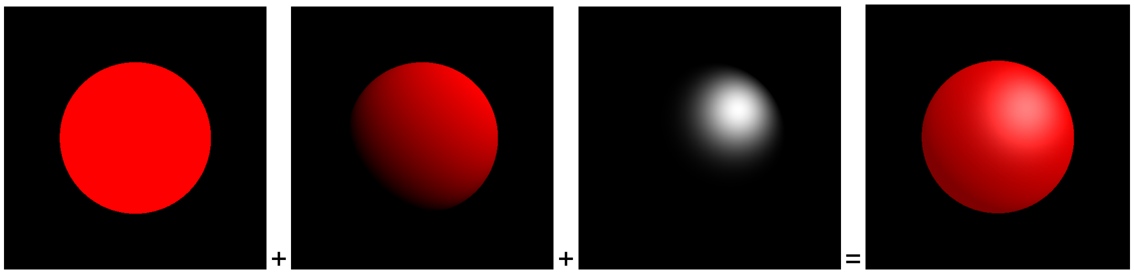
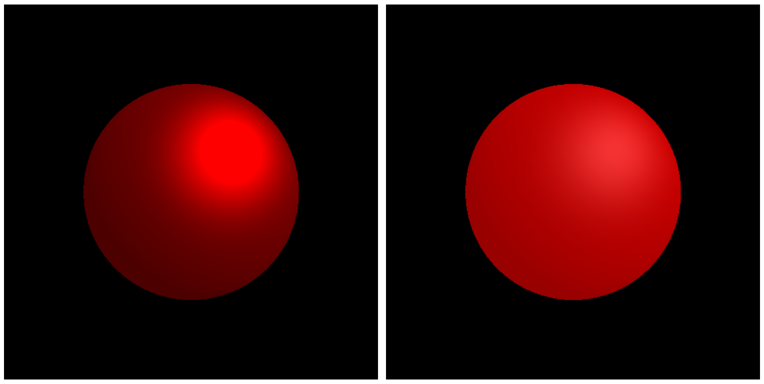
CSCI 5607 Assignment 1B Writeup

Messing around with Phong values

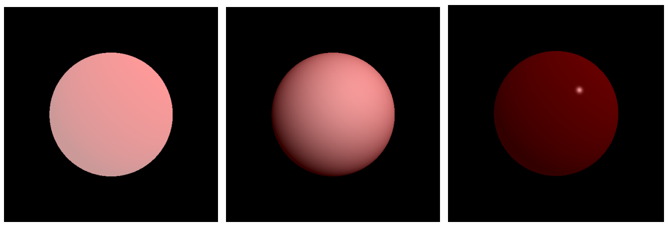


phong_test.input

The above images show the scene of a red dot being broken down into its components under the Phong-Blinn method, ambient, diffuse, and specular respectively.



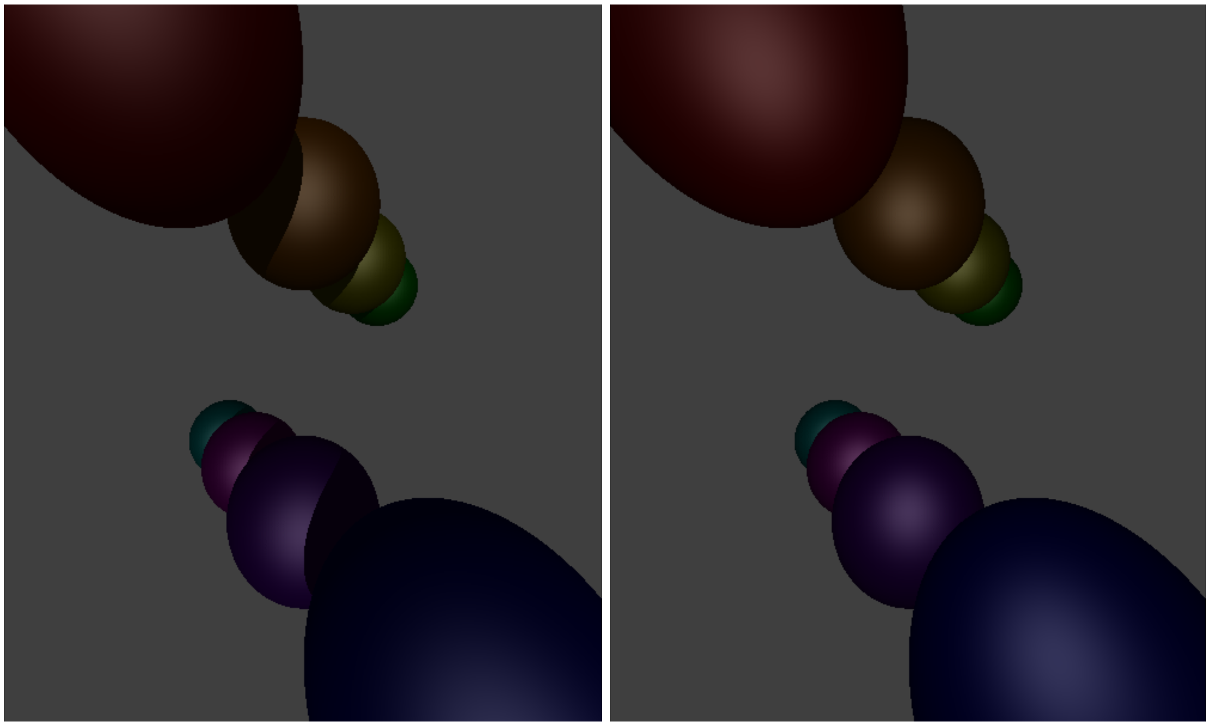
In the first image above, I tried to use the model to create a metallic material effect, with a high specular ratio and a red specular color. $k_a = 0.2$, $k_d = 0.2$, $k_s = 0.8$, $n = 10$.
In the second image, I made the sphere primarily reflect light diffusely, like chalk or plaster. $k_a = 0.6$, $k_d = 0.2$, $k_s = 0.2$, $n = 10$.



These three images all have $k_a = 0.2$, $k_d = 0.2$, $k_s = 0.6$, but they vary in n; the first has $n = 0$ (an interesting case, spec term always $k_s * 1$), the second $n = 1$, the third $n = 1000$.

Directional and Point light types

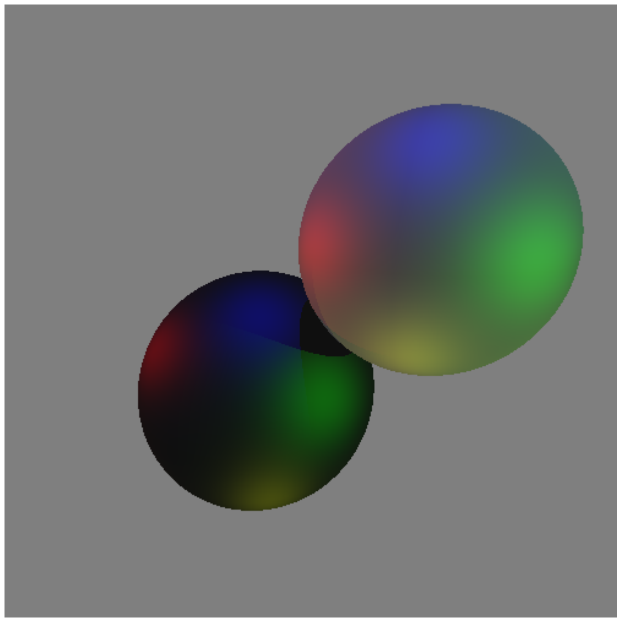
As per the specification of the assignment, my raytracer can now include light sources. The directional type has a constant vector direction to any point in space, while the point type's vector direction varies based on the location of a given point to the source.

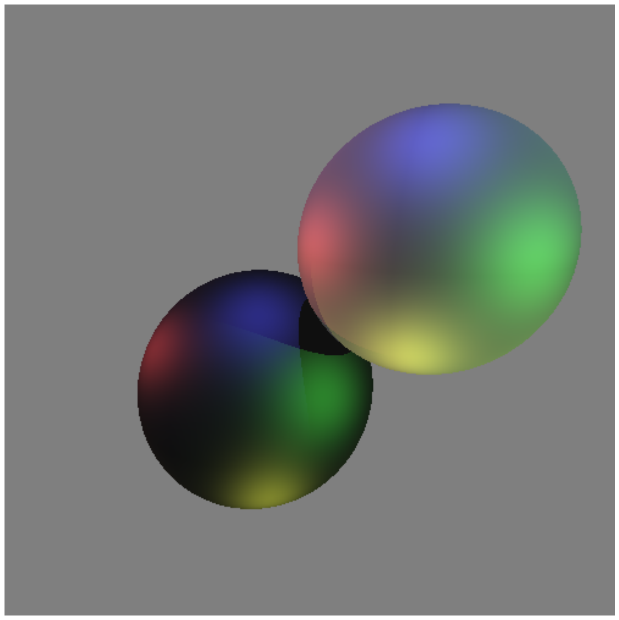


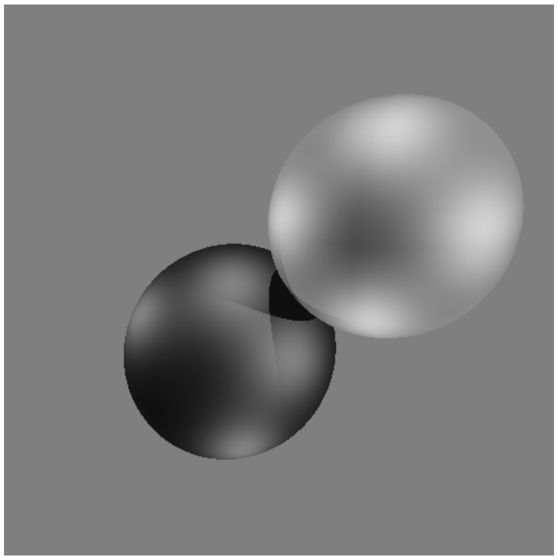
dots.input

At the left, I used a directional light source pointing the same way as the view, and you can see the specular highlight points towards the same side of each sphere. There is also a bit of shadow cast by one sphere onto the next.
At the right, I used a point light source originating from the same point as the eye. The highlights are pointed more towards the view here, and no visible shadow is cast due to the staggering of the spheres.

Multiple lights in one scene



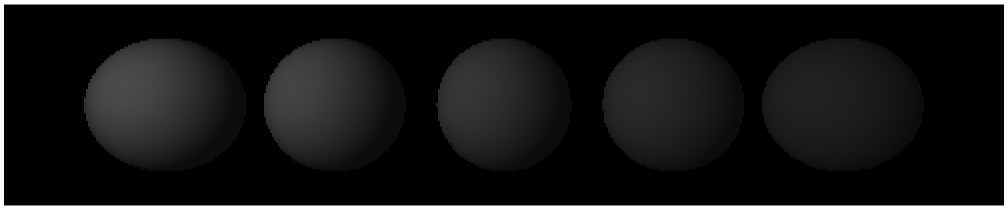




sphere.input

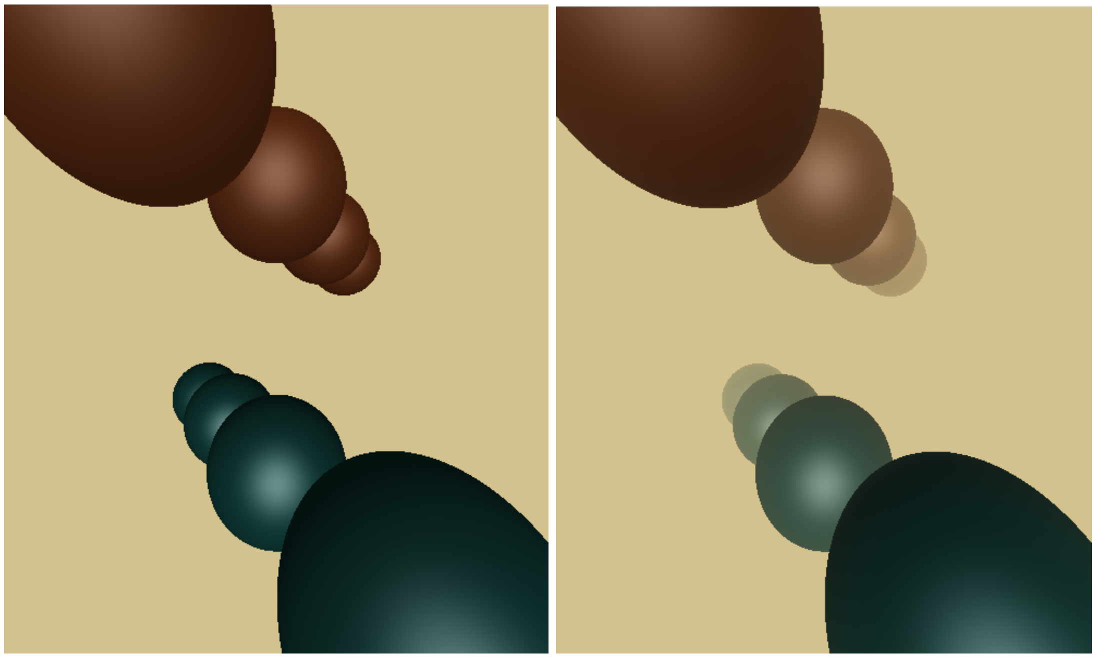
Using multiple lights on the same scene results in an interesting effect, mainly considering the multiple shadows being cast. You can see the sharp crossing point in the area interstitial to the two spheres. These scenes have four directional light sources, varying colors from one to another.
For the last image, I wanted to see if a problem would arise from using four light sources of the color (1, 1, 1), but nothing happened.

Point light attenuation


n/A

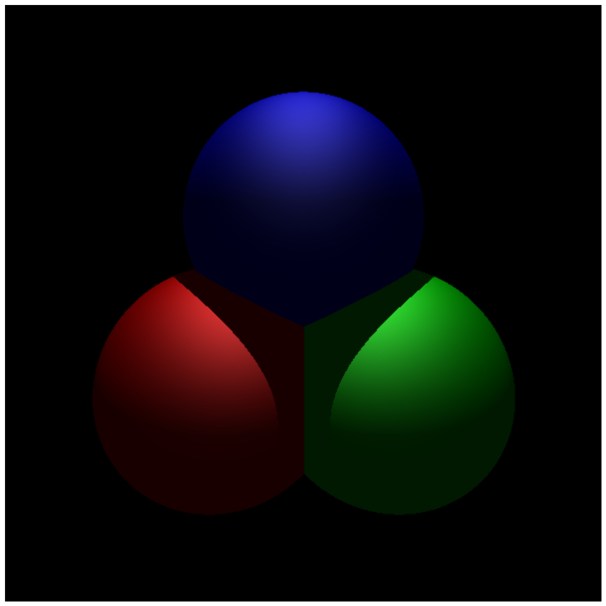
Around the beginning of the assignment, I managed to get point light attenuation working nicely. Unfortunately, after implementing depth cueing and shadows, I can no longer reproduce the correct effect, and this is the only surviving image.
I'll be working on fixing this in the future but for now I don't expect points for it.

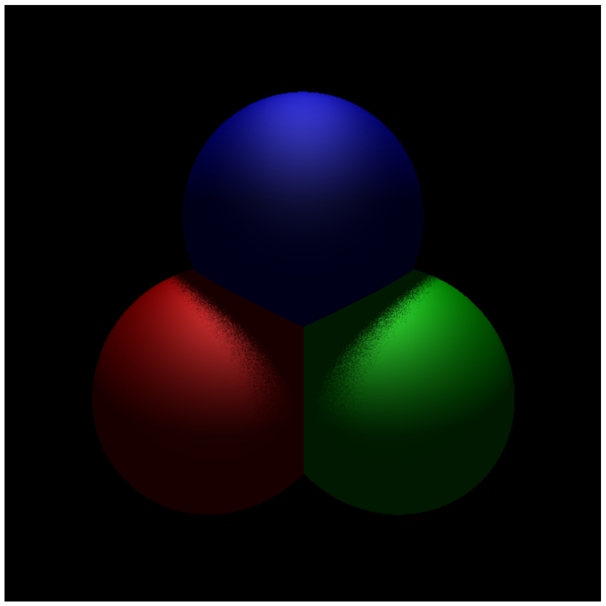
Depth cueing

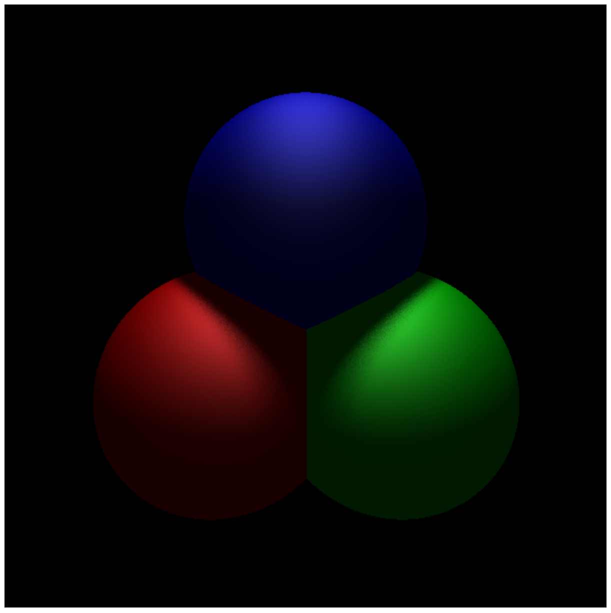

dc_test.input

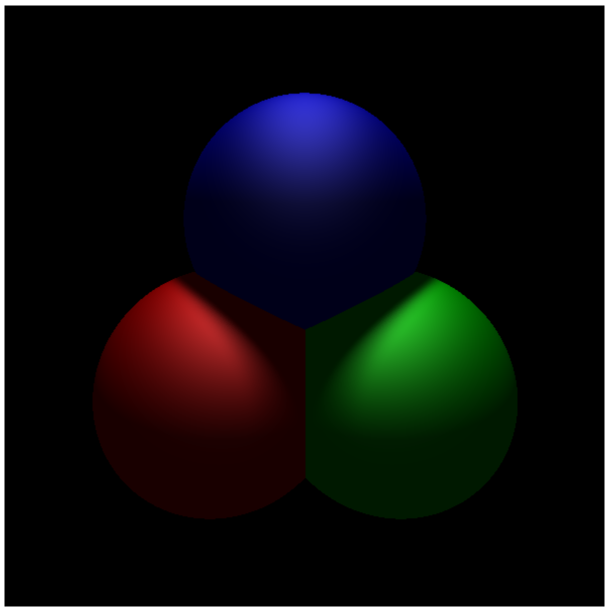
I implemented depth cueing into my raytracer, and the above right image is the result (with dc color = bkgcolor, a_{max} = 1, a_{min} = 0, d_{far} = 10, d_{near} = 2).

Soft shading









The above images show a scene with varying numbers of rays traced for shadows. The first has 1, the second 10, the third 100, and the fourth 10000 (this took my inefficient implementation about 7 minutes). As the number of rays increases, the blend between in-shadow and not-in-shadow becomes smoother with less noise.
A note on implementation: since this method is costly, I decided to add it as a file option instead of by default. It has the syntax "softshading *float:delta_light int:ray_ct*" where *delta_light* is the greatest amount the new supposed light point can stray from the actual origin on any axis and *ray_ct* is the number of rays cast for every shadow check.