CG Assignment 3

Anshul Mendiratta (2018219)

November 5, 2020

Abstract

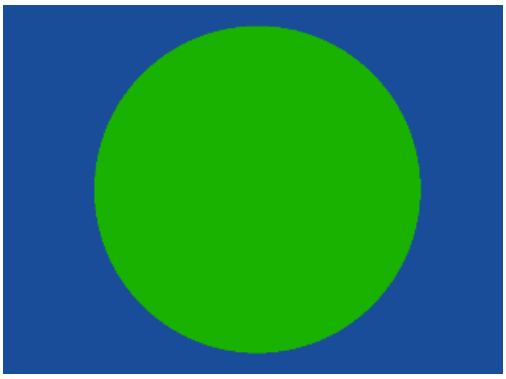
Make a simple raytracer to generate images with implementation of shading, shadows and different types of materials.

1 Introduction

We have initially been given a sphere with a flatly shaded color. We want to do the following

1. Implement a triangle
2. Implement Blinn Phong Shading
3. Implement shadows
4. Implement reflective and dielectric materials
5. Implement Schlick's Approximation and Beer's Law
6. Implement Jittered Supersampling to improve the quality of the render

The initial sphere looks like this -



(1)

2 Implementing Triangle

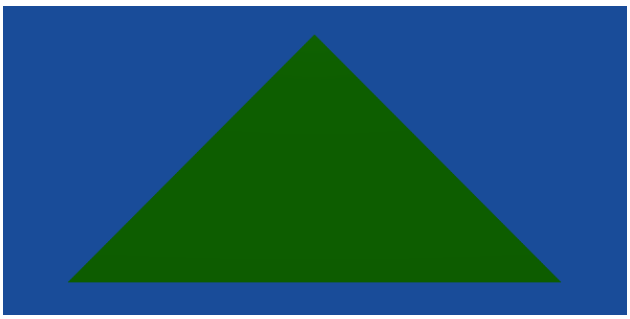
To construct a triangle we take the vertices we take its 3 vertices a, b and c as input. After we have the vertices we want to be detect when a ray from our camera hits the triangle and shade that region according to the light sources present in the scene.

We do the following -

1. Suppose we are given a ray R, e + td.

2. Trianle can represented as a parametric surface and hence using Barycentrc coordinates, intersection of ray occures when,
   e + td = a + β(b - a) + γ(c - a).
3. Now we solve for t, β and γ by writing above equation in matrix form and then using Cramer's rule.
4. Intersection is inside the triangle only if $\beta > 0$, $\gamma > 0$ and $\beta + \gamma < 1$
5. Using the t hence obtained and plugging it in e + td, we get the desired point on the triangle.
6. We will see in the next section how we shade our triangle.

Hence, out triangle is -



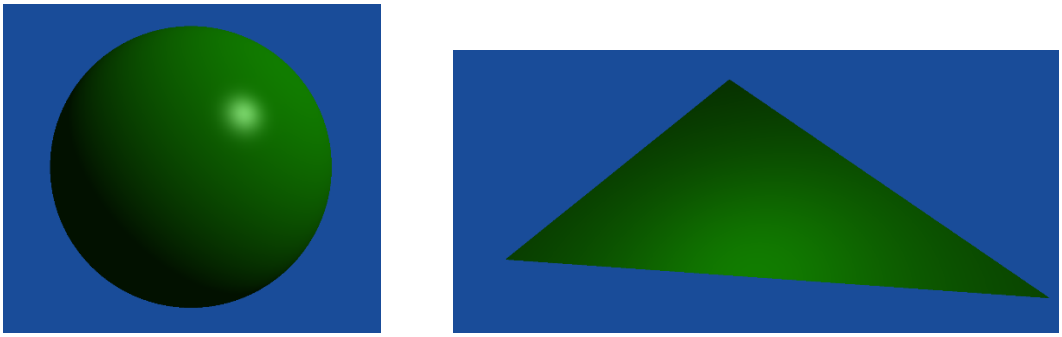
(2)

3 Blinn Phong Shading

To perform Blinn Phong Shading we need to find diffuse, ambient and specular shading and add them up. We find them in the following way -

1. For each ray coming out from the camera we check if a ray intersects with any object in the scene.
2. If a ray does not intersect we shade the pixel corresponding to that ray with the color assigned to the background.
3. However, if the ray does intersect, we compute the normal at that point and accordingly find the diffuse, ambient and specular shading for each light source in the scene.
4. We find the sum of the value obtained from each light source and assign it to the associated pixel (We have seen how to find these values in previous assignments).
5. For a sphere, normal is the normalized vetor from its centre to the point of intersection with the ray.
6. For a triangle, the normal is simply find the cross product of any of its two sides
7. Hence we have our desrired output

The resulting sphere and triangle are -



(a) Sphere (b) Triangle

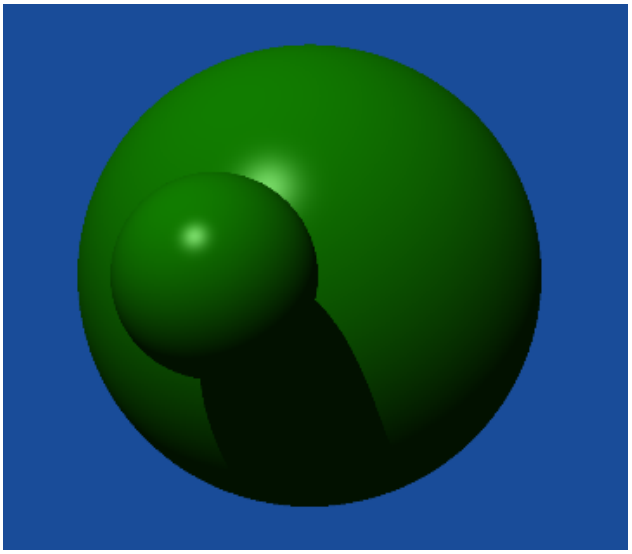
Figure 1: Blinn Phong Shading using Ray Tracing

4 Implementing Shadows

To find shadows we'll atleast need two surfaces. One surace is the object casting the shadow and the other is the object on which the shadow is being casted. To find shadows, we do the following -

1. For each ray coming out of the camera we check if it hits the object or not.
2. If it does hit the object, O_1, we construct another set of rays from the point of intersection to all the light sources present in the scene.
3. If this shadow ray hits another object O_2, then O_2 lies between the light source and O_1
4. Hence, O_2 will cast a shadow on O_1
5. I have assigned black color for shadows, but any color can be chosen.

Here, the smaller sphere casts shadow on the larger sphere -



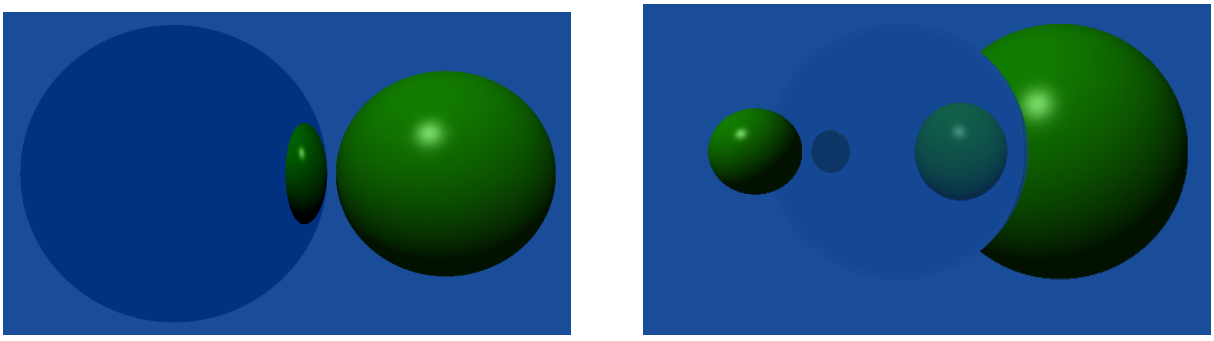
(3)

5 Implementing Reflective and Dielectric Materials

Reflective materials are those that will reflect all incoming light while Dielectric Materials will reflect some amount of light and refract some amount of light depending upon the refraction index (or eta).
We perform the following steps to implement reflective and Dielectric Materials

1. We must first be able to calculate the reflected and refracted ray for each ray that hits the surface.
2. To find the reflected ray, we use the formula, r = 2(l.n)n - l, where r is the reflected vector, light is the initial ray and n is the normal.
3. To find the refracted ray, we use the formula, t = eta(d - n(d.n)) - n $\sqrt{1 - eta^2(1 - (d.n)^2)}$
4. Now we recursively call the function to shade the pixel on the reflected and refracted ray in which they were being calculated.
5. Hence we will have a tree of rays where one branch is the reflected ray and the other ray is the refracted ray.
6. When a ray doesn't hit any object, hits a diffuse object or reaches maximum depth it stops recursing.
7. At each level the resulting color is the sum of the colors obtained from the reflected and refracted rays multiplied by coefficients deciding how much light is to be reflected and how much is to be refracted.

Resulting reflecting and dielectric materials are -



(a) Reflective Material (b) Dielectric Material

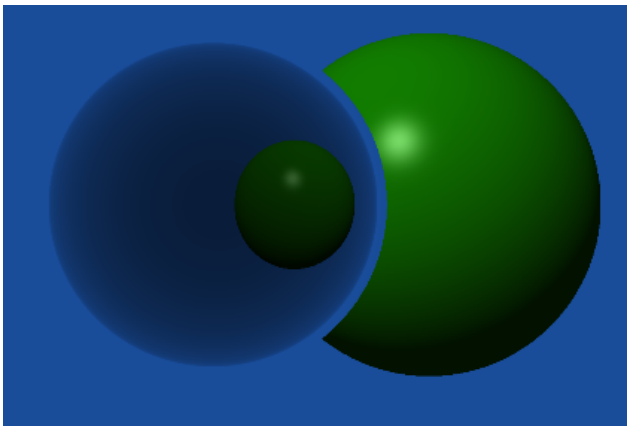
Figure 2: Blinn Phong Shading using Ray Tracing

6 Schlick's Approximation and Beer's Law

Schlick's Approximation helps us take into consideration the variation of reflection with angle of incidence while Beer's Law helps us in showing attenuation of light while passing through a material. We perform the following steps to implement Schlick's Approximation and Beer's Law

1. To apply Schlick's Approximation, we use the formula, $R_o = \frac{(eta-1)^2}{(eta+1)^2}$ and R = R_o + (1 - R_o) $(1 - cos\theta)^5$, then we multiply R with reflected light and 1 - R with transmitted light.
2. To apply Beer's Law, we use the formula, $k_i = exp(-a_i t)$, where t is the distance travelled, i is either red, green or blue and a_i is attenuation coefficient corresponding to i.

We get the following resluts -



(4)

7 Supersampling

Supersampling is used to improve the quality at expense of runtime. We use jittering supersampling to achieve this we take n points corresponding to each pixel and average out those values to get the corresponding value.

Resulting render is -

(a) Without Supersampling

(b) With Supersampling