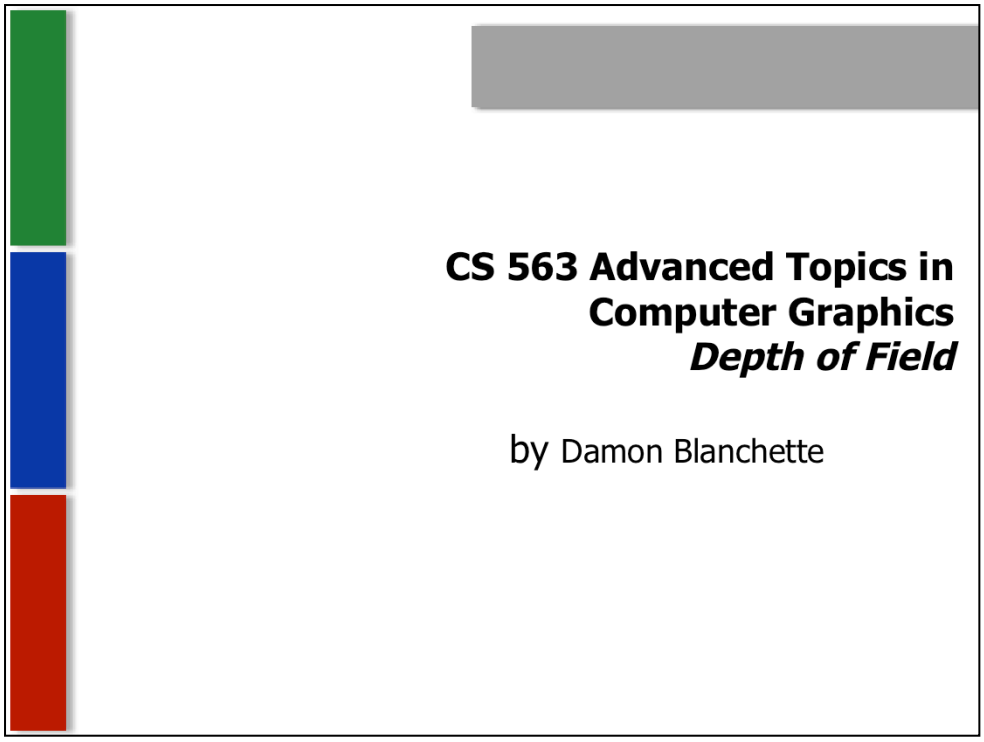
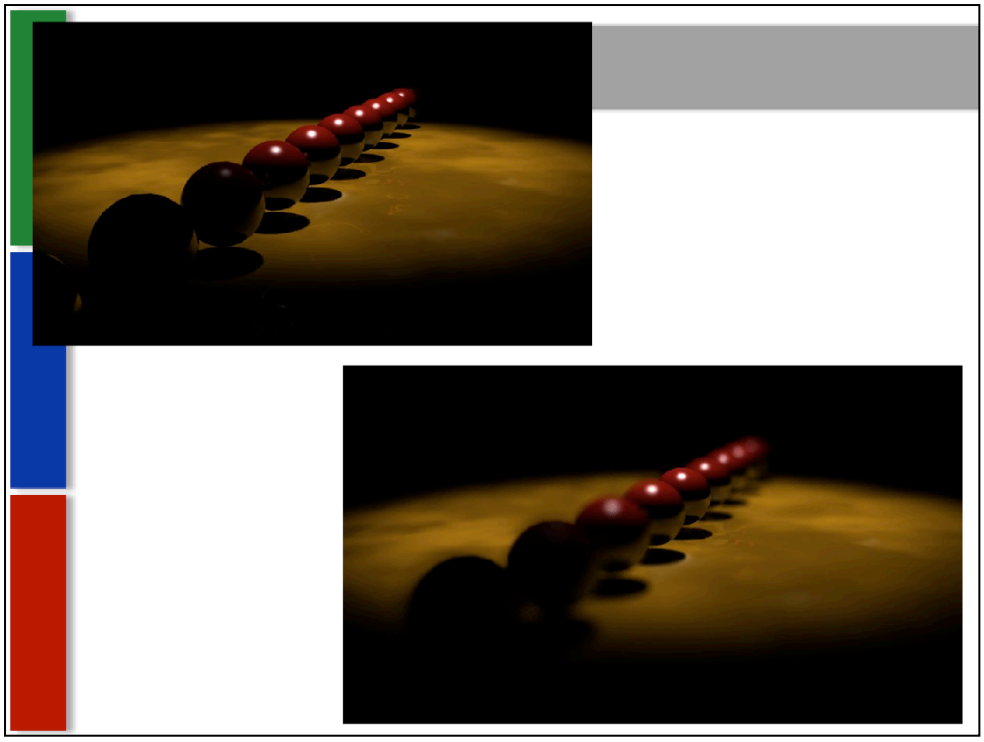

CS 563 Advanced Topics in Computer Graphics
Depth of Field
by Damon Blanchette

A real photo with a small depth of field. The depth of field is the part of the image that is in focus.

The size of the depth of field is adjusted with the lens's aperture. Big opening (smaller aperture number, f/2.8) = the above (more blur, smaller depth of field), small opening (bigger aperture number, f/22) = everything in focus.

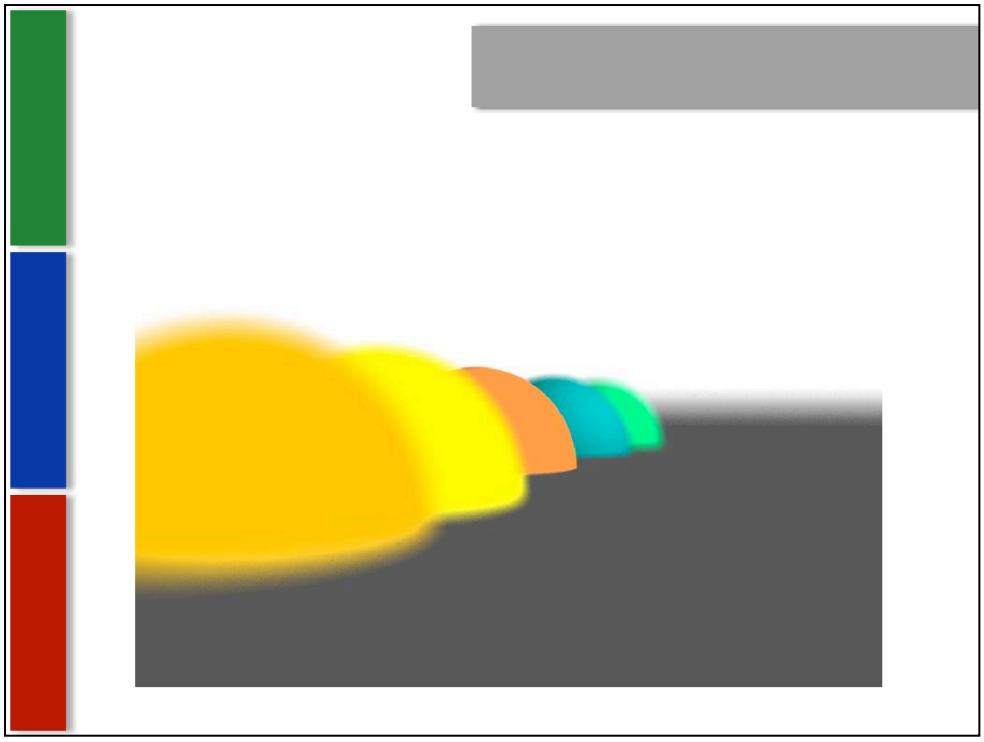
It's sort of similar with CG, but the thin lens is the aperture. Bigger thin lens = more blur.

Round bokeh. Bokeh is the way the blur looks, its aesthetic quality.



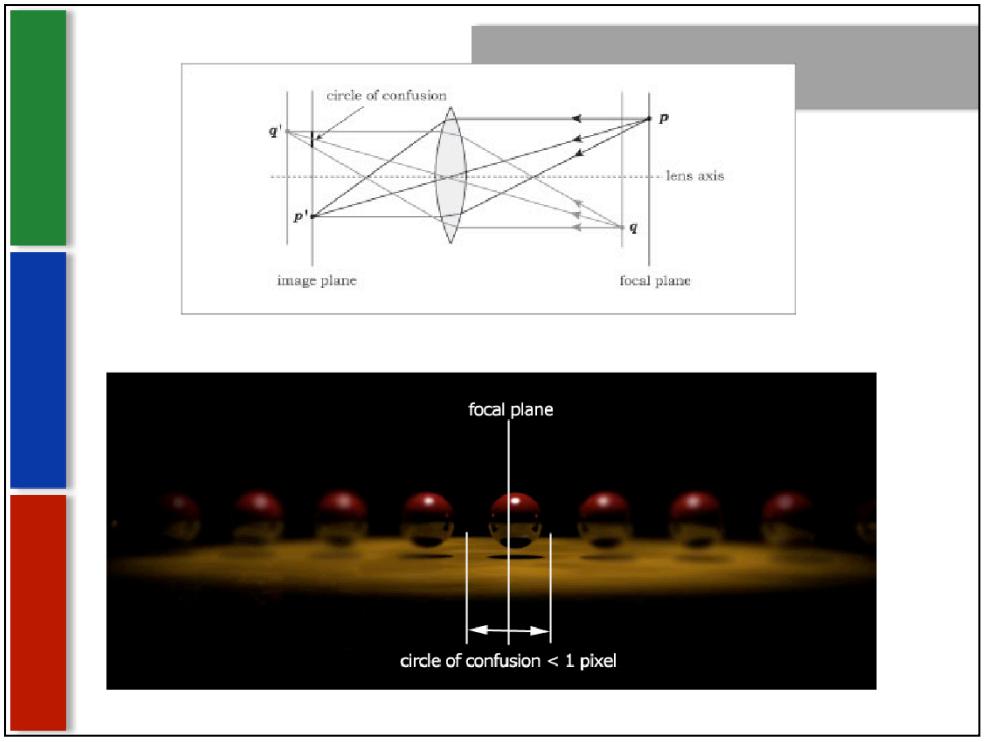
Top one, what could be called very large depth of field (everything in focus). small aperture.

Bottom one very small depth of field. Large aperture.



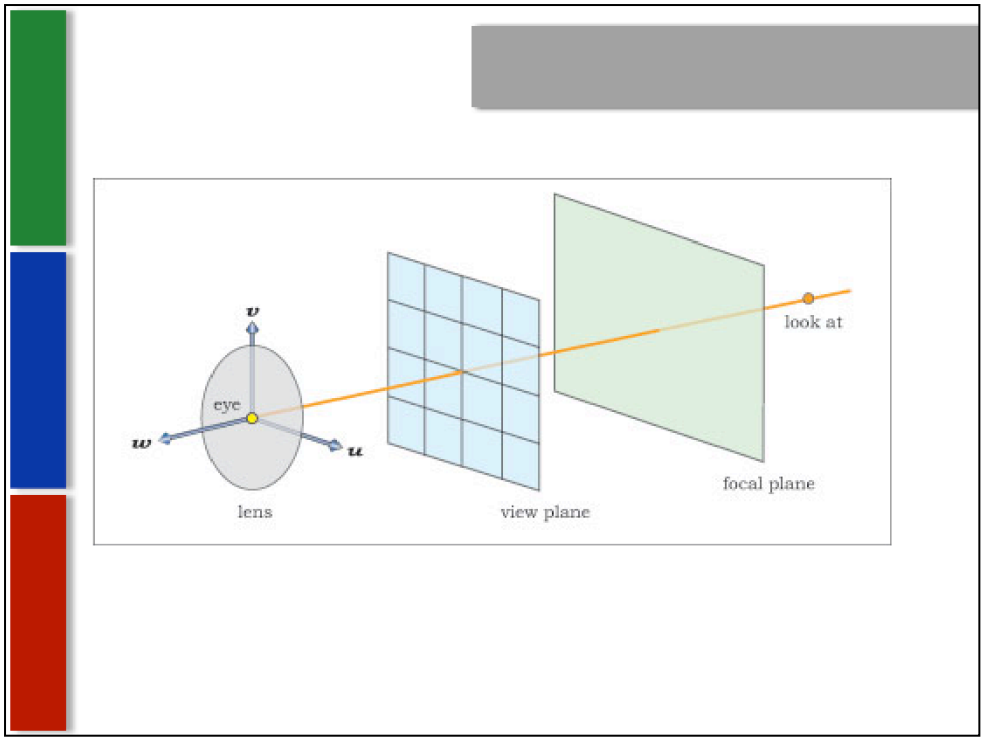
Using the class ray tracer, with 64 samples and a lens size of 2.

Antonio Peres, Evermotion.org



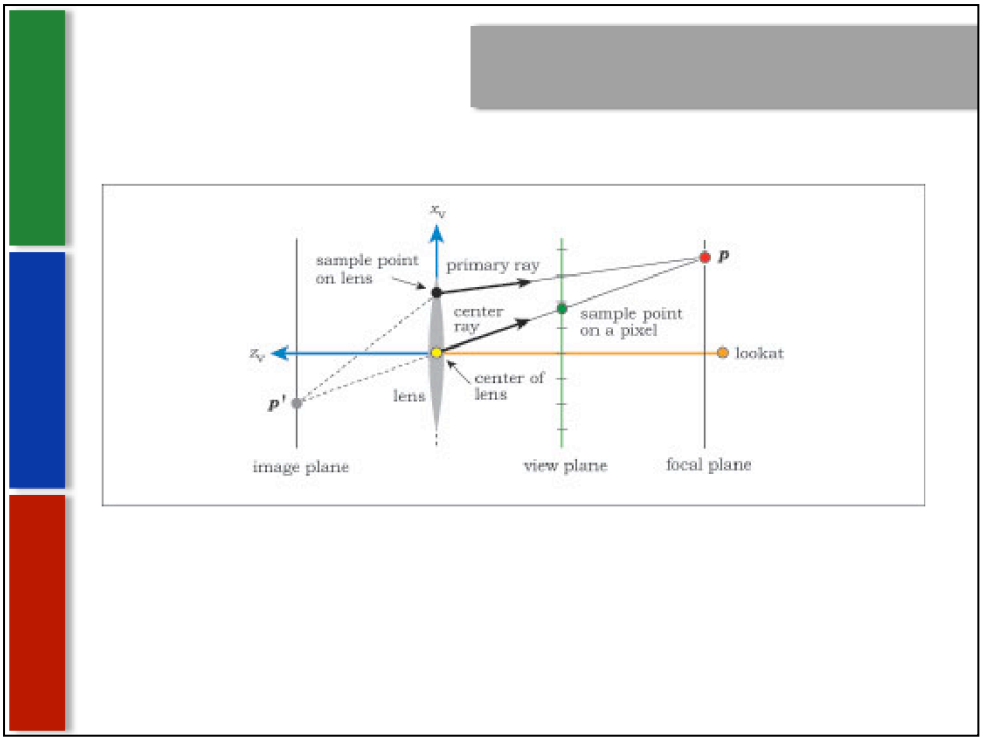
Thin Lens Theory. figure 10.2 from the book.

Circle of confusion – the larger the circle is, the more blurry something will be.



To make this work in ray tracing, a thin-lens camera is used, with the camera shaped like a disc.

As opposed to a pinhole camera, where all the rays come out of one infinitely small point, a thin-lens camera has rays coming out of sample points from the disc, which can be seen by the next slide….



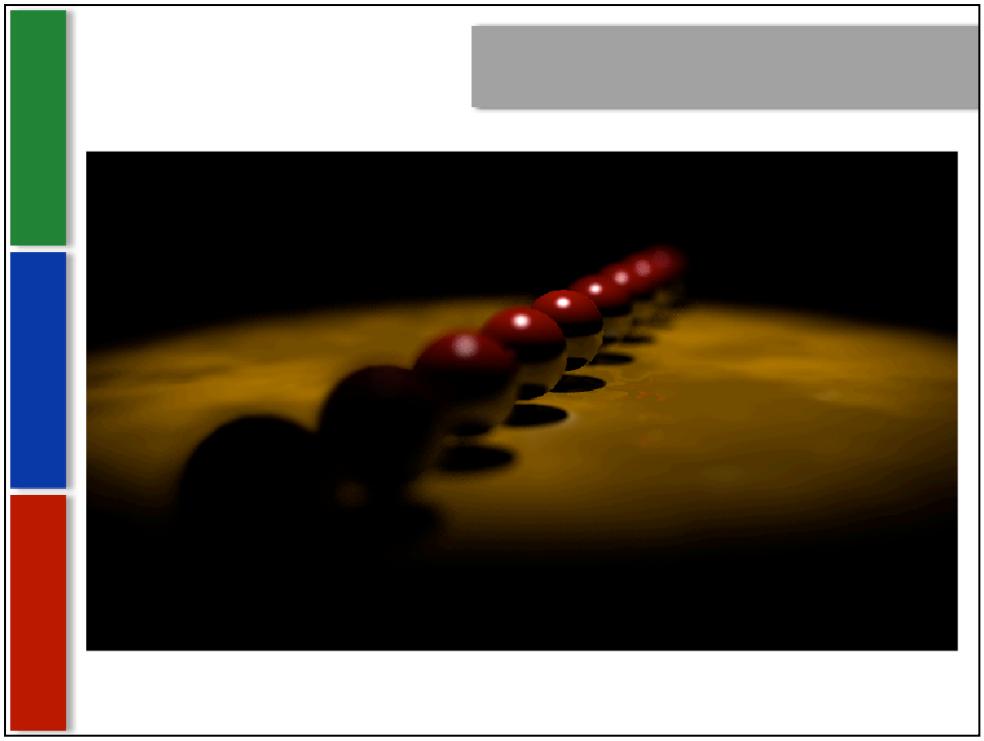
A note – the lens here is technically infinitely thin, but is drawn with thickness so we can see it.

Here we see a center ray at the eye point with a straight line through the center of the lens forward to a focal point and back to the image plane.

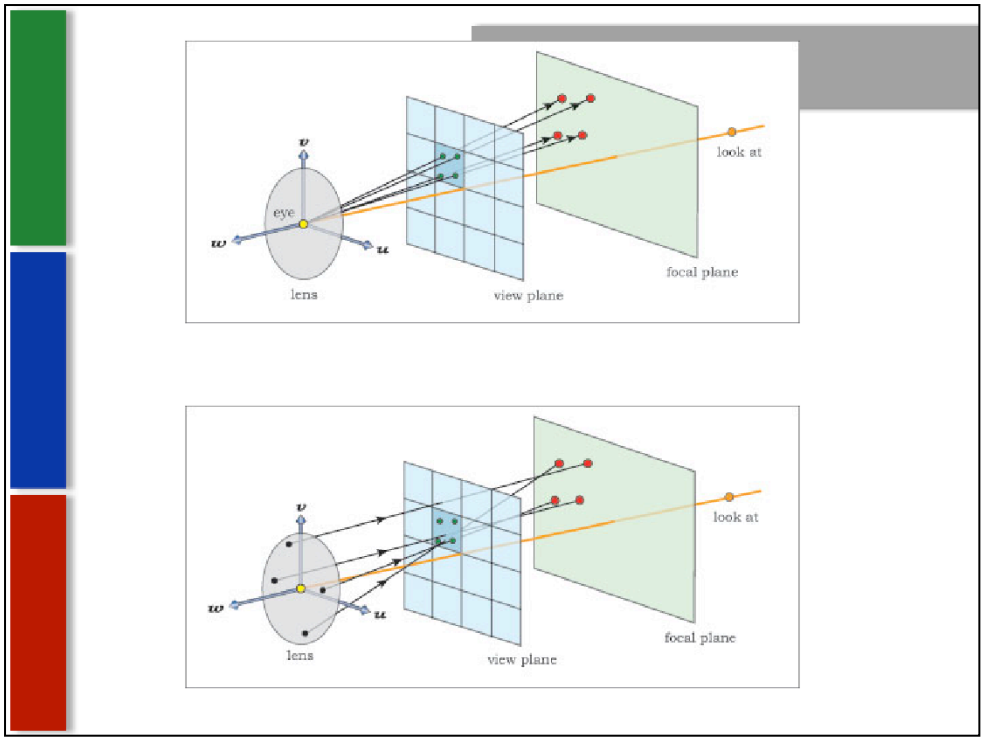
1. Figuring out point (P), where the center ray hits the focal plane, is the first step of simulating depth of field.
2. Second step is to use (P) and the sample point on the lens to compute the direction of the primary ray, so it also goes through (P), the point on the focal plane.
3. Third step is to ray trace the primary ray into the scene just like a ray from a pinhole camera.

The pixel is drawn according to the primary ray – the center ray does nothing to contribute color to the pixel, it's only used to figure out where (P) is.

How about antialiasing?



Depth of Field does a great job getting rid of aliasing, except for stuff right on the focal plane. The problem – if all the primary rays for each pixel use the same center ray, then everything on the focal plane will be in focus (good), but that part will not be antialiased. To fix this we can…



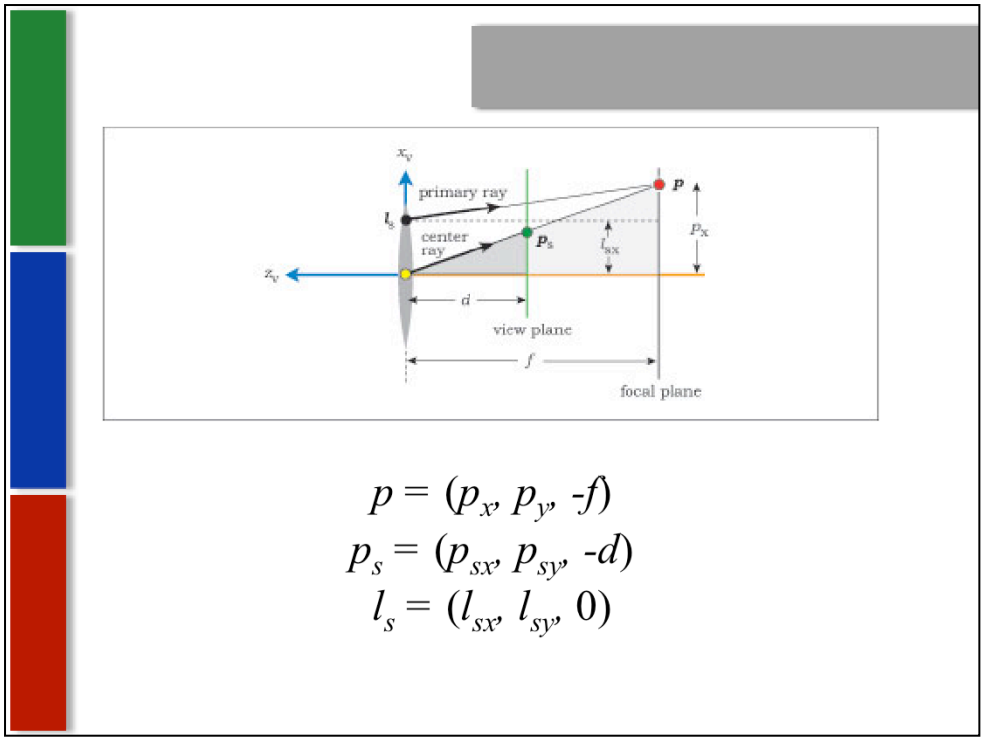
…use a different center ray for each primary ray.

The top diagram shows four different center rays going through different sample points on a pixel, and so hit the focal plane at different points.

The bottom diagram shows the four primary rays that start on their own sample points on the lens and hit the focal plane at the same four (P) points according to each center ray (one center ray for each primary ray).

In terms of sampling, we want to use the same number of samples for both the antialiasing and for the sampling of the lens. (here 4 samples each).

So, how do we find the (P) points on the focal plane where the center rays hit, for our primary rays to use?



$$p = (p_x, p_y, -f)$$
$$p_s = (p_{sx}, p_{sy}, -d)$$
$$l_s = (l_{sx}, l_{sy}, 0)$$

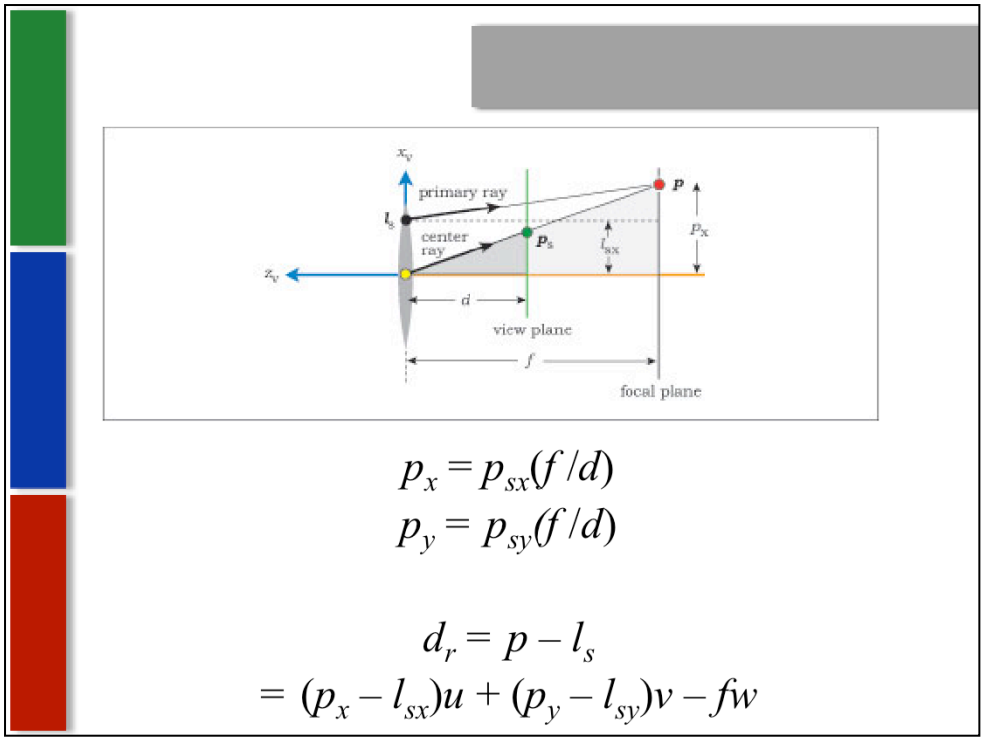
Luckily for speed's sake, we don't have to trace each center ray to find point (P). We can just use the geometry of similar triangles. Here the two gray-shaded triangles are similar.

The three points that we need for finding (P) and using it are: the pixel sample point P sub s on the view plane, the sample point L sub s on the lens, and the point P on the focal plane.

The viewing coordinates for these points are shown:
P is simply its x value, its y value, and f, which is the distance from the lens to the focal plane (which is where everything is in focus)
Ps is the same for its own x and y values, and d, which is just the distance from the lens to the view plane.
Ls has its own x and y axis, and 0 for the z value, because it rests right on the lens.

f and d are user-specified parameters for the thin-lens camera we are working on.

So, how do we figure out the values for point P?



$$p_x = p_{sx}(f/d)$$
$$p_y = p_{sy}(f/d)$$

$$d_r = p - l_s$$
$$= (p_x - l_{sx})u + (p_y - l_{sy})v - fw$$

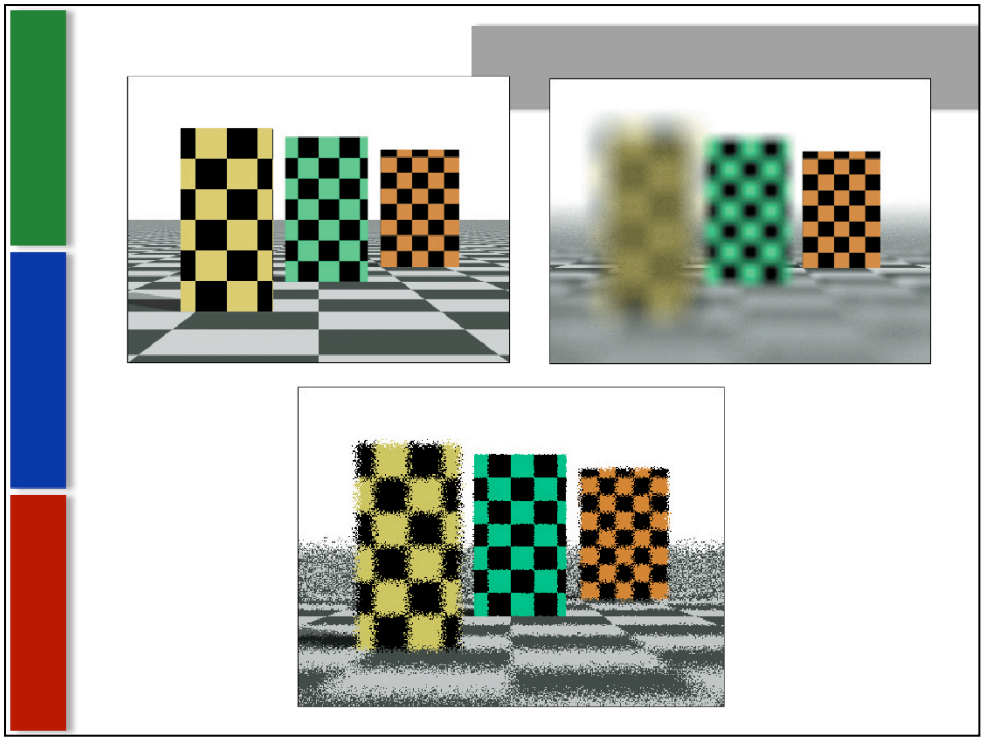
(same diagram to help with understanding)

The x and y components of the pixel sample point Ps, Psx and Psy, are known values, thanks to the equations back in chapter 9.4 (page 156), when figuring out directions for the rays in world coordinates.

The x and y components of the lens sample point Ls are also known because the sampler object stored in the thin-lens camera returns a sample point on a unit disk as part of the programming in the ray tracer, which makes sense (knowing where your sample points are seems pretty crucial).

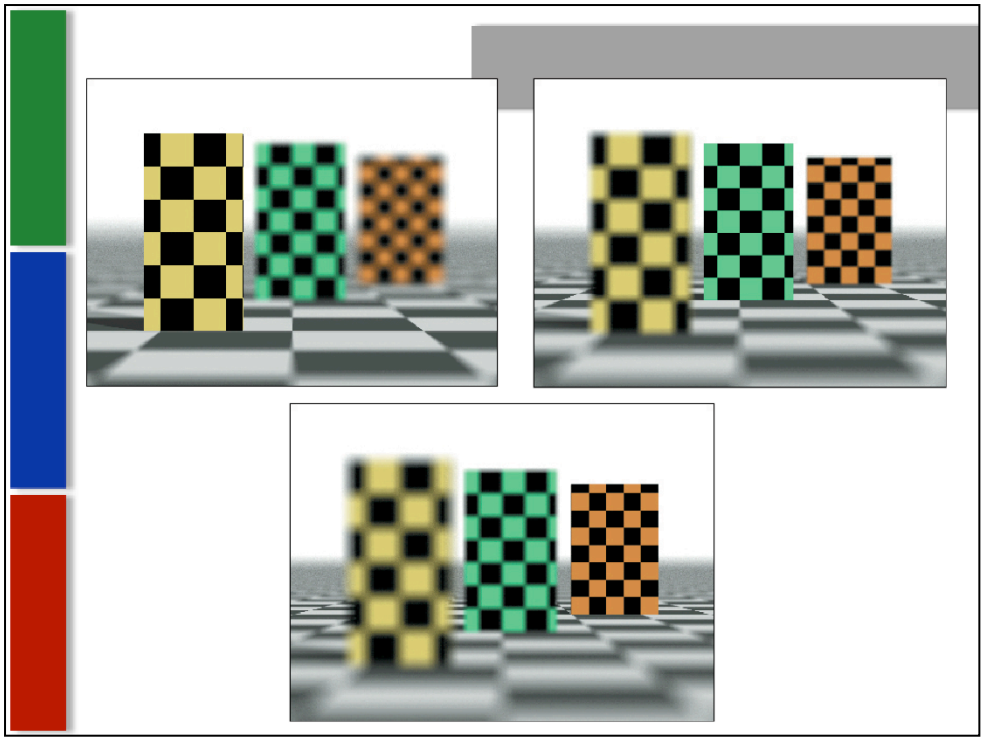
Now, as for our point P on the focal plane, its components can be calculated from the equations above. You're basically taking the points from Ps on the view plane and extending them backwards along the center ray vector based on the distances d and f.

For Dr, the direction of the primary ray, we take its end point (p) minus the start point (Ls). This will not be a unit vector, so it needs to be normalized after this calculation.

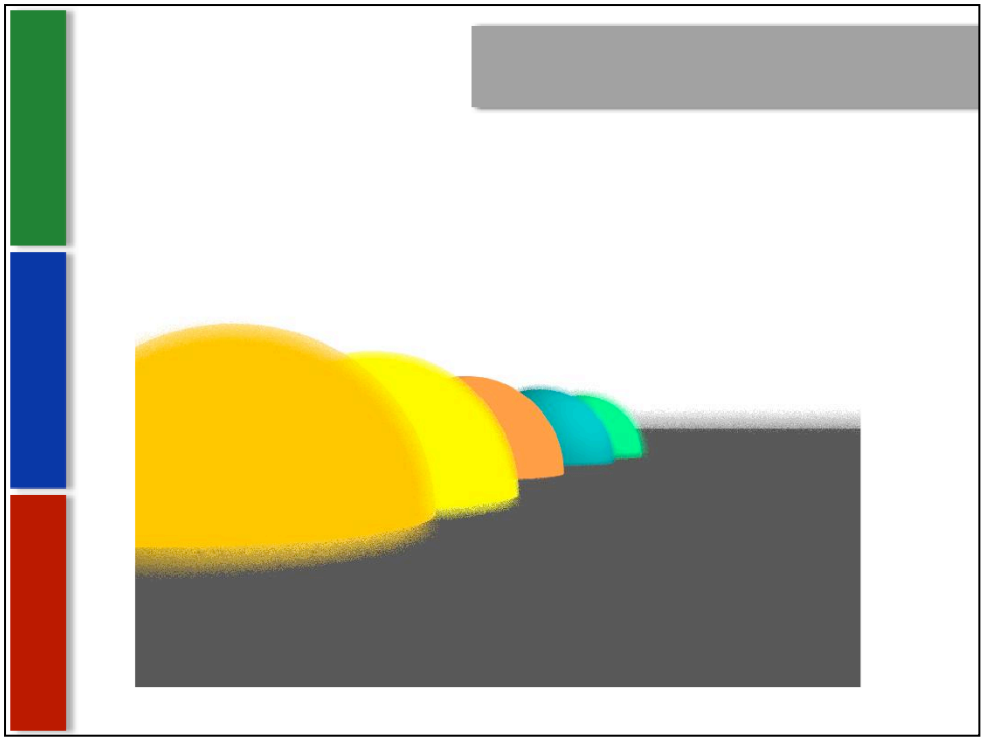


Top left image, lens radius is zero, so it's exactly the same as the pinhole camera (same as a tiny aperture on a real camera). Top right has a huge lens radius and a large number of samples per pixel (100), creating a very blurry section.

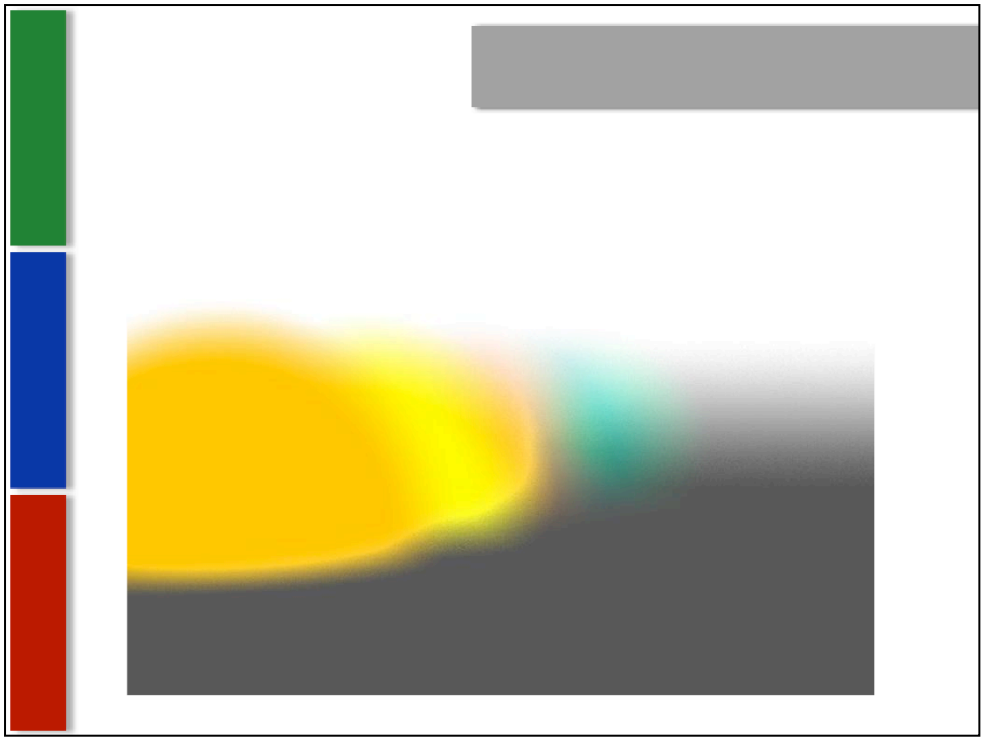
Bottom image has one random sample per pixel, so it's very noisy instead of blurry.



The effect of different focal distances.

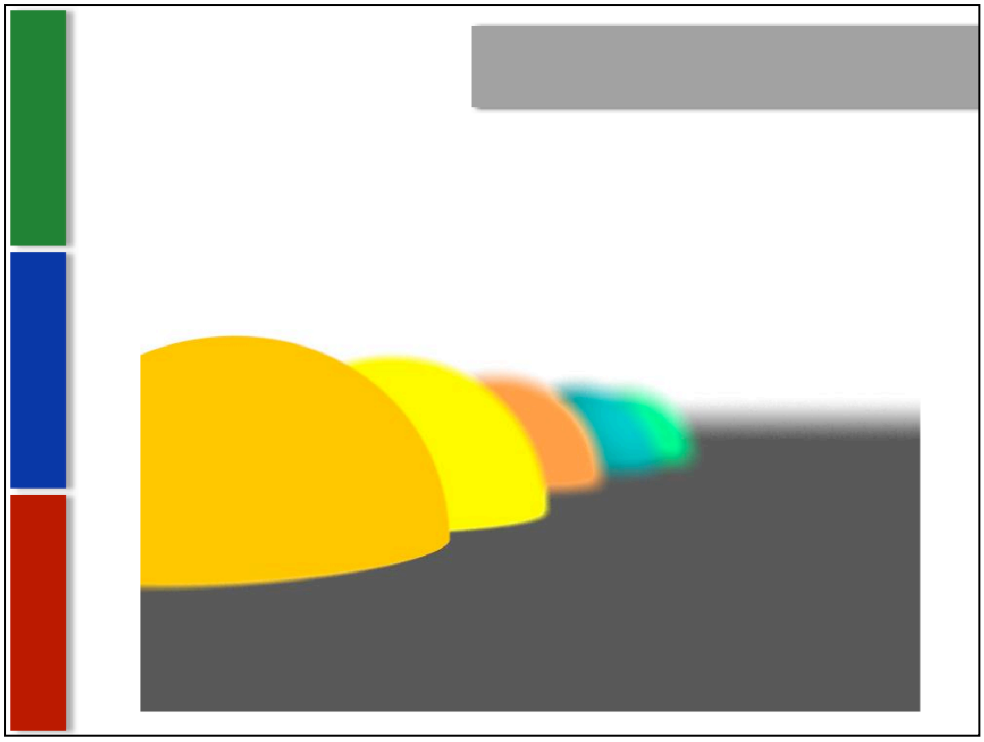


6 samples per pixel (thus the noise), lens size 2



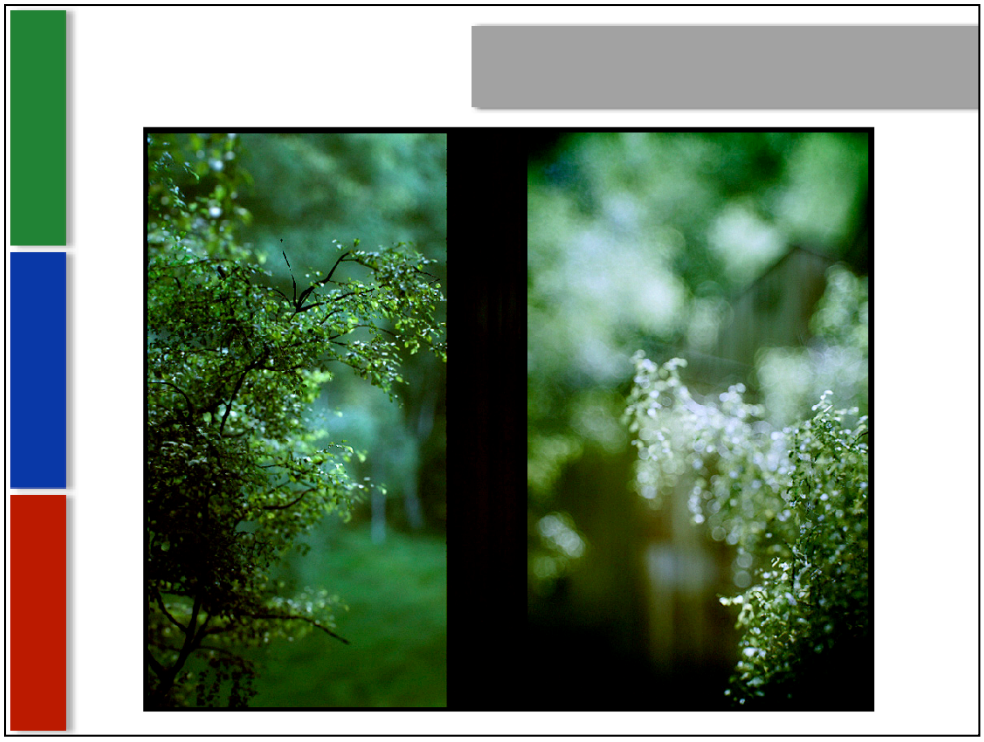
64 samples per pixel, focal distance too close to lens.

What happens when a bug flies in front of your camera just as you focus and take the picture.



64 samples per pixel, lens size 1 (less blur), and first sphere in focus.

Interesting yellow artifacts on the peach colored sphere.



The Third and the Seventh

Questions?