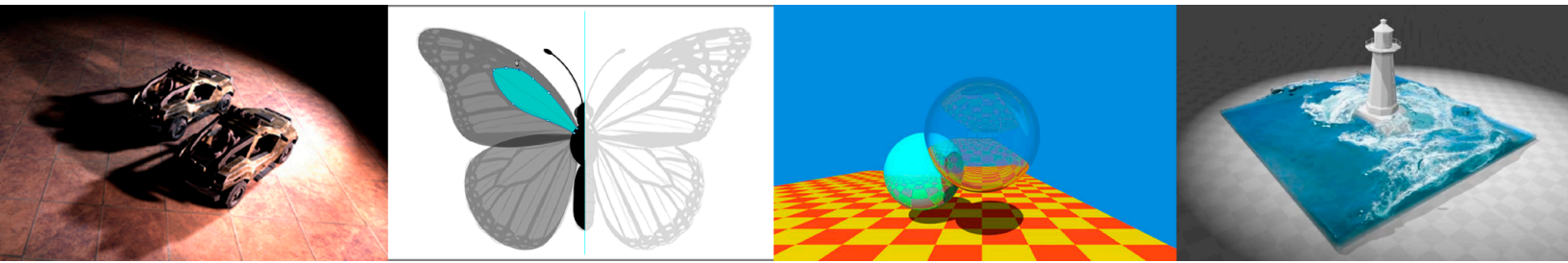
Introduction to Computer Graphics

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

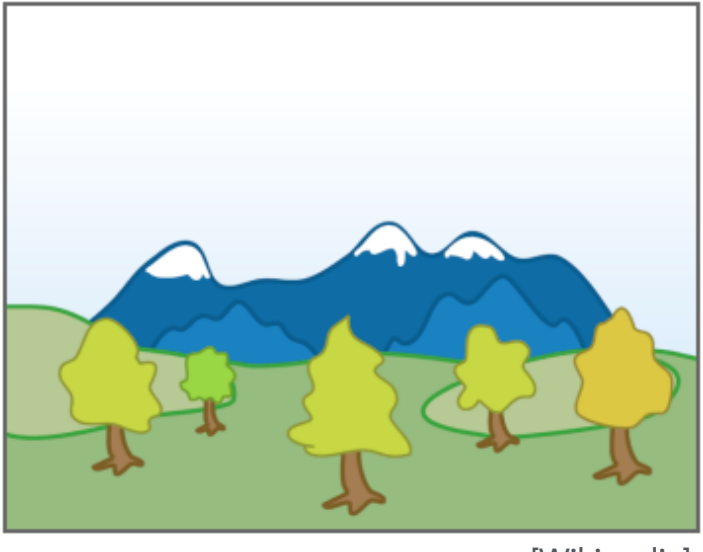
- Visibility / occlusion
  - Z-buffering 深度缓冲
- Shading
  - Illumination & Shading
  - Graphics Pipeline

Painter's Algorithm 画家算法

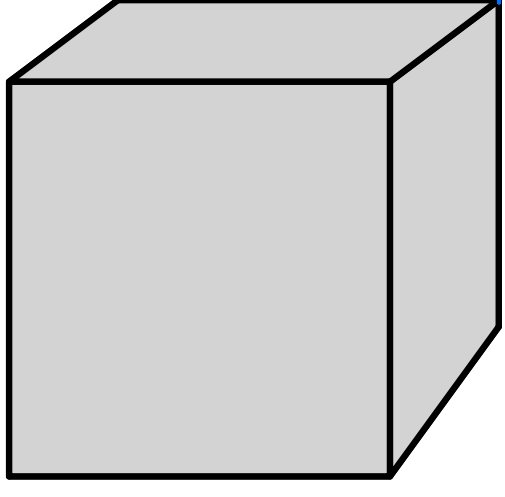
Inspired by how painters paint

Paint from back to front, overwrite in the framebuffer

先把远处物体光栅化，再把近处物体光栅化，并覆盖远处物体。



[Wikipedia]



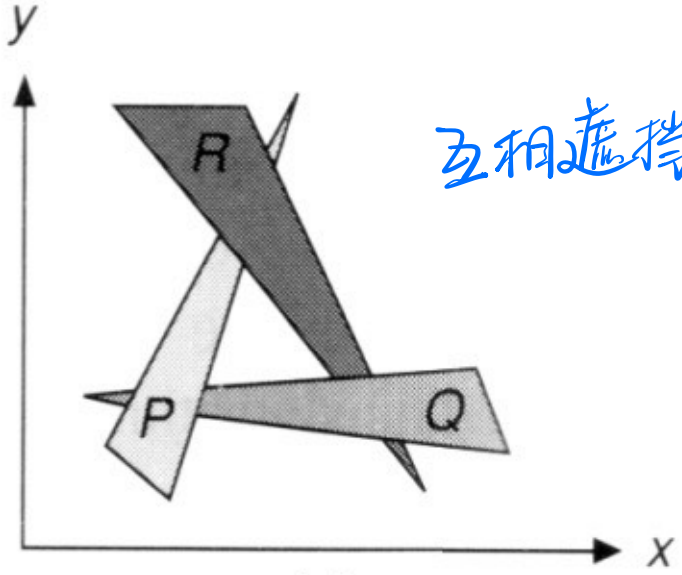
Painter's Algorithm

深度排序

对三角形时间复杂度 O(nlogn)

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

Can have unresolvable depth order



互相遮挡？

[Foley et al.]

Z-Buffer

This is the algorithm that eventually won.

Idea:

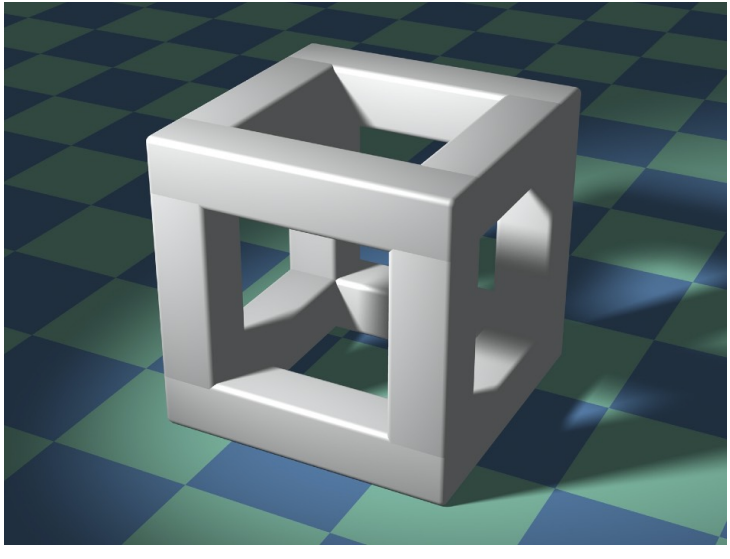
- Store current min. z-value for each sample (pixel)
- Needs an additional buffer for depth values
  - frame buffer stores color values
  - depth buffer (z-buffer) stores depth

IMPORTANT: For simplicity we suppose

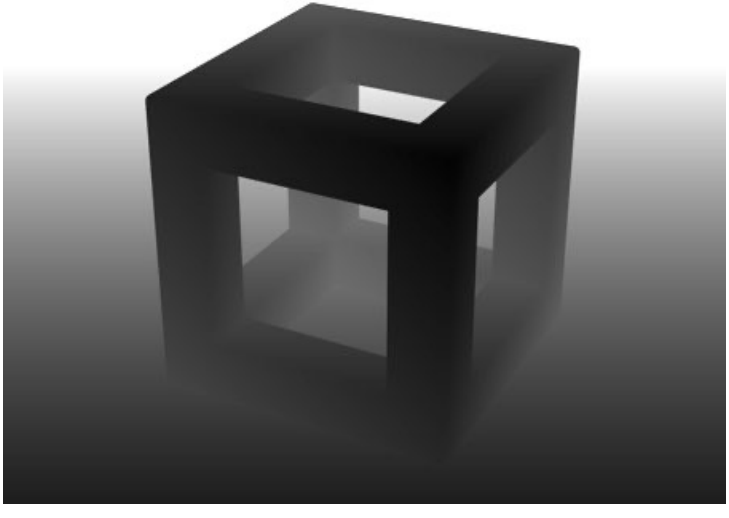
z is always positive

(smaller z -> closer, larger z -> further)

Z-Buffer Example



Rendering



Depth / Z buffer

Image source: Dominic Alves, flickr.

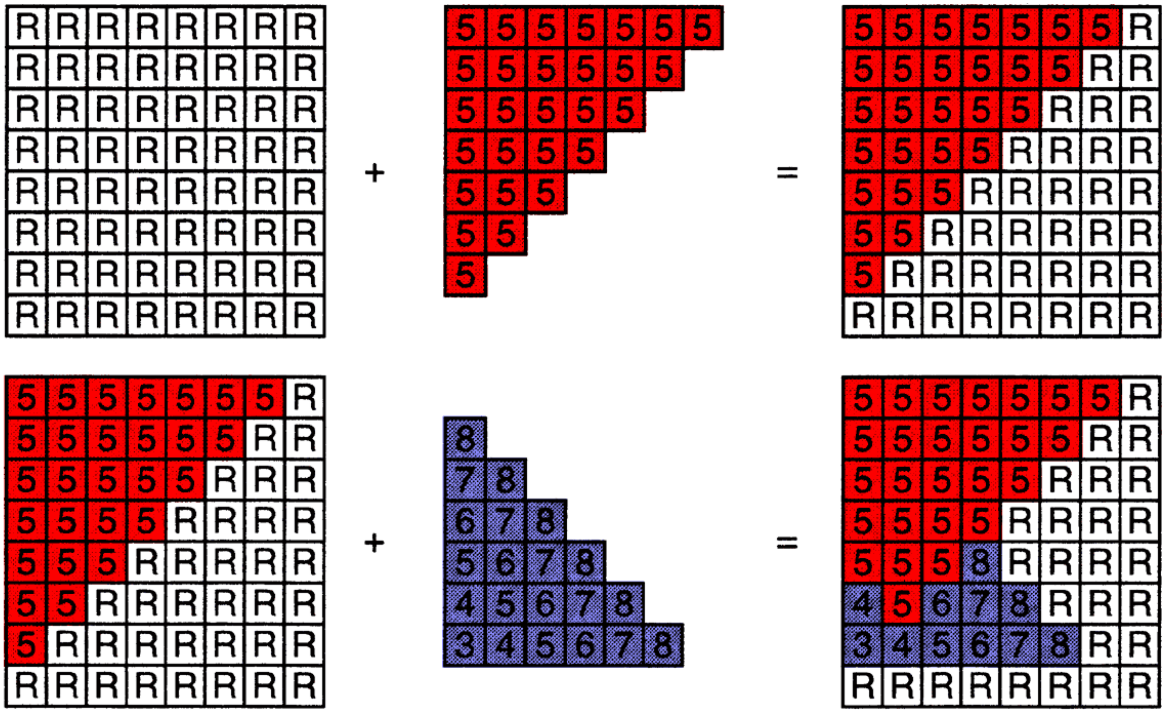
Z-Buffer Algorithm

Initialize depth buffer to ∞

During rasterization:

```
for (each triangle T)
    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
```

Z-Buffer Algorithm



Z-Buffer Complexity

Complexity

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

(并没有排序. 只是对每个像素记录深度最小值)

Drawing triangles in different orders?

和顺序无关.

Most important visibility algorithm

- Implemented in hardware for all GPUs

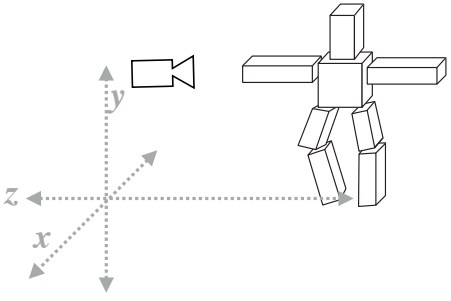
※ 考虑到MSAA, 深度缓存不仅需要对每个像素值记录深度, 还要对每个采样点记录深度.

Questions?

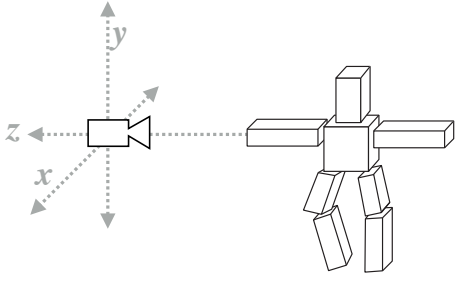
Today

- Visibility / occlusion
  - Z-buffering
- Shading 着色
  - Illumination & Shading
  - Graphics Pipeline

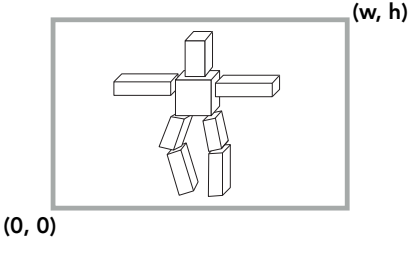
What We've Covered So Far



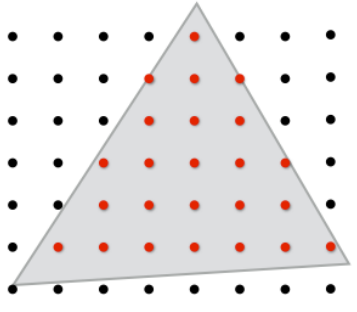
Position objects and the camera in the world



Compute position of objects relative to the camera

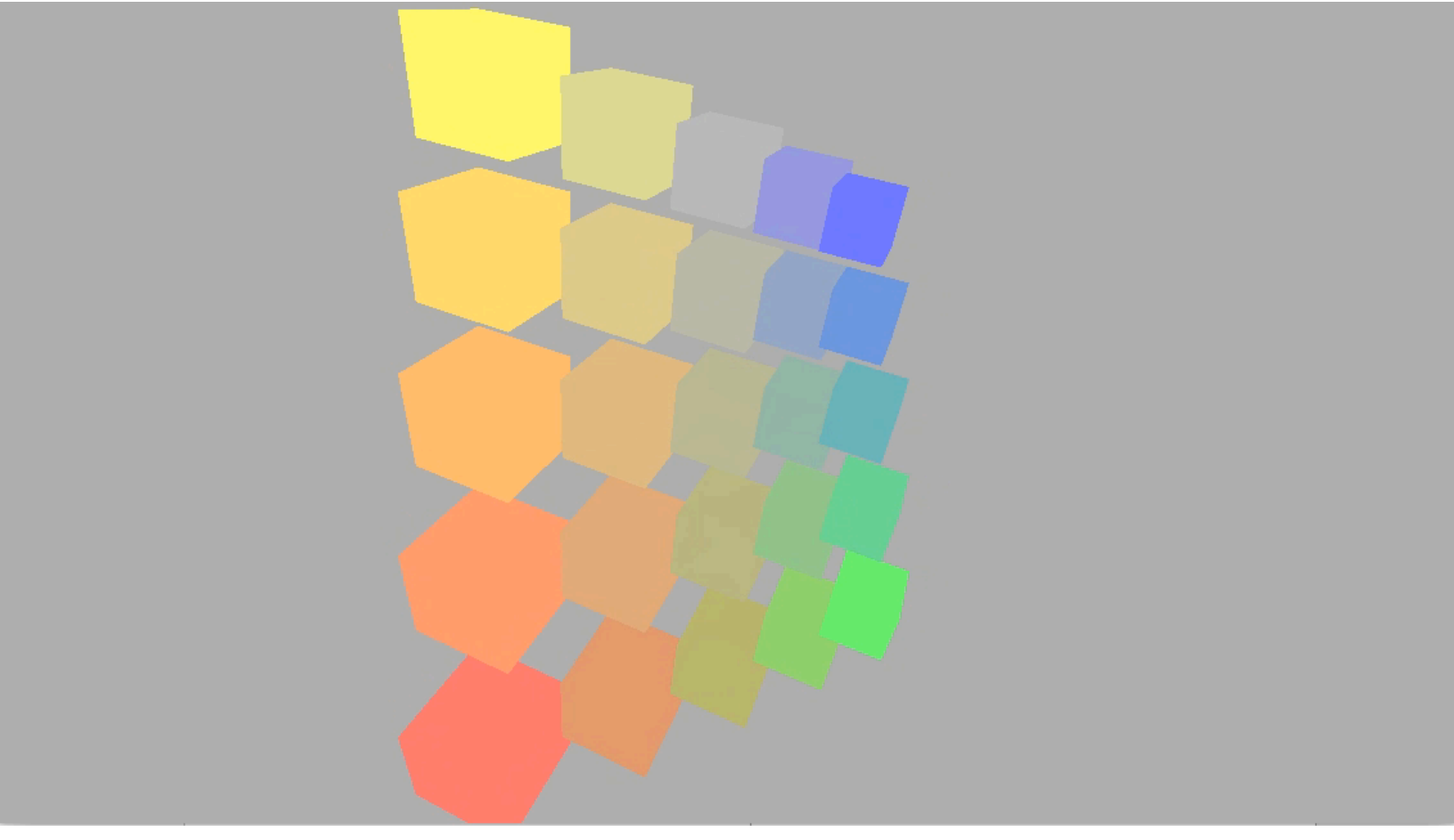


Project objects onto the screen

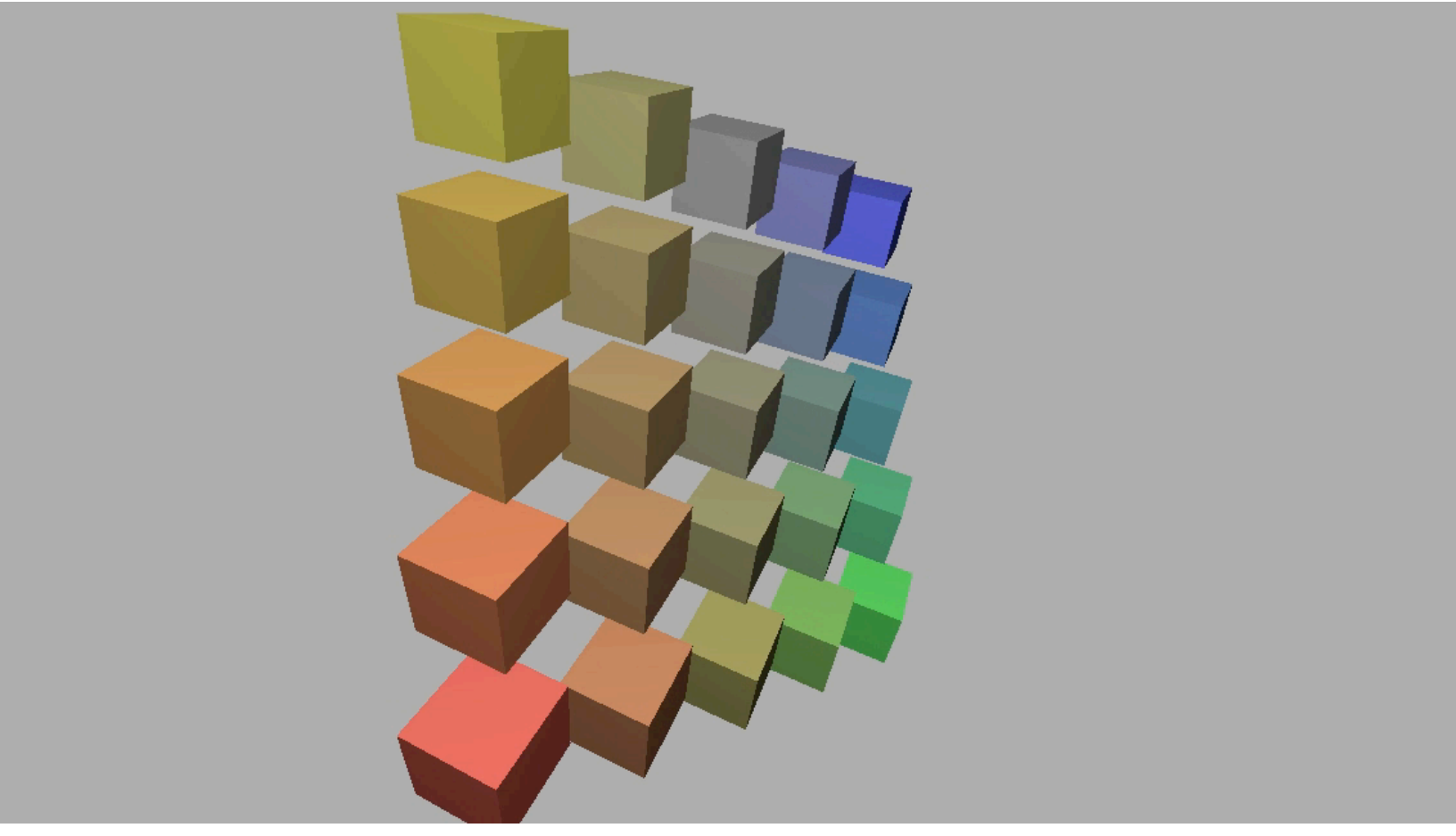


Sample triangle coverage

Rotating Cubes (Now You Can Do)



Rotating Cubes (Expected)



What Else Are We Missing?

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

Shading

着色、

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.

- In this course

  The process of applying a material to an object.

  对不同的物体应用不同材质的过程.

A Simple Shading Model (Blinn-Phong Reflectance Model)

反射模型

Perceptual Observations

Photo credit: Jessica Andrews, flickr

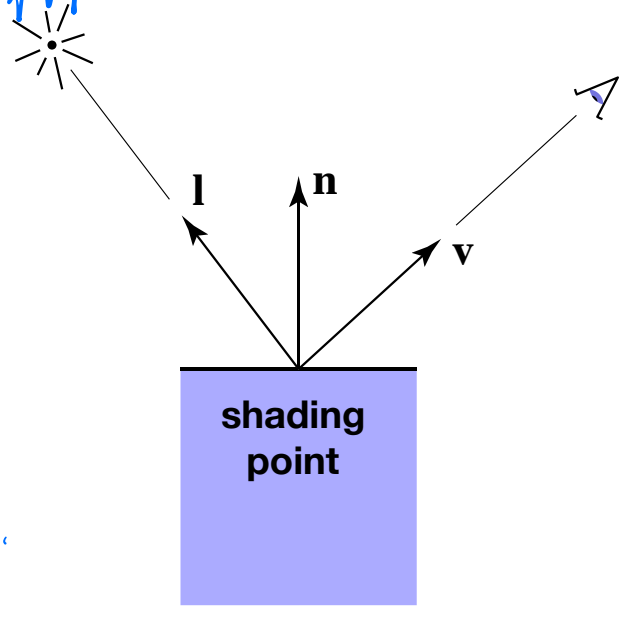
Shading is Local

Compute light reflected toward camera at a specific shading point

在一个特定的阴影点计算光反射到相机。

Inputs:

- Viewer direction, $\hat{v}$ 观察方向 <单位向量>
- Surface normal, $\hat{n}$ 法线
- Light direction, $\hat{l}$ 光照方向 (for each of many lights)
- Surface parameters 表面参数 (color, shininess, …)



Shading is Local "局部性"

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

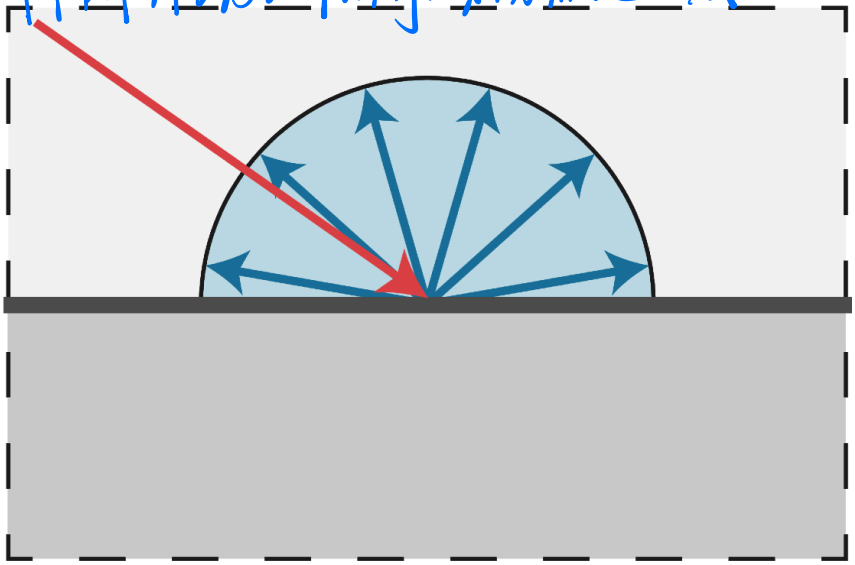
不产生阴影 着色 ≠ 阴影

Diffuse Reflection

光均匀地向各个方向散射.

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

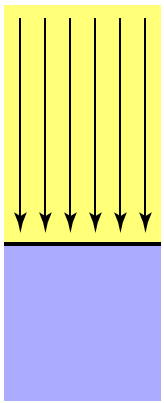
对于所有观测方向，表面颜色一致.



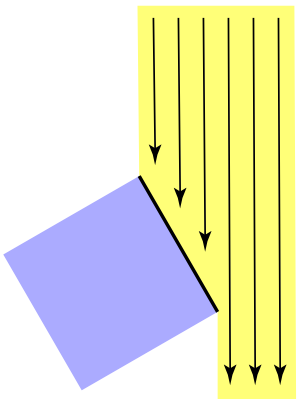
Diffuse Reflection

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

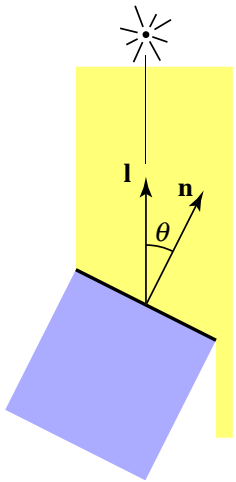
"Lambert's 余弦定律"



Top face of cube receives a certain amount of light



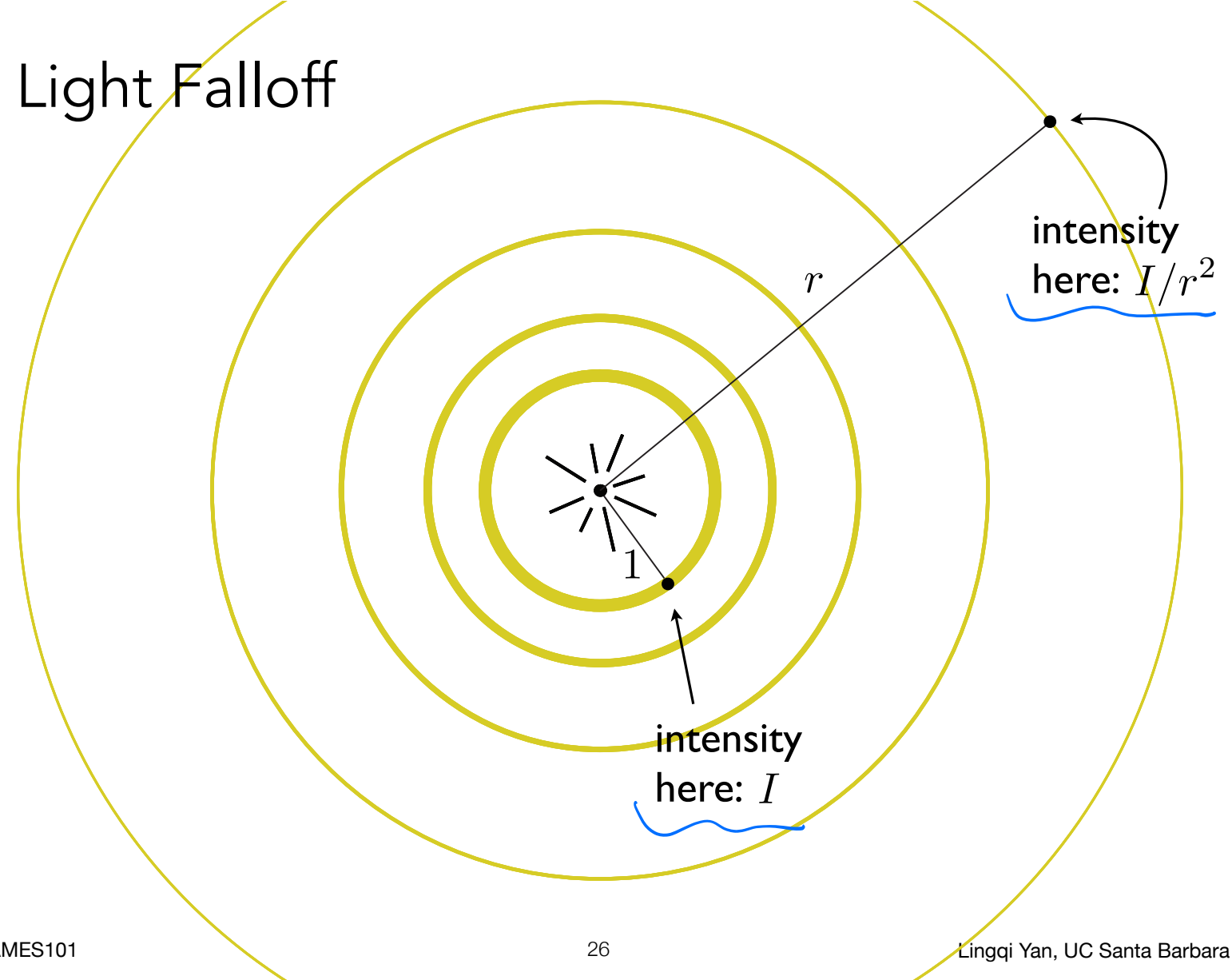
Top face of 60° rotated cube intercepts half the light



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

单位面积接收的光 $\propto \cos\theta = \hat{l} \cdot \hat{n}$ 成正比.

Light Falloff

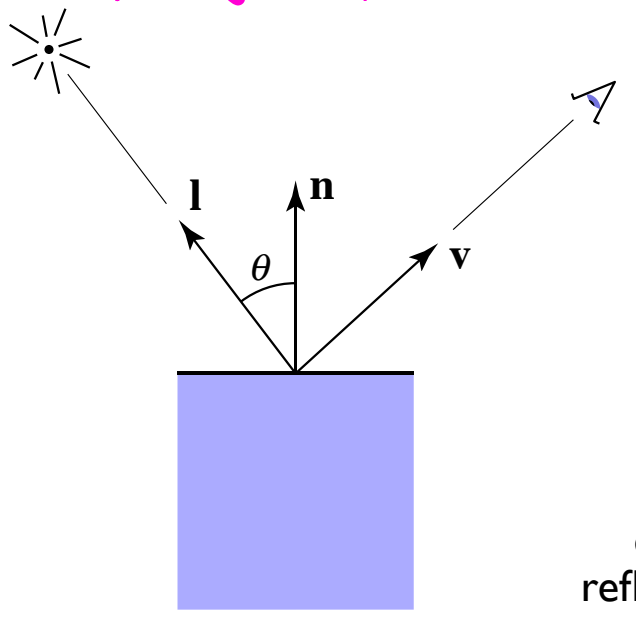


Lambertian (Diffuse) Shading

着色独立于观测方向.

Shading independent of view direction

从哪个方向观测，反射光都一样.



energy arrived at the shading point

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

除以半径平方.

漫反射系数.

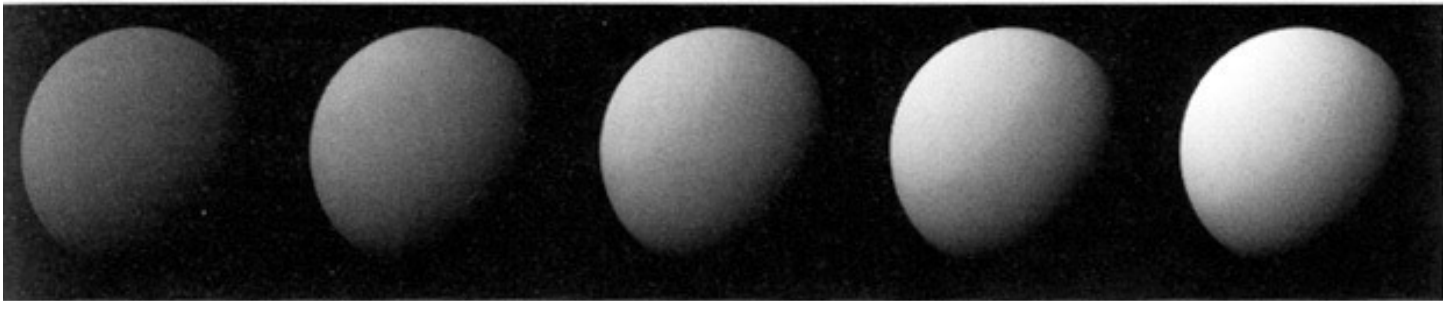
diffuse coefficient (color)

energy received by the shading point

diffusely reflected light

Lambertian (Diffuse) Shading

Produces diffuse appearance



k_d ↓ $k_d \longrightarrow$ k_d ↑

[Foley et al.]

Thank you!

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