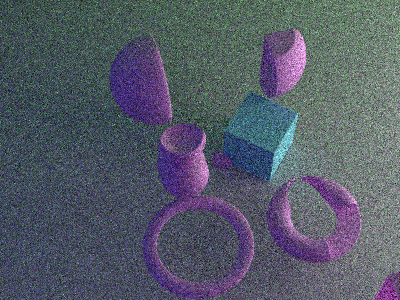
**Project Report**

For the final project I implemented raymarching and used it to add transformations, csg functionality, new shapes, and a few misc extras. I also implemented transformation functions. I added the ability to translate, rotate, and nonuniform scale the raymarched objects. Figures 1 and 2 show the testing scenes I used to make sure the transformations worked.

A picture containing grass

Description automatically generated

Figure 1: CSG Testing 1 Figure 2: CSG Testing 2

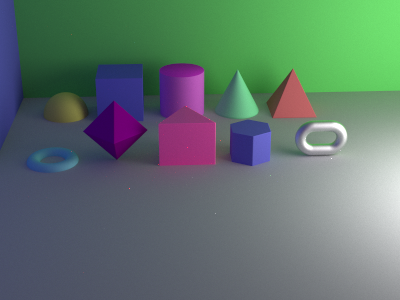
For the extra shapes I implemented the original sphere, cube, and cylinder with distance functions. I also added distance functions for the cone, pyramid, torus, octahedron, triangular prism, hexagonal prism, and link. The result of this can be seen in figure 3.

Figure 3: Various Shapes

CSG functionality was added with union, intersection, and difference. I tested this by making the classic shape seen in figure 4

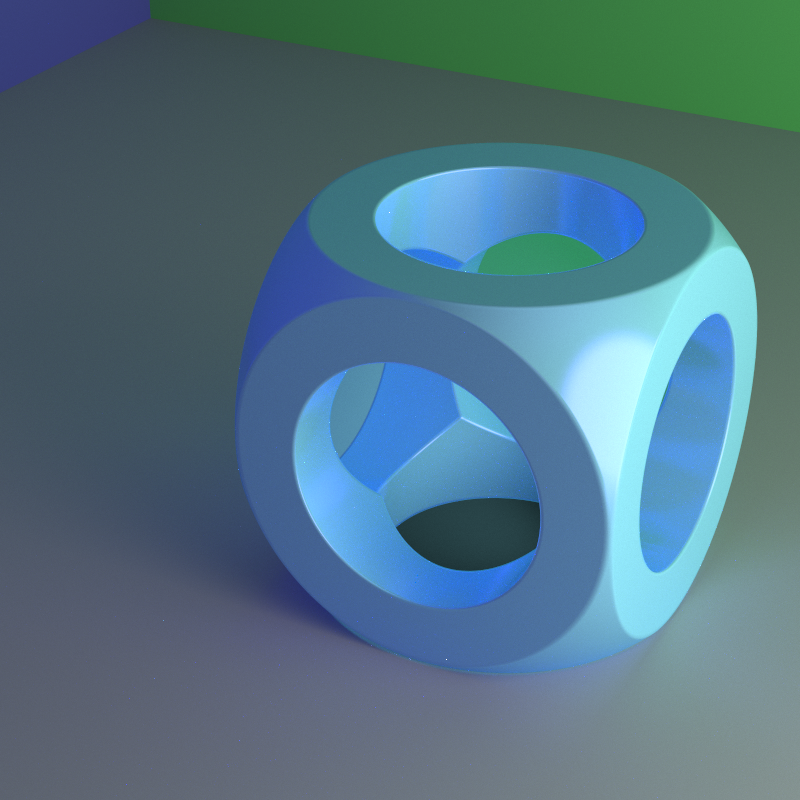


Figure 4: ITS THE SHAPE

Then for a bit extra I added twist and infinite field functionality. The twist can be seen in figure 5. For the twist I definitely noticed that it ran very slowly so it was hard to render an image with a lot of passes.

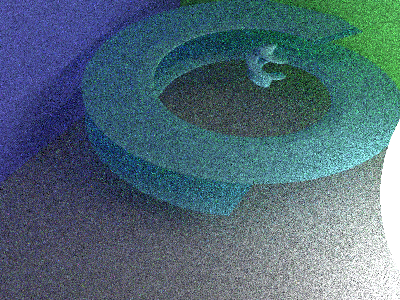
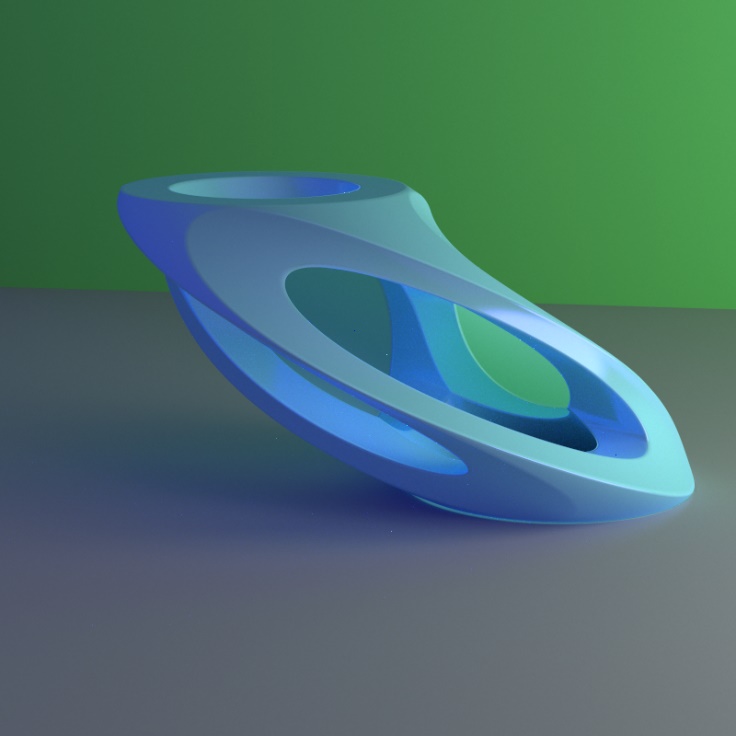


Figure 5: shape with a twist Figure 2: A twist of testing

For the infinite field I added the ability to make the field extend in one, two, or three directions. In the scene file the syntax I used was:

field\_inf repetitionPeriod1

field\_inf repetitionPeriod1 repetitionPeriod2

field\_inf repetitionPeriod1 repetitionPeriod2 repetitionPeriod3

This allowed for the specification of the number of directions to extend in as well as the repetition period, which was the distance each object was from the next in the field. This was very useful but I had to be careful since if the repetition period was too small, the object would intersect itself to create strange behavior. Another thing is that the bounding boxes for the fields is the whole screen. This makes sense for the 3D field but there’s prob a smarter way to calculate the boundaries on the 1D field.

I combined everything that I worked on (except for fields because they add a ridiculous amount of render time to the scene oh my god) in figure 6. You can see the use of csg shapes with the torus shape on the walls and the link shape on the ground. I used translate to move the ray marched objects around the scene. I used rotate and scale to tweak the look of some things, like flattening out the walls and rotating them 90 degrees which made the pattern much more interesting. I also used a lot of unions to be able to move things around as a unit. I used the 1D version of the field function for the chain and the 2D version for the walls. Since objects can’t intersect with themselves within a field, to make the chain work I had to use two fields of links and experimented a lot with transformations to make them intersect correctly. Finally I added a bunch of colored lights. This had nothing to do with anything I worked on but I just thought it would look fun (this scene was actually the first time I experimented with multiple lights or multiple colors so I’m glad the untested parts of my code held strong).

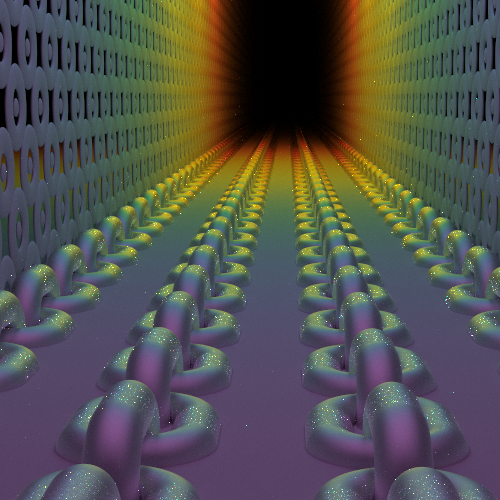


Figure 7: Something colorful