Ray Casting II
Last Time?

- Ray Casting / Tracing
- Orthographic Camera
- Ray Representation
  - \( P(t) = \text{origin} + t \times \text{direction} \)
- Ray-Sphere Intersection
- Ray-Plane Intersection
- Implicit vs. Explicit Representations
Ray Casting: Object oriented design

For every pixel
   Construct a ray from the eye
   For every object in the scene
      Find intersection with the ray
      Keep if closest
Object-Oriented Design

- We want to be able to add primitives easily
  - Inheritance and virtual methods
- Even the scene is derived from Object3D!

```
Object3D
  bool intersect(Ray, Hit, tmin)

Plane
  bool intersect(Ray, Hit, tmin)

Sphere
  bool intersect(Ray, Hit, tmin)

Triangle
  bool intersect(Ray, Hit, tmin)

Group
  bool intersect(Ray, Hit, tmin)
```
Assignment 4 & 5: Ray Casting/Tracing

- Write a basic ray caster
  - Orthographic camera
  - Sphere Intersection
  - Main loop rendering
  - 2 Display modes: color and distance

- We provide:
  - Ray: origin, direction
  - Hit: t, Material, (normal)
  - Scene Parsing
Books

• Peter Shirley
  *Fundamentals of Computer Graphics*
  AK Peters

• Ray Tracing
Questions?
Ray-Triangle Intersection

- Use ray-plane + in-triangle test
- Use general ray-polygon
- Or try to be smarter
  - Use barycentric coordinates
Barycentric Definition of a Plane

- \( P(\alpha, \beta, \gamma) = \alpha a + \beta b + \gamma c \)
  with \( \alpha + \beta + \gamma = 1 \)

- Is it explicit or implicit?

\( P \) is the *barycenter*: the single point upon which the plane would balance if weights of size \( \alpha, \beta, \& \gamma \) are placed on points \( a, b, \& c \).
Barycentric Definition of a Triangle

- $P(\alpha, \beta, \gamma) = \alpha a + \beta b + \gamma c$
  with $\alpha + \beta + \gamma = 1$
- AND $0 < \alpha < 1 \; \& \; 0 < \beta < 1 \; \& \; 0 < \gamma < 1$
How Do We Compute $\alpha$, $\beta$, $\gamma$?

- Ratio of opposite sub-triangle area to total area
  - $\alpha = A_a/A$  $\beta = A_b/A$  $\gamma = A_c/A$
- Use signed areas for points outside the triangle
Intuition Behind Area Formula

- P is barycenter of a and Q
- $A_a$ is the interpolation coefficient on $aQ$
- All points on lines parallel to $bc$ have the same $\alpha$ (All such triangles have same height/area)
Simplify

- Since $\alpha + \beta + \gamma = 1$, we can write $\alpha = 1 - \beta - \gamma$

$$P(\alpha, \beta, \gamma) = \alpha a + \beta b + \gamma c$$

$$P(\beta, \gamma) = (1 - \beta - \gamma)a + \beta b + \gamma c$$

$$= a + \beta(b-a) + \gamma(c-a)$$
Questions?
Intersection with Barycentric Triangle

- Set ray equation equal to barycentric equation

\[ P(t) = P(\beta, \gamma) \]

\[ R_o + t \cdot R_d = a + \beta(b-a) + \gamma(c-a) \]

- Intersection if \( \beta + \gamma < 1 \) & \( \beta > 0 \) & \( \gamma > 0 \) (and \( t > t_{\text{min}} \) … )
Intersection with Barycentric Triangle

• \( \mathbf{R}_o + t \mathbf{R}_d = \mathbf{a} + \beta(\mathbf{b}-\mathbf{a}) + \gamma(\mathbf{c}-\mathbf{a}) \)

\[
\begin{align*}
\mathbf{R}_{ox} + t\mathbf{R}_{dx} &= a_x + \beta(b_x-a_x) + \gamma(c_x-a_x) \\
\mathbf{R}_{oy} + t\mathbf{R}_{dy} &= a_y + \beta(b_y-a_y) + \gamma(c_y-a_y) \\
\mathbf{R}_{oz} + t\mathbf{R}_{dz} &= a_z + \beta(b_z-a_z) + \gamma(c_z-a_z)
\end{align*}
\]

\[
\begin{bmatrix}
  a_x - b_x & a_x - c_x & R_{dx} \\
  a_y - b_y & a_y - c_y & R_{dy} \\
  a_z - b_z & a_z - c_z & R_{dz}
\end{bmatrix}
\begin{bmatrix}
  \beta \\
  \gamma \\
  t
\end{bmatrix} =
\begin{bmatrix}
  a_x - R_{ox} \\
  a_y - R_{oy} \\
  a_z - R_{oz}
\end{bmatrix}
\]

3 equations, 3 unknowns
Cramer’s Rule

- Used to solve for one variable at a time in system of equations

\[
\beta = \frac{a_x - R_{ox}}{|A|} \quad \gamma = \frac{a_x - b_x}{|A|} \\
\gamma = \frac{a_y - R_{oy}}{|A|} \quad t = \frac{a_y - b_y}{|A|} \\
\gamma = \frac{a_z - R_{oz}}{|A|} \quad t = \frac{a_z - b_z}{|A|}
\]

|   | denotes the determinant

Can be copied mechanically into code
Advantages of Barycentric Intersection

• Efficient
• Stores no plane equation
• Get the barycentric coordinates for free
  – Useful for interpolation, texture mapping
Another interpretation

• Ray-plane intersection with parametric equation of plane
• Harder than with implicit
  – more unknown: parameters
• But provides more info
  – Easy test if the intersection is inside triangle
  – interpolation parameters (barycentric coordinates)
Questions?

- Image computed using the RADIANCE system by Greg Ward
Constructive Solid Geometry (CSG)

Given overlapping shapes A and B:

- **Union**
- **Intersection**
- **Subtraction**
For example:
How can we implement CSG?

Points on A, Outside of B
Points on B, Inside of A
Points on B, Outside of A
Points on A, Inside of B

Union

Intersection

Subtraction
Collect all the intersections
Implementing CSG

1. Test "inside" intersections:
   - Find intersections with A, test if they are inside/outside B
   - Find intersections with B, test if they are inside/outside A

2. Overlapping intervals:
   - Find the intervals of "inside" along the ray for A and B
   - Compute union/intersection/subtraction of the intervals
My early CSG raytraced Image
Questions?
Precision

• What happens when
  – Origin is on an object?
  – Grazing rays?

• Problem with floating-point approximation
The evil $\varepsilon$

- In ray tracing, do NOT report intersection for rays starting at the surface (no false positive)
  - Because secondary rays
  - Requires epsilons
The evil $\varepsilon$: a hint of nightmare

- Edges in triangle meshes
  - Must report intersection (otherwise not watertight)
  - No false negative
Questions?

Image by Henrik Wann Jensen
New topic: transformations

- We have seen that transformations such as affine transforms are useful for modeling & animation
- How do we incorporate them into ray casting?
Incorporating Transforms

1. Make each primitive handle any applied transformations

   Sphere {
       center 1 0.5 0
       radius 2
   }

2. Transform the Rays

   Transform {
       Translate { 1 0.5 0 }
       Scale { 2 2 2 }
       Sphere {
           center 0 0 0
           radius 1
       }
   }
Primitives handle Transforms

Sphere {
    center 3 2 0
    z_rotation 30
    r_major 2
    r_minor 1
}

- Complicated for many primitives
Transform the Ray

• Move the ray from *World Space* to *Object Space*

\[ p_{WS} = M \quad p_{OS} \]

\[ p_{OS} = M^{-1} \quad p_{WS} \]
Questions?
Transform Ray

- New origin:
  \[ \text{origin}_{OS} = M^{-1} \text{origin}_{WS} \]

- New direction:
  \[ \text{direction}_{OS} = M^{-1} (\text{origin}_{WS} + 1 \times \text{direction}_{WS}) - M^{-1} \text{origin}_{WS} \]
  \[ \text{direction}_{OS} = M^{-1} \text{direction}_{WS} \]
Transforming Points & Directions

• Transform point

\[
\begin{bmatrix}
x' \\
y' \\
z' \\
1
\end{bmatrix} =
\begin{bmatrix}
a & b & c & d \\
e & f & g & h \\
i & j & k & l \\
0 & 0 & 0 & 1
\end{bmatrix}
\begin{bmatrix}
x \\
y \\
z \\
1
\end{bmatrix} =
\begin{bmatrix}
ax+by+cz+d \\
ex+f y+gz+h \\
ix+jy+kz+l \\
1
\end{bmatrix}
\]

• Transform direction

\[
\begin{bmatrix}
x' \\
y' \\
z' \\
0
\end{bmatrix} =
\begin{bmatrix}
a & b & c & d \\
e & f & g & h \\
i & j & k & l \\
0 & 0 & 0 & 1
\end{bmatrix}
\begin{bmatrix}
x \\
y \\
z \\
0
\end{bmatrix} =
\begin{bmatrix}
ax+by+cz \\
ex+f y+gz \\
ix+jy+kz \\
0
\end{bmatrix}
\]

Homogeneous Coordinates: \((x,y,z,w)\)
\(w = 0\) is a point at infinity (direction)

• With the usual storage strategy (no \(w\)) you need different routines to apply \(M\) to a point and to a direction
Quiz

• 1: Basis was \((1, t, t^2, t^3)\), not \((t, t^2, t^3, 1)\)
• 2.2: there was a 1 missing in the third row, the scene was flattened. Did not take points off if you missed it (I had forgotten it)
No scale, homogeneous coordinate is 2 as well
• 3.2 need \(h<-1/a\) and \(h<-1/b\), (note the minus sign)

• mean 42.22222222
• std dev 6.96567548