



计算机图形学

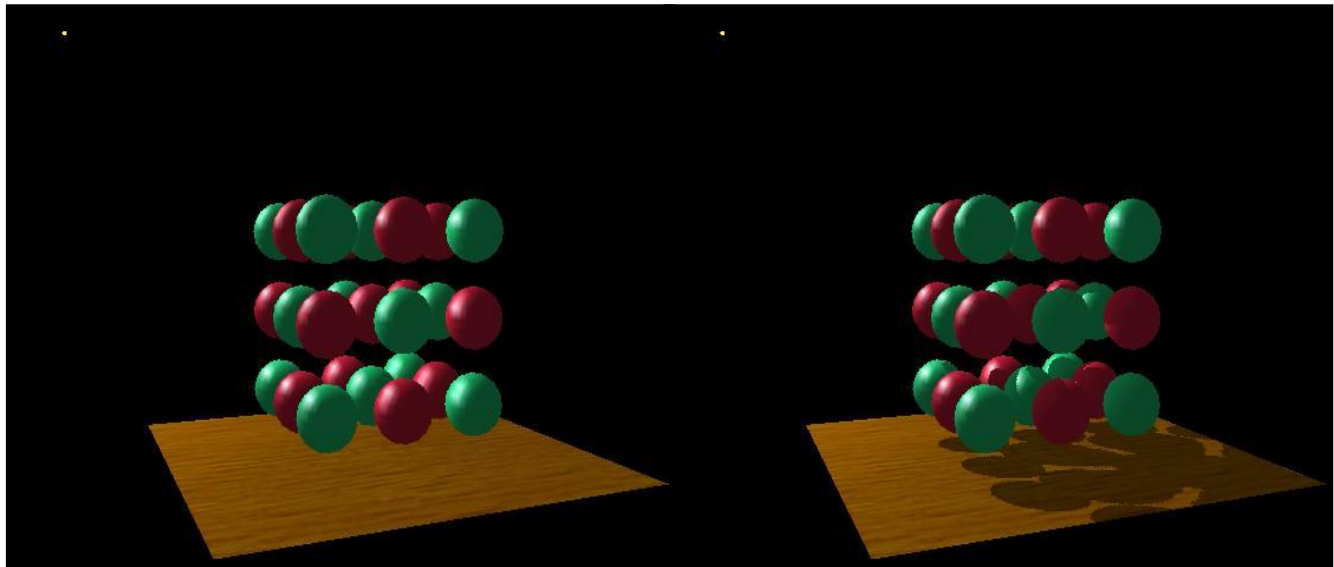
第九章 阴影

颜波

复旦大学计算机科学技术学院
byan@fudan.edu.cn

阴影 (Shadow)

- 阴影
 - 阴影是在构建真实感图像时的一个十分重要的元素，它可以为物体的位置和摆放 (placement) 提供视觉上的提示 (cues)



阴影 (Shadow)

- 阴影

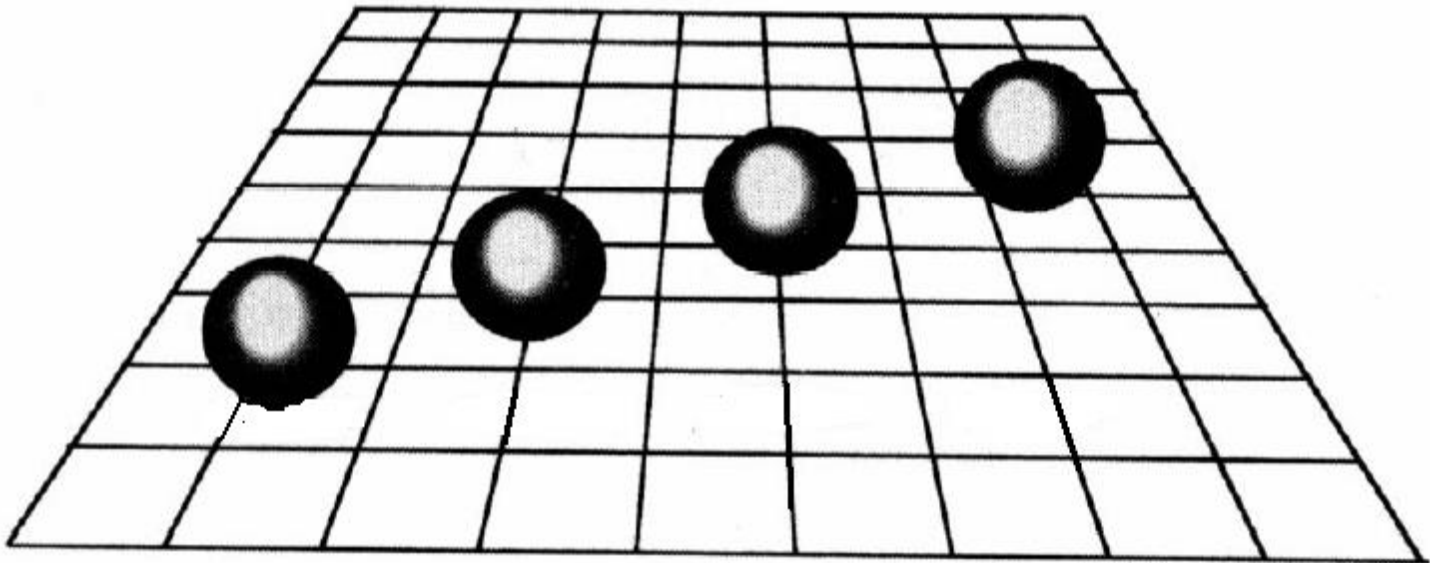
- 阴影是在构建真实感图像时的一个十分重要的元素，它可以为物体的位置和摆放 (placement) 提供视觉上的提示 (cues)
- 阴影的生成是计算机图形学中的一个非常基本的问题
- 今天，我们介绍最为重要的实时阴影绘制算法 (real-time shadow algorithms)

阴影 (Shadow)

- 为什么阴影如此重要?
 - 让我们来看一个例子

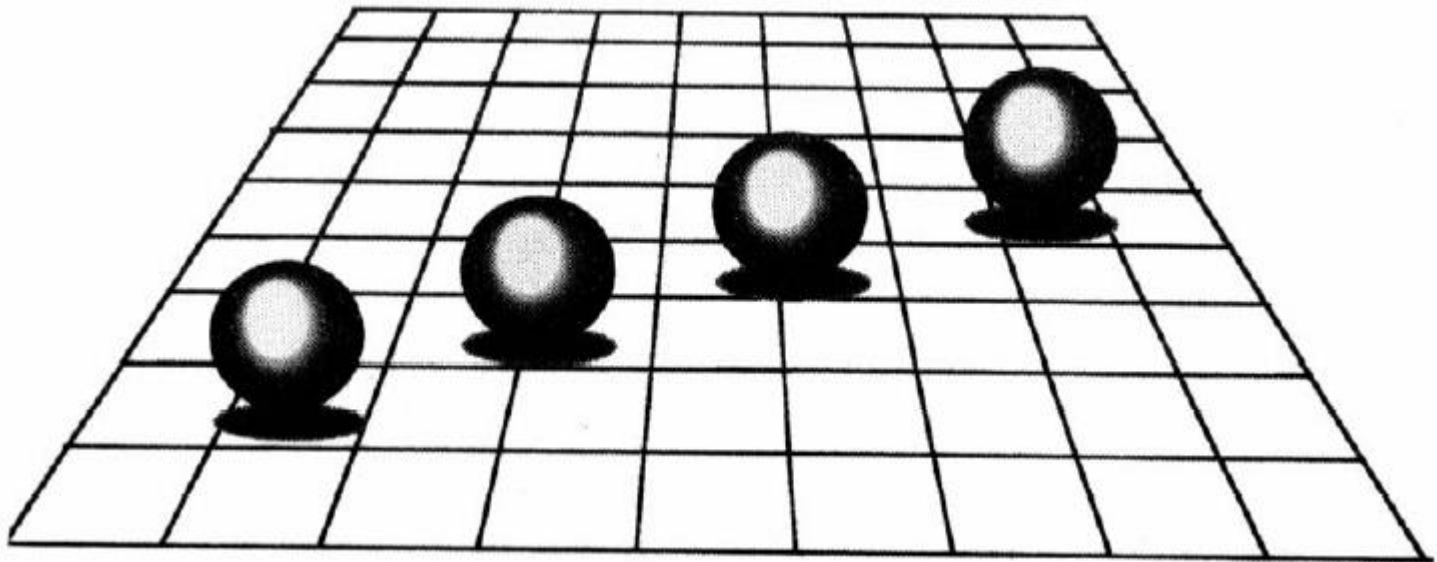
阴影 (Shadow)

- 为什么阴影如此重要?
 - 在没有阴影的提示下, 你能知道下图中的小球的空间位置吗?
 - 很难确定小球的实际位置



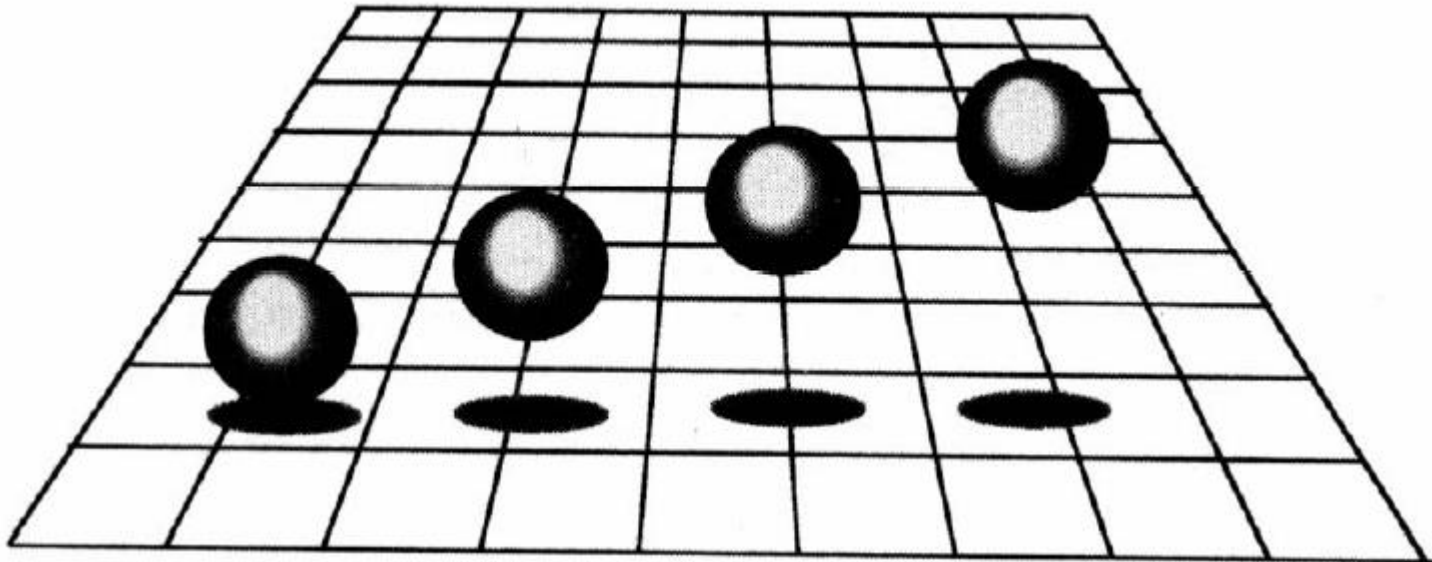
阴影 (Shadow)

- 为什么阴影如此重要?
 - 有了阴影的提示，我们可以清楚地知道小球的位置



阴影 (Shadow)

- 为什么阴影如此重要?
 - 不同的阴影能够对应不同的小球位置



阴影 (Shadow)

- 为什么阴影如此重要?
 - 通过这个例子，我们得出结论：
 - 阴影给物体的位置提供了非常重要的视觉提示 (visual cues)
 - 同样的图片，被赋予不同的阴影，可以代表不同的物体位置
 - 所以，阴影是非常重要的

阴影 (Shadow)

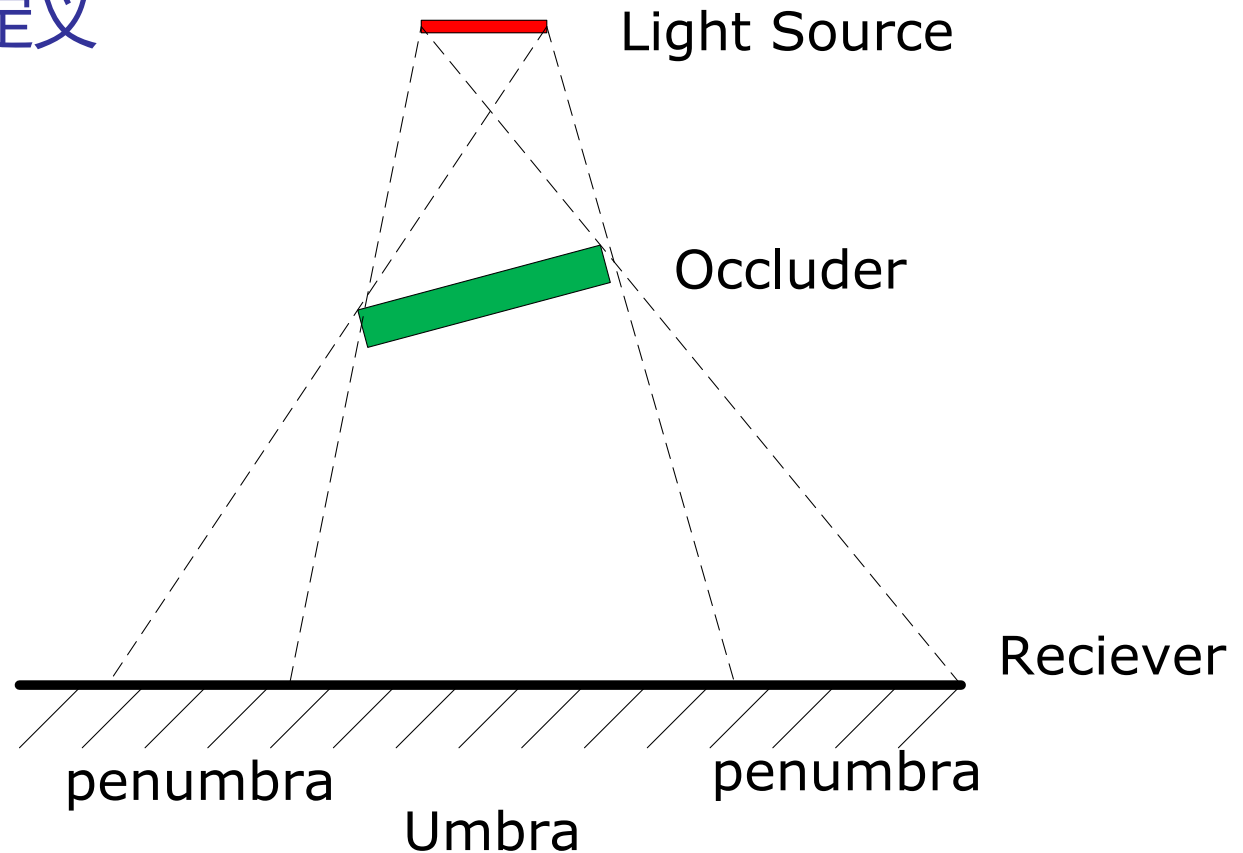
- 阴影是什么?

- 阴影的定义

- 考虑一个由光源 (light source) L 照明的场景:
 - 场景中的每个物体作为接收者 (receivers), 都有可能被光源 L 照明到
 - 如果从场景中的一点 P 点不能看到光源 L 的任何一部分, P 被称为本影 (umbra),
 - 如果从 P 点可以看到光源 L 的某一部分, 却不能看到全部, 那么将 P 称为半影 (penumbra)

阴影 (Shadow)

- 阴影是什么?
 - 阴影的定义

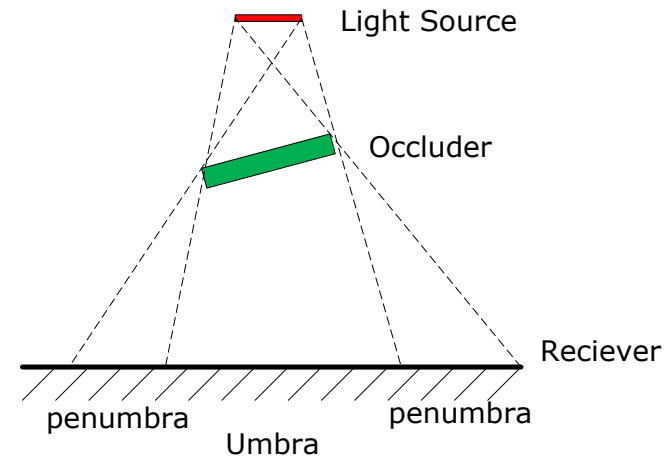


阴影 (Shadow)

- 阴影是什么?

- 阴影的定义

- 本影 (umbra) 和半影 (penumbra) 统称为阴影 (Shadow);
对阴影中的任一点, 光源中至少有一点会被遮挡
 - 由光源到阴影区域的中间物体被称为遮挡物 (occluders)



阴影 (Shadow)

- 阴影的类型 (Types of Shadows):
 - 附着阴影 (attached shadows): 发生于接收者 (receiver) 的法向背离光源方向时
 - 投射阴影 (cast shadows): 发生于接收者 (receiver) 的法向朝着光源方向, 但光源被遮挡物 (occluder) 遮挡时
 - 自阴影 (self shadows): 一类特殊的投射阴影; 对于自阴影, 接收者 (receiver) 和遮挡物 (occluder) 来自于同一物体

阴影 (Shadow)

- 阴影的重要性：
 - 阴影能够帮助我们理解场景中物体的相对位置及大小关系
 - 阴影能够帮助我们理解复杂接收者 (receiver) 的几何形状
 - 阴影能够帮助我们理解复杂遮挡物 (occluder) 的几何形状

阴影 (Shadow)

- 阴影的重要性



阴影能够帮助我们认识接收者 (receiver) 的几何形状

阴影 (Shadow)

- 阴影的重要性



阴影能够帮助我们认识遮挡物 (occluder) 的几何形状

硬阴影和软阴影

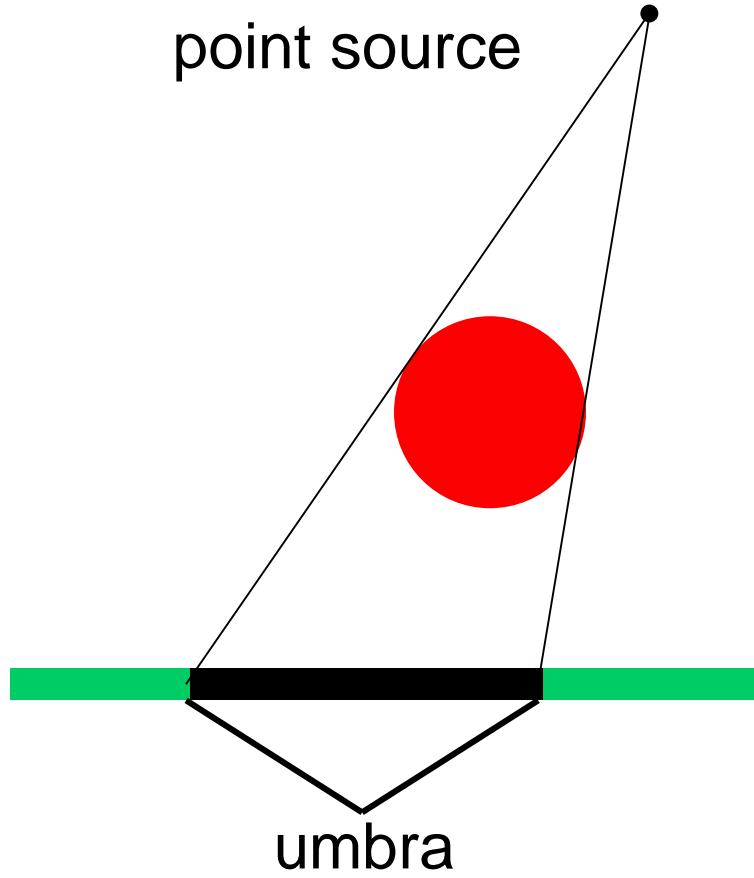
- 硬阴影 (Hard Shadow) 和 软阴影 (Soft Shadow)
 - 通常会将“阴影”理解为一个二值的状态：每个点要么在阴影中，要么不在；这样的理解对应的其实是硬阴影 (Hard shadows)，硬阴影由点光源产生
 - 然而，点光源在现实世界中并不存在，并且硬阴影会给图像带来一种相当不真实的感觉
 - 即使是我们日常生活中最常见的光源：太阳，产生的也并不是硬阴影

硬阴影和软阴影

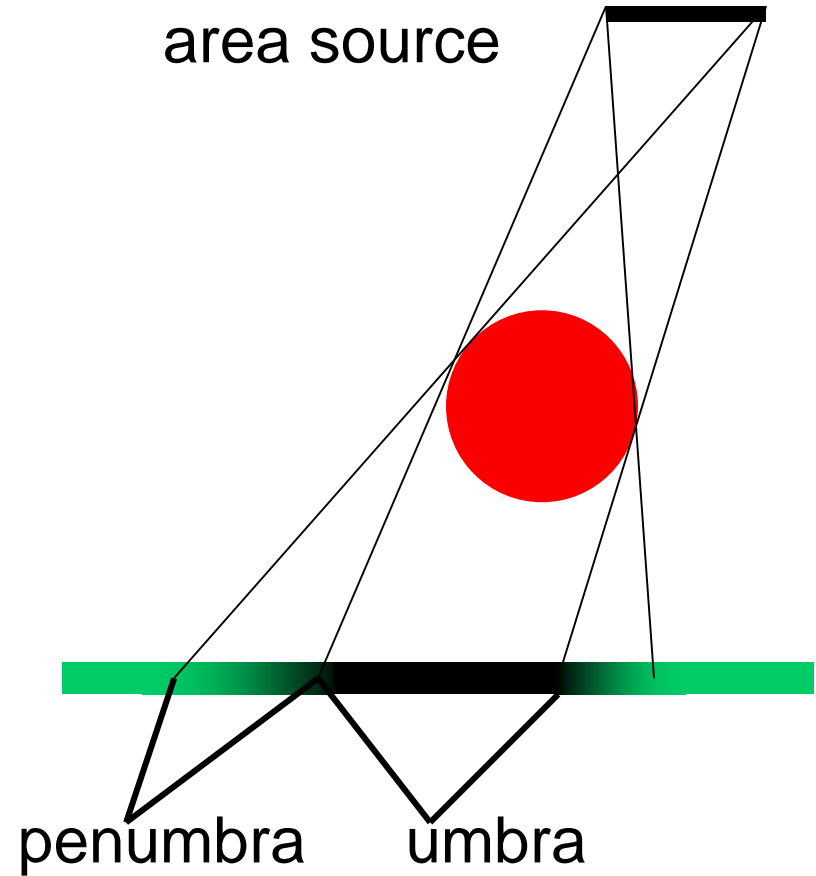
- 硬阴影 (Hard Shadow) 和 软阴影 (Soft Shadow)
 - 尽管如此, 由于点光源在图形学中容易被模拟; 对硬阴影的计算存在不少实时算法 (real-time algorithms)
 - 然而, 在有限展度 (finite extent) 的光源 (通常是面光源) 下, 确定本影 (umbra) 和半影 (penumbra) 区域通常比较困难, 因为这意味着需要求解 3D空间中的可见关系 (visibility relationships), 这是个非常困难的问题

硬阴影和软阴影

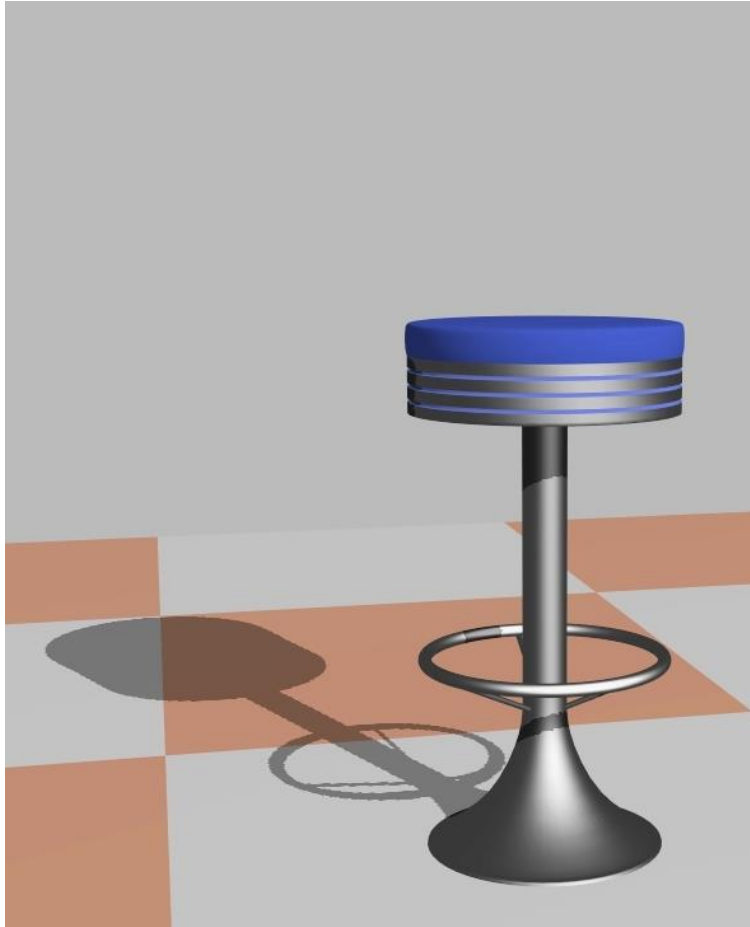
point source



area source



硬阴影和软阴影



平面 (Planar) 阴影

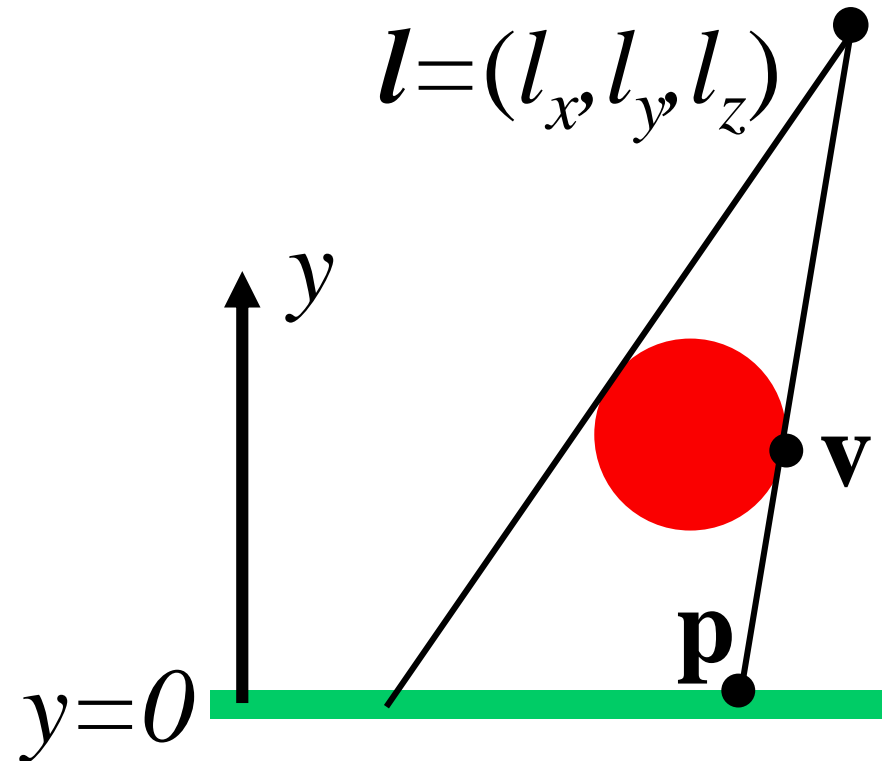
- 什么是平面阴影?
 - 当物体的阴影被投射到平的表面时，会产生平面阴影，平面阴影是一种最简单的阴影
 - 下面我们将介绍计算平面阴影的算法

平面 (Planar) 阴影

- 阴影投影 (Projection Shadows)
 - 在阴影投影方法中，三维空间中的物体需要被绘制两次以实现阴影效果
 - 具体来说，是使用一个矩阵将遮挡物表面的点投影到需要计算阴影的平面上

平面 (Planar) 阴影

- 阴影投影 (Projection Shadows)
 - 考虑下图的例子, 光源位于 l , 需要计算阴影的点位于 v , v 被投影到 p :
 - 进一步假设需要计算阴影的平面是 $y=0$ (推广到任意其他平面也并不困难)

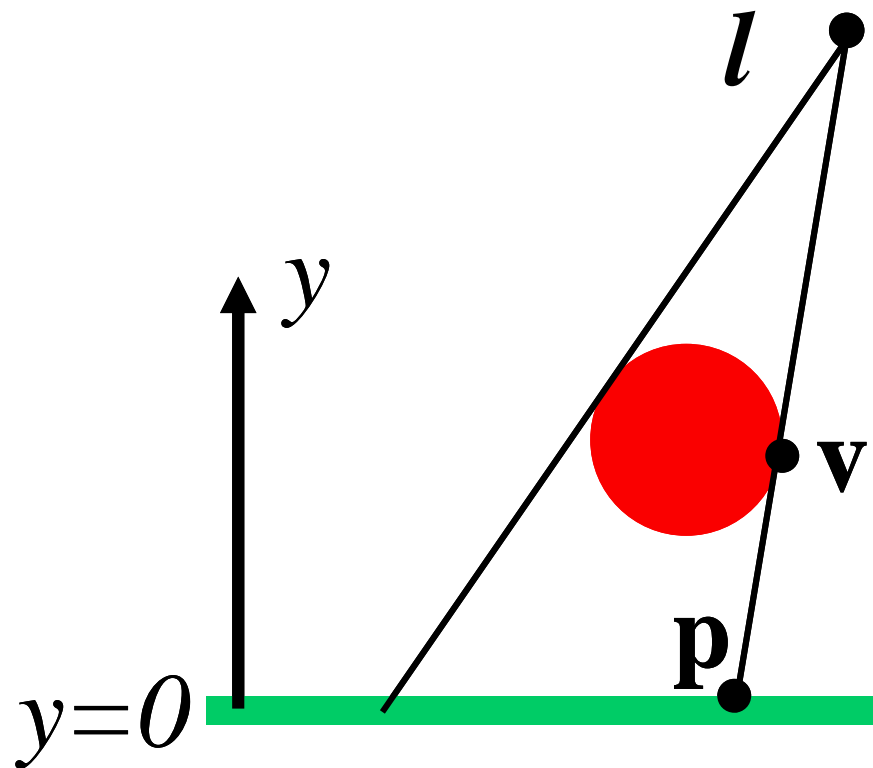


平面 (Planar) 阴影



- 阴影投影 (Projection Shadows)
 - 由相似关系，可以按照以下方法计算 p 点的 x 坐标：

$$\frac{p_x - l_x}{v_x - l_x} = \frac{l_y}{l_y - v_y}$$
$$\Rightarrow p_x = \frac{l_y v_x - l_x v_y}{l_y - v_y}$$



平面 (Planar) 阴影

- 阴影投影 (Projection Shadows)

- z 坐标可以类似计算得到:

$$p_z = \frac{l_y v_z - l_z v_y}{l_y - v_y}$$

- 这两个方程可以统一地使用一个投影矩阵 M 来表示

平面 (Planar) 阴影

- 阴影投影 (Projection Shadows)
 - 投影矩阵:

$$\mathbf{M} = \begin{pmatrix} l_y & -l_x & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & -l_z & l_y & 0 \\ 0 & -1 & 0 & l_y \end{pmatrix}$$

- 容易验证: $\mathbf{M} \mathbf{v} = \mathbf{p}$

平面 (Planar) 阴影

- 阴影投影 (Projection Shadows)
 - 通常情况下需要计算阴影的平面并不会是 $y=0$, 而是一般的平面: $n \cdot P + d = 0$
 - 类似于 $y=0$ 的情况, 可以算出 p 需要满足:

$$p = l - \frac{d + n \cdot l}{n \cdot (v - l)} (v - l)$$

平面 (Planar) 阴影

- 阴影投影 (Projection Shadows)

- 这个方程同样可以表示成为投影矩阵的形式, 使得:

$$Mv = p$$

$$M = \begin{pmatrix} \mathbf{n} \cdot \mathbf{l} + d - l_x n_x & -l_x n_y & -l_x n_z & -l_x d \\ -l_y n_x & \mathbf{n} \cdot \mathbf{l} + d - l_y n_y & -l_y n_z & -l_y d \\ -l_z n_x & -l_z n_y & \mathbf{n} \cdot \mathbf{l} + d - l_z n_z & -l_z d \\ -n_x & -n_y & -n_z & \mathbf{n} \cdot \mathbf{l} \end{pmatrix}$$

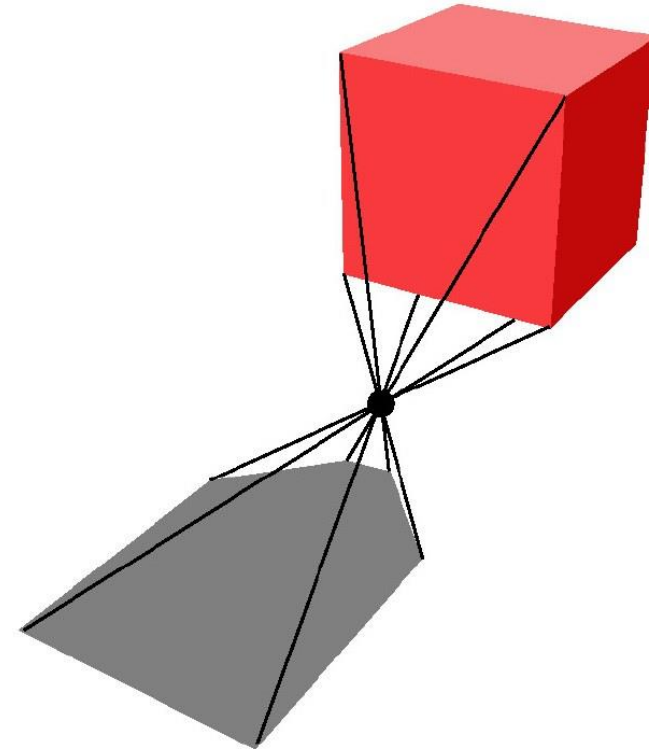
- 当平面恰好为 $y=0$ (即 $\mathbf{n}=(0,1,0)$, $d = 0$) 时, 矩阵 M 与之前的推导相吻合

平面 (Planar) 阴影

- 阴影投影 (Projection Shadows)
 - 为了绘制阴影，可以简单地利用以上矩阵将三维场景中的物体投影到目标平面上，然后对投影物体使用暗色 (dark color) 并去除光照 (without illumination) 进行绘制
 - 阴影投影 (projection shadow) 算法的局限性：
 - 接收者 (receiver) 必须是平面
 - 每一帧的阴影需要重新绘制，即使这些阴影没有变化

平面 (Planar) 阴影

- 两种会产生错误阴影的情况 (Failure cases):
 - *antishadow* (假阴影)
当光源位于最上方物体之下时, 算法会生成错误的阴影 (如右图所示):
 - *False shadow*
当物体位于接收平面背对光源的一方 (即下方) 时, 也会导致产生错误的阴影



曲面上的阴影

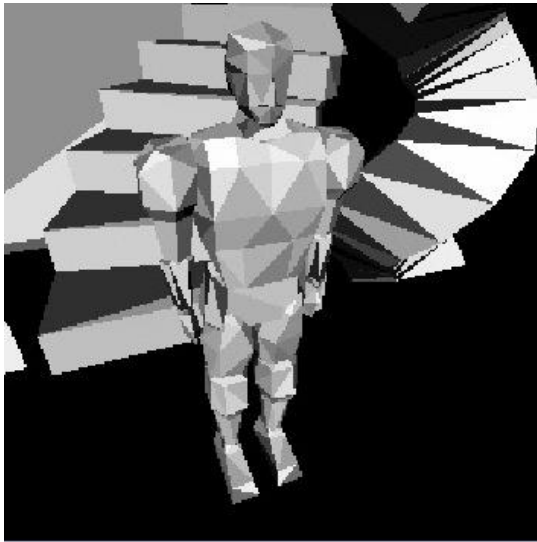
- 要将前面平面阴影的计算方法扩展到曲面 (curved surfaces) 上, 一个自然的想法是: 使用阴影图像 (shadow image) 作为投影的纹理 (projective texture)
- 试想, 将光源作为视点: 从光源看得到的区域, 就被绘制; 从光源看不到的区域, 就是阴影

曲面上的阴影

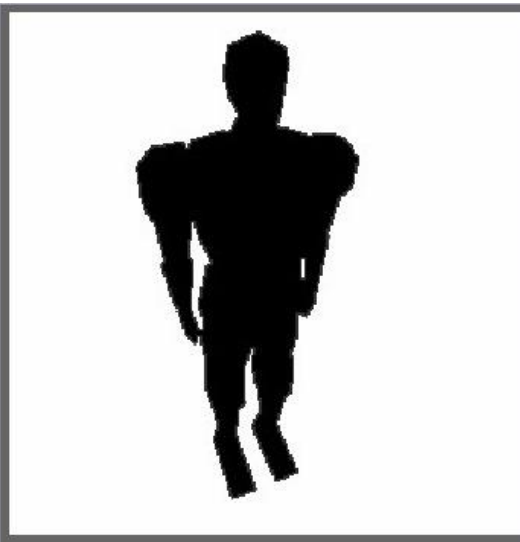
- 从光源出发进行绘制，将遮挡物绘制成黑白场景，在白色的背景上，所有的遮挡物都被绘制成黑色，这样可以获得一张阴影的纹理图
- 这张纹理被投影到需要绘制阴影的曲面之上，于是在曲面上的每一点都可以计算一个其纹理图上的UV坐标，而利用此坐标，则可以直接判定该点是否属于阴影区域
- 这个方法被称为阴影纹理（“shadow texture”）技术

曲面上的阴影

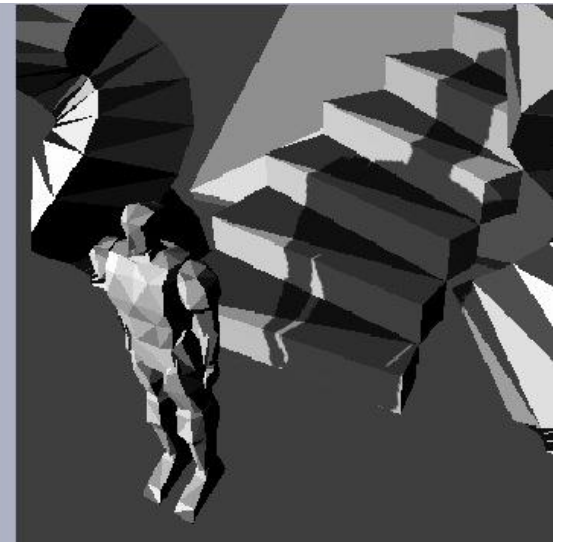
- 不足 (disadvantages):
 - 物体不能投射阴影到其自身



from light



shadow texture



shadows on stairs

阴影的绘制算法

- 下面我们介绍两个重要的阴影绘制算法：
 - 阴影体 (shadow volume) 算法
 - 阴影图 (shadow map) 算法
- 它们都可以用于绘制任意场景、任意几何的阴影效果 (包括自阴影)

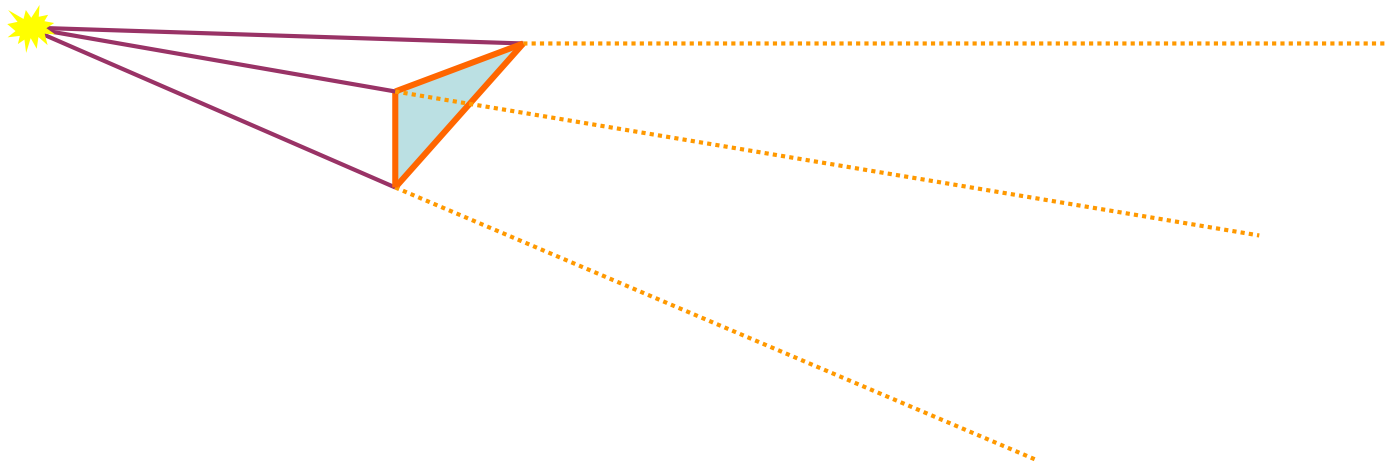
阴影体 (Shadow Volume)

- 阴影体 (shadow volume) 算法由 Crow 提出, 可以将阴影投射到任何物体表面
- 阴影体技术有时也被称为体阴影 (volumetric shadows)



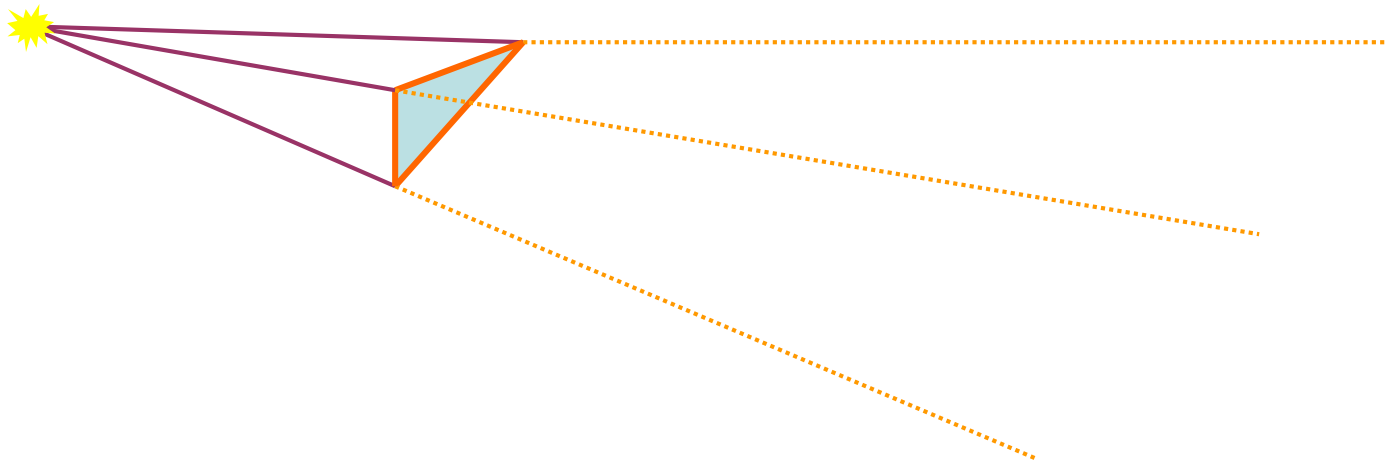
阴影体(Shadow Volume)

- 首先，想象三维空间中的一个点和一个三角形，连接点到三角形的三个顶点并延长，可以得到一个衍生到无穷远处的三棱锥



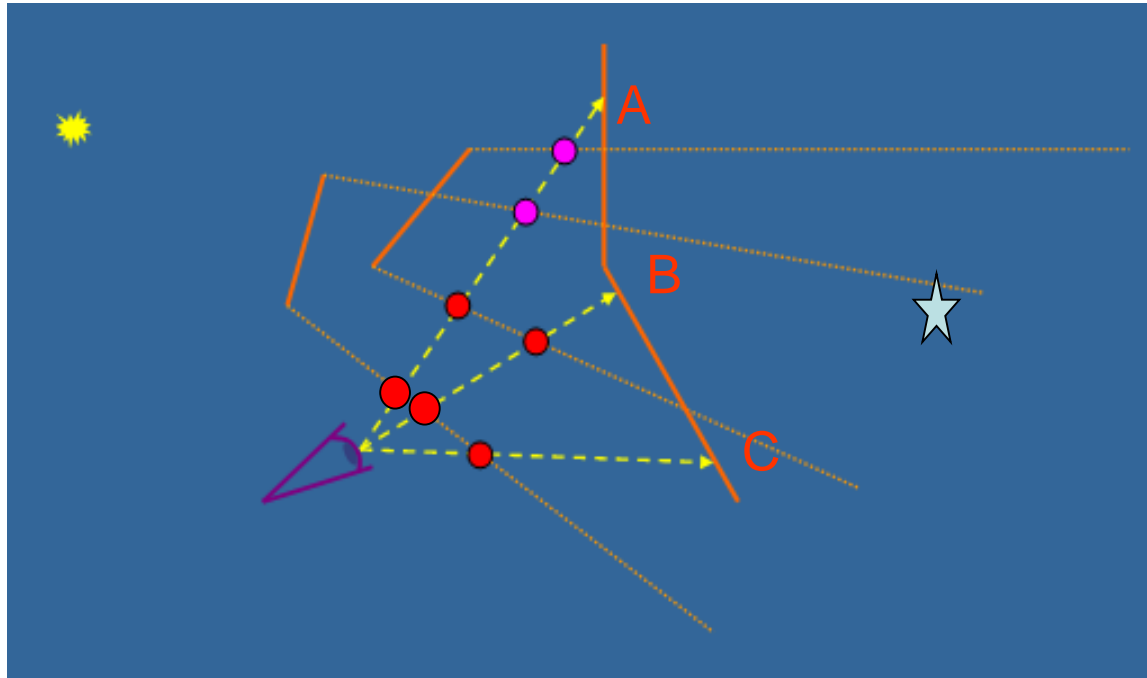
阴影体(Shadow Volume)

- 将点想象成点光源，那么任何物体的任何一部分，只要在下图中切去顶端的三棱锥中，就属于阴影区域
- 这个被切去顶端的三棱锥我们称之为阴影体 (shadow volume)



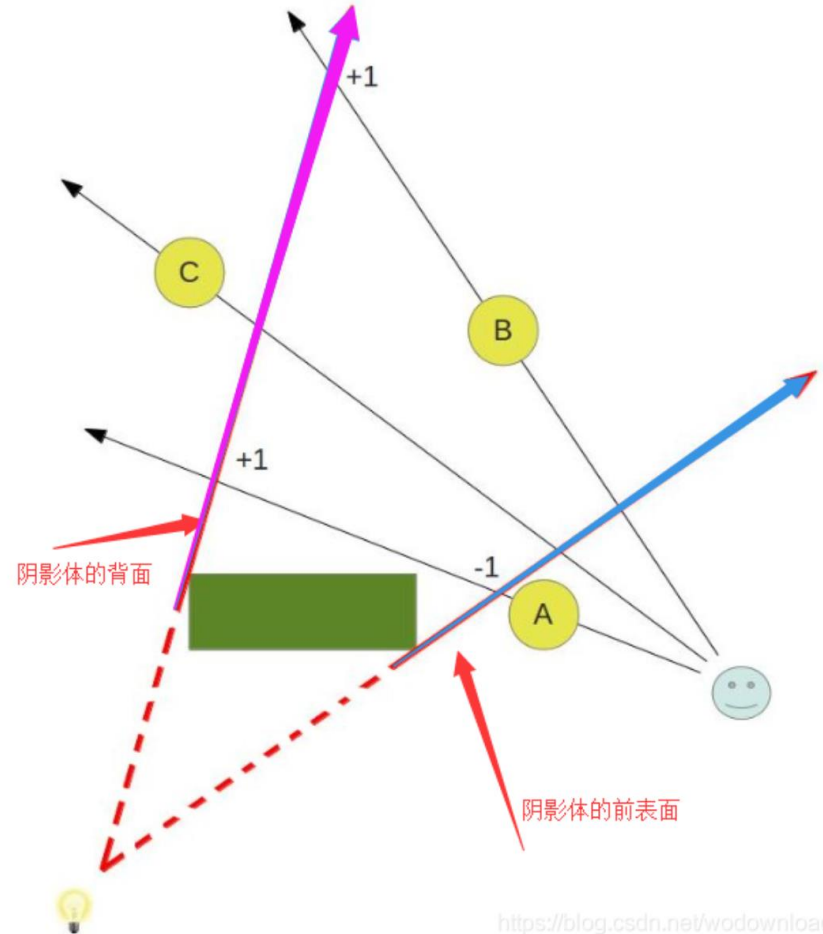
阴影体(Shadow Volume)

- 假设在绘制中，我们从视点向屏幕的某个像素投射一条射线，该射线与场景中某一物体交于一点，我们需要确定该交点是否位于阴影当中
- 我们需要做的仅仅是判断该交点是否位于某个阴影体 (shadow volume) 当中



阴影体(Shadow Volume)

- 假设视点在所有的 shadow volume 之外，我们维护一个计数器，其初始值是零
- 每次当从视点射向目标像素的射线进入到一个 shadow volume 中时，将计数器 +1；而当射线从一个 shadow volume 中射出时，将计数器 -1
- 这样，我们只需检验当射线到达交点时，计数器是否大于零：如果大于零，则交点位于阴影中；否则不属于阴影区域



阴影体(Shadow Volume)

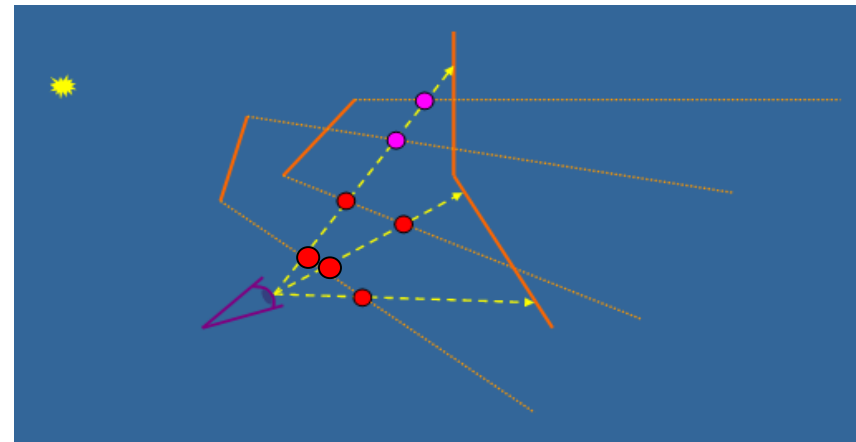
- 使用模板缓存 (Stencil Buffer)
 - 当然，如果从几何上实现上述算法是麻烦而费时的，一个巧妙的办法是使用硬件中的模板缓存 (stencil buffer) 来实现计数
 - 模板缓存 (stencil buffer) 可以对每个像素存储一个整数值；它与深度缓存 z-buffer 的区别仅仅在于z-buffer 存储的是实数表示的深度值 (depth value)

阴影体(Shadow Volume)

- 使用模板缓存 (Stencil Buffer) 阴影绘制
 - 首先, 清空模板缓存
 - 然后, 将整个场景绘制到帧缓存中, 这次绘制只使用环境光分量和发光分量, 并获取相应的颜色信息 (在 color buffer 中) 及深度信息 (在 z-buffer 中)

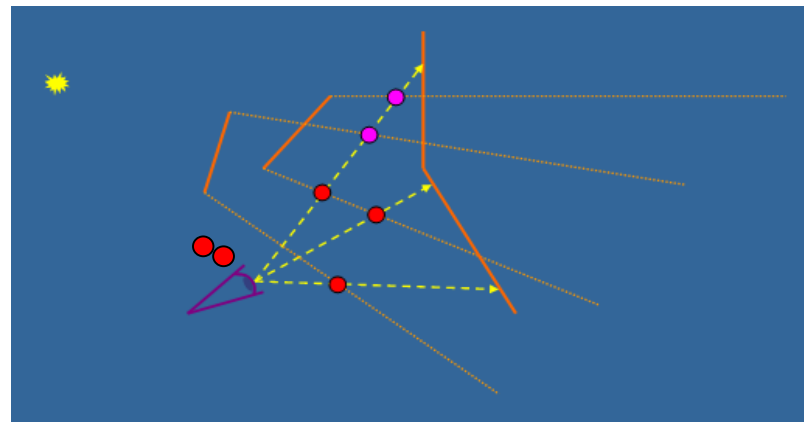
阴影体(Shadow Volume)

- 第三步，关闭颜色缓存的写入和深度检测，绘制所有 shadow volume 的正面 (即射线射入 shadow volume 时相交的面)
 - 在这个过程中，如果一个像素其深度值小于之前算好的 z-buffer 中的深度值，那么将这个像素的模板缓存 (stencil buffer) 上的计数器 +1



阴影体(Shadow Volume)

- 第四步，类似于前一步骤，将所有 shadow volume 的反面绘制一遍，只是这次是每发现一个像素其深度值小于之前算好的 z-buffer 中的深度值，将该像素的模板缓存上的计数器 -1
- 最后，再将整个场景根据模板缓存的信息绘制上漫反射分量和高光分量：只有模板缓存是 0 的像素非阴影区域才绘制，以实现阴影效果



阴影体(Shadow Volume)

- Shadow Volume 算法的优点：
 - 首先，它可以使用通用的图形学硬件实现，而仅仅需要一个模板缓存 (stencil buffer)
 - 其次，由于它不是基于图像的方法（不像下面将要介绍的 shadow map 算法），shadow volume 算法并不会产生由采样和分辨率带来的各种问题，从而可以在任何地方生成正确和清晰的阴影 (sharp shadows)

阴影体(Shadow Volume)

- Shadow Volume 算法的不足:

- 主要体现在性能方面:

由于shadow volume的多边形面片数通常都较多,且会覆盖住大量的像素,这在很大程度上影响了算法运行和光栅化过程的速度,使得绘制的效率较低

阴影体(Shadow Volume)

- 一些结果：



阴影体(Shadow Volume)

- 一些结果:



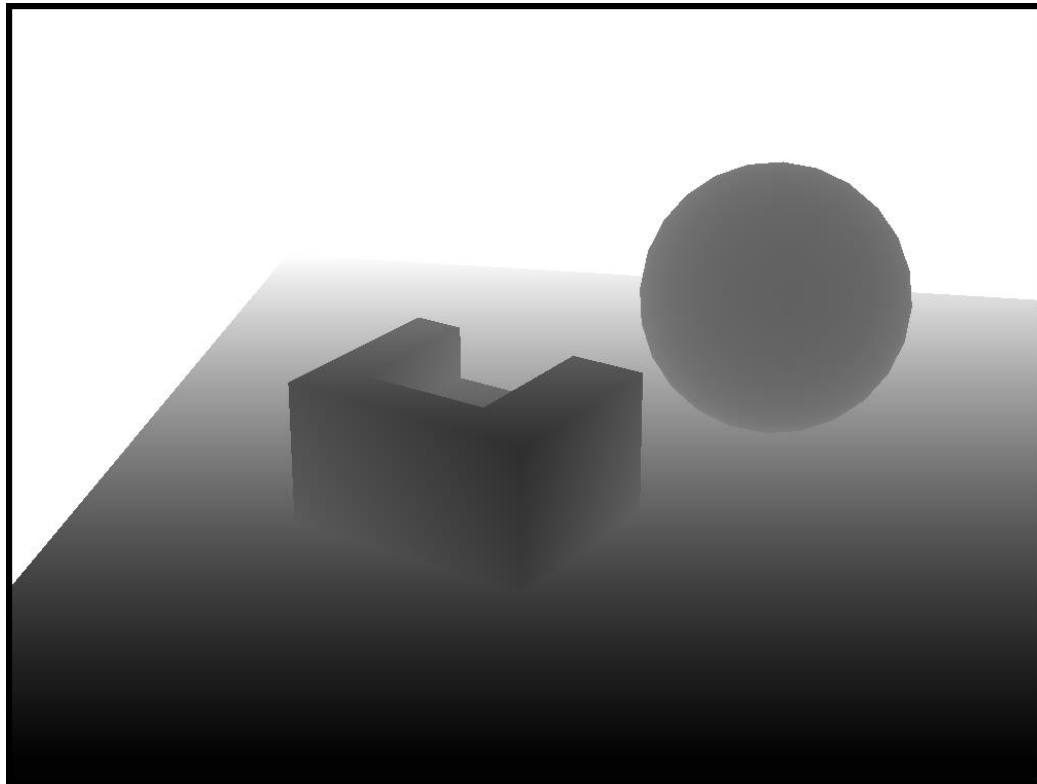
阴影图 (Shadow Map)

- Lance J. Williams 提出了一个基于 z-buffer 的算法，可以对于任意的场景物体快速地计算阴影
- 这个算法的思想是：以光源作为视点的位置，使用 z-buffer 算法绘制场景，获得阴影图 (shadow map)，并将其结果用于任意场景的阴影绘制



阴影图 (Shadow Map)

- 使用深度缓存 z-buffer, 可以获得从光源出发到任意一个方向最近点的距离, 并以图像的形式存储下来, 如图:
- 我们将这张深度图像称为阴影图 (shadow map)



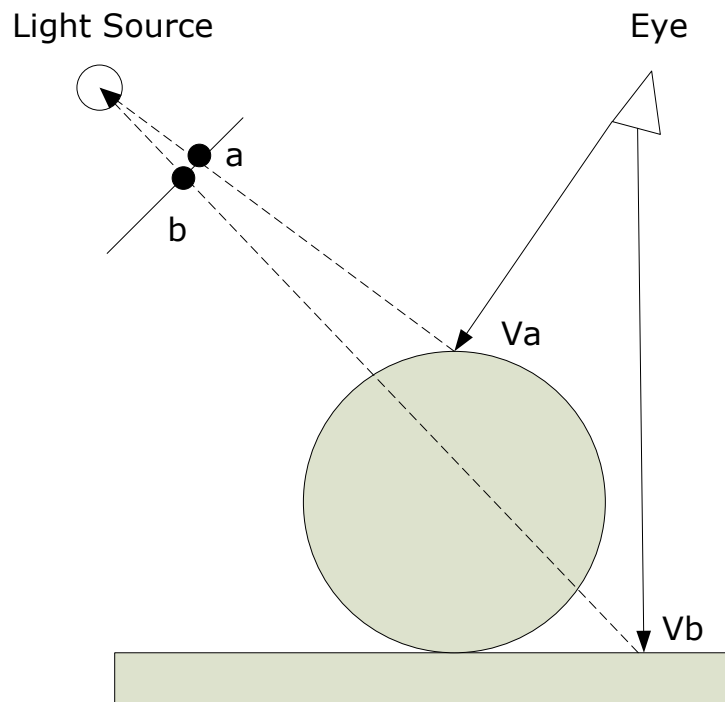
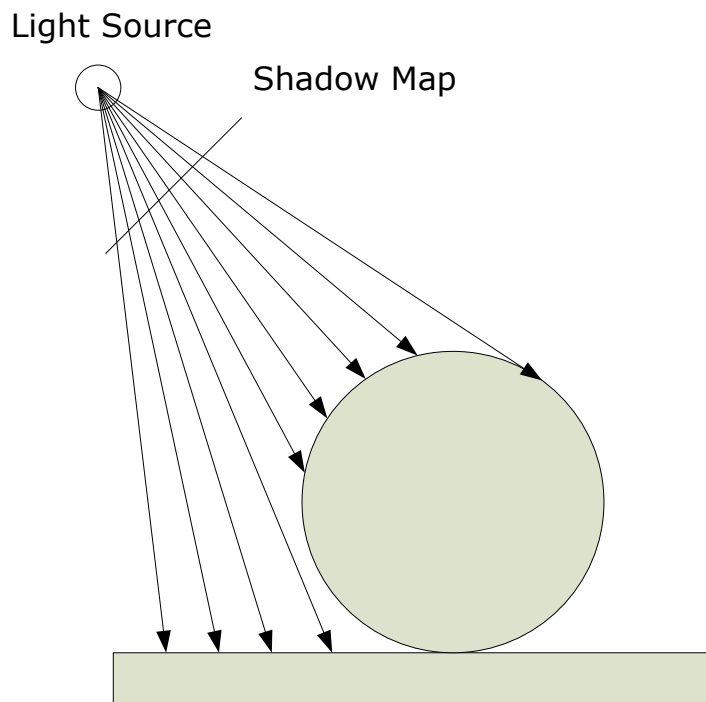
阴影图 (Shadow Map)

- 使用阴影图 (shadow map), 可以在任意视点位置对场景进行带阴影效果的第二次绘制
- 在进行绘制时, 对于沿视点向屏幕每个像素发出的光线与场景中物体的交点, 使用阴影图判断该点是否位于阴影区域:
 - 如果该交点到光源的距离大于阴影图上对应点所存储的深度值, 则该点位于阴影中
 - 否则, 该点不在阴影中

阴影图 (Shadow Map)

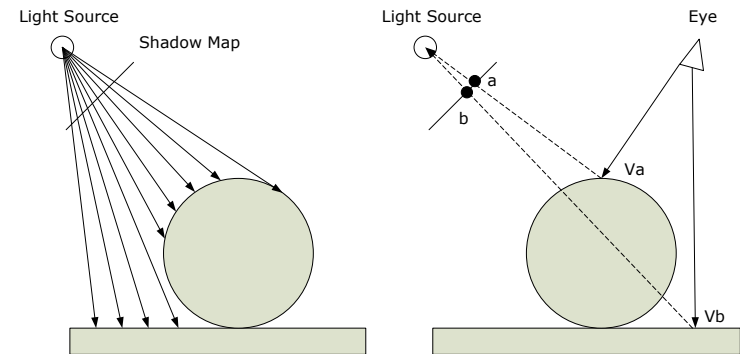


示意图:



阴影图

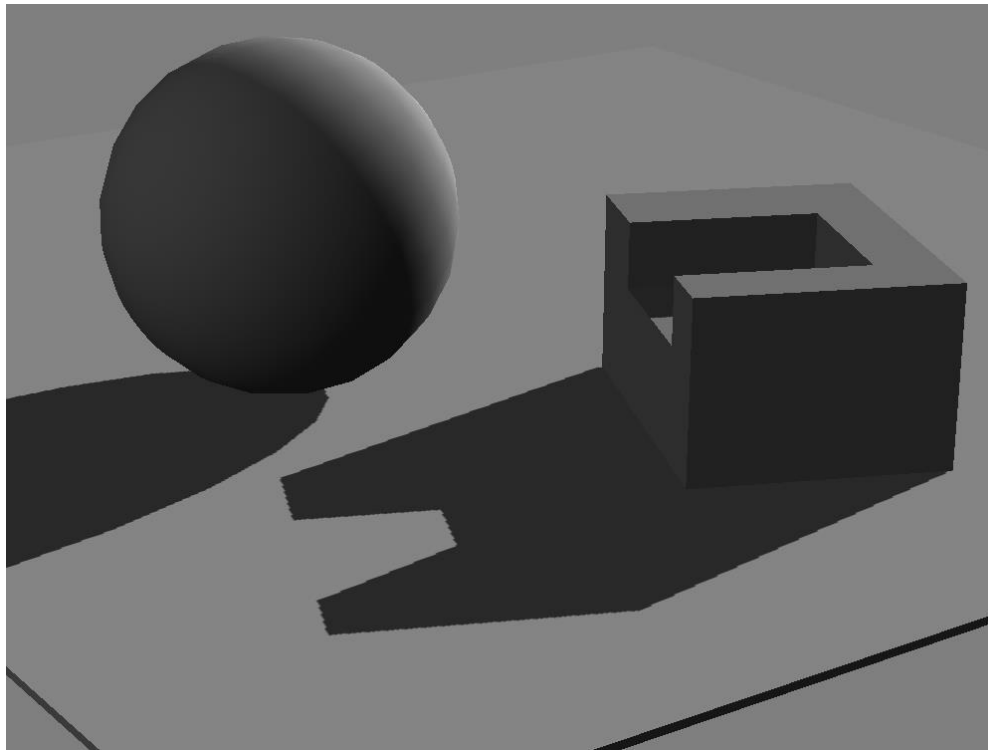
- 在示意图中，阴影图存储了由光源到场景中物体表面的深度信息；右图中，由视点对 Va 和 Vb 这两个位置进行观察：



- 对于 Va : Va 在阴影图上的纹理坐标为 a ，而存储在 a 的阴影图的深度信息不小于 Va 到光源的距离，因此 Va 不属于阴影区域
- 对于 Vb : 由 Vb 到光源的距离要比存储在 b 的阴影图的深度信息更大，因此 Vb 位于阴影区域

阴影图 (Shadow Map)

- 结果示例:



阴影图 (Shadow Map)

- 阴影图算法的优点：
 - 绝大部分的硬件都可以直接支持阴影图 (shadow map) 算法，以用于绘制任意场景的阴影效果
 - 阴影图算法很快，阴影图的构建开销与要绘制的基元 (rendered primitives) 数目成正比；而使用阴影图进行任意视点的阴影绘制时，每个像素只需要额外对阴影图进行一次查询，以及一次距离的比较，这可以在常数时间内完成

阴影图 (Shadow Map)



- 阴影图算法的不足：
 - 由于阴影图算法是基于图像的算法，因此其绘制质量会受到阴影图的分辨率、以及 z-buffer 的数值精度的影响

阴影

- 总结
 - 阴影图 (Shadow map) 和 阴影体 (shadow volume) 算法是最为广泛使用的阴影生成与绘制算法 (其中阴影图算法使用得更为广泛)
 - 我们介绍的只是这两个算法的最基本的原型，之后的研究者提出了它们的许多变种

阴影

- 使用 ATI 显卡绘制的一张效果：



阴影艺术 (Shadow Art)

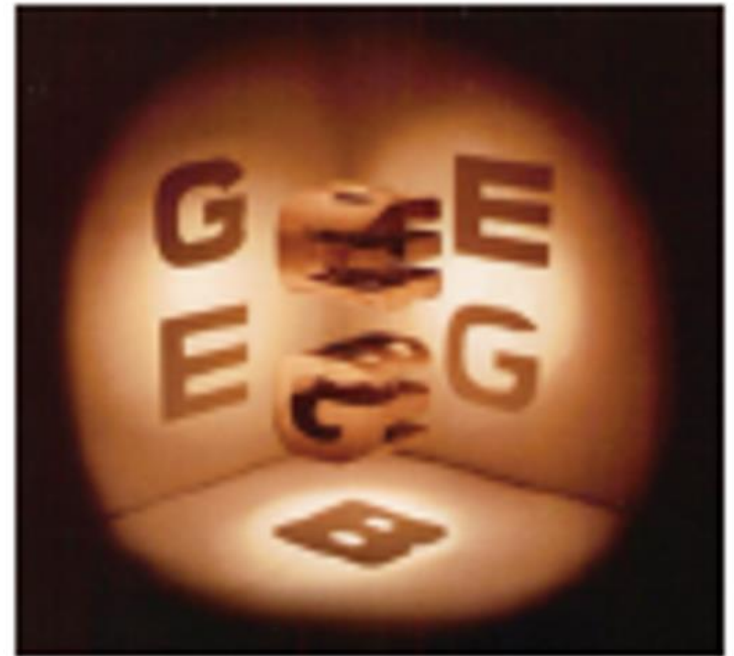


- Shadow Art, Niloy Mitra, Mark Pauly, ACM SIGGRAPH-ASIA 2009.



阴影艺术 (Shadow Art)

- 近代艺术家们使用各种材料, 通过组装和雕刻, 作出能产生有意义的阴影效果的艺术作品
- 一件雕塑作品能够在不同的光源照射下产生不同形状的、且具有独特意义的阴影效果
- 这项工作关注如何使用计算机自动或交互地设计出能在不同光源下产生多样化阴影效果的实体造型



阴影艺术 (Shadow Art)



- 一些实物效果:

