Triangle – Triangle Collision Detection Strategy

Paul Nathan 14/01/2015

Triangles obtained from binary .stl file. Pre-processed to custom data types of

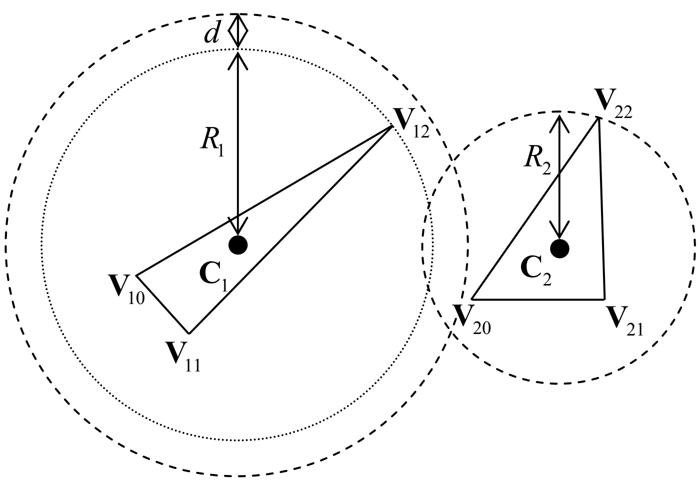
- Vertices (x, y, z), anti-clockwise winding
- Edge vectors and "**F**" vectors and inverse determinants
- Unit normal vectors (per primitive)
- Bounding sphere centre point (x, y, z) and radius

In all subsequent notation, the first subscript represent the triangle number (either 1 or 2) and the second represent either the vertex or edge number of that triangle

Collision detection strategy outline:

1. Bounding sphere collision test or Octree search ($O(N)$ vs. $O(\log(N))$, continue only if true
2. Edge – plane intersection test
   a. If any edge of triangle 1 intersects the plane of triangle 2 and the intersection point is within triangle 2 (homogeneous barycentric coordinates point-in-triangle test), collision
   b. Repeat test for edges of triangle 2 against triangle 1
   c. Part b will catch all collisions except the case of co-planar intersecting triangles. However since the triangles will be part of a convex hull (there should be no isolated triangles) then this degenerate case will be captured by the edge – plane test with the directly connected neighbouring triangles. Thus the added complexity of capturing this case is not necessary
3. If no collision, compute minimum distance between triangles, else return collision point
   a. Vertex – plane distance (both triangles), include if point is within triangle
   b. Edge – edge minimum separation distance, if points not on edge then clamp to vertices
   c. Vertex – vertex distance
   d. Return minimum value of all computed distances.

(STEP 1: BOUNDING SPHERE TEST) – *see separate documentation for Octree search*



$$\text{If } (\mathbf{C}_2 - \mathbf{C}_1)\bullet(\mathbf{C}_2 - \mathbf{C}_1) \le (R_1 + R_2 + d)^2 \text{ then proceed to step 3}$$

Details on the bounding sphere:

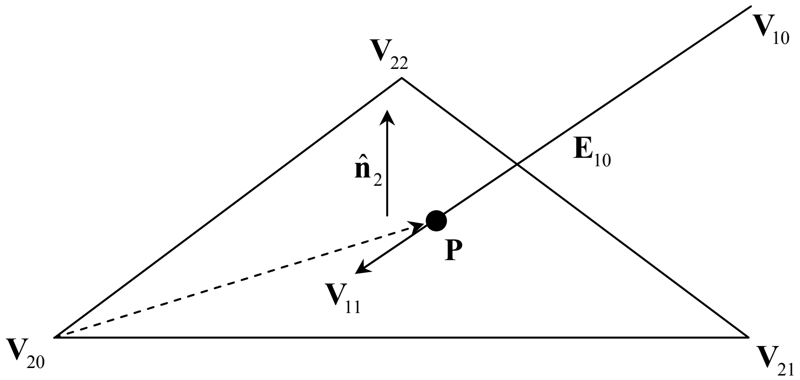
Although the circumsphere would pass through all three vertices, it may not be such a useful early test eliminator for highly skewed triangles where the radius would be very large compared to the triangle edge lengths, thus a more simple bounding sphere with its centre at the centroid of the triangle and radius that joins the centroid to the vertex furthest from the centroid shall be used

$$\mathbf{C}_i = \tfrac{1}{3}\left(\mathbf{V}_{i0} + \mathbf{V}_{i1} + \mathbf{V}_{i2}\right)$$
$$R_i = \max\left(\left\|\mathbf{V}_{i0} - \mathbf{C}_i\right\|, \left\|\mathbf{V}_{i1} - \mathbf{C}_i\right\|, \left\|\mathbf{V}_{i2} - \mathbf{C}_i\right\|\right)$$

STEP 2: EDGE – TRIANGLE TEST

1. Point-on-edge test:

Consider the example of edge 0 of triangle 1 with the plane of triangle 2



$$\mathbf{E}_{x0} = \mathbf{V}_{x1} - \mathbf{V}_{x0}$$
$$\mathbf{E}_{x1} = \mathbf{V}_{x2} - \mathbf{V}_{x1}$$
$$\mathbf{E}_{x2} = \mathbf{V}_{x0} - \mathbf{V}_{x2}$$

The parametric vector equation of a point **P** on the edge is

$$\mathbf{P} = \mathbf{V}_{10} + s\mathbf{E}_{10}$$

The signed distance d of the point **P** from the plane of the triangle is given by

$$d = \hat{\mathbf{n}}_2 \bullet \left(\mathbf{P} - \mathbf{V}_{20}\right)$$

On intersection, $d = 0$. Substituting the equation for **P** and solving for s gives

$$\hat{\mathbf{n}}_2 \bullet \left(\mathbf{V}_{10} + s\mathbf{E}_{10} - \mathbf{V}_{20}\right) = 0$$

$$s = \frac{\hat{\mathbf{n}}_2 \bullet \left(\mathbf{V}_{20} - \mathbf{V}_{10}\right)}{\hat{\mathbf{n}}_2 \bullet \mathbf{E}_{10}}$$

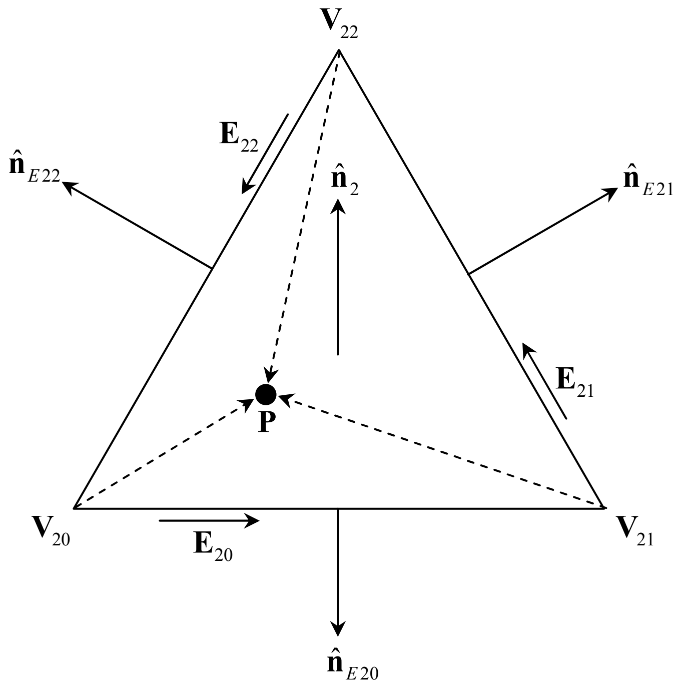
The edge intersects the plane if $0 \le s \le 1$, or, when carried out numerically

$$\left(s > -\varepsilon\right) \;\wedge\; \left(s < 1 + \varepsilon\right)$$

If false, move on to another edge. Once all edges are done, repeat for edges of triangle 2 on plane of triangle 1. Note that if $\hat{\mathbf{n}}_2 \bullet \mathbf{E}_{10} = 0$ the edge is parallel to the plane, and if also $\hat{\mathbf{n}}_2 \bullet \left(\mathbf{V}_{20} - \mathbf{V}_{10}\right) = 0$ then the edge is coplanar. To avoid a 0/0 ambiguity, ε can be added to the denominator.

2. Point-in-triangle test:

If the point lies on the positive (outwards facing, or normal-directed) side of any edge, then it must be outside the triangle.



$$\hat{\mathbf{n}}_{E2i} \bullet \left(\mathbf{P} - \mathbf{V}_{2i}\right) < \varepsilon \qquad i = 0,1,2$$

where

$$\hat{\mathbf{n}}_{E2i} = \frac{\mathbf{E}_{2i}}{\left\|\mathbf{E}_{2i}\right\|} \times \hat{\mathbf{n}}_2$$

These edge unit normal vectors can be pre-calculated and stored with the triangle custom data type. This test is inclusive of the point lying on an edge.

STEP 3: MINIMUM SEPARATION DISTANCE

If no collision then, if desired, return the minimum separation distance.

1. Vertex – triangle distance:

Consider the example of vertex x of triangle 1 ($\mathbf{V}_{1x}$) with triangle 2. The signed distance d of the vertex is

$$d = \hat{\mathbf{n}}_2 \bullet \left(\mathbf{V}_{1x} - \mathbf{V}_{20}\right)$$

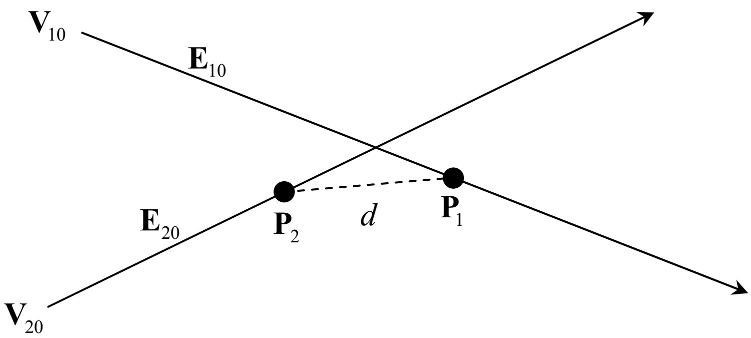
This distance is along the triangle normal. The associated point $\mathbf{P}$ on the plane of the triangle is given by

$$\mathbf{P} = \mathbf{V}_{1x} - \hat{\mathbf{n}}_2 d$$

A point-in-triangle test as derived previously is then carried out on this point. If the point is within the triangle, then add its *absolute* value to the list of valid separation distances to consider. Repeat for each vertex of triangle 1, and then do the same for the vertices of triangle 2 against triangle 1.

2. Edge – edge minimum separation distance:

Consider the example of edge 0 of triangle 1 with edge 0 of triangle 2



$$\mathbf{P}_1 = \mathbf{V}_{10} + s_1\mathbf{E}_{10}$$
$$\mathbf{P}_2 = \mathbf{V}_{20} + s_2\mathbf{E}_{20}$$

$$d = \left\|\mathbf{P}_1 - \mathbf{P}_2\right\| = \left\|\mathbf{Q} + s_1\mathbf{E}_{10} - s_2\mathbf{E}_{20}\right\| \qquad \mathbf{Q} = \mathbf{V}_{10} - \mathbf{V}_{20}$$

The objective is to minimise absolute value of *d*. This can be done analytically by minimising the square of *d*

$$\begin{aligned} d^2 &= (\mathbf{P}_1 - \mathbf{P}_2)\bullet(\mathbf{P}_1 - \mathbf{P}_2) \\ &= (\mathbf{Q} + s_1\mathbf{E}_{10} - s_2\mathbf{E}_{20})\bullet(\mathbf{Q} + s_1\mathbf{E}_{10} - s_2\mathbf{E}_{20}) \\ &= s_1{}^2\mathbf{E}_{10}\bullet\mathbf{E}_{10} + s_2{}^2\mathbf{E}_{20}\bullet\mathbf{E}_{20} - 2s_1s_2\mathbf{E}_{10}\bullet\mathbf{E}_{20} + 2s_1\mathbf{E}_{10}\bullet\mathbf{Q} - 2s_2\mathbf{E}_{20}\bullet\mathbf{Q} + \mathbf{Q}\bullet\mathbf{Q} \end{aligned}$$

The first derivatives are

$$\frac{\partial d^2}{\partial s_1} = 2s_1\mathbf{E}_{10}\bullet\mathbf{E}_{10} - 2s_2\mathbf{E}_{10}\bullet\mathbf{E}_{20} + 2\mathbf{E}_{10}\bullet\mathbf{Q}$$
$$\frac{\partial d^2}{\partial s_2} = 2s_2\mathbf{E}_{20}\bullet\mathbf{E}_{20} - 2s_1\mathbf{E}_{10}\bullet\mathbf{E}_{20} - 2\mathbf{E}_{20}\bullet\mathbf{Q}$$

The second derivatives are

$$\frac{\partial^2 d^2}{\partial s_1{}^2} = 2\mathbf{E}_{10}\bullet\mathbf{E}_{10} = 2E_{10}{}^2$$
$$\frac{\partial^2 d^2}{\partial s_2{}^2} = 2\mathbf{E}_{20}\bullet\mathbf{E}_{20} = 2E_{20}{}^2$$
$$\frac{\partial^2 d^2}{\partial s_1\partial s_2} = -2\mathbf{E}_{10}\bullet\mathbf{E}_{20} = -2E_{10}E_{20}\cos\theta_{E_{10}E_{20}}$$

The discriminant D is thus

$$\begin{aligned} D &= \frac{\partial^2 d^2}{\partial {s_1}^2} \cdot \frac{\partial^2 d^2}{\partial {s_2}^2} - \left(\frac{\partial^2 d^2}{\partial s_1 \partial s_2} \right)^2 \\ &= 4{E_{10}}^2 {E_{20}}^2 - 4{E_{10}}^2 {E_{20}}^2 \cos^2 \theta_{E_{10}E_{20}} \\ &= 4{E_{10}}^2 {E_{20}}^2 \sin^2 \theta_{E_{10}E_{20}} \end{aligned}$$

The distance returned will always be a minimum due to the following

$$\frac{\partial^2 d^2}{\partial {s_1}^2} > 0 \qquad \frac{\partial^2 d^2}{\partial {s_2}^2} > 0 \qquad D \geq 0$$

The case of $D = 0$ occurs when the edges are parallel. This is because all points along the edge return the same distance. In this case it will be necessary to clamp the point to one of the vertices.

Returning to the problem, the distance is minimised by setting the first derivatives to zero and solving the set of simultaneous equations

$$\begin{aligned} & s_1 \mathbf{E}_{10} \bullet \mathbf{E}_{10} - s_2 \mathbf{E}_{10} \bullet \mathbf{E}_{20} + \mathbf{E}_{10} \bullet \mathbf{Q} = 0 \\ & s_2 \mathbf{E}_{20} \bullet \mathbf{E}_{20} - s_1 \mathbf{E}_{10} \bullet \mathbf{E}_{20} - \mathbf{E}_{20} \bullet \mathbf{Q} = 0 \end{aligned}$$

$$\begin{pmatrix} \mathbf{E}_{10} \bullet \mathbf{E}_{10} & -\mathbf{E}_{10} \bullet \mathbf{E}_{20} \\ -\mathbf{E}_{10} \bullet \mathbf{E}_{20} & \mathbf{E}_{20} \bullet \mathbf{E}_{20} \end{pmatrix} \begin{pmatrix} s_1 \\ s_2 \end{pmatrix} = \begin{pmatrix} -\mathbf{E}_{10} \bullet \mathbf{Q} \\ \mathbf{E}_{20} \bullet \mathbf{Q} \end{pmatrix}$$

Let

$$\mathbf{H}_1 = \begin{pmatrix} \mathbf{E}_{10} \bullet \mathbf{E}_{10} \\ -\mathbf{E}_{10} \bullet \mathbf{E}_{20} \end{pmatrix} \qquad \mathbf{H}_2 = \begin{pmatrix} -\mathbf{E}_{10} \bullet \mathbf{E}_{20} \\ \mathbf{E}_{20} \bullet \mathbf{E}_{20} \end{pmatrix} \qquad \mathbf{K} = \begin{pmatrix} -\mathbf{E}_{10} \bullet \mathbf{Q} \\ \mathbf{E}_{20} \bullet \mathbf{Q} \end{pmatrix}$$

Then the equation can be written simply as

$$\mathbf{H}_1 s_1 + \mathbf{H}_2 s_2 = \mathbf{K}$$

$$\underbrace{\begin{pmatrix} H_{1x} & H_{2x} \\ H_{1y} & H_{2y} \end{pmatrix}}_{\mathbf{A}} \begin{pmatrix} s_1 \\ s_2 \end{pmatrix} = \begin{pmatrix} K_x \\ K_y \end{pmatrix}$$

The solution using Cramer's rule follows

$$s_1 = \frac{\begin{vmatrix} K_x & H_{2x} \\ K_y & H_{2y} \end{vmatrix}}{|\mathbf{A}|} = \frac{\mathbf{K} \times \mathbf{H}_2}{\mathbf{H}_1 \times \mathbf{H}_2}$$

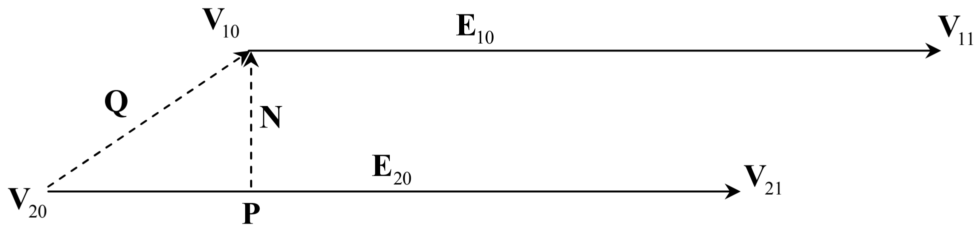
$$s_2 = \frac{\begin{vmatrix} H_{1x} & K_x \\ H_{1y} & K_y \end{vmatrix}}{|\mathbf{A}|} = \frac{\mathbf{H}_1 \times \mathbf{K}}{\mathbf{H}_1 \times \mathbf{H}_2}$$

The parameters must be clamped between 0 and 1 (to remain on the edges). Computationally this may be done as follows

$$s_i = \max\left(\min\left(s_i, 1\right), 0\right)$$

Now these bounded values of s_1 and s_2 are inserted into the equation for *d* and the result is added to the list of separation distances. This procedure is carried out for all edge pair combinations.

Before evaluating the parameters it is necessary to check whether $\mathbf{H}_1 \times \mathbf{H}_2 = 0$. This occurs when the edges are parallel and requires special treatment as shown below



$$\mathbf{N} = \mathbf{Q} - \left(\mathbf{Q} \bullet \frac{\mathbf{E}_{20}}{\|\mathbf{E}_{20}\|}\right)\frac{\mathbf{E}_{20}}{\|\mathbf{E}_{20}\|} = \mathbf{Q} - \frac{\left(\mathbf{Q} \bullet \mathbf{E}_{20}\right)\mathbf{E}_{20}}{\mathbf{E}_{20} \bullet \mathbf{E}_{20}}$$

$$d = \|\mathbf{N}\|$$

It is necessary to check whether the normally projected point **P** lies on the edge. This can be done by checking that the edge-directed distance of **P** from the edge starting point is both positive and less than the edge length

$$\mathbf{P} = \mathbf{V}_{10} - \mathbf{N}$$

$$0 \le \left(\mathbf{P} - \mathbf{V}_{20}\right) \bullet \frac{\mathbf{E}_{20}}{\|\mathbf{E}_{20}\|} \le \|\mathbf{E}_{20}\|$$

$$0 \le \left(\mathbf{P} - \mathbf{V}_{20}\right) \bullet \mathbf{E}_{20} \le \left(\mathbf{E}_{20} \bullet \mathbf{E}_{20}\right)$$

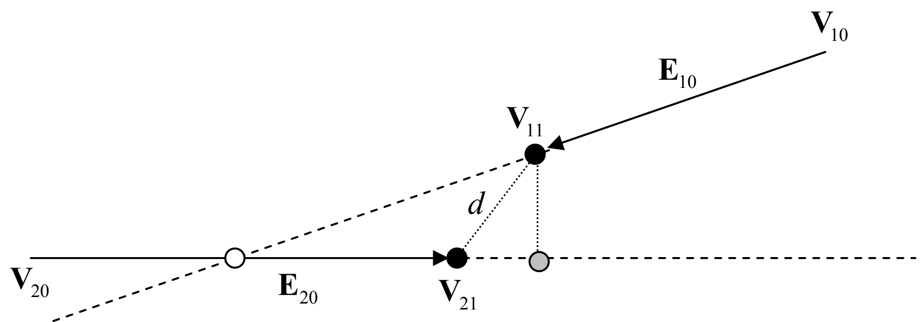
The final expression avoids the need to compute the edge unit vector and length. This will need to be checked for vertex $\mathbf{V}_{11}$ as well, and then carried out on the vertices of edge 2 against edge 1. For clarity the cases are explicitly listed below

$$\mathbf{P} = \mathbf{V}_{10} - \mathbf{N} \qquad 0 \le (\mathbf{P} - \mathbf{V}_{20}) \bullet \mathbf{E}_{20} \le (\mathbf{E}_{20} \bullet \mathbf{E}_{20})$$
$$\mathbf{P} = \mathbf{V}_{11} - \mathbf{N} \qquad 0 \le (\mathbf{P} - \mathbf{V}_{20}) \bullet \mathbf{E}_{20} \le (\mathbf{E}_{20} \bullet \mathbf{E}_{20})$$
$$\mathbf{P} = \mathbf{V}_{20} + \mathbf{N} \qquad 0 \le (\mathbf{P} - \mathbf{V}_{10}) \bullet \mathbf{E}_{10} \le (\mathbf{E}_{10} \bullet \mathbf{E}_{10})$$
$$\mathbf{P} = \mathbf{V}_{21} + \mathbf{N} \qquad 0 \le (\mathbf{P} - \mathbf{V}_{10}) \bullet \mathbf{E}_{10} \le (\mathbf{E}_{10} \bullet \mathbf{E}_{10})$$

As soon as one of these conditions is true, then exit the check and return *d*. If all are false, then the distance value is invalid and instead must be obtained by a vertex – vertex distance calculation, shown below.

3. Vertex – vertex distance:

Consider the following case:



The vertex – triangle distance test will fail as the normal projection of $\mathbf{V}_{11}$ onto edge $\mathbf{E}_{20}$ (grey dot) is not on the edge (and therefore not within the triangle). The edge – edge minimum distance test will return a value that is not actually the minimum distance (white dot) and so needs to be supplemented by a simple vertex – vertex distance test, which, in the present example will return the correct value for minimum separation

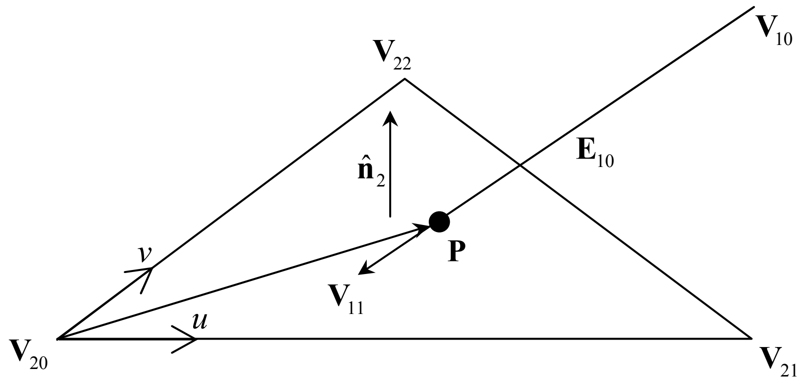
$$d_{ij} = \left\| \mathbf{V}_{1i} - \mathbf{V}_{2j} \right\| \qquad i, j = 0,1,2$$

Finally, the minimum value contained in the array of separation distances can be returned (either by using a MIN function, or by sorting the array in ascending order and returning the first value).

For practical applications it may be more useful to return the distance *vector* rather than the scalar. This would provide additional information regarding the directional components of minimum separation, particularly useful when traversing in a Cartesian coordinate system. This would require an additional corresponding array of distance vectors which can be indexed by the index of the minimum value found in the scalar distance array. It may be more computationally efficient to work with the distance squared as this would eliminate the need to compute square roots after the distance vector is dot-producted with itself.

The convention of the distance direction vector is the direction going FROM triangle 2 TO triangle 1.

ALTERNATIVE POINT-IN-TRIANGLE TEST:



This is done using homogeneous barycentric coordinates

$$w\mathbf{V}_{20} + u\mathbf{V}_{21} + v\mathbf{V}_{22} = \mathbf{P}$$

$$u + v + w = 1$$

Set $w = 1 - u - v$ and eliminate w to give

$$\begin{aligned}(1-u-v)\mathbf{V}_{20} + u\mathbf{V}_{21} + v\mathbf{V}_{22} &= \mathbf{P}\\ \mathbf{V}_{20} + u(\mathbf{V}_{21} - \mathbf{V}_{20}) + v(\mathbf{V}_{22} - \mathbf{V}_{20}) &= \mathbf{P}\end{aligned}$$

Let

$$\begin{aligned}\mathbf{E}_{20} &= \mathbf{V}_{21} - \mathbf{V}_{20}\\ -\mathbf{E}_{22} &= \mathbf{V}_{22} - \mathbf{V}_{20}\\ \mathbf{Q} &= \mathbf{P} - \mathbf{V}_{20}\end{aligned}$$

Then

$$u\mathbf{E}_{20} - v\mathbf{E}_{22} = \mathbf{Q}$$

The cross product cannot be used in 3D as vector division is undefined. Therefore begin by writing the equation in matrix form

$$\underbrace{\begin{pmatrix} E_{20x} & E_{22x} \\ E_{20y} & E_{22y} \\ E_{20z} & E_{22z} \end{pmatrix}}_{\mathbf{A}} \underbrace{\begin{pmatrix} u \\ -v \end{pmatrix}}_{\mathbf{X}} = \begin{pmatrix} Q_x \\ Q_y \\ Q_z \end{pmatrix}$$

$$\mathbf{AX} = \mathbf{Q}$$

Since **A** is non-square, this must be solved as follows

$$\left(\mathbf{A}^{\mathrm{T}}\mathbf{A}\right)\mathbf{X} = \mathbf{A}^{\mathrm{T}}\mathbf{Q}$$

$$\mathbf{X} = \left(\mathbf{A}^{\mathrm{T}}\mathbf{A}\right)^{-1}\left(\mathbf{A}^{\mathrm{T}}\mathbf{Q}\right)$$

The point is inside the triangle (including on the edges) if

$$\left(u > -\varepsilon\right) \ \wedge \ \left(v > -\varepsilon\right) \ \wedge \ \left(u + v < 1 + \varepsilon\right)$$

By writing out the solution step-by-step it may be possible to computationally simplify the procedure and avoid the matrix inversion step:

$$\begin{pmatrix} E_{20x} & E_{20y} & E_{20z} \\ E_{22x} & E_{22y} & E_{22z} \end{pmatrix}\begin{pmatrix} E_{20x} & E_{22x} \\ E_{20y} & E_{22y} \\ E_{20z} & E_{22z} \end{pmatrix}\begin{pmatrix} u \\ -v \end{pmatrix} = \begin{pmatrix} E_{20x} & E_{20y} & E_{20z} \\ E_{22x} & E_{22y} & E_{22z} \end{pmatrix}\begin{pmatrix} Q_x \\ Q_y \\ Q_z \end{pmatrix}$$

$$\begin{pmatrix} \mathbf{E}_{20} \bullet \mathbf{E}_{20} & \mathbf{E}_{20} \bullet \mathbf{E}_{22} \\ \mathbf{E}_{20} \bullet \mathbf{E}_{22} & \mathbf{E}_{22} \bullet \mathbf{E}_{22} \end{pmatrix}\begin{pmatrix} u \\ -v \end{pmatrix} = \begin{pmatrix} \mathbf{E}_{20} \bullet \mathbf{Q} \\ \mathbf{E}_{22} \bullet \mathbf{Q} \end{pmatrix}$$

$$u = \frac{\left(\mathbf{E}_{22} \bullet \mathbf{E}_{22}\right)\left(\mathbf{E}_{20} \bullet \mathbf{Q}\right) - \left(\mathbf{E}_{20} \bullet \mathbf{E}_{22}\right)\left(\mathbf{E}_{22} \bullet \mathbf{Q}\right)}{\left(\mathbf{E}_{20} \bullet \mathbf{E}_{20}\right)\left(\mathbf{E}_{22} \bullet \mathbf{E}_{22}\right) - \left(\mathbf{E}_{20} \bullet \mathbf{E}_{22}\right)\left(\mathbf{E}_{20} \bullet \mathbf{E}_{22}\right)}$$

$$v = \frac{\left(\mathbf{E}_{20} \bullet \mathbf{E}_{22}\right)\left(\mathbf{E}_{20} \bullet \mathbf{Q}\right) - \left(\mathbf{E}_{20} \bullet \mathbf{E}_{20}\right)\left(\mathbf{E}_{22} \bullet \mathbf{Q}\right)}{\left(\mathbf{E}_{20} \bullet \mathbf{E}_{20}\right)\left(\mathbf{E}_{22} \bullet \mathbf{E}_{22}\right) - \left(\mathbf{E}_{20} \bullet \mathbf{E}_{22}\right)\left(\mathbf{E}_{20} \bullet \mathbf{E}_{22}\right)}$$

Note the denominator is only a function of the triangle properties and so can be pre-stored. To match the 2D formulation of this problem, define the following variables

$$\mathbf{F}_{20} = \begin{pmatrix} \mathbf{E}_{20} \bullet \mathbf{E}_{20} \\ \mathbf{E}_{20} \bullet \mathbf{E}_{22} \end{pmatrix} \qquad \mathbf{F}_{22} = \begin{pmatrix} \mathbf{E}_{20} \bullet \mathbf{E}_{22} \\ \mathbf{E}_{22} \bullet \mathbf{E}_{22} \end{pmatrix} \qquad \mathbf{G} = \begin{pmatrix} \mathbf{E}_{20} \bullet \mathbf{Q} \\ \mathbf{E}_{22} \bullet \mathbf{Q} \end{pmatrix}$$

The problem can now be re-written as

$$u\mathbf{F}_{20} - v\mathbf{F}_{22} = \mathbf{G}$$

$$\underbrace{\begin{pmatrix} F_{20x} & F_{22x} \\ F_{20y} & F_{22y} \end{pmatrix}}_{\mathbf{B}}\begin{pmatrix} u \\ -v \end{pmatrix} = \begin{pmatrix} G_x \\ G_y \end{pmatrix}$$

and solved for u and v using Cramer's rule

$$u = \frac{\begin{vmatrix} G_x & F_{22x} \\ G_y & F_{22y} \end{vmatrix}}{|\mathbf{B}|} = \frac{\mathbf{G} \times \mathbf{F}_{22}}{\mathbf{F}_{20} \times \mathbf{F}_{22}}$$

$$v = -\frac{\begin{vmatrix} F_{20x} & G_x \\ F_{20y} & G_y \end{vmatrix}}{|\mathbf{B}|} = \frac{\mathbf{G} \times \mathbf{F}_{20}}{\mathbf{F}_{20} \times \mathbf{F}_{22}}$$

Note that the 2D cross product is used here, which returns a signed scalar.