CS148 Final Project Report

Crystal Cave

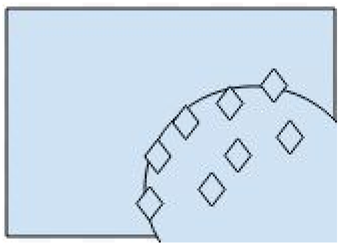
Amber Thomas (alt3), Andi Yang (andiy)

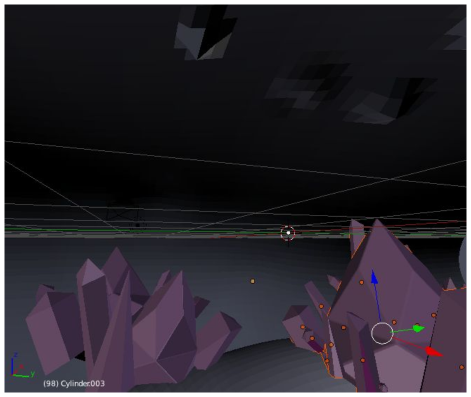
Project Statement

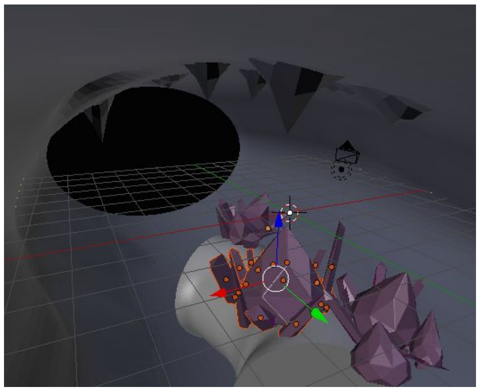
We wanted to use Deferred Rendering to generate an interesting scene with some light-emitting crystals, serving as the only light sources, and use texture mapping to create rocky surfaces. We wanted the chance to really explore the 3D modeling software and learn how to create a complex mesh structure, and interact with it in OpenGL. We were able to craft a scene using advanced lighting techniques referenced in the class. Many of the topics discussed in class, like phong shading and normal maps were useful while we were learning to fully access Blender's tool kit. The theme of 'hacking' a scene to make it look better came in handy while we were brainstorming ways to get around problems without adding excessive complexity to the project.

Modeling

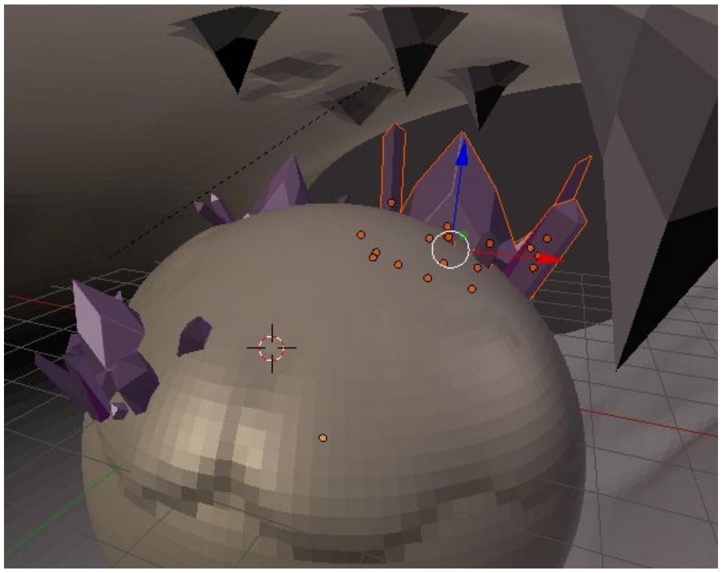
This was our first experience with any kind of modeling software, and after recommendations from classmates and we decided to use Blender. Before starting on any of the modeling for the scene Amber went through a few basic CGCookie tutorials (listed in the sources). Even with this bit of preparation, almost none of our first draft model made it to the final product. Initially, we wanted our scene to take place in a large cavern with all of the crystals resting upon a large rock dome, as shown in the simple mock up. However the composition of the scene never seemed to work out. Additionally our scene would only look good for a limited range of angles.



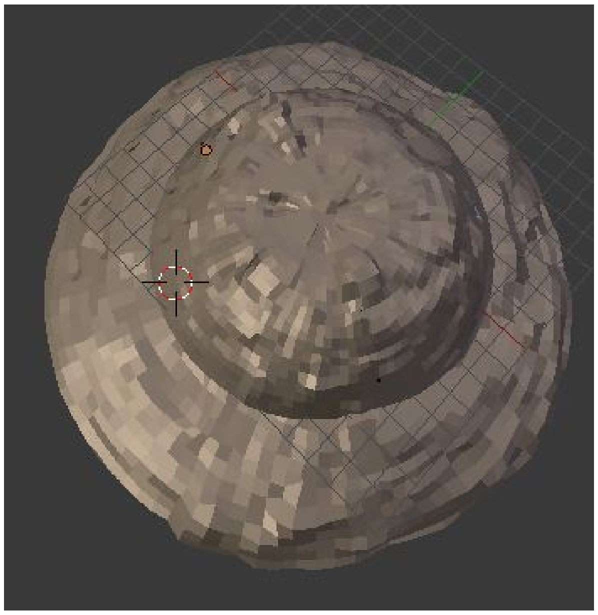




In our first attempt we used a cylinder for both our cave and dome after a tutorial suggested that cylinders texture better than spheres. We were unsatisfied with the effect and then tried to work with a sphere.

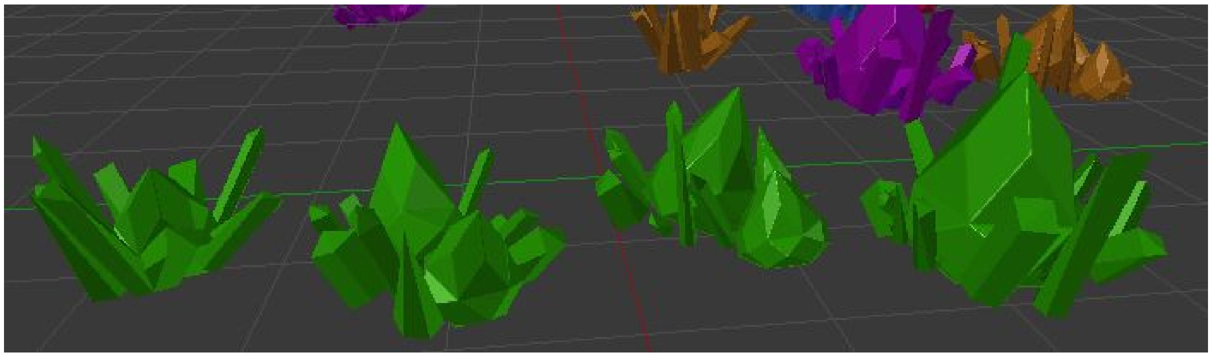


At this point we decided that the crystals should feature more prominently in the scene. We also wanted the cave to feel a bit more claustrophobic and hoped that a smaller space to light up would bring out our texturing work on the walls. Lastly, we decided that we wanted to be able to navigate the camera through the scene. This lead to our final cave design, shown here.

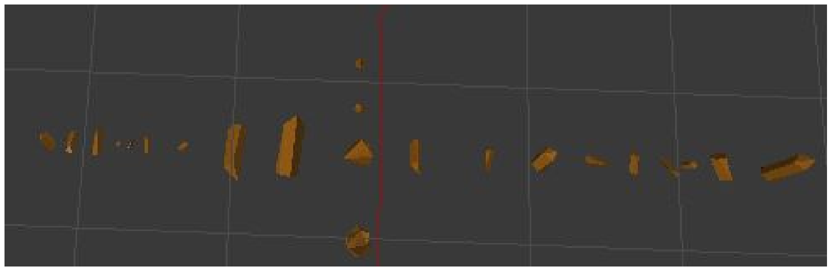


To get more interesting cave geometry we bent the mesh and applied the blender displacement modifier to add a rock like jaggedness. For the texturing on the walls we applied a color map, specular map, and normal map. We experimented with a few different texture maps before settling on a package supplied by the youtube tutorial cited below.

There are 4 different types of crystal clusters that were replicated, adjusted, and placed in the cave by hand.



There are also several distinct fragments that were collected into a vertex group and then spread across the cave surface using the blender hair function. To get a good looking distribution of fragments we experimented with the size, number, rotation, clumping and jiggering parameters.



In addition to generating the model, we had to figure out which modifiers and tools in Blender would properly export. To do that we looked through the mtl file format and tried to figure out what kinds of information would be stored. We experimented with the transmission filter, optical density and dissolve without satisfactory results. We had to do much Blender to mtl troubleshooting, which will be discussed in the next section.

Deferred Shading and Lighting

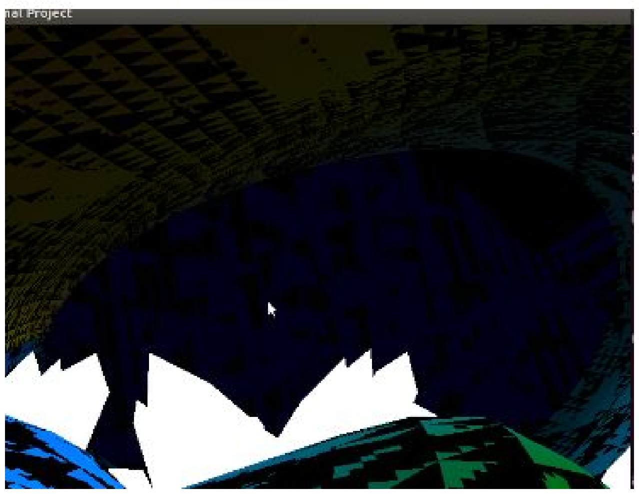
In this project, we explored the Deferred Shading technique. As introduced in lecture, in the first geometry pass of Deferred Shading, we keep all the geometrical information, such as the positions and normals, inside of G-Buffer. In our implementation, we stored in our G-Buffer the position vectors, normal vectors, color and specular values.

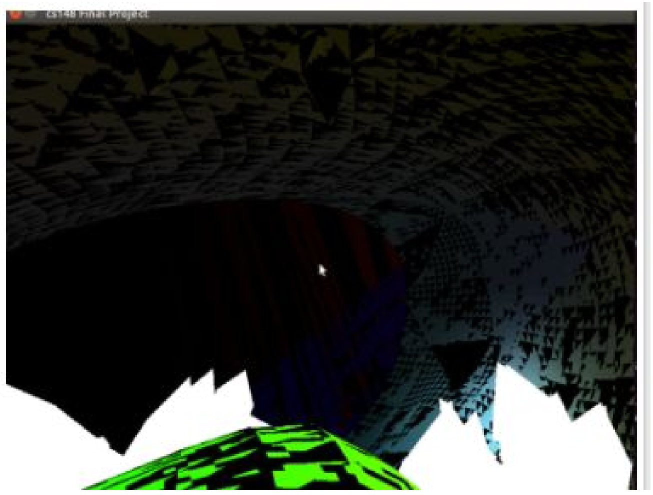
In the second lighting pass, for each fragment that ends up in the G-Buffer, we calculate its lighting. In our project, we compute the lighting using the Phong lighting model. As we did in homework 3, we calculated the ambient, diffuse and specular lighting for each light source. Note that we have a total of 48 light sources in our cave (not all crystals are emanating light). We also experimented with more lights (64, 96, and 128 light sources), but that just made the cave scene a little too bright.

In this project, we used the idea of light volume, as introduced in lecture, to shade fragments affected by any given light source within a certain distance. The light volume allowed us to make it appear as though the crystals were emanating light. The center of the light volume was placed at the base of a crystal. We gave the crystal a very high shininess factor, so that when it was enclosed by the light volume it would appear to glow. In addition, we took into account light attenuation, which is inversely proportional to the distance and distance squared from any light source. As the crystals are the only light sources in the scene, light attenuation allows us to have a rough but similar effect as reflection (but only reflecting once from crystals). The reasoning is that pixels which receive more crystal lights will be brighter than pixels that receive little light. As a result, a rough shape of a glowing crystal would appear on the body of a different crystal. To make the scene more interesting, this effect was only done on the surrounding crystals, not the center one in each crystal cluster.

Intermediate Result:

At this point, we still had quite a few things to figure out. For example, we had a very strange flickering pattern in the background and the crystals were appearing as just bright white objects, instead of our desired effect of emanating light.





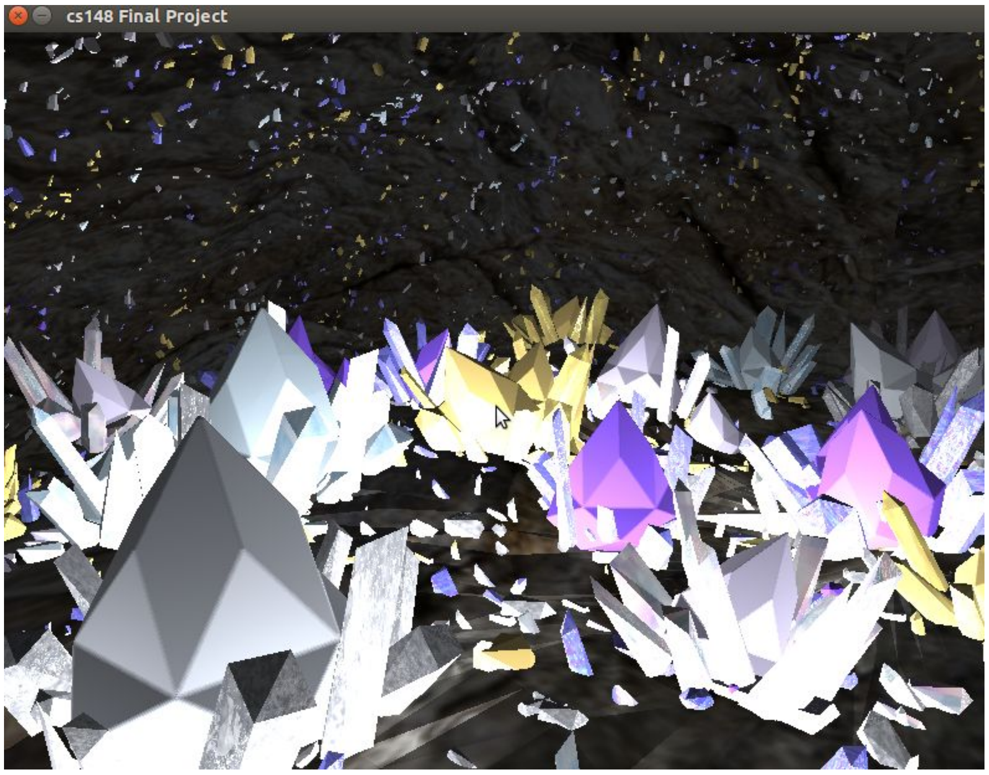
Challenges

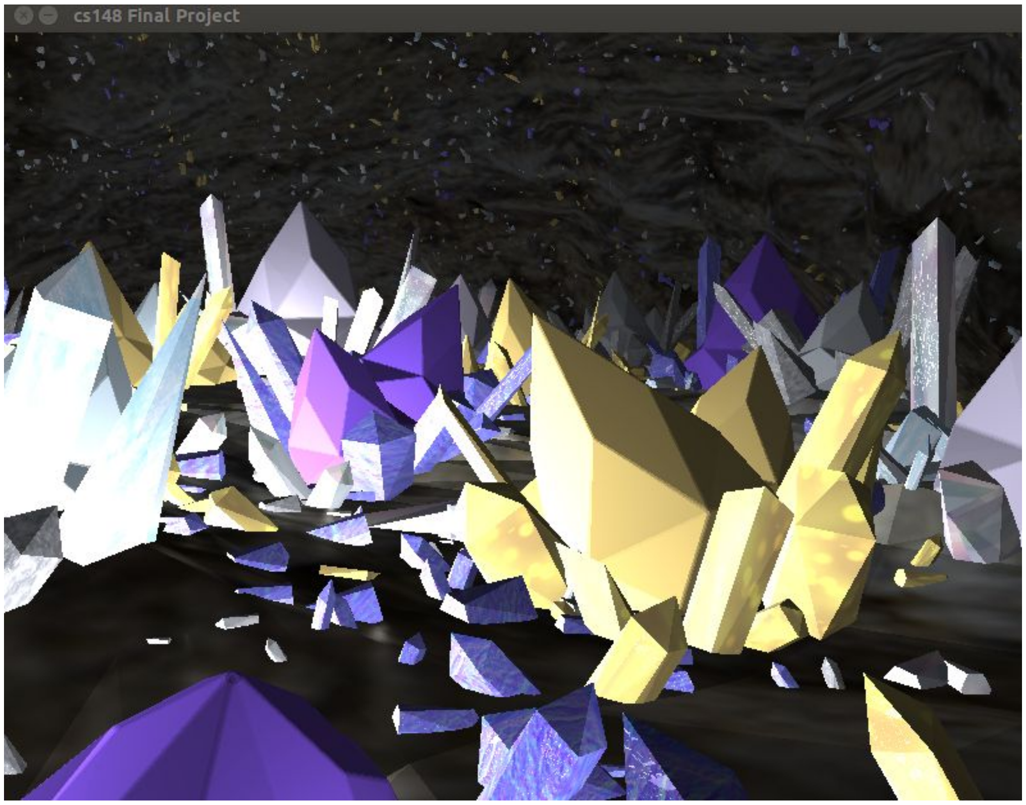
Combining the Blender outputs with our code is probably the biggest challenge that we encountered. To fix and refine the intermediate results (as shown above), we first examined all the Blender export options and made sure that the triangles and normals generated from Blender were correct. In addition, we tested and excluded the possibility that face culling is causing the trouble.

At our first stage, we did not assign textures because we wanted to keep the scene simple and make sure that it was working before we move on to our second milestone, in which we would handle textures and lighting. It turned out that using the incorrect normals and not assigning textures were the causes for the strange flickering background we had in the first milestone scene. However, after we assigned some simple textures, the objects all turned into black. To solve this problem, we continued with studying the .obj and .mtl file formats.

After going to different TAs' office hours and spending several days on this issue, we eventually found out that two separate things in the material file were causing the trouble. The first cause is that Blender automatically generated certain information (such as our project name) and embed it inside of some parameters in the .mtl file. The second cause is that for our model loading library *ASSIMP*, it does not recognize the 'disp' statement generated by Blender. Finally, after we fixed the above issues, we debugged and double checked our code with various 3D models that we designed, from the simplest triangles and cubes to the more complex crystal clusters.

Final Results:





Division of Labor:

Amber is in charge of Modelling, scene design, and Blender to .obj troubleshooting. Andi is in charge of architecting the code, loading models, implementing Deferred Rendering as well as lighting, .mtl and code troubleshooting.

References:

[1] Blender Tutorial: https://www.youtube.com/watch?v=1J4r0mt9zz0
CGCookie: Blender Basics, Mesh Modeling Fundamentals, Fundamentals of Texturing
[2] .mtl file specs: http://www.fileformat.info/format/material/
[3] Deferred Rendering Tutorial: http://www.john-chapman.net/content.php?id=13
[4] Deferred Shading Tutorial:
http://miss-cache.blogspot.com/2012/08/lighting-transparent-surfaces-with.html
[5] Deferred Transparency:
http://martindevans.me/game-development/2015/10/09/Deferred-Transparency/
[6] Lighting Tutorial, Model Loading Tutorial and Deferred Shading Tutorial:
http://learnopengl.com