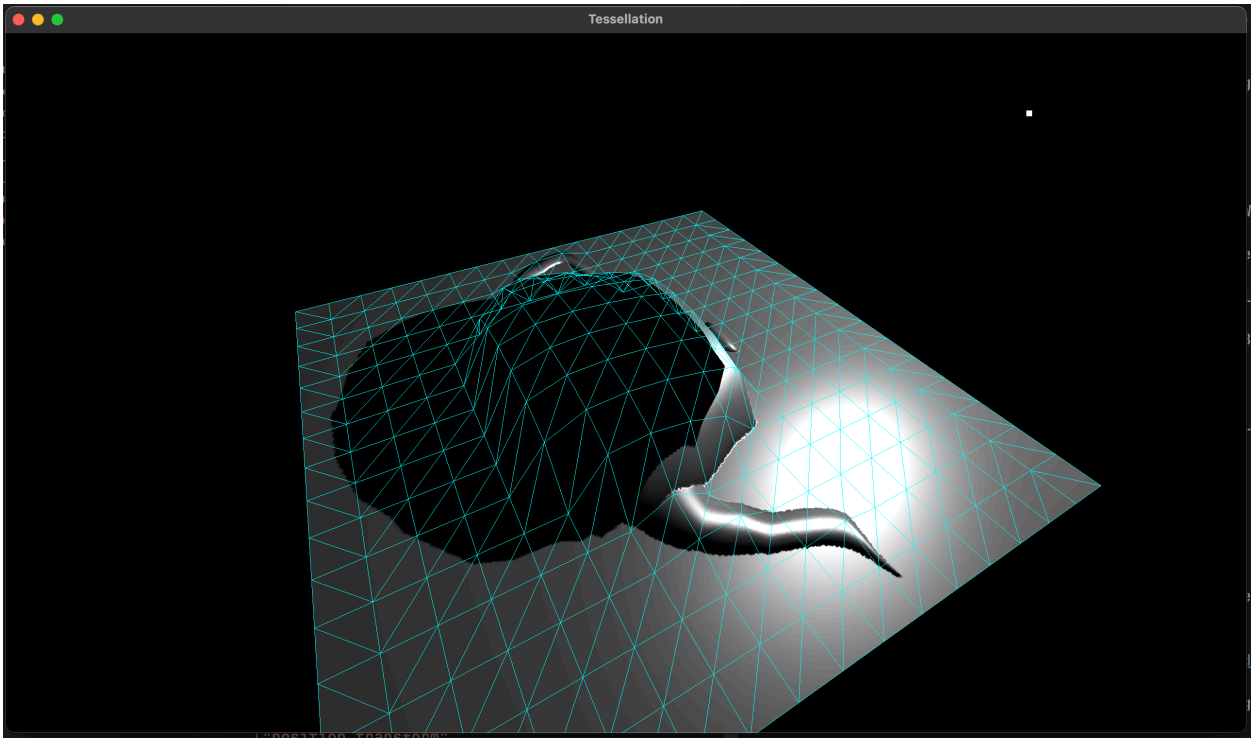
James Youngblood, CS 6610, Project 8



I am resubmitting this on Apr 7 to implement missing functionality in my first submission.

What I implemented
I copy normal and (optionally) displacement maps to textures. I create a plane as a patch, which gets sent to a tessellation control shader to apply variable tessellation levels based on a uniform parameter. The tessellation evaluation shader then applies transformations and displacement from the displacement texture to vertices. I have two shader programs, which render either a wireframe or the plane itself, based on the geometry shader they implement. I render a light view projection into a texture for shadows. In the fragment shader, I modify the normals by sampling the normal texture.

What I could not implement
As you can see, the shadows are not correct. I experimented with biases and transformations, rendered the light view projection to debug, created new shader programs to test, and I could still not figure out what the issue is.

The shadows on non-displaced vertices are correct, and are only incorrect near displaced vertices, as far as I can tell. This observation still didn't lead me closer to a solution. My shadow pre-pass displaces vertices correctly when I examine it on it's own, so I don't think that would be the issue.

Additional functionalities
None.

How to use the code
My code is a single .cpp file, with shaders included as string literals in the code, so it should be easy to compile and execute. You must pass a normal texture as a .png in the first argument, and optionally pass a displacement texture as a .png for the second argument.

What operating system and compiler did I use?
I used gcc as the compiler, on the latest version of macOS.

External libraries and other requirements for compilation
The use the following dependencies.

- 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.
- LodePNG

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. I included the LodePNG header, and compiled with the LodePNG .cpp file alongside my own. 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/ \
-I ../LodePNG/
-framework OpenGL \
-lc++ \
../LodePNG/lodepng.cpp \
tessellation.cpp -o tessellation
```