
CS770/870 Fall 2008

Ray Object Intersection

Related material in Hill and Kelley: Ch. 12

23:34

CS770/870 Fall 2008 Bergeron/Plumlee

1

Preview

- Ray-Object intersections
 - Spheres
 - Plane
 - Polygon
 - Box
 - Slab
 - Polyhedron
 - Quadric surfaces
 - Other surfaces

23:34

CS770/870 Fall 2008 Bergeron/Plumlee

2

Ray-Sphere Definitions

- *Implicit* representation of a sphere of radius, r , and center $C = (x_c, y_c, z_c)$:

$$(x - x_c)^2 + (y - y_c)^2 + (z - z_c)^2 = r^2$$

- *Parametric* representation of a ray

$$R(t) = R_o + tV \quad \text{for } t > 0$$

where $R_o = (x_o, y_o, z_o)$ is the ray *origin*

and $V = (x_v, y_v, z_v)$ is the ray *direction* as a *unit vector*

This is really 3 equations:

$$x(t) = x_o + t x_v$$

$$y(t) = y_o + t y_v$$

$$z(t) = z_o + t z_v$$

23:34

CS770/870 Fall 2008 Bergeron/Plumlee

3

Intersection Equation

- Substitute $x(t)$, $y(t)$, $z(t)$ from ray into sphere equation and use the quadratic formula to solve for t .

$$(x_o + tx_v - x_c)^2 + (y_o + ty_v - y_c)^2 + (z_o + tz_v - z_c)^2 = r^2$$

- Reorganize terms

$$(x_o - x_c + x_v t)^2 + (y_o - y_c + y_v t)^2 + (z_o - z_c + z_v t)^2 = r^2$$

- Expand square terms

$$(x_o - x_c)^2 + 2(x_o - x_c)x_v t + x_v^2 t^2 + (y_o - y_c)^2 + 2(y_o - y_c)y_v t + y_v^2 t^2 + (z_o - z_c)^2 + 2(z_o - z_c)z_v t + z_v^2 t^2 = r^2$$

- Simplify $at^2 + bt + c = 0$

- where $a = x_v^2 + y_v^2 + z_v^2 = 1$ because V is a unit vector

$$b = 2(x_o - x_c)x_v + 2(y_o - y_c)y_v + 2(z_o - z_c)z_v$$

$$c = (x_o - x_c)^2 + (y_o - y_c)^2 + (z_o - z_c)^2 - r^2$$

23:34

CS770/870 Fall 2008 Bergeron/Plumlee

4

Solve quadratic equation

- Get two solutions:

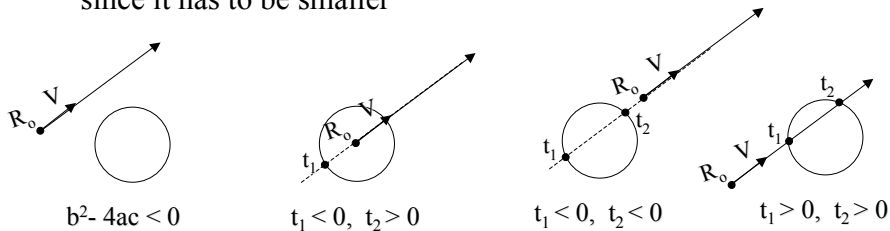
$b^2 - 4c < 0$ means no intersections

$t_i < 0$ means intersection on
“wrong” side of ray

$t_1 > 0$ and $t_2 > 0$ means use t_1
since it has to be smaller

$$t_1 = \frac{-b - \sqrt{b^2 - 4c}}{2} = -\frac{b}{2} - \sqrt{\frac{1}{4}b^2 - c}$$

$$t_2 = \frac{-b + \sqrt{b^2 - 4c}}{2} = -\frac{b}{2} + \sqrt{\frac{1}{4}b^2 - c}$$



23:34

CS770/870 Fall 2008 Bergeron/Plumlee

5

Efficiency Improvement

- Can we find “miss” cases more efficiently

- First find if R_o is inside or outside the sphere
 - if distance from R_o to $C < r$, then R_o is inside sphere

$$\text{distance}(C, R_o) = \sqrt{(C - R_o) \cdot (C - R_o)}$$

- But, no need to compute the square root

R_o is inside sphere if $(C - R_o) \cdot (C - R_o) - r^2 < 0$

$$\text{But, } (C - R_o) \cdot (C - R_o) = (x_c - x_o)^2 + (y_c - y_o)^2 + (z_c - z_o)^2$$

- So R_o is inside sphere if $c < 0$, for the c term of the quadratic equation.

23:34

CS770/870 Fall 2008 Bergeron/Plumlee

6

Efficiency Improvement (cont)

- Now, find point, t_c , on the ray closest to center, C

– Project $(C - R_o)$ onto V : $(C - R_o) \cdot V = t_c$

– $(C - R_o) \cdot V = -b/2$ -- the b from quadratic equation

- If $t_c > 0$, find d , distance from C to ray

$$L^2 = t_c^2 + d^2$$

$$\text{so, } d^2 = L^2 - t_c^2$$

$$\text{or } d^2 = (C - R_o) \cdot (C - R_o) - t_c^2$$

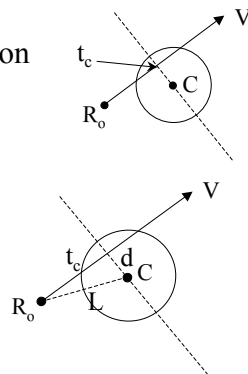
and $d^2 - r^2 > 0$ means ray misses sphere

or $(C - R_o) \cdot (C - R_o) - t_c^2 - r^2 > 0$ means miss

But $c = (C - R_o) \cdot (C - R_o) - r^2$ and $t_c^2 = (\frac{1}{2}b)^2$

for b and c from quadratic equation:

so condition is $c - (\frac{1}{2}b)^2 > 0 \equiv c - \frac{1}{4}b^2 > 0 \equiv 4c - b^2 > 0$



23:34

CS770/870 Fall 2008 Bergeron/Plumlee

7

Efficiency Improvement (cont)

- How does this help?

1. Calculate $c = (C - R_o) \cdot (C - R_o) - r^2$

2. Calculate $t_c = (C - R_o) \cdot V$

3. If $c > 0$ and $t_c < 0$, done; ray is
outside and points away

4. Calculate $h^2 = t_c^2 - c = \frac{1}{4}b^2 - c$

5. if $h^2 < 0$, done, no int.

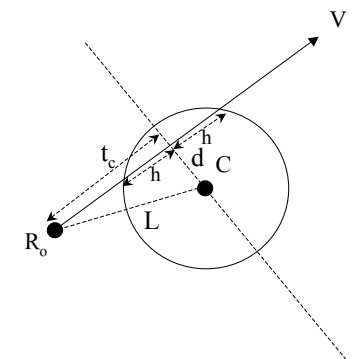
6. Calculate $h = \text{sqrt}(h^2)$

7. if $c > 0$, R_o outside: $t = t_c - h$

else if $c < 0$, R_o inside: $t = t_c + h$

8. Use t to get x, y, z

- If $C = (0, 0, 0)$, even simpler



23:34

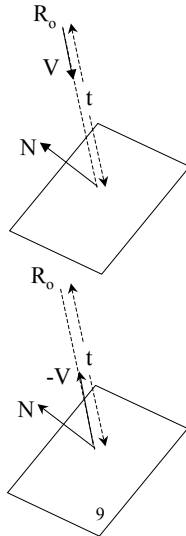
CS770/870 Fall 2008 Bergeron/Plumlee

8

Ray/Plane Intersections

- Ray:
 $R(t) = R_o + tV$ for $t > 0$
- Plane (implicit)
 $ax + by + cz + d = 0$ where $a^2 + b^2 + c^2 = 1$
 Note: $N = (a\ b\ c)$ is unit normal to plane
- Substitute ray equations into plane equation
 $a(x_o + tx_v) + b(y_o + ty_v) + c(z_o + tz_v) + d = 0$
 Solve for t

$$t = \frac{-(ax_o + by_o + cz_o + d)}{ax_v + by_v + cz_v} = \frac{N \cdot R_o + d}{N \cdot (-V)}$$

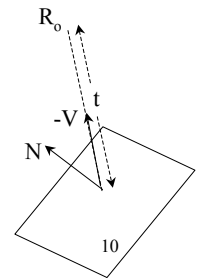
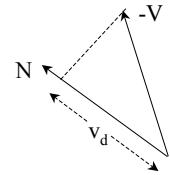


23:34

CS770/870 Fall 2008 Bergeron/Plumlee

Ray/Plane Intersections Steps

1. Calculate $v_d = N \cdot (-V)$
 $v_d == 0$ means ray is parallel to plane; no int.
 $v_d < 0$ normal in opposite direction from ray;
 if have 1-sided polygons and back-face culling, done: polygon not visible!
2. Calculate $v_o = (N \cdot R_o + d)$
3. Calculate $t = v_o / v_d$
 $t < 0$ means intersection “behind ray”,
 no int.
4. Calculate (x, y, z) from ray equations.

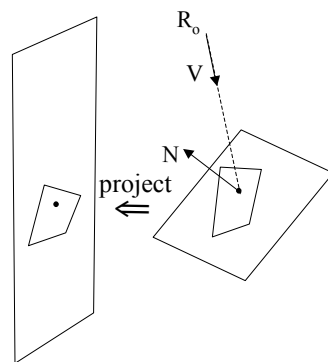


23:34

CS770/870 Fall 2008 Bergeron/Plumlee

Ray/Polygon Intersection

1. Find Ray/Plane intersection
2. Project polygon and point along one of major axes onto 2D plane.
3. Do *point-in-polygon* test to determine if intersection point is inside polygon



23:34

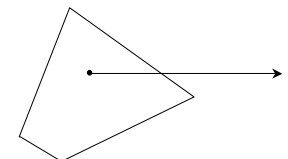
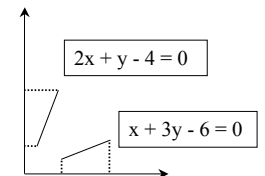
CS770/870 Fall 2008 Bergeron/Plumlee

11

Point-in-Polygon Test

1. Project 3D polygon along one coordinate axis to xy or yz or xz plane. Which is best?
 - want to avoid a thin projected polygon
 - to maximize projected area, project along axis with maximum coefficient in plane equation: see 2D analogy
2. 2D point-in-polygon
 - draw horiz (or vert) line from projected intersection point to infinity
 - count intersections with edges: odd means inside
 - take care with vertex intersections

2D analogy



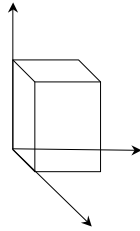
23:34

CS770/870 Fall 2008 Bergeron/Plumlee

12

Ray/Box Intersection

- Transform ray back to object space where box is oriented along major axes to origin and even unit cube
 - Intersect transformed ray with planes:
 - $x=0, x=1, y=0, y=1, z=0, z=1$
 - Line/plane intersections are trivial; intersection of ray with $x=0$:
 - $x(t) = x_0 + tx_v$ // ray equation for x
 - $t_0 = (0 - x_0) / x_v$ // intersection with $x=0$
 - The t for $x=1$ is just one more addition
 - $t_1 = (1 - x_0) / x_v = t_0 + 1 / x_v$



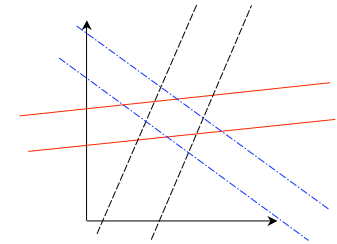
23:34

CS770/870 Fall 2008 Bergeron/Plumlee

13

Ray/Slab Intersection

- Slabs are pairs of parallel planes; they are good for creating tighter bounding volumes
- Just two sides of a box and once get t for one slab, the t for parallel one is just one addition



23:34

CS770/870 Fall 2008 Bergeron/Plumlee

14

Ray/Polyhedron Intersection

- A convex polyhedron can be defined by the intersection of half-planes (rather than by polygonal faces)

2D analogy

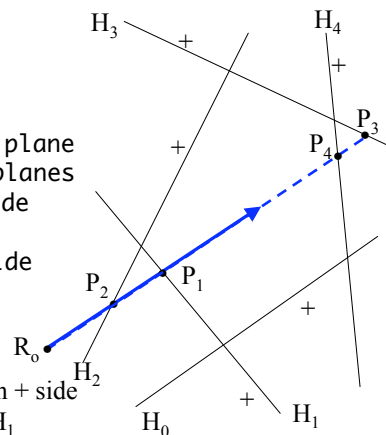
for each plane, H_i

P_i = intersection of ray with plane
if P_i on - side of all other planes
if R_0 and P_i are on same side
return P_i

// Note: P_i can be in or outside
return null

P_1 is the right intersection point

P_0 doesn't exist, P_2 is on + side of H_1 , P_3 is on + side of H_4 , P_4 and R_0 are on opposite sides of H_1



23:34

CS770/870 Fall 2008 Bergeron/Plumlee

15

Ray v. Quadric Surface

- Any quadric surface can be represented implicitly by

$$\begin{bmatrix} x & y & z & 1 \end{bmatrix} \begin{bmatrix} a & b & c & d \\ e & f & g & h \\ i & j & k & l \\ m & n & o & p \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} = 0$$

- Substitute the parametric equations for the ray into implicit equation and solve for t using the quadratic formula
- Just like sphere, but much more algebra!

23:34

CS770/870 Fall 2008 Bergeron/Plumlee

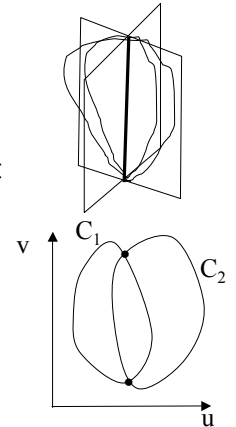
16

Ray v. Generic Implicit Surfaces

- Ray intersections with implicit surfaces of degree > 2 or non-polynomial surfaces are normally solved with numerical techniques

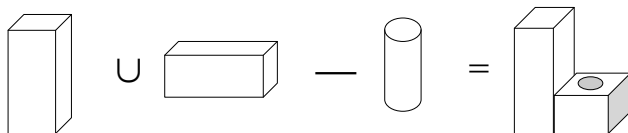
Ray vs. Explicit Surfaces

- Suppose the surface is defined parametrically (in 2 parameters):
 $S(u,v) = [x(u,v), y(u,v), z(u,v)]$
- Can make some progress by representing the ray as the intersection of any two planes that share it:
 $a_1x + b_1y + c_1z + d_1 = 0$
 $a_2x + b_2y + c_2z + d_2 = 0$
- Intersection of planes and surface yields 2 curves in 2D:
 $C_1(u,v) = a_1x(u,v) + b_1y(u,v) + c_1z(u,v) + d_1 = 0$
 $C_2(u,v) = a_2x(u,v) + b_2y(u,v) + c_2z(u,v) + d_2 = 0$
- Use numerical techniques in 2D to solve intersection in (u,v) space



CSG Surfaces

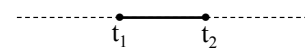
- Constructive Solid Geometry
 - Define objects as union, intersection or set difference between basic shapes, like parallelepiped and cylinders



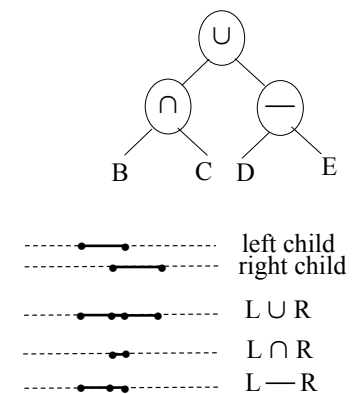
- Effective representation for analysis
 - validity of representation, many physical properties are easily computed, and much more
- Not very efficient for rendering purposes

Ray v. CSG

- Given a CSG tree
 - Do a postfix tree walk:
 - for each leaf
 - compute ray/solid intersection in t
 - // each primitive has 0 or 2 ints



- for each operator
 - combine ray *classification* of its children and *simplify* by merging



Other Kinds of Surfaces

- Sweep surfaces
- Procedural objects
 - parametric surfaces can be treated as procedural using subdivision techniques
 - Fractal surfaces
 - some can be handled by subdivision techniques
 - but, bounding volume test can be difficult
 - Particle systems and other true procedural surfaces have very specialized intersections
 - Volume data based on voxels: not very hard
 - Volume data based on octrees: also not hard