Lab 4: Morph Targets

Objective

- Implement morph target animation to interpolate between similar meshes

Goals

- Load in any number of morph targets and smoothly interpolate between them
- Export morph target meshes from blender and load them into the framework

Itinerary

Programming: Part 1
- Open MorphGameObject.h and take a look at the MorphGameObject class. Notice that we are inheriting from the GameObject class.
- In the MorphGameObject class you will see several member variables which will facilitate the morphing animation. Notice that we are storing an array of meshes, these are the morph targets.
- Open main.cpp and navigate to the InitializeScene function, this is how the morph targets are being loaded in. Take a look at MorphGameObject::addMorphTarget and get an idea of happening.
- Open MorphTarget.cpp and navigate to the update function. This is where you will implement the morphing logic. There is pseudo code here to help you get started.
- If the morphing is implemented correctly, you should see the cube get squished.

Blender
- Next up, you will play around with Blender to export your own morph targets
- You may use the provided "cube.blend" scene or make your own, the steps below will use cube.blend as the reference. If you are using your own model, scroll down and read the notes I provide.
- Once you open cube.blend, you will see a cube with two joints
- Right click one of the joints and go into "pose mode"
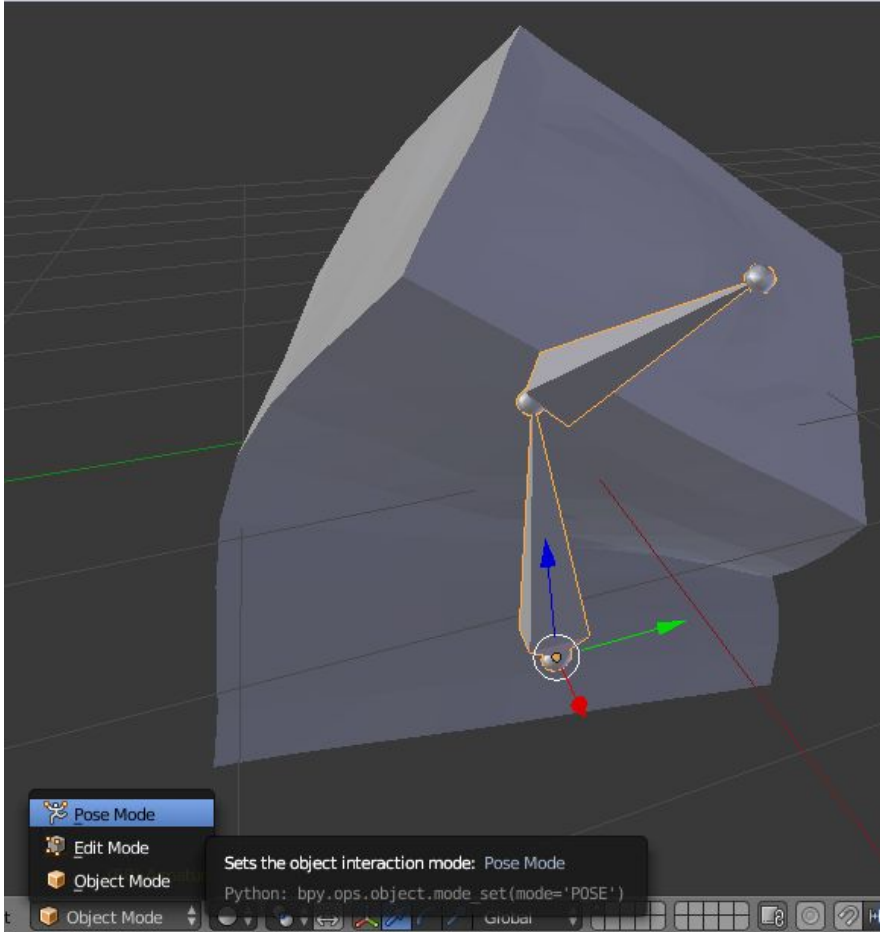
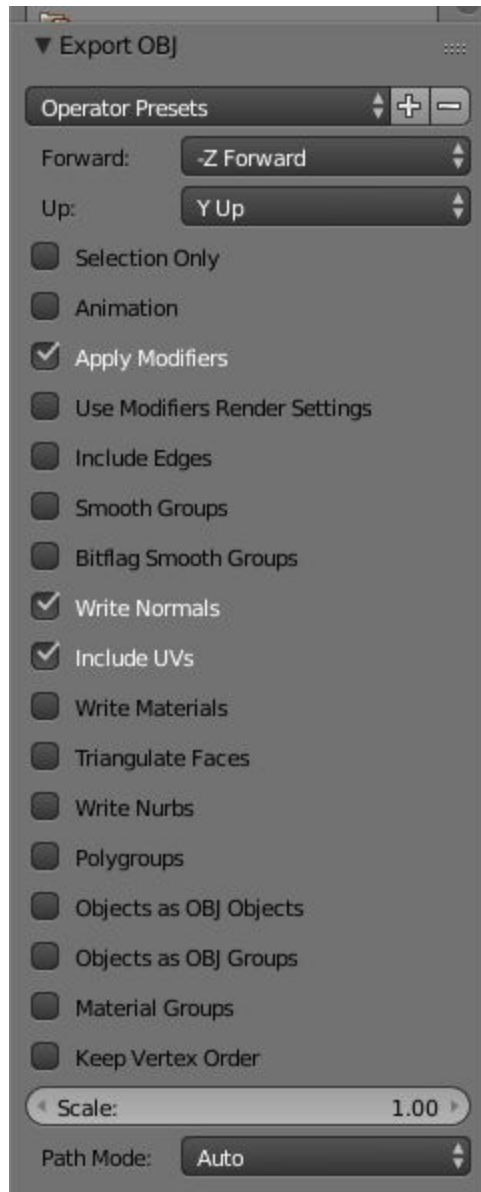- Once you are in pose mode, you should see some keyframes in the timeline:
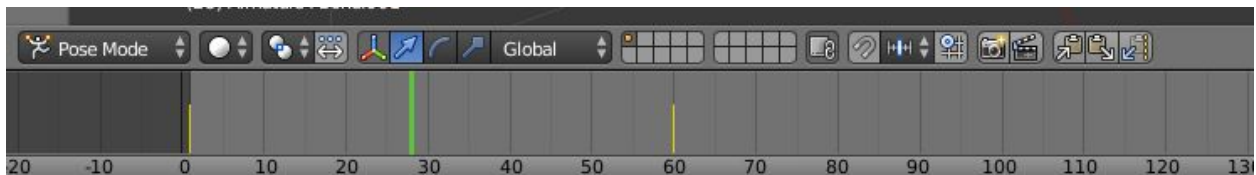
- The yellow lines represent the key frames and the green line is the current playback frame.
- To create a new key frame, click on the frame where you want to add a key, in the image below we are going to put a key on frame 120

- Now select a joint (right click it) and rotate it (press R) in some way

- Once you rotate the joint, you can press the "i' key and click on "locRotScale" this tells blender to save the current location, rotation and scale as a keyframe.
- Now you need to export the obj, to do this click "file > export > wavefront obj"
- Navigate to where you want to save the file
- Make sure you set the following obj export options

▼ Export OBJ

Operator Presets

Forward: -Z Forward

Up: Y Up

⬜ Selection Only
⬜ Animation
☑ Apply Modifiers
⬜ Use Modifiers Render Settings
⬜ Include Edges
⬜ Smooth Groups
⬜ Bitflag Smooth Groups
☑ Write Normals
☑ Include UVs
⬜ Write Materials
⬜ Triangulate Faces
⬜ Write Nurbs
⬜ Polygroups
⬜ Objects as OBJ Objects
⬜ Objects as OBJ Groups
⬜ Material Groups
⬜ Keep Vertex Order

Scale: 1.00

Path Mode: Auto

Notes If you are using your own model:
- Make sure you triangulate the object before you deform it
    - Go into edit mode, press "a" to select all vertices and press "ctrl + t"
    - You **must** triangulate this way and not using the triangulate option in the obj exporter. The obj exporter will give you artifacts.

Programming Part 2
- Go back to the C++ code and modify it to load in the target you just created.
- Once you have the animation working with all three keys create two more keyframes in blender. In total you should be submitting at least 5 keyframes.

Submission

Submit a zip file the following:
1. Any source files you modify
2. The assets folder

Make sure you put your name and student number in a comment at the top of each file!

Failure to follow these submission guidelines will result in a **zero**!

What's Next

Think about the following:
- Currently each of our morph targets take one second to complete, how could you add the ability to control this duration?