Lighting and Shading

Light Sources
Phong Illumination Model
Normal Vectors
[Angel Ch. 6.1-6.4]
Outline

• Global and Local Illumination
• Normal Vectors
• Light Sources
• Phong Illumination Model
Global Illumination

- Ray tracing
- Radiosity
- Photon Mapping
- Follow light rays through a scene
- Accurate, but expensive (off-line)
Raytracing Example

Martin Moeck,
Siemens Lighting
Radiosity Example

Restaurant Interior. Guillermo Leal, Evolucion Visual
Local Illumination

• Approximate model

• Local interaction between light, surface, viewer

• **Phong model** (this lecture): fast, supported in OpenGL

• GPU shaders

• Pixar Renderman (offline)
Local Illumination

- Approximate model

- Local interaction between light, surface, viewer

- Color determined only based on surface normal, relative camera position and relative light position

- What effects does this ignore?
Outline

• Global and Local Illumination
• Normal Vectors
• Light Sources
• Phong Illumination Model
Normal Vectors

• Must calculate and specify the normal vector
  – Even in OpenGL!

• Two examples: plane and sphere
Normals of a Plane, Method I

- Method I: given by $ax + by + cz + d = 0$
- Let $p_0$ be a known point on the plane
- Let $p$ be an arbitrary point on the plane
- Recall: $u \cdot v = 0$ if and only if $u$ orthogonal to $v$
- $n \cdot (p - p_0) = n \cdot p - n \cdot p_0 = 0$

- Consequently $n_0 = \begin{bmatrix} a & b & c \end{bmatrix}^T$
- Normalize to $n = n_0/|n_0|$
Normals of a Plane, Method II

• Method II: plane given by $p_0$, $p_1$, $p_2$
• Points must not be collinear
• Recall: $u \times v$ orthogonal to $u$ and $v$

• $n_0 = (p_1 - p_0) \times (p_2 - p_0)$

• Order of cross product determines orientation
• Normalize to $n = n_0/|n_0|$
Normals of Sphere

- Implicit Equation $f(x, y, z) = x^2 + y^2 + z^2 - 1 = 0$
- Vector form: $f(p) = p \cdot p - 1 = 0$
- Normal given by gradient vector

$$n_0 = \begin{bmatrix} \frac{\partial f}{\partial x} \\ \frac{\partial f}{\partial y} \\ \frac{\partial f}{\partial z} \end{bmatrix} = \begin{bmatrix} 2x \\ 2y \\ 2z \end{bmatrix} = 2p$$

- Normalize $n_0/|n_0| = 2p/2 = p$
Reflected Vector

- Perfect reflection: angle of incident equals angle of reflection
- Also: $l$, $n$, and $r$ lie in the same plane
- Assume $|l| = |n| = 1$, guarantee $|r| = 1$

Solution:

\[
\alpha = -1 \quad \text{and} \quad \beta = 2 (l \cdot n)
\]

\[
r = 2 (l \cdot n) n - l
\]
Outline

• Global and Local Illumination
• Normal Vectors
• Light Sources
• Phong Illumination Model
Light Sources and Material Properties

• Appearance depends on
  – Light sources, their locations and properties
  – Material (surface) properties:
  – Viewer position
Types of Light Sources

- **Ambient light**: no identifiable source or direction
- **Point source**: given only by point
- **Distant light**: given only by direction
- **Spotlight**: from source in direction
  - Cut-off angle defines a cone of light
  - Attenuation function (brighter in center)
Point Source

- Given by a point $p_0$

- Light emitted equally in all directions

- Intensity decreases with square of distance

\[ I \propto \frac{1}{|p - p_0|^2} \]
Limitations of Point Sources

- Shading and shadows inaccurate
- Example: penumbra (partial “soft” shadow)
- Similar problems with highlights
- Compensate with attenuation
  \[ q = \frac{1}{a + bq + cq^2} \]
  where \( q = \text{distance } |p - p_0| \)
  \( a, b, c \) constants
- Softens lighting
- Better with ray tracing
- Better with radiosity
Distant Light Source

• Given by a direction vector
• Simplifies some calculations
• In OpenGL:
  – Point source \([x \ y \ z \ 1]^T\)
  – Distant source \([x \ y \ z \ 0]^T\)
Spotlight

- Most complex light source in OpenGL
- Light still emanates from point
- Cut-off by cone determined by angle $\theta$
Global Ambient Light

- Independent of light source
- Lights entire scene
- Computationally inexpensive
- Simply add $[G_R \ G_G \ G_B]$ to every pixel on every object
- Not very interesting on its own. A cheap hack to make the scene brighter.
Outline

• Global and Local Illumination
• Normal Vectors
• Light Sources
• Phong Illumination Model
Phong Illumination Model

- Calculate color for arbitrary point on surface
- Compromise between realism and efficiency
- Local computation (no visibility calculations)
- Basic inputs are material properties and \( l, n, v \):

\[
\begin{align*}
  l &= \text{unit vector to light source} \\
  n &= \text{surface normal} \\
  v &= \text{unit vector to viewer} \\
  r &= \text{reflection of } l \text{ at } p \\
    &\quad \text{(determined by } l \text{ and } n) 
\end{align*}
\]
Phong Illumination Overview

1. Start with global ambient light \([G_R \ G_G \ G_B]\)
2. Add contributions from each light source
3. Clamp the final result to \([0, 1]\)

• Calculate each color channel \((R,G,B)\) separately
• Light source contributions decomposed into
  – Ambient reflection
  – Diffuse reflection
  – Specular reflection
• Based on ambient, diffuse, and specular lighting and material properties
Ambient Reflection

\[ I_a = k_a L_a \]

- Intensity of ambient light is uniform at every point
- Ambient reflection coefficient \( k_a \), \( 0 \leq k_a \leq 1 \)
- May be different for every surface and r,g,b
- Determines reflected fraction of ambient light
- \( L_a = \) ambient component of light source (can be set to different value for each light source)
- Note: \( L_a \) is not a physically meaningful quantity
Diffuse Reflection

- Diffuse reflector scatters light
- Assume equally all direction
- Called **Lambertian** surface
- Diffuse reflection coefficient $k_d$, $0 \leq k_d \leq 1$
- Angle of incoming light is important
Lambert’s Law

Intensity depends on angle of incoming light.
Diffuse Light Intensity Depends On Angle Of Incoming Light

- Recall
  \[ l = \text{unit vector to light} \]
  \[ n = \text{unit surface normal} \]
  \[ \theta = \text{angle to normal} \]
- \[ \cos \theta = l \cdot n \]
- \[ I_d = k_d \cdot L_d \cdot (l \cdot n) \]
- With attenuation:
  \[ I_d = \frac{k_dL_d}{a + bq + cq^2} (l \cdot n) \]

q = distance to light source,
\( L_d \) = diffuse component of light
Specular Reflection

- Specular reflection coefficient $k_s$, $0 \leq k_s \leq 1$
- Shiny surfaces have high specular coefficient
- Used to model specular highlights
- Does **not** give mirror effect
  (need other techniques)

specular reflection

specular highlights
Specular Reflection

• Recall
  \( \mathbf{v} = \) unit vector to camera
  \( \mathbf{r} = \) unit reflected vector
  \( \phi = \) angle between \( \mathbf{v} \) and \( \mathbf{r} \)
  \( \cos \phi = \mathbf{v} \cdot \mathbf{r} \)

• \( I_s = k_s \: L_s \: (\cos \phi)^\alpha \)

• \( L_s \) is specular component of light
• \( \alpha \) is shininess coefficient
• Can add distance term as well
Shininess Coefficient

- \( I_s = k_s L_s (\cos \phi)^\alpha \)
- \( \alpha \) is the shininess coefficient

Higher \( \alpha \) gives narrower curves

Source: Univ. of Calgary

Source:
Univ. of Calgary

\( \alpha = 1 \)
Summary of Phong Model

• Light components for each color:
  – Ambient \( (L_a) \), diffuse \( (L_d) \), specular \( (L_s) \)

• Material coefficients for each color:
  – Ambient \( (k_a) \), diffuse \( (k_d) \), specular \( (k_s) \)

• Distance \( q \) for surface point from light source

\[
I = \frac{1}{a + bq + cq^2} \left( k_d L_d (l \cdot n) + k_s L_s (r \cdot v)^\alpha \right) + k_a L_a
\]

\( l \) = unit vector to light  \( r = l \) reflected about \( n \)
\( n \) = surface normal  \( v = \) vector to viewer
BRDF

- Bidirectional Reflection Distribution Function
- Must measure for real materials
- Isotropic vs. anisotropic
- Mathematically complex
- Programmable pixel shading

Lighting properties of a human face were captured and face re-rendered;
Institute for Creative Technologies
Summary

- Global and Local Illumination
- Normal Vectors
- Light Sources
- Phong Illumination Model