Ray Tracing

Thanks to UDel and MIT

Outline

• Recursive rays
  - Reflection
  - Refraction

Ray Tracing

• Model: Perceived color at point $p$ is an additive combination of local illumination (e.g., Phong), reflection, and refraction effects
• Compute reflection, refraction contributions by tracing respective rays back from $p$ to surfaces they came from and evaluating local illumination at those locations
• Apply operation recursively to some maximum depth to get:
  - Reflections of reflections of ...
  - Refractions of refractions of ...
  - And of course mixtures of the two

Reflection

• Reflection angle = view angle

Reflection

• Reflection angle = view angle

\[ \vec{R} = \vec{V} - 2(\vec{V} \cdot \vec{N})\vec{N} \]
Mirror Reflection

- Compute mirror contribution
- Cast ray
  - In direction symmetric wrt normal
- Multiply by reflection coefficient (color)

Amount of Reflection

- Traditional (hacky) ray tracing
  - Constant coefficient reflectionColor
  - Component per component multiplication

Example: Reflections at depth = 0
Example: Reflections at depth = 1

Example: Reflections at depth = 2

Example: Reflections at depth = 3

Transparency

- Compute transmitted contribution
- Cast ray
  - In refracted direction
- Multiply by transparency coefficient (color)

Qualitative refraction

- From "Color and Light in Nature" by Lynch and Livingston

Refraction

Snell-Descartes Law

\[ n_1 \sin \theta_i = n_2 \sin \theta_r \]

<table>
<thead>
<tr>
<th>MATERIAL</th>
<th>INDEX OF REFRACTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>air/vacuum</td>
<td>1</td>
</tr>
<tr>
<td>water</td>
<td>1.33</td>
</tr>
<tr>
<td>glass</td>
<td>1.5</td>
</tr>
<tr>
<td>diamond</td>
<td>2.4</td>
</tr>
</tbody>
</table>

Note that \( n_1 \) is the negative of the incoming ray.
Note that \( I \) is the negative of the incoming ray.

\[
\frac{\sin \theta_i}{\sin \theta_t} = \frac{n_t}{n_i}
\]

\[
\frac{\cos \theta_i}{\cos \theta_t} = \frac{n_i}{n_t}
\]
**Example: Refraction**

![Refraction Example](image)

**Ray Tracing Example (with texture mapping)**

![Ray Tracing Example](image)

---

**Refraction and the lifeguard problem**

- Running is faster than swimming

![Lifeguard Problem](image)

---

**The Ray Tree**

![Ray Tree Diagram](image)

---

**Ray Tree**

![Ray Tree](image)

---

**Basic Ray Tracing: Notes**

- Intersection calculations are expensive, and even more so for more complex objects
  - Not currently suitable for real-time (i.e., games)
- Only global illumination effect is purely specular reflection/transmission
  - No "diffuse reflection" from other objects! Still using ambient term
  - One remedy is **radiosity** (slow, offline, precompute)
  - Ambient Occlusion
- Shadows have sharp edges, which is unrealistic – next lecture
Phong shading

Ambient Occlusion

Ambient Occlusion

Ray Tracing: Improvements

- Image quality: Anti-aliasing
  - Supersampling: Shoot multiple rays per pixel (grid or jittered)
    - Adaptive: More rays in areas where image is changing more quickly
- Efficiency: Bounding extents
  - Idea: Enclose complex objects in shapes (e.g., sphere, box) that are less expensive to test for intersection
  - Next lecture
Supersampling

• Rasterize at higher resolution
  - Regular grid pattern around each “normal” image pixel
  - Irregular jittered sampling pattern reduces artifacts
• Combine multiple samples into one pixel via weighted average
  - “Box” filter: All samples associated with a pixel have equal weight (i.e., directly take their average)
  - Gaussian/cone filter: Sample weights inversely proportional to distance from associated pixel

Adaptive Supersampling (Whitted’s method)

• Shoot rays through 4 pixel corners and collect colors
• Provisional color for entire pixel is average of corner contributions
  - If you stop here, the only overhead vs. center-of-pixel ray-tracing is another row, column of rays
• If any corner’s color is too different, subdivide pixel into quadrants and recurse on quadrants

Adaptive Supersampling: Details