Ray Tracing

Outline
• Ray tracing algorithm
• Global illumination model
• Acceleration techniques
  • Bounding volume hierarchy
  • Spatial partitions
    • Uniform grids
    • Octrees

Ray Casting vs. Ray Tracing

How to Ray Trace
• Shoot a ray through each of the pixels
• Find the closest intersection
  • Conditionally shoot a reflection ray & a transmission ray from the intersection
  • Shoot one shadow ray from the intersection to each light source
• Compute intensity using global illumination

Recursive Process

Sampling the Rendering Equation
• In rendering equation, lights coming from all directions contribute to the final color
  \[ L_r(x, \omega_i) = L_r(x, \omega_i) + \int f_r(x, \omega_i \rightarrow \omega_o) I_r(x, \omega_o) \cos \theta d\omega_o \]
• The ray tracing algorithm only considers several important directions:
  • Directions pointing to light sources
  • Reflection direction
  • Transmission direction
  • Can handle light path of form LDS*E
Global Illumination Model

- \[ I = I_1 + k_r I_r + k_t I_t \]
- \( I_1 \): Intensity calculated using local illumination model
- \( I_r \): Intensity of the reflection ray
- \( k_r \): Reflection coefficient
- \( I_t \): Intensity of the transmission ray
- \( k_t \): Transmission coefficient

In practice:
- Phong model is used as local illumination model
- Reflection coefficient equals specular-reflection coefficient: \( k_r = k_s \)

Algorithm

- \[ \text{bool RayTrace(Ray V, int step, float weight, Color I)} \]
  - \[ \text{if ( step>maxStep || weight<minWeight )} \]
  - \[ \text{return false;} \]
  - \[ \text{if (! RayCast(V, intersection))} \]
  - \[ \text{return false;} \]
  - \( I_1 = \) local illumination result;
  - \( \text{Ray R = getReflectionRay(intersection, V)} \);
  - \( \text{RayTrace(R, step+1, kr*weight, Ir)} \);
  - \( \text{Ray T = getTransmissionRay(intersection, V)} \);
  - \( \text{RayTrace(T, step+1, kt*weight, It)} \);
  - \( I = I_1 + k_r I_r + k_t I_t \);
  - \( \text{return true;} \)

Mirror Reflection

- The reflection of other objects can be seen on shiny surfaces:
  - Mirror, metal, etc.
- Reflection direction:
  - \( R = 2(N \cdot V)N - V \)

Transmission (Refraction)

- Light bends when entering new material
- Due to the speeds of light differences in different materials
- Refractive index:
  - The material's index number is \( n \) if the speed of light is \( c/n \) when traveling in the material
  - \( c \) is the speed of light in a vacuum
  - The material's index number is different for light of different wavelength
  - The cause of rainbows

Refraction Direction

- Snell's law:
  - When light passes from a material of index \( n_1 \) to a material of index \( n_2 \):
  - \( n_1 \sin \theta_1 = n_2 \sin \theta_2 \)
  - The angle is smaller in denser material
- Index numbers:
  - Air or vacuum: 1
  - Water: 1.33
  - Glass: 1.50
  - Diamond: 2.42

Acceleration Techniques

- Rendering a scene using ray tracing requires millions of ray-surface intersection calculation
- An image may contain millions of pixels, each requires at least one testing ray or more if anti-aliasing is used
- At each intersection location, a number shadow testing rays is needed
- On reflective and/or transparent surfaces, 1 or 2 more testing rays are generated at each intersection
- Acceleration is necessary for rendering complex scenes within reasonable time
Bounding Volume
- Objective is to cut the cost for ray-object intersection calculation:
  - Use a bounding volume to enclose an object
    - If ray does not intersect the volume, it does not intersect object itself
  - Simple shapes should be used as bounding volumes:
    - Sphere or box

Bounding Volume Hierarchy
- Build hierarchy of bounding volumes
  - If ray does not intersect a volume, it does not intersect any child volume
  - The objects are organized bottom-up

Uniform Partition
- Subdivide the virtual world top-down:
  - Divide the bounding box of the world into equal-sized cells
  - Each cell contains a list of objects that it wholly or partially contains

Uniform Partition (Cont’d)
- Trace ray in uniform partition:
  - Find the next cell that the ray passes using modified line-drawing algorithm
  - Test objects that are linked the cell
  - The first intersection may not be the closest
  - Need to test if the intersection found is inside the current cell

Octree Partition
- Subdivide the virtual world top-down:
  - Divide the bounding box adaptively
  - Divide a cell that contain more than one objects into 8 octants
- Comparison with uniform partition:
  - Fewer cells, save memory
  - More computation for finding neighbour cells