Recursive ray tracing

H&B, page 597

Recursive ray tracing rendering algorithm

- Cast ray from pinhole (projection center) through pixel, determine nearest intersection
- Compute components by casting rays
  - to sources = shadow ray (diffuse and for specular lobe)
  - along reflected direction = reflected ray
  - along transmitted dir = refracted ray
- Determine each component and add them up with contribution from ambient illumination.
- To determine some of the components, the ray tracer must be called recursively.

Recursive ray tracing rendering (cont)

- Recursion needs to stop at some point!
- Contributions die down after multiple bounces---there is no such thing as a perfect reflector---so we either set mirror reflections to be less than 100% (even if the user asks for 100%), or simply include an attenuation factor for each new ray.
- Can also model absorption due to light traveling in medium
  - Usually ignored in air, but depends on the application
  - Translucent absorption is exponential in depth
    \[ I = I_0 e^{-\alpha d} \]
- Recursion is stopped when contributions are too small
  - need to track the cumulative effect
  - common to also limit the depth explicitly

Mechanics

- Primary issue is intersection computations.
  - E.g. sphere, triangle.
- Polygon (should feel familiar!)
- Find point on plane of polygon and then determine if it is inside
  - One way is to make an argument with angles
  - Another way---thinking of the polygon as a surface of a polyhedra---is to check if the point is on the inside side of each of the other planes of the polyhedra.
- Sphere, relatively simple algebra (see book page 602)
Mechanics

- Primary issue is intersection computations.
  - E.g. sphere, triangle.
- Polygon (should feel familiar!)
- Find point on plane of polygon and then determine if it is inside
  - One way is to make an argument with angles
  - Another way---thinking of the polygon as a surface of a polyhedra---is to check if the point is on the inside side of each of the other planes of the polyhedra.
- Sphere, relatively simple algebra (see book page 602)

Poly details

To find the intersection of the ray and the plane, solve:

\[
(P_0 + su - P_p) \cdot n = 0
\]

Once you have the point of intersection, \( P_i \), test that it is inside by testing against all other faces.

\[
(P_i - P_p) \cdot n < 0
\]

Note that \( n \) and \( P_p \) are now from those other faces.

Sphere details (H&B, 602)

\[
|P - P_c| = r
\]
Sphere details (H&B, 602)

\[
|\mathbf{P}_0 + \mathbf{s}\mathbf{u} - \mathbf{P}_c| = r
\]

\[
|\Delta\mathbf{P} + \mathbf{s}\mathbf{u}| = r
\]

\[
(\Delta\mathbf{P} + \mathbf{s}\mathbf{u}) \cdot (\Delta\mathbf{P} + \mathbf{s}\mathbf{u}) = r^2
\]

\[
\Delta\mathbf{P} \cdot \Delta\mathbf{P} - r^2 + 2s\Delta\mathbf{P} \cdot \mathbf{u} + s^2\mathbf{u} \cdot \mathbf{u} = 0
\]

The last expression is easily solved using the quadratic equation. If the discriminant is negative (complex solutions), then the ray does not intersect the sphere.

Recall that if:

\[as^2 + bs + c = 0\]

The “discriminant” is:

\[b^2 - 4ac\]

The solution is:

\[s = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}\]

Note that in the book, \(\mathbf{u}\) is a unit vector, so \(\mathbf{u} \cdot \mathbf{u}=1\), thus \(a=1\), and \(b\) has a factor of 2 that is removed by dividing by \(2a=2\), to get equation 10-71.

Refraction Details

Index of refraction, \(n\), is the ratio of speed of light in a vacuum, to speed of light in medium.

Typical values:

- air: 1.00 (nearly)
- water: 1.33
- glass: 1.45-1.6
- diamond: 2.2

The indicies of refraction for the two media, and the incident angle, \(\theta_i\), yield the refracted angle \(\theta_r\).

\[n_i \sin \theta_i = n_r \sin \theta_r\]