

Lecture on Plane Geometry¹

As we've seen, CG usually breaks down a model into a large number of planar regions (quads and triangles). Even curved surfaces are ultimately rendered as a large number of planar facets.

Very often the CG system then needs to do some additional geometry with the planes, such as determining if a ray of light (say from a spotlight), intersects the planar region. To do that, we need a bit of geometry.

To motivate this, we will look at the following demo:

`~cs307/public_html/demos/animation/Laser.cc`

This program has an animation of a "UFO" flying over a field with a barn on it. It has a "photon torpedo" that it fires a random downward directions. We can animate the moving photon torpedo, but we need to determine when and where the torpedo intersects the field or the barn, so that we can draw the explosion.

1 Implicit Equation of a Plane

First, let's see how to define the implicit equation of a plane. Let P_0 be a specific point on the plane, any point, but one where we know the coordinate. Let n be the normal vector for the plane. Here is an example:

$$\begin{aligned}P_0 &= (5, 4, 7) \\ n &= (1, 2, 2)\end{aligned}$$

Now, note that any vector lying in the plane must be perpendicular to the normal vector. If we let P be a variable standing for every point in the plane:

$$P = (x, y, z)$$

Then we know that:

$$0 = n \cdot (P - P_0)$$

With a little abuse of notation, we can derive:

$$0 = n \cdot P - n \cdot P_0$$

Using our example:

$$\begin{aligned}0 &= n \cdot (P - P_0) \\ &= (1, 2, 2) \cdot [(x, y, z) - (5, 4, 7)] \\ &= (1, 2, 2) \cdot [(x - 5, y - 4, z - 7)] \\ &= 1(x - 5) + 2(y - 4) + 2(z - 7) \\ &= x + 2y + 2z - [(1 \cdot 5) + (2 \cdot 4) + (2 \cdot 7)] \\ &= x + 2y + 2z - 27\end{aligned}$$

Note that this is an *implicit* equation of a plane. It can tell us whether a point is on the plane or not, but it doesn't easily generate points on the plane. Also, the implicit equation doesn't generalize to higher dimensions (but that's a problem for mathematicians, not us).

In general, the equation of a plane in 3D is:

$$ax + by + cz + d = 0$$

¹Some of this development is helped and improved by Dan Sunday's work at http://geometryalgorithms.com/Archive/algorithm_0104/algorithm_0104.htm

2 Parametric Equation of a Plane

Another way to define a plane is by a point on the plane and two vectors that lie in the plane. (This works in higher dimensional spaces, not just 3D.) Let the two vectors be v and w . We can then define the equation:

$$P(s, t) = P_0 + sv + tw$$

A very common situation for this is when we have a triangle of three points, P_0, P_1, P_2 , and we develop it like this:

$$\begin{aligned}v &= P_1 - P_0 \\w &= P_2 - P_0 \\P(s, t) &= P_0 + sv + tw \\&= (1 - s - t)P_0 + sP_1 + tP_2\end{aligned}$$

This is a nice equation using only the three points we started with, plus two parameters. Notice that the point $P(s, t)$ is *inside* the triangle when:

$$\begin{aligned}0 &\leq s \\0 &\leq t \\s + t &\leq 1\end{aligned}$$

The point is on the perimeter if $s = 0$, $t = 0$ or $s + t = 1$. Each condition corresponds to one edge. For a parallelogram, we have: $0 \leq s, t \leq 1$

Of course, if we want the implicit equation, we can take the cross product of v and w to find n , and proceed from there.

3 Fun Facts

3.1 Any Point

Notice that the argument for the implicit equation works for any point on the plane, so that if we used P_1 instead of P_0 , we should get the same equation:

$$\begin{aligned}0 &= n \cdot P - n \cdot P_0 \\0 &= n \cdot P_1 - n \cdot P_0 \\n \cdot P_0 &= n \cdot P_1\end{aligned}$$

This means that the constant term, d , in the equation, is the same for any point on the plane. The normal vector dotted with any point on the plane yields this same value.

3.2 Finding the Normal Vector

Notice that you can just read off a normal vector to a plane from its implicit equation. Very convenient!

3.3 Normalized Normal Vector

In the development, you notice that we just chose an arbitrary normal vector. What happens if we choose a different one (a scalar multiple)?

$$0 = (kn) \cdot (P - P_0)$$

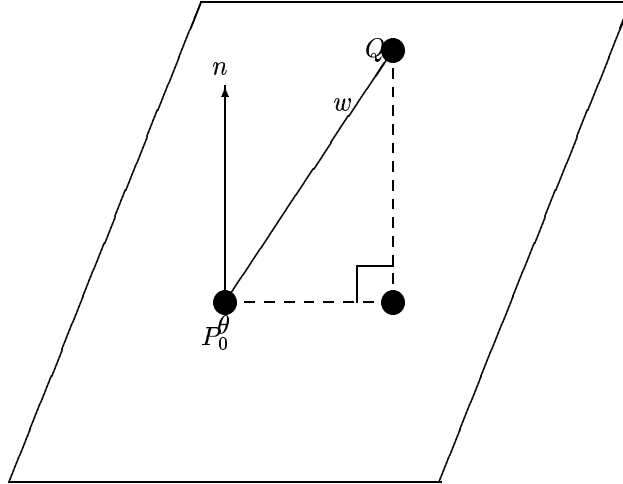


Figure 1: Distance from a Point to a Plane. Q is the point not on the plane, and P_0 is some point on the plane. θ is the angle between the normal vector and the vector from P_0 to Q .

Of course, we can just multiply both sides of this equation by $1/k$ and get the same result. So it doesn't matter. It's common to use a *normalized* normal vector, so that $|n| = 1$. Using our example, we have $n = (1, 2, 2)$, so $|n| = 3$, so:

$$\begin{aligned} 0 &= x + 2y + 2z - 27 \\ &= (1/3)x + (2/3)y + (2/3)z - 9 \end{aligned}$$

3.4 Dividing Space

If we define:

$$f(x, y, z) = ax + by + cz + d$$

Of course, this function will be zero when the point lies on the plane. More interestingly, this function divides space into two halves, and the points on one half yield a positive result and the points on the second half a negative result. Thus, you can use the function to tell what side of a plane a point is.

3.5 Distance from a Point to a Plane

If we use a normalized normal in our implicit equation, we have an interesting property, namely that the function gives the (signed) distance of the point to the plane. Why?

Let θ be the angle between the normal vector and the vector w from a point on the plane to the point off the plane. (See figure 1.) The perpendicular distance is then:

$$\begin{aligned} &= |P_0 - Q| \cos \theta \\ &= \frac{n \cdot (P_0 - Q)}{|n|} \end{aligned}$$

When $|n| = 1$, the denominator goes away. Also, since P_0 is a point on the plane, $n \cdot P_0$ is just d , the constant in the implicit equation. Thus we have:

$$\begin{aligned}
&= n \cdot P - n \cdot Q \\
&= d - n \cdot Q \\
&= d - aQ_x + bQ_y + cQ_z
\end{aligned}$$

This is just our implicit equation! So, the key thing is that if the normal vector is unit length, the implicit equation gives the distance from the point to the line.

4 Intersecting a Ray with a Plane

Suppose we have a point Q not on the plane (we'll reserve P for points on the plane), and a vector r indicating a ray starting at Q . Does the ray intersect the plane? If so, where? How far?

We could solve this many ways. We will do it by creating a parametric equation of the ray and intersecting that with the implicit equation of the plane. That is, we combine these two equations:

$$\begin{aligned}
Q(t) &= Q + tr \\
0 &= ax + by + cz + d
\end{aligned}$$

And we get:

$$0 = a(Q_x + tr_x) + b(Q_y + tr_y) + c(Q_z + tr_z) + d$$

This is an equation just in t , so we solve for t and we're almost home.

Let's do an example, with $Q = (11, 15, 8)$ and $r = (-1, 3, 2)$ and using our plane from before:

$$\begin{aligned}
0 &= (11 - t) + 2(15 + 3t) + 2(8 + 2t) - 27 \\
&= 11 - t + 30 + 6t + 16 + 4t - 27 \\
&= 9t + 30 \\
t &= -10/3
\end{aligned}$$

The fact that t is negative tells us that the ray does *not* intersect the plane. The photon torpedo is moving away from the plane. If we reverse r , we would get a positive t and the photon torpedo *would* intersect the plane.

If we compute this intersection parameter with *several* planes, we know that the torpedo will hit them in the order of the parameter values. Thus, it will blow up the one with the *smallest* parameter.

5 Intersecting a Ray with a Plane Again

The math in the last section is fine, but here's a better way, again thanks to geometryalgorithms.com. Given a line defined by Q and R and a plane defined by P and N (all knowns; I've dropped the subscripts for convenience), we can substitute the parametric representation of a line into one of our representations of a plane:

$$(Q + tR) \cdot N = P \cdot N$$

From there, we can use some reasonable algebra to solve for t :

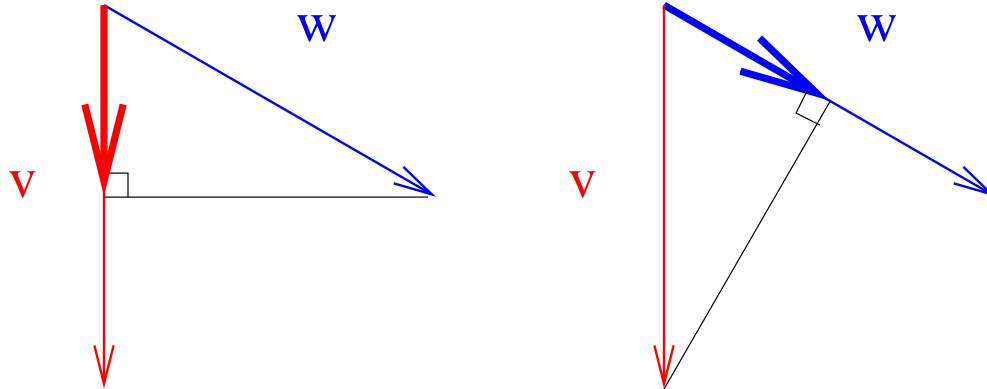


Figure 2: Two projections of one vector onto another.

$$\begin{aligned}
 (Q + tT) \cdot N &= P \cdot N \\
 Q \cdot N + t(R \cdot N) &= P \cdot N \\
 t(R \cdot N) &= P \cdot N - Q \cdot N \\
 t(R \cdot N) &= (P - Q) \cdot N \\
 t &= \frac{(P - Q) \cdot N}{R \cdot N}
 \end{aligned}$$

Not exactly intuitive, but simple to compute!

We should check for special cases:

- A ray parallel to the plane (which means there's no intersection). If that's the case, the ray will be perpendicular to the surface normal, or $R \cdot N = 0$. So, we just check that the denominator is not zero.
- A ray lying in the plane. If that happens, not only will the ray be parallel to the plane, but the point P will lie on the plane. That means $(P-Q)$ is a vector lying in the plane and therefore the dot product with the normal vector is zero. Thus, in this case, the numerator is zero.

6 Intersecting a Ray with a Triangle

But we don't care about whether the ray intersects the plane, we care whether it intersects a triangle or a quad. We can compute the point of intersection (from the parameter and our parametric equation of the ray), and then try to compute the s and t values so that we can use the constraints in section 2.

6.1 Dot Products as Projections

Before we develop the code for finding the intersection point within a triangle, it's helpful to have as a building block a more complete understanding the usefulness of the dot product.

We know that the dot product gives us something like the cosine of the angle between two vectors. In fact, for unit vectors, it gives us exactly the cosine of the angle between them. Let's start with unit vectors. Suppose we have unit vectors v and w . If we compute the following:

$$\begin{aligned}
 v' &= (v \cdot w)v \\
 w' &= (v \cdot w)w
 \end{aligned}$$

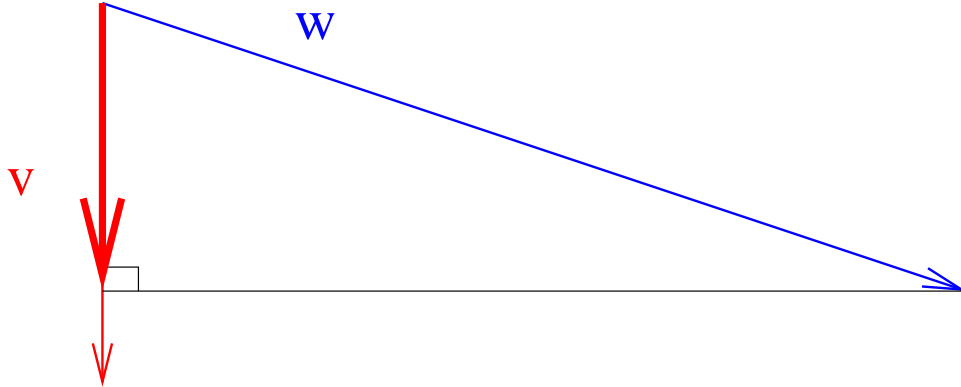


Figure 3: Projecting a non-unit vector.

The v' and w' vectors are scalings of the original vectors, where they are scaled by the dot product. Geometrically, this is equivalent to the projection of the other vector onto this one; see figure 2.

But what about non-unit vectors? Consider the projection in figure 3 and assume that v is a unit vector but w is not. To project w onto v , we want to resize v so that it has a length equal to the length of w multiplied by the cosine of the angle between v and w . Call that angle θ . The length of v' should be:

$$|w| \cos(\theta) = |w| \frac{v \cdot w}{|v||w|} = \frac{v \cdot w}{|v|}$$

Since $|v| = 1$, we can drop that, and we find that

The dot product gives the projection of any vector onto a unit vector.

6.2 Finding the Intersection Point Parameters

If I is the intersection point of the line and the plane, we have:

$$I = P_0 + s(P_1 - P_0) + t(P_2 - P_0) = P_0 + su + tv$$

We have to solve this for s and t , the parameters of the intersection point. Since we have three dimensions in two unknowns, we can certainly solve this. Indeed, we can solve it three different ways, depending on which equation we decide to leave out. I consulted geometryalgorithms.com and decided, for better or worse, to solve it using their math.

Here's my explanation of their math. If you'd like to see their way, consult

http://geometryalgorithms.com/Archive/algorithm_0105/algorithm_0105.htm.

Let w be the vector from P_0 to the intersection point: $w = I - P_0$. We want to solve the following equation for s and t .

$$w = su + tv$$

Note that this equation just says that w is a linear combination of vectors u and v .

They solve this in a very clever way. To solve for t , they construct a vector that is orthogonal (perpendicular) to u but that also lies in the plane; call it u^\perp , pronounced "u perp." The dot product of u^\perp with u is, of course, zero, so taking the dot product of the right side of this equation with u^\perp nullifies the su term, leaving an equation with only the t parameter to solve for. Of course, there are infinitely many vectors perpendicular to u ; it's important that they choose one that lies in the plane, since that gives us the situation shown in figure 4.

In that figure, a and b are the projections of w and v onto u^\perp . The scalar multiples are:

$$\begin{aligned} a &= w \cdot u^\perp \\ b &= v \cdot u^\perp \end{aligned}$$

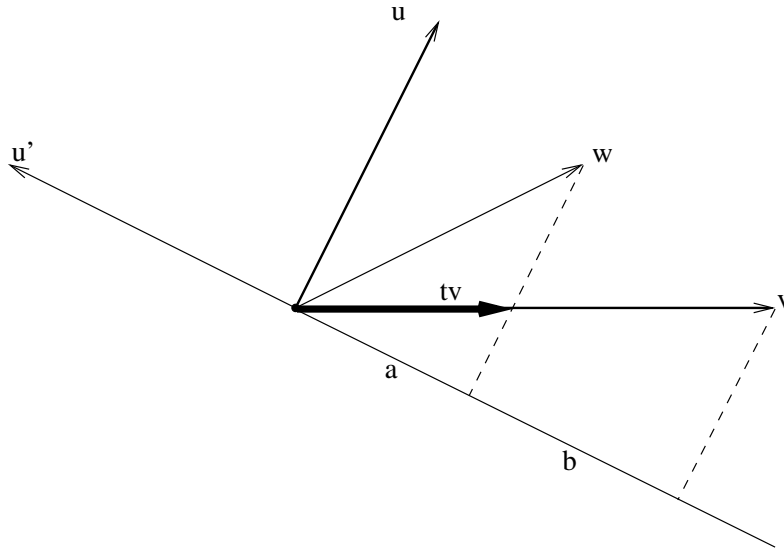


Figure 4: Taking the dot product of w with u^\perp (shown as u'). We start with known vectors u , v and w , where w is some linear combination of u and v : $w = su + tv$. We first find u^\perp , the perpendicular to u lying in the plane (denoted u' in the figure). Taking the dot product finds tv , the amount of vector v that is in the linear combination of u and v to make up w .

By similar triangles, the vector tv is to v as a is to b :

$$\frac{tv}{v} = \frac{a}{b}$$

Therefore, we can find $t = a/b$. Here it is again, purely algebraically:

$$\begin{aligned} w &= su + tv \\ w \cdot u^\perp &= (su + tv) \cdot u^\perp \\ &= s(u \cdot u^\perp) + t(v \cdot u^\perp) \\ &= t(v \cdot u^\perp) \\ t &= \frac{w \cdot u^\perp}{v \cdot u^\perp} \end{aligned}$$

Note that, because the numerator and denominator both have u^\perp in them, it doesn't matter whether u^\perp is a unit vector, because the scale factor to normalize it appear in both the numerator and denominator and therefore would cancel.

Similarly, we can solve for s by finding a v^\perp that is perpendicular to v and lies in the plane t .

$$s = \frac{w \cdot v^\perp}{u \cdot v^\perp}$$

How can we find these perpendicular vectors? Since they lie in the plane, they must be perpendicular to the plane normal, N . Since the cross product finds a vector perpendicular to two others, we have:

$$\begin{aligned} u^\perp &= N \times u \\ v^\perp &= N \times v \end{aligned}$$

Next, they introduce a computational shortcut. It turns out that there is an identity for cross products, namely:

$$(a \times b) \times c = (a \cdot b)b - (b \cdot c)a$$

We're not even going to think about proving that; we're just going to use it. Consequently,

$$\begin{aligned} u^\perp &= n \times u = (u \times v) \times u = (u \cdot u)v - (u \cdot v)u \\ v^\perp &= n \times v = (u \times v) \times v = (u \cdot v)v - (v \cdot v)u \end{aligned}$$

And now we can compute s and t using only dot products.

$$\begin{aligned} s &= \frac{(u \cdot v)(w \cdot v) - (v \cdot v)(w \cdot u)}{(u \cdot v)^2 - (u \cdot u)(v \cdot v)} \\ t &= \frac{(u \cdot v)(w \cdot u) - (u \cdot u)(w \cdot v)}{(u \cdot v)^2 - (u \cdot u)(v \cdot v)} \end{aligned}$$

Notice the similarity between the two calculations. Thus, the complete calculation only requires five distinct dot products.

We need to check for special cases where the triangle is degenerate:

- One way this can happen is if two of the points defining the triangle are the same. If this happens, either u or v will be the zero vector, and $u \cdot u = 0$ or $v \cdot v = 0$.
- Another way is if the three points are colinear, in which case u is a scalar multiple of v . One way to test for this is to test whether the cosine of the angle between u and v is 1, which happens when the angle is zero.

$$\begin{aligned} \frac{u \cdot v}{|u||v|} &= 1 \\ u \cdot v &= |u||v| \\ (u \cdot v)^2 &= (u \cdot u)(v \cdot v) \\ (u \cdot v)^2 - (u \cdot u)(v \cdot v) &= 0 \end{aligned}$$

Since the quantity on the left is the denominator of our fractions to compute s and t , all we need to do is check for zero before dividing.

7 Photon Torpedoes

Let's look at the photon torpedo (Laser) demo.

- Notice how the animation of the motion of the UFO is done. We give it an initial location and direction, controlled by global variables. The display function uses the location variable and the idle callback updates it.
- Notice how the random direction of the laser is generated. Feel free to use ideas like that if you want random numbers in your own animation.
- Notice how the laser is drawn, using `twNormalizeVector` and `twVectorScale` to ensure that the barrel is always 20 units long.
- Look at the implementation of `blast`. First, we find the nearest "fragment," using `twNearestFragment`. We'll look at the code for that function, which is in `~cs307/public_html/tw/tw-geometry.cc`

- `twNearestFragment` iterates over the fragments, determines the parameter values and finds the smallest.
- `twLineTriangleIntersection` computes three parameters, the one on the line (ray) and the two in the plane, for the intersection point
- `twLinePlaneIntersection` computes the parameter on the line, using the math we developed in section 5.
- `twPointInTriangle` computes the parameters in the triangle, using the math we did in section 6.
- See how the parameter of the torpedo depends on the frame number. This is how it advances in each frame.
- When the parameter exceeds the parameter of the intersection point, the collision with the surface has occurred, so start drawing the explosion.