

1 Ray-Surface Intersection

When rendering a 3D object, we must determine which parts of it are visible, where shadows fall, and how lighting interacts within the scene. A straightforward yet computationally expensive approach is to cast rays from the camera through each pixel, finding where those rays intersect the object's mesh. In general, a ray can intersect the mesh zero times, once, multiple times, or—if it lies exactly in a triangle's plane—infininitely many times.

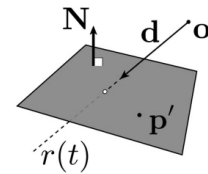
Recall that a ray is defined by its origin \mathbf{o} and its direction vector \mathbf{d} , and is parameterized by $t \geq 0$:

$$\mathbf{r}(t) = \mathbf{o} + t\mathbf{d}.$$

1. As a warm-up, let's re-derive the equation for a ray intersecting an arbitrary plane. Recall that a plane can be defined as the set of all points \mathbf{p} satisfying

$$(\mathbf{p} - \mathbf{p}') \cdot \mathbf{N} = 0,$$

where \mathbf{p}' is any point on the plane and \mathbf{N} is the plane's normal vector. Set \mathbf{p} equal to $\mathbf{r}(t)$ and solve for t .



2. What does it mean if we get a value of $t < 0$?

3. What does it mean if $\mathbf{d} \cdot \mathbf{N} = 0$?

4. Given the following implicit representation of an ellipsoid and the definition of a ray, compute where (and at what parameter value(s) of t) the ray intersects the ellipsoid:

$$f(x, y, z) = \frac{(x - 2)^2}{4} + (y - 2)^2 + \frac{z^2}{4} - 1$$

$$\mathbf{r}(t) = (0, 0, 0) + t(1, 1, 0)$$

Start by substituting the ray $\mathbf{r}(t)$ into the function $f(x, y, z)$ to obtain $f(\mathbf{o} + t\mathbf{d})$.

2 Ray-Triangle Intersection

Given a mesh representation of an object, we would like to render it onto a display. To do so, we need to know which parts of the object are visible, where to put shadows, how to apply the scene's lighting, and more. The simplest idea to handle these problems is to take a ray and intersect it with each triangle in the mesh.

Recall that a ray is defined by its origin \mathbf{O} and a direction vector \mathbf{D} and varies with "time" t for $0 \leq t < \infty$.

$$\mathbf{r}(t) = \mathbf{O} + t\mathbf{D}. \quad (1)$$

Recall that a point within a triangle $\mathbf{P}_0\mathbf{P}_1\mathbf{P}_2$ can be represented as

$$\mathbf{P} = \alpha\mathbf{P}_0 + \beta\mathbf{P}_1 + \gamma\mathbf{P}_2, \quad (2)$$

where $\alpha + \beta + \gamma = 1$. Defining $b_1 = \beta$ and $b_2 = \gamma$, we obtain $\alpha = 1 - b_1 - b_2$. Thus, we can rewrite the point \mathbf{P} in barycentric coordinates as:

$$\mathbf{P} = (1 - b_1 - b_2)\mathbf{P}_0 + b_1\mathbf{P}_1 + b_2\mathbf{P}_2. \quad (3)$$

1. Let's solve for the intersection of a ray and a triangle. Specifically, if we arrange the unknowns t, b_1 and b_2 into a column vector $\mathbf{x} = [t, b_1, b_2]^T$, can you get a matrix \mathbf{M} and a column vector \mathbf{b} so that $\mathbf{M}\mathbf{x} = \mathbf{b}$?

2. Now let's derive the **Möller-Trumbore algorithm!**

$$\begin{bmatrix} t \\ b_1 \\ b_2 \end{bmatrix} = \frac{1}{\mathbf{S}_1 \cdot \mathbf{E}_1} \begin{bmatrix} \mathbf{S}_2 \cdot \mathbf{E}_2 \\ \mathbf{S}_1 \cdot \mathbf{S} \\ \mathbf{S}_2 \cdot \mathbf{D} \end{bmatrix} \quad (7)$$

where $\mathbf{E}_1 = \mathbf{P}_1 - \mathbf{P}_0$, $\mathbf{E}_2 = \mathbf{P}_2 - \mathbf{P}_0$, $\mathbf{S} = \mathbf{O} - \mathbf{P}_0$, $\mathbf{S}_1 = \mathbf{D} \times \mathbf{E}_2$, $\mathbf{S}_2 = \mathbf{S} \times \mathbf{E}_1$.

Hint 1: (Cramer's rule) Linear equations $\mathbf{M}\mathbf{x} = \mathbf{b}$ can be simply solved using determinants of matrices as:

$$\mathbf{x} = \frac{1}{|\mathbf{M}|} \begin{bmatrix} |\mathbf{M}_1| \\ |\mathbf{M}_2| \\ |\mathbf{M}_3| \end{bmatrix}, \quad (8)$$

where \mathbf{M}_i is the matrix \mathbf{M} with its i -th column replaced by \mathbf{b} .

Hint 2: Suppose \mathbf{A} , \mathbf{B} , \mathbf{C} are column vectors, the determinant of the 3×3 matrix $[\mathbf{A}, \mathbf{B}, \mathbf{C}]$ satisfy:

$$|\mathbf{A}, \mathbf{B}, \mathbf{C}| = -(\mathbf{A} \times \mathbf{C}) \cdot \mathbf{B} = -(\mathbf{C} \times \mathbf{B}) \cdot \mathbf{A} = -(\mathbf{B} \times \mathbf{A}) \cdot \mathbf{C}. \quad (9)$$

3. Once you've solved for t , b_1 and b_2 , what conditions must be satisfied so that you have a valid ray-triangle intersection?

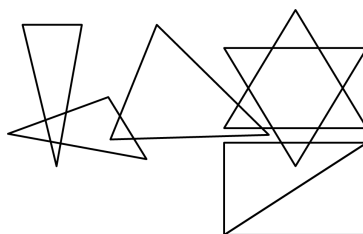
4. What does it mean when $\mathbf{S}_1 \cdot \mathbf{E}_1 = 0$ in the context of the Möller-Trumbore algorithm?

3 Building Beautifully Balanced Bounding Boxes

Bounding volumes are used to accelerate ray-triangle intersection tests. If a ray doesn't intersect a bounding volume, we can conclude that it doesn't intersect any triangles contained within.

A bounding volume hierarchy is a tree of bounding volumes. The bounding volume at a node encloses the bounding volumes of its children. The ray tracing algorithm traverses this hierarchy to determine if the ray intersects an object.

1. Given a set of planar triangles, build a BVH following these rules:
 - Always pick the longest axis to divide.
 - Use center of mass of triangles to decide their relative positions.
 - Keep the BVH balanced. Try to ensure the same number of triangles for children nodes.



2. A box has corners $(-2, -2, -2)$ and $(2, 2, 2)$. A ray has origin $(-3, 4, 5)$ and direction $(1, -1, -2)$. Compute the value(s) of t at which the ray intersects a yz -slab of the box.

3. Compute the value(s) of t at which the ray intersects a xz -slab of the box.

4. Compute the value(s) of t at which the ray intersects a xy -slab of the box.

5. Compute the entry and exit *points* of the ray for the given box.