Illumination Models

Local Illumination

Example

Local Illumination in the projective rendering pipeline

Local Illumination
- only models light arriving directly from light source
- interreflections and shadows
  - added through tricks, multiple rendering passes

Types of Models
- Simple, non-physical reflection models (Phong, Blinn)
- physically-based reflection models
  - BRDFs: Bidirectional Reflection Distribution Functions

Light Sources

Types of light sources
- `glLightfv(GL_LIGHT0, GL_POSITION, light[])`
- Directional/parallel lights
  - E.g. sun
  - Homogeneous vector
- (Homogeneous) point lights
  - Same intensity in all directions
- Spot lights
  - Limited set of directions:
    - Point+direction+cutoff angle
Local Illumination

Commonly used model (simple, non-physical)

Combine diffuse, specular, ambient
- E.g. OpenGL / graphics hardware:
  \[ I_{\text{out}}(x) = k_d \cdot I_d + k_f \cdot (I \cdot n) \cdot I_{\text{diff}} + k_s \cdot (h \cdot n)^g \cdot I_{\text{spec}} \]
  
  ambient diffuse specular

Materials

Ambient Light
- Incoming light component that is identical everywhere in the scene
- No direction
- Hack for replacing true global illumination (light bouncing off from other objects)
  \[ I_{\text{out}}(x) = k_a \cdot I_a \]

Diffuse component: Lambert’s Law
Johann Friedrich Lambert (1783):
- Power per unit area arriving at some object point \( x \) also depends on the angle of the surface to the light direction
  \[ \text{dA} \text{: differential surface area surrounding } x \]
  \[ I_{\text{out}}(x) = \cos(\angle(n, l)) \cdot I_a(x) = (n \cdot l) \cdot I_a(x) \]

Diffuse Component: a more detailed look
- Independent of viewing direction

Materials

Specular/Glossy
- Light is mostly reflected into the directions around the mirror direction \( n \) of \( l \)
Materials

Specular/Glossy
- Computing reflection direction $r_i$ of $I$
  - $n$ and $l$ are unit length!

\[
\frac{\mathbf{n}}{\mathbf{l}} = 2(\mathbf{n} \cdot \mathbf{r}_i)\mathbf{n} - \mathbf{r}_i
\]

Materials

Phong Model (Phong Bui-Tuong, 1975)
- Use cosine power as heuristic

\[
I_{\text{spec}}(x) = k_s \cdot (\mathbf{v} \cdot \mathbf{r}_i)^n \cdot I_n(x)
\]

Materials

Phong model
- Polar plot

\[n=10\]

\[n=100\]

Materials

Blinn-Phong model (Jim Blinn, 1977)
- Variation with better physical interpretation
  - $h$: halfway vector; $n$: shininess

\[
I_{\text{mat}}(x) = k_s \cdot \mathbf{h} \cdot \mathbf{n}^n \cdot I_n(x) \text{; with } h = (\mathbf{l} + \mathbf{v}) / 2
\]

Materials

Blinn-Phong Model
- Polar plot

\[r=0.1\]

\[r=0.01\]

Materials

Commonly used model (simple, non-physical)
- Combine diffuse, specular, ambient
  - E.g. OpenGL / graphics hardware:

\[
I_{\text{out}}(x) = k_d \cdot I_d + k_d \cdot (l \cdot n) \cdot I_{\text{diff}} + k_s \cdot (h \cdot n)^n \cdot I_{\text{spec}}
\]
Lighting in OpenGL

- **Light source**: amount of RGB light emitted
  - value represents percentage of full intensity, e.g., (1.0,0.5,0.5)
  - every light source emits ambient, diffuse, and specular light
- **Materials**: amount of RGB light reflected
  - value represents percentage reflected e.g., (0.0,1.0,0.5)

```c
glLightfv(GL_LIGHT0, GL_AMBIENT, amb_light_rgba );
glLightfv(GL_LIGHT0, GL_DIFFUSE, dif_light_rgba );
glLightfv(GL_LIGHT0, GL_SPECULAR, spec_light_rgba );
glLightfv(GL_LIGHT0, GL_POSITION, position);
gEnable(GL_LIGHT0);
gMaterialfv( GL_FRONT, GL_AMBIENT, ambient_rgba );
gMaterialfv( GL_FRONT, GL_DIFFUSE, diffuse_rgba );
gMaterialfv( GL_FRONT, GL_SPECULAR, specular_rgba );
gMaterialfv( GL_FRONT, GL_SHININESS, n );
```

Flat Shading

**Example:**

Gouraud Shading

**Example:**

Materials

**Summary**

- Very simple reflection models
- Fast (dot products & exponentiation)
- No physical justification
- Not very good for modeling real world

Light Sources

**Quadratic falloff**

- Brightness of objects depends on power per unit area that hits the object
- The power per unit area for a point or spot light decreases quadratically with distance
**Light Sources**

### Non-quadratic falloff:
- Many systems allow for other falloffs
- Allows for faking of the effect of area light sources
- OpenGL / graphics hardware:
  - $I_o$: intensity of light source
  - $x$: object point
  - $r$: distance of light from $x$
  - $I_{\text{in}}(x) = \frac{1}{ar^2 + br + c} \cdot I_0$

**Materials**

### Bi-directional Reflectance Distribution Function (BRDF):
- Describes fraction of light reflected for all combinations of incoming (light) and outgoing (viewing) directions
- Color channels (R, G, B) are treated separately
  - Actually: wavelengths (see later in course)

**Light Sources**

### Area lights:
- Light sources with a finite area
- More realistic model of many light sources
- Not available with projective rendering pipeline, (i.e., not available with OpenGL)

**Mach Bands**:
- Eye enhances discontinuity in first derivative
- Very disturbing, especially for highlights