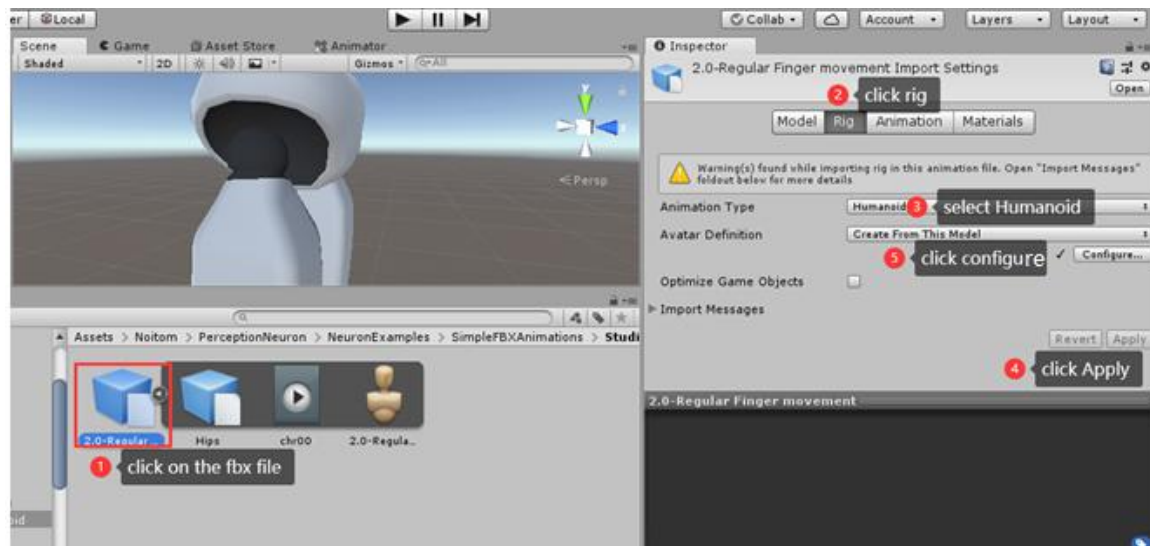
- Click 【Apply】
- If the bone Mapping is correct, but the character is not in the correct pose, you will see the message “Character not in T-Pose”. You can try to fix that with **Enforce T-Pose** or rotate the

remaining bones into T-pose.

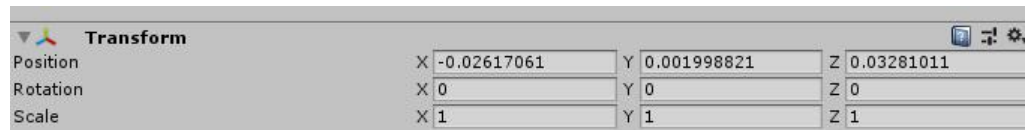
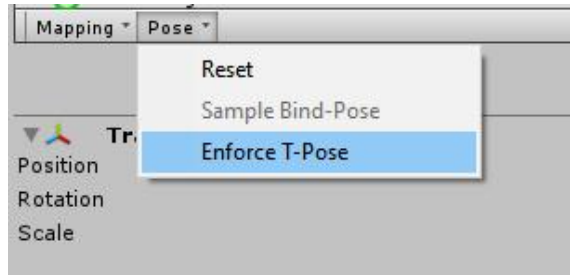
7. Export an fbx file from Axis Neuron/Axis Studio. In this case the export setting in Axis Studio is shown as the figure.



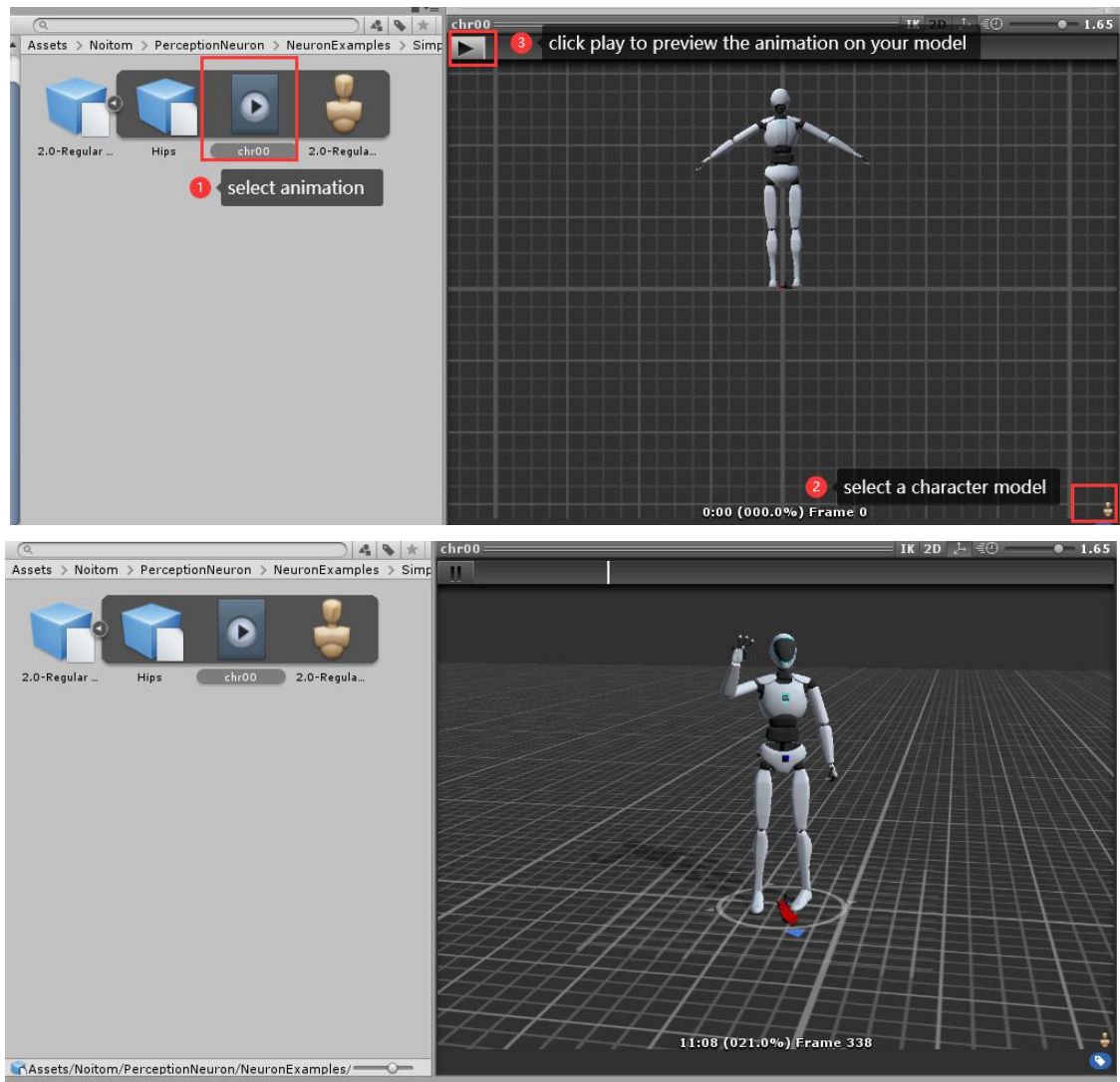
8. Import this .fbx format animation file into Unity and click on it. On the Rig tab in the inspector windows. Set your Animation Type to “Humanoid”, then click on **【Apply】**.



9. Make sure the bone Mapping and the initial T-pose are exactly same as the settings of the imported model we set in Step4, You may need to do Enforce T-Pose and rotate some bone's value in the Transform panel.



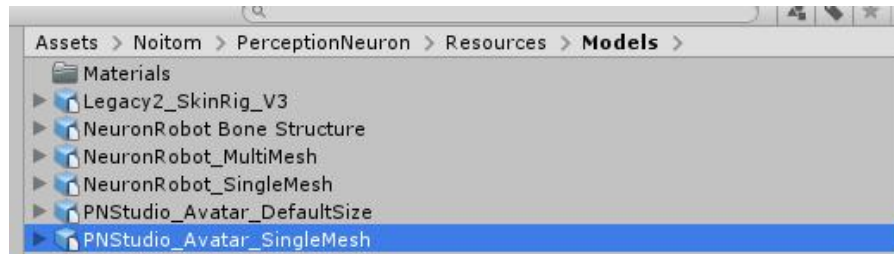
10. Click on **【Apply】** and then on **【Done】**
11. You can preview the animation on your model



11.2 Generic Animation Type

You can also apply Generic Animation type if the bone names and hierarchical structure of the model are exactly same as our animation .fbx file.

Refer to Noitom/PerceptionNeuron/Resources/Models/PNStudio_Avatar_SingleMesh.fbx



12. APPENDIX A: SKELETON MAPPING

12.1 PN Studio skeleton structure mapping

Head			
Neck	Neck (Transform)	Left Fingers	Left-handThumb1 (Transform)
Head	Head (Transform)	Thumb Proximal	Left-handThumb2 (Transform)
Left Eye	None (Transform)	Thumb Intermediate	Left-handThumb3 (Transform)
Right Eye	None (Transform)	Thumb Distal	Left-handIndex1 (Transform)
Jaw	None (Transform)	Index Proximal	Left-handIndex2 (Transform)
Body		Index Intermediate	Left-handIndex3 (Transform)
Hips	Hips (Transform)	Index Distal	Left-handMiddle1 (Transform)
Spine	Spine (Transform)	Middle Proximal	Left-handMiddle2 (Transform)
Chest	Spine1 (Transform)	Middle Intermediate	Left-handMiddle3 (Transform)
Upper Chest	Spine2 (Transform)	Middle Distal	Left-handRing1 (Transform)
Left Arm		Ring Proximal	Left-handRing2 (Transform)
Shoulder	LeftShoulder (Transform)	Ring Intermediate	Left-handRing3 (Transform)
Upper Arm	LeftArm (Transform)	Ring Distal	Left-handPinky1 (Transform)
Lower Arm	LeftForeArm (Transform)	Little Proximal	Left-handPinky2 (Transform)
Hand	LeftHand (Transform)	Little Intermediate	Left-handPinky3 (Transform)
Right Arm		Little Distal	
Shoulder	RightShoulder (Transform)	Right Fingers	Right-handThumb1 (Transform)
Upper Arm	RightArm (Transform)	Thumb Proximal	Right-handThumb2 (Transform)
Lower Arm	RightForeArm (Transform)	Thumb Intermediate	Right-handThumb3 (Transform)
Hand	RightHand (Transform)	Thumb Distal	Right-handIndex1 (Transform)
Lower Arm	RightForeArm (Transform)	Index Proximal	Right-handIndex2 (Transform)
Hand	RightHand (Transform)	Index Intermediate	Right-handIndex3 (Transform)
Left Leg		Index Distal	Right-handMiddle1 (Transform)
Upper Leg	LeftUpLeg (Transform)	Middle Proximal	Right-handMiddle2 (Transform)
Lower Leg	LeftLeg (Transform)	Middle Intermediate	Right-handMiddle3 (Transform)
Foot	LeftFoot (Transform)	Middle Distal	Right-handRing1 (Transform)
Toes	LeftFoot_End (Transform)	Ring Proximal	Right-handRing2 (Transform)
Right Leg		Ring Intermediate	Right-handRing3 (Transform)
Upper Leg	RightUpLeg (Transform)	Ring Distal	Right-handPinky1 (Transform)
Lower Leg	RightLeg (Transform)	Little Proximal	Right-handPinky2 (Transform)
Foot	RightFoot (Transform)	Little Intermediate	Right-handPinky3 (Transform)
Toes	RightFoot_End (Transform)	Little Distal	

12.2 PN/PN Pro skeleton structure mapping

▼ Body		
● Hips	Robot_Hips	○
● Spine	Robot_Spine	○
● Chest	Robot_Spine3	○
▼ Left Arm		
● Shoulder	Robot_LeftShoulder	○
● Upper Arm	Robot_LeftArm	○
● Lower Arm	Robot_LeftForeArm	○
● Hand	Robot_LeftHand	○
▼ Right Arm		
● Shoulder	Robot_RightShoulder	○
● Upper Arm	Robot_RightArm	○
● Lower Arm	Robot_RightForeArm	○
● Hand	Robot_RightHand	○
▼ Left Leg		
● Upper Leg	Robot_LeftUpLeg	○
● Lower Leg	Robot_LeftLeg	○
● Foot	Robot_LeftFoot	○
● Toes	Robot_LeftToeBase	○
▼ Right Leg		
● Upper Leg	Robot_RightUpLeg	○
● Lower Leg	Robot_RightLeg	○
● Foot	Robot_RightFoot	○
● Toes	Robot_RightToeBase	○
▼ Head		
● Neck	Robot_Neck	○
● Head	Robot_Head	○
○ Left Eye	None (Transform)	○
○ Right Eye	None (Transform)	○
○ Jaw	None (Transform)	○

▼ Left Fingers		
● Thumb Proximal	Robot_LeftHandThumb1	○
● Thumb Intermedi	Robot_LeftHandThumb2	○
● Thumb Distal	Robot_LeftHandThumb3	○
● Index Proximal	Robot_LeftHandIndex1	○
● Index Intermedi	Robot_LeftHandIndex2	○
● Index Distal	Robot_LeftHandIndex3	○
● Middle Proximal	Robot_LeftHandMiddle1	○
● Middle Intermedi	Robot_LeftHandMiddle2	○
● Middle Distal	Robot_LeftHandMiddle3	○
● Ring Proximal	Robot_LeftHandRing1	○
● Ring Intermediat	Robot_LeftHandRing2	○
● Ring Distal	Robot_LeftHandRing3	○
● Little Proximal	Robot_LeftHandPinky1	○
● Little Intermediat	Robot_LeftHandPinky2	○
● Little Distal	Robot_LeftHandPinky3	○
▼ Right Fingers		
● Thumb Proximal	Robot_RightHandThumb1	○
● Thumb Intermedi	Robot_RightHandThumb2	○
● Thumb Distal	Robot_RightHandThumb3	○
● Index Proximal	Robot_RightHandIndex1	○
● Index Intermedi	Robot_RightHandIndex2	○
● Index Distal	Robot_RightHandIndex3	○
● Middle Proximal	Robot_RightHandMiddle1	○
● Middle Intermedi	Robot_RightHandMiddle2	○
● Middle Distal	Robot_RightHandMiddle3	○
● Ring Proximal	Robot_RightHandRing1	○
● Ring Intermediat	Robot_RightHandRing2	○
● Ring Distal	Robot_RightHandRing3	○
● Little Proximal	Robot_RightHandPinky1	○
● Little Intermediat	Robot_RightHandPinky2	○
● Little Distal	Robot_RightHandPinky3	○

13. APPENDIX B: SKELETON BONES

Axis Neuron:

- | | |
|------------------------------------|-----------------------------------|
| 1. Hips [↵] | 31. RightHandRing2 [↵] |
| 2. RightUpLeg [↵] | 32. RightHandRing3 [↵] |
| 3. RightLeg [↵] | 33. RightInHandPinky [↵] |
| 4. RightFoot [↵] | 34. RightHandPinky1 [↵] |
| 5. LeftUpLeg [↵] | 35. RightHandPinky2 [↵] |
| 6. LeftLeg [↵] | 36. RightHandPinky3 [↵] |
| 7. LeftFoot [↵] | 37. LeftShoulder [↵] |
| 8. Spine [↵] | 38. LeftArm [↵] |
| 9. Spine1 [↵] | 39. LeftForeArm [↵] |
| 10. Spine2 [↵] | 40. LeftHand [↵] |
| 11. Spine3 [↵] | 41. LeftHandThumb1 [↵] |
| 12. Neck [↵] | 42. LeftHandThumb2 [↵] |
| 13. Head [↵] | 43. LeftHandThumb3 [↵] |
| 14. RightShoulder [↵] | 44. LeftInHandIndex [↵] |
| 15. RightArm [↵] | 45. LeftHandIndex1 [↵] |
| 16. RightForeArm [↵] | 46. LeftHandIndex2 [↵] |
| 17. RightHand [↵] | 47. LeftHandIndex3 [↵] |
| 18. RightHandThumb1 [↵] | 48. LeftInHandMiddle [↵] |
| 19. RightHandThumb2 [↵] | 49. LeftHandMiddle1 [↵] |
| 20. RightHandThumb3 [↵] | 50. LeftHandMiddle2 [↵] |
| 21. RightInHandIndex [↵] | 51. LeftHandMiddle3 [↵] |
| 22. RightHandIndex1 [↵] | 52. LeftInHandRing [↵] |
| 23. RightHandIndex2 [↵] | 53. LeftHandRing1 [↵] |
| 24. RightHandIndex3 [↵] | 54. LeftHandRing2 [↵] |
| 25. RightInHandMiddle [↵] | 55. LeftHandRing3 [↵] |
| 26. RightHandMiddle1 [↵] | 56. LeftInHandPinky [↵] |
| 27. RightHandMiddle2 [↵] | 57. LeftHandPinky1 [↵] |
| 28. RightHandMiddle3 [↵] | 58. LeftHandPinky2 [↵] |
| 29. RightInHandRing [↵] | 59. LeftHandPinky3 [↵] |
| 30. RightHandRing1 [↵] | |

Axis Studio:

- | | |
|------------------------------------|-----------------------------------|
| 1. Hips [↵] | 31. RightHandRing2 [↵] |
| 2. RightUpLeg [↵] | 32. RightHandRing3 [↵] |
| 3. RightLeg [↵] | 33. RightInHandPinky [↵] |
| 4. RightFoot [↵] | 34. RightHandPinky1 [↵] |
| 5. LeftUpLeg [↵] | 35. RightHandPinky2 [↵] |
| 6. LeftLeg [↵] | 36. RightHandPinky3 [↵] |
| 7. LeftFoot [↵] | 37. LeftShoulder [↵] |
| 8. Spine [↵] | 38. LeftArm [↵] |
| 9. Spine1 [↵] | 39. LeftForeArm [↵] |
| 10. Spine2 [↵] | 40. LeftHand [↵] |
| 11. Neck [↵] | 41. LeftHandThumb1 [↵] |
| 12. Neck1 [↵] | 42. LeftHandThumb2 [↵] |
| 13. Head [↵] | 43. LeftHandThumb3 [↵] |
| 14. RightShoulder [↵] | 44. LeftInHandIndex [↵] |
| 15. RightArm [↵] | 45. LeftHandIndex1 [↵] |
| 16. RightForeArm [↵] | 46. LeftHandIndex2 [↵] |
| 17. RightHand [↵] | 47. LeftHandIndex3 [↵] |
| 18. RightHandThumb1 [↵] | 48. LeftInHandMiddle [↵] |
| 19. RightHandThumb2 [↵] | 49. LeftHandMiddle1 [↵] |
| 20. RightHandThumb3 [↵] | 50. LeftHandMiddle2 [↵] |
| 21. RightInHandIndex [↵] | 51. LeftHandMiddle3 [↵] |
| 22. RightHandIndex1 [↵] | 52. LeftInHandRing [↵] |
| 23. RightHandIndex2 [↵] | 53. LeftHandRing1 [↵] |
| 24. RightHandIndex3 [↵] | 54. LeftHandRing2 [↵] |
| 25. RightInHandMiddle [↵] | 55. LeftHandRing3 [↵] |
| 26. RightHandMiddle1 [↵] | 56. LeftInHandPinky [↵] |
| 27. RightHandMiddle2 [↵] | 57. LeftHandPinky1 [↵] |
| 28. RightHandMiddle3 [↵] | 58. LeftHandPinky2 [↵] |
| 29. RightInHandRing [↵] | 59. LeftHandPinky3 [↵] |
| 30. RightHandRing1 [↵] | |

This is the complete structure of our skeleton model. If we use the humanoid skeleton in Unity, we skip some of them resulting in 51 bones remaining. However, we need to include the data of the bones we skip in the following bones for the correct values. We skip all the InHand bones and two of the spine bones.

14. APPENDIX C: BINARY DATA SEQUENCE

On the next two pages you'll find a complete graph of the whole sequence of the binary data received from Axis Neuron. It is a one-dimensional float array with different ordering and length depending on whether you're using displacement data or not.

Axis Neuron												
Bone	NO DISPLACEMENT						WITH DISPLACEMENT					
	Position			Rotation			Position			Rotation		
	X	Y	Z	Y	X	Z	X	Y	Z	Y	X	Z
Hips	0	1	2	3	4	5	0	1	2	3	4	5
RightUpLeg				6	7	8	6	7	8	9	10	11
RightLeg				9	10	11	12	13	14	15	16	17
RightFoot				12	13	14	18	19	20	21	22	23
LeftUpLeg				15	16	17	24	25	26	27	28	29
LeftLeg				18	19	20	30	31	32	33	34	35
LeftFoot				21	22	23	36	37	38	39	40	41
Spine				24	25	26	42	43	44	45	46	47
Spine1				27	28	29	48	49	50	51	52	53
Spine2				30	31	32	54	55	56	57	58	59
Spine3				33	34	35	60	61	62	63	64	65
Neck				36	37	38	66	67	68	69	70	71
Head				39	40	41	72	73	74	75	76	77
RightShoulder				42	43	44	78	79	80	81	82	83
RightArm				45	46	47	84	85	86	87	88	89
RightForeArm				48	49	50	90	91	92	93	94	95
RightHand				51	52	53	96	97	98	99	100	101
RightHandThumb1				54	55	56	102	103	104	105	106	107
RightHandThumb2				57	58	59	108	109	110	111	112	113
RightHandThumb3				60	61	62	114	115	116	117	118	119
RightInHandIndex				63	64	65	120	121	122	123	124	125
RightHandIndex1				66	67	68	126	127	128	129	130	131
RightHandIndex2				69	70	71	132	133	134	135	136	137
RightHandIndex3				72	73	74	138	139	140	141	142	143
RightInHandMiddle				75	76	77	144	145	146	147	148	149
RightHandMiddle1				78	79	80	150	151	152	153	154	155
RightHandMiddle2				81	82	83	156	157	158	159	160	161
RightHandMiddle3				84	85	86	162	163	164	165	166	167
RightInHandRing				87	88	89	168	169	170	171	172	173
RightHandRing1				90	91	92	174	175	176	177	178	179
RightHandRing2				93	94	95	180	181	182	183	184	185
RightHandRing3				96	97	98	186	187	188	189	190	191
RightInHandPinky				99	100	101	192	193	194	195	196	197
RightHandPinky1				102	103	104	198	199	200	201	202	203
RightHandPinky2				105	106	107	204	205	206	207	208	209
RightHandPinky3				108	109	110	210	211	212	213	214	215

AxisStudio

Bone	NO DISPLACEMENT						WITH DISPLACEMENT					
	Position			Rotation			Position			Rotation		
	X	Y	Z	X	Y	Z	X	Y	Z	X	Y	Z
Hips	0	1	2	3	4	5	0	1	2	3	4	5
RightUpLeg				6	7	8	6	7	8	9	10	11
RightLeg				9	10	11	12	13	14	15	16	17
RightFoot				12	13	14	18	19	20	21	22	23
LeftupLeg				15	16	17	24	25	26	27	28	29
LeftLeg				18	19	20	30	31	32	33	34	35
Left Foot				21	22	23	36	37	38	39	40	41
Spine				24	25	26	42	43	44	45	46	47
Spine1				27	28	29	48	49	50	51	52	53
Spine2				30	31	32	54	55	56	57	58	59
Neck				33	34	35	60	61	62	63	64	65
Neck1				36	37	38	66	67	68	69	70	71
Head				39	40	41	72	73	74	75	76	77
RightShoulder				42	43	44	78	79	80	81	82	83
RightArm				45	46	47	84	85	86	87	88	89
RightForeArm				48	49	50	90	91	92	93	94	95
RightHand				51	52	53	96	97	98	99	100	101
RightHandThumb1				54	55	56	102	103	104	105	106	107
RightHandThumb2				57	58	59	108	109	110	111	112	113
RightHandThumb3				60	61	62	114	115	116	117	118	119
RightInHandIndex				63	64	65	120	121	122	123	124	125
RightHandIndex1				66	67	68	126	127	128	129	130	131
RightHandIndex2				69	70	71	132	133	134	135	136	137
RightHandIndex3				72	73	74	138	139	140	141	142	143
RightInHandMiddle				75	76	77	144	145	146	147	148	149
RightHandMiddle1				78	79	80	150	151	152	153	154	155
RightHandMiddle2				81	82	83	156	157	158	159	160	161
RightHandMiddle3				84	85	86	162	163	164	165	166	167
RightInHandRing				87	88	89	168	169	170	171	172	173
RightHandRing1				90	91	92	174	175	176	177	178	179
RightHandRing2				93	94	95	180	181	182	183	184	185
RightHandRing3				96	97	98	186	187	188	189	190	191
RightHandPinky				99	100	101	192	193	194	195	196	197
RightHandPinky1				102	103	104	198	199	200	201	202	203
RightHandPinky2				105	106	107	204	205	206	207	208	209
RightHandPinky3				108	109	110	210	211	212	213	214	215