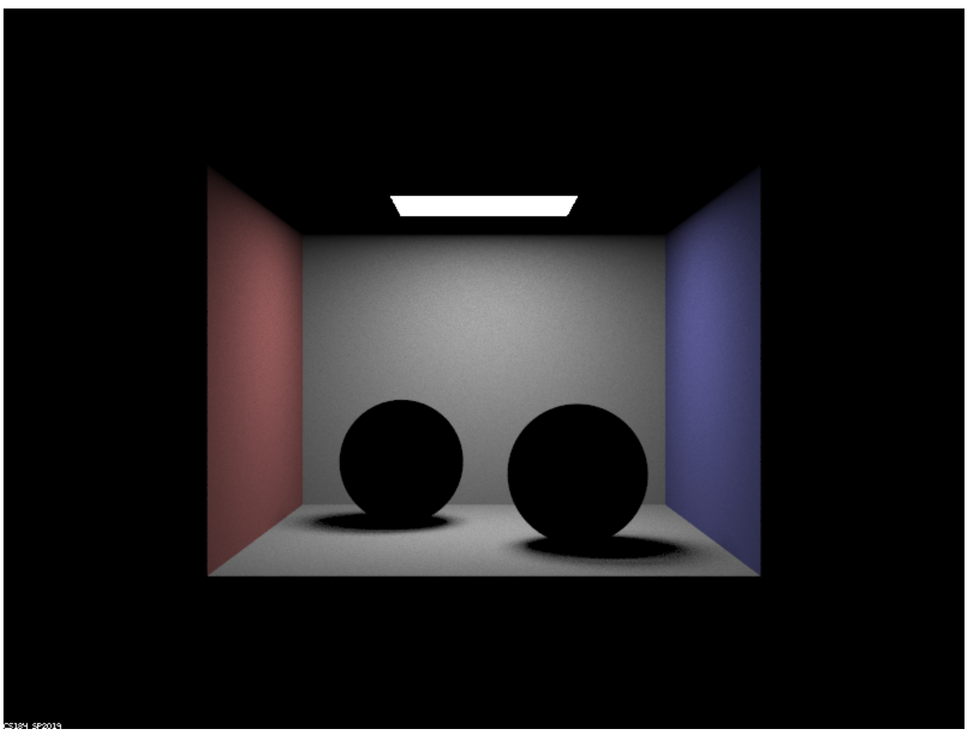
Assignment 3: PathTracer

Khadijah Flowers

Part 1: Ray Generation and Intersection

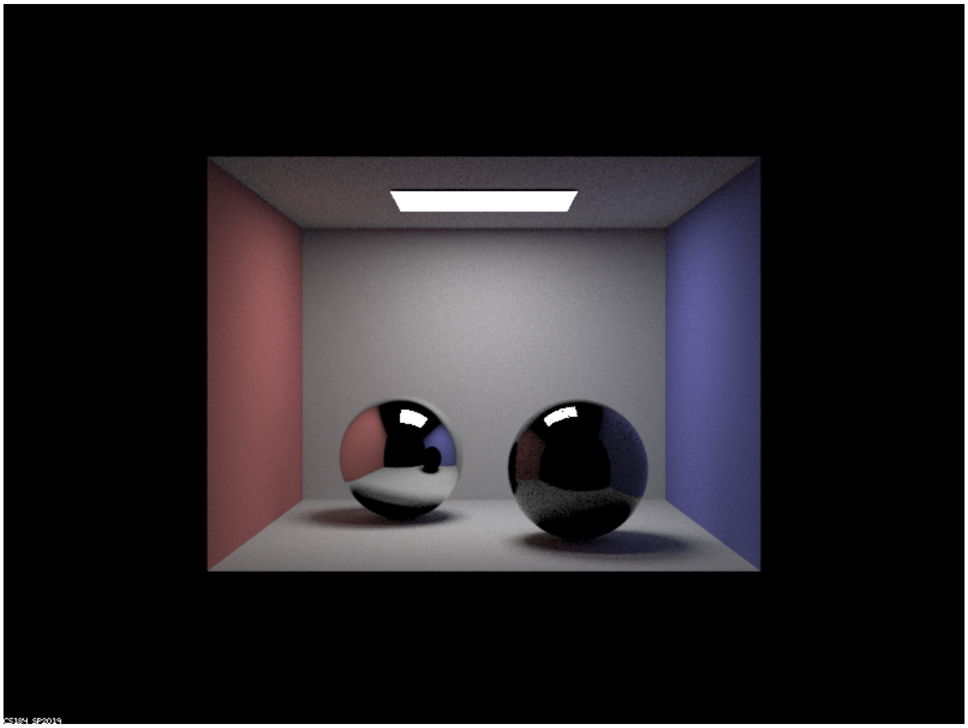
In Part 1 of the project, I implemented Ray Generation and Intersection of a light with mirror and glass objects in a room. I implemented a reflection function that took incoming rays and determined the outgoing reflection ray. From this ray, we can determine what should be reflected off of this part of the object. I also implemented a refraction function that took in a ray and computed the refraction ray on the surface of the glass object.

As we increase the number of bounces we find that the number of visible objects in the scene increase and the amount light increases as well, making the room brighter and brighter.



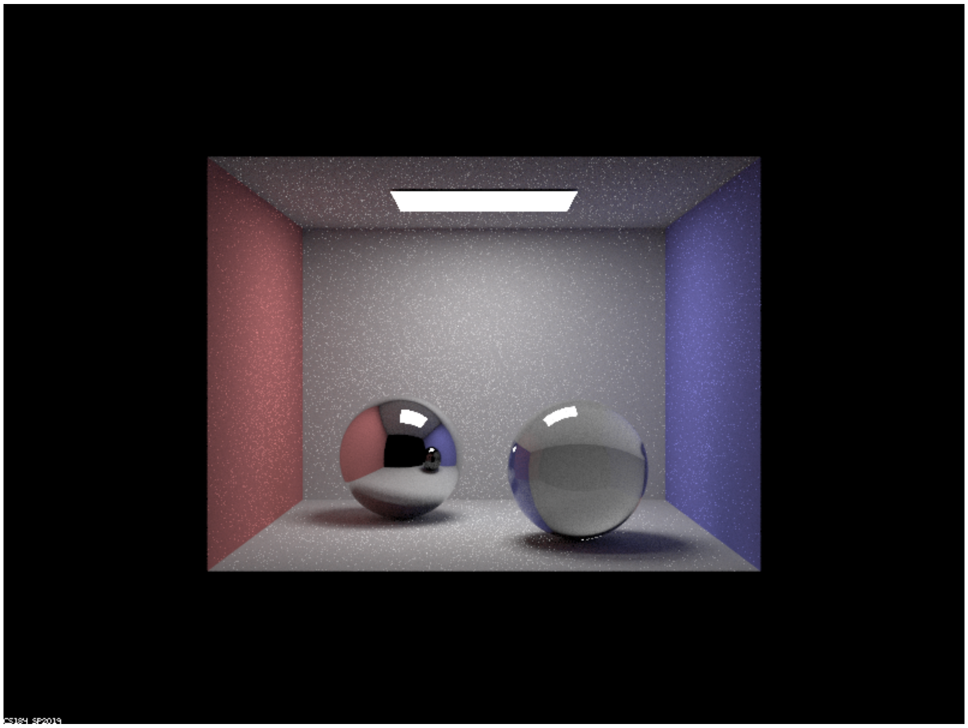
Zero Bounce

In the Zero Bounce image, the roof, balls, and shadows of the balls are completely black because we have not case a ray from these place to any of the parts of the scene that emit light.



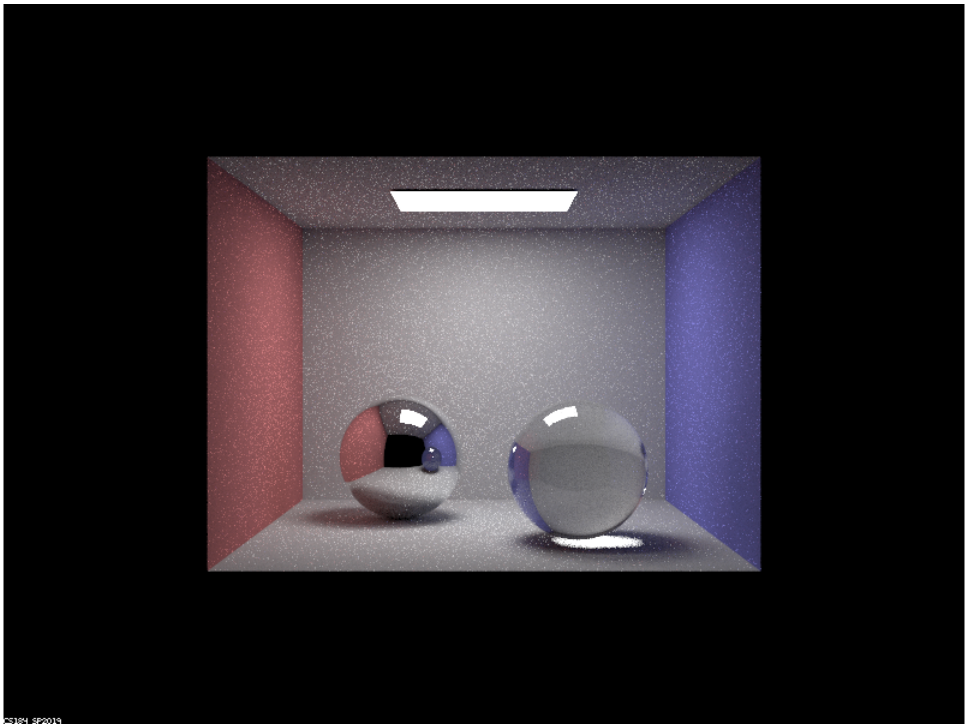
One Bounce

Now, we take into account all light that can reach the previously dark objects by one ray of light.



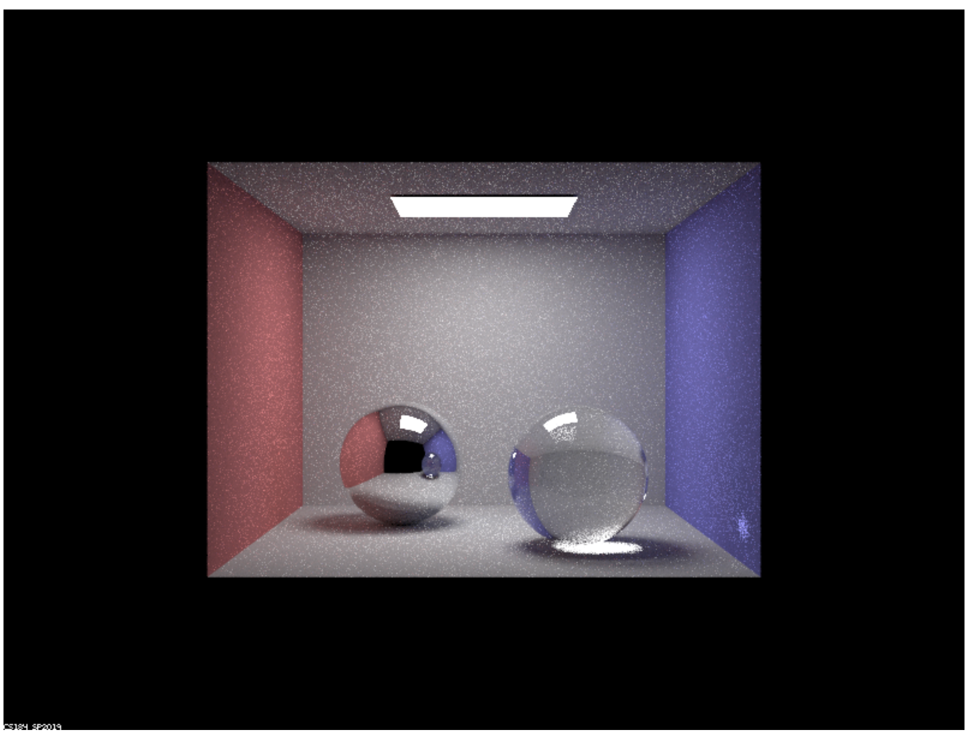
Two Bounce

Now we can see the glass ball start to refract because we are sending a refracted ray through the object. We can also see the glass ball being reflected in the mirror ball. If you look closely, in this reflection, the mirror ball is reflecting the red wall and the ceiling. The shadows of both balls have also started to get brighter along the edges.



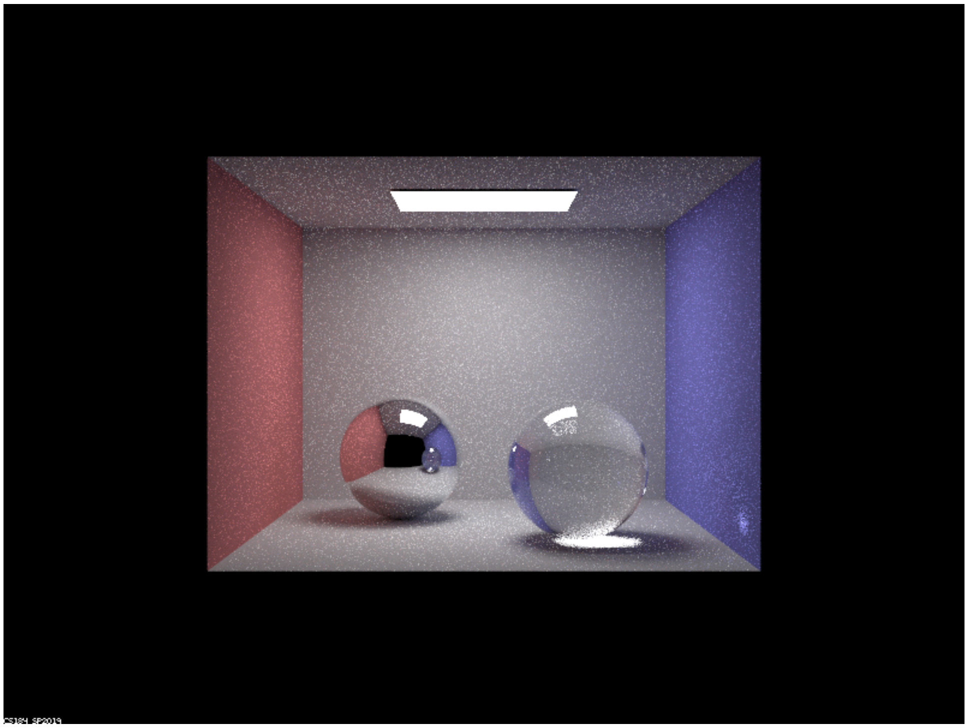
Three Bounce

Now, as we increase the number of bounces, we start to see the shadow of the glass ball behave stragely. It now has a bright circle in the middle because the light rays in the ball have refracted outside of it and hit the floor below it. Also, the other walls are starting to gain some noise.



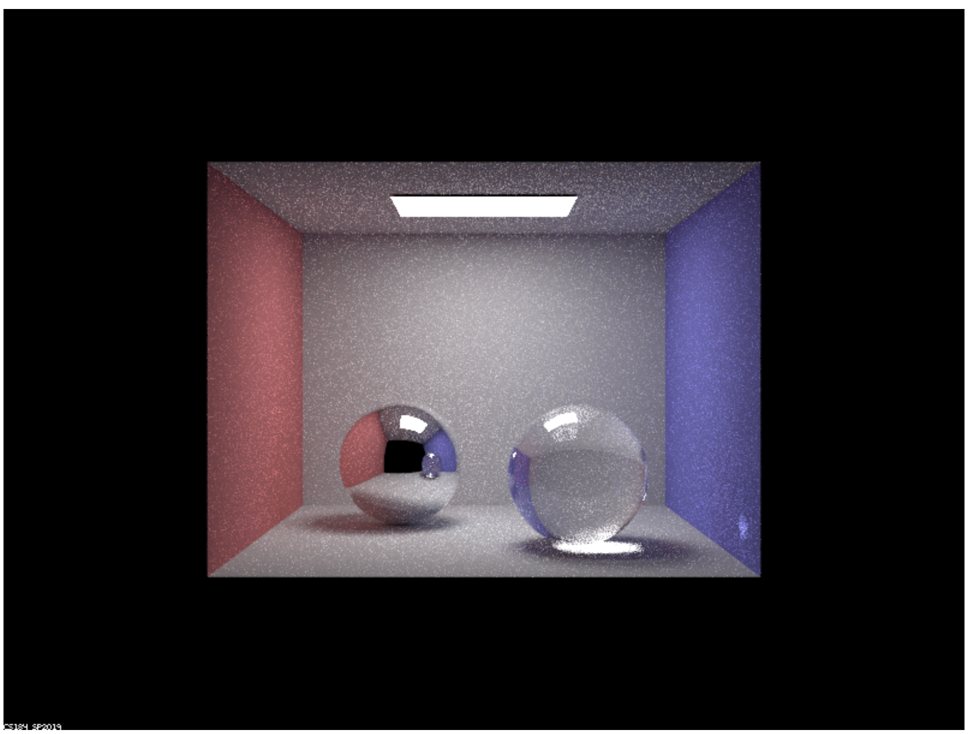
Four Bounce

The light that refracted through the mirror ball in the last bounce is now back to the ball, making that spot brighter than before. A spot on the blue wall is now very bright as rays from the floor are being shot out onto it. The mirror ball is also reflecting the mirror ball a bit better.



Five Bounce

In this image, the noise in the scene is increased more from the last image because we have several bounces from many objects all over the scene (balls, wall, floor, etc.) shining rays of light. The back wall is also getting brighter because of the amount of light rays being shot at it.



One Hundred Bounce

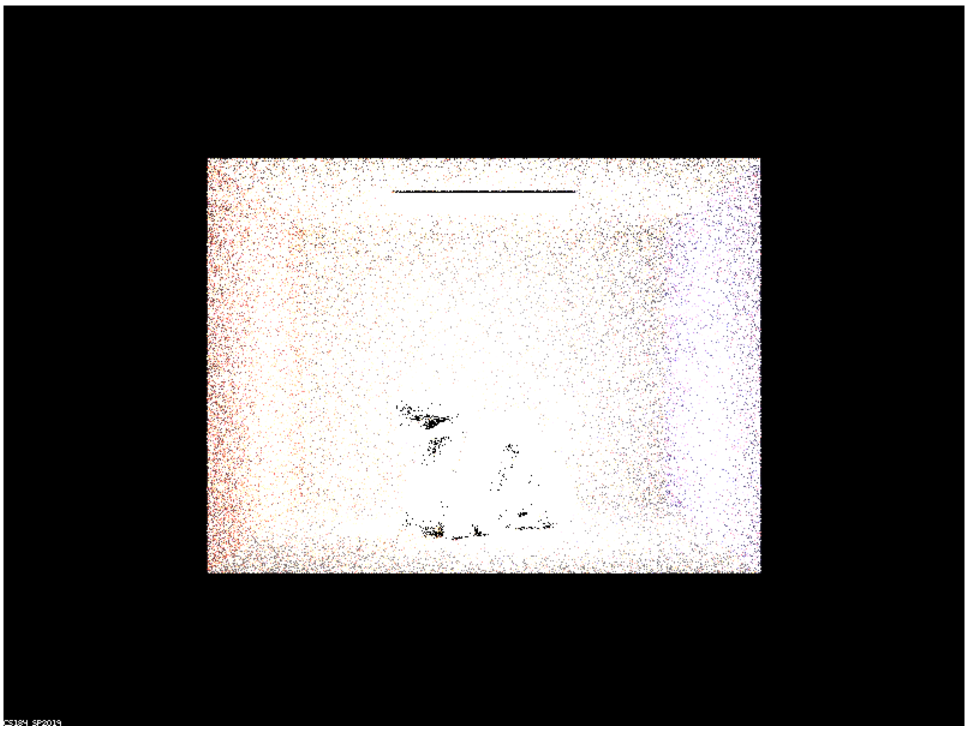
Finally, with one hundred ray bounces, we start to see the color of the nearby walls being shinned on the balls and the floors. For example, the left mirror ball has a lot of red in its shadow. Areas of both the red and blue wall seem to be brighter in a more consistent way.

Part 2: Microfacet Materials

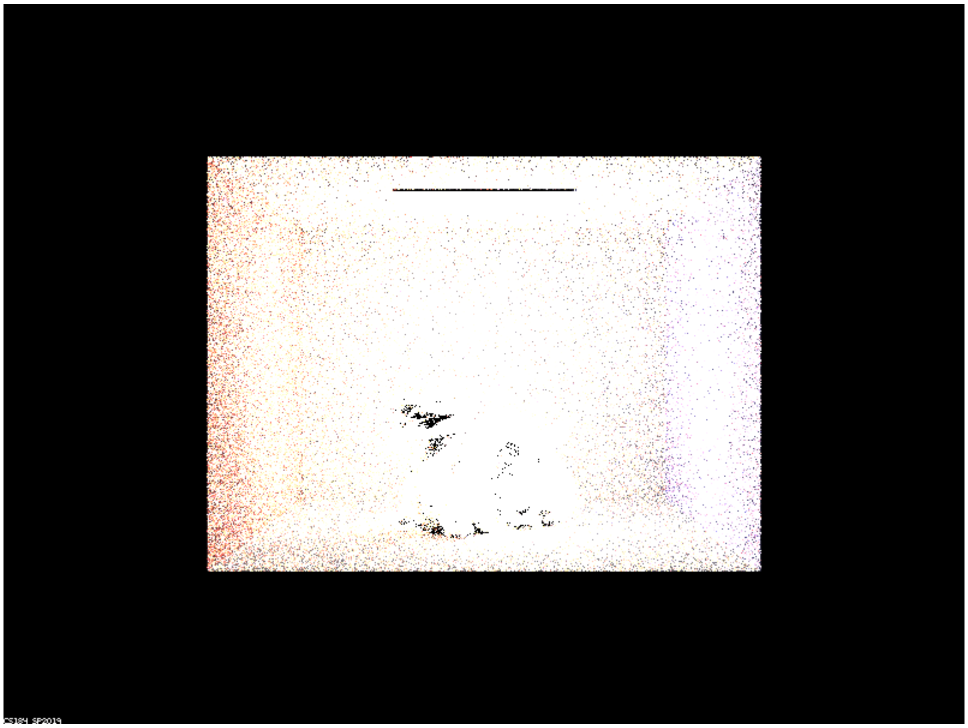
In Part 2, we computed the BRDF using the formula:

```
(F(wi) * G(wo, wi) * D(h))
/ (4 * dot(Vector3D(0, 0, 1), wo) * dot(Vector3D(0, 0, 1), wi))
```

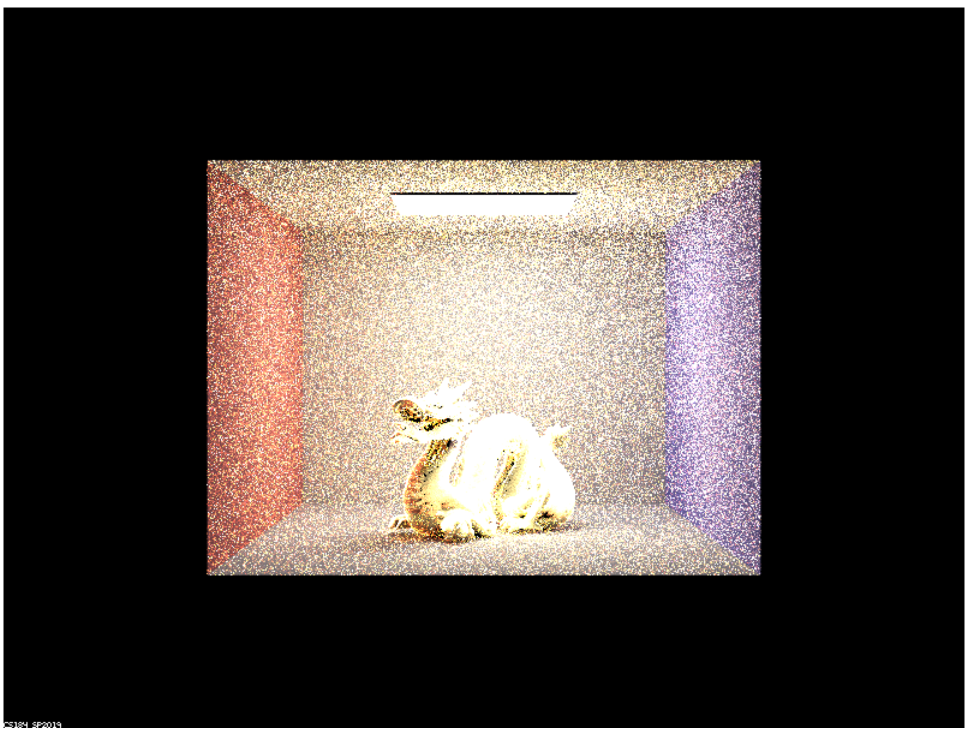
In this formula, we calculate the NDF (Normal Distribution Function), the definition of how the normal vectors along the objects are distributed and the Fresnel term to accurately determine the color of the object. Finally, we used importance sampling to sample the lights in the room and assign colors to the object.



alpha = 0.005

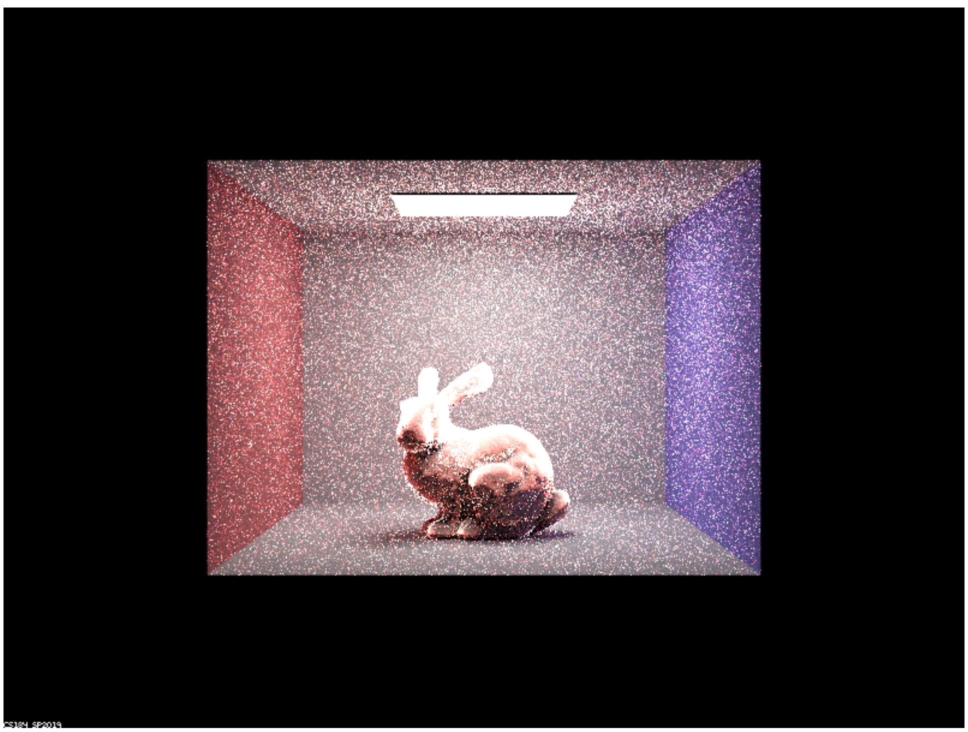


alpha = 0.05

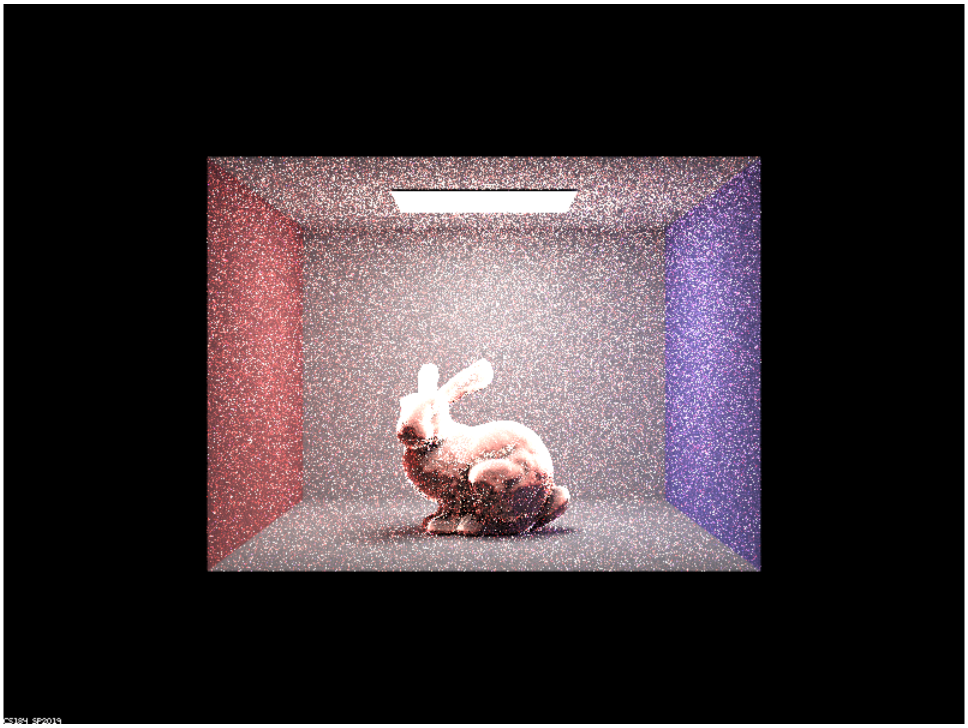


alpha = 0.25

alpha = 0.5

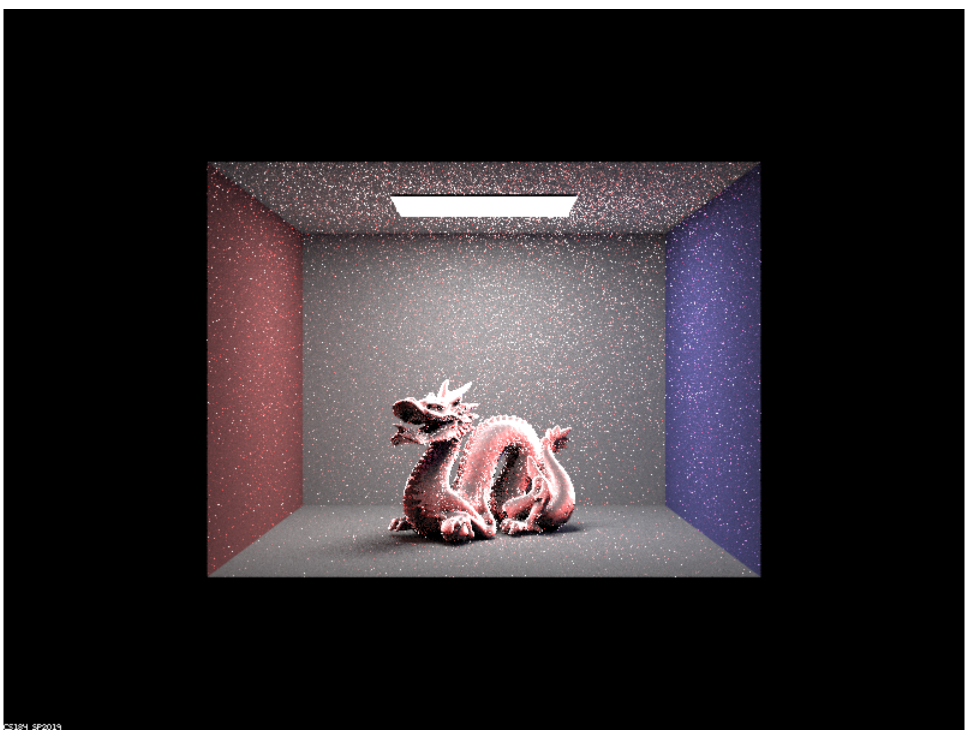


Hemisphere Bunny.



Importance Bunny.

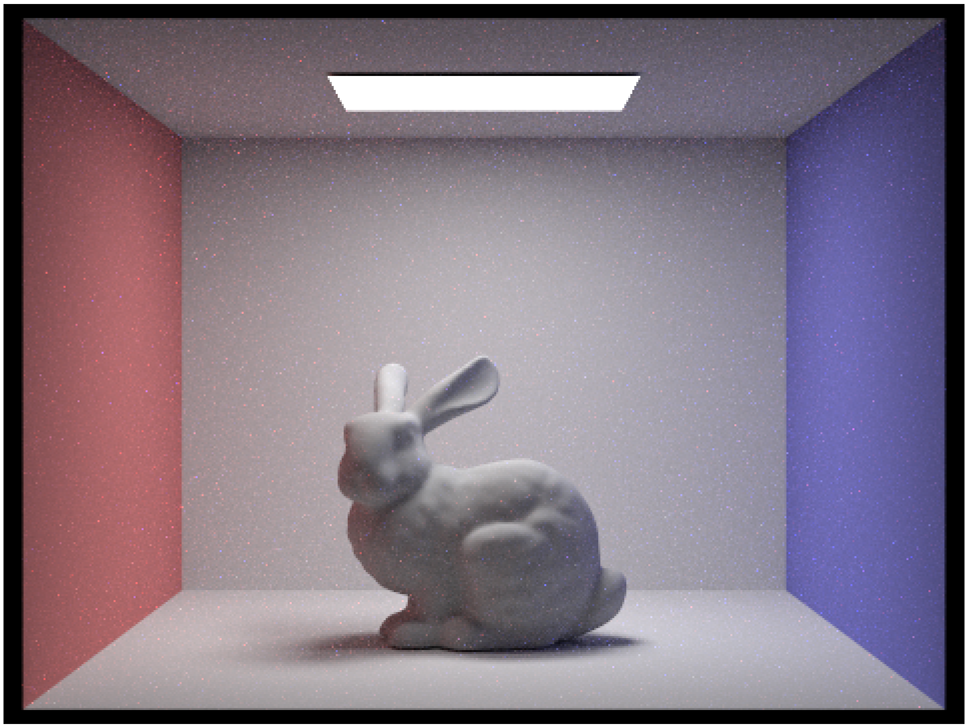
In the hemisphere sampling image, the colors in the scene are more uniform overall and more diffuse. In the importance sampling image, the bunny isn't as diffuse and looks as if it is made of a different, more realistic material.



Boron Dragon

Part 3: Environment Light

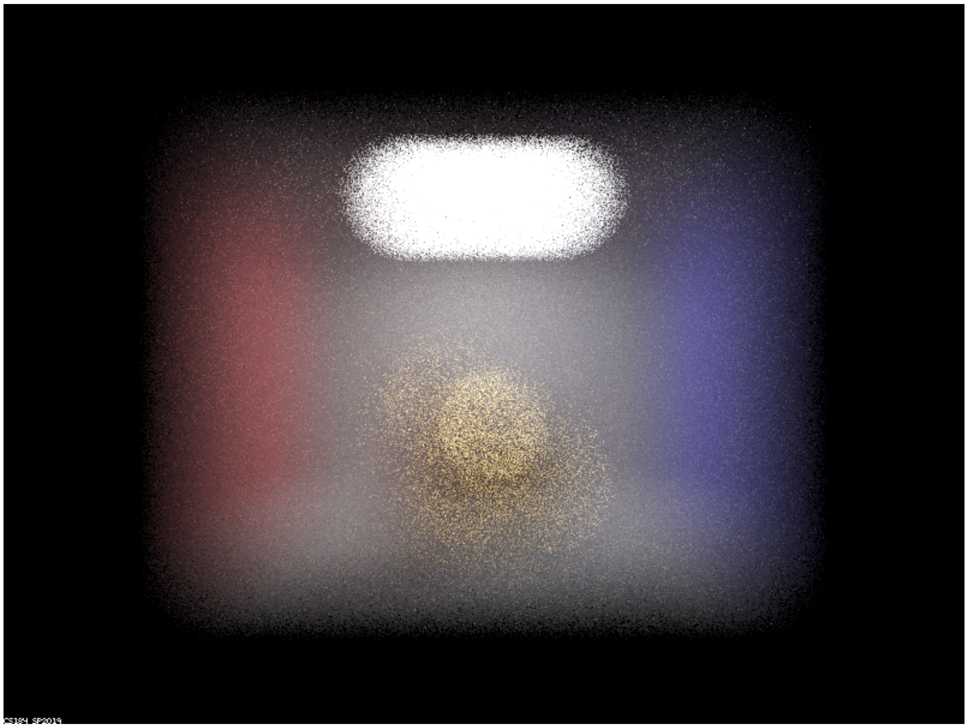
In Part 3, we introduce environment light, which is a light that is an infinite distance away, causing it to supply a larger amount of incident radiance to objects in the scene. Hardly any object is left without light and the distance of one object from the light source is no longer relevant as compared to other objects.



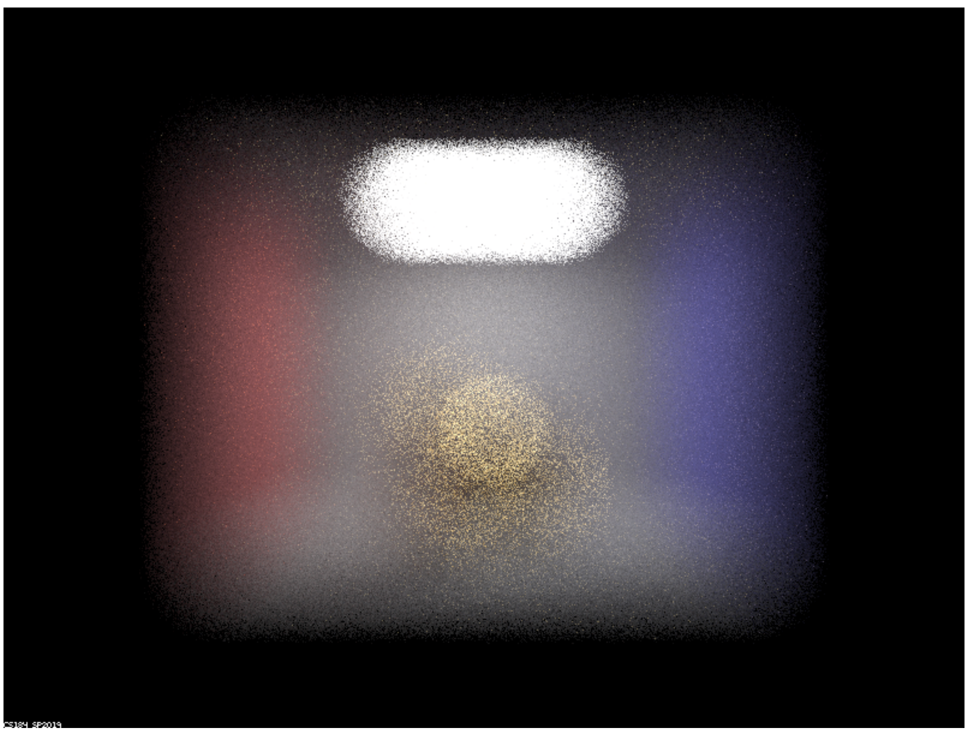
No images, but all code WAS written for this part.

Part 4: Depth of Field

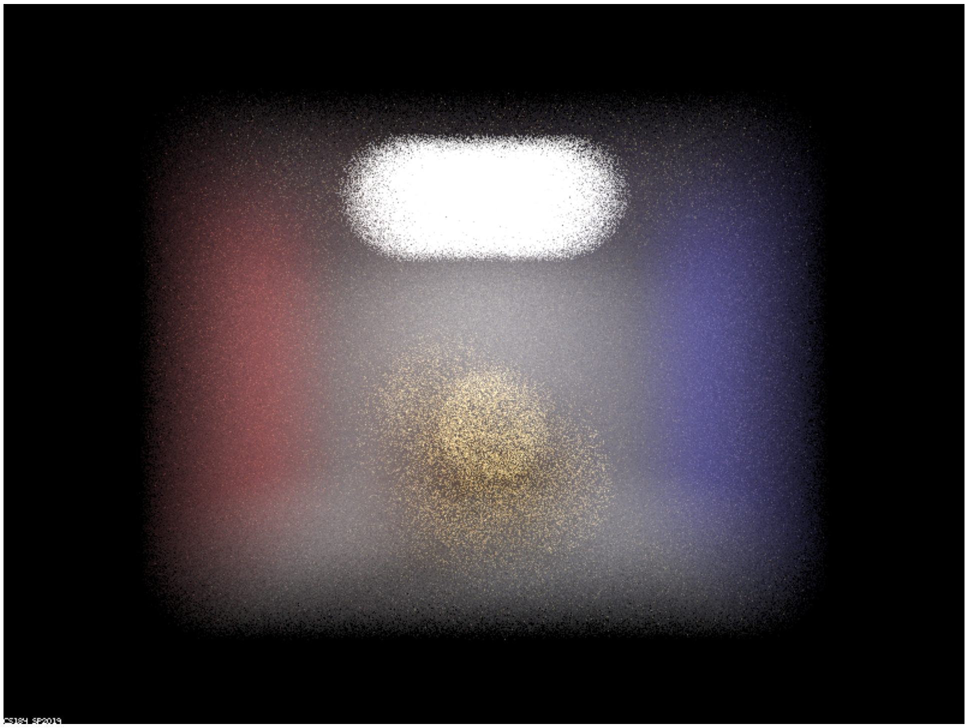
In Part 4, we create a thin lense to simulate depth of field where objects at a certain distance away from the lense are more visible than objects at other distances. For us, the distance is the focalDistance from the lense. In a Pinhole Camera, everything on the image plane appears upsidedown because we only care about what we can see through a hole in a center plane. A thin lense in more like taking the image and projecting it onto the image plane instead of tracing individual points through a single hole to the image plane.



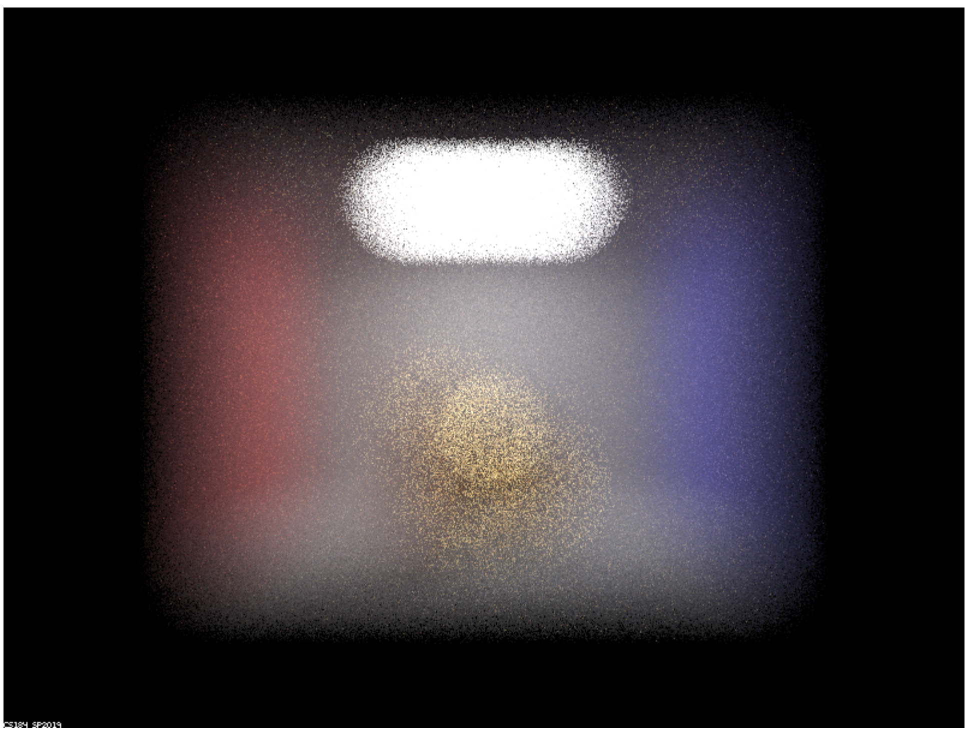
Depth = 1



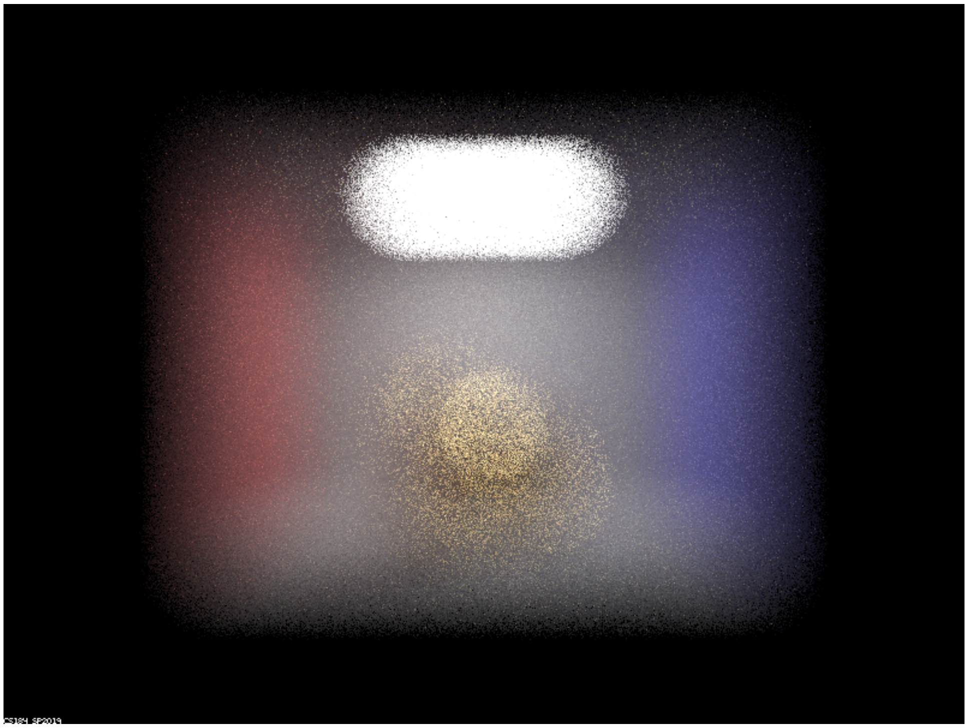
Depth = 2



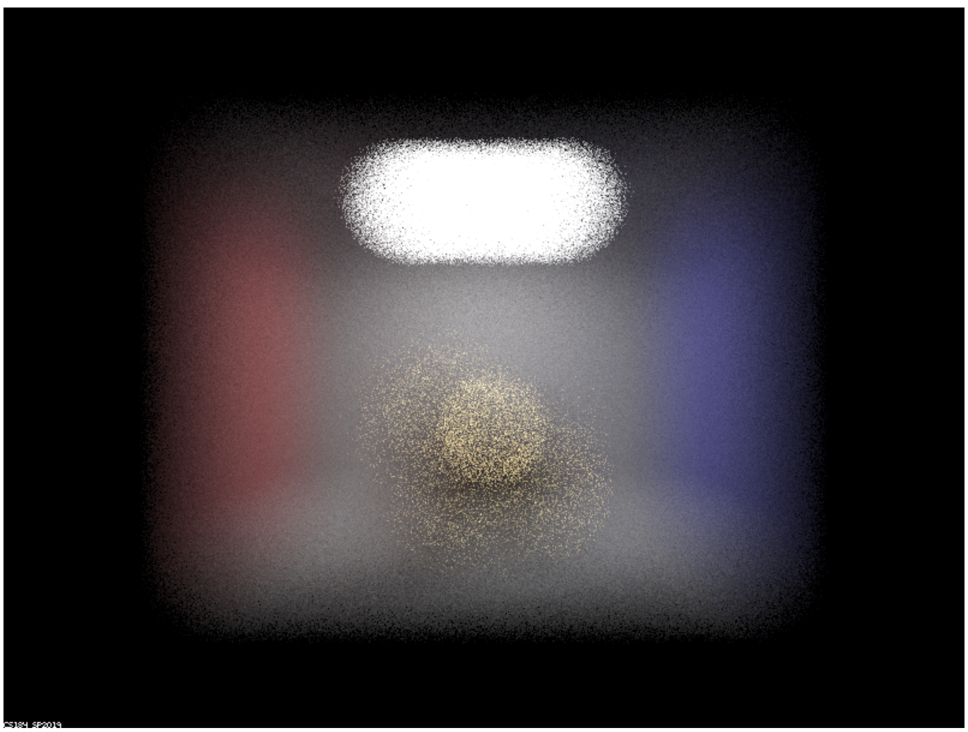
Depth = 3



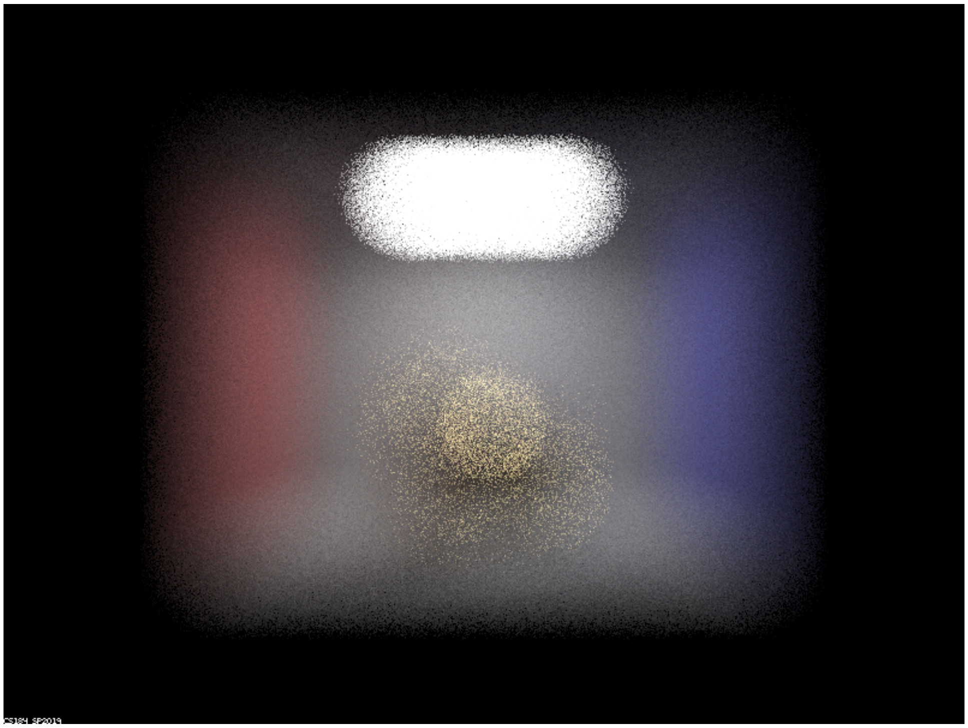
Depth = 4



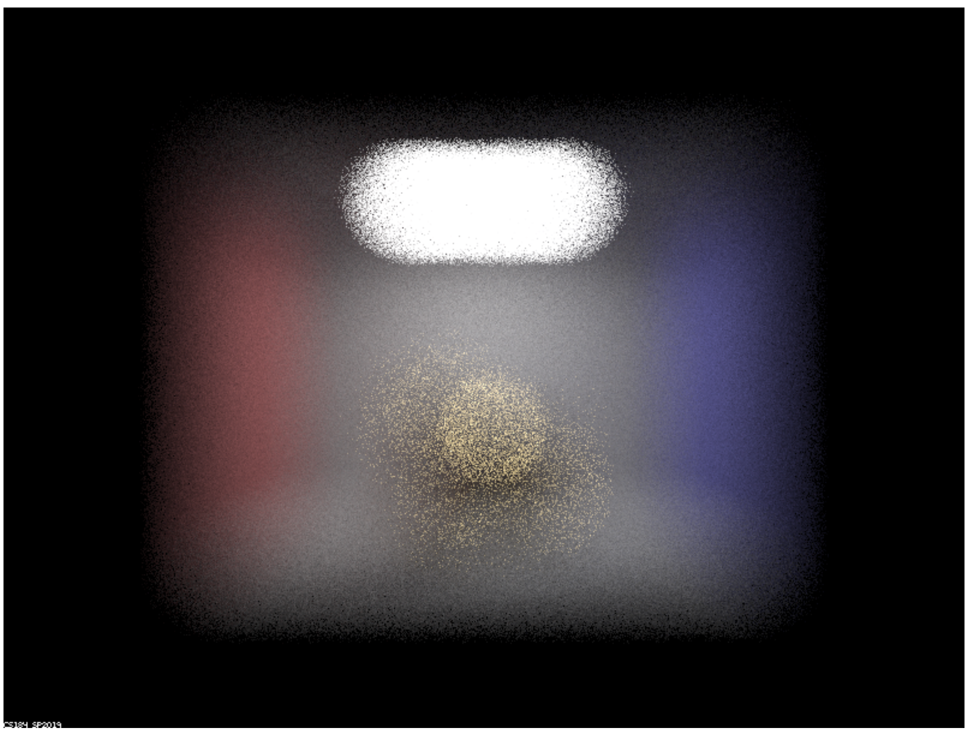
Aperture = 0



Aperture = 10



Aperture = 20



Aperture = 30