Marching Cubes in OpenGL

By Addison Miller

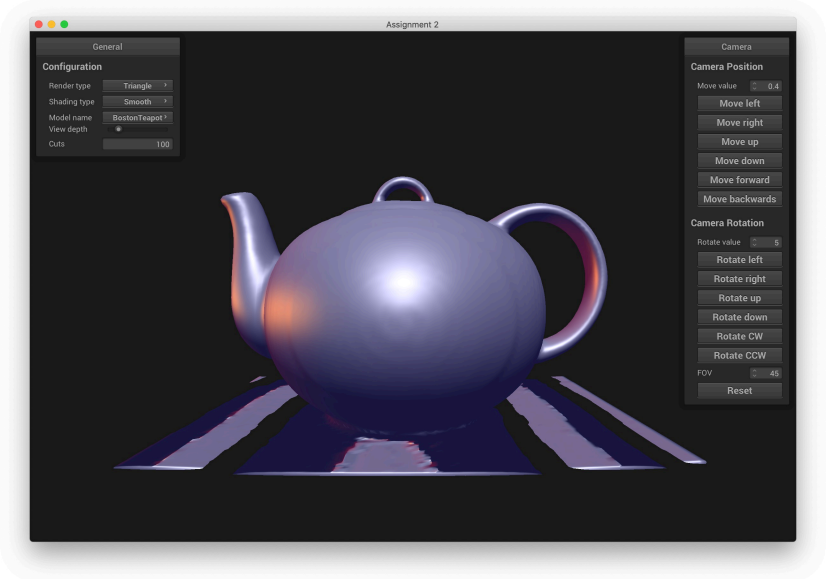
Problem Summary

I implemented the marching cubes algorithm in OpenGL and displayed the extracted mesh with smooth shading. The marching cubes algorithm allows volumetric data to be displayed in a much less resource intensive form than directly rendering it and allows further processing to be done on the extract mesh.

Description of Work

- I first had to implement a trilinear interpolation algorithm to approximate any coordinate from the volumetric data.
  - I found that I needed to surround the entire model with values of 0 so that the extracted mesh would have faces on the edges
- Next I implemented the marching cubes algorithm using lookup tables from online.
  - Quite a bit of time was spent figuring out how to index the edges and vertices of the cells into the lookup table.
- Finally I implemented smooth shading of the resulting mesh, which was much harder than I thought it would be.
  - My first approach was to use the gradient of the volumetric data to determine the normal at each point. This worked well in general, but created errors whenever the mesh got too thin and the gradient pointed in the wrong direction. (Figure 1)
  - I decided to instead calculate the normals using the average of all face normals at each vertex. This required me to change how I represented the vertex and face data so I could determine which faces touch each vertex.
- I spent quite a bit of time researching improvements to the marching cubes algorithm such as marching cubes-33, but I had to abandon this because I would need to generate new lookup tables and this was out of scope for this project. I haven't noticed any errors in the meshes I've generated, so I think most of the improvements are for edge cases.

Figure 1: Example of normal errors on the table below the teapot when using the gradient method.



Results

1. The user can choose what depth in the volumetric data to extract a mesh at. (Figure 2)
2. The user can set the number of cells used to sample the data. (Figure 3)
3. All models are displayed with smooth shading and lighting. (Figure 4-5)
4. The user can choose which volumetric dataset to use.

Figure 2: The same volumetric data with the mesh extracted at different depths.

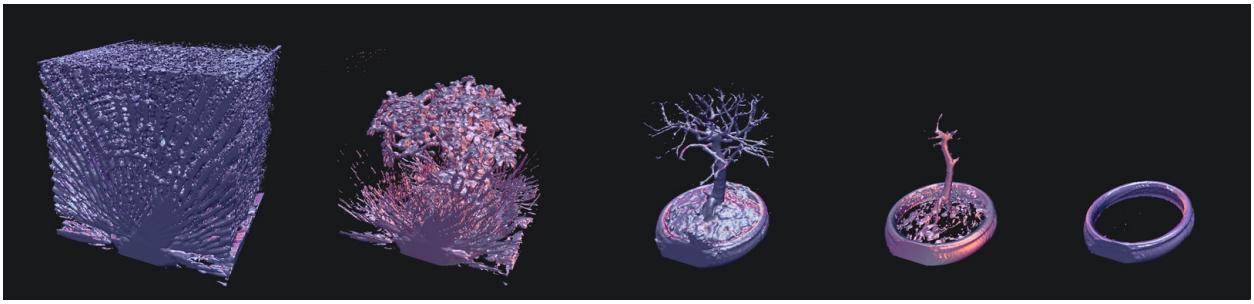
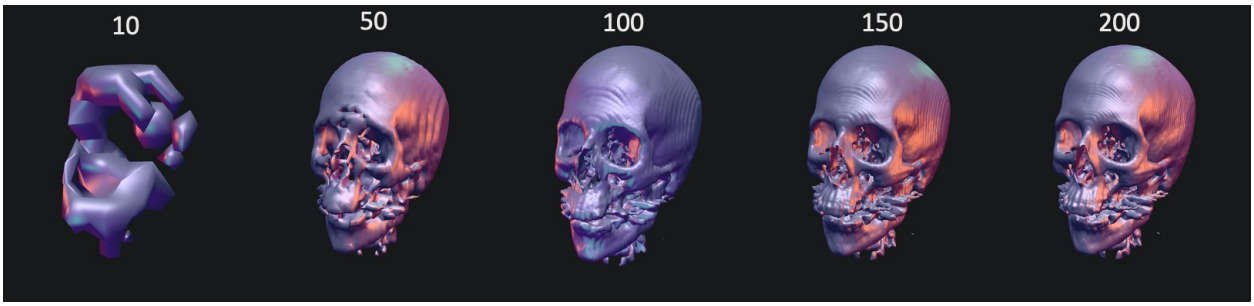
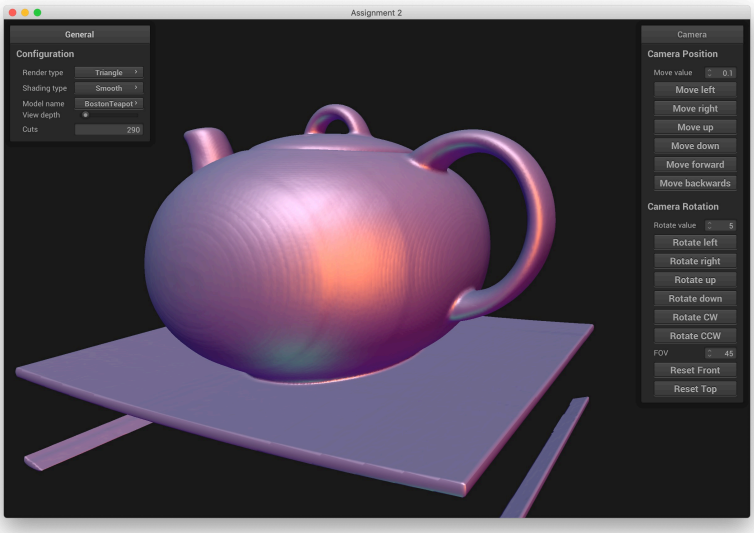


Figure 3: The same volumetric data with different amounts of divisions.



Figures 4-5: Smooth shading and lighting.



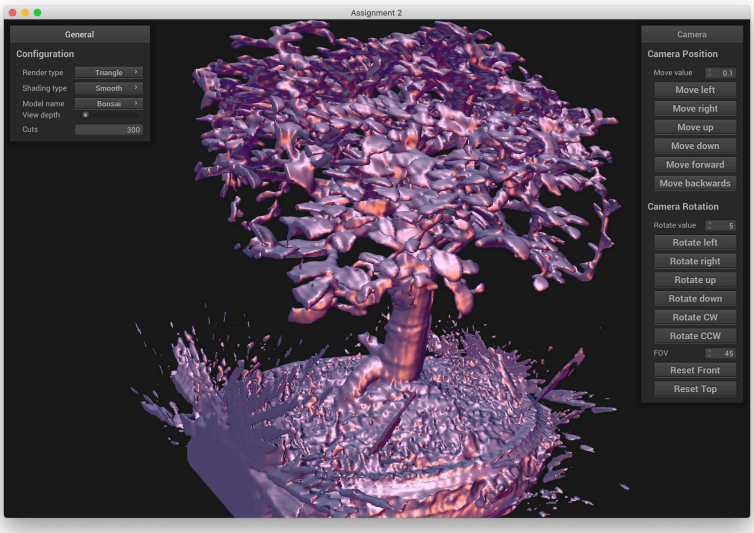
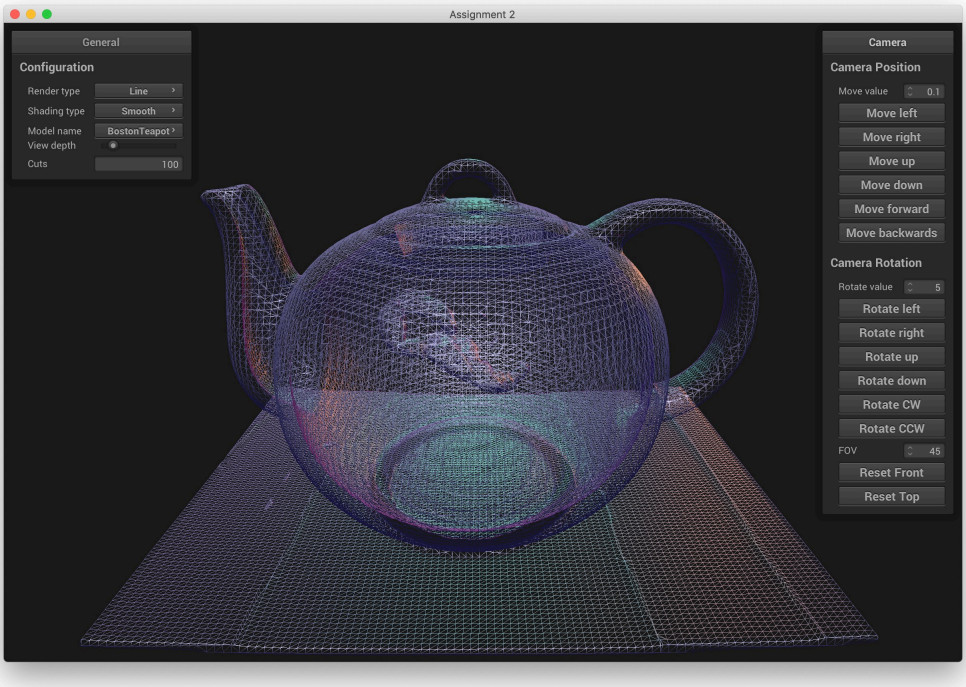


Figure 6: Wireframe render of an extracted mesh.



Analysis of Work

I was successful in meeting my original goals of implementing the marching cubes algorithm and rendering the result with smooth shading. Originally I had planned to just use the gradient normals, but after seeing some of the poor results that it gave, I decided to take it a step further and properly compute the vertex normals.

In my original proposal, I also wanted to add some of the Marching Cubes 33 improvements if time allowed, but I wasn't able to do this because I underestimated their complexity. The improvements involve splitting some of the cases into multiple sub cases, which would either involve generating new lookup tables or abandoning lookup tables altogether. Either way, it would greatly increase the complexity of the project and wasn't feasible to complete in time.