The first thing we’re going to do is neaten up our rendering code.

Text

Description automatically generated

Then we’ll neaten up the initialization by breaking it out into a few functions.

Text

Description automatically generated

Now we’ll create a framebuffer to store data for the geometry pass.

Text

Description automatically generated

Now let’s make a vertex shader for the geometry pass

Text

Description automatically generated

The goal is not to do any lighting, but just to send positions and directions through. Now let’s take that in the fragment shader.

Text

Description automatically generated

Now we’ll create a geometry pass shader and set up its uniforms.

We can now complete the geometry pass

Text

Description automatically generated

Now let’s set things up for the lighting pass. The geometry pass wrote data to textures, now we need to get that on the screen. We’ll use a dummy quad to cover the screen, then render our output onto that.

Text

Description automatically generated

Now we can set up the vertex shader for the lighting pass

Graphical user interface, text

Description automatically generated

Super simple, gl\_Position gives us the position on the screen, and fragmentTexCoord lets us access the geometry data for that position. Let’s look at the fragment shader now.

We can now prepare the lighting pass

Text

Description automatically generated