Ray Tracing

- State of the art in visual realism
- Apply a detailed lighting model
  - Reflections
  - Refractions
  - Shadows
  - Non-ideal light sources
  - etc.

1

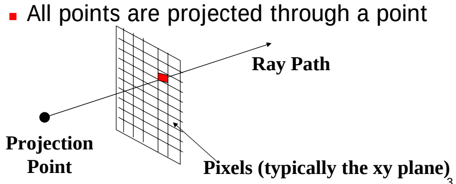
Naïve Approach

- Examine each light source
- Determine the directions it casts light
- Trace the paths of the light as it
  - Reflects
  - Refracts
- Until…
  - Most are not of interest

2

Pinhole Camera Approach

- Goal: How much light hits a pixel?
- Trace the light rays that strike a pixel
- Recall the projection model
  - All points are projected through a point



3

Basic Ray Tracing

- Determine the ray path for each pixel
  - P_{prp} – Projection reference point
  - P_0 – Pixel position
  - $u = P_0 - P_{prp}$; Ray direction (normalized)
  - Actual Ray: $P = P_0 + su$
- Intersect the ray with all surfaces
  - Find the closest (smallest s)

4

Intersecting With a Sphere

- Sphere equation
  - $|P - P_c|^2 = r^2$
- Derivation
  - $|P_0 + su - P_c|^2 = r^2$
  - $s = u \cdot (P_c - P_0) \pm \sqrt{(u \cdot (P_c - P_0))^2 - |P_c - P_0|^2 + r^2}$
  - If determinant < 0, no intersection

5

Intersecting With a Polyhedron

- Discard all back faces
  - $u \cdot N > 0$, pos. cosine means facing away
- Planar equation
  - $Ax + By + Cz + D = 0 = N \cdot P + D$
- Derivation
  - $s = -\frac{D + N \cdot P_0}{N \cdot u}$
- Complication: Did it intersect?

6

Reducing Intersection Costs

- Most expensive computations
- Tessellated objects may have 100s of faces
  - Do we “intersect” with each?
- Reduce the number of potential intersections
  - “Simplify” objects
  - Bound each one with a sphere
    - Intersect with the sphere first

7

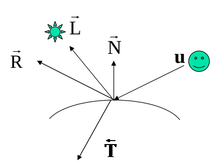
Reducing Intersection Costs

- Extending the bounding sphere
  - Put groups of objects in a sphere
- Subdivide the space
  - Grid of cubes
  - Note which objects are in which cube
  - Trace the ray through a path of cubes

8

Ray Interaction With a Surface

- Now we have the closest surface
  - What next?
- Follow illumination rules



9

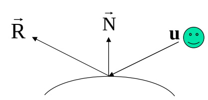
Shadow Ray (L)

- Is this point directly illuminated?
- Identify shadow rays to all light sources
- Trace them back toward the light source
- If it intersects a surface first, ignore
  - May miss some reflected illumination
- If not, apply illumination to the surface
  - Use the rules from lecture 5

10

Reflected Secondary Ray (R)

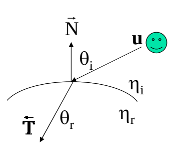
- Continue to trace the ray in reflection
- Use the specular reflection derivation
  - $\hat{R} = (2\hat{N} \cdot \hat{u})\hat{N} - \hat{u}$
- Continue to trace the reflected ray



11

Refracted Secondary Ray (T)

- Continue to trace the ray in refraction
- Apply Snell's Law
  - Consider total internal reflection



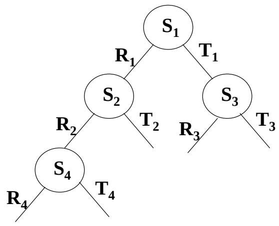
$$\sin\theta_r = \frac{\eta_i}{\eta_r}\sin\theta_i$$

$$\vec{T} = \left(\frac{\eta_i}{\eta_r}\cos\theta_i - \cos\theta_r\right)\vec{N} - \frac{\eta_i}{\eta_r}\vec{L}$$

12

Ray Tracing Tree

- Recursively record the rays in a tree



13

Exit Conditions and Color

- When do we stop tracing?
  - Case I: We strike a light source
  - Case II: Maximum tree depth or distance
- Processing the color
  - Start at the bottom (light or ambient surface color)
  - Recursively accumulate the colors
    - Don't forget the shadow rays (L)
  - Attenuation is common

14

Anti-Aliasing

- Supersampling
  - Use a small set of rays for each pixel
- Distributed rays
  - Pick the set with some randomness
  - This injects helpful noise
- Extended light sources
  - Result in sets of shadow rays

15