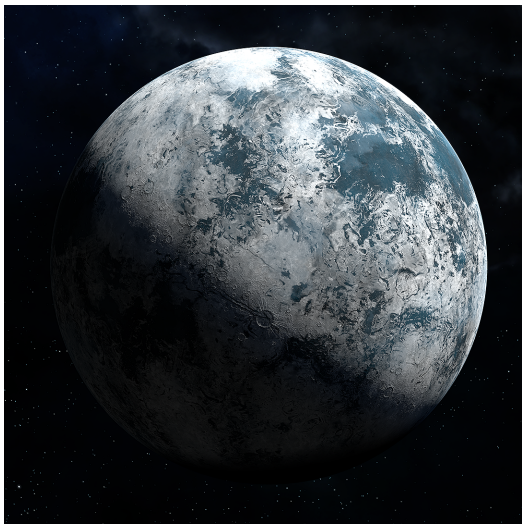
James Youngblood, CS 6610, Final Project Proposal



I propose rendering a procedurally generated planet as my final project. This will be an individual project. The completed project might look something like the image to the right.

I would like to implement this project in Rust, using the graphics library "wgpu" (https://wgpu.rs/) and the windowing library "winit" (https://github.com/rust-windowing/winit).

The first, and simplest, component of the render would be the starry background as an environment map. There would be a sun on the environment map, and I would define a directional light coming from the sun.

The second component would be generating a cube-map texture and height-map for the planet—the "procedurally generated" component. I anticipate 3D noise functions will be required to get these textures to be seamless when mapped to a sphere/cube. I would use some noise library similar to "noise-rs" (https://github.com/razaekel/noise-rs). Some tinkering will be required in order to get realistic and aesthetically pleasing planet surfaces.

The third component would be the planet, defined with a spherical mesh. The planet would be Blinn-shaded. Fragments would sample from the cube-map texture for color and vertices from the height-map for displacement mapping. The planet mesh would be set to rotate on it's axis, creating a day/night cycle.

The fourth component would be a "cloud layer", rendered as another spherical mesh, slightly larger than the planet sphere. This would require another cube-map texture to be generated. It would be set to rotate similarly to the planet mesh, but the rotation speed would be shifted, to give the illusion of drifting clouds. Depending on how much time I have, I may experiment with more sophisticated methods for moving clouds.

The fifth component, which might also be subject to time constraints, would be "atmosphere scattering". I plan on adapting this example on Shadertoy into my own code: https://www.shadertoy.com/view/wlBXWK. It uses ray-marching to generate atmospheric colors. Because of the simplicity of the scene, I don't believe the cost of ray-marching will be preventative.

The controls would be simple: drag left-click to orbit, drag right-click to zoom, press "c" to toggle clouds, and press "f" to make the camera follow the planet's rotation.