Get an armature for your mod, so that you can reuse your mod model in scenery mods, in blender scenes, check your VG weights, or whatever you want to do.

Thanks to Silent. This method is 90% based on his VG mapping algorithm from his script (here)

Before we start, i recommend to prepare the armature in a new blender file. You can save this blender file elsewhere and append the armature to your modding file.

1. Data prepare

① 3dmigoto-dumped meshes

Import the 3dmigoto-dumped meshes of the character to blender. It should be the origin data of character, not a modded one.

Select all the meshes, **clear** all of their transformation. (Do not apply them)

Join the meshes by IB hash.

② Character asset from game unpack

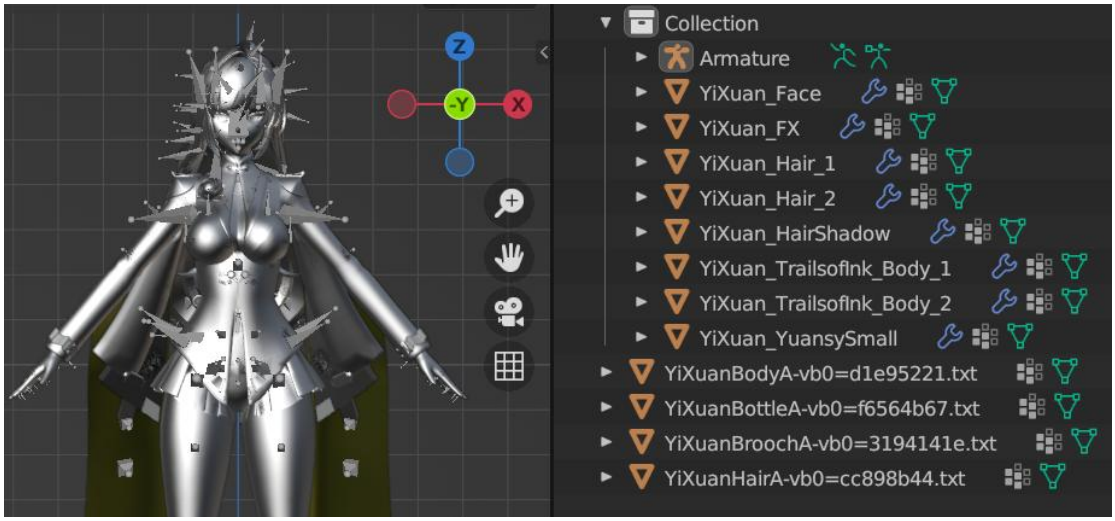
Import the asset, clear parent and keep transform for all asset meshes.

Mirror all asset meshes and the armature in x-axis.

Rotate the armature and asset meshes, until asset meshes match the dumped meshes perfectly. (normally you should only rotate in x-axis.)

Apply the transformation of all asset meshes.

Don't apply the transform of the armature.



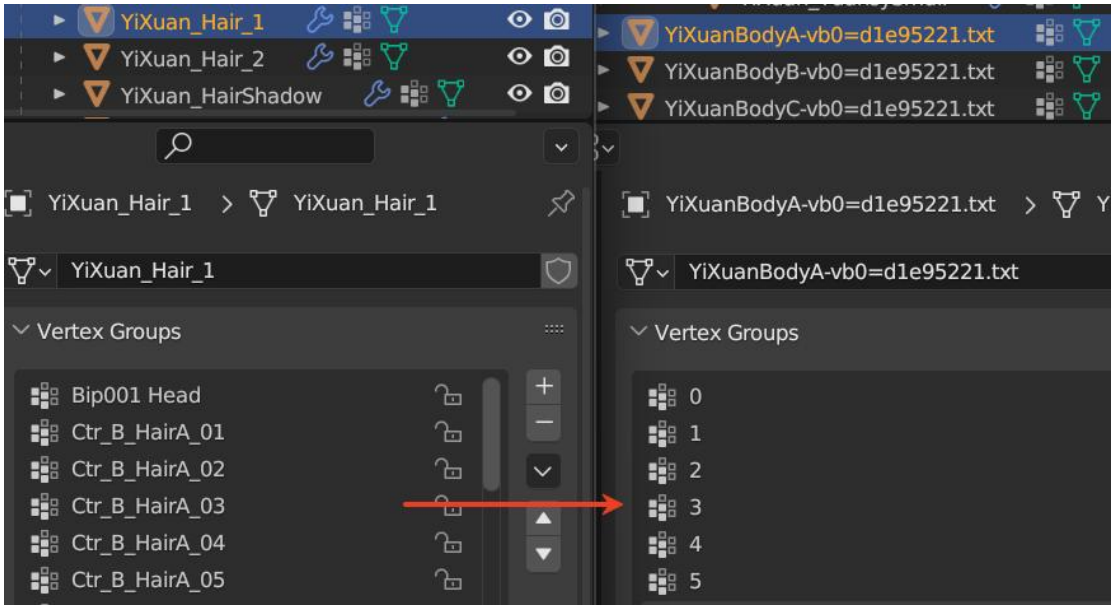
Some explanation:

One of the place You can find game unpack asset is the HoyoToon Discord server (invite link).

They provide characters or NPCs' assets for GI/HSR/ZZZ/WW.

Each one of these asset meshes correspond to one IB hash in mod. (LoD is out of consider.) Some of the asset meshes have more than one material. Each material corresponds to one mesh in mod.

Mod meshes dumped by 3dmigoto is a data structure compressed by unity and passing to d3d11. The VG names of the asset meshes are arranged and mapped to integers during the compress process. Therefore we are seeing a integer format VG name in the mod meshes. What we are going to do is to rename the bones according to this mapping, so that these bones can control the mod meshes.



2. (optional) Edit the armature

I don't know why these bones in the asset armature are pointing outside, instead of pointing their child, and i'm uncomfortable with it.

The script "bones_rotate_90_z.py", as it's name suggests, will rotate the edit bones and make it pointing to it's child like MMD armatures. Select the armature and run the script.

There is another script "bone_rename_LR.py". It will rename the bones with "L"/"R" in it's name to a format that blender can read, so that you can use the x-axis mirror function in pose mod. Again, Select the armature and run the script.

The result looks like this. I made a simple pose for examination later. The more metallic one is the asset model, the rougher one is the dumped model.

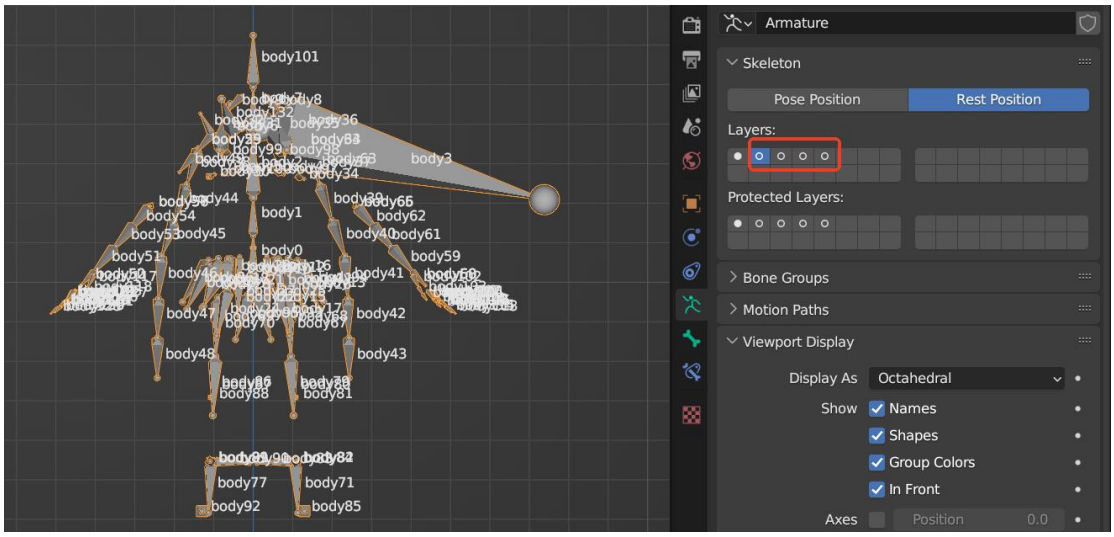
3. Add control bones for mod model

The script “add_bone_by_vg_remap.py” detect the VG mapping by comparing the VG weight of asset mesh and dumped mesh. There are 5 values in the main function that serve as input and need to be set before running the script:

```
def main():
    # source: name of an dumped mesh
    source = "YiXuanBodyA-vb0=d1e95221.txt"
    # target: name of the corresponding asset mesh
    destination = "YiXuan_TrailsofInk_Body_1"
    # armature_name: name of the armature you are going to add control bones
    armature_name = "Armature"
    # prefix: a special prefix to distinguish the control bones of this mesh to others
    prefix = "body"
    # layer: the bone layer the control bones going to be located
    layer = 1
```

Now run the script. You should find that control bones for the dumped mesh set as the source have been added to the bone layer you chose.

Do the same for other asset meshes until all dumped meshes you need in your mod got control bones in the armature.



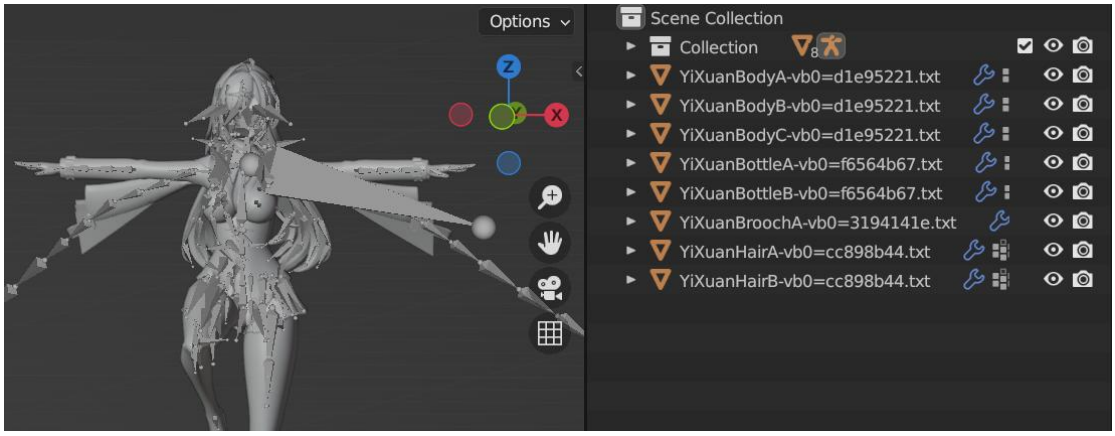
The preparation of armature is done for now. You can go back to the modding file if you do the preparation in a new blender file.

4. Add prefix for mod meshes

We added a prefix for the control bones in step 3. To make these bones able to control the mod mesh, we need to add the same prefix to the VG of mod meshes, too. This won't effect the mod export process, as i tested with XXMI.

The script "add_prefix_to_vg.py" can save your time. Set the "prefix" in the script, select a mesh, and run the script one by one.

Finally, append the prepared armature if it is save in external file, add armature modifier to mod meshes and you should be able to control it now. When making pose, you should only transform the origin bones in layer 0, all control bones added in step 3 are assigned as a child of origin bones.



Limitations

Beware of any warning when running the script in step 3. There might be some VG groups failed to find a match and therefore have no control bones added to the armature. You need to add them manually if that happens.

The armature from assets contains only the bones. There isn’t any bone constrains like IK, making it inefficient in posing, unless you add the constrains manually. I’m still experimenting on matching the asset armature with an official MMD armature. Maybe this will handle the constrains and physics all in once.