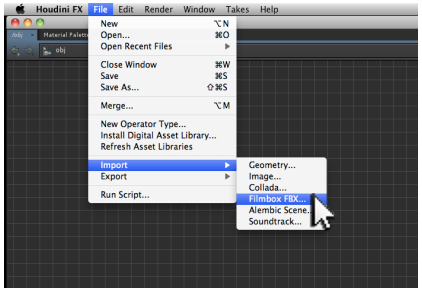
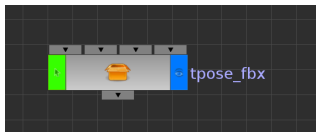
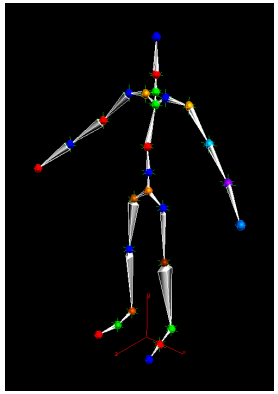
MOTION CAPTURE & HOUDINI

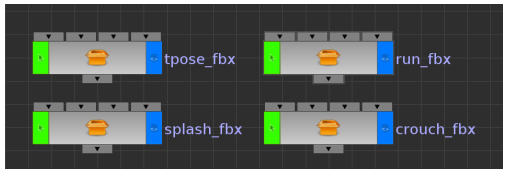
Houdini works well with Motion Capture data. MOCAP is used to provide automatic animation of real world performances. In **a new Houdini scene**, go to the main **File Menu** and select **Import > Filmbox FBX**.



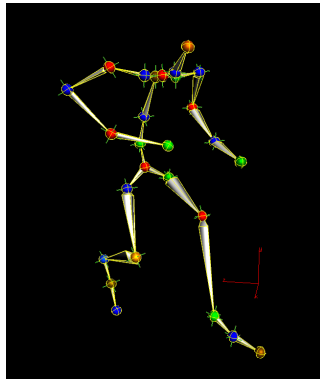
From the resulting dialog window, use the **File Chooser** button to import the **rest_pose.fbx** file and press the **Merge** button. This will import a subnetwork of a bone skeleton in a default rest pose.







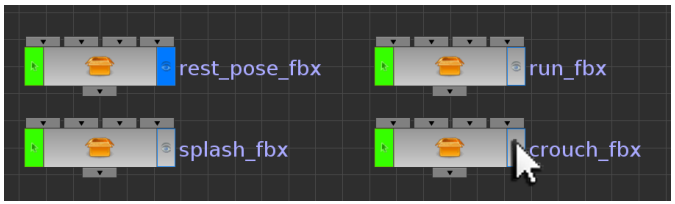
Repeat this process, reading in the following fbx files: run.fbx; splash.fbx; and crouch.fbx ensuring these files are merged into the current scene.



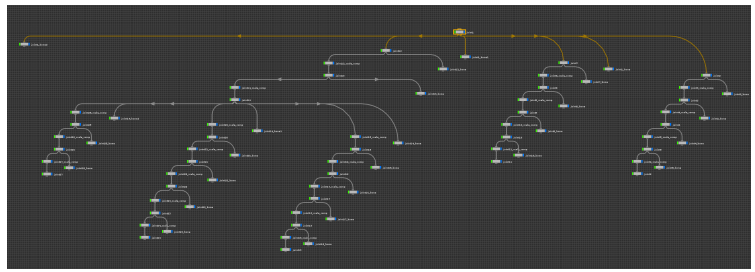
This will import duplicates of the skeleton rig, but with MOCAP animation data assigned to them. **See file fbx_mocap_stage1.hipnc**

Currently these MOCAP clips exist as separate subnetwork objects with no formal linking. If a character were to be parented or captured onto this MOCAP data it would need to be done for each MOCAP clip. A better approach however is to use CHOPS to re-assign these animated MOCAP clips directly onto the rest_pose_fbx object, so character geometry only needs to be assigned to the rest_pose_fbx object.

Alongside the fbx MOCAP clips, create a **CHOP Network** and rename it to **MOCAP_CONTROL**. **Turn off** the **Display Flags** for all the **animated MOCAP clips** aside from the rest_pose_fbx object.



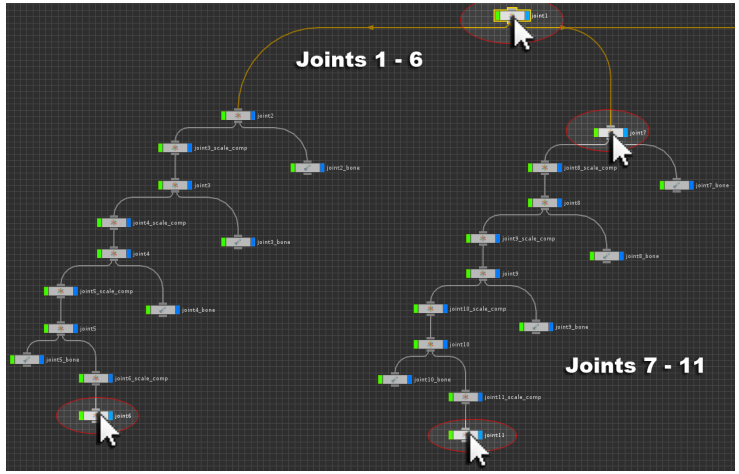
Examination of the contents of the **rest_pose_fbx** object reveals the skeleton heirachy in its entirety. Bones and controllers are presented as a tree network.



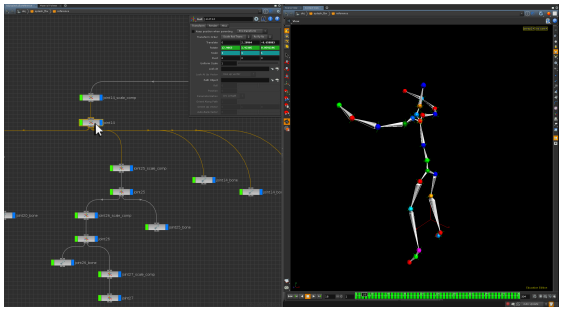
In order for MOCAP data to be efficiently re-assigned onto this network, understanding of each network branch is required. If the entire network was read into CHOPS (for example by using a Fetch CHOP) this would result in many extraneous data channels being generated, slowing down the MOCAP setup. For efficient remapping of MOCAP data, an Object Chain CHOP can be used instead to read in the initial data. This however requires knowledge of the beginning and end node for each node branch of the rest_pose_fbx skeleton hierarchy.

When this hierarchy is examined, the beginning and end nodes for each node branch can be identified as:

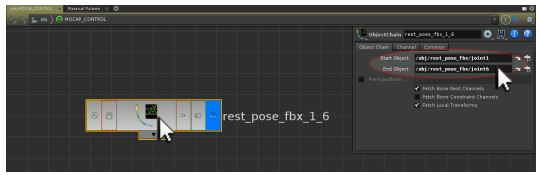
Joint 1 to Joint 6
Joint 7 to Joint 11
Joint 12 to Joint 19
Joint 20 to Joint 24
Joint 25 to Joint 27



Examination of the **animated MOCAP subnetworks** reveals the same skeleton hierarchy. The animation data itself is stored as **keyframes** on **each joint object**. This keyframe data will need to be transferred onto the equivalent joint object in the **rest_pose_fbx subnetwork**.



Head inside the **MOCAP_CONTROL CHOP Network**, and here create an **Object Chain CHOP**.



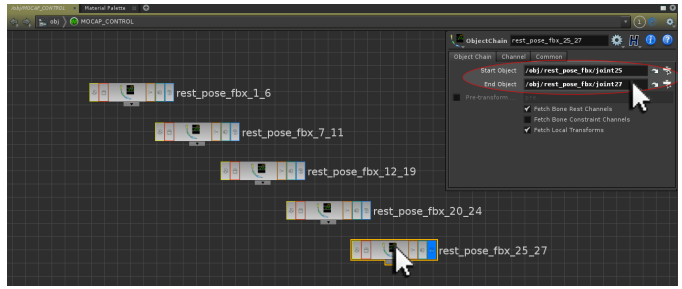
In the **parameters** for the **Object Chain CHOP** specify:

Start Object **/obj/rest_pose_fbx/joint1**

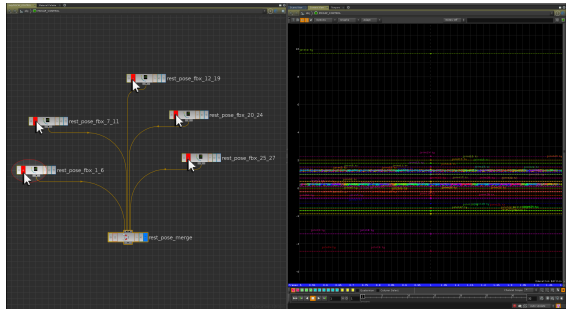
End Object **/obj/rest_pose_fbx/joint6**

■ **Pre-transform Name**

Repeat this process, creating Object Chain CHOPS for the remaining node branches of the rest_pose_fbx heirachy; modifying the parameters accordingly.

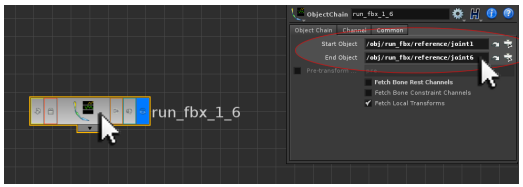


These **Object Chain CHOPs** can then be combined with a **Merge CHOP**, and its combined data verified in the **Context View**.

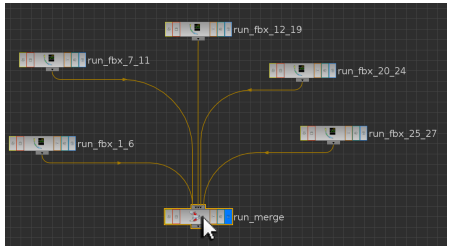


As a final step, **lock** all of the **Object Chain CHOPS** nodes. This will prevent a recursion error when animated MOCAP data is re-assigned back over the original rest_pose_fbx object. **See file fbx_mocap_stage2.hipnc**

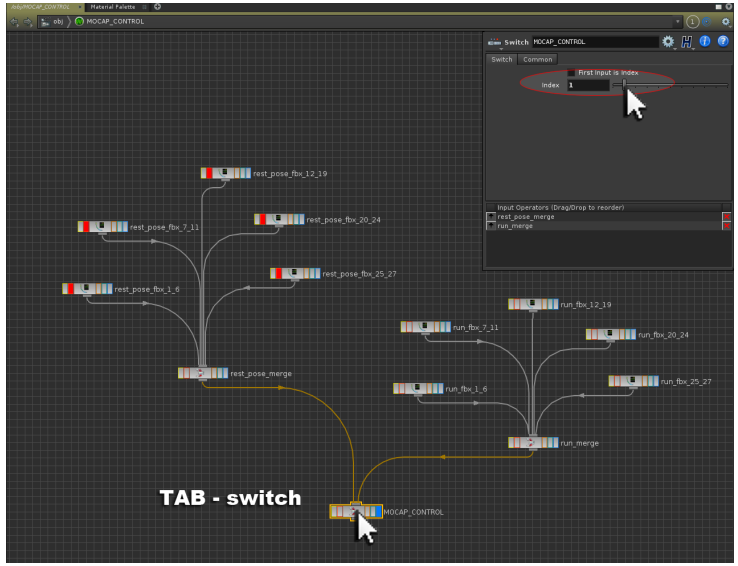
Repeat this **Object Chain CHOP creation process** for the **run_fbx object**.



As before, the run_fbx data can be merged together. It is not however necessary to lock these Object Chain Nodes.



Both Merge CHOPs can now be fed into a **Switch CHOP**, which will allow for the rest_pose_fbx data channels to be switched with the run_fbx data channels.

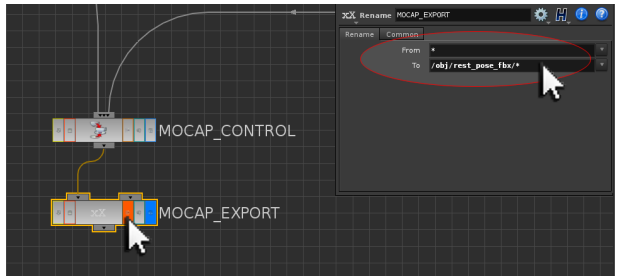


NOTE: As both the static and animated fbx skeletons, and their associated CHOP data imports are identical, mapping one set of data onto another is straightforward.

To the **output** of the **Switch CHOP** append a **Rename CHOP**. This node can be used to add the path of the **rest_pose_fbx object** as a prefix to the data channel names. In the **parameters** for the **Rename CHOP** specify:

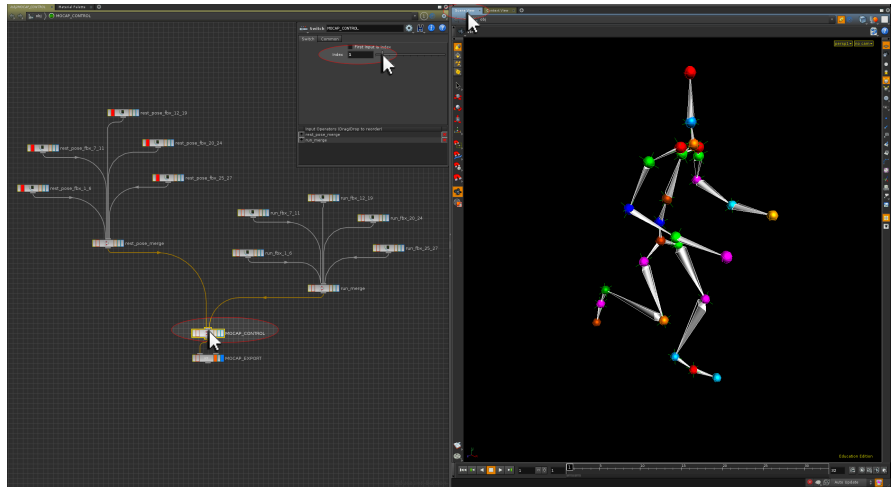
From *
To **/obj/rest_pose_fbx/***

When full paths are assigned to CHOP data channel names, this allows for automatic remapping of these modified channels directly onto a node's equivalent parameters when the Rename CHOP's Export Flag is activated.



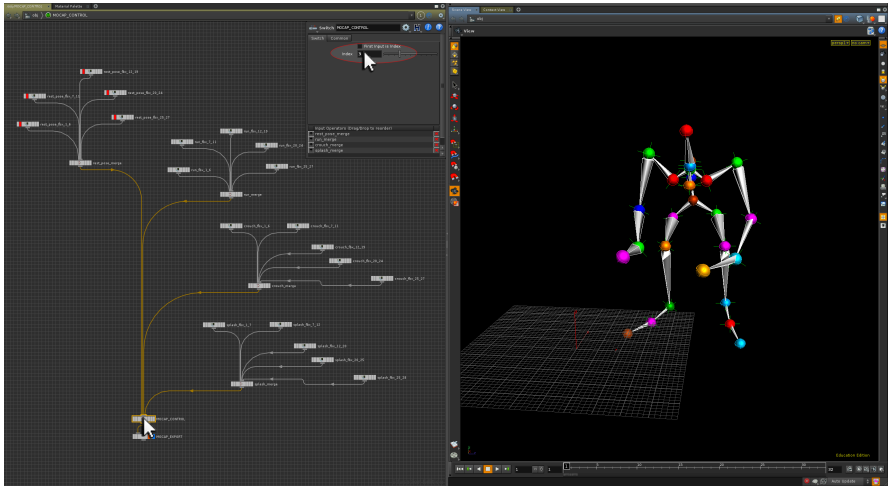
As a final step, **activate** the **orange Export Flag** on the **Rename CHOP**. This will port the animated data channels directly onto the joint objects found in the rest_pose_fbx object.

The MOCAP setup can be confirmed by setting the **Switch CHOP** to a value of **1** (the run clip), and examining the Scene View.



With the Switch CHOP set to 1, the animated MOCAP run clip is now passed onto the rest_pose_fbx object. **See file fbx_mocap_stage3.hipnc**

Any other MOCAP animation clips can also be read into CHOPS and added to the Switch CHOP's input.



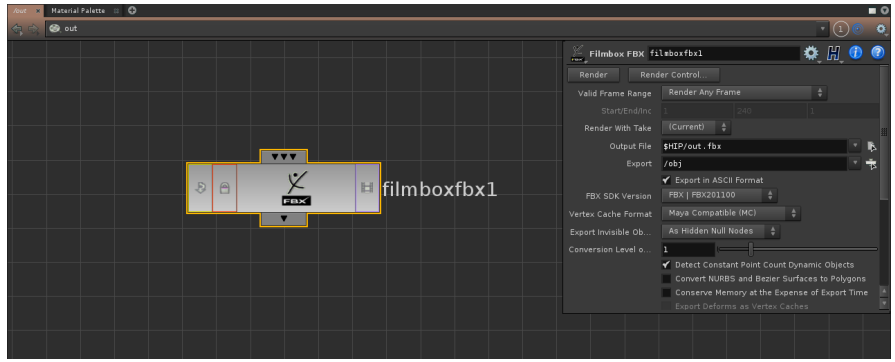
See file fbx_mocap_complete.hipnc

NOTE: A procedural script for automatically generating this MOCAP network will be developed in a later lecture looking at HScripting.

CREATING SKELETONS FOR MOCAP DATA

Houdini skeletons can be configured and exported to Motion Builder. Please speak with Zhidong Xiao and Phil Spicer about any custom skeleton design required. If necessary Zhidong can provide the default skeleton (used in this example) without requiring a new one to be generated in Houdini.

If a custom skeleton is required, it can be drawn using the **Bones Tool** of Houdini in a new scene. The skeleton network can then be collapsed into a **subnetwork** by selecting all of the nodes and pressing **SHIFT + c** with the **mouse over the Network Editor**.



This **subnetwork** can then be exported from Houdini from **Outputs Level** using a **Filmbox FBX ROP** pointing to the subnetwork.