Raytracing

HIMANSHU RAJ (2022216), Indraprastha Institute of Information Technology Delhi, India

In this assignment, we have to modify a raytracer to generate nice images. We are given a basic raytracer named Lumina. Currently, it doesn't do much except render a sphere in a flat colour.

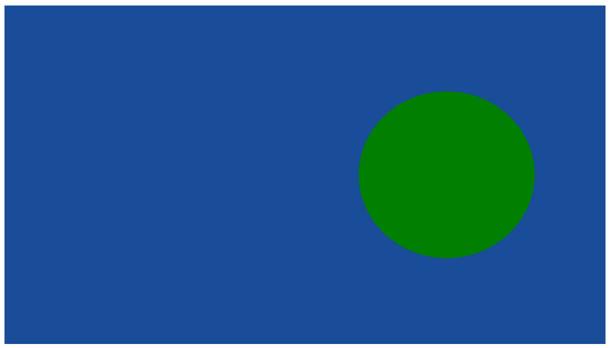

Fig. 1. Initial Output

1 RENDER A TRIANGLE

I implemented the Triangle class by extending the Object class to render in the raytracer. The triangle constructor takes 3 vectors as input for the 3 vertices of the triangle, and normal is also calculated after assigning vertices.
I have implemented the Ray-Triangle intersection. Given a ray p(t) = e + td and a triangle with vertices a, b and c, intersection will occur when

$$e + td = a + \beta(b - a) + \gamma(c - a)$$

This equation is written in matrix form and solved using Cramer's rule.
The intersection is inside the triangle iff $\beta > 0$, $\gamma > 0$, and $\beta + \gamma < 1$; otherwise, the ray has hit the plane outside the triangle.

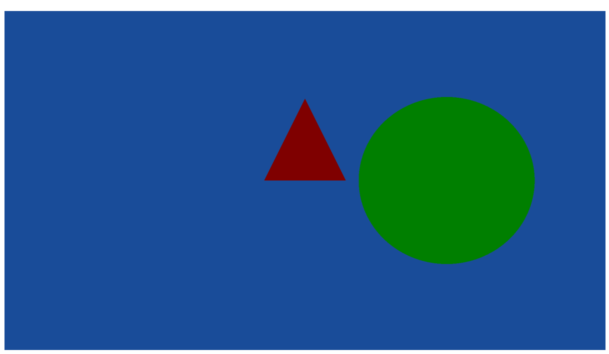

Fig. 2. Rendered a Triangle

Author's address: Himanshu Raj (2022216), himanshu22216@iiitd.ac.in, Indraprastha Institute of Information Technology Delhi, India.

2 IMPLEMENT BLINN-PHONG SHADING

I have implemented Blinn-Phong shading in the shade function of the Material class. I get the normal and view vector from the incident ray. I calculate the light direction unit vector and halfway vector for each light source to find the diffuse and specular effects caused by all light sources. I also calculate the ambient component and return the final colour as the sum of all the light components calculated.
The lighting equation is $I = I_{ambient} + I_{diffuse} + I_{specular}$

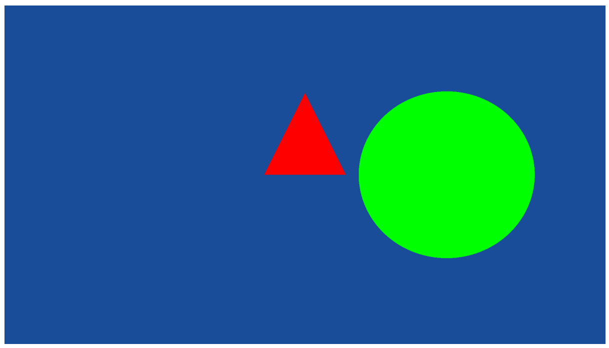

Fig. 3. Implemented Blinn-Phong shading for shapes

3 IMPLEMENT SHADOWS

The surface is illuminated if nothing blocks its view of light. I shoot out a shadow ray from the surface hit point to each light source. If the shadow ray intersects any object before reaching the light source, this means that this point will be a shadow, and only an ambient component is present. If the shadow ray doesn't hit any object, I consider all lighting components, i.e. ambient, diffuse and specular.
To cast the shadows, we need a surface or a plane to cast upon. I have used the sphere to cast the shadows of the triangle corresponding to 2 light sources.

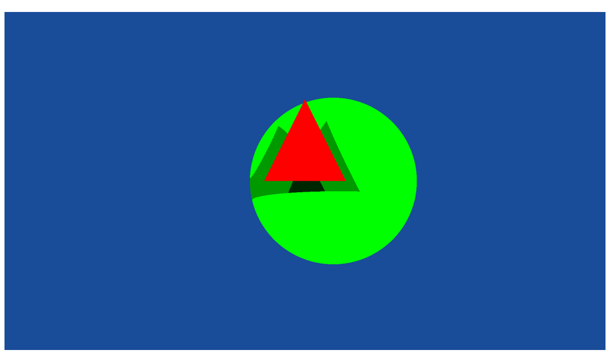

Fig. 4. Added Shadows for shapes and objects

4 IMPLEMENT REFLECTION AND REFRACTION

4.1 Reflection

Reflection in ray tracing simulates how light bounces off surfaces, which adds realism to objects with mirror-like properties. The reflection direction, calculated using the incident light direction $\mathbf{d}$ and the surface normal $\mathbf{n}$, is given by:

$$\mathbf{r} = \mathbf{d} - 2(\mathbf{d} \cdot \mathbf{n})\mathbf{n}$$

This formula determines the direction of the reflected ray by "mirroring" the incident direction across the surface normal. For materials with a reflection coefficient k_r, the reflected colour is computed by tracing this reflection ray further into the scene and scaling its resulting colour by k_r. By allowing recursive tracing, reflections can interact with other objects in the scene, yielding multi-layered reflections.

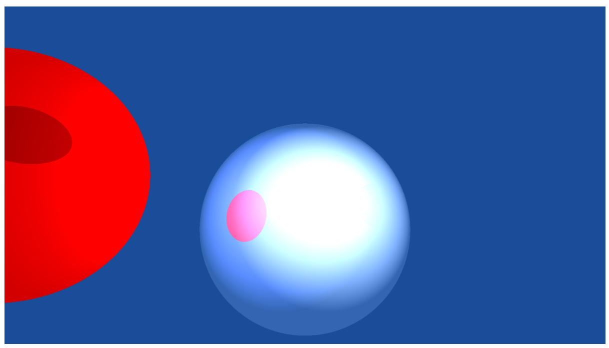

Fig. 5. Reflection

4.2 Refraction

Refraction simulates the bending of light as it passes through transparent materials, governed by Snell's Law. Snell's Law states that the ratio of the sines of the incident angle θ_i and the refracted angle θ_t is equal to the inverse ratio of the refractive indices:

$$\eta_i \sin\theta_i = \eta_t \sin\theta_t$$

where η_i is the refractive index of the incident medium (typically 1.0 for air) and η_t is the refractive index of the material. Given the incident direction $\mathbf{d}$ and surface normal $\mathbf{n}$, we calculate the cosine of the incident angle as:

$$\cos\theta_i = -\mathbf{d} \cdot \mathbf{n}$$

If $\cos\theta_i$ is negative, we reverse the normal direction and swap the refractive indices for correct internal refraction.

For materials with transparency ($k_t > 0$), if the light experiences total internal reflection, we calculate a reflected direction as in the reflection model. If total internal reflection does not occur, the refraction direction $\mathbf{t}$ can be derived as:

$$\mathbf{t} = \frac{\eta_i}{\eta_t}\mathbf{d} + \left(\frac{\eta_i}{\eta_t}\cos\theta_i - \sqrt{1 - \sin^2\theta_t}\right)\mathbf{n}$$

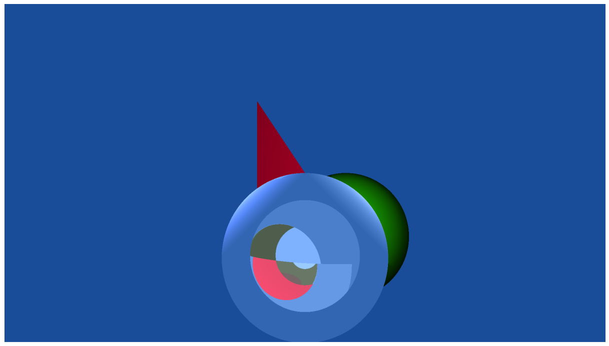

Fig. 6. Refraction