Lecture 12: Ray Tracing Basics
Reading

**Required:**
- Hearn & Baker, 14.6

**Optional:**
- Glassner, chapter 1
- Foley *et al.*, 16.12
Ray Tracing

• A term from optics
• A “physical” simulation of the particle theory of light

• In the 1960s, ray tracing seemed like a great idea, but nobody could do it well enough to beat cheaper image synthesis methods.
• These days, we can follow the simulation deeply enough to get great results!
• But there are some visual phenomena that ray tracing cannot do.
Why Ray Tracing?

- So far, we can do ray casting: for each pixel in the projection plane, find the object visible at that pixel and apply your favorite shading model.
- What does this model miss?
Forward Ray Tracing

- Rays emanate from light sources and bounce around in the scene.
- Rays that pass through the projection plane and enter the eye contribute to the final image.

- What’s wrong with this method?
Backward Ray Tracing

• Rather than propagating rays indiscriminately from light sources, we’d like to ask “which rays will definitely contribute to the final image?”

• We can get a good approximation of the answer by firing rays from the eye, through the projection plane and into the scene
  – These are the paths that light must have followed to affect the image
Kinds of Rays

- A ray that leaves the eye and travels out to the scene is called a **primary ray**.
- When a ray hits an object, we spawn three new (backward) rays to collect light that must contribute to the incoming primary ray:
  - **Shadow rays** to light sources, used to attenuate incoming light when applying the shading model
  - **Reflection rays**, which model light bouncing off of other surfaces before hitting this surface
  - **Transparency rays**, which model light refracting through the surface before leaving along the primary ray
- Shadow rays stop at light sources, but reflection and transparency rays behave just like primary rays!
Example of Ray Tracing
The Ray Tree

- A primary ray hits a surface and spawns reflection and transparency rays. Those rays may hit surfaces and spawn their own rays, etc.
- We can represent this process schematically using a ray tree:

```
  Eye ray
    /   \
   /     \
  S1     S2
     /\    /\  \
    /  \  /  \  \
  Object 3
    /     \
   /       \
 S3     S4
   /\      /\  \
  /  \  /  \  \
 Object 6
    /\    /\  \
   /  \  /  \  \
  T1     R1
    /     \
   /       \
 S5     S6
   /\      /\  \
  /  \  /  \  \
 Object 9
    /     \
   /       \
 R2     T2
    /     \\n   /       \\n R3      R3
 ...     ...
```

- What is a ray tree good for?
Controlling Tree Depth

• Ideally, we’d spawn child rays at every object intersection forever, getting a “perfect” colour for the primary ray.
• In practice, we need heuristics for bounding the depth of the tree (i.e., recursion depth)
• ?
Parts of a Ray Tracer

• What major components make up the core of a ray tracer?
Summary

• Understanding of basic ray tracing concepts
• Forward vs. backward tracing
• Classification of rays
• The ray tree
• Terminating recursion
• Parts of a ray tracer