Gloss and translucency

The mirror-like form of reflection, when used to approximate glossy surfaces, introduces a kind of aliasing, because we are under-sampling reflection (and refraction).

For example:

Distributing rays over reflection directions gives:

Simulating glossy reflection

Let’s return to the glossy reflection model, and modify it – for purposes of illustration – as follows:

We can visualize the span of rays we want to integrate over, within a pixel:
Returning to the reflection example, Whitted ray tracing replaces the glossy reflection with mirror reflection:

Thus, we render with anti-aliasing as follows:

We can model a glossy surface by choosing the reflection direction to be randomly perturbed away from the ideal reflection direction.

To ensure good (well-distributed) perturbations, we decompose reflection directions into bins:

We can also perturb the sub-pixel viewing ray directions to get:

These ideas can be combined to give a particular method called distribution ray tracing [Cook84]:

- uses non-uniform (jittered) samples.
- replaces aliasing artifacts with noise.
- provides additional effects by distributing rays to sample:
  - Reflections and refractions
  - Light source area
  - Camera lens area
  - Time

In the next few slides, you will see illustration of these effects. In each case, they can be simulated efficiently with distribution ray tracing.

[This approach was originally called “distributed ray tracing,” but we will call it distribution ray tracing (as in probability distributions) so as not to confuse it with a parallel computing approach.]
**Depth of field**

To simulate a camera, we can model the refraction of light through a lens. This will give us a “depth of field” effect: objects close to the in-focus plane are sharp, and the rest is blurry.

![Depth of field diagram](image)

**Summary**

What to take home from this lecture:

1. The limitations of Whitted ray tracing.
2. The main idea behind distribution ray tracing and what effects it can simulate.