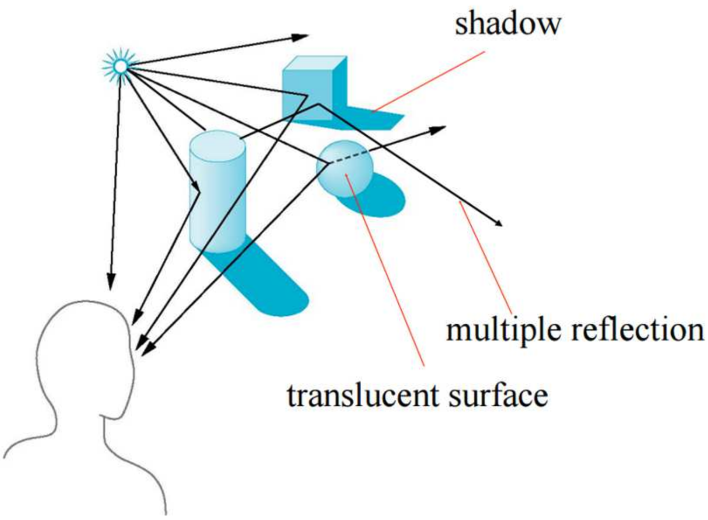
Lighting and Shading

Lighting

- *Lighting* refers to the handling of interactions between the light sources and the objects in the 3D scene.

- When light from a light source strikes a surface, some gets absorbed, some is reflected or scattered
- The are two classes of lighting models:
  1. Local illumination: consider only the direct lightings. The color of the surface depends on the reflectance properties of the surface and the direct lightings.
  2. Global illumination: in real world, objects received indirect lighting reflected from other objects and the environment. The global illumination model consider indirect lightings reflected from other objects in the scene. Global illumination model is complex and compute intensive.

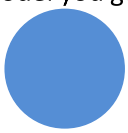
Global Effect



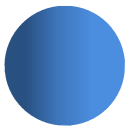
3

Why we need Shading

- Shading refers to the process of altering the color of an object in the 3D scene, based on from lights to create a photorealistic effect
- When you build a 3D model you get something like this



- But, what you want it

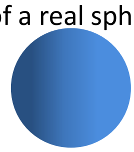


- The more realistically lit sphere has gradations in its color that give us a sense of its three dimensionality

4

Shading

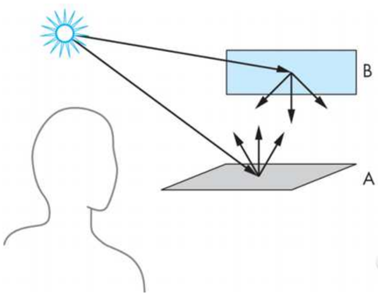
- Why does the image of a real sphere look like



- Light-material interactions cause each point to have a different color or shade
- Need to consider
  - Light sources
  - Material properties
  - Location of viewer
  - Surface orientation (normal)
- We are going to develop a local lighting model by which we can shade a point independently of the other surfaces in the scene

Light and Matter

- The most general approach is based on physics - using principles such as conservation of energy
- A surface either emits light (e.g., light bulb) or reflects light for other illumination sources, or both
- Light interaction with materials is recursive the rendering equation is an integral equation describing the limit of this recursive process



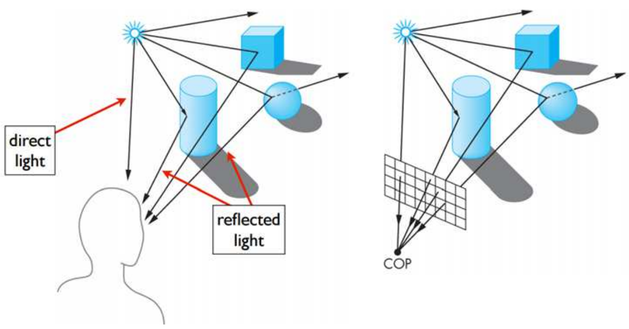
Fast Local Shading Models

- The rendering equation can't be solved analytically
- Numerical methods aren't fast enough for real-time for our fast graphics rendering pipeline, instead local model: **Phong model**
  - shading based on local light-material interactions
- Some approximations to the rendering equation include radiosity and ray tracing, but they are still not as fast as the local model in the pipeline architecture

7

Local Shading Model

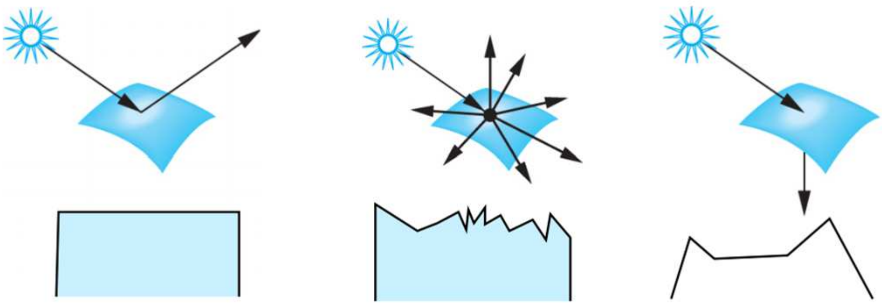
- Direct light is the color of the light source
- Reflected light is the color of the light reflected from the object surface for rendering
- To determines the colors of pixels in the frame buffer only need to consider the rays that leave the source and reach the viewer's eye



8

Light-Material Interaction

- At a surface, light is absorbed, reflected, or transmitted
  - Specular: shiny, smooth surface. light scattered in narrow range close to angle of reflection e.g., mirror is perfectly specular
  - Diffuse: matte, rough surface. light scattered in all directions
  - Translucent: allows some light to pass through object.

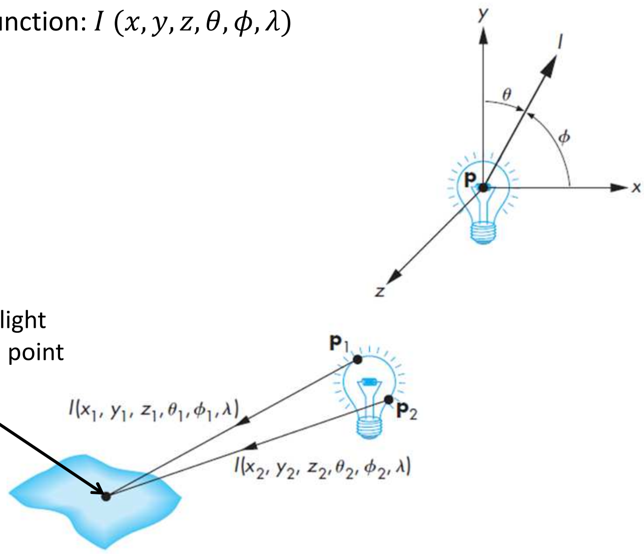


9

General light source

- Illumination function: $I\ (x, y, z, \theta, \phi, \lambda)$

integrate over the light surface to shade the point



10

Idealized light sources

- We describe a source through a three-component intensity, or **luminance**, function

$$I = \begin{bmatrix} I_r \\ I_g \\ I_b \end{bmatrix}$$

- Ambient light
- Point light
- Spotlight
- distant (directional) light

11

Ambient light

- Achieves a uniform light level
- No black shadows
- Ambient light intensity at each point in the scene

$$I_a = \begin{bmatrix} I_{ar} \\ I_{ag} \\ I_{ab} \end{bmatrix}$$

- Although every point in our scene receives the same illumination from I_a, each surface can reflect this light differently.

12

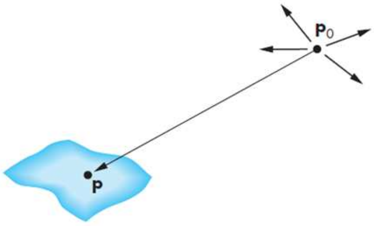
Point Sources

- An ideal **point source** emits light equally in all directions

$$I(p_0) = \begin{bmatrix} I_r(p_0) \\ I_g(p_0) \\ I_b(p_0) \end{bmatrix}$$

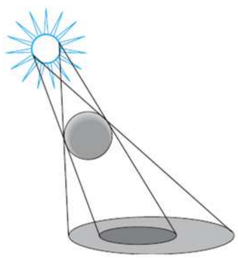
- Illumination intensity at P:

$$i(p, p_0) = \frac{1}{|p - p_0|^2} I(p_0)$$



Point Sources

- Most real-world scenes have large light sources



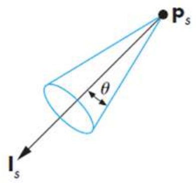
- Drop off intensity more slowly

$$i(p, p_0) = \frac{1}{a + bd + cd^2} I(p_0)$$

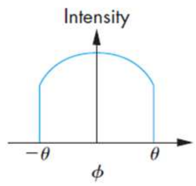
- Where d is the distance between p and p_0, a, b, and c chosen to soften the lighting

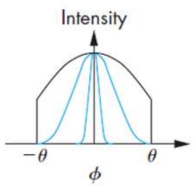
Spotlight

- Spotlights are characterized by a narrow range of angles through which light is emitted



- The intensity is a function of the angle ϕ between the direction of the source and a vector **s** to a point on the surface





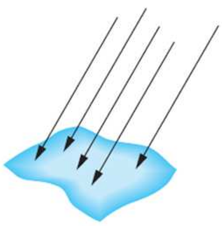
$cos^e\phi \; i(p, p_s)$

15

Distant Light Sources

- Most shading calculations require direction from the surface point to the light source position if the light source is very far, the direction vectors don't change e.g., sun

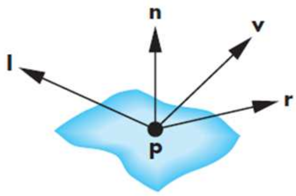
- Characterized by direction rather than position



16

The Phong Lighting Model

- Introduced by Phong and later modified by Blinn
- Efficient and to be a close-enough approximation to physical reality



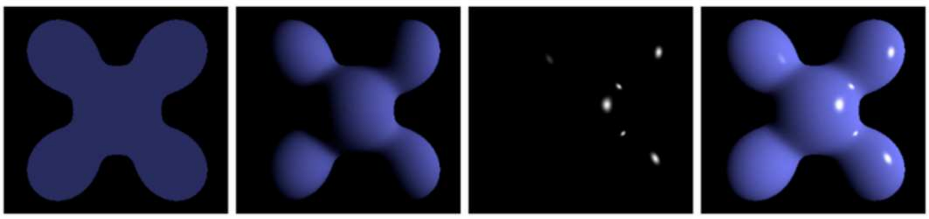
- The vector **n** is the normal at **p**
- The vector **v** is in the direction from **p** to the viewer or COP.
- The vector **l** is in the direction of a line from **p** to an arbitrary point on the source
- The vector **r** is in the direction that a perfectly reflected ray from **l** would take.

17

The Phong Lighting Model

- The Phong reflection model combines the Ambient and Lambertian reflections with a specular reflection to capture highlights
- The highlight is a reflection of the light and it is the color of the light.

$$I = I_a + I_d + I_s = L_a R_a + L_d R_d + L_s R_s$$



18

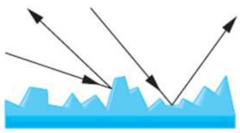
Ambient

- The intensity of ambient light I_a is the same at every point on the surface
- Refection coefficient $R_a = k_a$, because only a positive fraction of the light is reflected, we must have $0 \leq k_a \leq 1$
- And thus $I_a = k_a L_a$
- For example, a sphere appears yellow under white ambient light if its blue ambient coefficient is small and its red and green coefficients are large.

19

Diffuse

- A perfectly diffuse reflector scatters the light that it reflects equally in all directions. Hence, such a surface appears the same to all viewers.



- Lambert's law states that

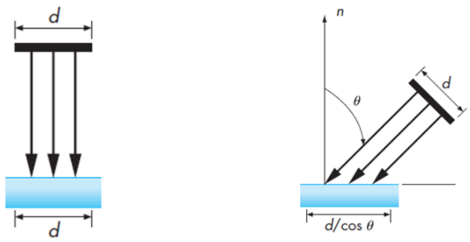
$$R_d \propto cos\theta$$

- Where θ is the angle between the normal at the point of interest **n** and the direction of the light source **l**

20

Diffuse

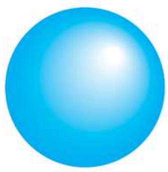
$$cos\theta = I.n$$
$$I_d = k_d(I.n)L_d$$



21

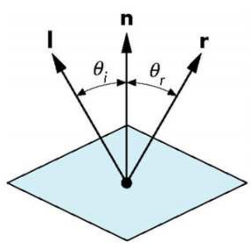
Specular

- Specular surface is smooth



- Angle of incidence = angle of reflection

$$\theta_i = \theta_r$$



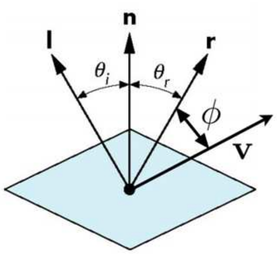
22

Specular

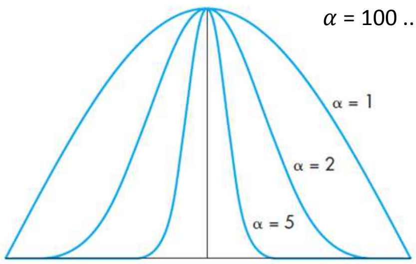
- The amount of light that the viewer sees depends on the angle ϕ between **r**, the direction of a perfect reflector, and **v**, the direction of the viewer

$$I_s = k_s L_s cos^{\alpha}\phi$$

- The exponent α is a **shininess** coefficient.



α = 5 .. 10 Plastic
α = 100 .. 200 Metal



23

Specular

- If we have normalized **r** and **n** to unit length

$$I_s = k_s L_s \max((r.v)^{\alpha}, 0)$$

- The **Phong model**, including the distance term, is written:

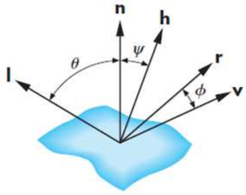
$$I = \frac{1}{a + bd + cd^2}(k_d L_d \max(0, I.n) + k_s L_s \max((r.v)^{\alpha}, 0)) + k_a L_a$$

24

The Modified Phong Model

- If we use the Phong model with specular reflections in our rendering, the dot product **r . v** should be recalculated at every point on the surface

$$h = \frac{I + v}{\|I + v\|}$$



- **h** does not depend on **n**

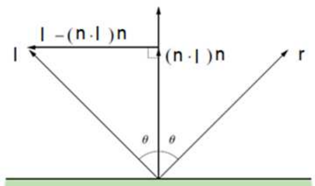
$$I = \frac{1}{a + bd + cd^2}\left(k_d L_d \max(0, I.n) + k_s L_s \max((n.h)^{\alpha}, 0)\right) + k_a L_a$$

Angle of Reflection

- Normal is determined by local orientation
- Use **n** and **l** to compute **r**
- Vectors **n** and **l** have been normalized to unit length
- First, calculate the component of **l** that is perpendicular to the normal direction

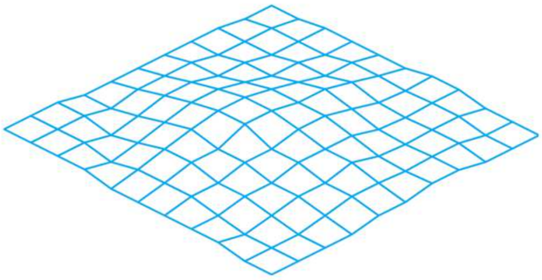
$$perp_n I = I - (n.I)n$$

- The vector **r** lies at twice the distance from **L** as does its projection onto the normal vector **n**
- r = $I - 2perp_n I$
- r = $I - 2[I - (n.I)n]$
- r = $2(n - I)n - I$



Polygonal Shading

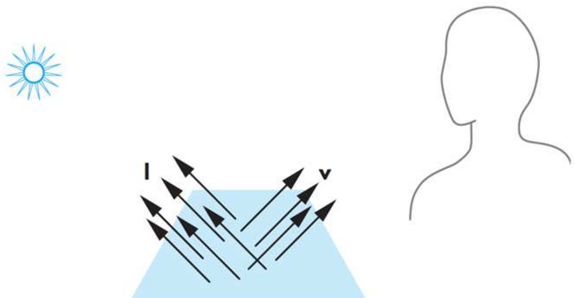
- Smooth surfaces are often approximated by polygons
- Shading approaches:
  - Flat Shading
  - Smooth (Gouraud)
  - Phong



27

Flat Shading

- The three vectors—**l**, **n**, and **v**—can vary as we move from point to point on a surface.
- For a flat polygon, however, **n** is constant. If we assume a distant viewer, **v** is constant over the polygon. Finally, if the light source is distant, **l** is constant.

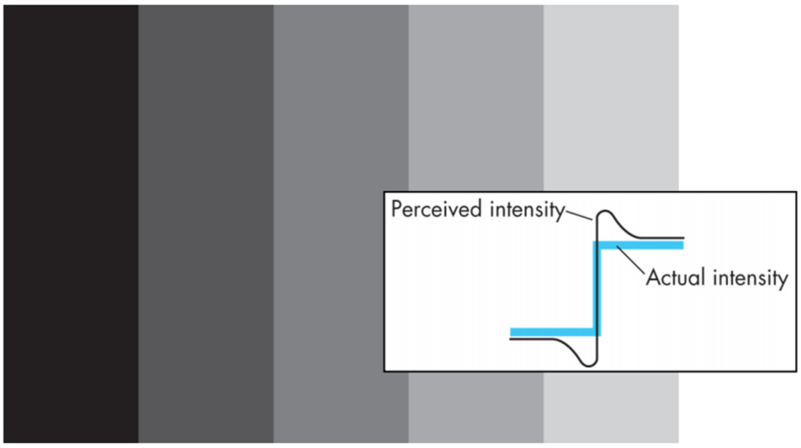


Distant source and viewer

28

Mach Bands

- Flat shading doesn't usually look too good
- The **lateral inhibition** effect makes flat shading seem even worse

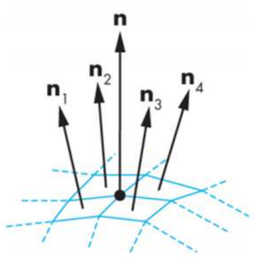


29

Smooth (Gouraud) Shading

- Do the shading calculation once per vertex
- We assign the vertex normals based on the surrounding polygon normals

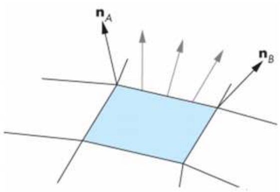
$$n = \frac{n_1 + n_2 + n_3 + n_4}{\|n_1 + n_2 + n_3 + n_4\|}$$



30

Phong Shading

- Do the shading calculation once per fragment
- Phong shading requires normals to be interpolated across each polygon
- This can now be done in the pipeline in the fragment shader



31

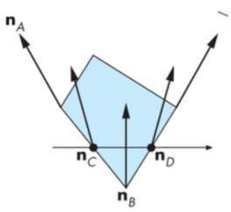
Phong Shading

- We can use the interpolated normals at vertices *A* and *B* to interpolate normals along the edge between them:

$$n_C(\alpha) = (1 - \alpha)n_A + \alpha n_B$$

- We can do a similar interpolation on all the edges. The normal at any interior point can be obtained from points on the edges by

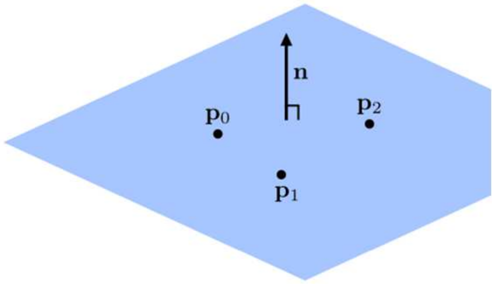
$$n(\alpha, \beta) = (1 - \beta)n_C + \beta n_D$$



32

Compute Normals

$$v = (p_1 - p_0) \times (p_2 - p_0)$$
$$n = \frac{v}{\|v\|}$$



33