Lecture 13

光栅化不能很好的处理全局效果，比如

- soft shadows 软阴影
- 光线在空间中多次弹射

光线

- 光并不沿着直线传播
- 光线之间如果交叉会产生影响
- 光线从光源传播到摄影机，这个过程是 **可逆的**

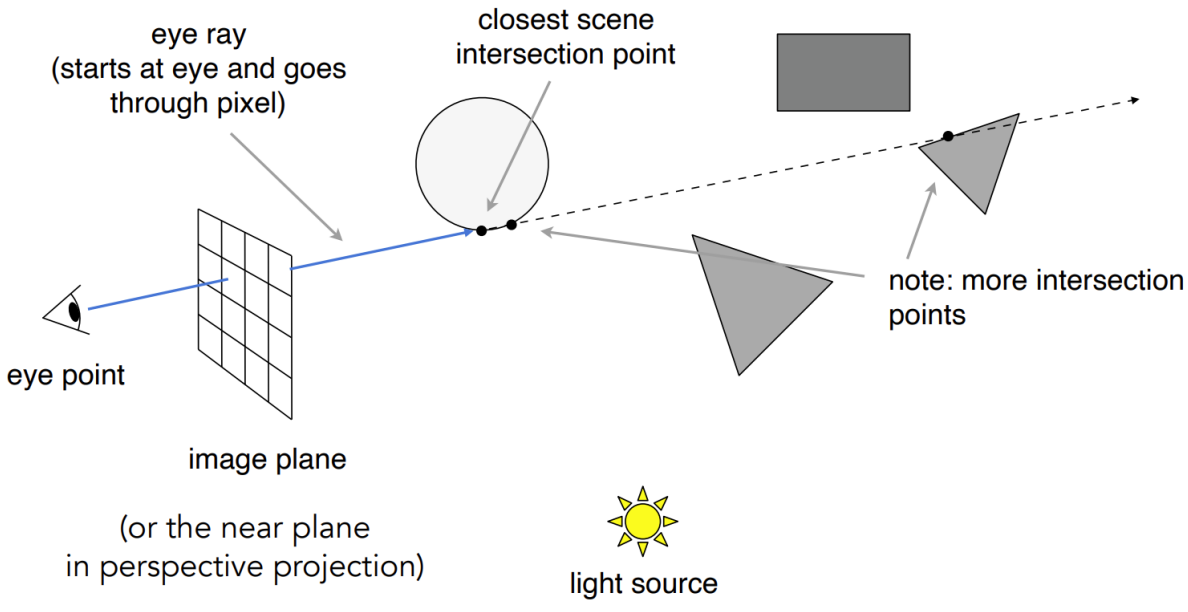
Ray Casting

该方法分为两步

- 在每个像素上投射光线
- 检查每个像素上的阴影状况

generating eye rays Open: Pasted image 20240216163543.png

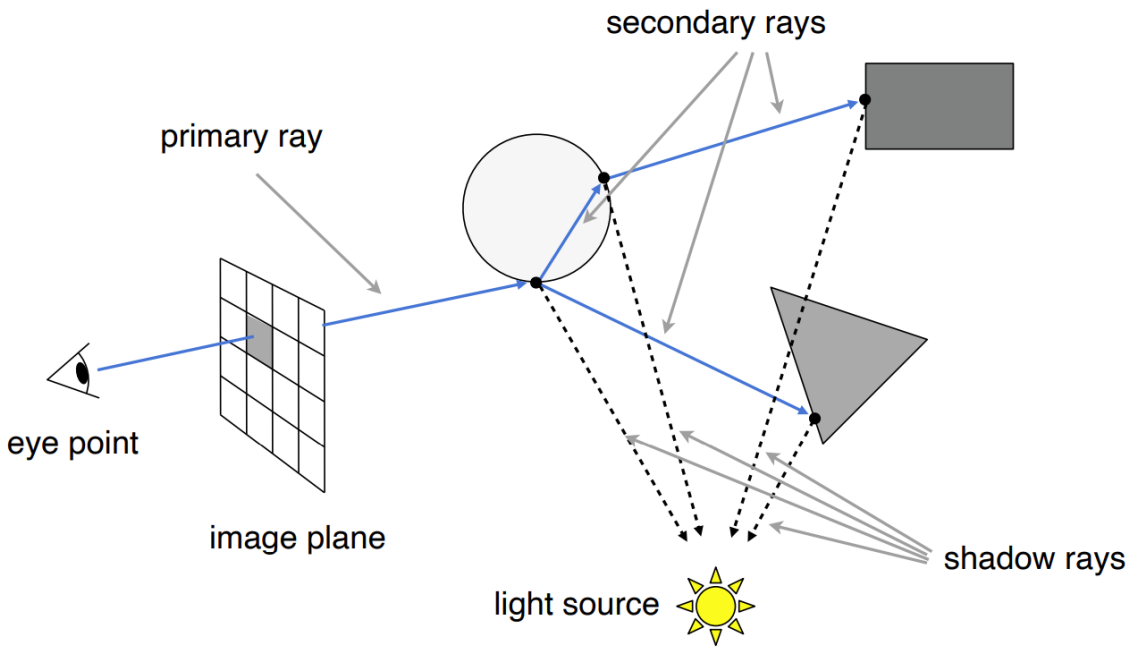
Pinhole Camera Model



- 在摄影机附近做一个平面，并遍历平面上每一个像素
- 获取 **摄影机-该像素** 方向上最近与物体的交点
- 获取该点与光源的连线，做出法线并通过 Lecture 08 > 3.Blinn-Phong Reflection Model 完整公式 计算该点的着色

Recursive(Whitted-Style) Ray Tracing

Open: Pasted image 20240216164013.png



Whitted-Style 中在平面对像素的颜色计算是递归的，会将 **摄影机-该像素** 形成的视线在物体表面点不断 **反射/折射**，直到射线碰到 **diffuse** 类的物体才会停下，伪代码可以写成这样

- 调用反射，折射函数，形成新的射线并调用着色函数
- 如果碰到特定类型物体则返回
- 最后返回所有物体着色结果之和。

在其中阴影的计算可以通过光源与物体的连线可以判断该部分是否处于阴影中

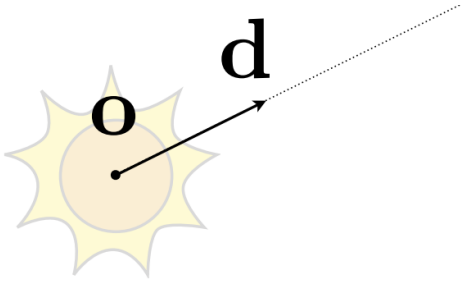
Ray-Surface Intersection

Whitted-Style 解决了 Ray Casting 中只计算一次 **摄影机-物体-光线** 的问题，实现了 **摄影机-(物体-光源)-(物体-光源)-...** 过程，实现了光线的多次反射。 现在需要实现的，是判断射线与物体上某个三角形**是否相交**。

Ray Equation 定义视线向量需要两个参数

- 起点
- 方向
- 此处需要规定 t 的范围，因为视线是射线需要朝正方向

Example:



Ray equation:

$$\mathbf{r}(t) = \mathbf{o} + t\mathbf{d} \qquad 0 \le t < \infty$$

point along ray　"time"　origin　(normalized) direction

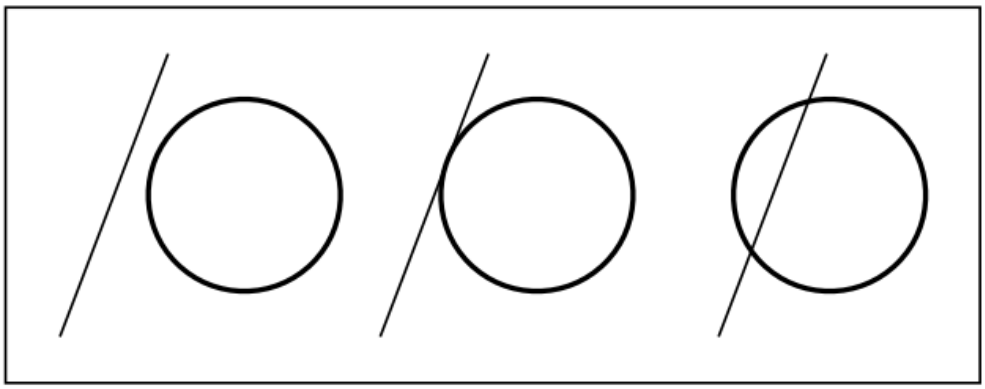
Ray Interseciton With Sphere

$$Ray : r(t) = o + td, 0 \le t < \infty$$
$$Sphere : p : (p - c)^2 - R^2 = 0$$

- 交点即是满足两个方程式的解

$$(o + td - c)^2 - R^2 = 0$$
$$at^2 + bt + c = 0$$
$$a = d \cdot d$$
$$b = (o - c) \cdot (o - c) - R^2$$

- 根据根的个数可以判断直线与圆的交点个数，和与圆的位置关系。



Ray Intersection With Surface 对于隐式表面，将饮食表面的函数 $f(p)$ 与光线表达式联立求解 **实数，非负** 的解即可。

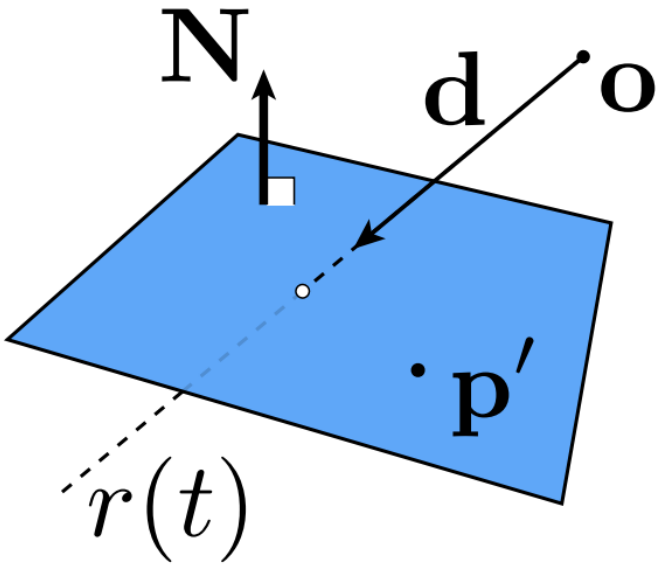
Ray Intersection With Triangle Mesh 对于一般性的算法，过程是判断每一根光线与每个三角形是否相交。此处设定 **存在光线经过平面且平行于平面**，只设定光线与平面的交点为 **0/1** 个。

对于优化算法

- 光线与较大的平面计算交点
- 计算交点是否在三角形内

我们表示一个平面需要 **法线与其上一点**

$$
\begin{aligned}
&r(t) = o + td \\
&p : (p - p') \cdot N = 0 \\
&t = \frac{(p' - o) \cdot N}{d \cdot N} \\
&noteeeee! : \mathbf{0} \leq \mathbf{t} < \boldsymbol{\infty}
\end{aligned}
$$



Möller Trumbore 算法

- 该算法与上述优化算法一致，不过利用重心坐标直接用三角形的三个点表示了与平面的交点，判断系数合理性则可以判断光线与三角形是否相交

$$\vec{\mathbf{O}} + t\vec{\mathbf{D}} = (1 - b_1 - b_2)\vec{\mathbf{P}}_0 + b_1\vec{\mathbf{P}}_1 + b_2\vec{\mathbf{P}}_2$$

Where:

$$\begin{bmatrix} t \\ b_1 \\ b_2 \end{bmatrix} = \frac{1}{\vec{\mathbf{S}}_1 \bullet \vec{\mathbf{E}}_1} \begin{bmatrix} \vec{\mathbf{S}}_2 \bullet \vec{\mathbf{E}}_2 \\ \vec{\mathbf{S}}_1 \bullet \vec{\mathbf{S}} \\ \vec{\mathbf{S}}_2 \bullet \vec{\mathbf{D}} \end{bmatrix}$$

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

Cost = (1 div, 27 mul, 17 add)

公式推导

$$
\begin{aligned}
P &= (1-b_1-b_2)\mathbf{P}_0 + b_1\mathbf{P}_1 + b_2\mathbf{P}_2 \\
O + tD &= \mathbf{P}_0 + b_1(\mathbf{P}_1 - \mathbf{P}_0) + b_2(\mathbf{P}_2 - \mathbf{P}_0) \\
O - \mathbf{P}_0 &= -tD + b_1(\mathbf{P}_1 - \mathbf{P}_0) + b_2(\mathbf{P}_2 - \mathbf{P}_0)
\end{aligned}
$$

我们可以将未知数写成列向量的形式

$$
\begin{bmatrix} -D & (\mathbf{P}_1 - \mathbf{P}_0) & (\mathbf{P}_2 - \mathbf{P}_0) \end{bmatrix}
\begin{bmatrix} t \\ b_1 \\ b_2 \end{bmatrix} = O - \mathbf{P}_0
$$

根据**克莱姆法则**，可以写出三个未知参数的解

$$
\begin{bmatrix} t \\ b_1 \\ b_2 \end{bmatrix} = \frac{1}{\begin{bmatrix} |-D & E_1 & E_2| \end{bmatrix}}
\begin{bmatrix} |T & E_1 & E_2| \\ |-D & T & E_2| \\ |-D & E_1 & T| \end{bmatrix}
$$

$$
\begin{aligned}
T &= O - P_0 \\
E_1 &= P_1 - P_0 \\
E_2 &= P_2 - P_1
\end{aligned}
$$

由于 1x3 **矩阵的值**为 $|ABC| = -(A \times C) \cdot B = -(C \times B) \cdot A$
可以更进一步的化简成上图中的形式，不再叙述。

Ray Tracing Performence Chanllenges

- 由于光线对三角形的遍历由于三角形的数量上升和光线的折射的代价会非常大，**采用物体部分结构代替三角形** 作为光线遍历的单位。

Bounding Volume

- 对于光线投射到物体表面三角形的计算还是过于耗费时间，**使用一个将物体包围的盒，对光线进行大致判断，将光线分为两类，简化了详**

细计算的数量、

- Object is fully contained in the volume
- If it doesn't hit the volume, it doesn't hit the object
- So test BVol first, then test object if it hits

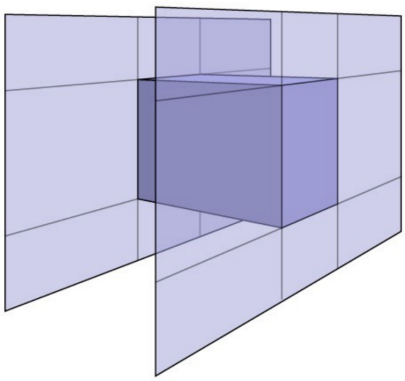
Ray-Intersection With Box

正常的包围盒应当是长方体，这里引入一个概念——AABB 轴对齐包围盒

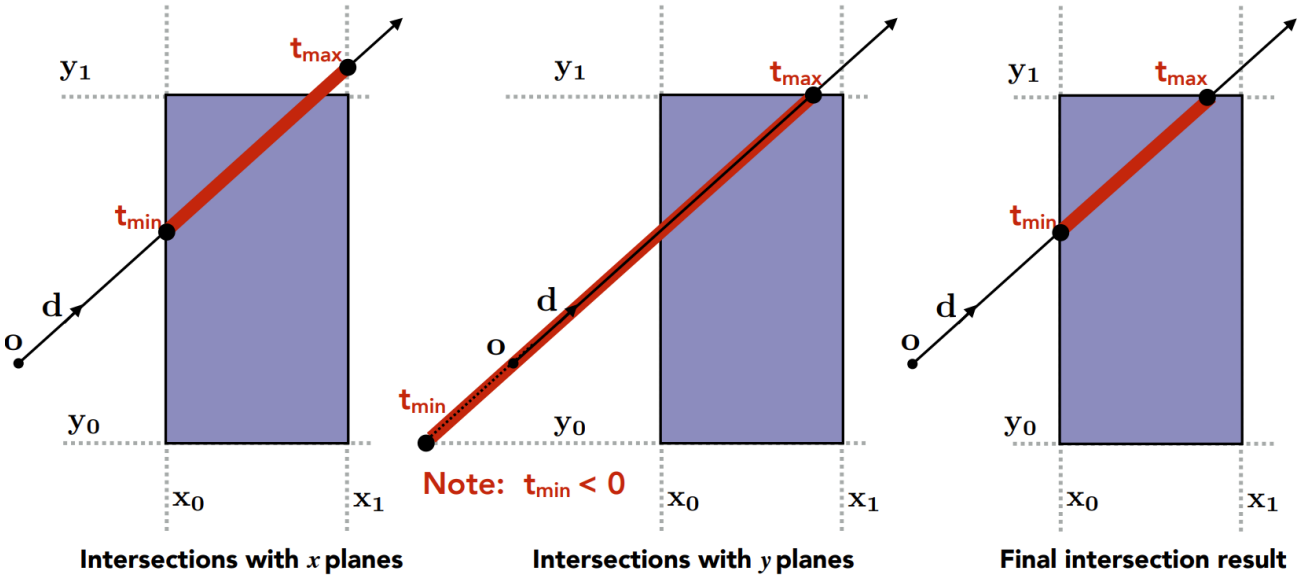
Specifically:

We often use an **Axis-Aligned Bounding Box (AABB)** (轴对齐包围盒)

i.e. any side of the BB is along either x, y, or z axis



- AABB的轴都与 x/y/z 轴其中一条轴平行
- AABB轴上每个面都无限大
- 光线与相互平行的两个面（**称为一个slab**）做交点，固定水平轴为 slab 平行的轴（x/y/z），求其进入包围盒的时间 t_{min} 与离开包围盒的时间 t_{max}



Ray-Intersection With Axis-Aligned Box 对于判断光线是否进入包围盒，以两点做判断

- 对所有 slab，光线都能进入面中，即 $t_{enter} = max\ t_{min}$
- 对其中一个 slab，光线能穿出面中，即 $t_{exit} = min\ t_{max}$
- 如果 $t_{enter} < t_{exit}$ **说明光线穿过了包围盒**

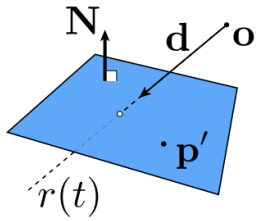
特例

- $t_{enter} < 0$ 说明光线反向延长与包围盒相交，**即光线从包围盒内出发**
- $t_{exit} < 0$ 物体在光线出发点后，不相交
- **note：光线是射线**

所以我们得出光线与轴平行包围盒相交的条件：$t_{enter} < t_{exit}$且$t_{exit} > 0$

轴平行对于计算的优势，计算量少

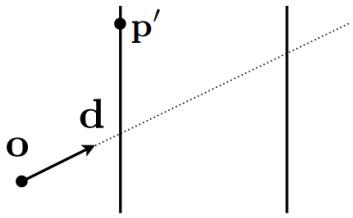
General



$$t = \frac{(\mathbf{p}' - \mathbf{o}) \cdot \mathbf{N}}{\mathbf{d} \cdot \mathbf{N}}$$

3 subtractions, 6 multiplies, 1 division

Slabs perpendicular to x-axis



$$t = \frac{\mathbf{p}'_x - \mathbf{o}_x}{\mathbf{d}_x}$$

1 subtraction, 1 division