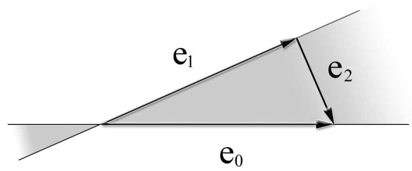


**Figure 1**. Divide plane.

$\mathbf{e}_1 = \mathbf{v}_2 - \mathbf{v}_0$, and $\mathbf{e}_2 = \mathbf{v}_1 - \mathbf{v}_2$. First, the signed distances to the plane from $\mathbf{s}$ and $\mathbf{e}$ are computed as $t_s = \mathbf{n} \cdot \mathbf{s} + d$, and $t_e = \mathbf{n} \cdot \mathbf{e} + d$. If $t_s t_e > 0$, then $\mathbf{s}$ and $\mathbf{e}$ are on the same side of the plane, and no intersection can occur. Otherwise, the point of intersection between the plane and the line segment is

$$\mathbf{p} = \mathbf{s} + t(\mathbf{e} - \mathbf{s}), \tag{1}$$

where

$$t = \frac{t_s}{t_s - t_e}. \tag{2}$$

Next, I notice that a triangle edge extended infinitely in the triangle's plane divides the plane into two parts. Two triangle edges divide the plane into four sectors as shown in Figure 1. My test involves determining whether the point $\mathbf{p}$ of intersection is between edges $\mathbf{e}_0$ and $\mathbf{e}_1$, and also between $\mathbf{e}_0$ and $\mathbf{e}_2$.

Now, compute the following normals of the triangle:

$$\begin{aligned} \mathbf{n}_0 &= \mathbf{e}_0 \times \mathbf{q}, \\ \mathbf{n}_1 &= \mathbf{e}_1 \times \mathbf{q}, \\ \mathbf{n}_2 &= \mathbf{e}_2 \times \mathbf{q}, \end{aligned} \tag{3}$$

where $\mathbf{q} = \mathbf{p} - \mathbf{v}_0$. Those normals may differ in sign and length, but not in direction. Then, the idea of my algorithm is to use dot products to determine whether $\mathbf{p}$ is inside the triangle, as shown below.

$$\begin{aligned} &\texttt{if(}\mathbf{n}_0 \cdot \mathbf{n}_1 < 0\texttt{) return no intersection;} \\ &\texttt{if(}\mathbf{n}_0 \cdot \mathbf{n}_2 < 0\texttt{) return no intersection;} \end{aligned} \tag{4}$$

The entire test up to this point does not say that this algorithm will perform well. However, as explained next, there are several different optimizations that can be implemented to obtain good performance.