Subsections

Ray tracing primitives

Mathematical preliminaries

Coordinate systems

To deal easily with the various primitive objects, you need to be able to work in all three of the standard 3D coordinate systems: rectangular (x,y,z), spherical polar (r, θ, ϕ), and cylindrical polar (r, θ, z). To convert from one to another you use the following formulae.
Spherical polar to rectangular

$$x = r \cos\phi \cos\theta \quad (1)$$

$$y = r \cos\phi \sin\theta \quad (2)$$

$$z = r \sin\phi \quad (3)$$

Cylindrical polar to rectangular

$$x = r \cos\theta \quad (4)$$

$$y = r \sin\theta \quad (5)$$

$$z = z \quad (6)$$

Rectangular to spherical polar[1]

$$r = \sqrt{x^2 + y^2 + z^2} \quad (7)$$

$$\theta = \tan^{-1} y/x \tag{8}$$

$$\phi = \tan^{-1}\left(z/\sqrt{x^2+y^2}\right) \tag{9}$$

Rectangular to cylindrical polar

$$r = \sqrt{x^2+y^2} \tag{10}$$

$$\theta = \tan^{-1} y/x \tag{11}$$

$$z = z \tag{12}$$

Vector algebra

It is helpful to remember your vector arithmetic. A 3D vector is represented thus:

$$\mathbf{P} = \begin{bmatrix} x \\ y \\ z \end{bmatrix} \tag{13}$$

For ease of writing such definitions in text we may say $\mathbf{P} = (x, y, z)$, where we understand that this ordered triple is equivalent to the vector.
The magnitude of vector *P* is:

$$|\mathbf{P}| = \sqrt{x^2+y^2+z^2} \tag{14}$$

The dot product of two vectors ($\mathbf{A} = (x_A, y_A, z_A)$ and $\mathbf{B} = (x_B, y_B, z_B)$) is:

$$\mathbf{A} \cdot \mathbf{B} = x_A x_B + y_A y_B + z_A z_B \tag{15}$$

which could also be written as a matrix multiplication:

$$\mathbf{A} \cdot \mathbf{B} = \mathbf{A}^T \mathbf{B} = [x_A, y_A, z_A] \begin{bmatrix} x_B \\ y_B \\ z_B \end{bmatrix} \tag{16}$$

The dot product is a scalar value equal to:

$$\mathbf{A} \cdot \mathbf{B} = |A||B| \cos\theta \tag{17}$$

where θ is the angle between the two vectors. Note that this means that:

$$\mathbf{P} \cdot \mathbf{P} = |P|^2 \tag{18}$$

and hence:

$$|P| = \sqrt{\mathbf{P} \cdot \mathbf{P}} \tag{19}$$

Also note that the dot product between two mutually perpendicular vectors is always zero.
The cross product (vector product) of these two vectors is:

$$\mathbf{A} \times \mathbf{B} = \begin{bmatrix} y_A z_B - z_A y_B \\ z_A x_B - x_A z_B \\ x_A y_B - y_A x_B \end{bmatrix} \tag{20}$$

The cross product is a vector which is perpendicular to both $\mathbf{A}$ and $\mathbf{B}$. This is a very handy way to make a new vector which is guaranteed perpendicular to a given vector. It also means that:

$$\mathbf{A} \cdot (\mathbf{A} \times \mathbf{B}) = 0 \tag{21}$$

$$\mathbf{B} \cdot (\mathbf{A} \times \mathbf{B}) = 0 \tag{22}$$

though this may be getting a bit esoteric and so let's move on to something more interesting ...

Equation of a ray

A ray is defined by an origin or eye point, $\mathbf{E} = (x_E, y_E, z_E)$, and an offset vector, $\mathbf{D} = (x_D, y_D, z_D)$.

The equation for the ray is:

$$\mathbf{P}(t) = \mathbf{E} + t\mathbf{D}, t \geq 0 \tag{23}$$

This is obviously equivalent to the three equations:

$$\left.\begin{aligned} x(t) &= x_E + t x_D \\ y(t) &= y_E + t y_D \\ z(t) &= z_E + t z_D \end{aligned}\right\} t \geq 0 \tag{24}$$

When finding a ray-object intersection point, we are looking for the intersection point with the lowest non-negative value of t.

Equations for the primitives

Sphere

The unit sphere, centred at the origin, has the implicit equation:

$$x^2+y^2+z^2=1 \tag{25}$$

In spherical polar coordinates it is even simpler:

$$r=1 \tag{26}$$

In vector arthmetic, it becomes:

$$\mathbf{P} \cdot \mathbf{P} = 1 \tag{27}$$

To find the intersection between this sphere and an arbitrary ray, substitute the ray equation (Equation 24) in the sphere equation (Equation 25):

$$(x_E+tx_D)^2+(y_E+ty_D)^2+(z_E+tz_D)^2=1 \tag{28}$$

$$\Rightarrow\ t^2(x_D^2+y_D^2+z_D^2)+t(2x_Ex_D+2y_Ey_D+2z_Ez_D) +(x_E^2+y_E^2+z_E^2-1)=0 \tag{29}$$

$$\Rightarrow\ at^2+bt+c=0 \tag{30}$$

$$\Rightarrow\ t = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a} \tag{31}$$

where $a=x_D^2+y_D^2+z_D^2$, $b=2x_Ex_D+2y_Ey_D+2z_Ez_D$, and $c=x_E^2+y_E^2+z_E^2-1$. This gives zero, one, or two real values for t. If there are zero real values then there is no intersection between the ray and the sphere. If there are either one or two real values then chose the smallest, non-negative value, as the intersection point. If there is no non-negative value, then the line (of which the ray is a part) *does* intersect the sphere, but the intersection point is not on the part of the line which consistutes the ray. In this case there is again no intersection point between the ray and the sphere.

An alternative formulation is to use the vector versions of the equations (Equations 23 and 27):

$$(\mathbf{E} + t\mathbf{D}) \cdot (\mathbf{E} + t\mathbf{D}) = 1 \tag{32}$$

$$\Rightarrow\ t^2(\mathbf{D} \cdot \mathbf{D}) + t(2\mathbf{E} \cdot \mathbf{D}) + (\mathbf{E} \cdot \mathbf{E} - 1) = 0 \tag{33}$$

$$\Rightarrow\ at^2+bt+c=0 \tag{34}$$

$$\Rightarrow\ t = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a} \tag{35}$$

Where $a = \mathbf{D} \cdot \mathbf{D}$, $b = 2\mathbf{E} \cdot \mathbf{D}$, and $c = \mathbf{E} \cdot \mathbf{E} - 1$. In other words, exactly the same result, expressed in a more compact way. *Graphics Gems I* (p. 388) decribes yet another way of arriving at the same result.

Cylinder

The *infinite* unit cylinder aligned along the z-axis is defined as:

$$x^2+y^2=1 \tag{36}$$

In cylindrical polar coordinates it is just:

$$r=1 \tag{37}$$

To intersect a ray with this, substitute Equation 24 in Equation 36.

$$(x_E+tx_D)^2+(y_E+ty_D)^2=1 \tag{38}$$

$$\Rightarrow \quad t^2(x_D^2+y_D^2)+t(2x_Ex_D+2y_Ey_D) +(x_E^2+y_E^2-1)=0 \tag{39}$$

$$\Rightarrow \quad at^2+bt+c=0 \tag{40}$$

$$\Rightarrow \quad t=\frac{-b\pm\sqrt{b^2-4ac}}{2a} \tag{41}$$

where $a=x_D^2+y_D^2$, $b=2x_Ex_D+2y_Ey_D$, and $c=x_E^2+y_E^2-1$.

The *finite* open-ended unit cylinder aligned along the z-axis is defined as:

$$x^2+y^2=1, z_{\min}<z<z_{\max} \tag{42}$$

The only difference between this and Equation 36 being the restriction on z. In cylindrical polar coordinates this is obviously:

$$r=1, z_{\min}<z<z_{\max} \tag{43}$$

To handle this finite length cylinder, solve Equation 41 above. This gives, at most, two values of t. Call these t_1 and t_2. Calculate z_1 and $z_2$using Equation 24 ($z_1=z_E+t_1z_D$ and $z_2=z_E+t_2z_D$) and then check $z_{\min}<z_1<z_{\max}$ and $z_{\min}<z_2<z_{\max}$. Whichever intersection point passes this test and, if both pass the test, has the smallest non-negative value of t, is the closest intersection point of the ray with the open-ended finite cylinder.

If we wish the finite length cylinder to be closed we must formulate an intersection calculation between the ray and the cylinder's end caps. The end caps have the formulae:

$$z=z_{\min}, \quad x^2+y^2\le 1 \tag{44}$$

$$z=z_{\max}, \quad x^2+y^2\le 1 \tag{45}$$

Once you have calculated the solutions to Equation 41 you will either know that there are no intersections with the infinite cylinder or you will know that there are one or two real intersection points (t_1 and t_2). The previous paragraph explained how to ascertain whether these correspond to points on the finite length open-ended cylinder. Now, if z_1 and z_2 lie either side of $z_{\min}$ we know that the ray intersects the $z_{\min}$ end cap, and can calculate the intersection point as:

$$t_3 = \frac{z_{\min} - z_E}{z_D} \tag{46}$$

A similar equation holds for the $z_{\max}$ end cap. Note that the ray may intersect both end caps, for example when $z_1 < z_{\min}$ and $z_2 > z_{\max}$.

Cone

The *infinite* double cone[2] aligned along the *z*-axis is defined as:

$$x^2+y^2=z^2 \tag{47}$$

In cylindrical polar coordinates it is:

$$r^2=z^2 \tag{48}$$

To intersect a ray with this, substitute Equation 24 in Equation 47.

$$(x_E+tx_D)^2+(y_E+ty_D)^2=(z_E+tz_D)^2 \tag{49}$$

$$\Rightarrow \quad t^2(x_D^2+y_D^2-z_D^2)+t(2x_Ex_D+2y_Ey_D-2z_Ez_D) +(x_E^2+y_E^2-z_E^2)=0 \tag{50}$$

$$\Rightarrow \quad at^2+bt+c=0 \tag{51}$$

$$\Rightarrow \quad t = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a} \tag{52}$$

where $a=x_D^2+y_D^2-z_D^2$, $b=2x_Ex_D+2y_Ey_D-2z_Ez_D$, and $c=x_E^2+y_E^2-z_E^2$.

The *finite* open-ended cone aligned along the *z*-axis is defined as:

$$x^2 + y^2 = z^2, z_{\min} < z < z_{\max} \tag{53}$$

The only difference between this and Equation 47 being the restriction on *z*. Note that if $z_{\min}$ and $z_{\max}$ are both positive or both negative then you get a single cone with its top truncated. If either $z_{\min}$ or $z_{\max}$ is zero you get a single cone with its apex at the origin.

To handle this finite length cone you proceed as for the finite length cylinder, with the obvious simple modifications.

Torus

A torus is defined by two parameters: the radius of the torus (that is the radius of the torus's defining circle, measured from the origin) and the radius of the tube (the perpendicular distance from the defining circle to the surface of the torus). These are R and $r$respectively. Normally $R > r$.

An implicit definition of the torus is:

$$\left(\sqrt{x^2 + y^2} - R\right)^2 + z^2 = r^2 \tag{54}$$

The torus can also be definied parametrically in terms of two angles, θ and ϕ, where θ can be thought of as the angle around the defining circle and ϕ the angle around the inside of the tube:

$$x = (R + r\cos\phi)\cos\theta \tag{55}$$

$$y = (R + r\cos\phi)\sin\theta \tag{56}$$

$$z = r\sin\phi \tag{57}$$

Substituting these three equations into Equation 54 will show that they are correct and is a useful exercise in algebraic manipulation.

To find the intersection points of a ray with a torus you need to substitute Equation 24 in Equation 54. Equation 58 is that substitution with the $(\sqrt{x^2 + y^2} - R)^2$ term expanded, the resulting square root term placed on one side of the equals sign, and all other terms placed on the other side:

$$2R\sqrt{x_E^2 + 2tx_Ex_D + t^2x_D^2 + y_E^2 + 2ty_Ey_D + t^2y_D^2}$$

$$= R^2 + x_E{}^2 + 2tx_Ex_D + t^2x_D{}^2 + y_E{}^2 + 2ty_Ey_D + t^2y_D{}^2 + z_E{}^2 + 2tz_Ez_D + t^2z_D{}^2 - r^2 \tag{58}$$

If we now square both sides we will get a quartic equation in t. This can be solved using a standard quartic root finder to find the four roots of the equation[3] (there are up to four intersection points between a torus and an arbitrary ray). A quartic root finder is described in *Graphics Gems V* (p. 3).

Plane

A plane can be defined by a normal vector, $\mathbf{N}$ and a point on the plane, $\mathbf{Q}$. A point, $\mathbf{P}$, is on the plane if:

$$\mathbf{N} \cdot (\mathbf{P} - \mathbf{Q}) = 0 \tag{59}$$

To find the ray/plane intersection substitute Equation 23 in Equation 59:

$$\mathbf{N} \cdot (\mathbf{E} + t\mathbf{D} - \mathbf{Q}) = 0 \tag{60}$$

$$\Rightarrow \quad t = \frac{\mathbf{N} \cdot (\mathbf{Q} - \mathbf{E})}{\mathbf{N} \cdot \mathbf{D}} \tag{61}$$

If t<0 then the plane is behind the eye point and there is no intersection. If $t \geq 0$ then the intersection point is $\mathbf{E} + t\mathbf{D}$. If $\mathbf{N} \cdot \mathbf{D} = 0$ then the ray is parallel to the plane, and there is no intersection point.

Polygon

A polygon can be defined by an ordered set of vertices: $(\mathbf{V_1}, \mathbf{V_2}, \mathbf{V_3}, \ldots)$. To find the intersection point between a polygon and a ray, we first find the intersection point between the polygon's plane and the ray, and then ascertain whether this intersection point lies inside the polygon or not.

The normal of the polygon's plane can be found by the simple cross product: $\mathbf{N} = (\mathbf{V_3} - \mathbf{V_2}) \times (\mathbf{V_1} - \mathbf{V_2})$. A point on the plane's surface is even easier to find: $\mathbf{Q} = \mathbf{V_1}$. The intersection calculation then proceeds as above.

If an intersection point between the ray and the plane is found then we can check whether or not the point lies inside the polygon in the following manner. First we project the intersection point and the polygon to two dimensions by simply throwing away one coordinate. Obviously we will throw away the coordinate in which the polygon has the smallest extent. We then test to see if the intersection point lies inside the two dimensional polygon using the odd/even test.

The odd/even test checks to see whether or not an arbitrary point lies inside an arbitrary polygon in two dimensions. This is done by drawing a horizontal ray from the point to infinity. If the ray crosses an even number of polygon edges then the point lies outside the polygon. Contrariwise, if the ray crosses an odd number of polygon edges then the point lies inside the polygon. A full discussion of the implementation details of this, and other point-in-polygon algorithms, can be found in *Graphics Gems IV* pp. 24-46.

Disc

The disc is similar to the polygon. Both are planar objects. A disc can be defined by its centre, $\mathbf{Q}$, its radius, r, and a normal vector, $\mathbf{N}$. Finding the intersection point, $\mathbf{P}$, of the ray with the disc's plane proceeds as for a plane/ray or polygon/ray intersection. Discovering whether $\mathbf{P}$ lies inside the disc requires you to simply check that:

$$(\mathbf{P} - \mathbf{Q}) \cdot (\mathbf{P} - \mathbf{Q}) \leq r^2 \tag{62}$$

Intersections with arbitrarily positioned primitives

Of the above primitives, only the plane and polygon are arbitrarily defined[4]. The sphere, cylinder, cone, and torus are all defined as being centred at the origin, and all have other restrictions on their definition[5]. In order to ray trace one of these primitives in an arbitrary location we have two alternatives: (1) find general intersection algorithms between a ray and the arbitrarily located versions of the primitives; or (2) use

geometric transforms to scale, rotate, and translate these primitives into the desired locations. This second option will be followed here.

Transforming the primitives

The basic idea here is a very simple. We specify a scaling, a rotation, and a translation which, between them, transform the primitive from its standard position to the desired location. You will remember that this was covered in the IB *Computer Graphics & Image Processing* course. To perform the intersection we take the inverse transform of the ray, intersect this with the primitive in its standard position, and then transform the resulting intersection point to its correct location.

We now need to take a small diversion in order to explain the difference between a vector representing a point and a vector representing a displacement. A displacement can be thought of as the difference between two points. When transforming objects, displacements are scaled and rotated but *not* translated. Think of it this way: if you translate two points, the displacement between them stays exactly the same. But if you scale or rotate the two points, the displacement between them scales or rotates accordingly.

All this is by way of explaining how we transform a ray in order to intersect the appropriate ray with the primitive in its standard position. Let us assume that the primitive object in its standard position, $\hat{\mathbf{B}}$, undergoes transformation[6] $\mathbf{TRS}$ to get into the desired position $\mathbf{B} = \mathbf{TRS}\hat{\mathbf{B}}$. To intersect ray, $\mathbf{E} + t\mathbf{D}$, with $\mathbf{B}$ we transform the point, $\mathbf{E}$, and the displacement $\mathbf{D}$ as follows:

$$\hat{\mathbf{E}} = \mathbf{S}^{-1}\mathbf{R}^{-1}\mathbf{T}^{-1}\mathbf{E} \quad (63)$$

$$\hat{\mathbf{D}} = \mathbf{S}^{-1}\mathbf{R}^{-1}\mathbf{D} \quad (64)$$

The point is translasted, rotated and scaled but the displacement is only rotated and scaled.

Now intersect ray, $\hat{\mathbf{E}} + t\hat{\mathbf{D}}$, with the object in its standard position, $\hat{\mathbf{B}}$, as described in the previous sections. This gives the value of *t* and consequently allows you to directly calculate $\mathbf{P} = \mathbf{E} + t\mathbf{D}$.

In addition to scaling, rotating and translating the standard primitives, this mechanism allows us to stretch and squash them by scaling them by different factors in the different dimensions. For example, an ellipoid is simply a stretched sphere. However, such anisotropic scaling causes interesting problems with the normal vectors, as discussed in the following section.

Transforming normal vectors

In ray tracing we need to know not just the intersection point but also the normal vector to the surface at the intersection point. This normal vector is used in the illumination calculations. However, if we scale an object anisotropically, the normal vector scales as the *inverse* of the object scaling, although it rotates in the *same* way as the object.

This can be mathematically explained by remembering that the normal vector is related to the derivative of the surface. The normal vector is perpendicular to the surface, which means that it is perpendicular to any derivative vector. As an example, consider the ellipsoid created by scaling the unit sphere by the matrix:

$$\mathbf{S} = \begin{bmatrix} l & 0 & 0 & 0 \\ 0 & m & 0 & 0 \\ 0 & 0 & n & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (65)$$

The intersection point between the unit sphere and the inverse transformed ray will be at some point $\hat{\mathbf{P}} = (\hat{x}, \hat{y}, \hat{z})$. This equates to the spherical polar coordinate $(1, \theta, \phi)$ where[7]

$(\hat{x}, \hat{y}, \hat{z}) = (\cos\phi\cos\theta, \cos\phi\sin\theta, \sin\phi)$. By virtue of the fact that the normal to every point on the sphere passes through the centre of the sphere, it is easy to see that the normal vector, $\hat{\mathbf{N}}$, is also $\hat{\mathbf{N}} = (\cos\phi\cos\theta, \cos\phi\sin\theta, \sin\phi)$.

Transforming $\hat{\mathbf{P}}$ to the true intersection point $\mathbf{P}$ is simply a matter of applying $\mathbf{P} = \mathbf{S}\hat{\mathbf{P}}$. Thus the intersection point of the ray with the ellipsoid is at rectangular coordinate $(x, y, z) = (l\cos\phi\cos\theta, m\cos\phi\sin\theta, n\sin\phi)$.

To find the normal vector to the ellipsoid at this intersection point, we need to find two non-parallel derivative vectors to the surface at the intersection point, and take their cross product to give the normal vector, $\mathbf{N}$. Two derivative vectors are:

$$\frac{\partial(\mathbf{P})}{\partial\phi} = \frac{\partial(x,y,z)}{\partial\phi} = (-l\sin\phi\cos\theta, -m\sin\phi\sin\theta, n\cos\phi) \tag{66}$$

$$\frac{\partial(\mathbf{P})}{\partial\theta} = \frac{\partial(x,y,z)}{\partial\theta} = (-l\cos\phi\sin\theta, m\cos\phi\cos\theta, 0) \tag{67}$$

This leads to the normal vector being[8]:

$$\mathbf{N} = \frac{\partial(\mathbf{P})}{\partial\phi} \times \frac{\partial(\mathbf{P})}{\partial\theta} \tag{68}$$

$$= (\frac{1}{l}\cos\phi\cos\theta, \frac{1}{m}\cos\phi\sin\theta, \frac{1}{n}\sin\phi) \tag{69}$$

Thus, we conclude that while:

$$\mathbf{P} = \mathbf{S}\hat{\mathbf{P}} \tag{70}$$

$$\mathbf{N} = \mathbf{S}^{-1}\hat{\mathbf{N}} \tag{71}$$

This leaves open the question of what to do about the rotation and translation components of the transformation. It is intuitively obvious that rotating an object causes the normal vector to rotate by the same amount and, because normal vectors are displacements rather than points, they should not be translated. So the final analysis is that:

$$\mathbf{P} = \mathbf{TRS}\hat{\mathbf{P}} \tag{72}$$

$$\mathbf{N} = \mathbf{RS}^{-1}\hat{\mathbf{N}} \tag{73}$$

This analysis has been applied to the unit sphere, but the same result holds for any object.

Converting the primitives to polygons

We may be in a situation where we define objects in terms of the various primitives, but where we wish to draw the objects using polygon scan conversion. In this case we need to convert primitives into polygons.

Some polygon scan conversion algorithms can only deal with triangles; in these cases we may need to do a little bit of extra work to ensure that all of the generated polygons are triangles.

Converting the curved primitives (sphere, cylinder, cone, torus, disc) involves approximating a curved profile by a series of straight line segments. The simplest example is the disc. This can be approximated by a regular n-gon, where n is chosen to give an adequate approximation to the disc. ``Adequate'' in this case will depend on the desired rendering resolution, the desired speed of rendering, and the desired quality of the final image. If necessary, this n-gon can be converted to triangles in one of a number of ways: (1) define a central vertex and connect every edge to this vertex to make $n$isosceles triangles; (2) select one vertex on the n-gon, and make a triangle fan emanating from this vertex; or (3) start at one edge of the n-gon and make a triangle strip set which proceeds from this edge of the polygon to the opposite edge.

A cylinder or cone can be converted to polygons by polygonising the discs at either end into n-gons, and then connecting corresponding vertices on the two n-gons. In the special case of a cone with a point, one of the n-gons obviously degenerates to that point.

Spheres and tori can be most easily converted to polygons by considering their parameterisation in terms of θ and ϕ(for a sphere these are Equations 1-3; for a torus they are Equations 55-57). By selecting

appropriate steps in the two parameters we can generate a set of quadrilaterals which approximates the curved primitive.

It should be noted that spheres can be polygonised with more uniform polygons by starting with one of the five Platonic solids, and subdividing its faces accordingly. The details of this are left as an exercise to the reader.

Next Up Previous

Next: Bezier curves **Up:** Some Mathematics for Advanced **Previous:** Some Mathematics for Advanced

nad@cl.cam.ac.uk