Intersections

Code considerations

class **Ray**:
Vector3f Q // Starting point
Vector3f D // Direction – should be scaled to a unit length vector
Consider a method **eval(t)** returning $Q + t\,D$

class **Shape**:
virtual method: **void intersect(Ray, Intersection&))**:
returns true (and fills Intersection structure) if ray intersects shape

classes **Sphere, Box, Cylinder, Triangle, …** (derived from **Shape**)
Each Shape's **intersect** returns a boolean and fills in an **Intersection** object

class **Intersection**
This records the intersection of a ray with any Shape object
t: Parameter value on ray of the point of intersection
object*: A pointer to the Shape intersected
P: Point of intersection (in world coordinates)
N: Normal of surface at intersection point (in world coordinates)
other: for instance 2D or 3D texture coordinates

class **Interval**
Represents an interval along a ray, useful for several later intersection calculations
t0, **t1**: beginning and ending point along a ray
N0, **N1**: Surface normals at **t0** and **t1** respectively.
Methods:
Constructor () initializes (t0, t1) to (0, INFINITY)
Constructor (t0,t1,N0,N1): Reorders t0,t1 (and N0,N1) so t0 <= t1, then stores all 4.
empty() Sets t0,t1 to 0,-1 to represent an empty interval
intersect(**other**): Modifies **this*** with intersection of **this*** and **other.**
Modifies N0,N1 in sync with t0 and t1.
intersect(ray and slab): forms interval by intersecting ray with slab,
and intersects with **this***

A general (easy) form of intersections

Surface in implicit form: $F(P)=0$
Ray in parametric form: $P = Q + t\,D$
Substitute to get $F(Q+t\,D)=0$ and solve for t (i.e., find roots).

There are many techniques for finding roots of such equations:
If F is a polynomial of degree n, so is $F(Q+t\,D)=0$
Linear and quadratic have simple solutions.
Higher degree polynomials and non-polynomials can be solved
numerically with iterative techniques (Newton's method and so on …)

Sphere

Parameters: center C, and radius r
Bounding box: from two points $C \pm (r, r, r)$.
Intersection:
The plan:

A point P is on a sphere if $length(P-C)=r$ or better $(P-C)\cdot(P-C) = r^2$
In implicit form: $F(P) = (P-C)\cdot(P-C) - r^2 = 0$
Substitute for P any point on the ray $P = Q + tD$.
Solve $F(Q+tD) = 0$ for the t of the intersection point.

Substitute ray: $F(Q+tD) = (Q + tD - C)\cdot(Q + tD - C) - r^2 = 0$

Simplify with $\bar{Q} = Q - C$: $(\bar{Q} + tD)\cdot(\bar{Q} + tD) - r^2 = 0$

Expand: $(D\cdot D)t^2 + 2(\bar{Q}\cdot D)t + (\bar{Q}\cdot\bar{Q} - r^2) = 0$

This is a quadratic of form $at^2 + bt + c = 0$ for $a = D\cdot D$, $b = 2(\bar{Q}\cdot D)$, $c = \bar{Q}\cdot\bar{Q} - r^2$

Solutions via quadratic eq: $t_{\pm} = \dfrac{-b \pm \sqrt{b^2 - 4ac}}{2a}$

Interpretation:

If discriminant<0 => No intersections
Calculate both roots: $t_- = -b - \ldots$ and $t_+ = -b + \ldots$
If both are negative => No intersections
Otherwise return the smallest positive value of the two t_- and t_+

The two roots can be calculated more efficiently than shown above:
Assume ray has been normalized to unit length (i.e., $D\cdot D = 1$)
Then $a = 1$
so

$$
\begin{aligned}
t &= \frac{-b \pm \sqrt{b^2 - 4c}}{2} \\
&= -\frac{b}{2} \pm \sqrt{\frac{b^2 - 4c}{4}} \\
&= -(\bar{Q}\cdot D) \pm \sqrt{(\bar{Q}\cdot D)^2 - \bar{Q}\cdot\bar{Q} + r^2}
\end{aligned}
$$

Other values:
Point : $P = Q + tD$ (That is, eval ray at t)
Normal: $N = P - C$, normalized to unit length

UV:
$$
\begin{aligned}
\theta &= atan2(N.Y, N.X) \\
\phi &= acos(N.Z) \\
(u, v) &= (\theta/2\pi, \phi/\pi)
\end{aligned}
$$

Plane equations:

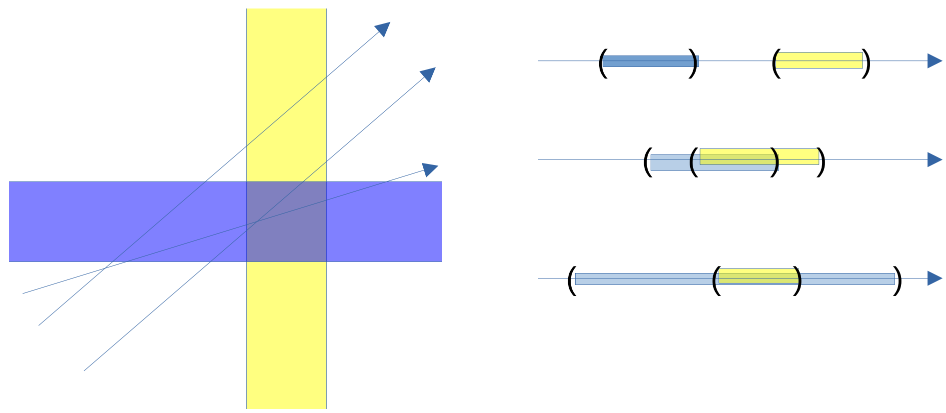
Usually $ax+by+cz+d=0$ for plane defined by coefficients a,b,c,d and a point $P=(x,y,z)$.
An equivalent but shorter notation: Let $N=(a,b,c)$
So the plane is represented by N,d, and the plane euqation is $N \cdot P + d = 0$.

Generalized shapes as intersections of slabs

slab: infinite volume bounded by two parallel planes
represented by two plane equations N, d_0 and N, d_1 sharing a normal

Intersection a ray with a collection of n slabs: broken into three steps:

1: Calculate the **interval** of a ray's intersection with each slab
2: Calculate intersection along the ray of n such intervals
3: Return the first (if any) positive endpoint of the resulting interval



Details:

1: Intersection of ray $Q + t\,D$ and slab $N,\ d_0,\ d_1$ returns an interval $t_0,\ t_1$

if $N \cdot D \neq 0$: (Ray intersects both slab planes)

Plane 0: $t_0 = -(d_0 + N \cdot Q) / N \cdot D$
Plane 1: $t_1 = -(d_1 + N \cdot Q) / N \cdot D$
Reorder (if needed) so that $t_0 \leq t_1$
Return interval $[t_0, t_1]$

else (ray is parallel to slab planes) so test if ray is inside the slab or not:

Calculate which side of each plane is the ray's starting point Q
$s_0 = N \cdot Q + d_0$
$s_1 = N \cdot Q + d_1$
If signs of s_0 and s_1 differ:
Q (and full ray) is between planes: return interval $[0, \infty]$
else
Q (and full ray) is outside planes: return **empty interval [1,0]**

2: Intersecting a list of slabs with a ray:

Initialize starting interval $[t_0,\ t_1] = [0, \infty]$ (indicating full ray)
for each slab $N^i,\ d_0^i,\ d_1^i$ (with superscript i indexing through the list of slabs):
if intersection of ray with slab i returns an interval $[t_0^i,\ t_1^i]$:
$$[t_0,\ t_1] = \left[max(t_0,\ t_0^i),\ min(t_1, t_1^i) \right]$$

3: Analyze final accumulated interval

If $t_0 > t_1$, then **NO-INTERSECTION** // This catches "off the corner" reversal
Else, return the smallest positive value of $t_0,\ t_1$ if any, or **NO-INTERSECTION**

AAB

Parameters: (x_0, y_0, z_0) and (dx, dy, dz) (corner and diagonal vector).

Bounding box: from two points (x_0, y_0, z_0) and (x_0+dx, y_0+dy, z_0+dz)

Intersection: Box is described as the intersection of three axis-aligned SLABS

The three slabs are

$(1,0,0), -x_0, -x_0-dx$

$(0,1,0), -y_0, -y_0-dy$

$(0,0,1), -z_0, -z_0-dz$

Other values:

Point : $P=Q+tD$ (That is, eval ray at t)

Normal: Easiest way: Attach to each t (as it is calculated), an associated normal. If you ensure that $d_0>d_1$ (as these notes do), then t_0's normal is $-N$ and t_1's normal is N .

UV: Not supported in generalized slab intersections, but you could rig up something reasonable for boxes.

Triangle

Parameters: Vertices V_0, V_1, and V_2, possibly with normals N_0, N_1, and N_2

Bounding box: From the three vertices V_0, V_1, and V_2.

Intersection:

Ray $Q + tD$

Let $E_1 = V_1 - V_0$, $E_2 = V_2 - V_0$, and $S = Q - V_0$

(That is, translate everything so V_0 goes to the origin.)

Parameterize triangle as $V_0 + uE_1 + vE_2$ (for $u \ge 0$, $v \ge 0$, $1 - v - u \ge 0$)

Equating a point on ray $Q + tD$ with a point on triangle $V_0 + uE_1 + vE_2$

gives a 3x3 system of equations: 1 equation from each XYZ coordinate, variables u, v, t

$Q + tD = V_0 + uE_1 + vE_2$, or

$-tD + uE_1 + vE_2 = S$

Cramer's rule can solve for each variable, t, u, v (using 4 3x3 determinants):

$$d = det(-D, E_1, E_2)$$
$$t = det(S, E_1, E_2)/d$$
$$u = det(-D, S, E_2)/d$$
$$v = det(-D, E_1, S)/d$$

Each of the determinants can be calculated with one cross and one dot product

(sometimes called the **triple product**)

(reordered to remove – on D, and reuse some cross-products)

$$det(-D, E_1, E_2) = (D \times E_2) \cdot E_1$$
$$det(S, E_1, E_2) = (S \times E_1) \cdot E_2$$
$$det(-D, S, E_2) = (D \times E_2) \cdot S$$
$$det(-D, E_1, S) = (S \times E_1) \cdot D$$

Algorithm:

$E_1 = V_1 - V_0$

$E_2 = V_2 - V_0$

$p = D \times E_2$

$d = p \cdot E_1$

if $d = 0$ return **NoIntersection** // Ray is parallel to triangle

$S = Q - V_0$

$u = (p \cdot S)/d$

if $u < 0$ or $u > 1$ return **NoIntersection** // Ray intersects plane, but outside E2 edge

$q = S \times E_1$

$v = (D \cdot q)/d$

if $v < 0$ or $(u + v) > 1$ return **NoIntersection** // Ray intersects plane, but outside other edges

$t = (E_2 \cdot q)/d$

if $t < 0$ return **NoIntersection** // Ray's negative half intersects triangle

return **Intersection, t**

Other values:

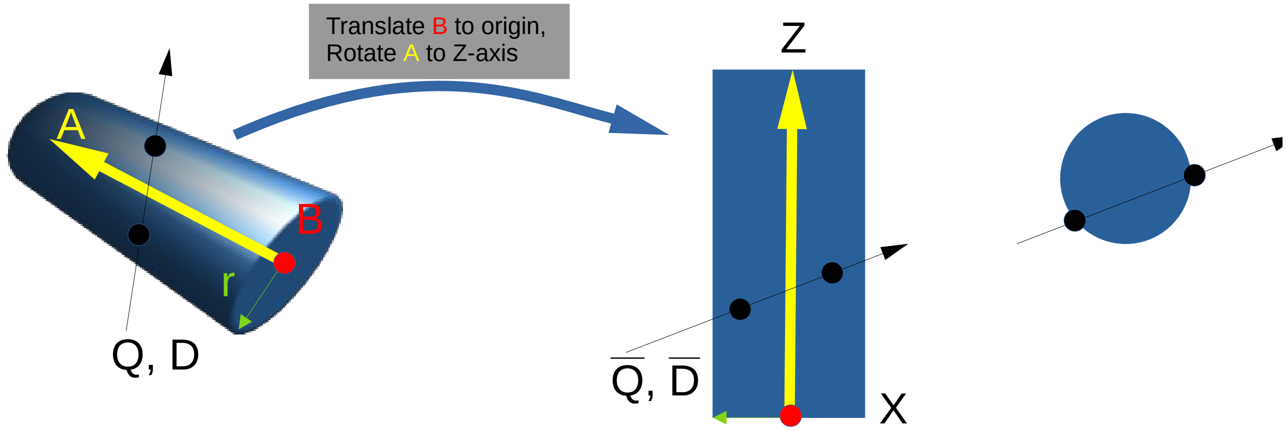
Point : $P = Q + tD$ (That is, eval ray at t)

Normal:

If vertex normals are known: $(1 - u - v)N_0 + uN_1 + vN_2$

else use triangle normal: $N = E_2 \times E_1$

UV: $(1 - u - v)T_0 + uT_1 + vT_2$, if vertices texture coordinates T_0, T_1, T_2 are known



Cylinder

Parameters: Base B, axis A, radius r (Cylinder goes from B to $B+A$.)

Bounding box: From 4 points (a sphere at each endpoint): $B\pm(r,r,r)$ and $A+B\pm(r,r,r)$

Plan:

Transform the problem to an easier space:

Translate $B \rightarrow (0,0,0)$

Rotate $A \rightarrow Z$

Applied to the ray:

$\bar{Q} = R_{A\rightarrow Z}(Q-B)$ (See separate document for derivation of $R_{A\rightarrow Z}$.)

$\bar{D} = R_{A\rightarrow Z}(D)$

Solve for intersection in this *easy* orientation

Back-transform the solution into the original space

Intersection with ray Q,D:

Transformed ray is: $(Q_x, Q_y, Q_z) = \bar{Q}$

$(D_x, D_y, D_z) = \bar{D}$

Algorithm:

Intersect two **intervals** on the ray

- First interval:
  $[a_0, a_1]$ = the ray's intersection with the end plate slab $N=(0,0,1)$, $d_0=0$, $d_1=-\|A\|$
  (Ray-intersect-slab is on a previous page.)
- Second interval: The ray's intersection with this z-aligned infinite cylinder: $x^2+y^2=r^2$

  Substitute transformed ray $(Q_x, Q_y, Q_z) + t\,(D_x, D_y, D_z)$ into $x^2+y^2-r^2=0$ to get
  $$t^2(D_x D_x + D_y D_y) + t\,2(D_x Q_x + D_y Q_y) + (Q_x Q_x + Q_y Q_y - r^2) = 0$$
  Which is in standard quadratic form $a t^2 + b t + c = 0$
  If discriminant $(b^2-4ac)<0$, **return NO-INTERSECTION**

  Calculate two roots with quadratic equation: $t_\pm = \left(-b\pm\sqrt{b^2-4ac}\right)/(2a)$
  Which gives interval: $[b_0, b_1] = (t_-, t_+)$

- Calculate intersection of two intervals
  $$[t_0, t_1] = [\,max(a_0, b_0),\ min(a_1, b_1)\,]$$

- If $t_0 > t_1$, then **NO-INTERSECTION** // The “off the corner” case
- Return smallest positive of t_0 or t_1 if any, or **return NO-INTERSECTION**

Other values:

Point : $P = Q + t\,D$ // Evaluate ray at t. (Note: The **ORIGINAL** not the **transformed** ray.)

Normal: Using the interval class,

calculate normals (an z-aligned space) along with intervals:

- 1st interval: No normals at $[a_0, a_1] = [0, \infty]$
- 2nd interval: Normals are $\bar{N} = (0, 0, \pm 1)$
- 3rd interval: For each intersection t,

$$\bar{N} = (Q_X + t\ D_X,\ Q_Y + t\ D_Y,\ 0)$$

Rotate $\bar{N}$ back to world with inverse rotation:

quaternion: conj(q)

matrix: $R^{-1} = R^T$

UV:
$$\theta = atan2(\bar{N}_y, \bar{N}_x)$$
$$(u, v) = (\theta / 2\pi,\ \bar{N}_z / \|A\|)$$