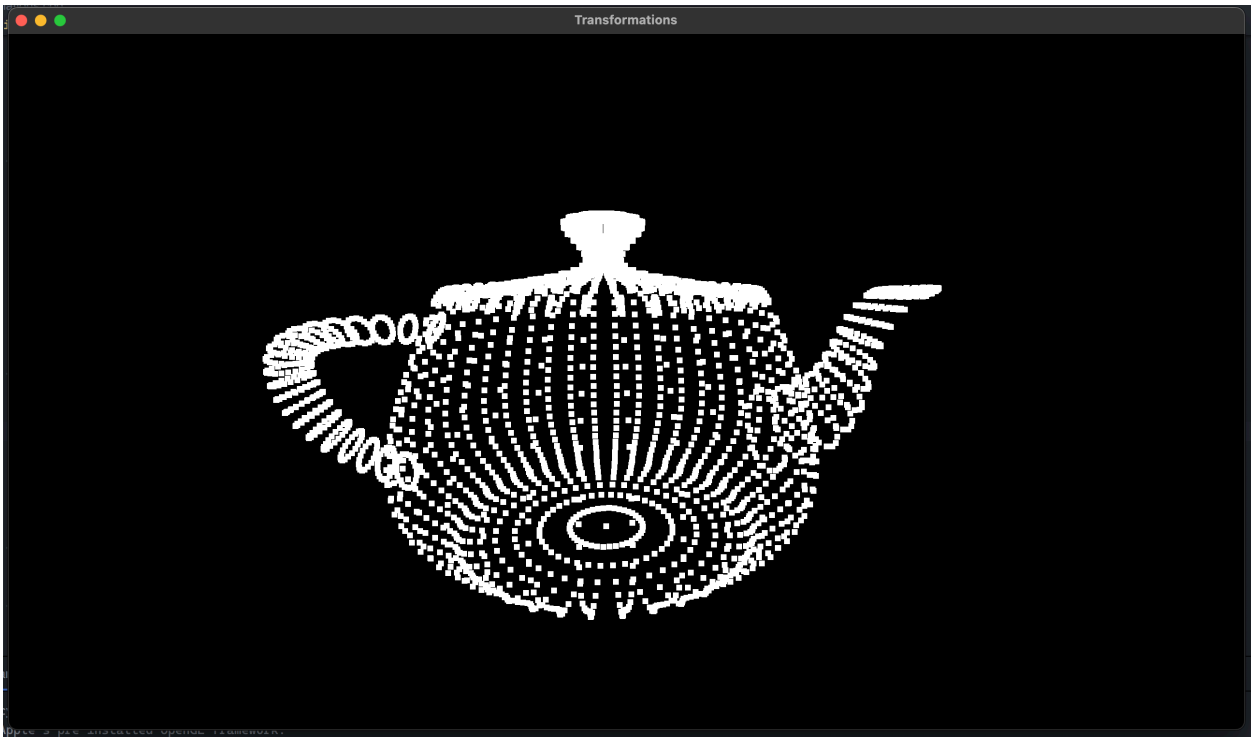
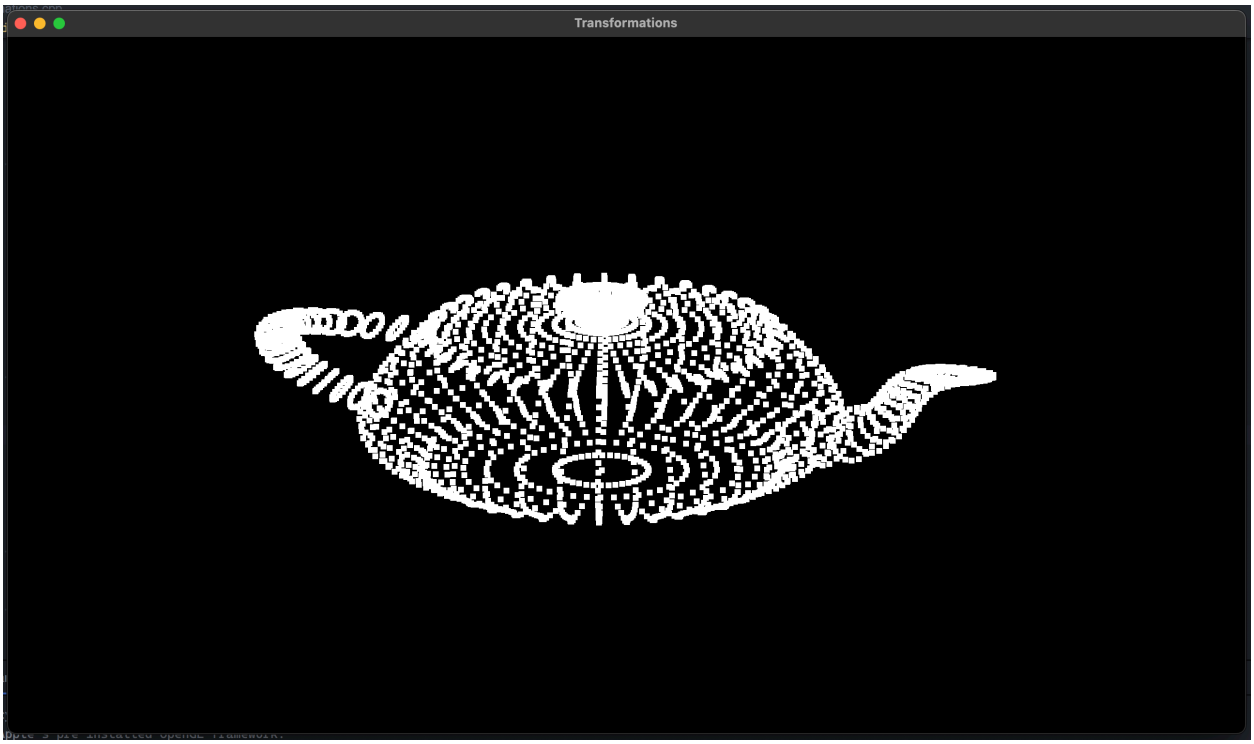
James Youngblood, CS 6610, Project 2

Perspective view:



Orthogonal view:



At the date of this resubmission (Feb 8), I implemented a point cloud renderer, using a .obj file's vertices as the points (using cyTriMesh.h to parse the .obj file). A VBO is created to store positions, a VAO and the OpenGL shader program is linked with those values, and points are transformed and drawn using matrix transformations from the cyMatrix.h code.

The mesh is translated so that the view is centered in the middle of the mesh's bounding box, and the camera is placed far enough away such that the mesh is visible in its entirety upon startup. Dragging the mouse orbits around the mesh by altering the transformation matrix, and dragging right click will zoom in and out. When F6 is hit, shaders are recompiled, and when P is hit, perspective mode is disabled in favor of orthogonal mode.

My code is a single .cpp file, with shaders written as string literals inside. (This means that pressing F6 doesn't actually do much.) My code depends on:
- GLFW (include GLFW/glfw3.h)
- GLEW (include GL/glew.h)
- OpenGL >= 3.3
- C++11 standard lib
- cyCodeBase headers cyVector.h, cyTriMesh.h, cyGL.h, and cyMatrix.h

To compile on my Mac M1, I installed GLFW and GLEW using homebrew (a package manager for Mac), including them and linking to their libraries using flags `-I, -L, -l,` for each when compiling with gcc. I also included the cyCodeBase headers in a similar way, and I linked to the pre-installed OpenGL distribution on macOS using the flag `-framework OpenGL`. Finally, I included the C++ standard lib using the flags `-std=c++11 -lc++`.

Here is the compilation command I used:

```
gcc -std=c++11 \
-I /opt/homebrew/Cellar/glfw/3.3.8/include \
-L /opt/homebrew/Cellar/glfw/3.3.8/lib \
-l GLFW \
-I /opt/homebrew/Cellar/glew/2.2.0_1/include \
-L /opt/homebrew/Cellar/glew/2.2.0_1/lib \
-l GLEW \
-I ../cyCodeBase/ \
-framework OpenGL \
-lc++ \
transformations.cpp -o transformations
```