Ray Tracing

Today

- **Basic algorithms**
- **Ray-surface intersection for single surface**

Next lecture

- **Acceleration techniques for ray tracing large numbers of geometric primitives**

Classic Ray Tracing

Greeks: Do light rays proceed from the eye to the light, or from the light to the eye?

Gauss: Rays through lenses

Three ideas about light

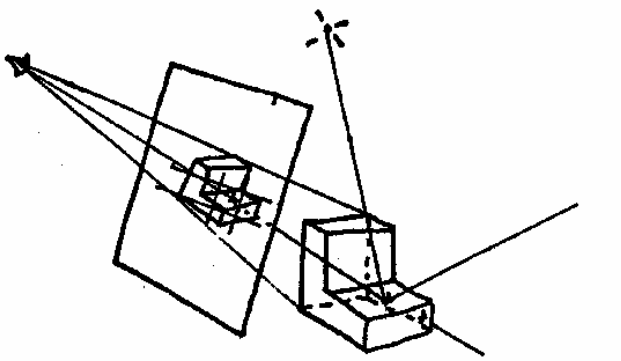
1. **Light rays travel in straight lines**
2. **Light rays do not interfere with each other if they cross**
3. **Light rays travel from the light sources to the eye (but the physics is invariant under path reversal - reciprocity).**

Ray Tracing in Computer Graphics

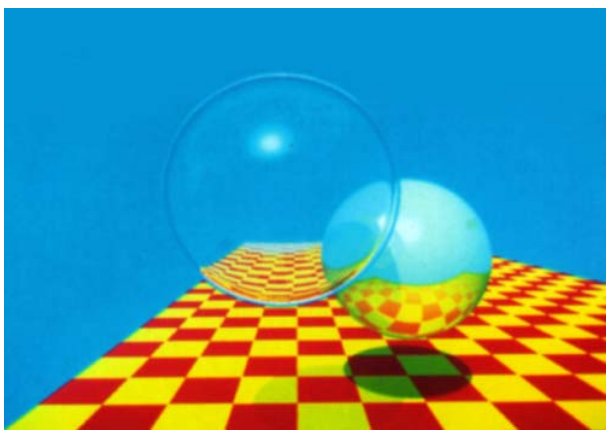
Appel 1968 - Ray casting

1. Generate an image by sending one ray per pixel

2. Check for shadows by sending a ray to the light



Ray Tracing in Computer Graphics

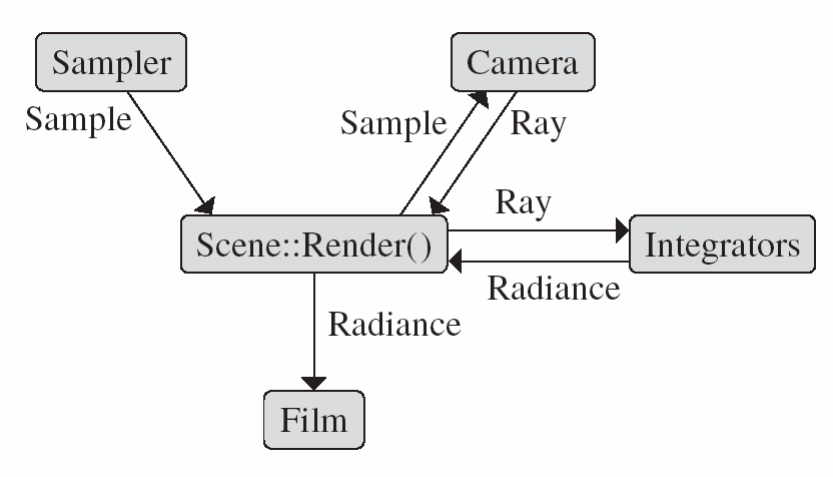


Whitted 1979

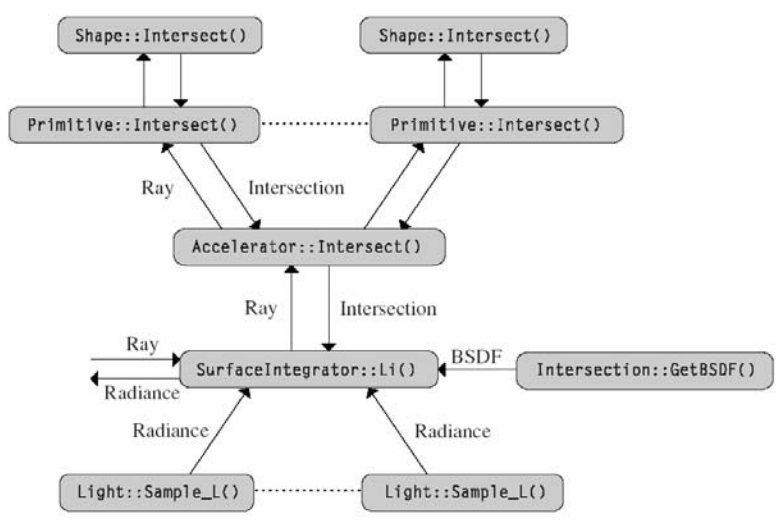
Recursive ray tracing (reflection and refraction)

Ray Tracing Demo

PBRT Architecture



PBRT Architecture

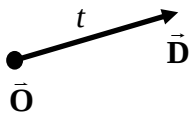


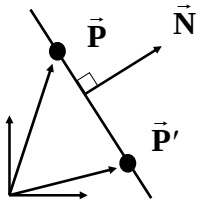
Ray-Surface Intersection

Ray-Plane Intersection

Ray: $\vec{\mathbf{P}} = \vec{\mathbf{O}} + t\,\vec{\mathbf{D}}$

$0 \le t < \infty$





Plane: $(\vec{\mathbf{P}} - \vec{\mathbf{P}}') \bullet \vec{\mathbf{N}} = 0$

$ax + by + cz + d = 0$

Solve for intersection

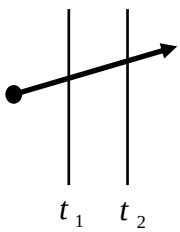
Substitute ray equation into plane equation

$$(\vec{\mathbf{P}} - \vec{\mathbf{P}}') \bullet \vec{\mathbf{N}} = (\vec{\mathbf{O}} + t\vec{\mathbf{D}} - \vec{\mathbf{P}}') \bullet \vec{\mathbf{N}} = 0$$

$$t = -\frac{(\vec{\mathbf{O}} - \vec{\mathbf{P}}') \bullet \vec{\mathbf{N}}}{\vec{\mathbf{D}} \bullet \vec{\mathbf{N}}}$$

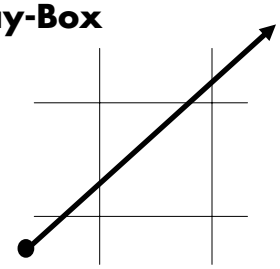
Ray-Polyhedra

Ray-Slab

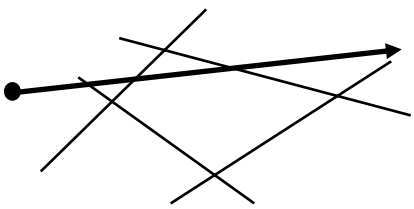


Note: Procedural geometry

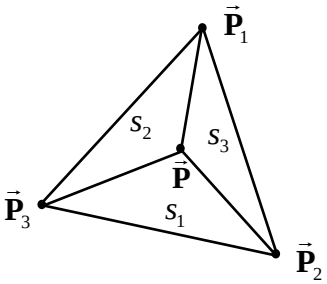
Ray-Box



Ray-Convex Polyhedra



Ray-Triangle Intersection 1



Barycentric coordinates

$$\vec{\mathbf{P}} = s_1\vec{\mathbf{P}}_1 + s_2\vec{\mathbf{P}}_2 + s_3\vec{\mathbf{P}}_3$$

Inside triangle criteria

$$0 \le s_1 \le 1$$
$$0 \le s_2 \le 1$$
$$0 \le s_3 \le 1$$
$$s_1 + s_2 + s_3 = 1$$

$$s_1 = \text{area}(\Delta \mathbf{P}\mathbf{P}_2\mathbf{P}_3)$$
$$s_2 = \text{area}(\Delta \mathbf{P}\mathbf{P}_3\mathbf{P}_1)$$
$$s_3 = \text{area}(\Delta \mathbf{P}\mathbf{P}_1\mathbf{P}_2)$$

Ray-Triangle Intersection 2

$$\vec{\mathbf{P}} = s_1\vec{\mathbf{P}}_1 + s_2\vec{\mathbf{P}}_2 + s_3\vec{\mathbf{P}}_3$$

$$\begin{bmatrix}\mathbf{P}_1 & \mathbf{P}_2 & \mathbf{P}_3\end{bmatrix}\begin{bmatrix}s_1\\ s_2\\ s_3\end{bmatrix} = \begin{bmatrix}\mathbf{P}\end{bmatrix}$$

$$\begin{bmatrix}s_1\\ s_2\\ s_3\end{bmatrix} = \begin{bmatrix}\mathbf{P}_2 \times \mathbf{P}_3\\ \mathbf{P}_3 \times \mathbf{P}_1\\ \mathbf{P}_1 \times \mathbf{P}_2\end{bmatrix}\begin{bmatrix}\mathbf{P}\end{bmatrix}$$

Ray-Triangle Intersection 3

$$\begin{bmatrix} s_1 \\ s_2 \\ s_3 \end{bmatrix} = \begin{bmatrix} \mathbf{P}_2 \times \mathbf{P}_3 \\ \mathbf{P}_3 \times \mathbf{P}_1 \\ \mathbf{P}_1 \times \mathbf{P}_2 \end{bmatrix} \begin{bmatrix} \mathbf{P} \end{bmatrix}$$

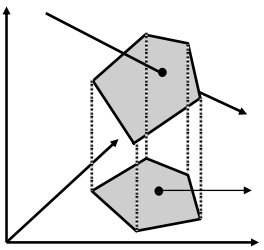
$$s_1 = \frac{\begin{vmatrix} \mathbf{P} & \mathbf{P}_2 & \mathbf{P}_3 \end{vmatrix}}{\begin{vmatrix} \mathbf{P}_1 & \mathbf{P}_2 & \mathbf{P}_3 \end{vmatrix}} = \mathbf{P} \bullet \frac{\mathbf{P}_2 \times \mathbf{P}_3}{\Delta}$$

$$s_2 = \frac{\begin{vmatrix} \mathbf{P}_1 & \mathbf{P} & \mathbf{P}_3 \end{vmatrix}}{\begin{vmatrix} \mathbf{P}_1 & \mathbf{P}_2 & \mathbf{P}_3 \end{vmatrix}} = \mathbf{P} \bullet \frac{\mathbf{P}_3 \times \mathbf{P}_1}{\Delta}$$

$$s_3 = \frac{\begin{vmatrix} \mathbf{P}_1 & \mathbf{P}_2 & \mathbf{P} \end{vmatrix}}{\begin{vmatrix} \mathbf{P}_1 & \mathbf{P}_2 & \mathbf{P}_3 \end{vmatrix}} = \mathbf{P} \bullet \frac{\mathbf{P}_1 \times \mathbf{P}_2}{\Delta}$$

Ray-Polygon Intersection

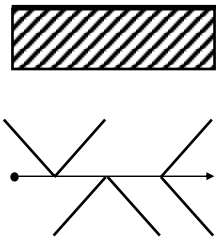
1. **Find intersection with plane of support**
2. **Test whether point is inside 3D polygon**
   a. **Project onto *xy* plane**
   b. **Test whether point is inside 2D polygon**



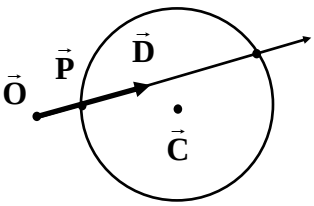
Point in Polygon

```
inside(vert v[], int n, float x, float y)
{
  int cross=0; float x0, y0, x1, y1;

  x0 = v[n-1].x - x;
  y0 = v[n-1].y - y;
  while( n-- ) {
    x1 = v->x - x;
    y1 = v->y - y;
    if( y0 > 0 ) {
      if( y1 <= 0 )
        if( x1*y0 > y1*x0 ) cross++;
    }
    else {
      if( y1 > 0 )
        if( x0*y1 > y0*x1 ) cross++;
    }
    x0 = x1; y0 = y1; v++;
  }
  return cross & 1;
}
```



Ray-Sphere Intersection



Ray: $\vec{\mathbf{P}} = \vec{\mathbf{O}} + t\vec{\mathbf{D}}$

Sphere: $\left(\vec{\mathbf{P}} - \vec{\mathbf{C}}\right)^2 - R^2 = 0$

$$\left(\vec{\mathbf{O}} + t\,\vec{\mathbf{D}} - \vec{\mathbf{C}}\right)^2 - R^2 = 0$$

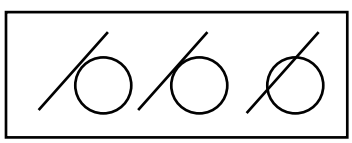
$$t = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$$

$$at^2 + bt + c = 0$$

$$a = \vec{\mathbf{D}}^2$$

$$b = 2(\vec{\mathbf{O}} - \vec{\mathbf{C}}) \bullet \vec{\mathbf{D}}$$

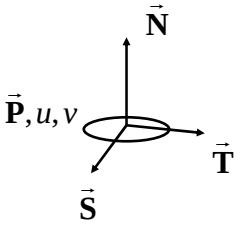
$$c = (\vec{\mathbf{O}} - \vec{\mathbf{C}})^2 - R^2$$



Geometric Methods

Methods

- **Find normal and tangents**
- **Find surface parameters**



e.g. Sphere

Normal $\vec{\mathbf{N}} = \vec{\mathbf{P}} - \vec{\mathbf{C}}$

Parameters

$$x = \sin\theta\cos\phi \qquad \phi = \tan^{-1}(x, y)$$
$$y = \sin\theta\sin\phi \qquad \theta = \cos^{-1} z$$
$$z = \cos\theta$$

Ray-Implicit Surface Intersection

$$f(x, y, z) = 0$$

$$x = x_0 + x_1 t$$
$$y = y_0 + y_1 t$$
$$z = z_0 + z_1 t$$

$$f^*(t) = 0$$

1. Substitute ray equation

2. Find *positive, real* roots

Univariate root finding

- **Newton's method**
- ***Regula-falsi***
- **Interval methods**
- **Heuristics**

Ray-Algebraic Surface Intersection

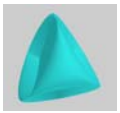
$$p_n(x, y, z) = 0$$

$$x = x_0 + x_1 t$$
$$y = y_0 + y_1 t$$
$$z = z_0 + z_1 t$$

$$p_n^*(t) = 0$$

Degree *n*

- ***Linear*: Plane**
- ***Quadric*: Spheres, ...**
- ***Quartic*: Tori**



Polynomial root finding

- **Quadratic, cubic, quartic**
- **Bezier/Bernoulli basis**
- **Descartes' rule of signs**
- **Sturm sequences**

History

Polygons	**Appel '68**
Quadrics, CSG	**Goldstein & Nagel '71**
Tori	**Roth '82**
Bicubic patches	**Whitted '80, Kajiya '82**
Superquadrics	**Edwards & Barr '83**
Algebraic surfaces	**Hanrahan '82**
Swept surfaces	**Kajiya '83, van Wijk '84**
Fractals	**Kajiya '83**
Height fields	**Coquillart & Gangnet '84, Musgrave '88**
Deformations	**Barr '86**
Subdivision surfs.	**Kobbelt, Daubert, Siedel, '98**

P. Hanrahan, A survey of ray-surface intersection algorithms