Ray-Surface Intersections:
- Necessary in ray tracing
- General implicit surfaces
- General parametric surfaces
- Specialized analysis for special surfaces
  - Spheres
  - Planes
  - Polygons
  - Quadrics

Intersection of Rays and Parametric Surfaces:
- Ray in parametric form
  - Origin \( \mathbf{p}_0 = [x_0, y_0, z_0]^T \)
  - Direction \( \mathbf{d} = [x_d, y_d, z_d]^T \)
  - Assume \( \mathbf{d} \) is normalized \( (x_d^2 + y_d^2 + z_d^2 = 1) \)
  - Ray \( \mathbf{p}(t) = \mathbf{p}_0 + \mathbf{d} t \) for \( t > 0 \)
- Surface in parametric form
  - Point \( \mathbf{q} = g(u, v) \), possible bounds on \( u, v \)
  - Solve \( \mathbf{p}(t) = \mathbf{q} \)
  - Three equations in three unknowns \( (t, u, v) \)

Intersection of Rays and Implicit Surfaces:
- Ray in parametric form
  - Origin \( \mathbf{p}_0 = [x_0, y_0, z_0]^T \)
  - Direction \( \mathbf{d} = [x_d, y_d, z_d]^T \)
  - Assume \( \mathbf{d} \) normalized \( (x_d^2 + y_d^2 + z_d^2 = 1) \)
  - Ray \( \mathbf{p}(t) = \mathbf{p}_0 + \mathbf{d} t \) for \( t > 0 \)
- Implicit surface
  - Given by \( f(\mathbf{q}) = 0 \)
  - Consists of all points \( \mathbf{q} \) such that \( f(\mathbf{q}) = 0 \)
  - Substitute ray equation for \( \mathbf{q} \): \( f(\mathbf{p}_0 + \mathbf{d} t) = 0 \)
  - Solve for \( t \) (univariate root finding)
    - Closed form (if possible), otherwise numerical approximation

Ray-Sphere Intersection I:
- Common and easy case
- Define sphere by
  - Center \( \mathbf{c} = [x_c, y_c, z_c]^T \)
  - Radius \( r \)
  - Surface \( f(\mathbf{q}) = (x-x_c)^2 + (y-y_c)^2 + (z-z_c)^2 - r^2 = 0 \)
- Plug in ray equations for \( x, y, z \):
  - \( x = x_0 + x_d t \), \( y = y_0 + y_d t \), \( z = z_0 + z_d t \)
- And we obtain a scalar equation for \( t \):
  - \( (x_0 + x_d t - x_c)^2 + (y_0 + y_d t - y_c)^2 + (z_0 + z_d t - z_c)^2 = r^2 \)

Ray-Sphere Intersection II:
- Simplify to
  - \( at^2 + bt + c = 0 \)
  - where \( a = x_d^2 + y_d^2 + z_d^2 = 1 \) since \( |\mathbf{d}| = 1 \)
  - \( b = 2(x_d(x_0 - x_c) + y_d(y_0 - y_c) + z_d(z_0 - z_c)) \)
  - \( c = (x_0 - x_c)^2 + (y_0 - y_c)^2 + (z_0 - z_c)^2 - r^2 \)
- Solve to obtain \( t_0 \) and \( t_1 \)
  - \( t_{0,1} = \frac{-b \pm \sqrt{b^2 - 4ac}}{2} \)
  - Check if \( t_0, t_1 > 0 \) (ray)
  - Return \( \min(t_0, t_1) \)
Ray-Sphere Intersection III

- For lighting, calculate unit normal
  \[ n = \frac{1}{r} \left[ (x_i - x_c) \quad (y_i - y_c) \quad (z_i - z_c) \right] \]

- Negate if ray originates inside the sphere!
- Note possible problems with roundoff errors

Simple Optimizations

- Factor common subexpressions
- Compute only what is necessary
  - Calculate \( b^2 - 4c \), abort if negative
  - Compute normal only for closest intersection
  - Other similar optimizations

Ray-Quadric Intersection

- Quadric \( f(p) = f(x, y, z) = 0 \), where \( f \) is polynomial of order 2
- Sphere, ellipsoid, paraboloid, hyperboloid, cone, cylinder
- Closed form solution as for sphere
- Important case for modelling in ray tracing
- Combine with CSG

Ray-Polygon Intersection I

- Assume planar polygon in 3D
  1. Intersect ray with plane containing polygon
  2. Check if intersection point is inside polygon
- Plane
  - Implicit form: \( ax + by + cz + d = 0 \)
  - Unit normal: \( n = [a \ b \ c]^T \) with \( a^2 + b^2 + c^2 = 1 \)
- Substitute:
  \[ a(x_0 + x_d t) + b(y_0 + y_d t) + c(z_0 + z_d t) + d = 0 \]
- Solve:
  \[ t = \frac{-ax_0 - by_0 - cz_0 - d}{ax_d + by_d + cz_d} \]

Test if point inside polygon

- Use even-odd rule or winding rule
- Easier if polygon is in 2D (project from 3D to 2D)
- Easier for triangles (tessellate polygons)
**Point-in-triangle testing**

• Critical for polygonal models

• Project the triangle, and point of plane intersection, onto one of the planes $x = 0$, $y = 0$, or $z = 0$ (pick a plane not perpendicular to triangle) (such a choice always exists)

• Then, do the 2D test in the plane, by computing barycentric coordinates (follows next)

**Outline**

• Ray-Surface Intersections
• Special cases: sphere, polygon
• Barycentric Coordinates

---

**Interpolated Shading for Ray Tracing**

• Assume we know normals at vertices
• How do we compute normal of interior point?
• Need linear interpolation between 3 points
• Barycentric coordinates
• Yields same answer as scan conversion

**Barycentric Coordinates in 1D**

• Linear interpolation
  - $p(t) = (1-t)p_1 + tp_2$, $0 \leq t \leq 1$
  - $p(t) = \alpha p_1 + \beta p_2$, where $\alpha + \beta = 1$
  - $p$ is between $p_1$ and $p_2$ iff $0 \leq \alpha, \beta \leq 1$
• Geometric intuition
  - Weigh each vertex by ratio of distances from ends
• $\alpha, \beta$ are called barycentric coordinates

**Barycentric Coordinates in 2D**

• Now, we have 3 points instead of 2

• Define 3 barycentric coordinates, $\alpha, \beta, \gamma$
  - $p = \alpha p_1 + \beta p_2 + \gamma p_3$
  - $p$ inside triangle if $0 \leq \alpha, \beta, \gamma \leq 1$, $\alpha + \beta + \gamma = 1$
  - How do we calculate $\alpha, \beta, \gamma$ given $p$?

**Barycentric Coordinates for Triangle**

• Coordinates are ratios of triangle areas
  - $\alpha = \frac{\text{Area}(C_1C_2C)}{\text{Area}(C_1C_2C_3)}$
  - $\beta = \frac{\text{Area}(C_1C_3C_2)}{\text{Area}(C_1C_2C_3)}$
  - $\gamma = \frac{\text{Area}(C_2C_3C_1)}{\text{Area}(C_1C_2C_3)} = 1 - \alpha - \beta$
• Areas in these formulas should be signed, depending on clockwise (-) or anti-clockwise orientation (+) of the triangle! Very important for point-in-triangle test.
Computing Triangle Area in 3D

- Use cross product
- Parallelogram formula
- Area(ABC) = (1/2) |(B – A) x (C – A)|
- How to get correct sign for barycentric coordinates?
  - tricky, but possible: compare directions of vectors (B – A) x (C – A), for triangles C0;C1 vs C2;C1, etc.
  - easier alternative: project to 2D, use 2D formula
  - projection to 2D preserves barycentric coordinates

Computing Triangle Area in 2D

- Suppose we project the triangle to xy plane
- Area(xy-projection(ABC)) =
  (1/2) ((b_x – a_x)(c_y – a_y) – (c_x – a_x) (b_y – a_y))
- This formula gives correct sign (important for barycentric coordinates)

Summary

- Ray-Surface Intersections
- Special cases: sphere, polygon
- Barycentric Coordinates

Class video, Programming Assignment 2