Ray Tracing

Writing a Very Simple Version

Today's Short Film

Ice Age trailer

from

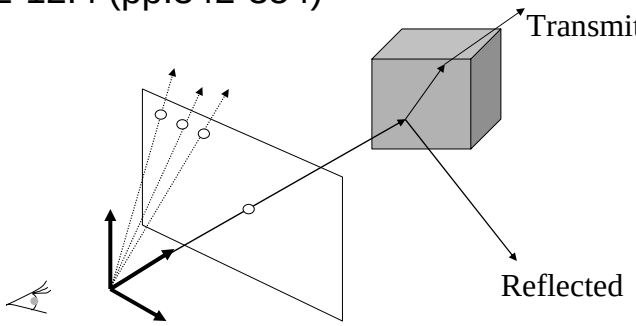
Blue Sky Studios

What Makes a Good Picture?

- Contents (3D models).
- Lighting.
- Reflection.
- Shadow.
- Surface textures.

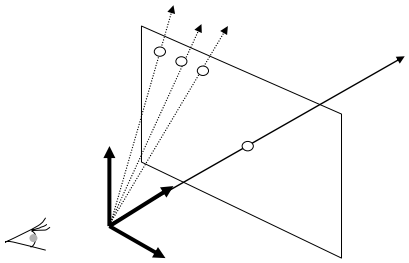
Ray Tracing Algorithm

- An overview in Pharr's 1.2
- More detail in Watt's 10.3.1 (pp.284-286) and 12.2-12.4 (pp.342-354)



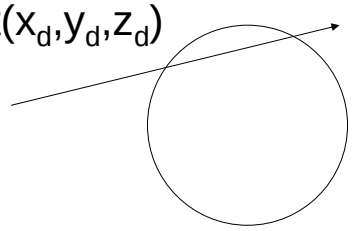
Creating a Ray

- Parameters:
  - Image Plane (position, size, and resolution)
  - Viewpoint
  - Which ray (x, y)?



Ray-Object Intersection

- For example: sphere
  $(x-x_0)^2+(y-y_0)^2+(z-z_0)^2=r^2$
- Ray: $(x,y,z)=(x_1,y_1,z_1)+t(x_d,y_d,z_d)$
- Find t that satisfy
  $(x-x_0)^2+(y-y_0)^2+(z-z_0)^2=r^2$
- Normal vector?
- Also easy for planes, cones, …etc.



Shading Models

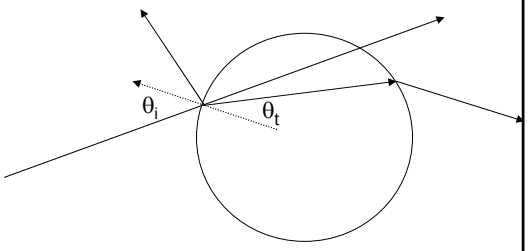
- Pixel color = ambient + diffuse + specular + reflected + transmitted
- The weight of each is determined by the surface properties.
- We will discuss each of them within the next a few lectures.

Light Source & Shadow

- Point light is easy to implement, but does not look real.
- How to determine a surface point is in the shadow?
- In real world: area light with soft shadow.

Reflection and Refraction

- Reflected ray is determined by:
  - incoming ray and normal vector.
- Refracted ray is determined by:
  - Incoming ray
  - Normal vector
  - And density
- Snell's law:
- $\eta_l \sin \theta_i = \eta_t \sin \theta_t$



Recursive Algorithm

- The reflected ray and refracted ray are traced recursively.
- Termination condition:
  - Depth of trace
  - Weight (to the final pixel color) of ray

Advantage

- We get all the following automatically:
  - Hidden surface removal
  - Shadow
  - Reflection
  - Transparency and refraction

Disadvantage

- Slow. Many rays are spawned.
- Slow. Ray-object intersection for every ray and every object. (We will discuss how to avoid this in the next lecture).
- The lighting is still not completely right!

Assignment 1 – A Ray Tracer

- Split into two parts.
- Part A due October 3.
  - Camera module
  - Object module (sphere only)
  - No recursive ray tracing
  - Simple output (in text mode)
- The rest (Part B) are due October 17.

Required Modules

- Camera Module
- Object Module
- Ray Tracer Module (main program)
- Display (Output) Module

Camera Module

- Definition of eye position and image plane.
- Generating a ray if given (x, y)
  - Note that x and y may be real numbers (not integers).

Object Module

- Sphere type only (for now).
- Ray-object intersection.
- Light.
- Read from files.
- Camera is sometimes defined in the object file for convenience.

Ray Tracer Module

- Integration of other modules.
- Shading.
- Spawn reflected and refracted rays.

Display (Output) Module

- Output to a text file for now.
- Example: output 0 if no intersection and 1 if intersecting an object.
- May create PPM, TIFF, or JPEG files later.

Part A due October 2

- Camera module
- Object module
  - Read from a file
  - Sphere and Light only
- Ray tracer module:
  - No shading. No reflection and refraction.
- Display module (in text mode)

Part B due October 16

- Object module
  - Add at least a plane type.
- Ray tracer module:
  - Add shading, reflection, and refraction.
- Display module:
  - PPM, TIFF, or JPEG library will be provided.
- Add a demo scene of your own.