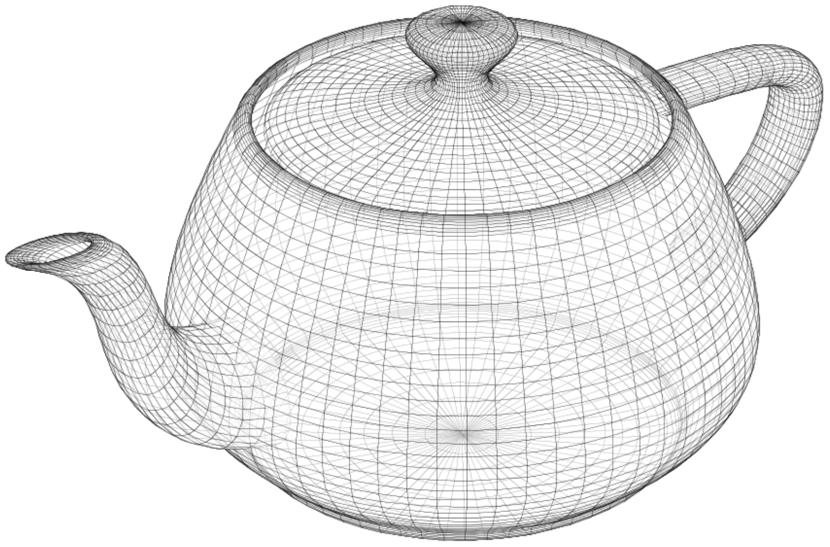


Barycentric Coordinates

Ray-Triangle Intersection

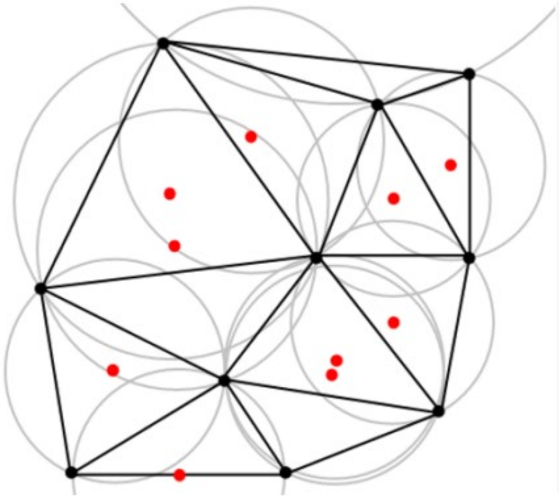
Production Computer Graphics

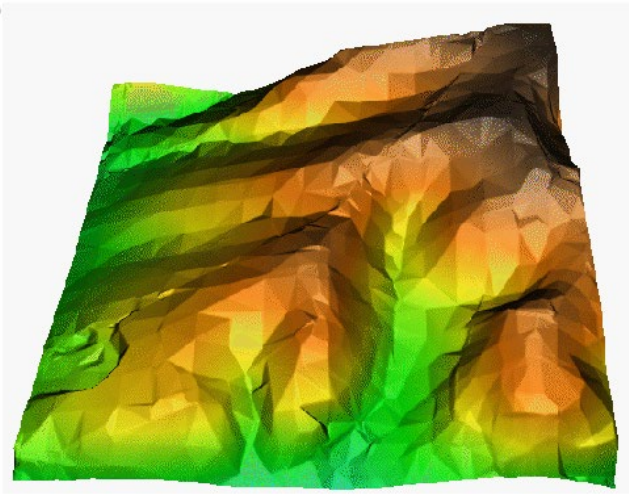
Professor Eric Shaffer

ILLINOIS

Barycentric Interpolation

How can we linearly interpolate a function over triangles?

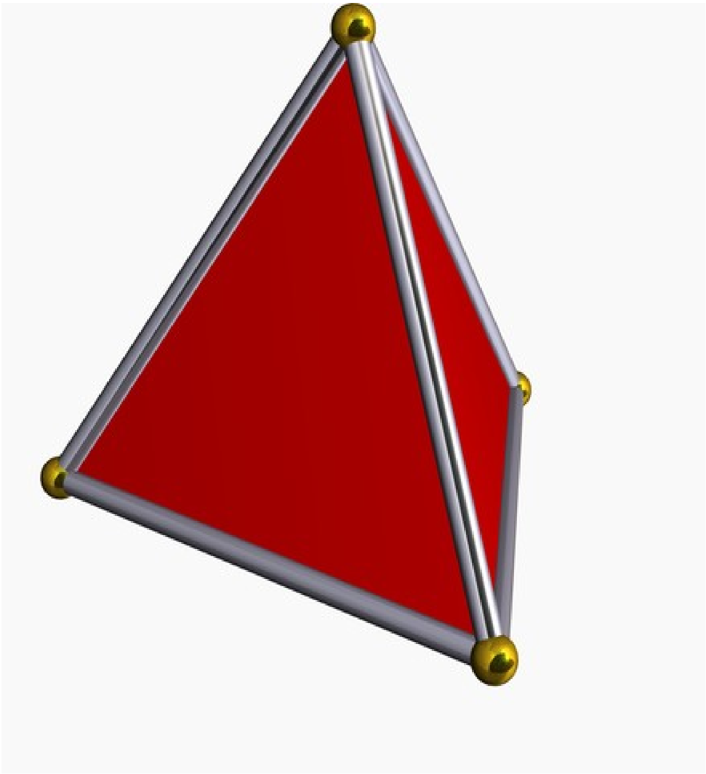




Can be useful in graphics for what purposes?

Barycentric Interpolation

- Barycentric coordinates apply to more than just triangles
- Used on any simplex
- A simplex is a convex hull of k+1 points in a k-dimensional space
  - Simplest convex “polygon” in a k-dimensional space
  - A 3-simplex is a triangle
- Barycentric coordinates provide a way to interpolate over simplices



Barycentric Coordinates for Triangles

Describe location of a point in relation to the vertices of a given triangle

Express point p in barycentric coordinates $p=(\lambda_1, \lambda_2, \lambda_3)$

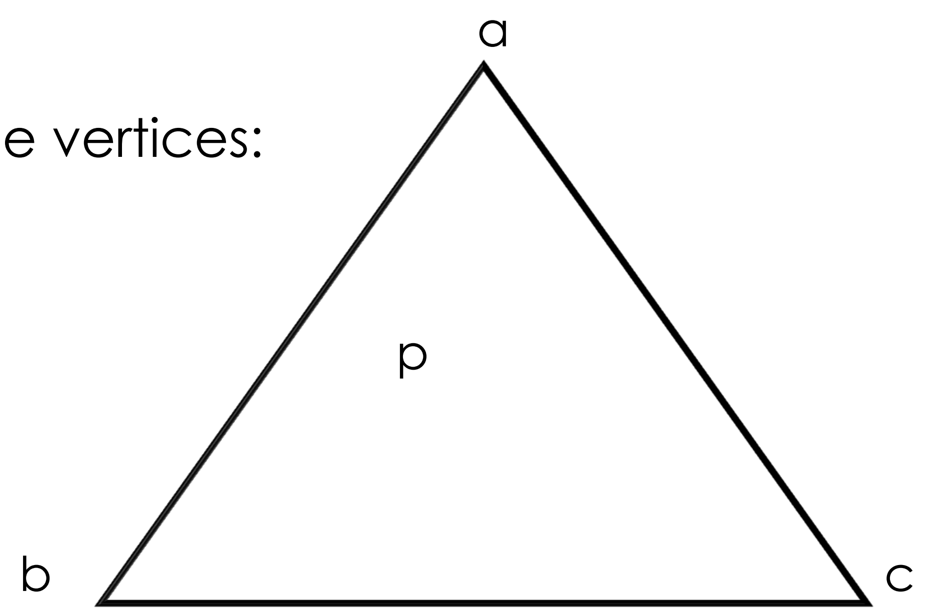
The following must be true

$p=\lambda_1 a + \lambda_2 b + \lambda_3 c$

$\lambda_1 + \lambda_2 + \lambda_3 = 1$

To interpolate a function sampled at the vertices:

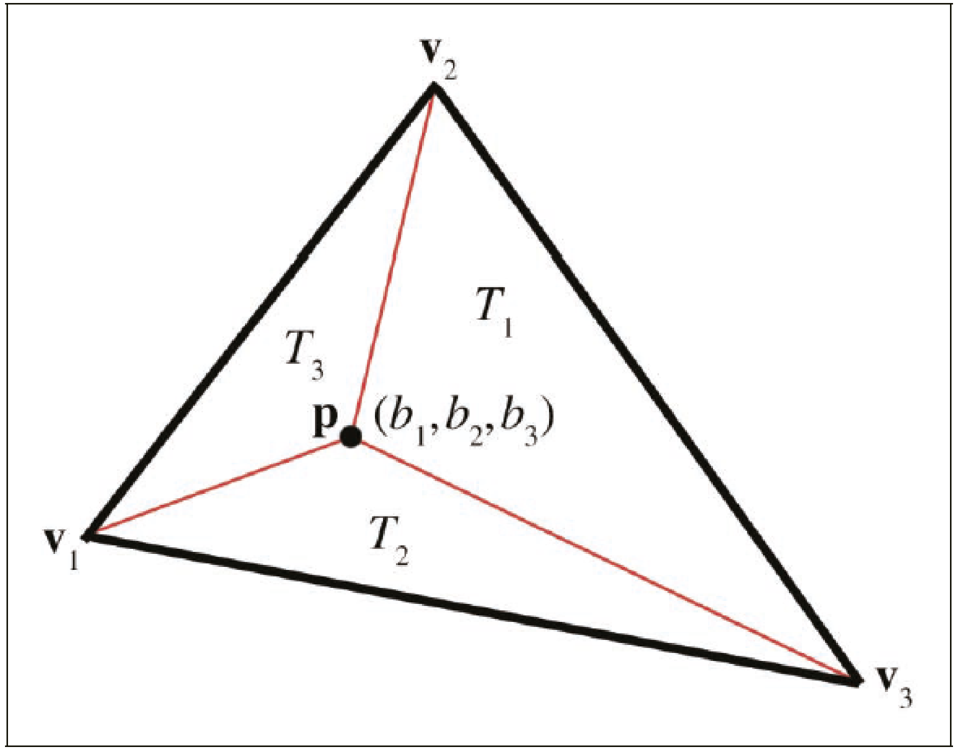
$f(p)=\lambda_1 f(a) + \lambda_2 f(b) + \lambda_3 f(c)$



ILLINOIS

Computing Barycentric Coordinates for Triangles

Coordinates are the signed area of the opposite subtriangle divided by area of the triangle



$$\begin{aligned} b_1x_1 + b_2x_2 + b_3x_3 &= p_x, \\ b_1y_1 + b_2y_2 + b_3y_3 &= p_y, \\ b_1 + b_2 + b_3 &= 1. \end{aligned}$$

$$b_1 = \frac{(p_y - y_3)(x_2 - x_3) + (y_2 - y_3)(x_3 - p_x)}{(y_1 - y_3)(x_2 - x_3) + (y_2 - y_3)(x_3 - x_1)},$$

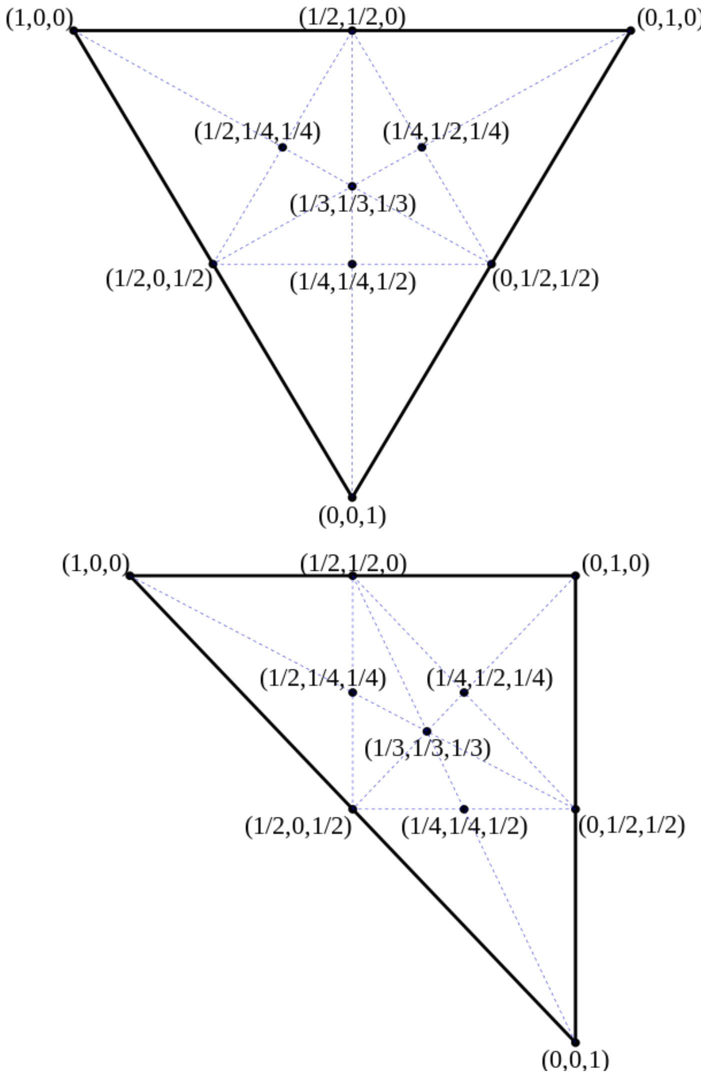
$$b_2 = \frac{(p_y - y_1)(x_3 - x_1) + (y_3 - y_1)(x_1 - p_x)}{(y_1 - y_3)(x_2 - x_3) + (y_2 - y_3)(x_3 - x_1)},$$

$$b_3 = \frac{(p_y - y_2)(x_1 - x_2) + (y_1 - y_2)(x_2 - p_x)}{(y_1 - y_3)(x_2 - x_3) + (y_2 - y_3)(x_3 - x_1)}.$$

$$b_1 = A(T_1)/A(T), \qquad b_2 = A(T_2)/A(T), \qquad b_3 = A(T_3)/A(T)$$

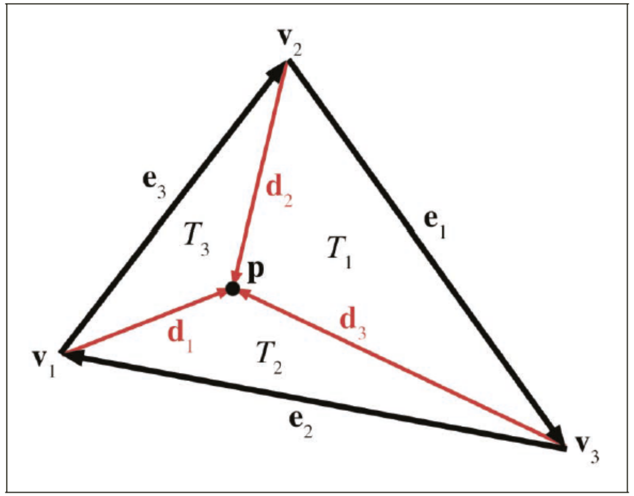
ILLINOIS

Some Important Points



ILLINOIS

Computing Coordinates for 3D Triangles



$$\mathbf{e}_1 = \mathbf{v}_3 - \mathbf{v}_2, \quad \mathbf{e}_2 = \mathbf{v}_1 - \mathbf{v}_3, \quad \mathbf{e}_3 = \mathbf{v}_2 - \mathbf{v}_1,$$
$$\mathbf{d}_1 = \mathbf{p} - \mathbf{v}_1, \quad \mathbf{d}_2 = \mathbf{p} - \mathbf{v}_2, \quad \mathbf{d}_3 = \mathbf{p} - \mathbf{v}_3.$$

$$\hat{\mathbf{n}} = \frac{\mathbf{e}_1 \times \mathbf{e}_2}{\|\mathbf{e}_1 \times \mathbf{e}_2\|}.$$

$$A(T) = ((\mathbf{e}_1 \times \mathbf{e}_2) \cdot \hat{\mathbf{n}})/2,$$
$$A(T_1) = ((\mathbf{e}_1 \times \mathbf{d}_3) \cdot \hat{\mathbf{n}})/2,$$
$$A(T_2) = ((\mathbf{e}_2 \times \mathbf{d}_1) \cdot \hat{\mathbf{n}})/2,$$
$$A(T_3) = ((\mathbf{e}_3 \times \mathbf{d}_2) \cdot \hat{\mathbf{n}})/2.$$

$$b_1 = A(T_1)/A(T) = \frac{(\mathbf{e}_1 \times \mathbf{d}_3) \cdot \hat{\mathbf{n}}}{(\mathbf{e}_1 \times \mathbf{e}_2) \cdot \hat{\mathbf{n}}},$$
$$b_2 = A(T_2)/A(T) = \frac{(\mathbf{e}_2 \times \mathbf{d}_1) \cdot \hat{\mathbf{n}}}{(\mathbf{e}_1 \times \mathbf{e}_2) \cdot \hat{\mathbf{n}}},$$
$$b_3 = A(T_3)/A(T) = \frac{(\mathbf{e}_3 \times \mathbf{d}_2) \cdot \hat{\mathbf{n}}}{(\mathbf{e}_1 \times \mathbf{e}_2) \cdot \hat{\mathbf{n}}}.$$

Barycentric Coordinates for Tetrahedra

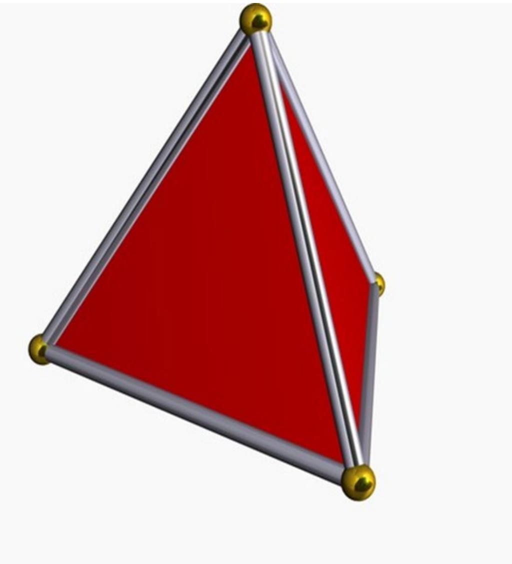
We have 4 vertices of a tetrahedron r_1, r_2, r_3, and r_4

To find the coordinates of a point r we can compute

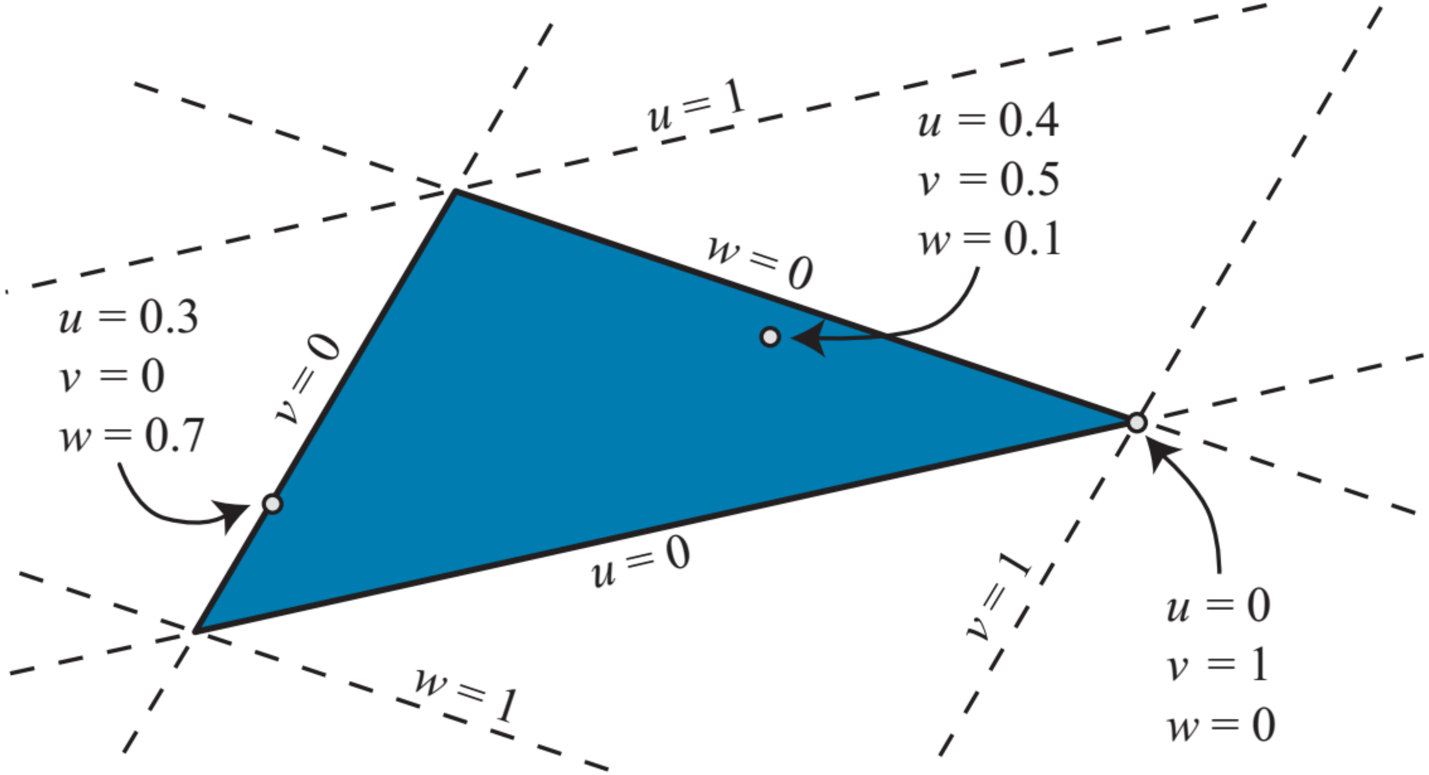
$$\begin{pmatrix}\lambda_1\\ \lambda_2\\ \lambda_3\end{pmatrix} = \mathbf{T}^{-1}(\mathbf{r}-\mathbf{r}_4)$$

$$\mathbf{T} = \begin{pmatrix} x_1 - x_4 & x_2 - x_4 & x_3 - x_4 \\ y_1 - y_4 & y_2 - y_4 & y_3 - y_4 \\ z_1 - z_4 & z_2 - z_4 & z_3 - z_4 \end{pmatrix}$$

and $\lambda_4 = 1 - \lambda_1 - \lambda_2 - \lambda_3$



Ray Triangle Intersection Test



How can we test if a point is in a triangle?

Intersecting a Ray and Triangle

A point on triangle is given as $(u, v, 1 - u - v)$ in barycentric coordinates.

$$\mathbf{f}(u, v) = (1 - u - v)\mathbf{p}_0 + u\mathbf{p}_1 + v\mathbf{p}_2$$

The intersection with a ray is:

$$\mathbf{o} + t\mathbf{d} = (1 - u - v)\mathbf{p}_0 + u\mathbf{p}_1 + v\mathbf{p}_2$$

Linear Algebra!

Rearranging terms gives us

$$\begin{pmatrix} -\mathbf{d} & \mathbf{p}_1 - \mathbf{p}_0 & \mathbf{p}_2 - \mathbf{p}_0 \end{pmatrix} \begin{pmatrix} t \\ u \\ v \end{pmatrix} = \mathbf{o} - \mathbf{p}_0$$

This is a system of equations…3 equations and 3 unknowns

Solve the System

$$\mathbf{e}_1 = \mathbf{p}_1 - \mathbf{p}_0,\ \mathbf{e}_2 = \mathbf{p}_2 - \mathbf{p}_0,\ \text{and } \mathbf{s} = \mathbf{o} - \mathbf{p}_0$$

$$\begin{pmatrix} t \\ u \\ v \end{pmatrix} = \frac{1}{\det(-\mathbf{d}, \mathbf{e}_1, \mathbf{e}_2)} \begin{pmatrix} \det(\mathbf{s}, \mathbf{e}_1, \mathbf{e}_2) \\ \det(-\mathbf{d}, \mathbf{s}, \mathbf{e}_2) \\ \det(-\mathbf{d}, \mathbf{e}_1, \mathbf{s}) \end{pmatrix}$$

Using Cramer’s Rule (don’t tell any numerical analysts you know…)

Why not? Well, computational cost for n x n system is O(n! x n) and Gaussian Elimation is O(n^3)

Here, it’s only 3x3…so maybe OK although it can still be less numerically stable than the GE

Solve the System

We know $\det(\mathbf{a}, \mathbf{b}, \mathbf{c}) = |\mathbf{a}\ \ \mathbf{b}\ \ \mathbf{c}| = -(\mathbf{a}\times\mathbf{c})\cdot\mathbf{b} = -(\mathbf{c}\times\mathbf{b})\cdot\mathbf{a}$

So
$$\begin{pmatrix} t \\ u \\ v \end{pmatrix} = \frac{1}{(\mathbf{d}\times\mathbf{e}_2)\cdot\mathbf{e}_1}\begin{pmatrix} (\mathbf{s}\times\mathbf{e}_1)\cdot\mathbf{e}_2 \\ (\mathbf{d}\times\mathbf{e}_2)\cdot\mathbf{s} \\ (\mathbf{s}\times\mathbf{e}_1)\cdot\mathbf{d} \end{pmatrix} = \frac{1}{\mathbf{q}\cdot\mathbf{e}_1}\begin{pmatrix} \mathbf{r}\cdot\mathbf{e}_2 \\ \mathbf{q}\cdot\mathbf{s} \\ \mathbf{r}\cdot\mathbf{d} \end{pmatrix}$$

ILLINOIS

Alternatively…

$$\begin{pmatrix} t \\ u \\ v \end{pmatrix} = \frac{1}{(\mathbf{d} \times \mathbf{e}_2) \cdot \mathbf{e}_1} \begin{pmatrix} (\mathbf{s} \times \mathbf{e}_1) \cdot \mathbf{e}_2 \\ (\mathbf{d} \times \mathbf{e}_2) \cdot \mathbf{s} \\ (\mathbf{s} \times \mathbf{e}_1) \cdot \mathbf{d} \end{pmatrix}$$

$$= \frac{1}{-(\mathbf{e}_1 \times \mathbf{e}_2) \cdot \mathbf{d}} \begin{pmatrix} (\mathbf{e}_1 \times \mathbf{e}_2) \cdot \mathbf{s} \\ (\mathbf{s} \times \mathbf{d}) \cdot \mathbf{e}_2 \\ -(\mathbf{s} \times \mathbf{d}) \cdot \mathbf{e}_1 \end{pmatrix} = \frac{1}{-\mathbf{n} \cdot \mathbf{d}} \begin{pmatrix} \mathbf{n} \cdot \mathbf{s} \\ \mathbf{m} \cdot \mathbf{e}_2 \\ -\mathbf{m} \cdot \mathbf{e}_1 \end{pmatrix}$$

Useful if you already have ***n*** the unnormalized normal of the triangle

The Algorithm

$\textbf{RayTriIntersect}(\mathbf{o}, \mathbf{d}, \mathbf{p}_0, \mathbf{p}_1, \mathbf{p}_2)$
$\texttt{returns}\ (\{\texttt{REJECT}, \texttt{INTERSECT}\}, \texttt{u}, \texttt{v}, \texttt{t});$

1 : $\mathbf{e}_1 = \mathbf{p}_1 - \mathbf{p}_0$
2 : $\mathbf{e}_2 = \mathbf{p}_2 - \mathbf{p}_0$
3 : $\mathbf{q} = \mathbf{d} \times \mathbf{e}_2$
4 : $a = \mathbf{e}_1 \cdot \mathbf{q}$
5 : $\texttt{if}(a > -\epsilon\ \texttt{and}\ a < \epsilon)\ \texttt{return}\ (\texttt{REJECT}, 0, 0, 0);$
6 : $f = 1/a$
7 : $\mathbf{s} = \mathbf{o} - \mathbf{p}_0$
8 : $u = f(\mathbf{s} \cdot \mathbf{q})$
9 : $\texttt{if}(u < 0.0)\ \texttt{return}\ (\texttt{REJECT}, 0, 0, 0);$
10 : $\mathbf{r} = \mathbf{s} \times \mathbf{e}_1$
11 : $v = f(\mathbf{d} \cdot \mathbf{r})$
12 : $\texttt{if}(v < 0.0\ \texttt{or}\ u + v > 1.0)\ \texttt{return}\ (\texttt{REJECT}, 0, 0, 0);$
13 : $t = f(\mathbf{e}_2 \cdot \mathbf{r})$
14 : $\texttt{return}\ (\texttt{INTERSECT}, u, v, t);$

Line 4 computes determinant of matrix M.

Then test to avoid determinants close to zero.

Good choice $\epsilon = 10^{-5}$

Get the Code

Möller, Tomas, and Ben Trumbore, "Fast, Minimum Storage Ray-Triangle Intersection," *journal of graphics tools*, vol. 2, no. 1, pp. 21–28, 1997. Also collected in [112]. Cited on p. 962, 965

https://en.wikipedia.org/wiki/M%C3%B6ller%E2%80%93Trumbore_intersection_algorithm

Article Talk Read Edit View history Search Wikipedia

Möller–Trumbore intersection algorithm

From Wikipedia, the free encyclopedia

The **Möller–Trumbore ray-triangle intersection algorithm**, named after its inventors Tomas Möller and Ben Trumbore, is a fast method for calculating the intersection of a ray and a triangle in three dimensions without needing precomputation of the plane equation of the plane containing the triangle.[1] Among other uses, it can be used in computer graphics to implement ray tracing computations involving triangle meshes.[2]

Contents [hide]