CSE-170 Computer Graphics

Lecture 9

Illumination

Dr. Renato Farias
rfarias2@ucmerced.edu

Illumination and Shading

- Surfaces are shaded following illumination models
  - for each point in a surface, its final color has to be computed according to the illumination model before it can be painted in the image buffer during rasterization

- Lights and materials must be declared first

Illumination and Shading

- Illumination models
  - local models: interaction between individual points in a surface and light sources
    - Gouraud (1971) and Phong (1975)
    - Very fast – but do not automatically account for refractions, reflections and shadows
  - global models: interchange of light between all surfaces
    - Ex: ray tracing
    - Very realistic, but slower

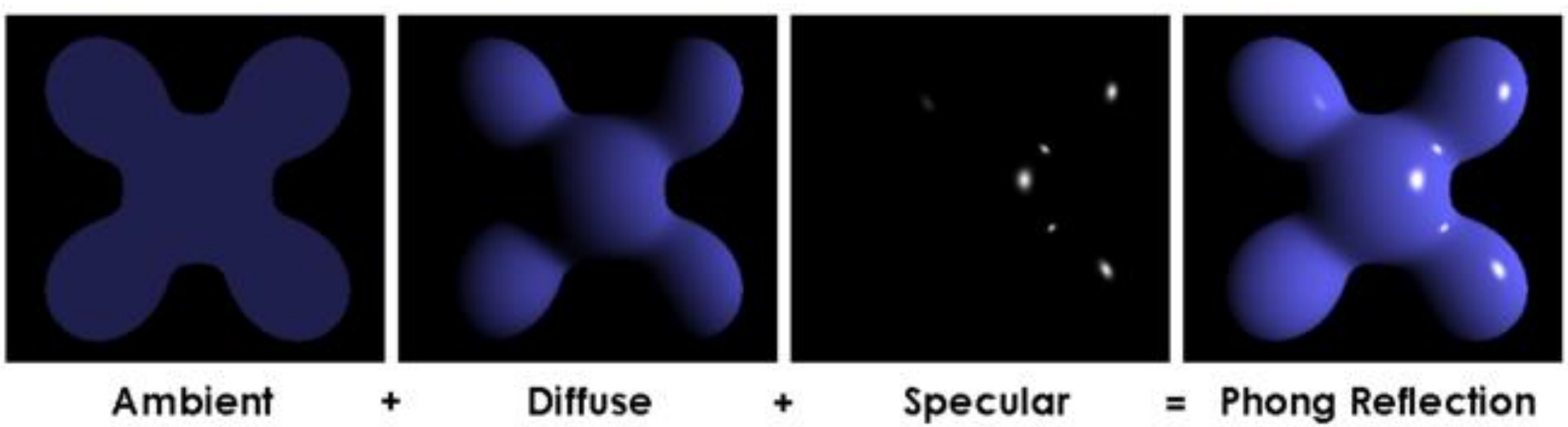
Illumination and Shading

- Illumination models
  - Set the color of a surface point according to light and surface properties
  - Local models
    - Phong lighting model (most used)
    - Cook-Torrance lighting model (more complex)
    - “physically inspired”, not “physically correct”
- Shading
  - Applies an illumination model to several pixels
  - Colors and brightness vary smoothly across a surface using interpolation methods

Phong Illumination Model

Phong Illumination

- Bui Tuong Phong's thesis 1973
- It is a simplification of the more general rendering equation
  - it is local (no 2nd order reflections)
  - reflection is divided into ambient, diffuse, and specular components



Phong Illumination Model

- Parameters:
  - Light intensities for ambient, diffuse, and specular reflection:

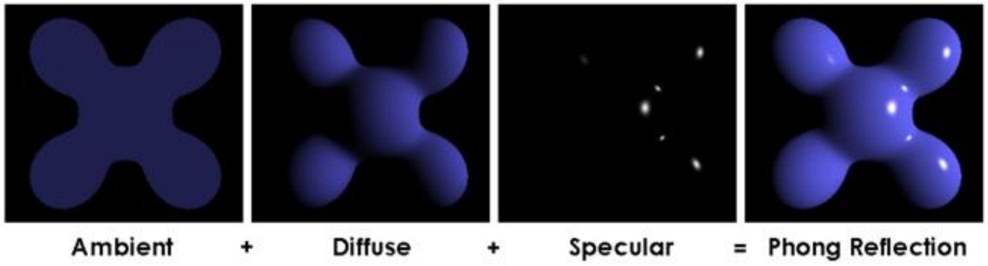
$$I_a, I_d, I_s$$ each in [0,1]

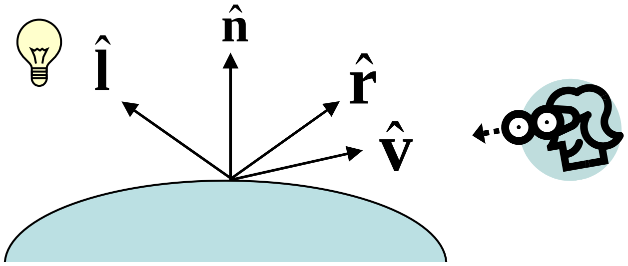
  - Material coefficients for each type of light intensity:

$$k_a, k_d, k_s$$ each in [0,1]

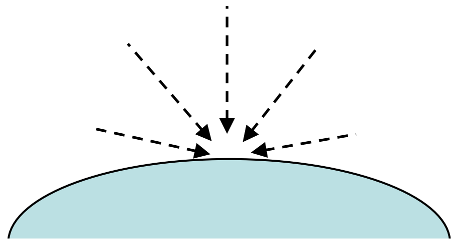
  - Scene parameters involving unit vectors: light direction, surface normal, reflected ray, viewer direction

$$\hat{\mathbf{l}}, \hat{\mathbf{n}}, \hat{\mathbf{r}}, \hat{\mathbf{v}}$$





Ambient

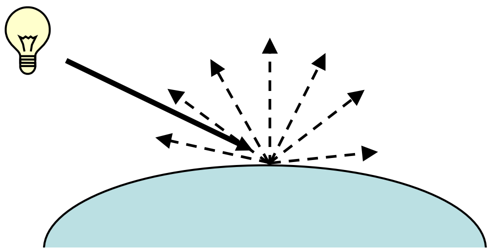


- Ambient light
  - non-directional source of light, coming from the environment, equally coming from all surfaces and directions
  - k_a is the ambient reflection coefficient, a material property
  - does not correspond to physical properties of real materials
  - provides a uniform illumination across the surfaces, similar to “self illumination”

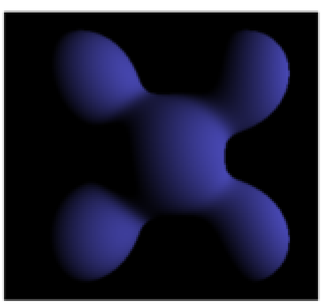
$$I = I_a k_a$$

$$k_a \in [0,1]$$

Diffuse



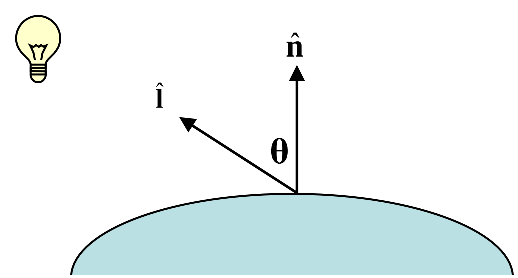
- Diffuse reflection
  - takes into account point light source
    - I_d is the point light source intensity
  - rays emanate uniformly in all directions
  - object's brightness varies according to the direction of the light source and the surface normal
  - such diffuse reflection is also known as Lambertian reflections
    - Lambertian surfaces appear equally bright from all viewing angles, they reflect light equally in all directions (ex: chalk)



$$I = I_d k_d \cos\theta = I_d k_d (\hat{\mathbf{l}} \cdot \hat{\mathbf{n}})$$

$$k_d \in [0,1]$$

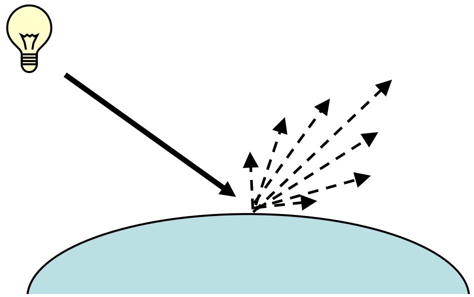
$$\theta \in [0,90]$$



(Reminder: $\mathbf{a} \cdot \mathbf{b} = \|\mathbf{a}\| \, \|\mathbf{b}\| \cos(\theta)$)

Specular

- Specular reflection
  - mirror-like reflections, for shiny surfaces
  - the highlight given by a bright light in an object
  - at the highlight the color has the color of the incident light

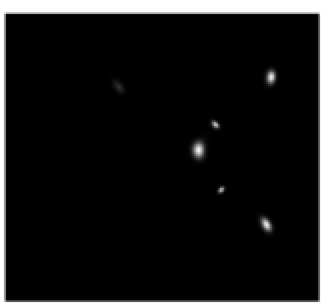


Specular

- Specular reflection
  - **r** is the “perfect reflection direction”
  - projection of **l** on **n**: $(\hat{\mathbf{l}} \cdot \hat{\mathbf{n}})\hat{\mathbf{n}}$

$$\hat{\mathbf{l}} - (\hat{\mathbf{l}} \cdot \hat{\mathbf{n}})\hat{\mathbf{n}} = -\left(\hat{\mathbf{r}} - (\hat{\mathbf{l}} \cdot \hat{\mathbf{n}})\hat{\mathbf{n}}\right) \Rightarrow \hat{\mathbf{r}} = 2(\hat{\mathbf{l}} \cdot \hat{\mathbf{n}})\hat{\mathbf{n}} - \hat{\mathbf{l}}$$

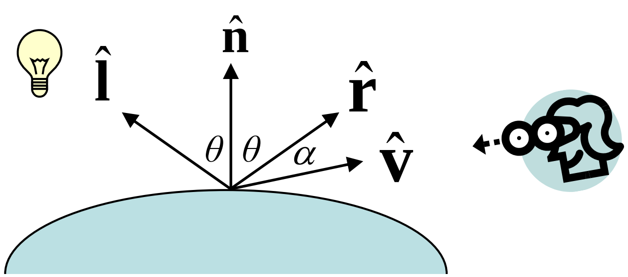
  - the closer α is to zero, the more intense is the specular reflection
  - large exponent *f* means specular highlights are smaller



$$I = I_s k_s (\cos \alpha)^f$$

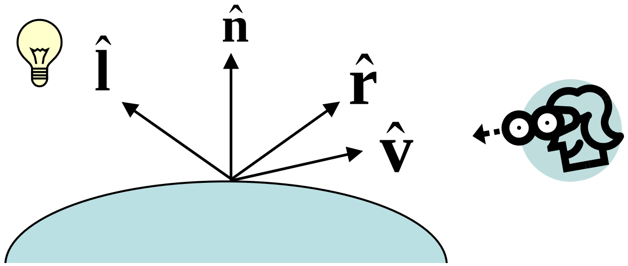
$$I = I_s k_s (\hat{\mathbf{v}} \cdot \hat{\mathbf{r}})^f$$

$$f \geq 0$$

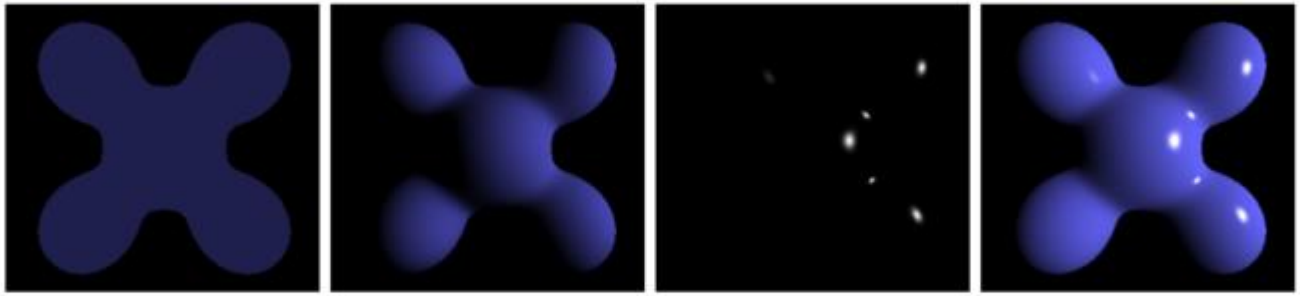


Phong Illumination Model

- Putting all together:
  - Final equation involves properties of materials and lights

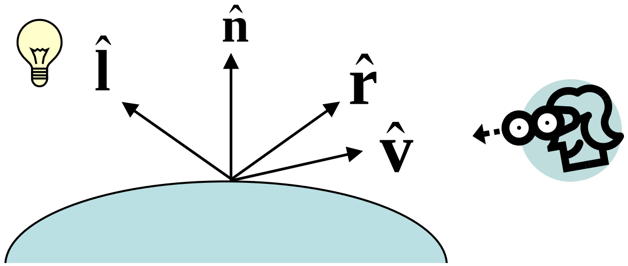


$$I = I_a k_a + I_d k_d (\hat{\mathbf{l}} \cdot \hat{\mathbf{n}}) + I_s k_s (\hat{\mathbf{v}} \cdot \hat{\mathbf{r}})^f$$

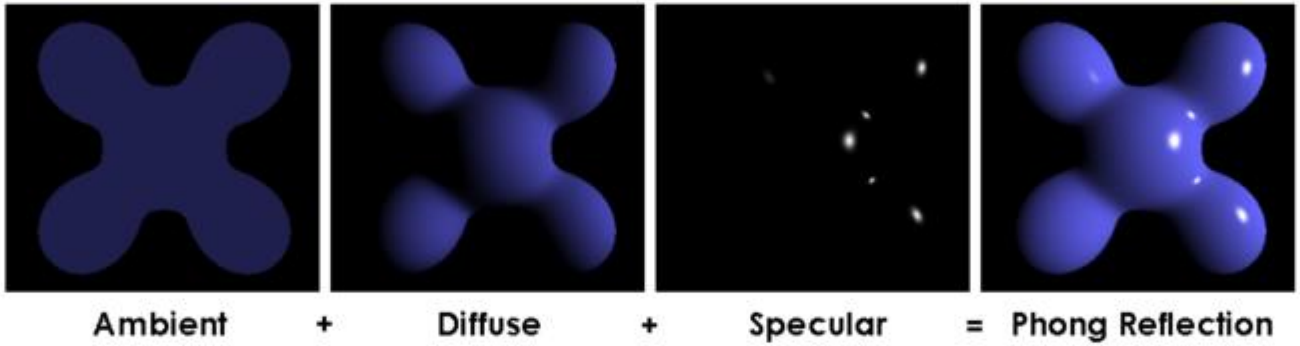


Phong Illumination Model

- Putting all together:
  - Final equation involves properties of materials and lights



$$I = I_a k_a + I_d k_d \underbrace{(\hat{\mathbf{l}} \cdot \hat{\mathbf{n}})}_{\text{Encodes alignment of light direction and surface normal}} + I_s k_s \underbrace{(\hat{\mathbf{v}} \cdot \hat{\mathbf{r}})}_{\text{Encodes alignment between viewer position and light reflection}}^{f}$$



Extensions

- Emissive light
  - Emissive intensity constant k_e
  - Similar to ambient color, but does not depend on the color of the light
  - Models an amount of emitted light

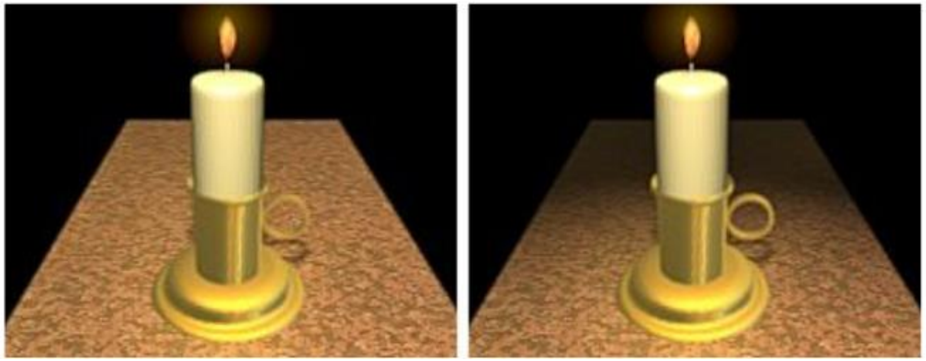
$$I = k_e + I_a k_a + I_d k_d (\hat{\mathbf{l}} \cdot \hat{\mathbf{n}}) + I_s k_s (\hat{\mathbf{v}} \cdot \hat{\mathbf{r}})^f$$

Extensions

- Light-source attenuation
  - With the equation seen so far, two parallel surfaces of same material, no matter their distances to the light source, will have the same diffuse value
  - A new term can be introduced to correct that:

$$f_{att} = \frac{1}{d_L^2}$$

$$I = I_a k_a + f_{att} I_d k_d (\hat{\mathbf{l}} \cdot \hat{\mathbf{n}}) + I_s k_s (\hat{\mathbf{v}} \cdot \hat{\mathbf{r}})^f$$



Colored lights

- The illumination model is a combination of **light properties** and **material properties:**
  - **Light intensities** (*I*) are determined per component r,g,b
  - **Material properties** (*k*) also per component

$$I^R = k_e{}^R + I_a{}^R k_a{}^R + I_d{}^R k_d{}^R (\hat{\mathbf{l}} \cdot \hat{\mathbf{n}}) + I_s{}^R k_s{}^R (\hat{\mathbf{v}} \cdot \hat{\mathbf{r}})^f$$

$$I^G = k_e{}^G + I_a{}^G k_a{}^G + I_d{}^G k_d{}^G (\hat{\mathbf{l}} \cdot \hat{\mathbf{n}}) + I_s{}^G k_s{}^G (\hat{\mathbf{v}} \cdot \hat{\mathbf{r}})^f$$

$$I^B = k_e{}^B + I_a{}^B k_a{}^B + I_d{}^B k_d{}^B (\hat{\mathbf{l}} \cdot \hat{\mathbf{n}}) + I_s{}^B k_s{}^B (\hat{\mathbf{v}} \cdot \hat{\mathbf{r}})^f$$

Example GLSL code

- File phong.frag
  - fragment shader applying the illumination equation for every pixel

```
#version 400

uniform vec3     lPos;     // light position
uniform vec3[3]  lInt;     // light intensities: ambient, diffuse, and specular
uniform vec3[4]  mColors;  // material colors   : ambient, diffuse, specular, and emission
uniform float[2] mParams;  // material params   : shininess, transparency

in vec3 Pos;
in vec3 Norm;
out vec4 fColor;

vec4 shade( vec3 p, vec3 n, vec3 lp, vec3[3] li, vec3 ka, vec3 kd, vec3 ks, vec3 emi,
            float sh, float alpha );

void main()
{
    fColor = shade( Pos, Norm, lPos, lInt,
                    mColors[0], mColors[1], mColors[2], mColors[3],
                    mParams[0], mParams[1] );
}
```

Example GLSL code

- File phong.frag

```
vec4 shade ( vec3 p, vec3 n, vec3 lp, vec3[3] li, vec3 ka, vec3 kd, vec3 ks, vec3 emi,
             float sh, float alpha )
{
   vec3 l = normalize( lp );          // light direction from p
   vec3 v = normalize( vec3(0,0,1) ); // viewer vector (along z)
   vec3 r = reflect( -l, n );         // the reflected light ray

   float dotln = dot(l,n);

   vec3 amb = li[0] * ka;
   vec3 dif = li[1] * kd * max( dotln, 0.0 ); // lambertian component
   vec3 spe = li[2] * ks * pow( max( dot(v,r), 0.0 ), sh ) * max( dotln, 0.0 );

   return vec4( amb + dif + spe + emi, alpha );
}
```

$$I = k_e + I_a k_a + I_d k_d (\hat{\mathbf{l}} \cdot \hat{\mathbf{n}}) + I_s k_s (\hat{\mathbf{v}} \cdot \hat{\mathbf{r}})^f$$