Chap.6
Local Illumination

Ensino de Informática (3326) - 4º ano, 2º semestre
Engenharia Electrotécnica (2287) - 5º ano, 2º semestre
Engenharia Informática (2852) - 4º ano, 2º semestre
Motivation

- Illumination model = approximation of real-world illumination
- 3D feel, depth perception
- For most OpenGL applications, we use the Phong illumination model because it is time efficient and good enough at representing the real world.
Light Source Independent Illumination Models

- Depth Shading
  - Color or intensity determined solely by "depth" of polygon.
  - Darker colors or intensities at lower elevations.
  - Effective in modeling terrain or surface data
  - Avoids complex calculations of lighting dependent models
  - Simulates realism

- Depth Cueing
  - Reduce intensity of pixel as the distance from the observer increases
  - Simulates reduction in clarity as distances from the observer increases
  - Image fades in the distance
  - Often used in medical imaging
Light Source Dependent Illumination Models

- What an object looks like depends on
  - Properties of the light source such as color, distance from object, direction from object, intensity of source
  - Surface characteristics of object such as color and reflectance properties
  - Location of the observer

- Light striking a surface of an object can be
  - Reflected (Diffuse reflection & Specular reflection)
  - Absorbed
  - Transmitted (Translucent or transparent)
  - Combination of all three
Lighting-based Illumination Models

- Direct or Local Illumination
  - Single interaction of light & objects
  - Real-time supported by OpenGL
  - Example: *Phong illumination model*

- Indirect or Global Illumination
  - Multiple interaction of light & objects: *inter-objects reflections, shadows, refractions*
  - Not real-time (yet)
  - Examples: *raytracing, radiosity, photon mapping …*
Overview:
lighting-based models

- Direct or Local Illumination
  - Light types
  - Light sources (emission)
  - Surface materials (reflectance)

- Global Illumination
  - Shadows
  - Refractions
  - Inter-object reflections
Light types

- **Ambient** light
  - It comes from all directions; when it strikes the surface, it is scattered equally in all directions.
  - Scattering consequence: it does not depend on the viewpoint (viewer).

- **Diffuse** light
  - It comes from one direction; when it strikes the surface, it is scattered equally in all directions.
  - Scattering consequence: it does not depend on the viewpoint (viewer).

- **Specular** light
  - It comes from one direction; it tends to bounce off the surface in a preferred direction.
  - Scattering consequence: it does depend on the viewpoint (viewer).


Light types (cont.)

- **Ambient light**
  - It can be used to give a feel for the *main color in a room*.
  - Source light contributions:
    - backlighting is a room has a large ambient component
    - a spotlight outdoors has a tiny ambient component

- **Diffuse light**
  - It is the light type that is *closest to the color of light*.
  - Source light contributions:
    - any light coming from a particular position or direction

- **Specular light**
  - It is the light type that is ....
  - Source light contributions:
    - a well-collimated laser beam bouncing off a high-quality mirror produces almost 100 percent specular reflection
    - shiny metal or plastic has a high specular component
    - chalk or carpet has almost none.
Light sources and material

**Light sources**
- **Types:**
  - ambient, positional, directional, and spot light sources
- **Color**
  - The emitted light color is given by the amount of red, green, and blue light.
- **Number**
  - Light sources can individually be turned off and on.
- **Emitted light types:** ambient, diffuse, specular

**Surface material**
- It specifies how light is reflected (and absorbed)
  - The color of the surface material is given by the percentage of incoming red, green, and blue components that are reflected in various directions.
  - Different surfaces may have very different properties; some are shiny, and preferentially reflect light in certain directions, while others scatter incoming light equally in all directions. Most surfaces are somewhere in between.
- **Emitted light type:** emitted
- **Reflected light types:** ambient, diffuse, specular
Modeling Light Sources

- **Light source model:** \( I_L(P, \vec{D}, \lambda) \)
  - describes the intensity of energy,
  - leaving a light source
  - arriving at location \( P(x, y, z) \)
  - from direction \( \vec{D} \) (normalized vector)
  - with wavelength \( \lambda \)
Light Source Types

They are:

- Ambient light source
- Point light source
- Directional light source
- Spot light source
Ambient Light Source

- An object not directly lit is still visible
  - Caused by light reflected from other surfaces

- Modelled by a single ambient light source
  - Instead of computing surface reflections, specify constant ambient light for all surfaces
  - Defined solely by ambient RGB light intensities

- Intensity of ambient light of intensity $I_L$ arriving at a point $P(x,y,z)$:

\[ I(P, I_L) = I_L \]
Global Ambient Light in OpenGL

- It is not from any particular source.
- It allows us to see objects in the scene even when no light sources are specified.
- Its RGBA intensity is specified by using the `GL_LIGHT_MODEL_AMBIENT` parameter as follows:

**Example: (Global Ambient Light)**

```c
// sets global ambient light
GLfloat lmodel_ambient[]={0.2,0.2,0.2,1.0};
glLightModelfv(GL_LIGHT_MODEL_AMBIENT, lmodel_ambient);
```
Directional Light Source

- Models Point Light Source at Infinity (e.g., Sun)
  - Defined by intensities of emitted RGB light for all types, and
  - direction $\vec{D}$
- Direction important to compute reflected light
- Intensity of point light of intensity $I_L$ arriving at a point $P(x, y, z)$:

$$I(P, \vec{D}, I_L) = I_L$$

No attenuation with distance
Point Light Source

- Light emitted *radially* from single point in all directions (omni-directional source)
  - Defined by intensities of emitted RGB light for all types,
  - position $L(x,y,z)$, and
  - factors $(k_c, k_l, k_q)$ for attenuation with distance $d$ to $P(x,y,z)$

- Intensity of point light of intensity $I_L$ arriving at a point $P(x,y,z)$:

$$I(P, L, k_c, k_l, k_q, I_L) = \frac{I_L}{k_c + k_1d + k_qd^2}$$
Spot Light Source

- Light emitted in a cone (Luxo Jr. lamp)
  - Defined by intensities of emitted RGB light for all types,
  - position $L$, direction $D$, spot cut-off exponent
  - constant, linear and quadratic attenuation ($k_c, k_l, k_q$)

- Intensity of spot light of intensity $I_L$ arriving at a point $P(x,y,z)$:

$$I(P, L, k_c, k_l, k_q, I_L) = \frac{I_L (\vec{D} \cdot \vec{L})}{k_c + k_1 d + k_q d^2}$$
Direction and Position of Light Sources in OpenGL

- **Directional light source.** It is located infinitely far away from the scene.
- **Positional or point light source.** Its distance to the scene is finite.

**Example: (Directional Light Source)**

```c
// sets GL_LIGHT0 with direction (x=1.0,y=1.0,z=1.0) at an infinite position (w=0.0) in homogeneous coordinates
GLfloat light_position[]={1.0,1.0,1.0,0.0};
gllightfv(GL_LIGHT0, GL_POSITION, light_position);
```

**Example: (Positional or Point Light Source)**

```c
// sets GL_LIGHT0 at the position (x=1.0,y=1.0,z=1.0) that is finite (w≠0.0) in homogeneous coordinates
GLfloat light_position[]={1.0,1.0,1.0,1.0};
gllightfv(GL_LIGHT0, GL_POSITION, light_position);
```
Like a point light source, a spot light is also a positional light source.

Point light source. By default, the spotlight feature is disabled because the GL_SPOT_CUTOFF parameter is 180.0 degrees. This value means that light is emitted in all directions (the angle at the cone’s apex is 360 degrees, so it isn’t a cone at all).

Spot light source. The value for GL_SPOT_CUTOFF is restricted to being within the range [0.0,90.0].

**Example: (Spot Light Source)**

```c
// sets GL_LIGHT0 as a spotlight with a cutoff angle of 30 degrees
glLight(GL_LIGHT0, GL_SPOT_CUTOFF, 30.0);

// sets spotlight’s direction or the light cone axis
GLfloat spot_direction[]={-1.0,-1.0,0.0};
glLightfv(GL_LIGHT0, GL_SPOT_DIRECTION, spot_direction);
```
Color of Light Sources in OpenGL

- OpenGL allows us to associate 3 different color-related parameters with any particular light:

- **GL_AMBIENT**. It refers to the RGBA intensity of the ambient light that a particular light source adds to the scene. Default RGBA values: \((0.0,0.0,0.0,1.0)\) ⇒ no ambient light

- **GL_DIFFUSE**. It refers to the RGBA intensity of the diffuse light that a particular light source adds to the scene. Default RGBA values: \((1.0,1.0,1.0,1.0)\) for LIGHT0 (⇒ bright, white diffuse light) and \((0.0,0.0,0.0,0.0)\) for any other light.

- **GL_SPECULAR**. It refers to the RGBA intensity of the specular light that a particular light source adds to the scene. Default RGBA values: \((1.0,1.0,1.0,1.0)\) for LIGHT0 and \((0.0,0.0,0.0,0.0,0.0)\) for any other light.
Color of Light Sources in OpenGL (cont.)

Example: (Color of Ambient, Diffuse, and Specular Light)

// sets the ambient component of GL_LIGHT0
Glfloat light_ambient[]={0.0,0.0,1.0,1.0};    // blue color
Glfloat light_diffuse[]={1.0,1.0,1.0,1.0};    // white color
Glfloat light_specular[]={1.0,1.0,1.0,1.0};  // white color

glfwLightfv(GL_LIGHT0, GL_AMBIENT, light_ambient);
glfwLightfv(GL_LIGHT0, GL_DIFFUSE, light_diffuse);
glfwLightfv(GL_LIGHT0, GL_SPECULAR, light_specular);
Lighting Components

- The image to the right shows the effects of ambient (top left), diffuse (top right), specular (bottom left), and all 3 combined (bottom right).
Overview: lighting-based models

- Direct Illumination
  - Emission at light sources
  - Scattering at surfaces

- Global Illumination
  - Shadows
  - Refractions
  - Inter-object reflections
Modeling Surface Reflection

\[ R(\theta, \phi, \gamma, \psi, \lambda) \]

- Describes the amount of incident energy
- arriving from direction \((\theta, \phi)\)
- leaving in direction \((\gamma, \psi)\)
- with wavelength \(\lambda\)

\(\lambda\)

\((\theta, \phi)\)

\((\psi, \lambda)\)

Surface
Reflectance Model

Simple Analytic Model:

- Diffuse reflection +
- Specular reflection +
- Emission +
- Ambient

Based on model proposed by Phong
Diffuse Reflection

- An *ideal diffuse reflector*, at the microscopic level, is a very rough surface (real-world example: chalk)
- Because of these microscopic variations, an incoming ray of light is equally likely to be reflected in all directions over the hemisphere.
- That is, one assumes that *surface reflects equally in all directions*. 
**Diffuse Reflection**

**How Much Light is Reflected?**

- Depends on angle $\theta$ of incident light.
- The greater $\theta$ is, the less light is reflected.
- The amount of reflected light is dependent on the position of the light source and the object, but independent of the observer's position.
Diffuse Reflection

Lambertian Model

- Lambert’s cosine law (dot product)
  *The intensity of light diffuse $I_D$ reflected from a surface point $P(x,y,z)$ is proportional to the cosine of the angle between the vector $\vec{L}$ to the light source and the normal vector $\vec{N}$ perpendicular to the surface.*

  - $I_D = \text{reflected diffuse light intensity}$
  - $I = \text{light source intensity at } P(x,y,z)$
  - $K_D = \text{surface reflection coefficient}$ ($0 \leq K_D \leq 1$)
  - $\theta = \text{must be between 0 and 90 degrees}$

- \[ I_D = K_D I \cos \theta \]
- with \[ \cos \theta = \frac{\vec{N} \cdot \vec{L}}{||\vec{N}|| ||\vec{L}||} = \vec{N} \cdot \vec{L}, \]
- being $\vec{N}$ e $\vec{L}$ unit vectors

\[ I_D = K_D (\vec{N} \cdot \vec{L}) I \]
Specular Reflection

Reflection is Strongest Near Mirror Angle
- Examples: mirrors, metals
- Visible when the angle of incidence of the light from the point light source is equal to the angle of reflection toward the observer.

How Much Light is Seen?
- Depends on angle of incident light
- And angle to viewer
- For a non-perfect reflector, intensity of reflected light decreases rapidly as angle to observer increases beyond the angle of incidence.
Specular Reflection

- **Phong Model**
  - $(\cos \alpha)^n$
  - $n = \text{specular reflection exponent}$
    (perfect reflector $n=\infty$)
  - $I_S = \text{reflected specular light intensity}$
  - $I = \text{light source intensity at } P(x,y,z)$
  - $K_S = \text{coefficient of reflected specular light } (0 \leq K_S \leq 1)$
  - $\theta = \text{must be between 0 and 90 degrees}$

\[ I_S = K_S I (\cos \alpha)^n \]

with \[ \cos \alpha = \frac{\vec{V} \cdot \vec{R}}{||\vec{V}|| \cdot ||\vec{R}||} = \vec{V} \cdot \vec{R}, \]

being $\vec{V}$ and $\vec{R}$ unit vectors

\[ I_S = K_S \left( \vec{V} \cdot \vec{R} \right)^n I \]
Emissive light from an area light source

- It is produced by an **area light source**, not a point light source.
- Represents light emitting directly from one polygon or disc of an object.
- This is necessary because some real-world objects like lamps do emit light.
- So, if a lamp is part of the scene, we have to specify not only its positional light source, but also its surface material as emitting material of light.

\[ I_{EL} = I_E \]

material emission \(\neq (0,0,0,0)\)
Emissive light from an area light source in OpenGL

- By specifying an RGBA color for GL_EMISSION, you can make an object appear to be giving off light of that color.
- Since most real-world objects (except lights) do not emit light, so we can use this feature mostly to simulate lamps and other light sources in a scene.
- However, an area light source does not actually act as a light source. It is necessary to create a light source and position it at the same location as the lighting object to create such effect.

*Example: (Emissive light from material)*

```c
GLfloat mat_emission[] = {0.3, 0.2, 0.2, 0.0};
glMaterialfv(GL_FRONT, GL_EMISSION, mat_emission);
```
Ambient Reflection

- Represents Reflection of All Indirect Illumination
  - Ambient light is the illumination of an object caused by reflected light from other surfaces.
  - To calculate this exactly would be very complicated.
  - A simple model assumes ambient light is uniform in the environment.

- $I_{AL} = \text{reflected ambient light intensity}$
- $K_A = \text{coefficient of reflected ambient light}$
- $I_A = \text{ambient light intensity}$

This is a total hack (avoids complexity of global illumination)!
Reflectance Model

Simple Analytic Model:
- Diffuse reflection +
- Specular reflection +
- Emission +
- “Ambient”
Reflectance Model

- Sum Diffuse, Specular, Emission, and Ambient

<table>
<thead>
<tr>
<th>Phong</th>
<th>$\rho_{\text{ambient}}$</th>
<th>$\rho_{\text{diffuse}}$</th>
<th>$\rho_{\text{specular}}$</th>
<th>$\rho_{\text{total}}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\phi_i = 60^\circ$</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$\phi_i = 25^\circ$</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$\phi_i = 0^\circ$</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Surface Illumination Calculation

- Single Light Source:

\[ I = I_E + K_A I_A + K_D (\vec{N} \cdot \vec{L}) I + K_S (\vec{V} \cdot \vec{R})^n I \]
Surface Illumination Calculation

- Multiple Light Sources:

\[ I = I_E + K_A I_A + \sum_{i=1}^{\text{# lights}} \left[ K_D (\vec{N} \cdot \vec{L}_i) + K_S (\vec{V} \cdot \vec{R}_i)^n \right] I_i \]
Materials in OpenGL

- The material properties of an object define how it interacts with light sources to produce a final color.
- Material properties are defined using:

  ```c
  glMaterial{fi}(GLenum face,GLenum pname,T param);
  glMaterial{fi}v(GLenum face,GLenum pname,T *params);
  ```

- Objects can have different materials for front and back facing polygons.
Materials properties in OpenGL

- GL_AMBIENT, GL_DIFFUSE, GL_AMBIENT_AND_DIFFUSE, and GL_SPECULAR are used to define how the material interacts with the equivalent components from the light source.
- GL_SHININESS controls the size of the specular highlight.
- GL_EMISSION controls how much light the object appears to emit on its own.
Color Material in OpenGL

- Normally when lighting is enabled, the primary color (specified by `glColor()`) is ignored.
- However, it can be convenient to change material colors using `glColor()` rather than calling `glMaterial()`. This is possible if you enable color material:

  ```
  glEnable(GL_COLOR_MATERIAL);
  ```

- Which material component (ambient, diffuse, ambient and diffuse, or specular) and which face (front, back, or both) affected by color material can be controlled with:

  ```
  glColorMaterial(GLenum face, GLenum mode);
  ```
Normals in OpenGL

- The current normal is set with:
  ```c
  void glNormal3{bsifd}(TYPE nx, TYPE ny, TYPE nz);
  void glNormal3{bsifd}v(const TYPE *v);
  ```

- Normals should be of unit length for correct results. If the modelview matrix might change the length of your normals, you can have OpenGL renormalize them using:
  ```c
  glEnable(GL_NORMALIZE);
  ```

- If you are scaling uniformly, a cheaper alternative to GL_NORMALIZE is:
  ```c
  glEnable(GL_RESCALE_NORMAL);
  ```
The Lighting Model in OpenGL

- The lighting model can be modified using:

  ```
  void glLightModel{if}(GLenum pname, TYPE param);
  void glLightModel{if}v(GLenum pname, const TYPE *param);
  ```

- Properties you can modify include:
  - `GL_LIGHT_MODEL_AMBIENT` – controls the global ambient light applied to all objects
  - `GL_LIGHT_MODEL_LOCAL_VIEWER` – controls whether the viewer is infinitely far away (cheaper) or located at the camera position (more accurate)
  - `GL_LIGHT_MODEL_TWO_SIZE` – controls whether lighting is calculated separately for front and back faces
  - `GL_LIGHT_MODEL_COLOR_CONTROL` – allows you to have OpenGL interpolate the specular color separately and apply it after texturing, to preserve highlights
Overview:

lighting-based models

- **Direct Illumination**
  - Emission at light sources
  - Scattering at surfaces

- **Global Illumination**
  - Shadows
  - Refractions
  - Inter-object reflections