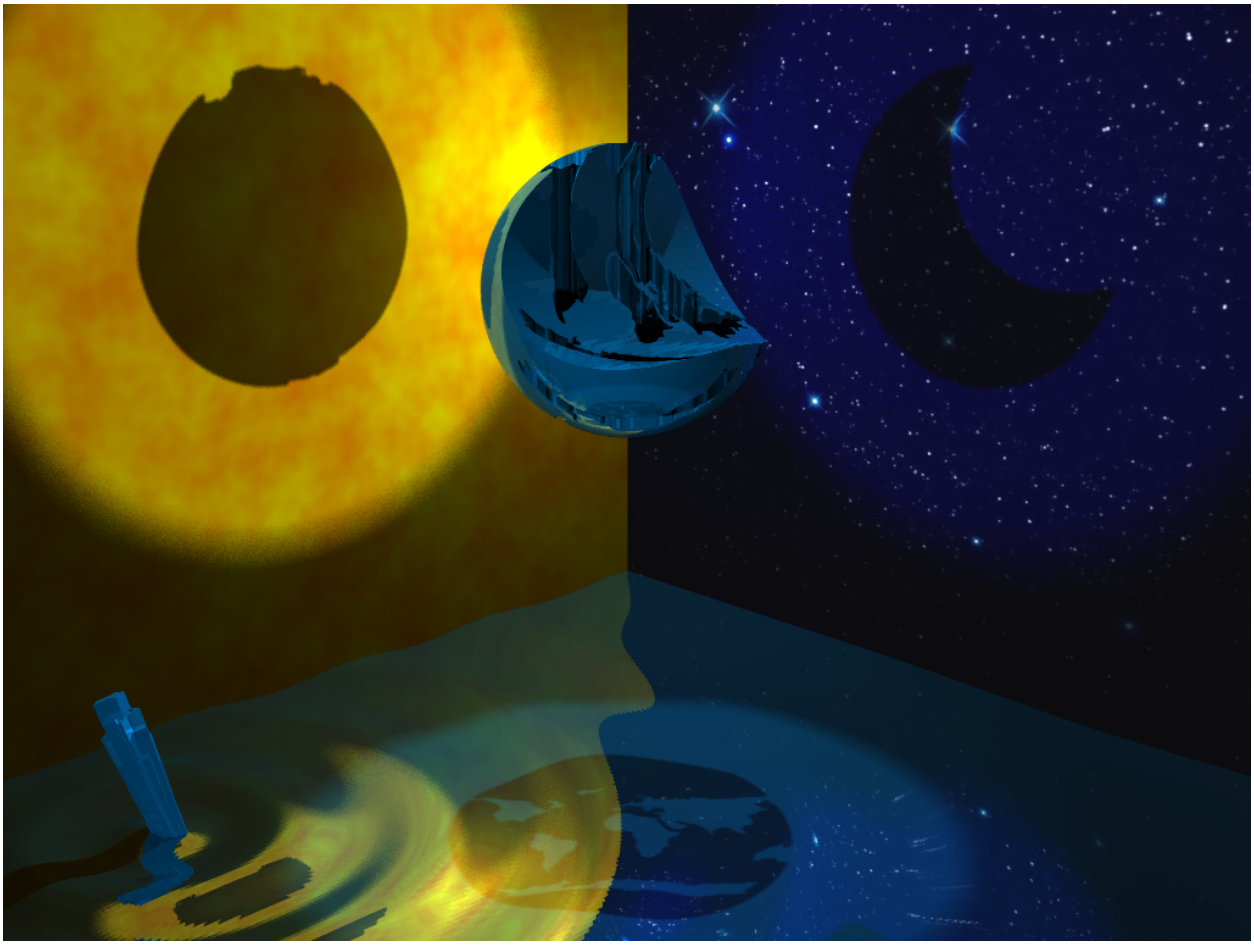
Celestial Triality

Reuben Britto
CS 148 Final Project, Fall 2018



My scene centers around a single object that casts three orthogonal shadows: a sun, a moon, and an earth. I wanted to capture the triality of our corner of the solar system, which is dominated by those three celestials.

The surfaces on which the shadows lie also symbolize those three objects: on the left, the surface of the sun, on the right, the starry night sky—where the moon is often found—and below, a rippling pond—something unique to earth.

Scene Details

Central Object – custom made in Cinema4D
The central object was modeled in Adobe Illustrator and Cinema4D. I found a simple online world map and used Adobe Illustrator to trace it and export it to Cinema4D. In Cinema4D, I extruded the map through a sphere and used a boole to carve it out of the sphere. I then used a cylinder and another boole to carve out a crescent into the sphere in an orthogonal direction.

The object was exported as an .obj without a texture. The material parameters were set in the ray tracer. I chose a deep ocean-ish blue to pair with the water below and I made the object reflective. The color, the reflection of the stars towards the top of the object, and the reflection of the circular ripples towards the bottom all come together to give the object an emerald glassy look that I like!

I also really like the chasms that the continents carved into the sphere. They draw your eye, showcase the complexity of the central object, and add an artistic earthly void.

Left Wall – procedurally generated with my own Perlin noise algorithm
The left wall is a blank plane with an image texture added in the ray tracer. The image mimics the surface of the sun and was procedurally generated in python. I implemented a Perlin noise algorithm based off lecture and resources found online, including Ken Perlin's paper! To quickly recap Perlin noise, it mimics the randomness found in nature by 1) limiting randomness to a grid within your image and having each pixel's value be an interpolation between the corners of the grid and 2) calculating randomness at different orders of magnitude (i.e. grid density) and summing them with different weights.

My Perlin noise algorithm details (Fig 2):

- My randomly generated gradient normals spanned all 2D space.
- I used linear interpolation paired with the following fading function: $6x^5 - 15x^4 + 10x^3$.
- I generated four images with frequencies of 8, 16, 32, and 64.
- I summed these images using a persistence of 0.7, which I found gave the graininess that looked sun-like to me.
- To give the color palette of the sun, I only varied G with Perlin noise and set R to 1 and B to 0. This skewed the color more orange than I liked, so I hacked it a little bit by multiplying all G values by 1.5 and adding an if statement stating that if any G values exceeded 1, set them to 1.

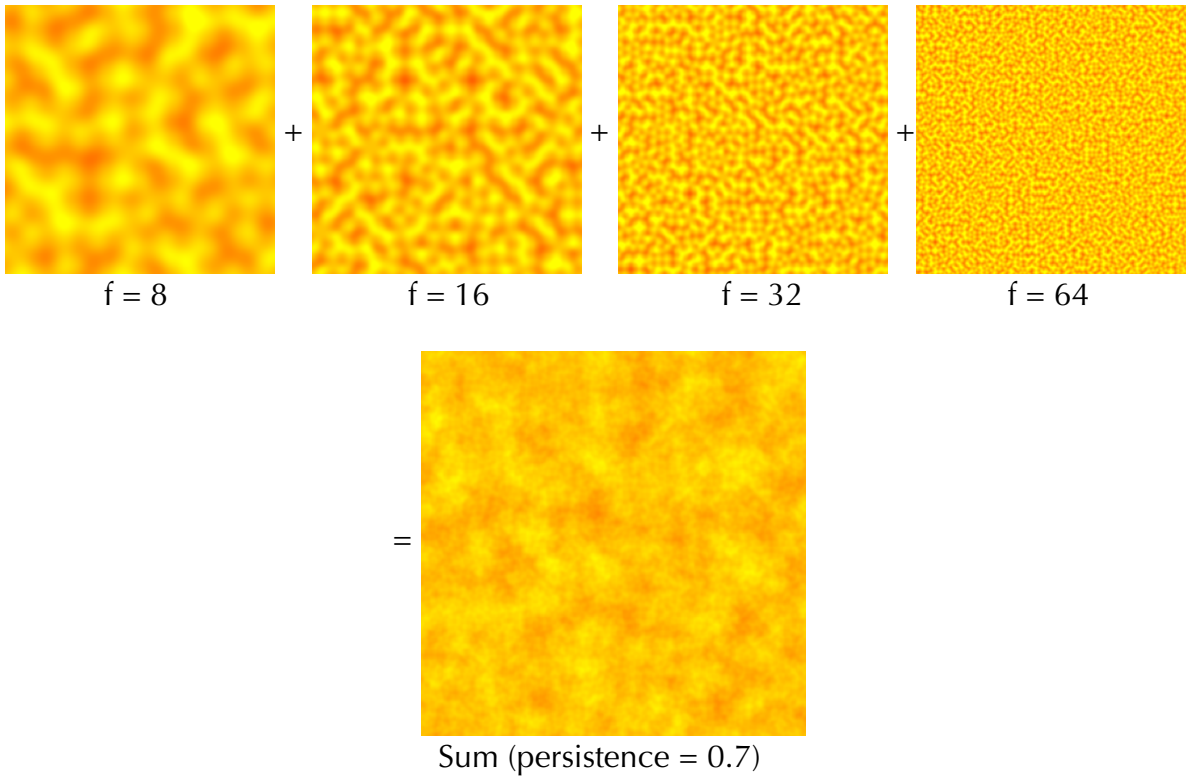


Figure 2. *Four Perlin noise images I summed to create my final sun surface texture. By weighting the lower frequency images more, we preserve the big orange splotches but also get the graininess we associate with the sun's surface. Note: these aren't the actual four images that were summed since I had to run the code again to isolate the individual frequencies' images.*

Right Wall – starry sky found on the internet

The right wall is a blank plane with an image texture added in the ray tracer. This image was found online (https://www.desktopbackground.org/wallpaper/beautiful-night-sky-with-stars-wallpapers-invitation-templates-144115). I chose this image because the stars were bright and isolated against a deep blue/black background which would look really cool reflecting in the water and on the central object.

Glowing Shard – found on turbosquid

The glowing shard is made to look like it could be the source of the ripples. It also could be a piece of the central object that just broke off. I found the crystal object in turbosquid and used Cinema 4D to eliminate the texture and to size and place the object. I matched the material parameters of my central object to bolster the appearance that it could be a shard that fell from the object. I also added some ambient lighting to achieve the glowy look.

Floor – plane with normal mapped texture of a ripple

The floor is a 3D plane that has been molded to the shape of a ripple using the Cinema 4D formula tool. To most accurately model a ripple, one has to sum Bessel functions which was not compatible with the formula bar in Cinema 4D. Instead, I used the following mathematical approximation which works by decaying the cosine:

$$\left(\frac{A}{1+u}\right)\cos\left(\frac{Bu}{\log(u+2)}\right)$$

The plane had no intrinsic texture so to make the ripple water-like, I made it reflective. This worked really well due to the low light conditions. My scene is also surrounded by colorful walls so I got some nice reflections of those too.

Lights

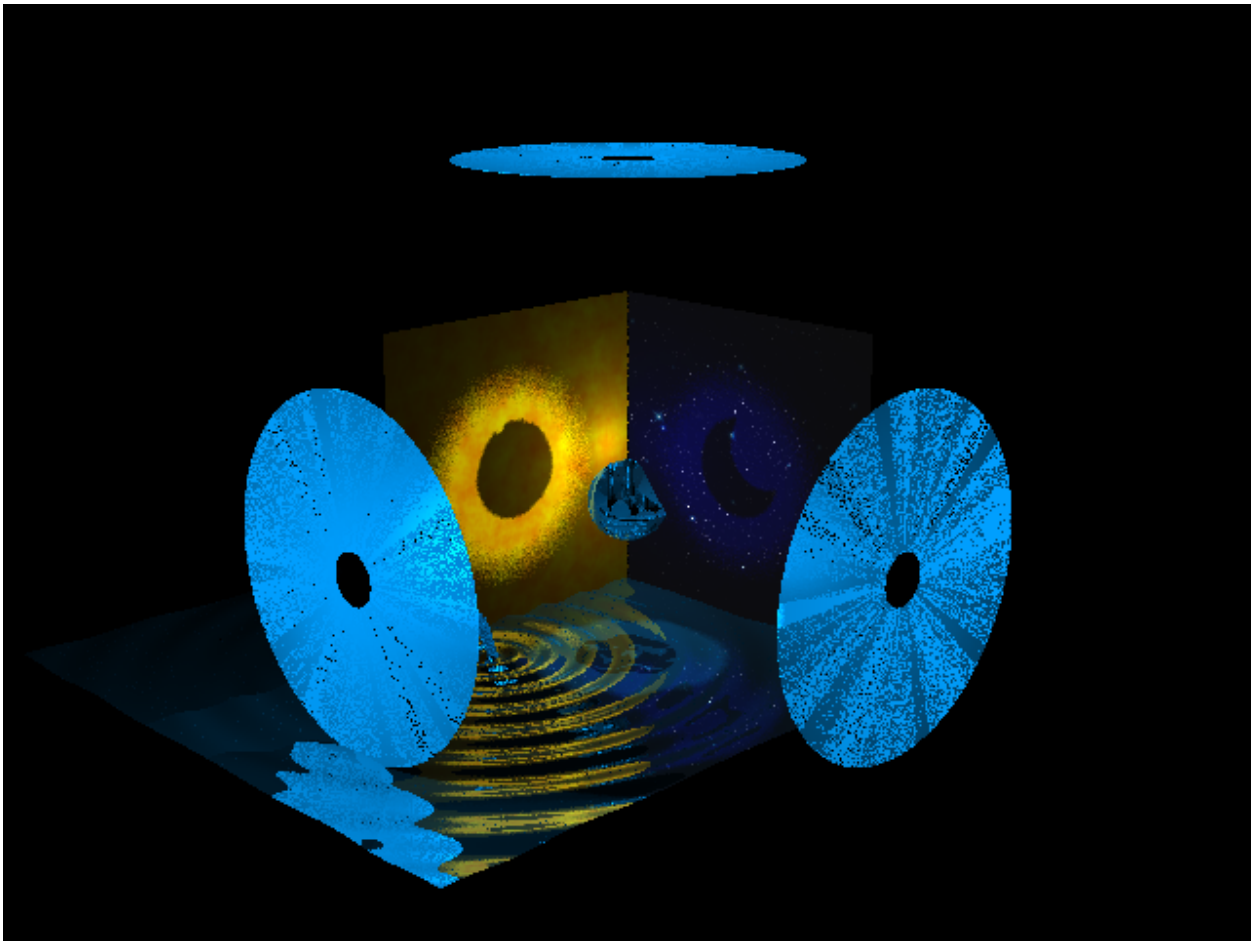
I implemented a total of four lights: three area lights and one point light. The three area light point in orthogonal directions and make each shadow separately. I added a ring object in front of the area lights to effectively make them spotlights to highlight the shadows and not over-illuminate the scene. The point light then acts as global illuminator and provides a dim light to the rest of the scene.

Work Breakdown

I completed this project individually, primarily because I'm a 5th year grad student and didn't want to try to make new undergrad friends. In hindsight, pairing with someone who knew C++ well would've been very helpful.

Variant A: different camera view

I just zoomed out since this shows the overall box I constructed and the discs I used to mimic spotlights.



Variant B: no textures, all gray