



Lighting and Shading (Part I)

Computer Graphics

Yu-Ting Wu

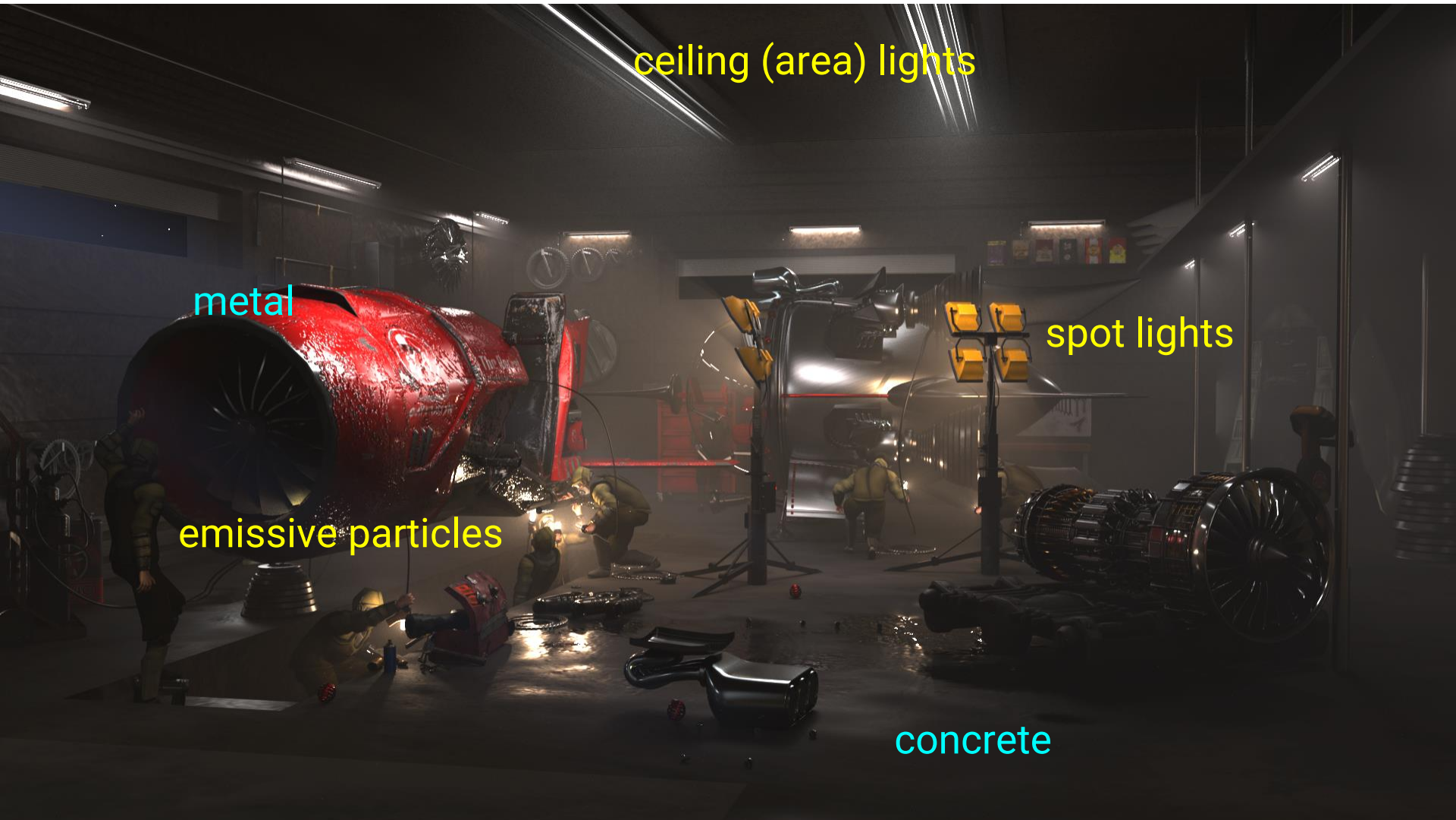
Outline

- [Overview](#)
 - [Lights](#) (Part I)
 - [Materials](#)
-
- Material file format (Part II)
 - OpenGL implementation

Outline

- **Overview**
- Lights
- Materials
- Material file format
- OpenGL implementation

Shading: Materials and Lighting

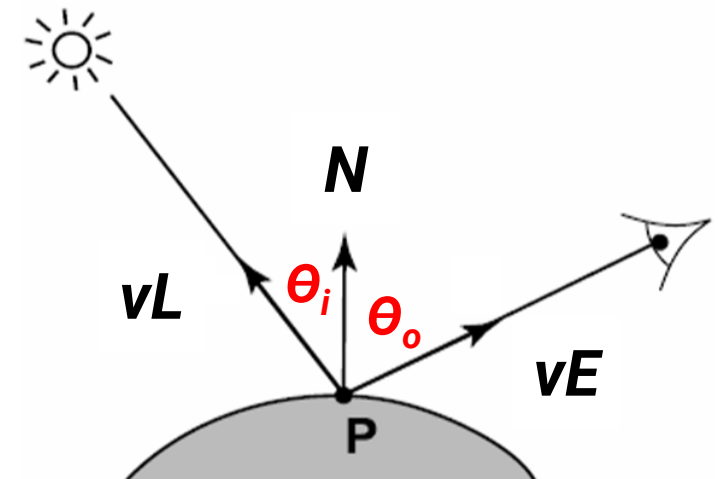


Shading: Materials and Lighting (cont.)



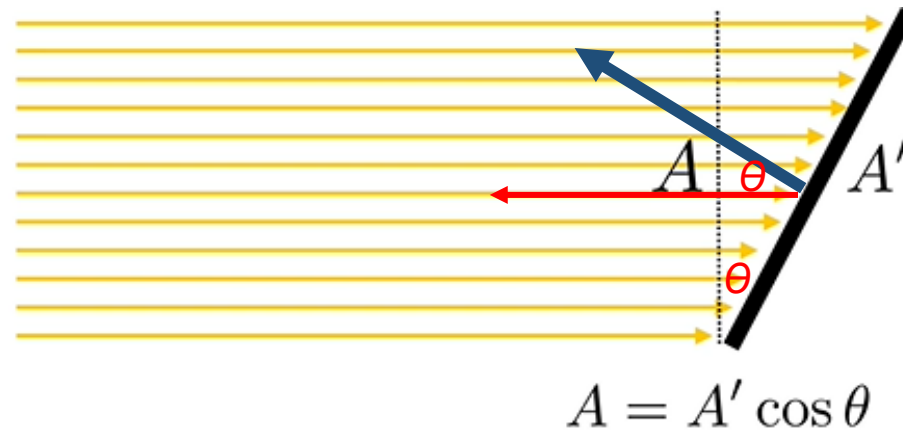
Shading

- Shading refers to the process of altering the color of an object/surface/polygon in the 3D scene
- In physically-based rendering, shading tries to approximate the **local behavior** of lights on the object's surface, based on things like
 - Surface orientation (normal) N
 - Lighting direction \mathbf{vL} (and θ_i)
 - Viewing direction \mathbf{vE} (and θ_o)
 - Material properties
 - Participating media
 - etc.



Lambertian Cosine Law

- Illumination on an oblique surface is less than on a normal one
- Generally, illumination falls off as **cos θ**



$$E = \frac{\Phi}{A'} = \frac{\Phi \cos \theta}{A}$$

Outline

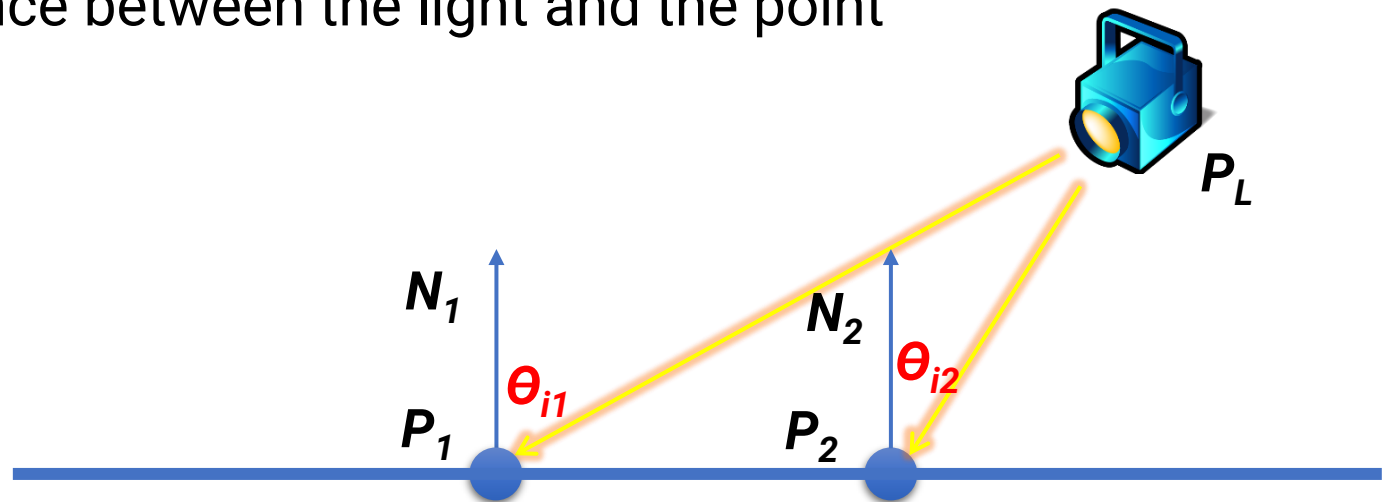
- Overview
- **Lights**
- Materials
- Material file format
- OpenGL implementation

Lights in Computer Graphics

- Point light
 - Spot light
 - Area light
- } local lights
-
- Directional light
 - Environment light
- } distant lights

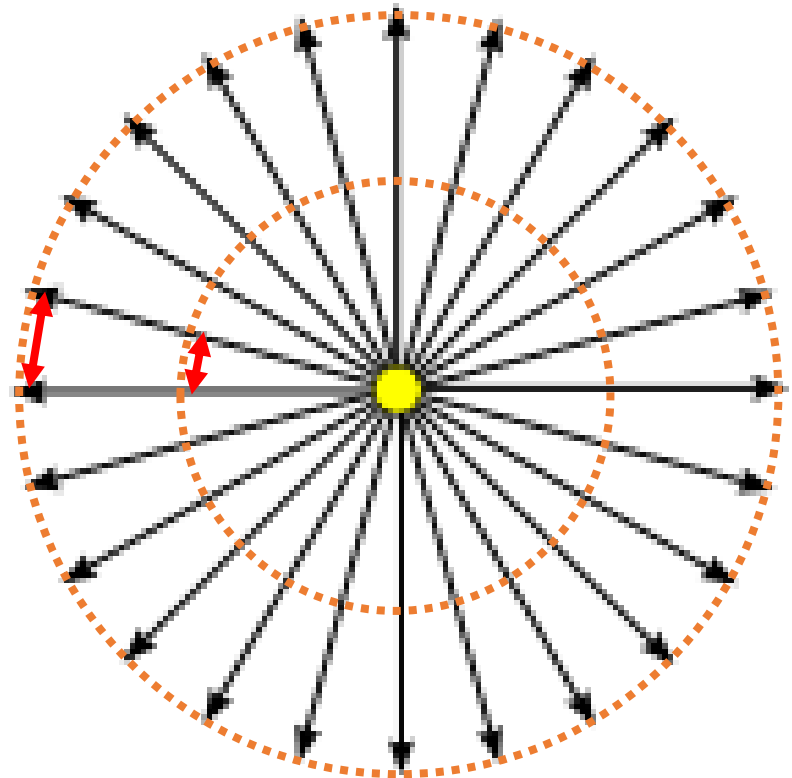
Local Light

- The distance between a light and a surface is **NOT** long enough compared to the scene scale
- The position of light needs to be considered during shading
 - **Lighting direction** $\mathbf{v}_L = |\mathbf{P}_L - \mathbf{P}|$
 - **Lighting attenuation** is proportional to the square of the distance between the light and the point

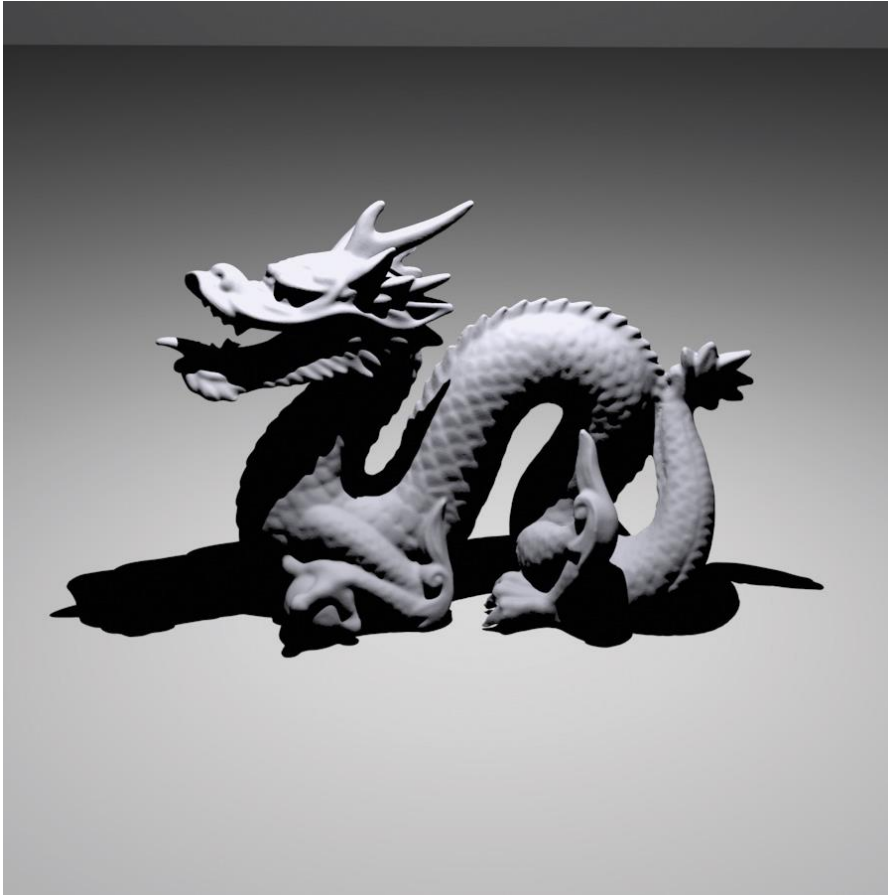


Local Light Attenuation

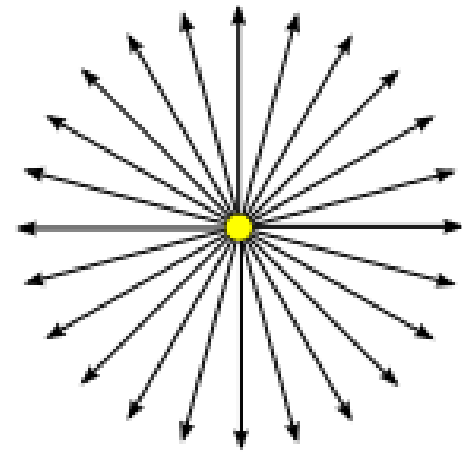
- The length of the side of a receiver patch is proportional to its distance from the light
- As a result, the average energy per unit area is proportional to the **square of the distance** from the light



Point Light

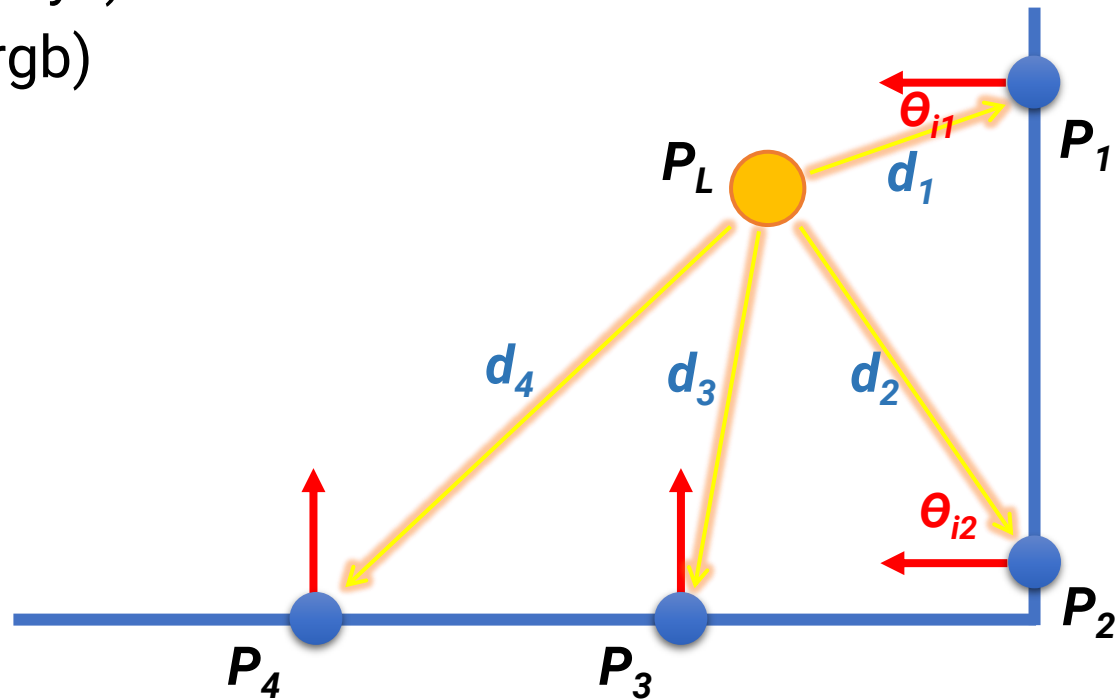
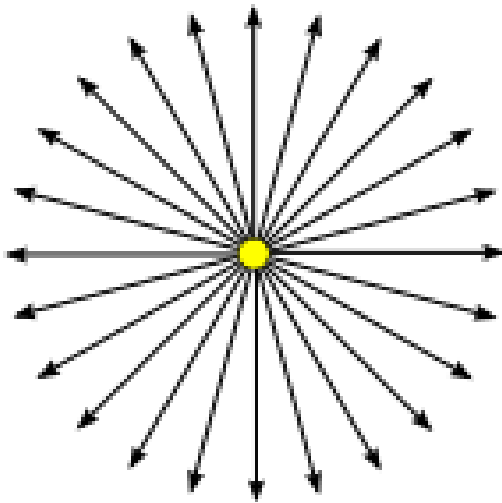


A scene illuminated by a point light



Point Light (cont.)

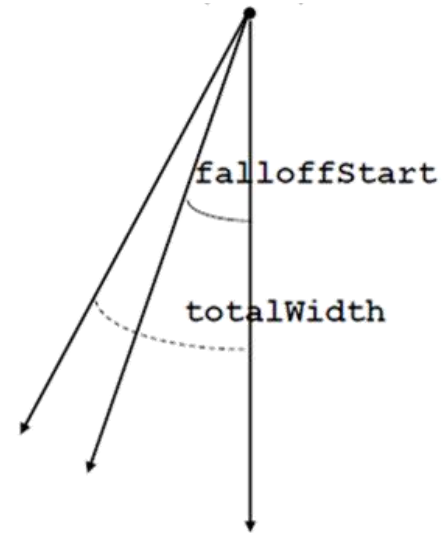
- An isotropic point light source that emits the same amount of light in all directions
- Described by
 - Light position (\mathbf{P}_L , xyz)
 - Light intensity (\mathbf{I} , rgb)



Spot Light

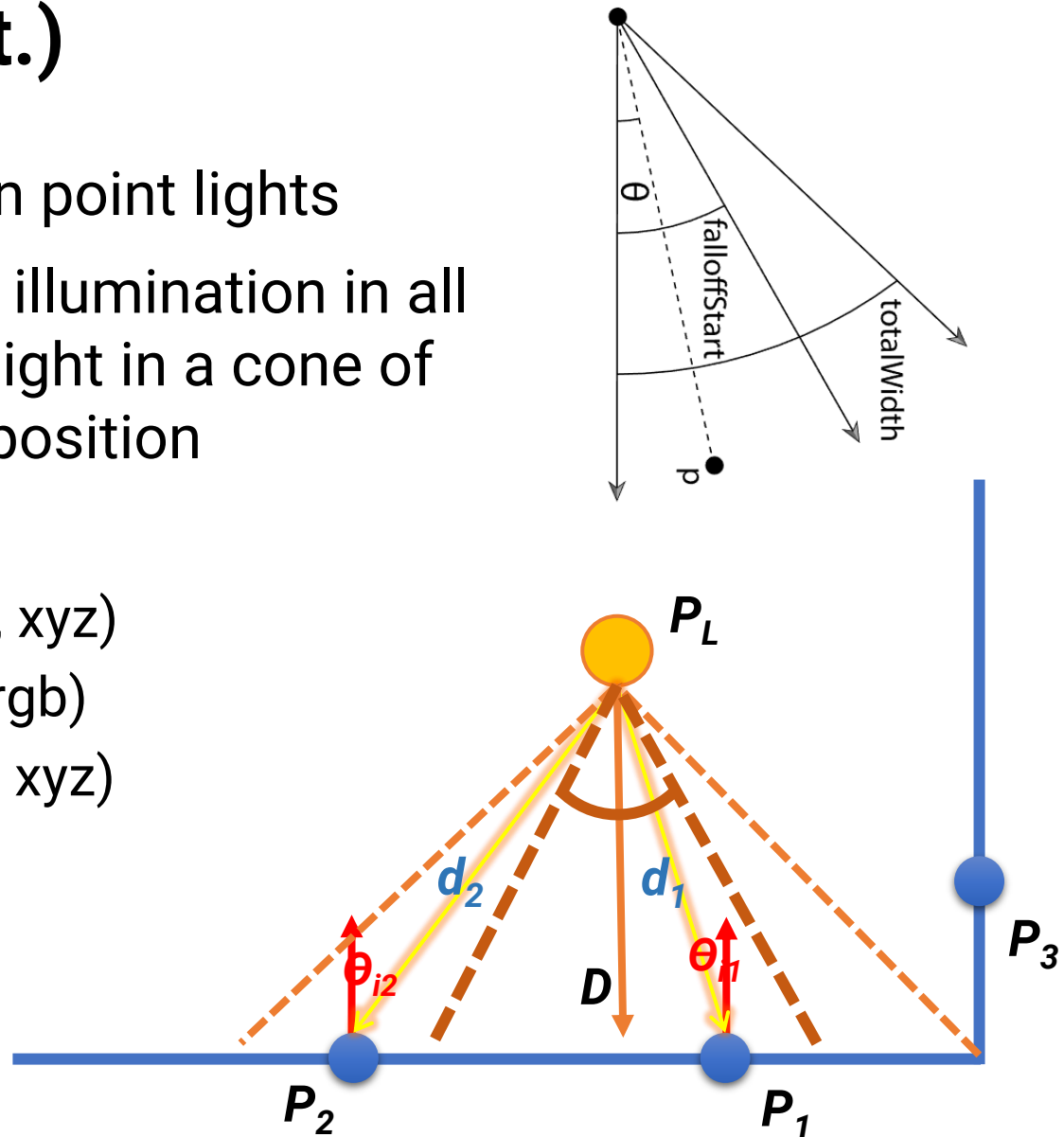


A scene illuminated by a spot light



Spot Light (cont.)

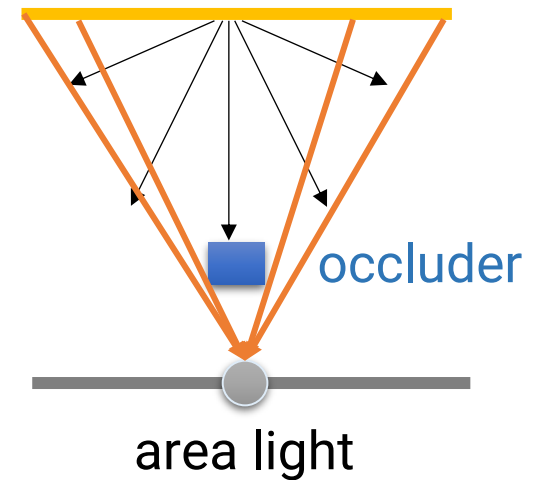
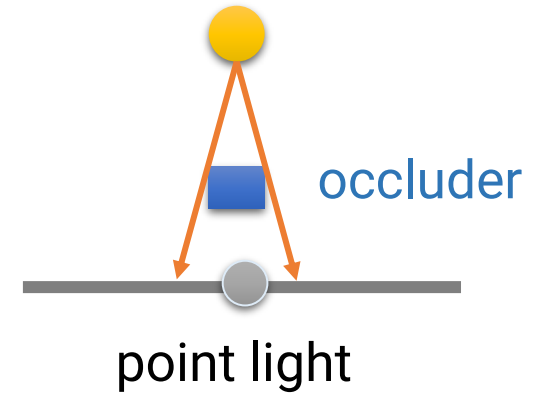
- A handy variation on point lights
- Rather than shining illumination in all directions, it emits light in a cone of directions from its position
- Described by
 - Light position (\mathbf{P}_L , xyz)
 - Light intensity (\mathbf{I} , rgb)
 - Light direction (\mathbf{D} , xyz)
 - TotalWidth
 - FalloffStart



Area Light

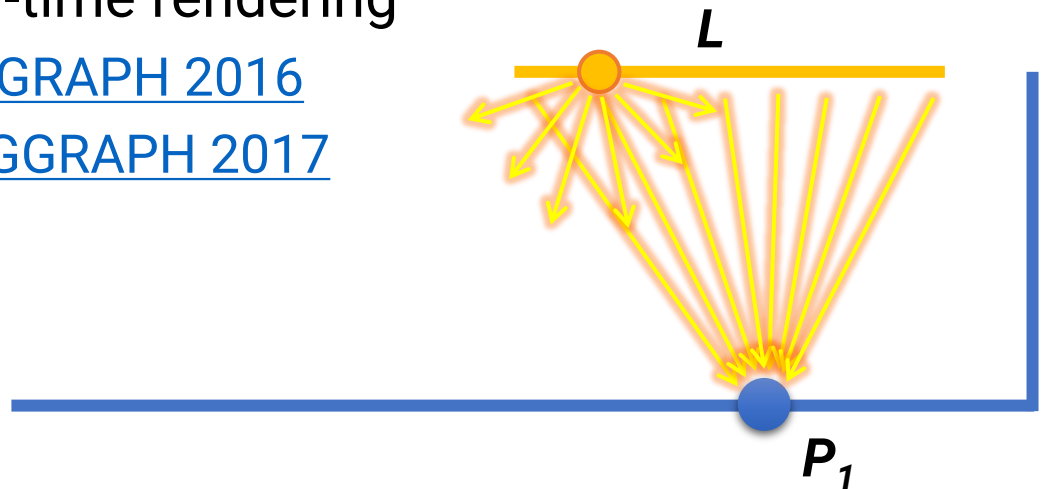


A scene illuminated by an area light



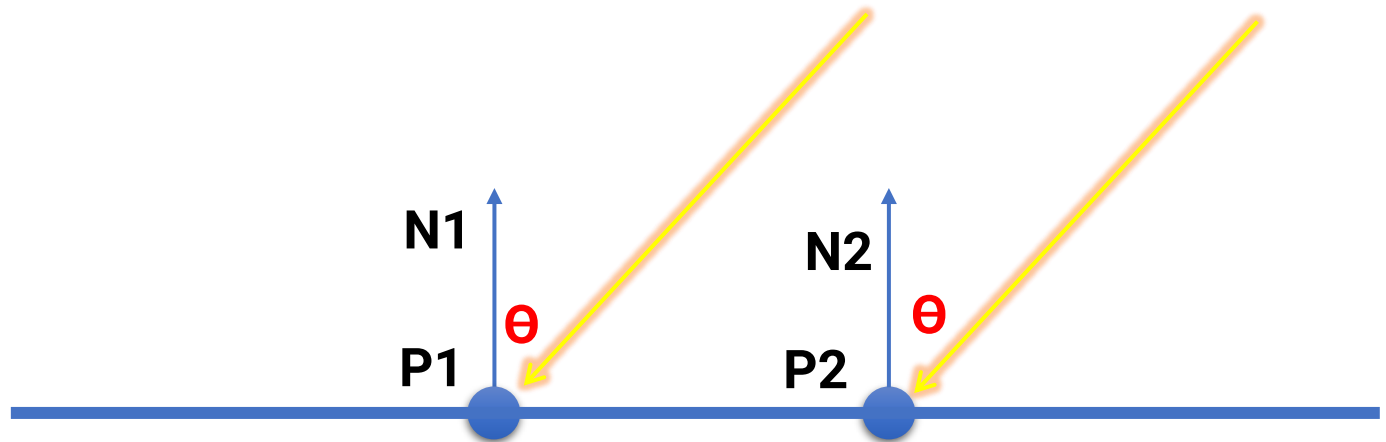
Area Light (cont.)

- Defined by one or more **shapes** that emit light from their surface, with some directional distribution of energy at each point on the surface
- Require **integration** of lighting contribution across the light surface
 - In offline rendering, usually estimated by sampling
 - Expensive for real-time rendering
 - [Heitz et al., SIGGRAPH 2016](#)
 - [Dupuy et al., SIGGRAPH 2017](#)



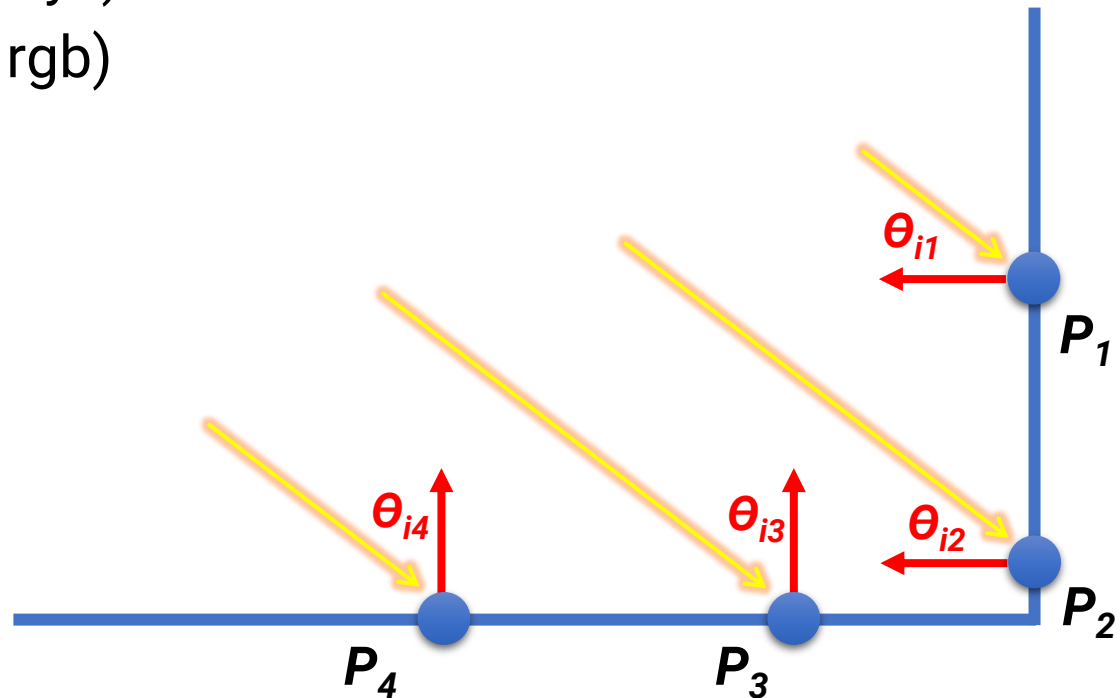
Distant Light

- The distance between a light and a surface is long enough compared to the scene scale and **can be ignored**
 - **Lighting direction** is **fixed**
 - **No lighting attenuation**
- **Directional light (sun)** is the most common distant light



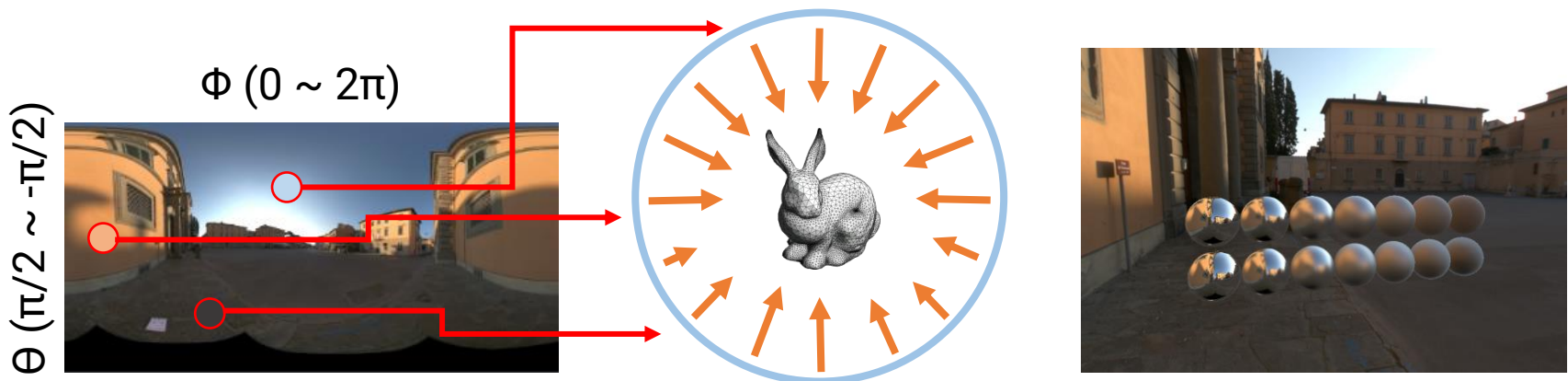
Directional Light

- Describes an emitter that deposits illumination from the **same direction** at every point in space
- Described by
 - Light direction (\mathbf{D} , xyz)
 - Light radiance (\mathbf{L} , rgb)



Environment Light

- Use a **texture** (cube map or longitude-latitude image) to represent a **spherical energy distribution**
 - Each texel maps to a spherical direction, considered as a directional light
 - The whole map illuminates the scene from a virtual sphere at an infinite distance
- Also called **image-based lighting (IBL)**

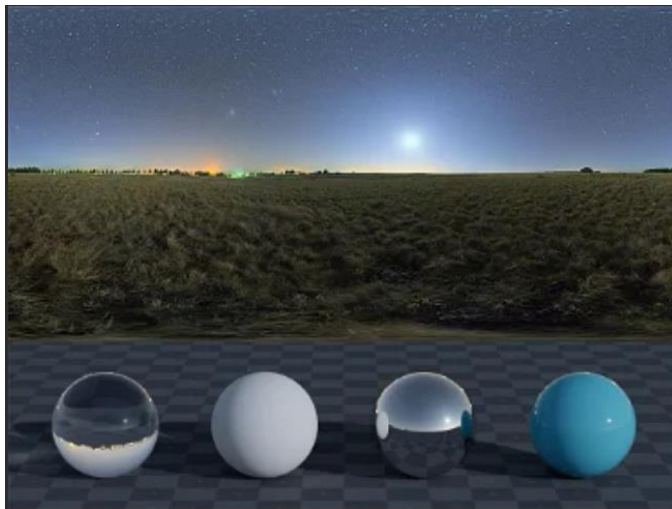


Environment Light (cont.)

- Widely used in digital visual effects and film production

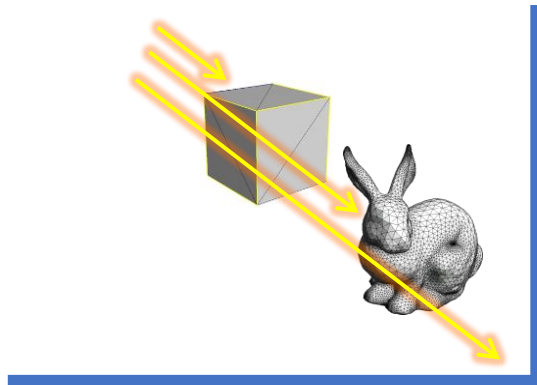


Environment Light (cont.)

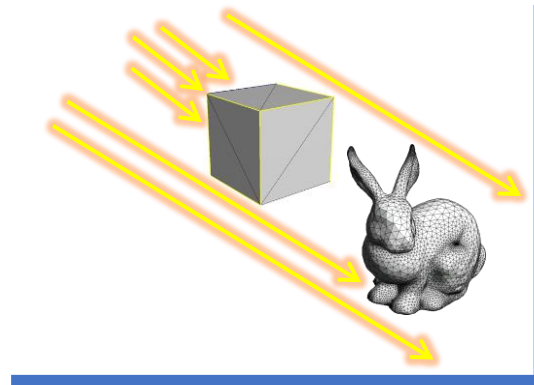


Local, Direct, and Global Illumination

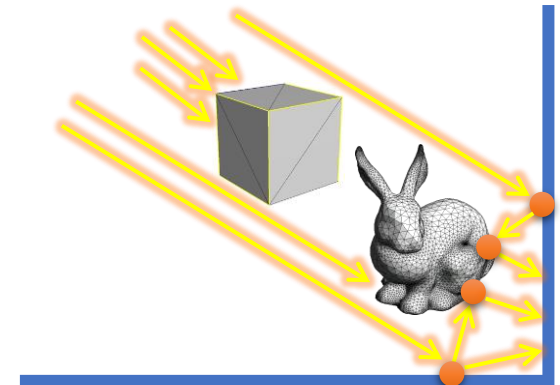
- Direct illumination considers only the **direct** contribution of lights
- Local illumination can be considered as direct lighting **without occlusion** (all lights are fully visible, no shadows)
- Global illumination includes **multi-bounce** illumination reflected from other surfaces (need **recursive** computation!)



local illumination



direct illumination



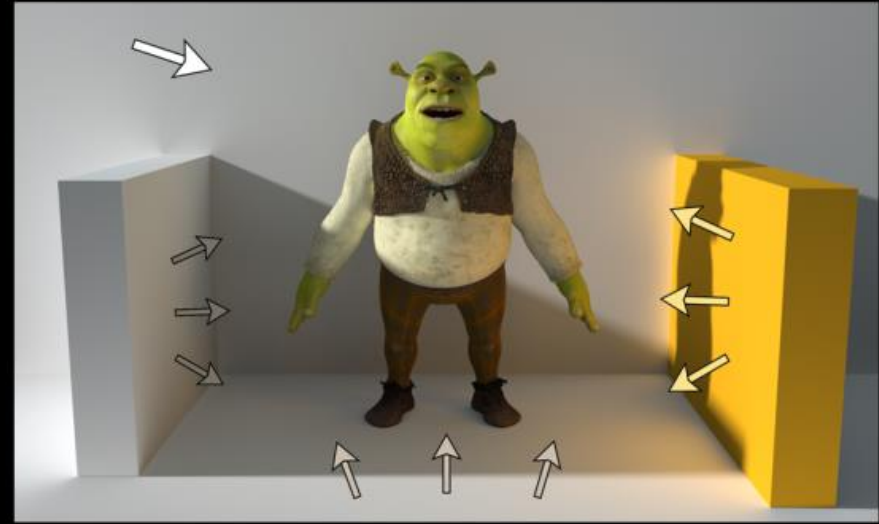
global illumination

Local, Direct, and Global Illumination (cont.)

Direct Lighting Only



Direct + Indirect Lighting

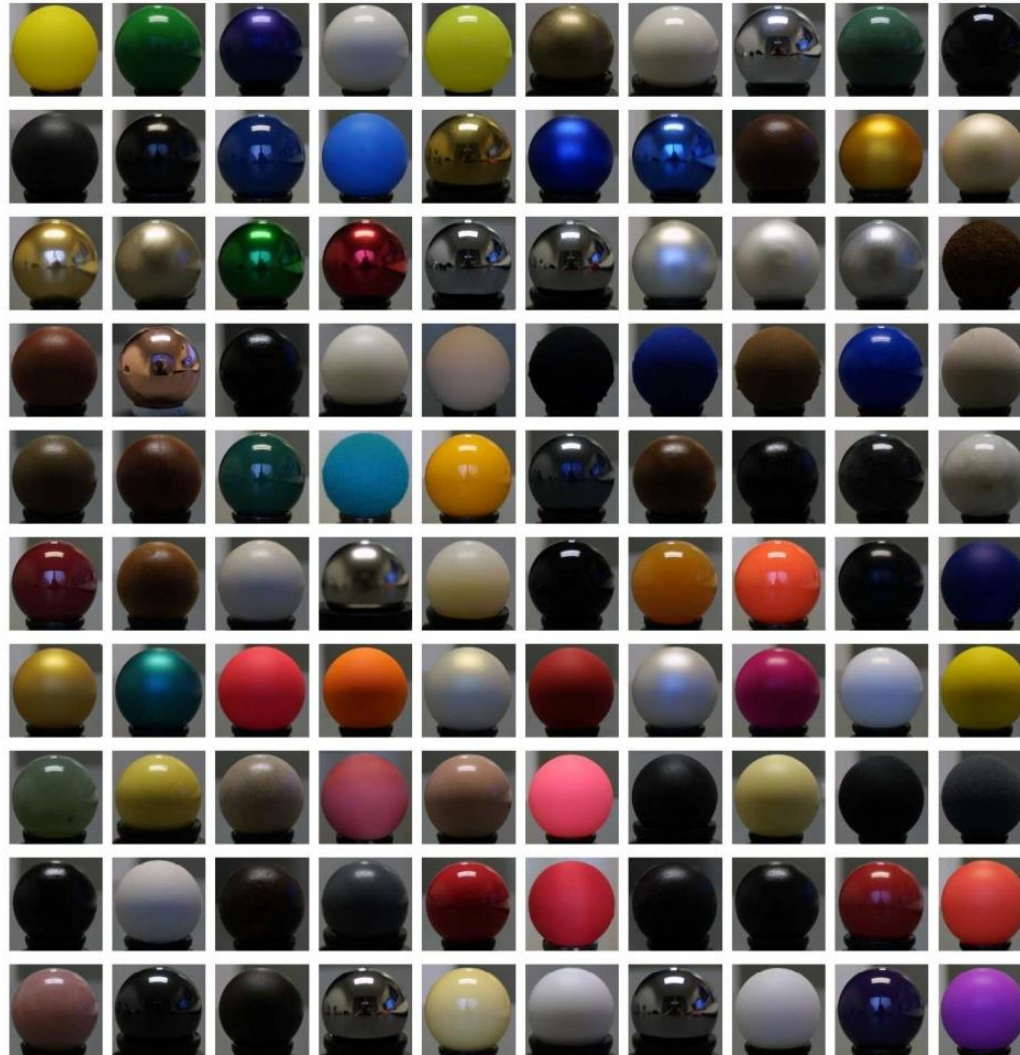


Comparison of direct and global illumination

Outline

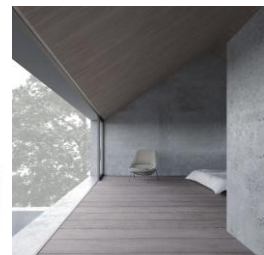
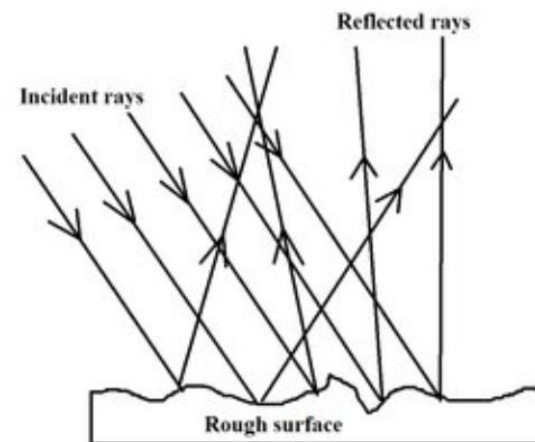
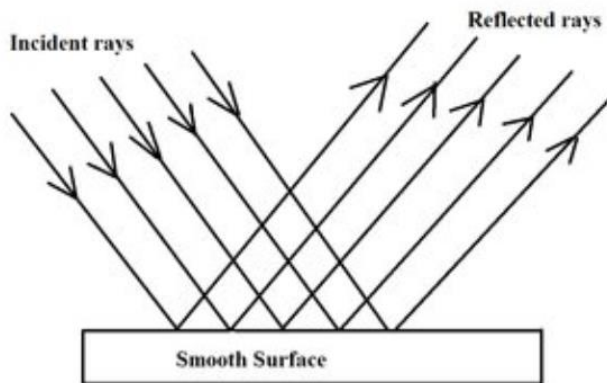
- Overview
- Lights
- **Materials**
- Material file format
- OpenGL implementation

Materials



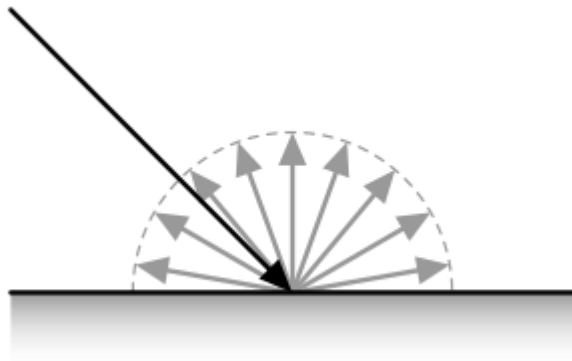
Materials (cont.)

- Highly related to surface types
- The **smoother** a surface, the more reflected light is concentrated in the direction a **perfect mirror** would reflect the light

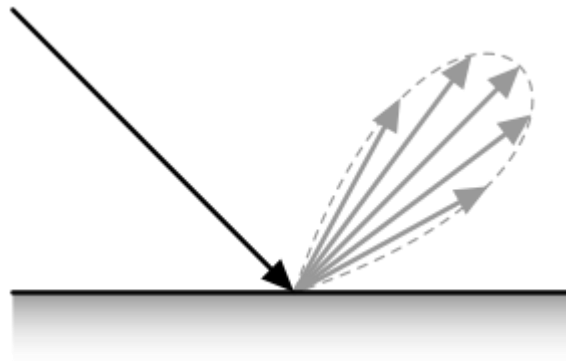


Materials (cont.)

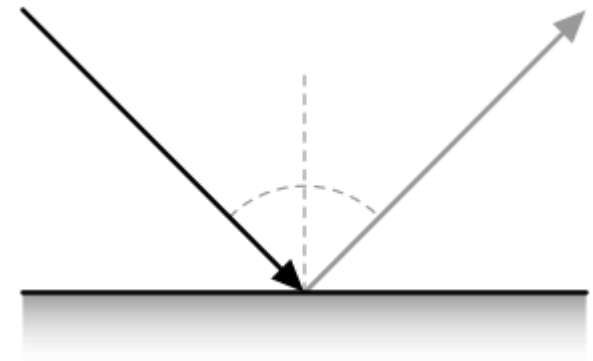
- Highly related to surface types
- The **smoother** a surface, the more reflected light is concentrated in the direction a **perfect mirror** would reflect the light



diffuse



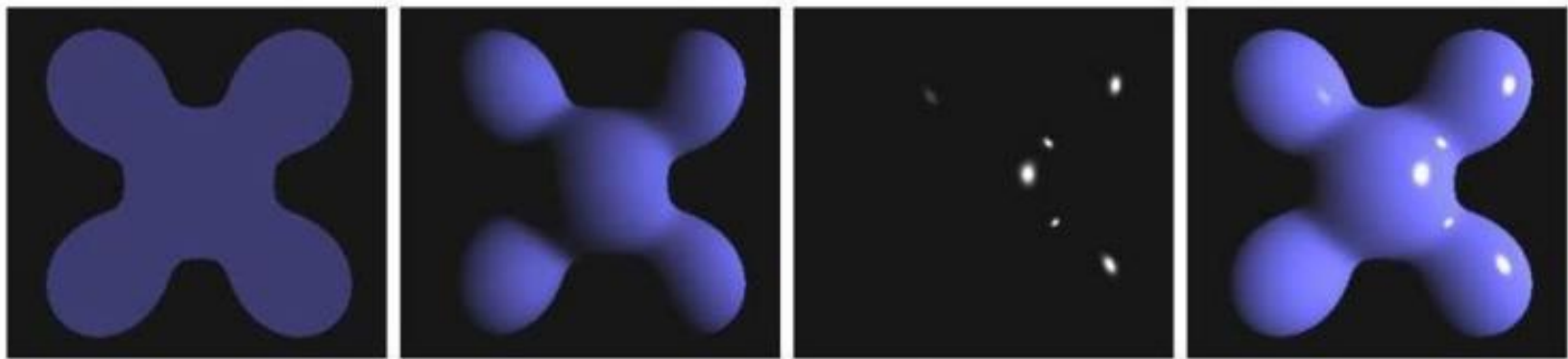
glossy



specular

Phong Lighting Model

- **Diffuse reflection**
 - Light goes everywhere; colored by object color
- **Specular reflection**
 - Happens only near mirror configuration; usually white
- **Ambient reflection**
 - Constant accounted for global illumination (cheap hack)



ambient

diffuse

specular

Ambient Shading

- Add constant color to account for disregarded illumination and fill black shadows



Flat Ambient



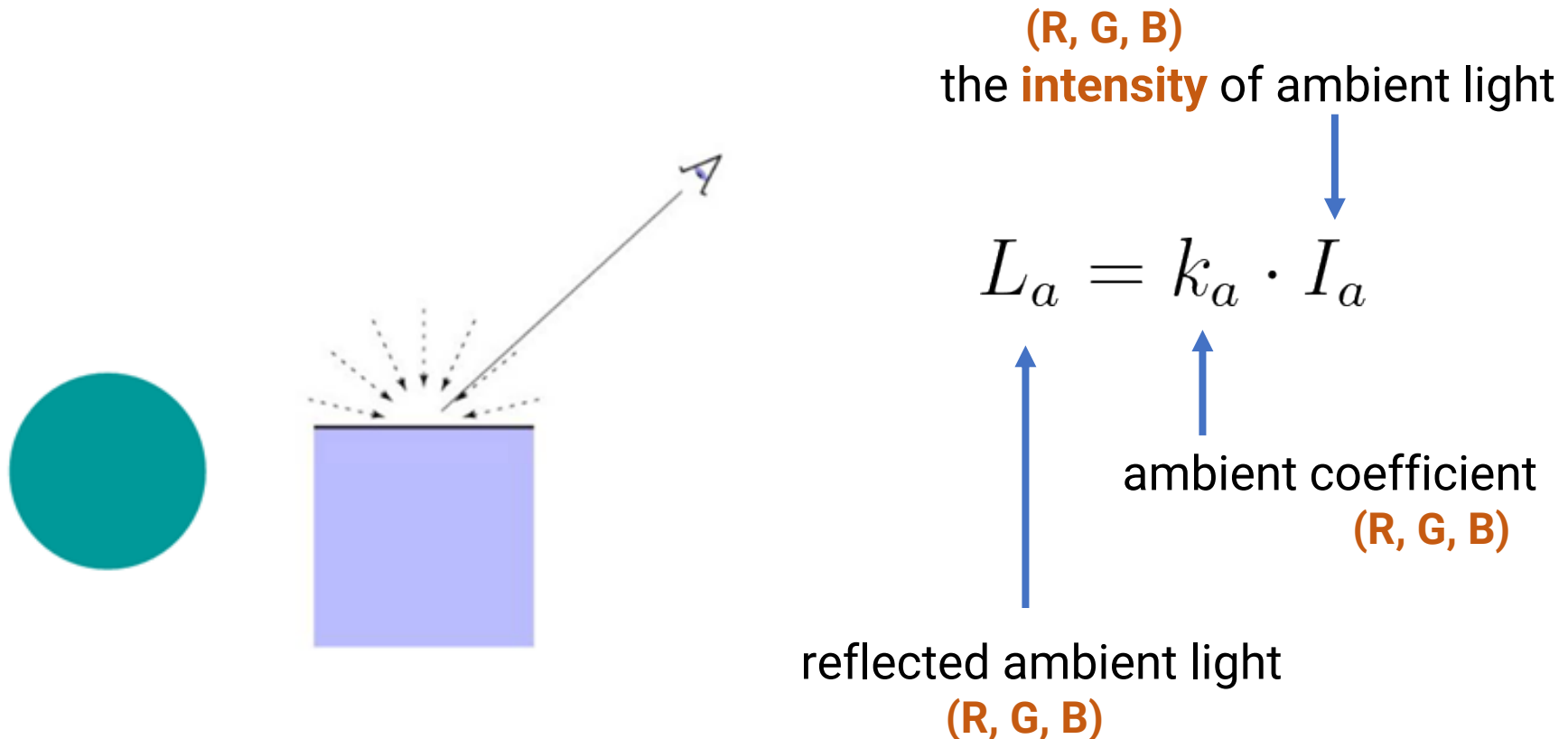
No Ambient



True Ambient

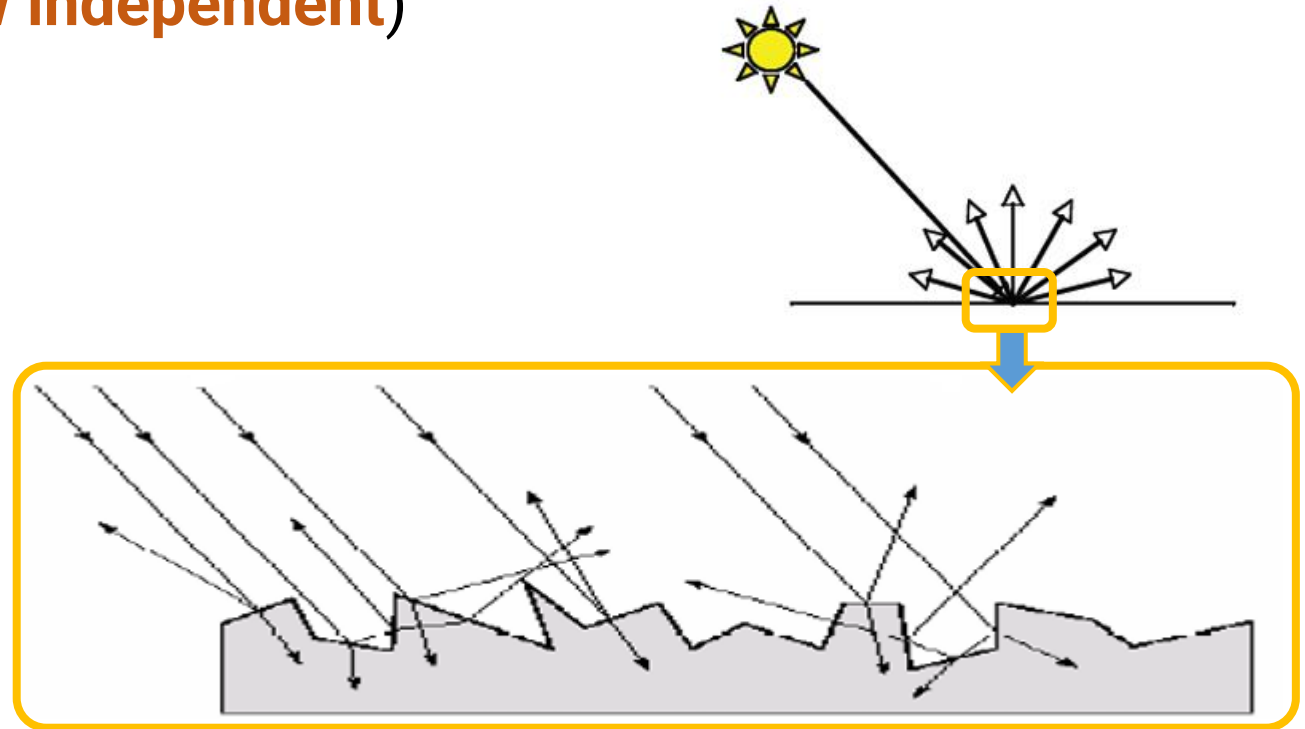
Ambient Shading (cont.)

- Add constant color to account for disregarded illumination and fill black shadows



Diffuse Shading

- Assume light reflects **equally in all directions**
 - The surface is rough with lots of tiny microfacets
- Therefore, the surface looks the same color from all views (**view independent**)



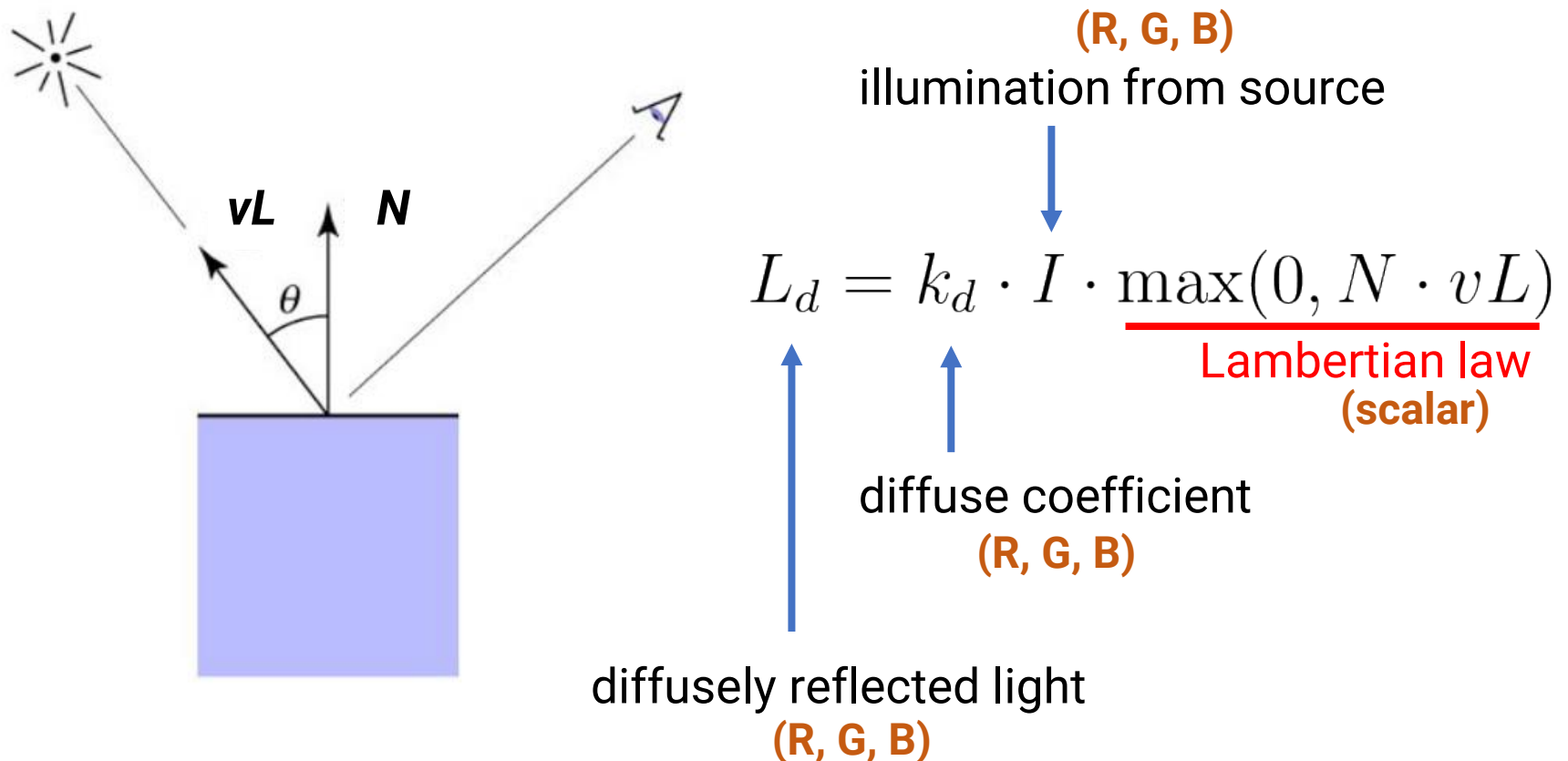
Diffuse Shading (cont.)

- Assume light reflects **equally in all directions**
 - The surface is rough with lots of tiny microfacets
- Therefore, the surface looks the same color from all views (**view independent**)

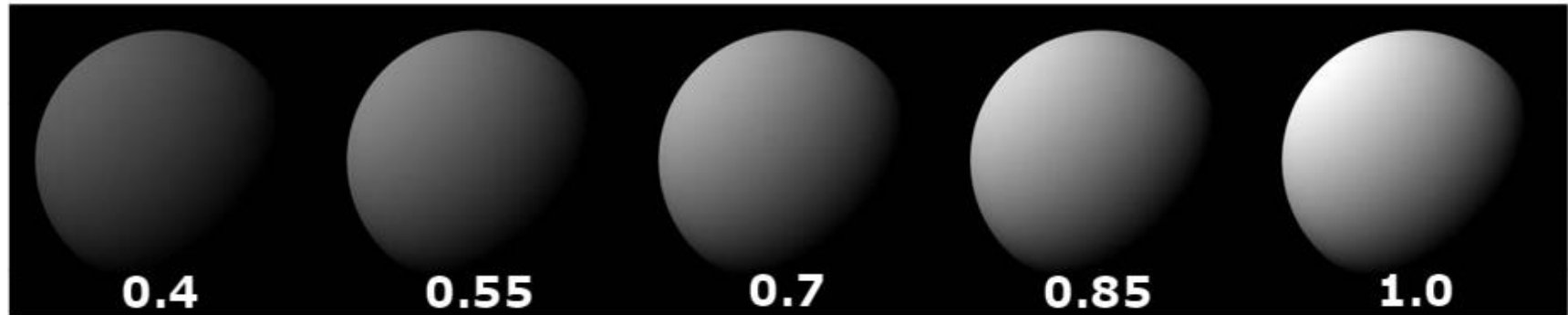


Diffuse Shading (cont.)

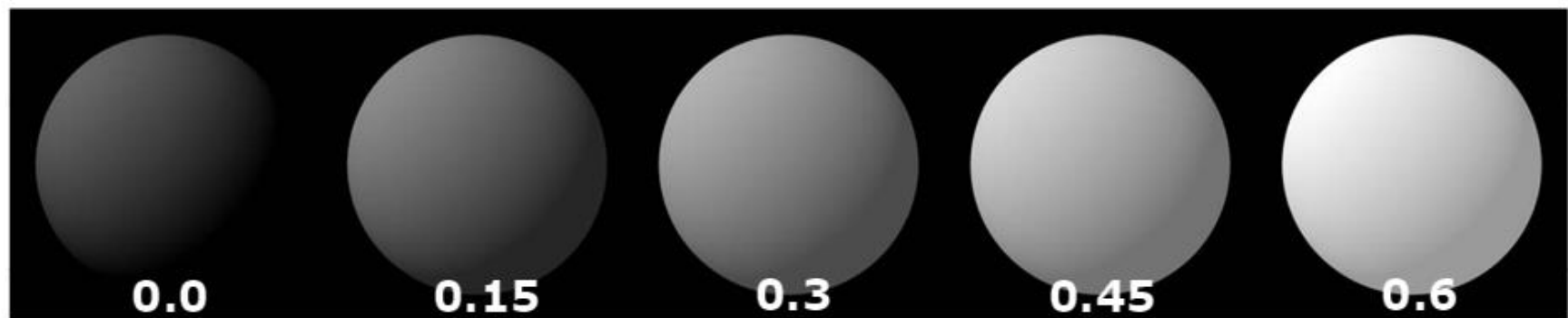
- Applies to diffuse or matte surface



Diffuse Shading (cont.)



diffuse-reflection model with different k_d



ambient and diffuse-reflection model with different k_a

$$I_a = 1.0 \quad k_d = 0.4$$

Diffuse Shading (cont.)

- For color objects, apply the formula for each color channel separately
- Light can also be non-white

Example:

white light: (0.9, 0.9, 0.9)

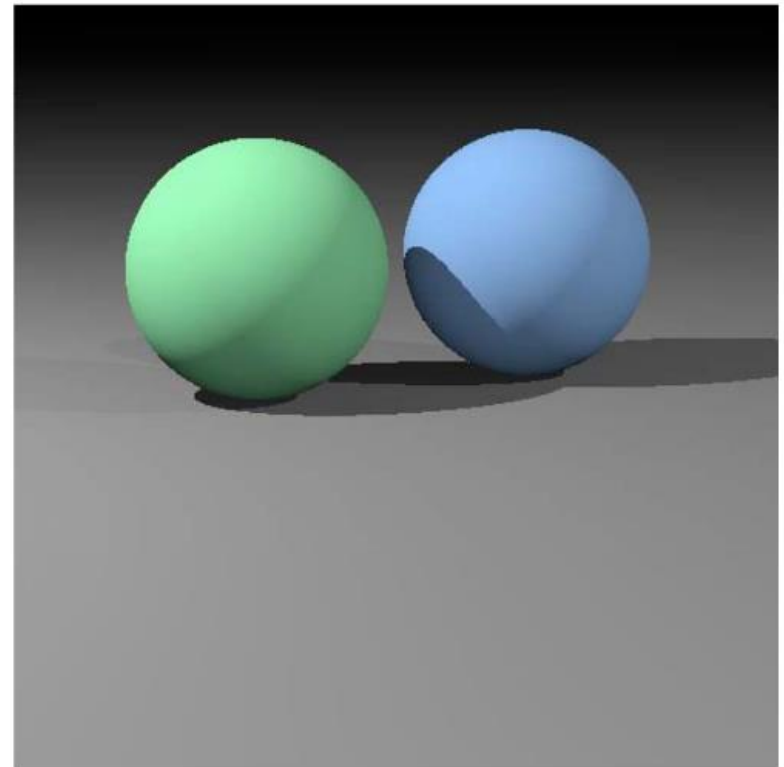
yellow light: (0.8, 0.8, 0.2)

$$L_d = k_d \cdot I \cdot \max(0, N \cdot vL)$$

Example:

green ball: (0.2, 0.7, 0.2)

blue ball: (0.2, 0.2, 0.7)



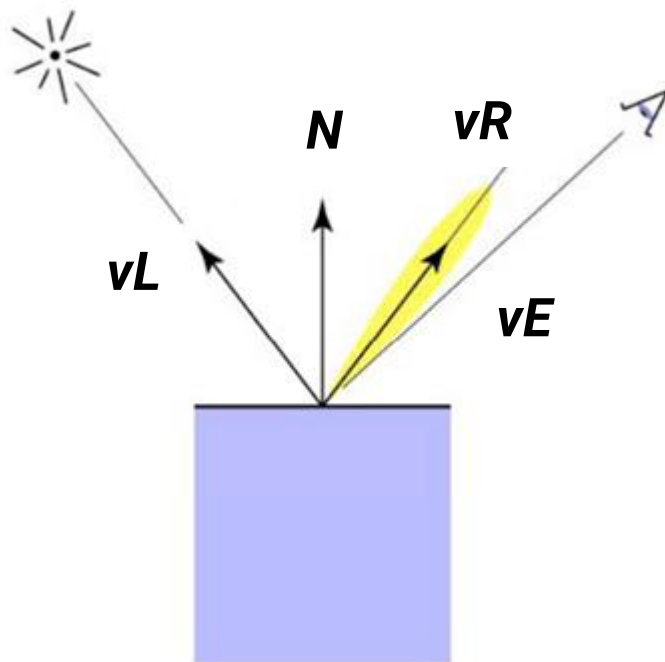
Specular Shading

- Some surfaces have highlights, mirror-like reflection
- **View direction dependent**
- Especially obvious for smooth shiny surfaces



Specular Shading (cont.)

- Phong specular model [1975]



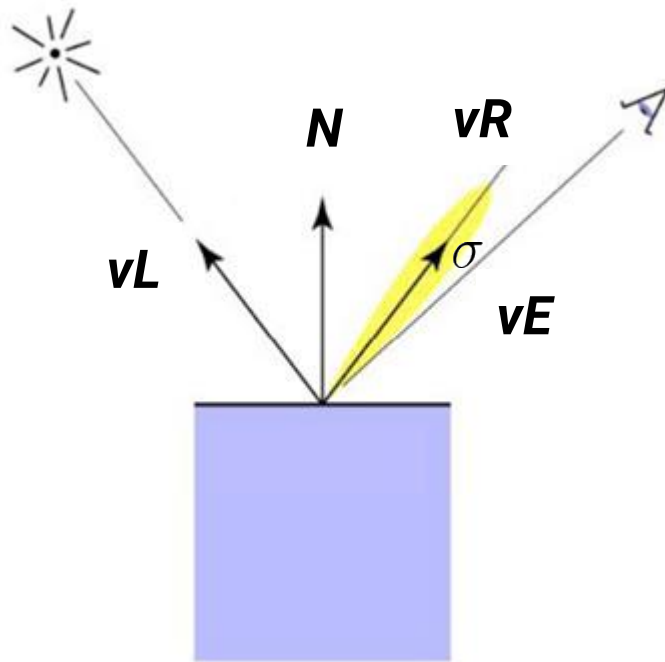
$$\begin{aligned} vR &= vL + 2((N \cdot vL)N - vL) \\ &\quad \uparrow \\ &= 2(N \cdot vL)N - vL \end{aligned}$$

perfectly reflected direction

(you can find the proof [here](#))

Specular Shading (cont.)

- **Phong specular model [1975]**
 - Fall off gradually from the perfect reflection direction



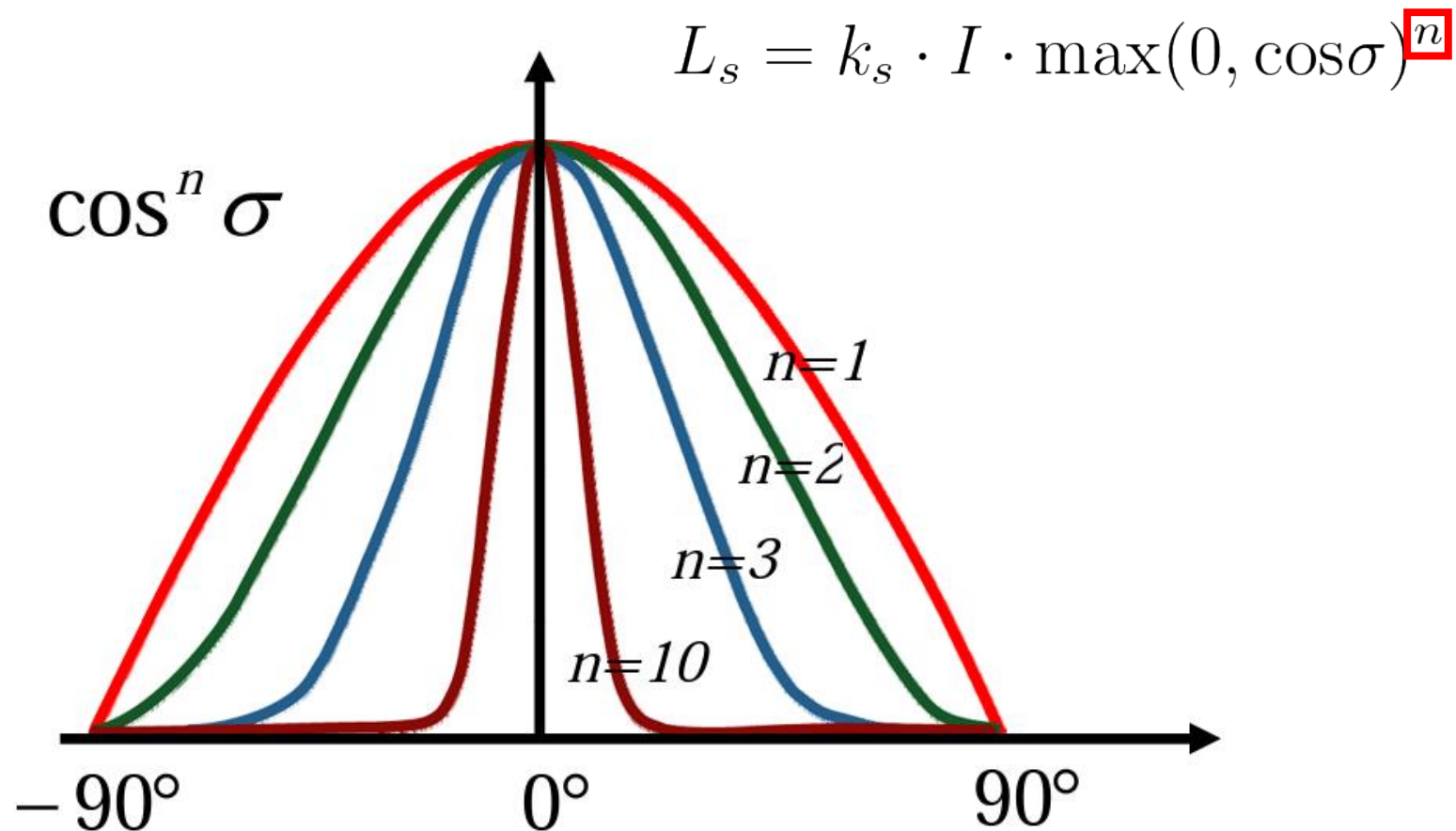
$$\begin{aligned} vR &= vL + 2((N \cdot vL)N - vL) \\ &= 2(N \cdot vL)N - vL \end{aligned}$$

$$\begin{aligned} L_s &= k_s \cdot I \cdot \max(0, \cos \sigma)^n \\ &= k_s \cdot I \cdot \max(0, vE \cdot vR)^n \end{aligned}$$

(R, G, B) (scalar)
 specular exponent
 specular coefficient (R, G, B)
 specularly reflected light (R, G, B)

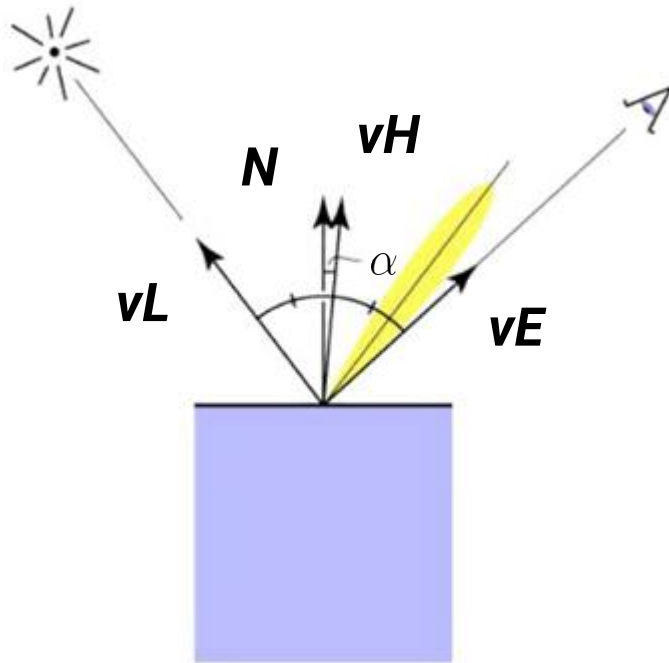
Specular Shading (cont.)

- Increase n narrows the lobe



Phong specular Variant: Blinn-Phong

- Rather than computing reflection directly, just compare to normal bisection property
- One can prove $\cos^n(\sigma) = \cos^{4n}(\alpha)$



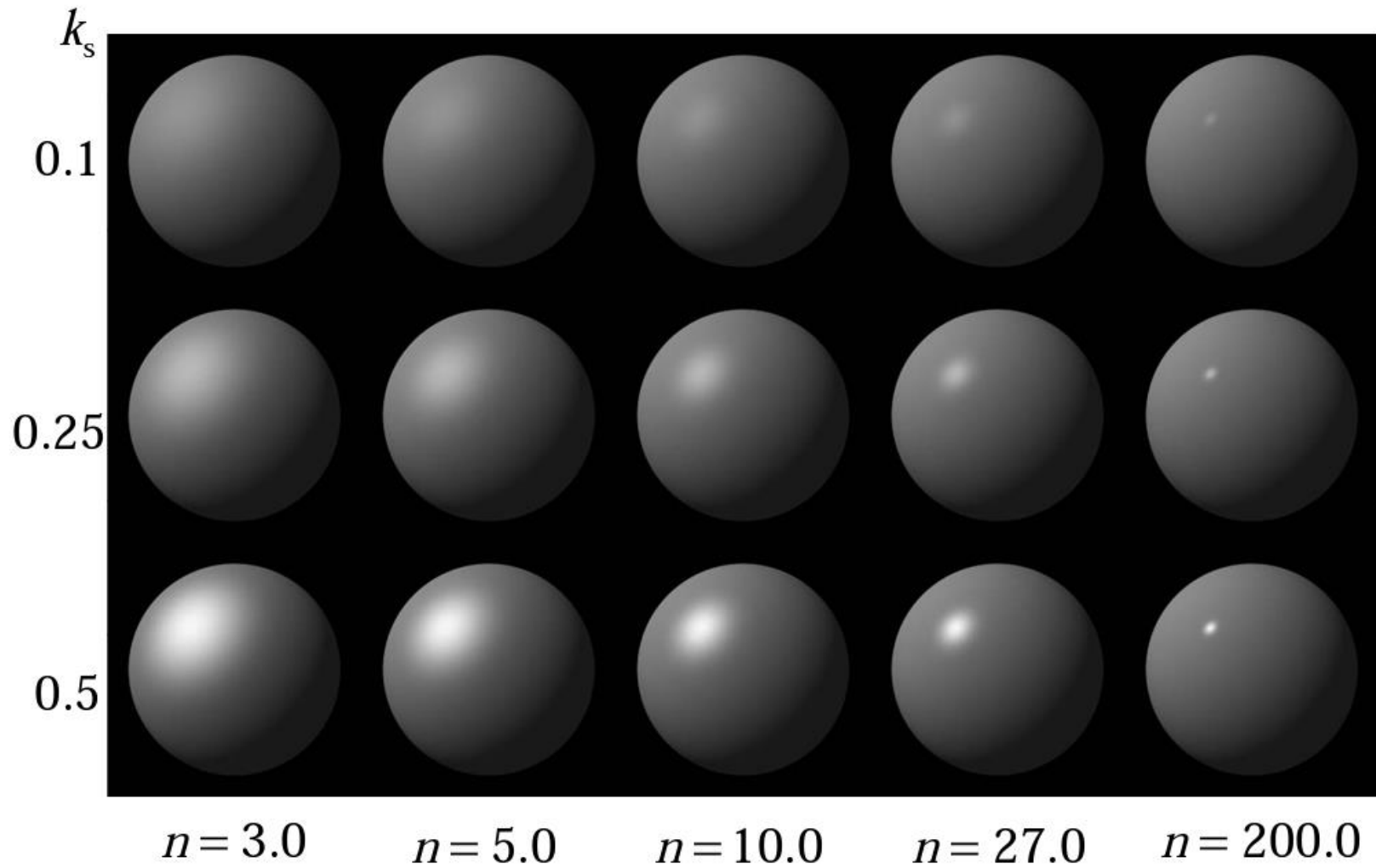
half vector

$$vH = \text{bisector}(vL, vE)$$

$$= \frac{(vL + vE)}{\|vL + vE\|}$$

$$\begin{aligned} L_s &= k_s \cdot I \cdot \max(0, \cos\alpha)^n \\ &= k_s \cdot I \cdot \max(0, N \cdot vH)^n \end{aligned}$$

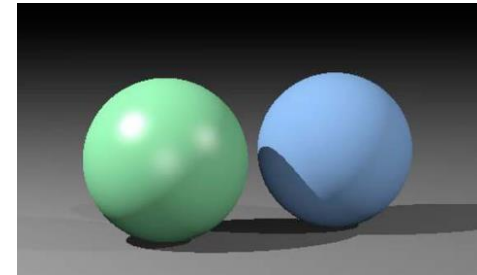
Specular Shading (cont.)



Complete Phong Lighting Model

- Compute the contribution from a light to a point by including **ambient**, **diffuse**, and **specular** components

$$\begin{aligned}
 L &= L_a + L_d + L_s \\
 &= k_a \cdot I_a + I(k_d \cdot \max(0, N \cdot vL) + k_s \cdot \max(0, N \cdot vH)^n)
 \end{aligned}$$



- If there are **s** lights, just sum over all the lights because the lighting is **linear**

$$L = k_a \cdot I_a + \sum_i^s (I_i(k_d \cdot \max(0, N \cdot vL_i) + k_s \cdot \max(0, N \cdot vH_i)^n))$$

Some Results with Phong Lighting Model

