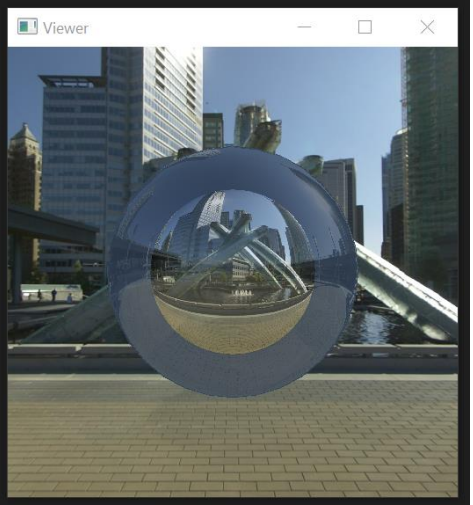
Faking Global Illumination on the GPU

Reflection & Refraction

```
#version 330 core

in vec3 normal;
in vec3 position;
in vec2 tc;

uniform samplerCube tex;

uniform vec4 colour;
uniform vec3 Eye;
uniform vec3 light;
uniform vec4 material;

void main() {
    vec3 tc;
    float Ro, R;
    float eta = 1.5;

    vec3 incident = normalize(position - Eye);
    vec3 reflection = reflect(incident, normal);
    vec3 refraction = refract(incident, normal, eta);

    Ro = pow((1.0 - eta)/(1.0 + eta), 2.0);
    R = Ro + (1 - Ro)*pow(1 - cos(eta), 5.0);

    gl_FragColor = texture(tex, refraction)*R + texture(tex, reflection)*(1 - R);
}
```

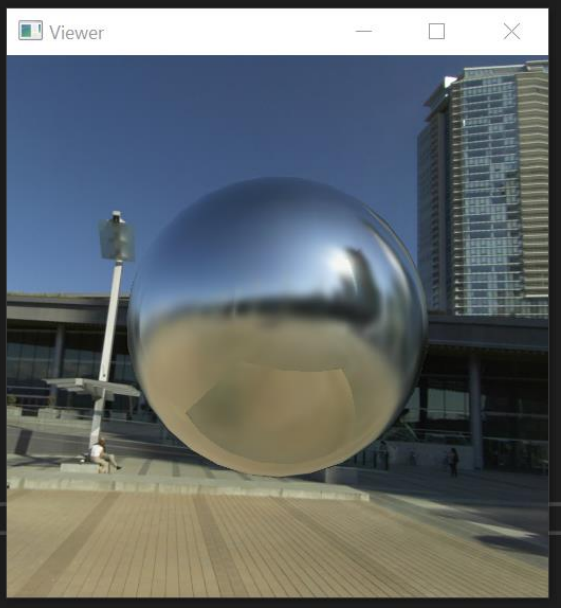


I applied Schlick's approximation to the fragment shader code (`example10.fs`) to calculate the approximate Fresnel reflection. First, find the reflectance at $\theta = 0$, where light is going straight at the surface. To do this, use the refractive indices of air, `1.0`, and glass, `1.5`. Then find the specular reflection coefficient, R.

Next, I apply the texture to the fragment shader. Use textures from the refraction vector and the reflection vector. This creates an inner sphere for refraction and an outer sphere for reflection. The sum of the two textures defines the colors of each pixel on each sphere. As a result, the outer sphere comes out more opaque than the inner sphere.

Diffuse Reflection – Part 1

```
#version 330 core

in vec3 normal;
in vec3 position;
in vec2 tc;

uniform samplerCube tex;

uniform vec4 colour;
uniform vec3 Eye;
uniform vec3 light;
uniform vec4 material;

void main() {
    vec3 tc = reflect(position - Eye, normal);
    gl_FragColor = texture(tex,tc);
}
```



I reverted all changes made in the fragment shader back to its original code. I created a new folder in the `VancouverConventionCentre` directory to store the blurred images. In the `viewer.cpp` code, I then create a new environment map, `blurMap`, and load it in the same way you load in the environment map `envMap`.

I also created a new texture, `blurred`, which would be the texture for the irradiance map. First, call `glGenTextures` and `glBindTextures` to create textures on the `blurMap`. Iterate through each image in the `VancouverConventionCentre/blurred` directory and assign each image to the `blurred` texture and set the texture parameters for the `blurMap`. The `blurMap` has now been loaded. Afterwards, in the `display` method bind irradiance map to the `blurMap` using `glBindTexture`.

Diffuse Reflection – Part 2

Unfortunately, this section is incomplete…