Lighting and Shading

Light Sources
Phong Illumination Model
Normal Vectors

[Angel, Ch. 6.1-6.4]
Remarks About Assignment 2

• Remember that object transformations are applied in the reverse order in which they appear in the code!
• Remember that transformation matrices are multiplied on the right and executed from right to left: \((R \ S \ T)v = R \ (S \ (T \ v))\)!
• Look at the model solution (when it is out) and make sure you understand it before the midterm
Outline

• Light Sources
• Phong Illumination Model
• Normal Vectors
Lighting and Shading

- Approximate physical reality
- Ray tracing:
  - Follow light rays through a scene
  - Accurate, but expensive (off-line)
- Radiosity:
  - Calculate surface inter-reflection approximately
  - Accurate, especially interiors, but expensive (off-line)
- Phong Illumination model (this lecture):
  - Approximate only interaction light, surface, viewer
  - Relatively fast (on-line), supported in OpenGL
Radiosity Example

Restaurant Interior. Guillermo Leal, Evolucion Visual
Raytracing Example

Martin Moeck, Siemens Lighting
Light Sources and Material Properties

- Appearance depends on
  - Light sources, their locations and properties
  - Material (surface) properties
  - Viewer position
- Ray tracing: from viewer into scene
- Radiosity: between surface patches
- Phong Model: at material, from light to viewer
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)
- **Light source described by a luminance**
  - Each color is described separately
  - \( I = [I_r \ I_g \ I_b]^T \) (I for intensity)
  - Sometimes calculate generically (applies to r, g, b)
Ambient Light

- Global ambient light
  - Independent of light source
  - Lights entire scene
- Local ambient light
  - Contributed by additional light sources
  - Can be different for each light and primary color
- Computationally inexpensive

\[
I_a = \begin{bmatrix}
I_{ar} \\
I_{ag} \\
I_{ab}
\end{bmatrix}
\]
Point Source

• Given by a point \( p_0 \)
• Light emitted equally in all directions

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

• Intensity decreases with square of distance

\[
I(p, p_0) = \frac{1}{|p - p_0|^2}I(p_0)
\]
Limitations of Point Sources

• Shading and shadows inaccurate
• Example: penumbra (partial “soft” shadow)
• Similar problems with highlights
• Compensate with attenuation
  
  \[
  \frac{1}{(a + bd + cd^2)} \quad \text{d} = \text{distance } |p - p_0| \\
  \text{a, b, c constants}
  \]

• Softens lighting
• Better with ray tracing
• Better with radiosity
Distant Light Source

- Given by a vector \( \mathbf{v} \)
- 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 $\theta$
Spotlight Attenuation

- Spotlight is brightest along \( \mathbf{l}_s \)
- Vector \( \mathbf{v} \) with angle \( \phi \) from \( \mathbf{p} \) to point on surface
- Intensity determined by \( \cos \phi \)
- Corresponds to projection of \( \mathbf{v} \) onto \( \mathbf{l}_s \)
- **Spotlight exponent** \( e \) determines rate

\[
I = \cos^e(\phi) = (\mathbf{v} \cdot \mathbf{l}_s)^e
\]

Diagram correction \([u = \theta, f = \phi]\)

for \( e = 1 \)

for \( e > 1 \)

curve narrows
Outline

• Light Sources
• Phong Illumination Model
• Normal Vectors
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 \( l, n, v: \)

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

- Calculate each primary color separately
- Start with global ambient light
- Add reflections from each light source
- Clamp to [0, 1]
- Reflection decomposed into
  - Ambient reflection
  - Diffuse reflection
  - Specular reflection
- Based on ambient, diffuse, and specular lighting and material properties
Ambient Reflection

- Intensity of ambient light 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
- $L_a = \text{ambient component of light source}$
- Ambient intensity $I_a = k_a L_a$
- 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 still critical
Lambert’s Law

- Intensity depends on angle of incoming light
- Recall
  \[ l = \text{unit vector from light} \]
  \[ n = \text{unit surface normal} \]
  \[ \theta = \text{angle to normal} \]
- \[ \cos \theta = l \cdot n \]
- \[ I_d = k_n (l \cdot n) L_d \]

\[ I_d = \frac{k_d}{a + bq + cq^2}(l \cdot n)L_d \]

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
- Do not get mirror effect (need other techniques)

specular reflection  specular highlights
Shininess Coefficient

- $L_s$ is specular component of light
- $r$ is vector of perfect reflection of $l$ about $n$
- $v$ is vector to viewer
- $\phi$ is angle between $v$ and $r$
- $I_s = k_s L_s \cos^\alpha \phi$
- $\alpha$ is shininess coefficient
- Compute $\cos \phi = r \cdot v$
- Requires $|r| = |v| = 1$
- Multiply distance term

Higher $\alpha$ is narrower
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

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

I = vector from light  \quad r = \mathbf{l} \text{ reflected about } \mathbf{n}

n = surface normal  \quad v = \text{ vector to viewer}
Outline

- Light Sources
- Phong Illumination Model
- Normal Vectors
Normal Vectors

• Summarize Phong

\[I = \frac{1}{a + bq + cq^2}(k_dL_d(l \cdot n) + k_sL_s(r \cdot v)^\alpha) + k_aL_a\]

• Surface normal \(n\) is critical
  – Calculate \(l \cdot n\)
  – Calculate \(r\) and then \(r \cdot v\)

• 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: $u \cdot v = 0$ iff $u$ orthogonal $v$
• $n \cdot (p - p_0) = n \cdot p - n \cdot p_0 = 0$
• Consequently $n_0 = [a\ b\ c\ 0]^T$
• 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 may 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 produce 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 $n_0/|n_0| = 2p/2 = p$
Angle of Reflection

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

\[
l \cdot n = \cos \theta = n \cdot r
\]

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

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

Perhaps easier geometrically
Summary: Normal Vectors

• Critical for Phong model (diffuse and specular)
• Must calculate accurately (even in OpenGL)
• Pitfalls
  – Not unit length
  – How to set at surface boundary?
• Omitted
  – Refraction of transmitted light (Snell’s law)
  – Halfway vector (yet another optimization)
Summary

- Light Sources
- Phong Illumination Model
- Normal Vectors
• Polygonal shading
• Lighting and shading in OpenGL
• [Demo]
• Moving and stationary light sources
Announcements

- Assignment 2 back Thursday
- Check out model solution (before midterm)
- Assignment 3 due a week from Thursday