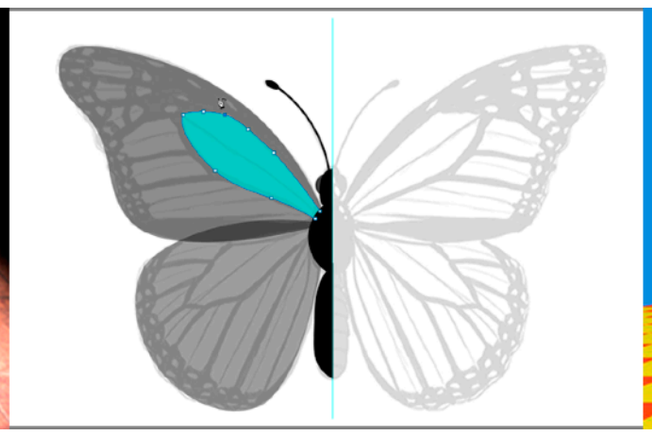
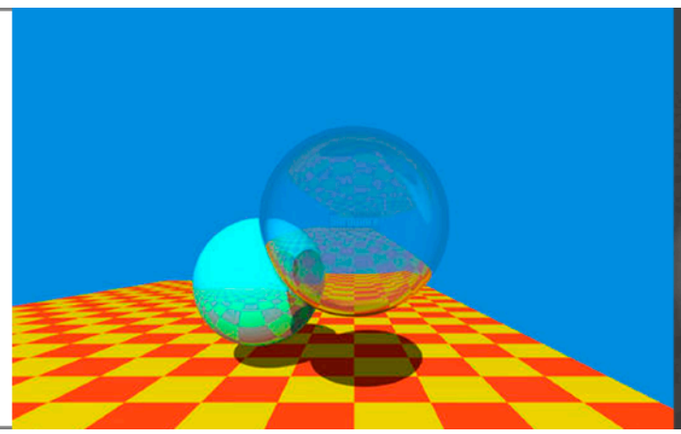
Introduction to Computer Graphics

GAMES101, Lingqi Yan, UC Santa Barbara

Lecture 7:
Shading 1 (Illumination, Shading and Graphics Pipeline)


http://www.cs.ucsb.edu/~lingqi/teaching/games101.html

Announcements

- Homework 1
  - 300+ submissions
  - Will start TA recruiting (from existing applications) soon
- Homework 2 will be out today
  - About Z-buffering
  - Much easier than HW1
- May need an additional lecture for shading

Last Lectures

- Rasterization
  - Rasterizing **one triangle**
  - Sampling theory
  - Antialiasing

Today

可见性/遮挡

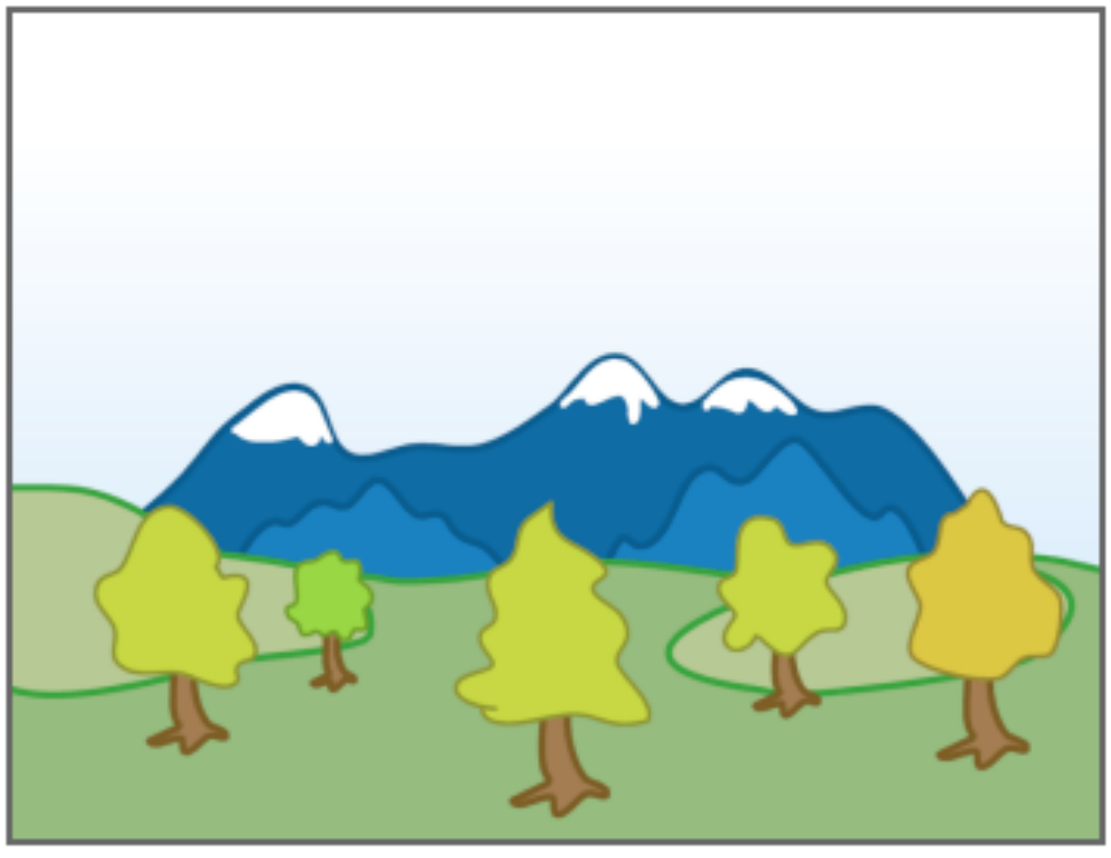
- Visibility / occlusion
  - Z-buffering 深度缓冲
    （Depth Buffering）
- Shading
  - Illumination & Shading
  - Graphics Pipeline

Painter's Algorithm

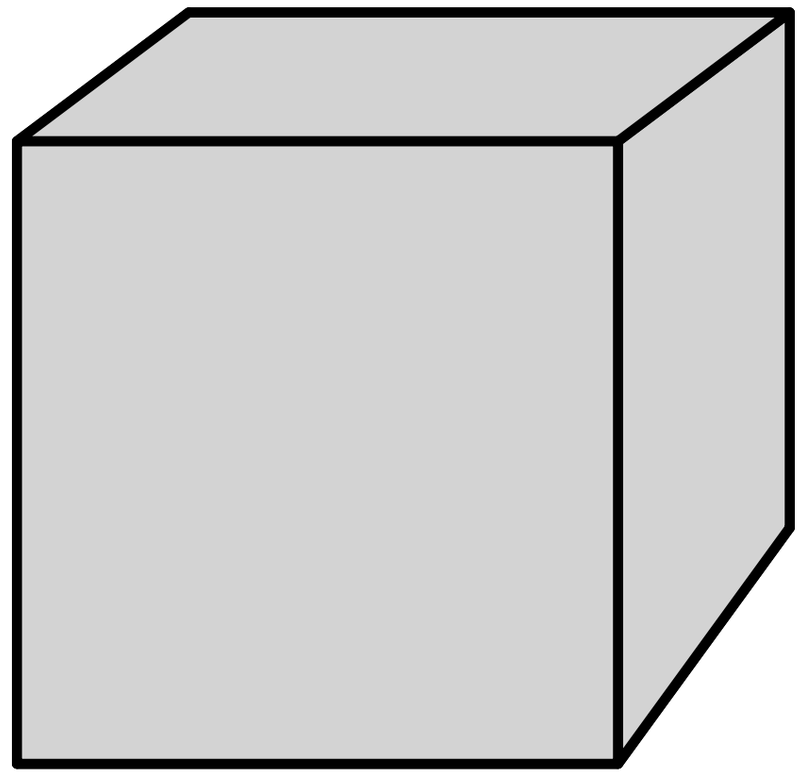
Inspired by how painters paint

Paint from back to front, overwrite in the framebuffer

绘制侧面的顺序有讲究：距离观测点的距离很难定义



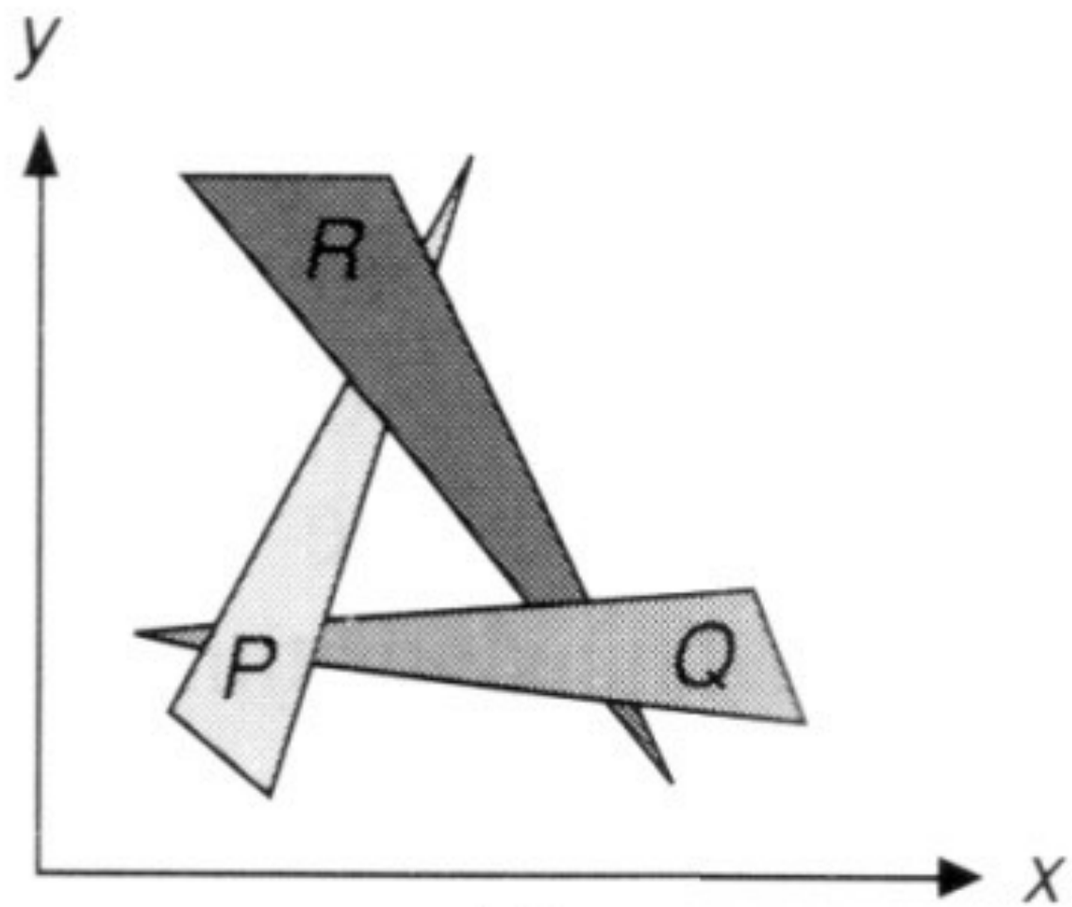
[Wikipedia]



Painter's Algorithm

Requires sorting in depth (O(n log n) for n triangles)

Can have unresolvable depth order



[Foley et al.]

Z-Buffer 深度缓存/深度缓冲

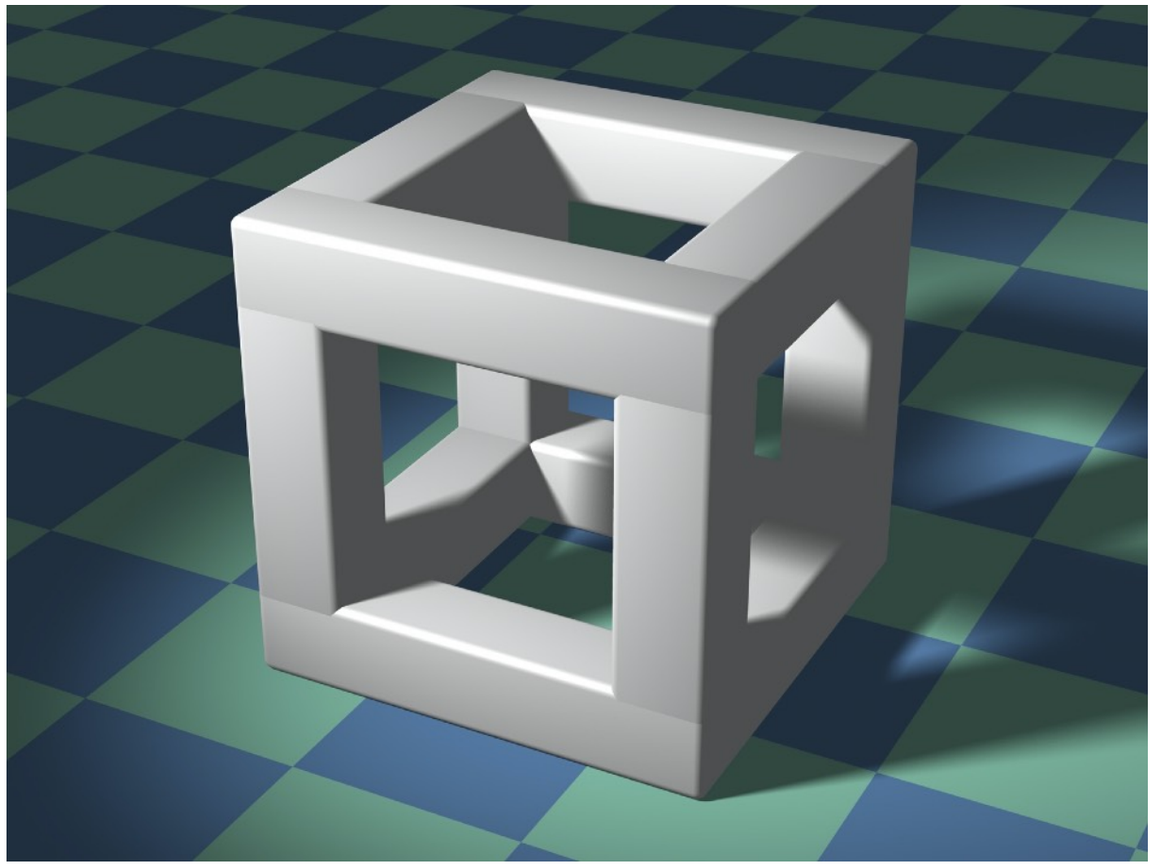
This is the algorithm that eventually won.

Idea:

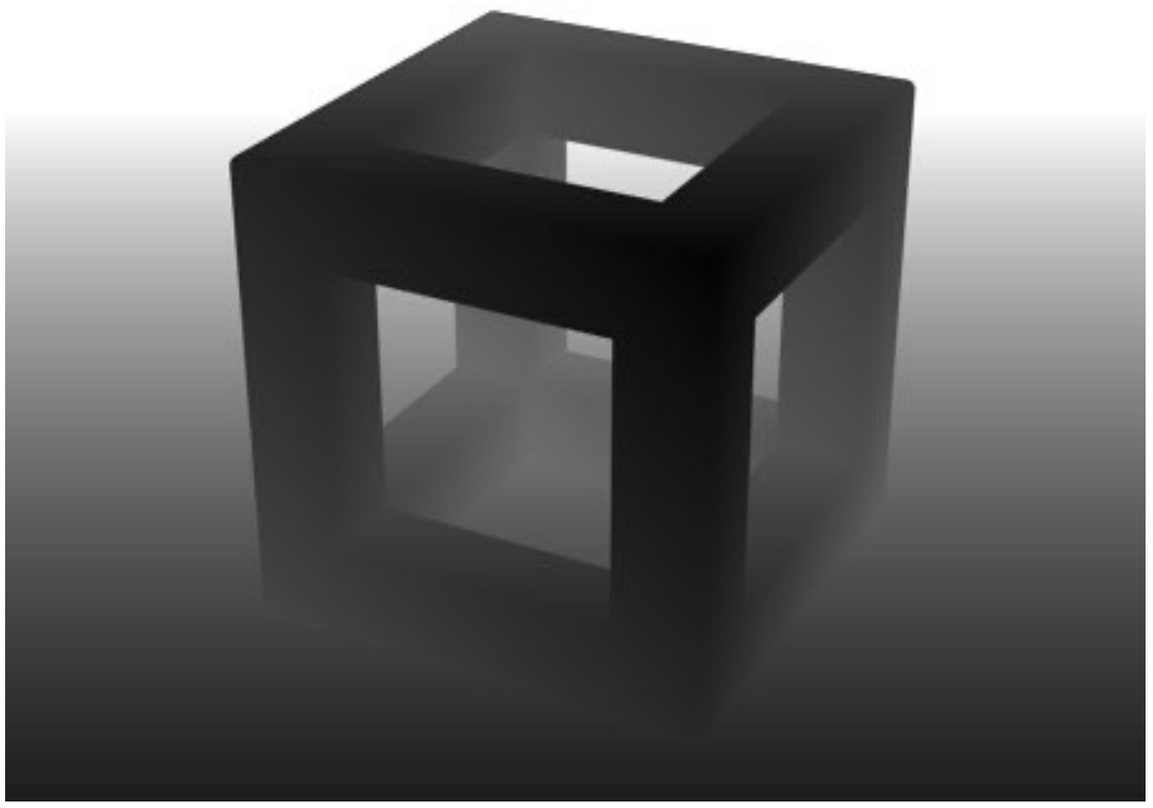
- Store current min. z-value for each sample (pixel)
- Needs an additional buffer for depth values
  - frame buffer stores color values 存储最终结果
  - depth buffer (z-buffer) stores depth 存储任一像素的深度

IMPORTANT: For simplicity we suppose
z is always positive
(smaller z -> closer, larger z -> further)

Z-Buffer Example



Rendering



Depth / Z buffer

Image source: Dominic Alves, flickr.

Z-Buffer Algorithm 深度缓存：每一个像素内记录最浅深度

Initialize depth buffer to ∞ positive z

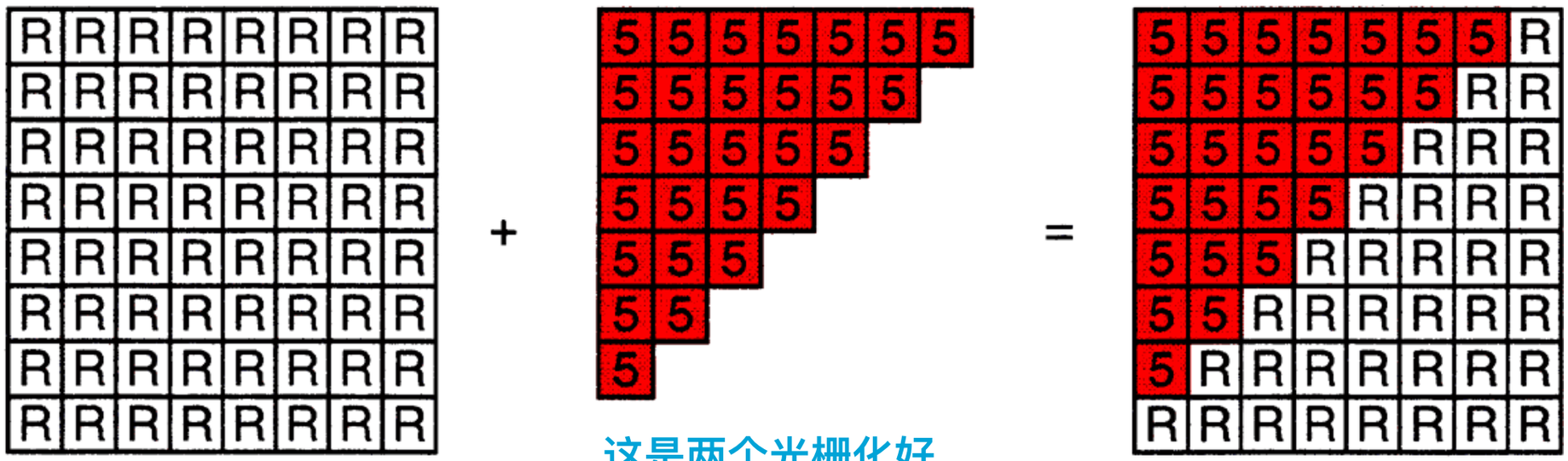
During rasterization: 对画面中的每一个三角形做光栅化

```
for (each triangle T)                    找到任意一个三角形覆盖的某一个像素点（二重循环）
    for (each sample (x,y,z) in T)
        if (z < zbuffer[x,y])                    // closest sample so far
            framebuffer[x,y] = rgb;              // update color
            zbuffer[x,y] = z;                    // update depth
        else
            ;                                    // do nothing, this sample is occluded
```

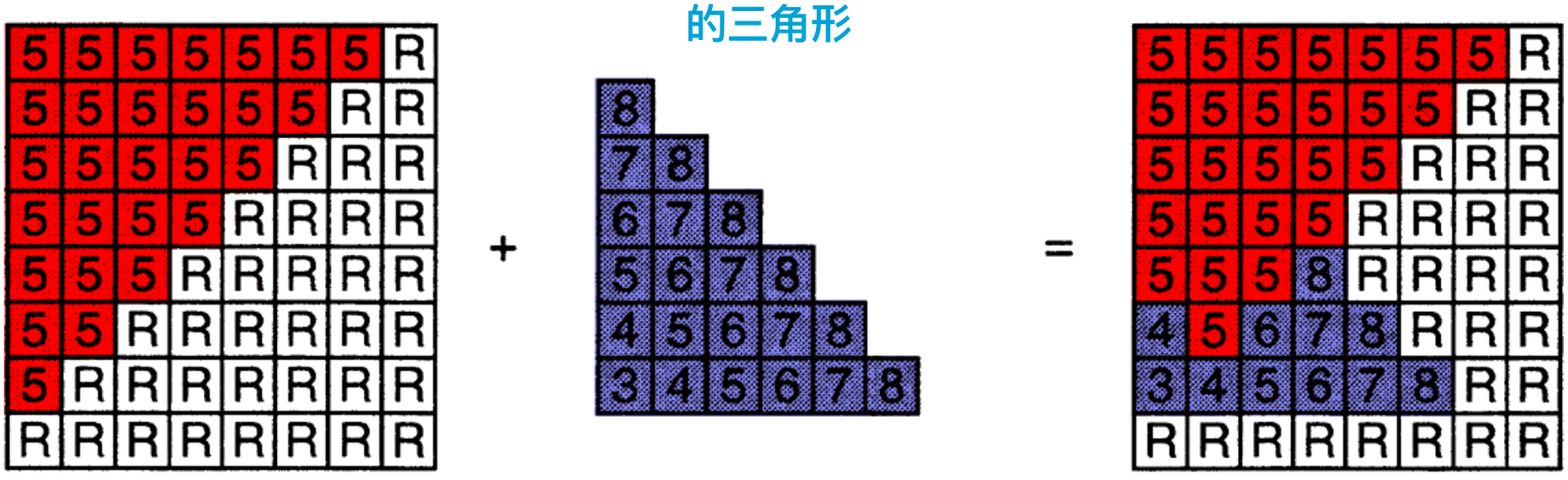
所有操作都在像素上进行

Z-Buffer Algorithm

之后会讲解深度具体如何计算



这是两个光栅化好的三角形



颜色代表framebuffer，数字代表z-buffer

Z-Buffer Complexity

Complexity

- O(*n*) for *n* triangles (assuming constant coverage)
- How is it possible to sort *n* triangles in linear time?

其实这里并没有用到排序，只是挑一个最小值

Drawing triangles in different orders?

Same Result (if no two triangles have same depth)

图形学中一般用浮点型数字表示数据，而浮点型的数据相同是很困难的（实际上还是会有的，概率小）

Most important visibility algorithm

- Implemented in hardware for all GPUs

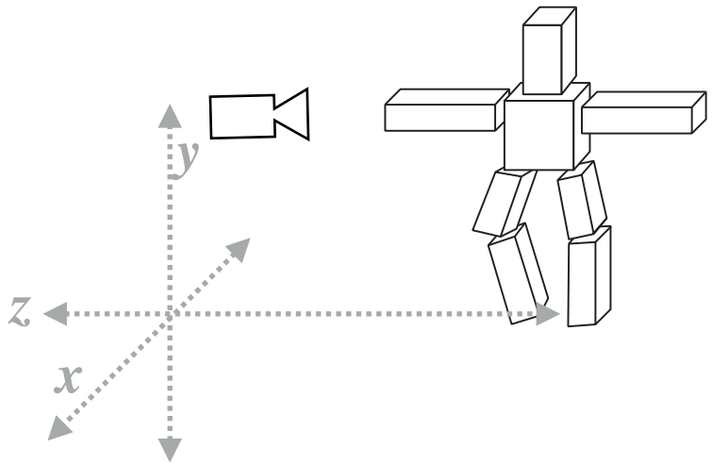
如果使用之前的MSAA反走样，则Zbuffer不止对每个像素做更新，还应该对像素中的每个采样点更新

Questions?

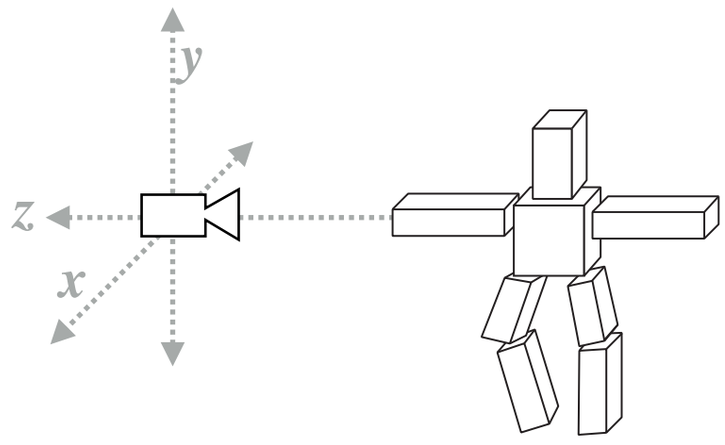
Today

- Visibility / occlusion
  - Z-buffering 无法处理透明物体
- Shading 着色
  - Illumination & Shading
  - Graphics Pipeline

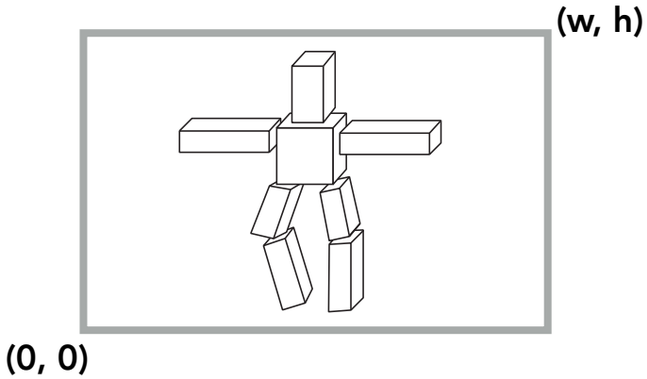
What We've Covered So Far



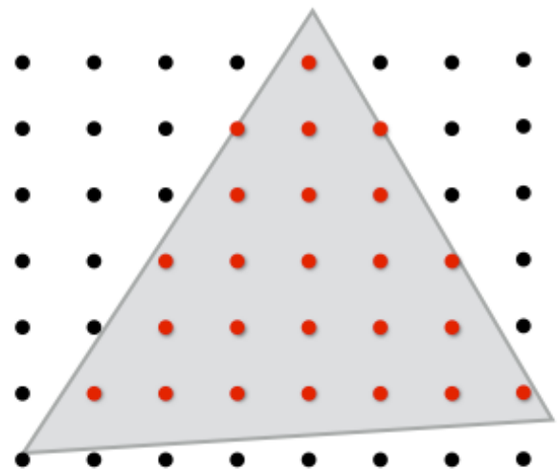
Position objects and the camera in the world



Compute position of objects relative to the camera

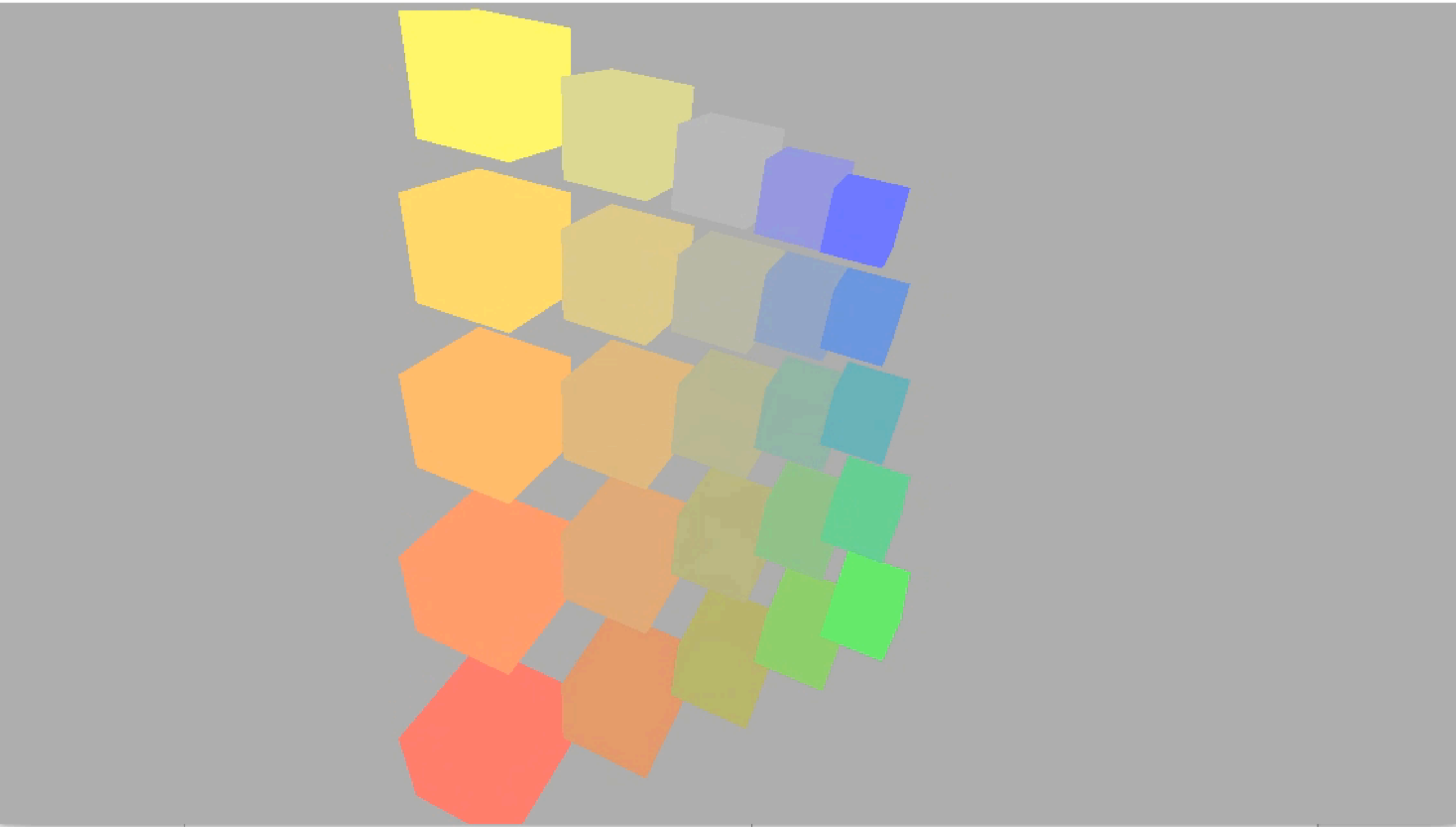


Project objects onto the screen

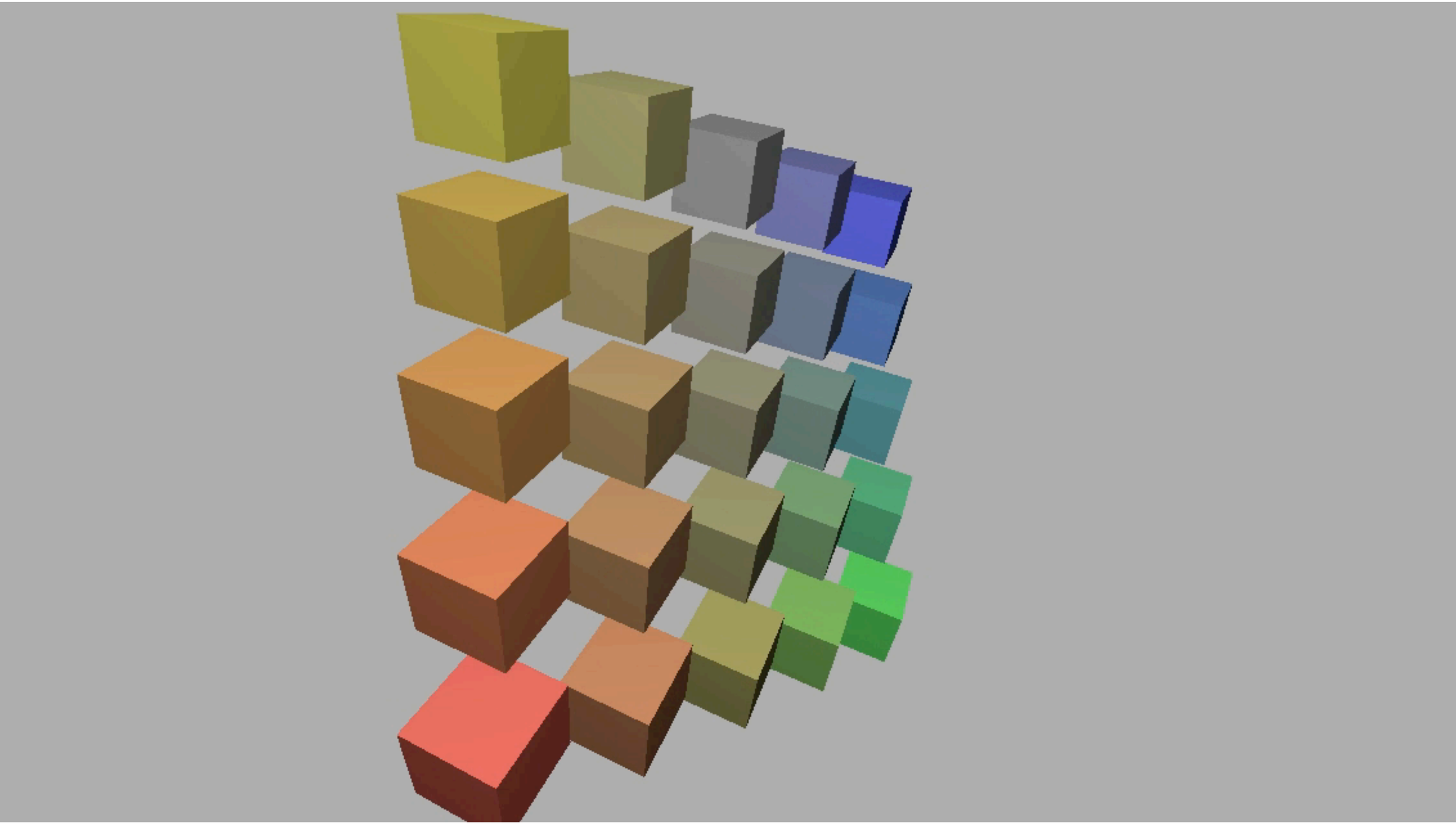


Sample triangle coverage

Rotating Cubes (Now You Can Do)



Rotating Cubes (Expected)



What Else Are We Missing?

Credit: Bertrand Benoit. “Sweet Feast,” 2009. [Blender /VRay]

Shading

Shading: Definition

- In Merriam-Webster Dictionary

  shad·ing, ['ʃeɪdɪŋ], noun
  The darkening or coloring of an illustration or diagram with parallel lines or a block of color.

  明暗 颜色

- In this course

  The process of applying a material to an object.

  对不同物体应用不同的材质

A Simple Shading Model (Blinn-Phong Reflectance Model)

反射模型

Perceptual Observations

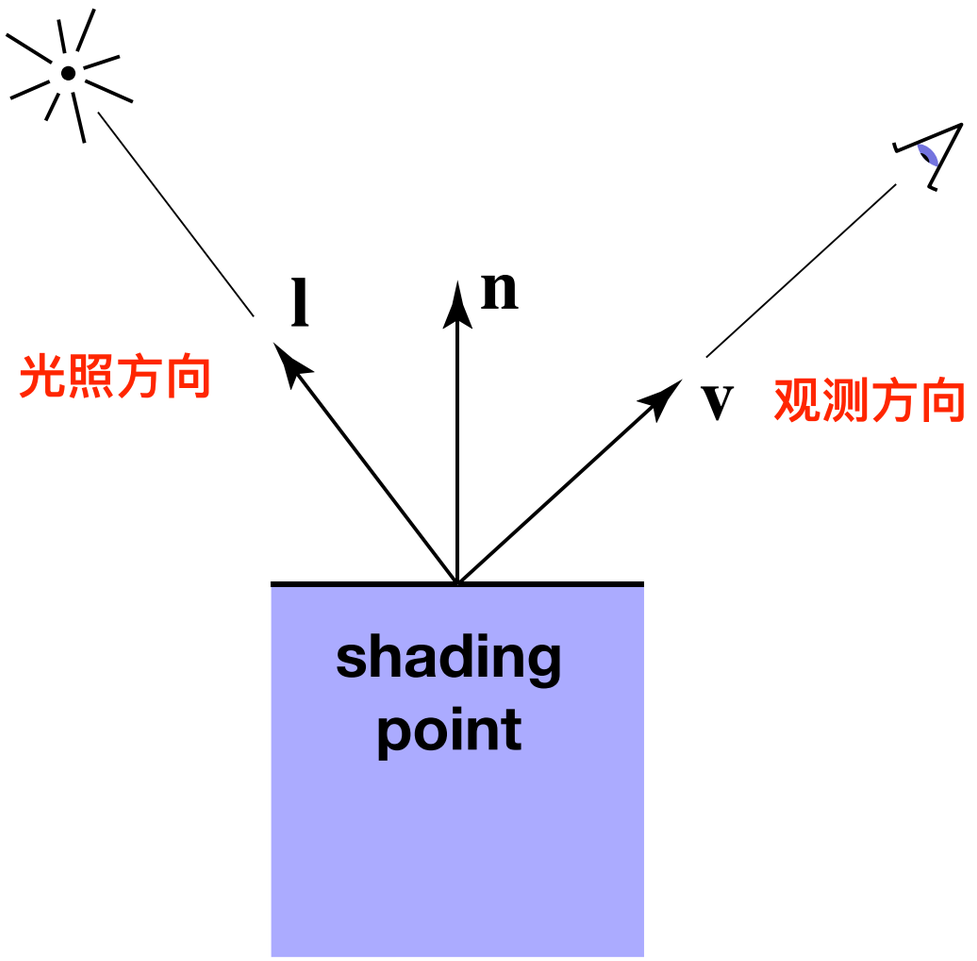
Photo credit: Jessica Andrews, flickr

Shading is Local

Compute light reflected toward camera
at a specific shading point

Inputs:

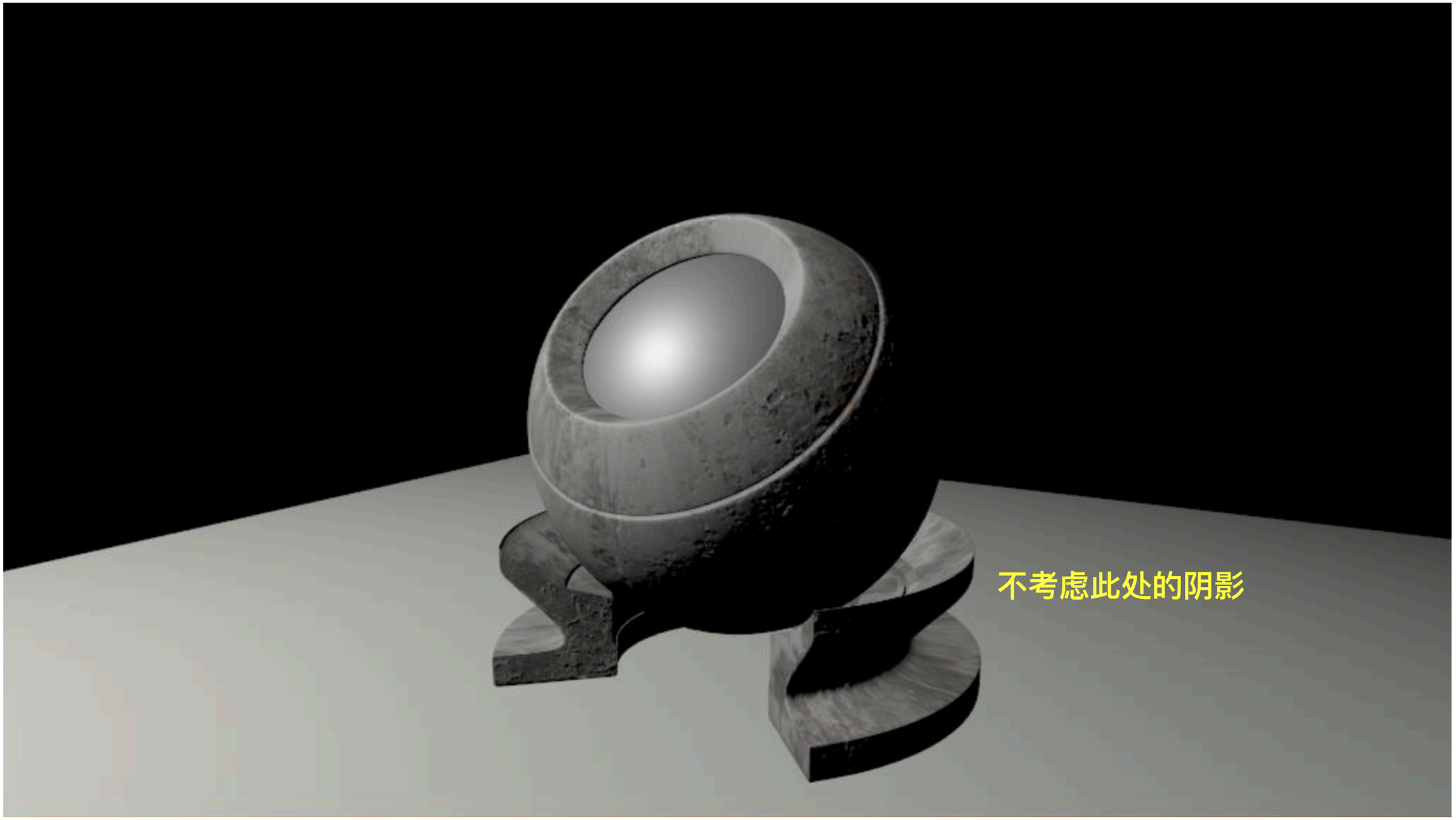
- Viewer direction, v
- Surface normal, n 均为单位向量
- Light direction, l
  (for each of many lights)
- Surface parameters
  (color, shininess, …)



Shading is Local

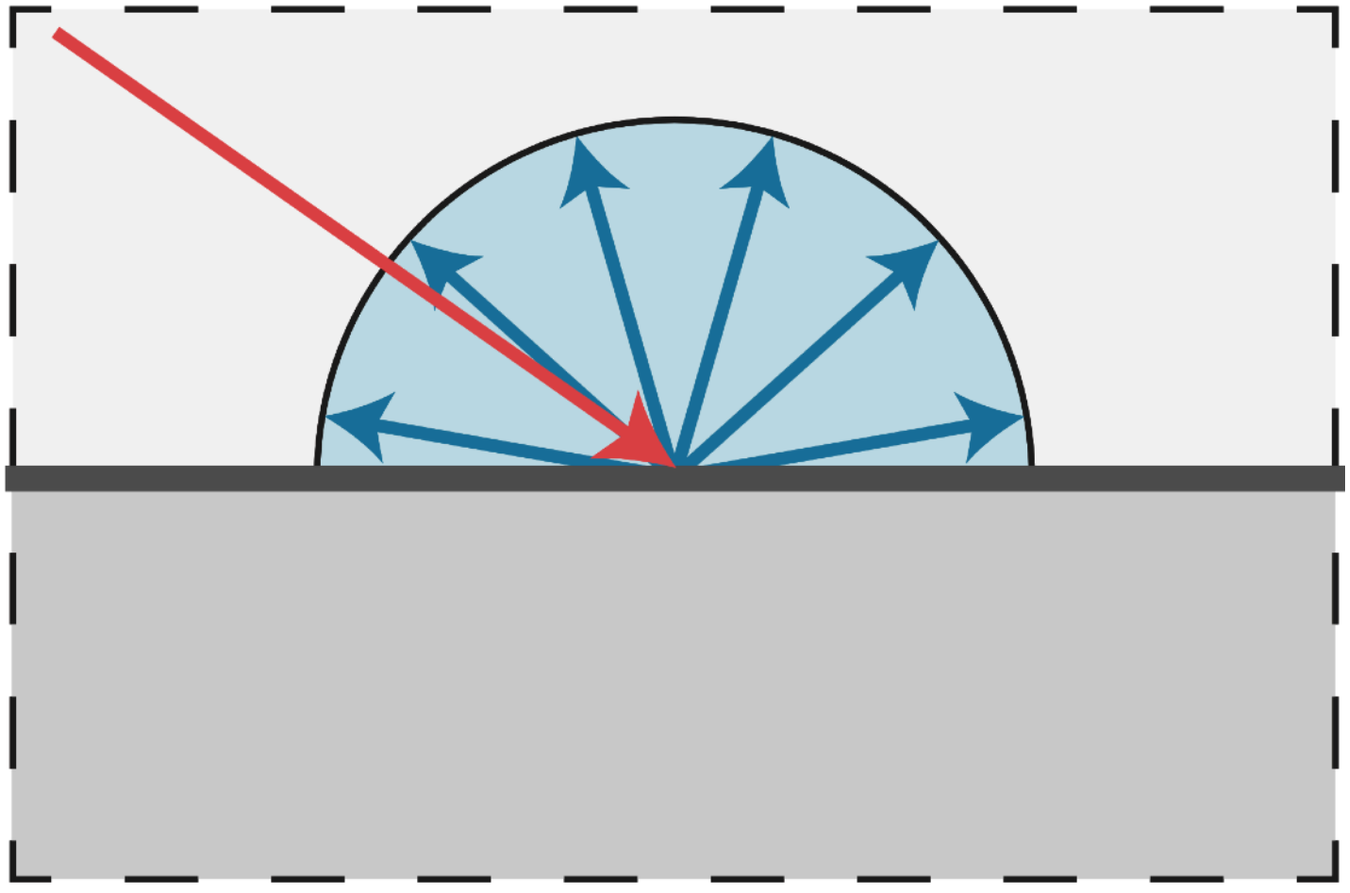
着色 点： 不考虑其他物体的存在

No shadows will be generated! (**shading ≠ shadow**)



Diffuse Reflection

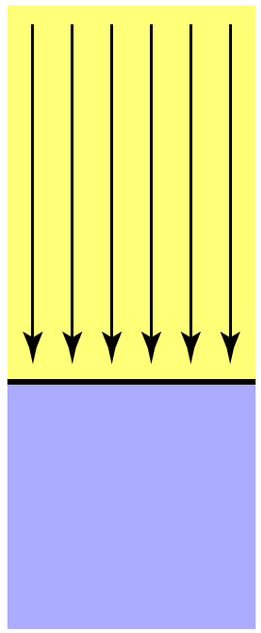
- Light is scattered uniformly in all directions
  - Surface color is the same for all viewing directions



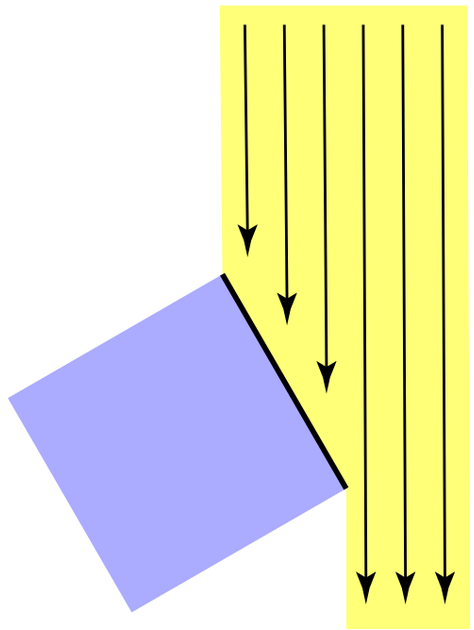
Diffuse Reflection

- But how much light (energy) is received?
  - Lambert's cosine law

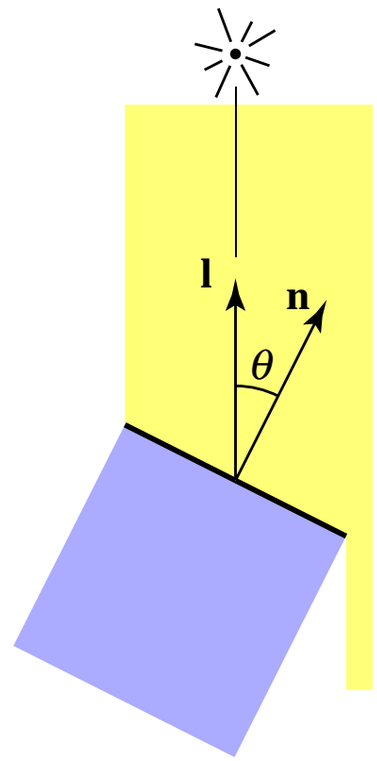
考虑着色点周围的面元（单位面积）



Top face of cube receives a certain amount of light

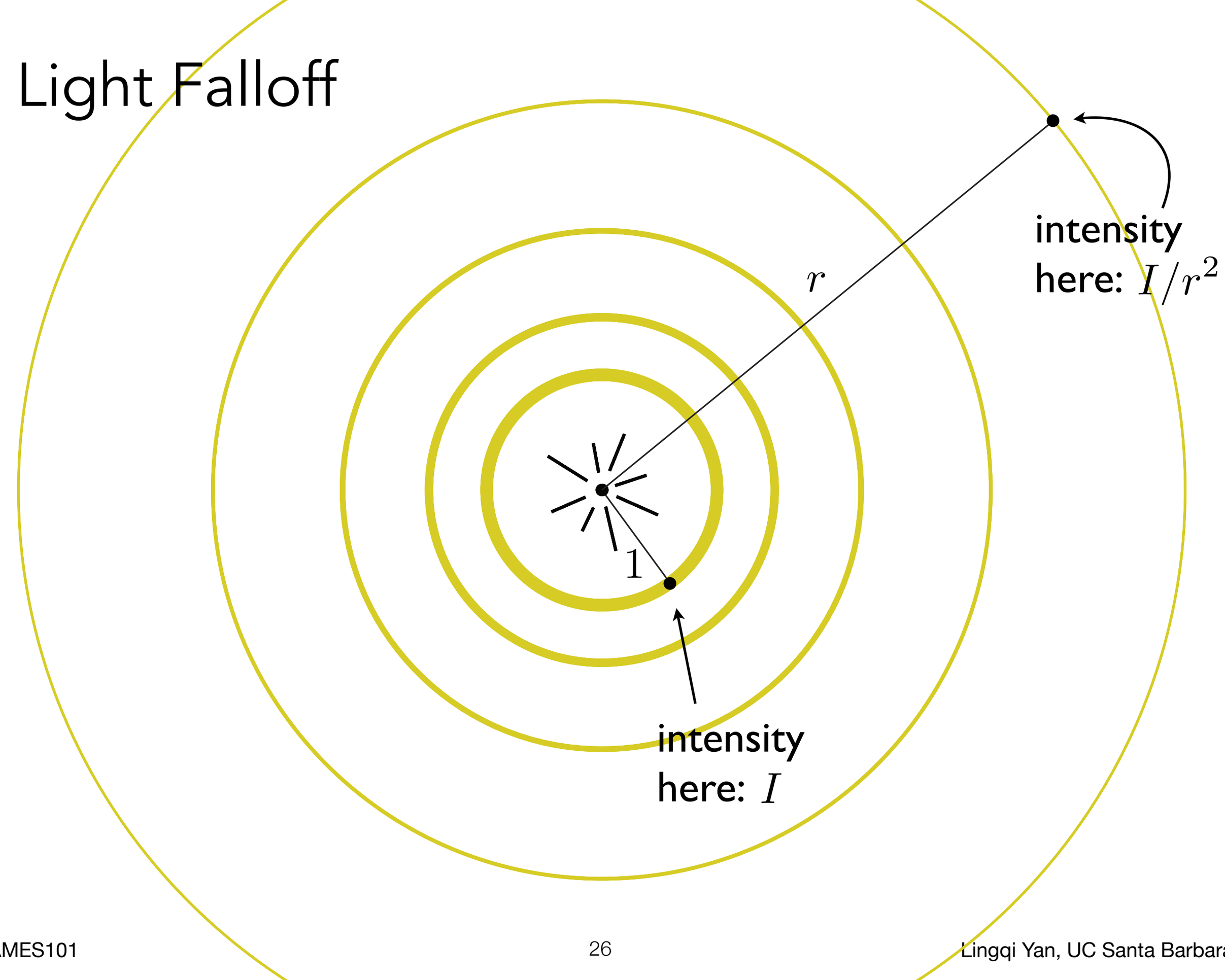


Top face of 60° rotated cube intercepts half the light



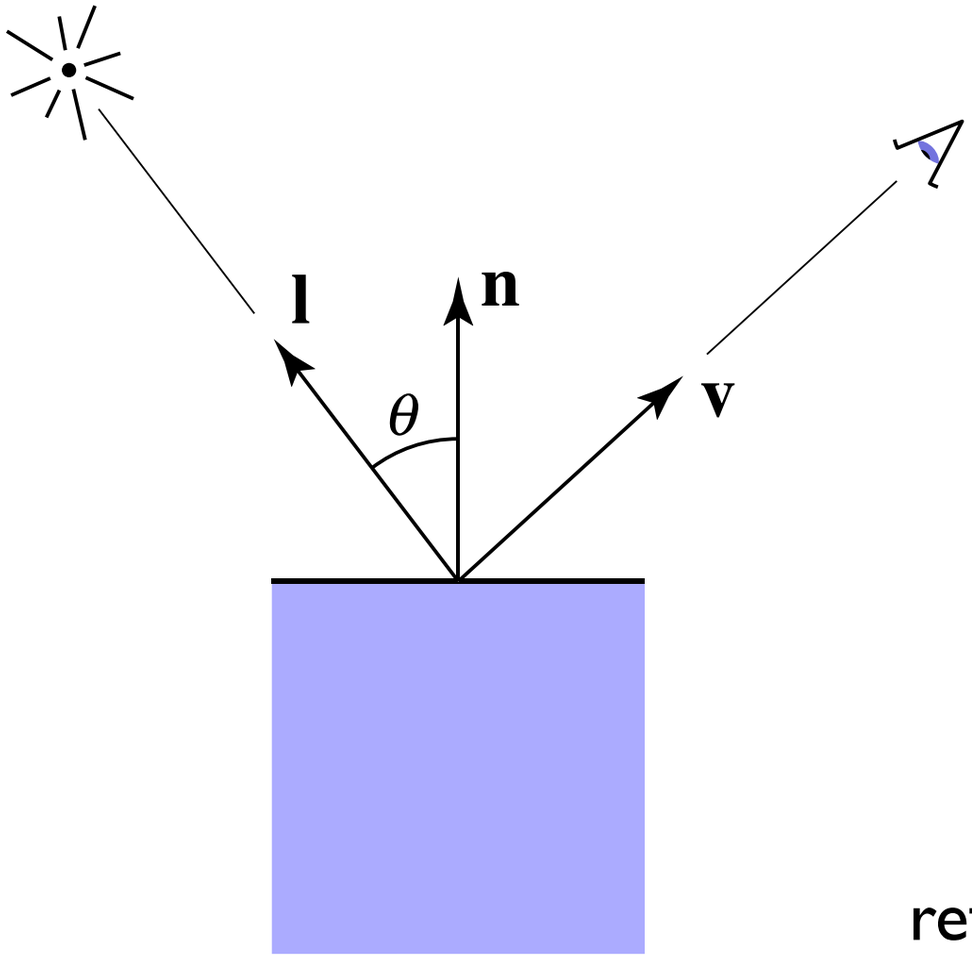
In general, light per unit area is proportional to $\cos\theta = l \cdot n$

Light Falloff



Lambertian (Diffuse) Shading

Shading independent of view direction



energy arrived
at the shading point

kd=1，全反射

漫反射系数

$$L_d = k_d\,(I/r^2)\max(0, \mathbf{n}\cdot\mathbf{l})$$

diffusely reflected light

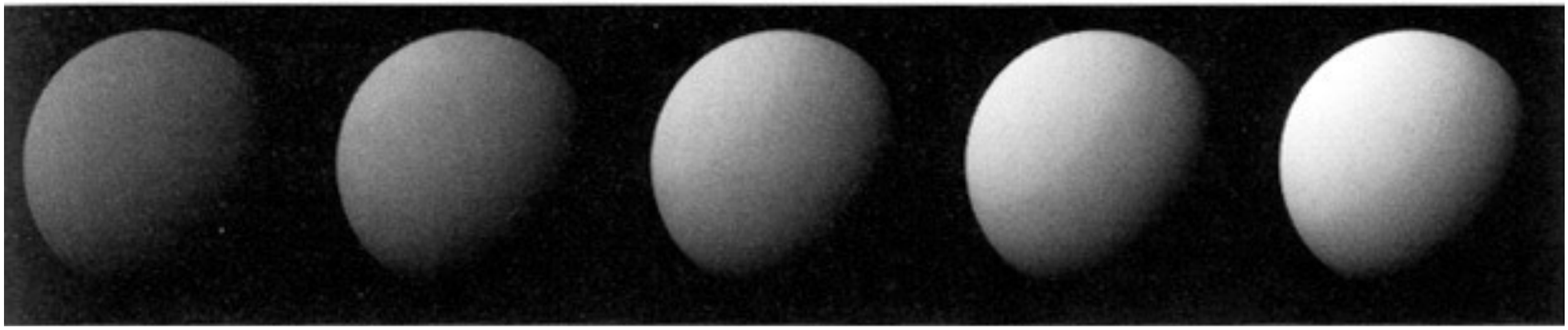
diffuse coefficient (color)

energy received by the shading point

漫反射项与观测角度v无关

Lambertian (Diffuse) Shading

Produces diffuse appearance



[Foley et al.]

kd小

k_d ⟶

kd大

Thank you!

(And thank Prof. Ravi Ramamoorthi and Prof. Ren Ng for many of the slides!)