Assignment 3: PathTracer

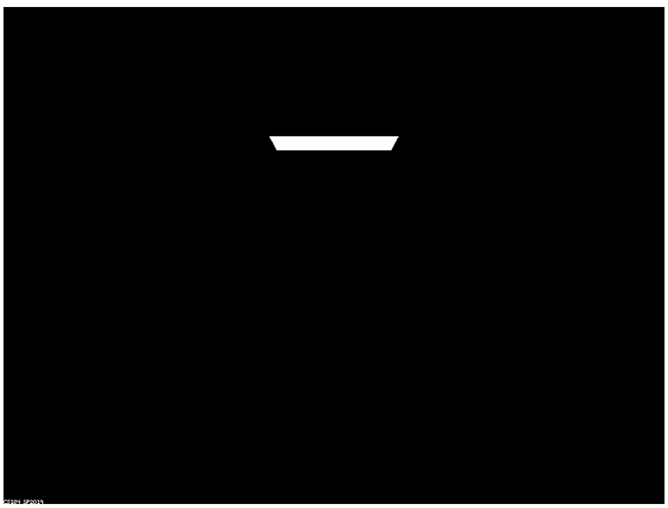
Raymond Ly

In this portion of Assignment 3: Pathtracing, we implemented functionality that would allow Glass and Mirror-like surfaces to behave as they do in the real world. Alongside that, we also simulated microfacet materials and the behavior light has with these microscopically rough surfaces through importance sampling using the Beckmann distrubution. In the "bigger picture" of graphics, so to speak, we created functionality for global illumination, as to emulate realistic lighting environments under an infinitely far light source emitting in all directions. Mechanically, we changed the way our rays and bounces behaved to refract through a lens, providing us the ability to render images with depth of field, i.e. a sense of dimensionality where only things at focal distance from our lens is in focus while objects closer or further are blurry.

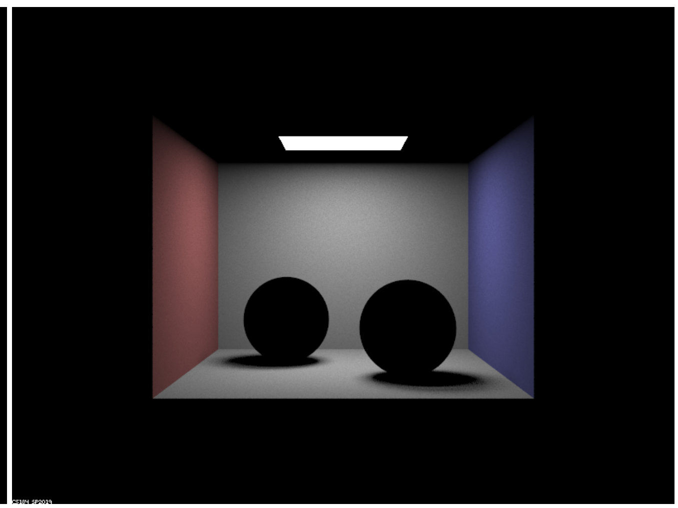
Part 1: Mirror and Glass Materials

To simulate mirror materials, we simply needed to reflect raycasts about our normal/viewing vector, since the assumption here is our material behaves accordingly to specular reflection. The same cannot be said of glass materials, as there are both reflective and refractive properties to glass. To achieve this behavior, we find the ray that results from light "bending" as it passing through space at the interface of two materials. Our model only allows us to generate and display a single ray from this interface, so we use Schlick's approximation and random chance to determine which ray we return. Below, you can see several renders of the same scene at increasing ray depth/max bounces.

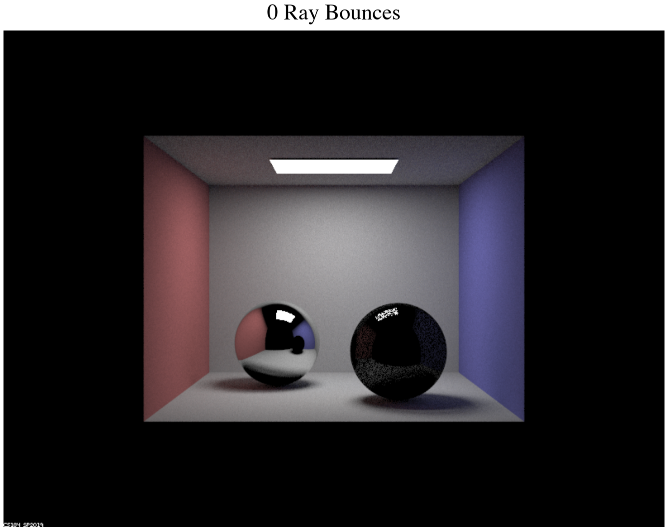
Spheres rendered at 64 samples/pixel and 4 samples/light



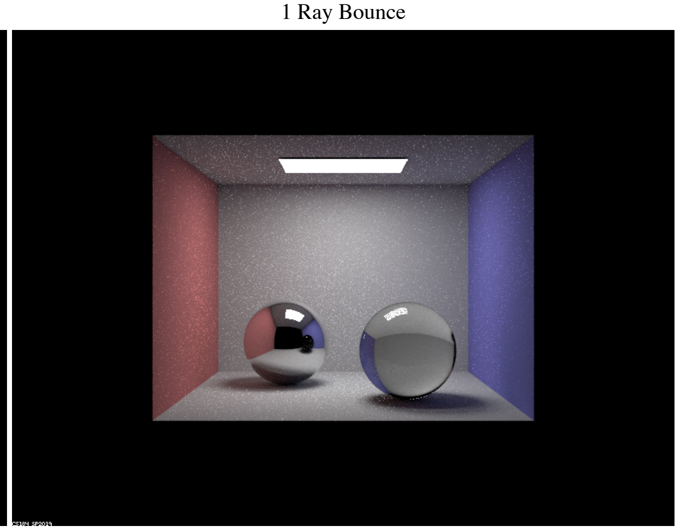
0 Ray Bounces



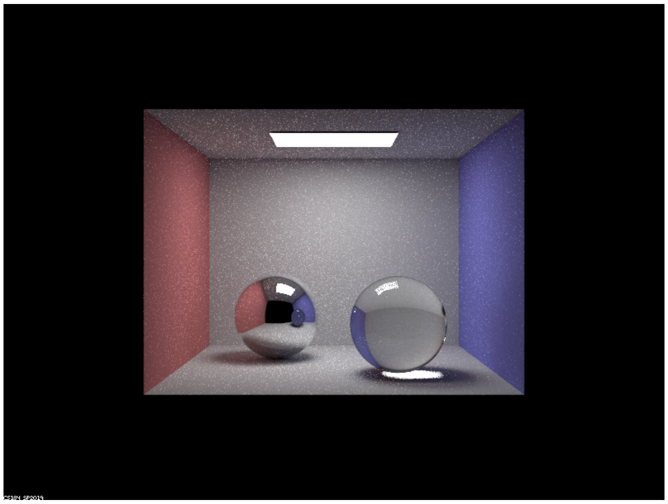
1 Ray Bounce

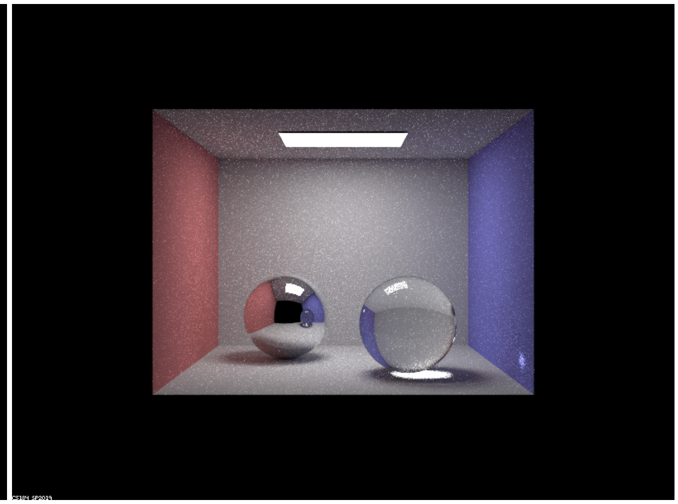


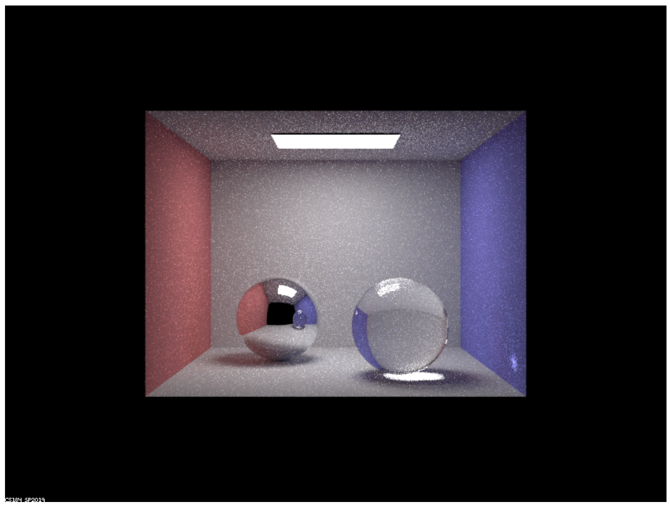
2 Ray Bounces



3 Ray Bounces


4 Ray Bounces


5 Ray Bounces

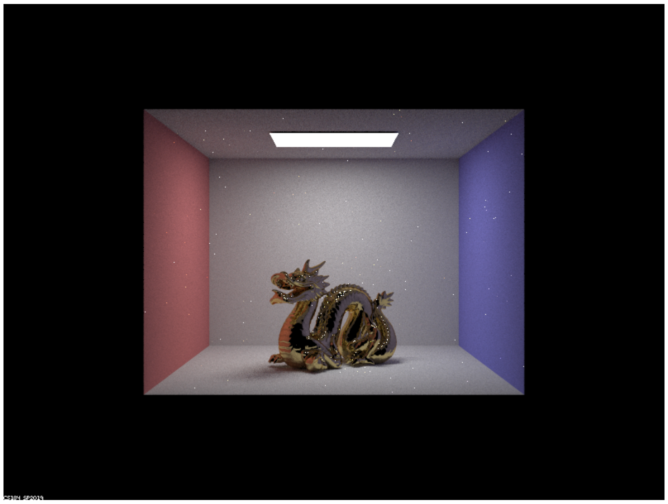

100 Ray Bounces

As you can see, as we increase the number of bounces, we begin to get artifacts, likely due to the fact that we my be reflecting the white light more often when we don't need to. Of course, we can see the stark difference between 1, 2, and 3 bounces. At 1 bounce, we only see uncolored spheres, since there is no further depth to how far we want to view our rays. At 2 bounces, notice how the sphere on the left still shows a black ceiling, so the ray depth is not yet great enough to reflect the space it is in. At 3 bounces, we can clearly see the whole room is reflected in the left sphere and the walls are passing though in the right. However, if you look closely, the rays through the ball on the right are still yet to be reflected by the left. This detail only comes about at a ray depth of 4 and onwards. From there, smaller details concerning lighting will come into view, such as the bright spot on the right wall through the right ball.

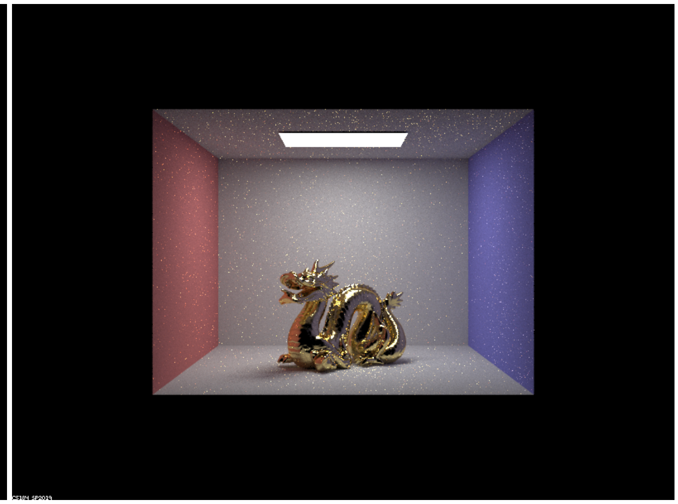
Part 2: Microfacet Material

Below, we can see 4 renders of the same scene at different alpha values. What we notice immediately is that the amount of noise and brightness of the scene in general all increase as alpha increases. This gives us an overall cleaner, smoother, and more detailed resulting image. Specifically, looking at alpha=.05 and alpha =.5, we can see that the dragon itself seems like a harsher reflective metal since the areas it reflects light are incredibly bright, but the areas in the shadows are exceptionally dark. Meanwhile, at a higher alpha, the dragon displays a "softer" simulation of the material, but successfully captures the shading as a gradient and details that are unrefined, such as the scales.

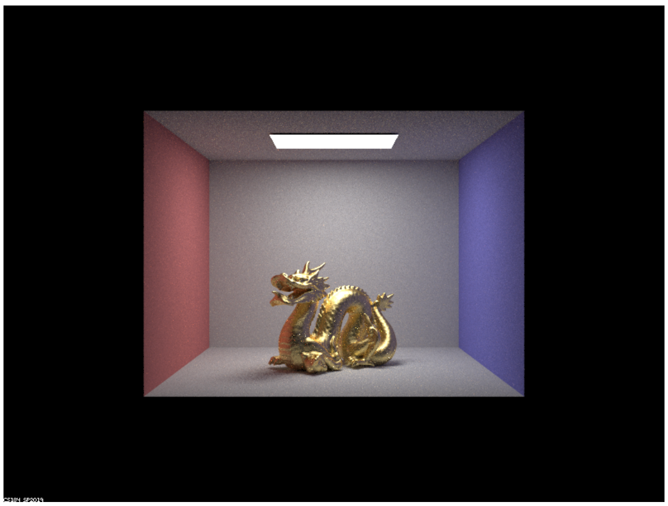
Gold Dragon rendered at 128 samples/pixel, 1 sample/light, ray depth of 5



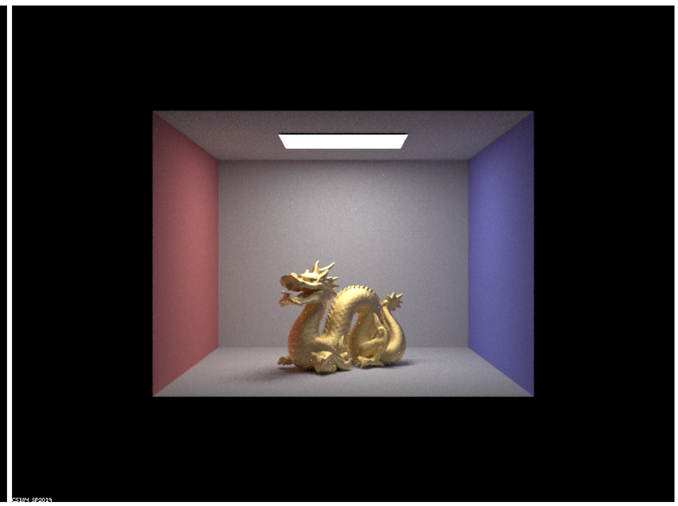
alpha = 0.005



alpha = 0.05



alpha = 0.25



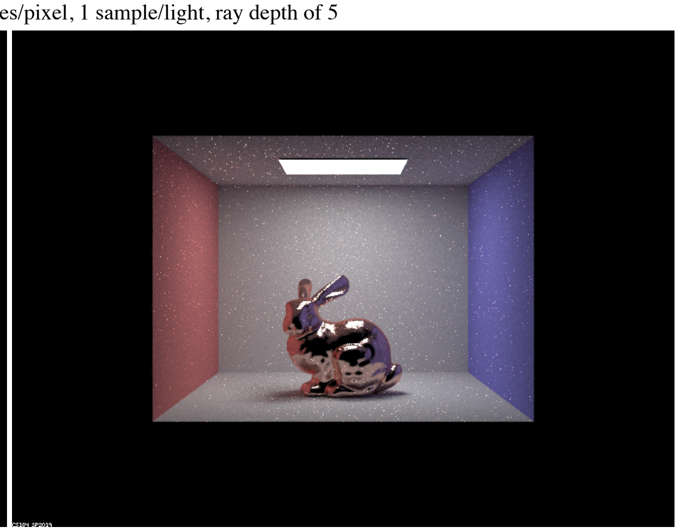
alpha = 0.5

By inspection, it is clear to see that at the same render settings, the importance sampled bunny has far greater fidelity and detail with slightly less noise. What we may notice is that the importance sampled bunny also displays an image that seems more "complete" than the cosine hemisphere sampled one, showing us it converges to the real image far faster.

Comparing Uniform Hemisphere and Importance Sampling

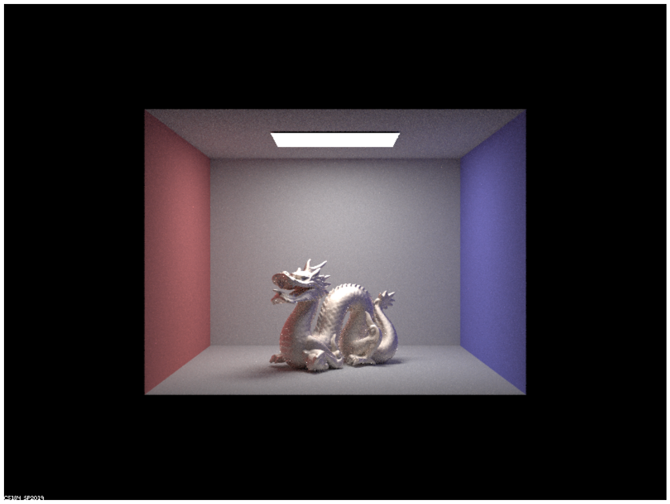
Copper Bunny rendered at 64 samples/pixel, 1 sample/light, ray depth of 5

Uniform Hemisphere Sampling



Importance Sampling

Tungsten Dragon

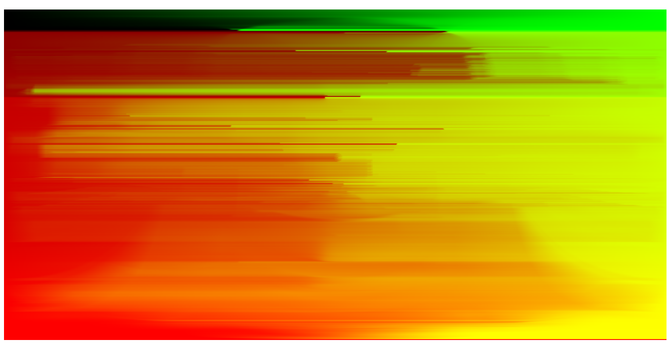


eta = 0.96709 1.3573 2.1893
k = 6.2738 5.2210 5.0244

Part 3: Environment Light

The concept of environment lighting we are implementing is that an "infinitely" distant light source illuminates the scene by casting light in every direction on its surface. Equivalently, we can consider the Sun illuminating our surroundings as the source of global illumination in our scenes.

Environment: Grace

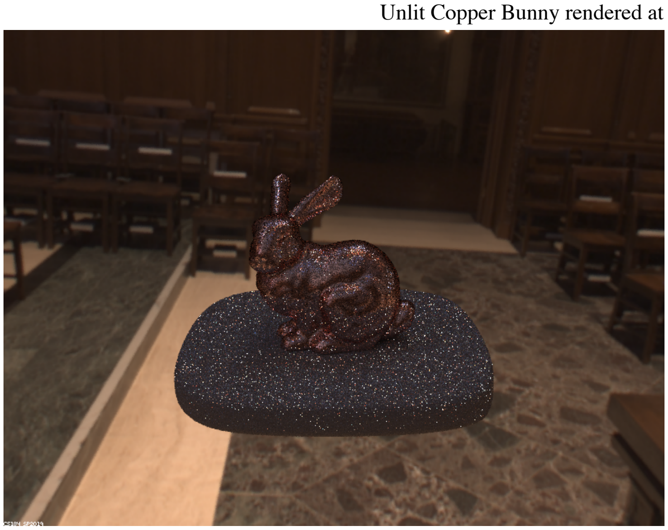


Probability Debug for grace.exr

Comparing Uniform Hemisphere and Importance Sampling on Environment
Unlit Bunny rendered at 4 samples/pixel, 64 sample/light


Uniform Hemisphere Sampling | Importance Sampling

Unlit Copper Bunny rendered at 4 samples/pixel, 64 sample/light

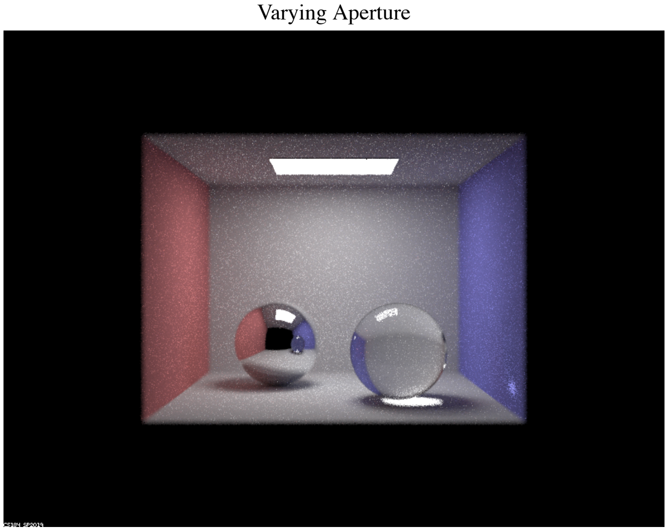
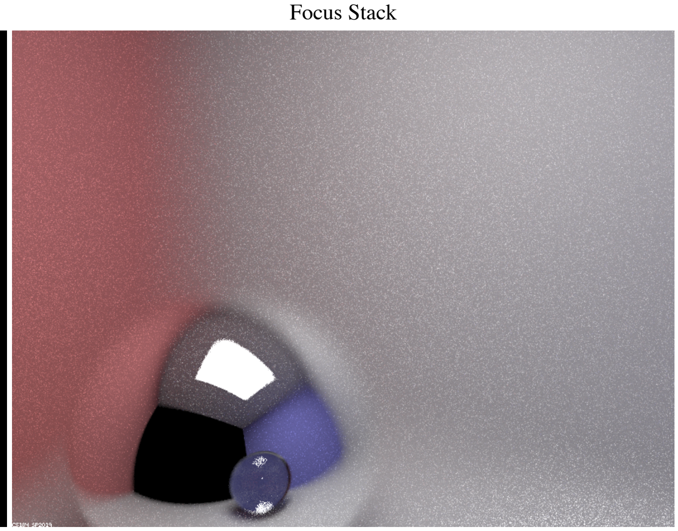



Uniform Hemisphere Sampling | Importance Sampling

Part 4: Depth of Field

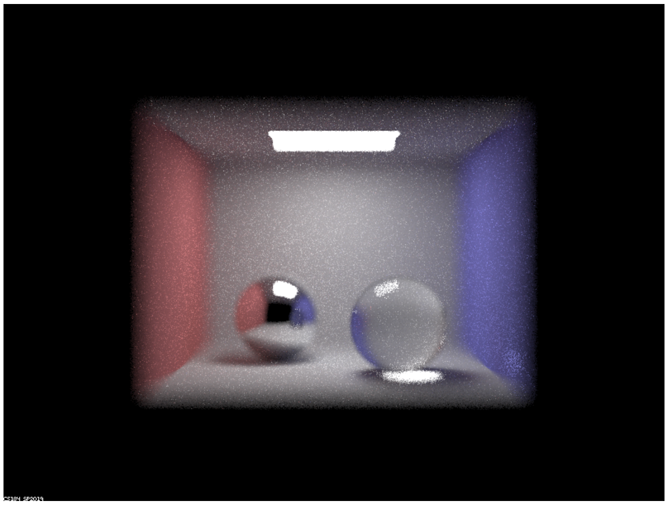
To simulate depth of field, we implemented functionality for a thin lens camera. Up until this point, we have been working with a pinhole camera, which is basically just a thin lens camera with an aperture of 0. This means, when viewing images, a pinhole camera keeps all objects in focus and at high enough settings, every object would be clean and sharp. However, that does not reflect what cameras and human eyes perceive in reality. We have finite windows through which to view the world, so to speak, so only a limited amount of light and distance can be percieved on a cameras sensor or our retinas. We simulate a thin lens by refracting light at distance in front of our camera and project the ray from the lens to objects at focal distance. Anything closer or further than the focal distance becomes blurry and unfocused, similarly to how a real camera would detect objects in space.

Varying Aperture | Focus Stack

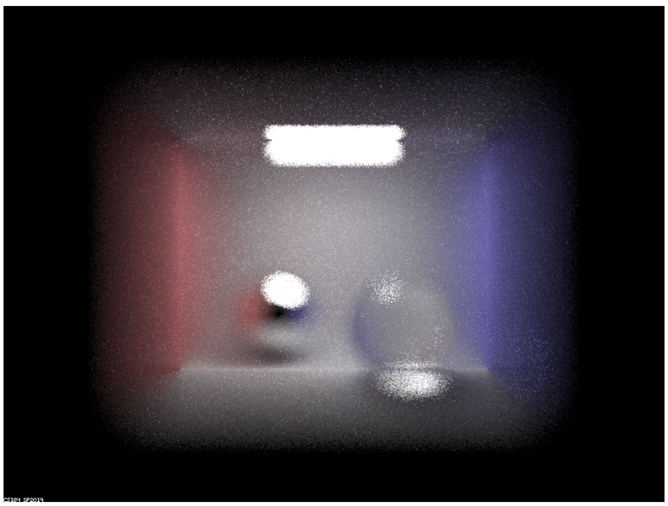
Aperture = 0.1 | Aperture = 0.25

Focal Distance = 4.7



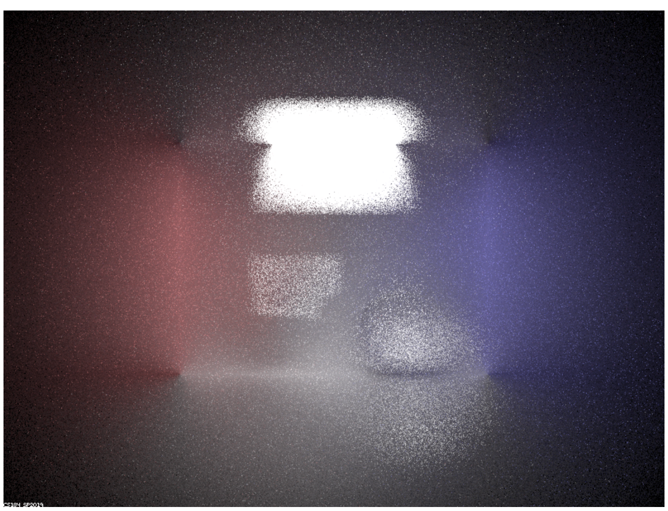
Aperture = 0.4

Focal Distance = 4.7



Aperture = 1.6

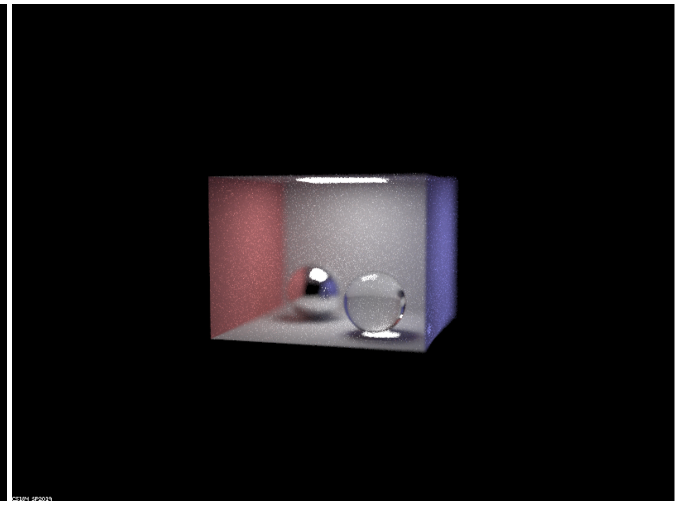
Focal Distance = 4.7



Aperture = 6.7

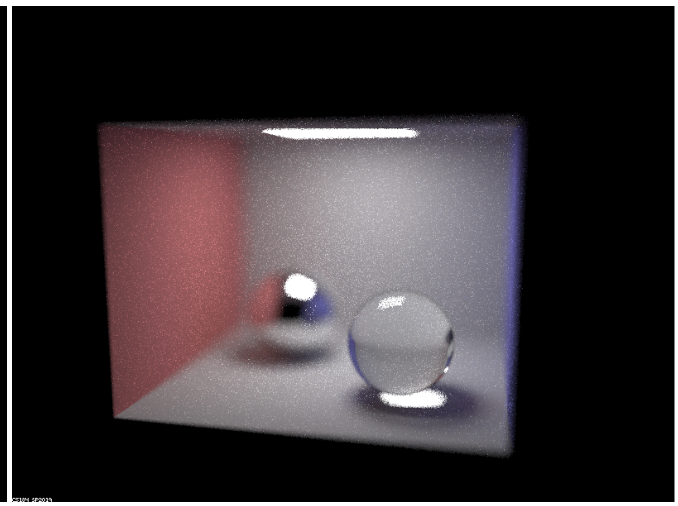
Focal Distance = 4.7

Focal Distance = 1.3



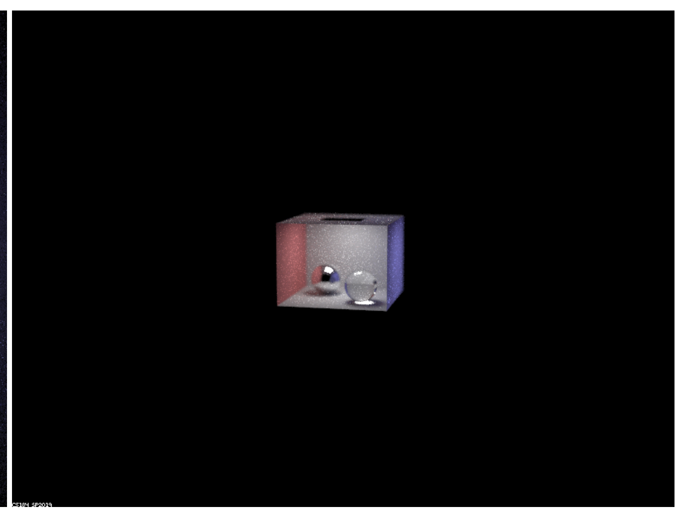
Aperture = 0.25

Focal Distance = 6.7



Aperture = 0.25

Focal Distance = 3.9



Aperture = 0.25

Focal Distance = 12.7