Local Illumination

Introduction to Computer Graphics
Torsten Möller / Manfred Klaffenböck
Image Pipeline

SPD → XYZ → Tone Reproduction → RGB → Display

Dither → \( \gamma \)
Graphics Pipeline

Hardware

Modelling → Transform → Visibility

Illumination + Shading

Perception, Interaction

Color

Texture/Realism
Reading

• Chapter 5 - Angel
• Chapter 27.1-27.5 - Hughes, van Dam, et al.
• Chapter 10, Shirley+Marschner
Illumination

• No lights
Illumination

- No lights in OpenGL
Illumination (2)

- Ambient Lights
Illumination (3)

- Diffuse + specular lights
Importance of illumination

• **Visual realism** with surface appearance
• Provides **3D cues**, e.g., essential in shape from shading or inference from a photograph or photographs

Illumination vs. shading

• Often collectively referred to as shading, e.g., in text and [Foley et al.]

• If one has to make a distinction
  – **Illumination**: calculates intensity at a particular point on a surface
  – **Shading**: uses these calculated intensities to shade the whole surface or the whole scene
Difficulty

• Visibility is conceptually easy (computationally, still rather expensive) and we can be quite precise
  – The single criterion is whether something lies along the projector

• The physics (optics, thermal radiation, etc.) behind the interaction between lights and material is much more complex
  – Simulating exact physics is expensive
  – Even if we make simplifying assumptions, e.g., point light, only consider RGB components, etc.
Scattering of light

• Light strikes surface A
  – Some scattered
  – Some absorbed

• Some of scattered light strikes B
  – Some scattered
  – Some absorbed

• Some of this scattered light strikes A, and so on
Rendering equation

- The infinite scattering and absorption of light can be described by the rendering equation
  - The equation sets up an equilibrium situation
  - Cannot be solved analytically in general and need numerical approximations
  - (Whitted) Ray tracing is a special case for perfectly reflecting surfaces

- Rendering equation is global and includes
  - Shadows
  - Multiple scattering from object to object

© Machiraju/Zhang/Möller
Global illumination

• More physical and illuminates the whole scene by taking into account the interchange of light between all surfaces – follows the rendering equation

• Prominent examples: radiosity and ray tracing

• More visual realism with more faithful simulation of physics, but much slower than local illumination

• In contrast to pipeline model which shades each polygon independently (leads to local illumination)
Local illumination

- Much more simplified and non-physical
- Computes the color of a point independently through direct illumination by the light source(s)
- No explicit account for inter-object reflections
- Shading calculations depend only on:
  - Material properties, e.g., reflectance of R, G, & B
  - Location and properties of the light source
  - Local geometry and orientation of the surface
  - Location of viewer
Recall assumption on light

- Light travels in a straight line
- Ignore its frequency composition and care about its component-wise intensity (energy) – R, G, B
- **Light source**: general light sources are difficult to work with because we must integrate light coming from all points on the source
Simplified light source models

- Ideal point light source
  - Modeled by pos+color; shines equally in all directions
  - Can take into account decay with respect to distance
  - Follows perspective projections
  - Parallel projections: distant source = point at infinity

- Spotlight
  - Restrict light from ideal point source

- (Global) ambient light
  - General brightness (same amount of light everywhere)
  - A rough model for inter-surface reflections from all light sources
Light Effects

Light = refl. + absorbed + trans.

Light = ambient + diffuse + specular

\[ I = k_a I_a + k_d I_d + k_s I_s \]
Light-material interactions

• Specular reflection:
  – Most light reflected in a narrow range of angles
  – Perfect reflection: minus absorption, all light is reflected about the angle of reflection

• Diffuse reflection:
  – Reflected light is scattered in all directions
  – Perfect diffuser: follows Lambertian Law and surface looks the same from all directions

• Translucence and light refraction:
  – Light is refracted (and reflected), e.g., glass, water
Geometry of perfect reflection

• Perfect reflection, a.k.a. mirror reflection

\[ n: \text{normal vector} \]
\[ l: \text{vector pointing to light source} \]
\[ r: \text{ray of reflection} \]
\[ u_i : \text{angle of incidence} \]
\[ u_r : \text{angle of reflection} \]

Vectors \( l, n, \) and \( r \) lie in the same plane

\[ u_i = u_r \]
Geometry of refraction

- Consider a surface transmitting all light

\( t \): ray of refraction
\( u_l \): angle of incidence
\( u_t \): angle of refraction

Vectors \( l, n, \) and \( t \) all lie in the same plane

Snell’s Law:

\[
\frac{\sin u_l}{\sin u_t} = \frac{\eta_t}{\eta_l}
\]

\( \eta_l, \eta_t \): indices of refraction – measure of relative speed of light in the two materials \((c/c_l, c/c_t)\)
An ad-hoc local illumination model

• Phong reflection or local illumination model: the color at a point on a surface is composed of
  \[ I = \text{ambient} + \text{diffuse} + \text{specular} \]

  – Introduced by Phong in 1975 and used by OpenGL
  – Not concerned with translucent surfaces
  – Efficient enough to be interactive
  – Rendering results can reasonably approximate physical reality under a variety of lighting conditions and material properties
Phong local illumination

• Phong reflection or local illumination model: the color at a point on a surface is composed of
  • \( I = \text{ambient} + \text{diffuse} + \text{specular} \)
    – Ambient component: general brightness due to the light source, regardless of surface orientation and light position or direction
    – Diffuse component: light diffusely reflected off surface
    – Specular component: light speculatively reflected off surface
  • There is also a global ambient light
    – Models general brightness of scene regardless of light sources
Phong model: the four vectors

- Phong model uses four vectors to calculate a color for a point $p$ on a surface:
  - $l : p$ to point light source
  - $n$ : normal vector to the plane or surface
  - $v : p$ to eye (view point) or center of projection
  - $r$ : vector of perfect reflection
Exercise: computation of vectors

• How to compute \( \mathbf{r} \), given \( \mathbf{n} \) and \( \mathbf{l} \)?
Exercise: computation of vectors

• How to compute \textbf{r}, given \textbf{n} and \textbf{l}?

Answer: \( \textbf{r} = 2(\textbf{n} \cdot \textbf{l})\textbf{n} - \textbf{l} \)
Exercise: computation of vectors

• How to compute \( \mathbf{r} \), given \( \mathbf{n} \) and \( \mathbf{l} \)?

   Answer: \( \mathbf{r} = 2(\mathbf{n} \cdot \mathbf{l})\mathbf{n} - \mathbf{l} \)

• Why?

   \[ \mathbf{l} + \mathbf{r} = (2 \cos \theta) \mathbf{n} \]
Perfect diffuse reflections

- Point of surface appears the same from all views
- Models dull, rough surface and follows Lambertian Law

\[ I_d = k_d L_d \cos \theta \]
\[ = k_d L_d (n \cdot l), 0 \leq \theta \leq \pi/2 \]
\[ I_d = 0, \text{ otherwise} \]

- Equal area, same reflected light

\( L_d \): diffuse term of light source
\( k_d \): material’s diffuse reflection coefficient
\( n \): normal vector at point (normalized)
\( l \): light source vector (normalized)

© Machiraju/Zhang/Möller
Aside 1: Lambertian reflection explained

• What we want:
• Per unit area towards the viewer, how much reflected diffuse light do we see?
• Unit area towards the viewer $\rightarrow \frac{1}{\cos \phi}$
  units of area on the surface
• How much light per unit surface area comes from the light source?
Aside 2: Lambertian reflection explained

• Consider a light beam with a cross-sectional area dA
  – Surface 1: receives 1 unit of light per unit surface area
  – Surface 2: receives \( \cos(q) \) unit of light per unit surface area
  – So light received is proportional to cosine of incidence angle \( q \)

• This is independent of the viewer
Aside 3: Take into account of viewer

- Thus, amount of light received per unit surface area is $L_d \cos \theta$
- **Lambertian Law**: Amount of light reflected from a unit differential surface area towards the viewer is directly proportional to cosine of the view angle $\phi$ – i.e., the smaller the angle, the more reflects towards that viewer

Putting all together, the amount of light seen by the viewer per unit area is

$$I_d = k_d L_d \cos \theta \cdot \left(1 / \cos \phi\right) \cdot \cos \phi$$

$$= k_d L_d \cos \theta$$
Examples: diffuse and ambient

Lambertian reflection ($L_d = 1; k_d = 0.4, 0.55, 0.7, 0.85, 1$):

Add ambient light ($L_a = L_d = 1; k_a = 0, 0.15, 0.3, 0.45, 0.6$):
Specular Light

- Light that is reflected from the surface unequally in all directions
- Models reflections on shiny surfaces

\[
I = k_s I_s \cos^n \alpha
\]

\[
= k_s I_s (E \cdot R)^n
\]
Specular reflection examples

Graphs of $\cos^n()$:

$L_a = L_d = 1.0$

$k_a = 0.1$  \hspace{1cm}  $k_s = 0.1$

$k_d = 0.45$

$k_s = 0.25$

$n = 3.0$  \hspace{1cm}  $n = 5.0$  \hspace{1cm}  $n = 10.0$  \hspace{1cm}  $n = 27$  \hspace{1cm}  $n = 200$

Roughly, values of $n$ between 100 and 200 correspond to metals and values between 5 and 10 give surface that look like plastic.
Ambient light component

• Models general brightness due to light source
  – Light or eye location does not matter
  – Intensity of ambient light is the same at all points
  – Really crude way to account for hard-to-compute lighting effects, e.g., inter-surface reflections, etc.

\[ I_a = k_a L_a, \quad 0 \leq k_a \leq 1 \]

\( L_a \): ambient light energy (property of light source)
\( k_a \): ambient reflection coefficient (a material property)
Phong Illumination Model cont.

- BRDF for Phong illumination
Phong Illumination Model cont.

- Illustration of ambient, diffuse, and specular components
- Different angles of incident light

<table>
<thead>
<tr>
<th>Phong</th>
<th>$f_{ambient}$</th>
<th>$f_{diffuse}$</th>
<th>$f_{specular}$</th>
<th>$f_{total}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\phi_i = 60^\circ$</td>
<td><img src="image1.png" alt="Image" /></td>
<td><img src="image2.png" alt="Image" /></td>
<td><img src="image3.png" alt="Image" /></td>
<td><img src="image4.png" alt="Image" /></td>
</tr>
<tr>
<td>$\phi_i = 25^\circ$</td>
<td><img src="image5.png" alt="Image" /></td>
<td><img src="image6.png" alt="Image" /></td>
<td><img src="image7.png" alt="Image" /></td>
<td><img src="image8.png" alt="Image" /></td>
</tr>
<tr>
<td>$\phi_i = 0^\circ$</td>
<td><img src="image9.png" alt="Image" /></td>
<td><img src="image10.png" alt="Image" /></td>
<td><img src="image11.png" alt="Image" /></td>
<td><img src="image12.png" alt="Image" /></td>
</tr>
</tbody>
</table>
Depth Cueing / Illumination

- Problems with $f_{\text{att}} = 1/d^2_L$ (square decay of energy):
  - Too little change when light moves in far-away regions
  - Wild change when object moves closer to light
- Often a factor of $1/d$ or $1/(d+c)$ is used
- A way to simulate fog - linear fade …
Depth Cueing / Illumination (2)

- use start f (front) and end b (back) fade distances
- \(d = \text{distance from viewer:}\)

\[
I = I_f \quad \text{for } d < z_f
\]
\[
= \frac{(z_b - d)}{(z_b - z_f)} I_f + \frac{(d - z_f)}{(z_b - z_f)} I_b \quad \text{for } z_f < d < z_b
\]
\[
= I_b \quad \text{for } d > z_b
\]
Depth Cueing / Illumination (3)

• typical: use the attenuation coefficient

\[ f_{\text{att}} = \min\left( \frac{1}{a + b d_L + c d_L^2}, 1 \right) \]

• Applied to diffuse and specular terms only
Transmission

- Objects can transmit light!
- Light may be transmitted specularly (transparent) or diffusely (translucent), like reflected light
  - specular transmission is exhibited by materials such as glass
  - diffuse transmission is exhibited by material which scatter the transmitted light (translucent, e.g. frosted glass, church)
Transparency

- 2 types of transparency - nonrefractive and refractive (more realistic)
Non-Refractive Transparency

- Requires sorting! (BTF vs. FTB)
- “over” operator - Porter & Duff 1984

\[ C_{\text{out}} = C_{\text{in}} \cdot (1 - \alpha) + C \cdot \alpha \]
\[ C_{\text{in}}(i) = C_{\text{out}}(i - 1) \]
Refractive Transparency

- Transmitted light may be refracted according to Snell’s law!

\[
\frac{\sin q_i}{\sin q_t} = \frac{\eta_t}{\eta_i}
\]
Refractive Transparency

• $\eta_t$, $\eta_i$ - indices of refraction (ratio of speed of light in vacuum to the speed of light in the medium)

• depend on:
  – temperature
  – wavelength of light
  – a good book :)

• hence - refracted light can have a different color
Total internal reflection

• When light passes from one medium into another whose index of refraction is lower the transmitted angle is larger than the incident angle

• at one point the light gets reflected - called total internal reflection
Optical Manhole

- Total Internal Reflection
- For water $\eta_w = 4/3$
Refractive Transparency

• In general - refraction effects are ignored in simple renderers (e.g. scanline z-buffer)

• most materials in scene are non-transmitting (so it’s ok most of the time)

• Extension to simple model:

\[ I = k_a I_a + k_d I_d + k_s I_s + k_t I_t \]

• \( k_t \) - transmission coefficient of a material

• \( I_t \) - intensity of transmitted light
Implementation - an example

• Modify the z-buffer so that it maintains a depth-sorted list of visible surfaces at each pixel (A-buffer)

• for each polygon:
  – if its in front of nearest opaque surface
    • if its opaque
      – it becomes the nearest opaque surface (i.e. insert in the list and delete everything behind it)
    • else
      – insert by z into the list
  – else ignore it
Implementation - an example

• Blend two polygons like in the non-refractive version

\[ C_{\text{out}} = C_{\text{in}} \cdot (1 - \alpha) + C \cdot \alpha \]
Implementation - an example

• linear approximation doesn’t model curved surfaces well. It does not account for the increased thickness of the material at silhouette edges, which reduces transparency.

• To approximate these effects - use a non-linear transparency factor:

\[ \alpha = \alpha_{\text{max}} + (\alpha_{\text{min}} - \alpha_{\text{max}}) \times \left[ 1 - (1 - |N_z|)^p \right] \]
Implementation - an example

\[\alpha = \alpha_{\text{max}} + (\alpha_{\text{min}} - \alpha_{\text{max}}) \times \left[ 1 - (1 - |N_z|)^p \right]\]

- \(\alpha_{\text{min}}, \alpha_{\text{max}}\) - minimum and maximum transparencies for the surface
- \(N_z\) - is the z component of the surface normal
- \(p\) - is a modeling parameter (transparency power factor)
Improved light sources

• Distant light source
  – Parallel rays of light
  – Light vector $\mathbf{l}$ does not change from point to point (efficiency)

• Spotlight source
  – A “cone” of light controlled by angle $u$ ($u = \pi$ for point source)
  – Varying intensity as if for specular reflection using $\cos^n(\theta)$
Improved light sources

- General Spotlight
- vary the intensity