Announcements

• HW 1 back

• HW 2 out
  – Due next Friday
  – Due date?
Pipeline overview

APPLICATION

COMMAND STREAM

VERTEX PROCESSING

TRANSFORMED GEOMETRY

RASTERIZATION

FRAGMENTS

FRAMEBUFFER IMAGE

you are here

3D transformations; shading

conversion of primitives to pixels

blending, compositing, shading

user sees this

DISPLAY

Result of minimal pipeline (no z test)
Result of z-buffer pipeline

Result of flat-shading pipeline
Result of Phong shading pipeline

How to achieve shading?
Shading

- Compute light reflected toward camera
- Inputs:
  - eye direction
  - light direction and light intensity (for each of many lights)
  - surface normal
  - surface parameters (color, shininess, …)

Light is scattered uniformly in all directions
- the surface color is the same for all viewing directions
- Lambert’s cosine law

Diffuse reflection

- Light is scattered uniformly in all directions
- the surface color is the same for all viewing directions
- Lambert’s cosine law

Top face of cube receives a certain amount of light
Top face of 60° rotated cube intercepts half the light
In general, light per unit area is proportional to \( \cos \theta = \mathbf{l} \cdot \mathbf{n} \)
Lambertian shading

• Shading independent of view direction

\[ L_d = k_d I \max(0, \n \cdot \l) \]

Lambertian shading

• Produces matte appearance
Diffuse shading

Light

- Local light
  - Position

- Directional light (e.g., sun)
  - Direction, no position
Specular shading (Phong)

- Intensity depends on view direction
  - bright near mirror configuration
  - measure “near” by dot product of unit vectors

\[ \cos(\alpha) = v \cdot r \]

\[
L_S = k_s I_{\text{max}}(0, \cos(\alpha))^n
\]

\[
\cos(\alpha) = v \cdot r
\]

\[
L_S = k_s I_{\text{max}}(0, v \cdot r)^n
\]
Reflected direction

- Intensity depends on view direction
  - reflects incident light from mirror direction

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

Specular shading (Blinn-Phong)

- Close to mirror \(\Leftrightarrow\) half vector near normal

\[ \mathbf{h} = \text{bisector}(\mathbf{v}, \mathbf{l}) \]
\[ = \frac{\mathbf{v} + \mathbf{l}}{\|\mathbf{v} + \mathbf{l}\|} \]
\[ L_s = k_s I \max(0, \cos \alpha)^n \]
\[ = k_s I \max(0, \mathbf{n} \cdot \mathbf{h})^n \]
Phong model—plots

- Increasing $n$ narrows the lobe

Fig. 16.9 Different values of $\cos^n \alpha$ used in the Phong illumination model.

Specular shading

$\kappa_s$  

$n$
Diffuse + Phong shading

Multiple lights

• Just loop over lights, add contributions
• Important to fill in black shadows

• Ambient shading
  – black shadows are not really right
  – one solution: dim light at camera
  – alternative: add a constant “ambient” color to the shading…
Ambient shading

- Shading that does not depend on anything
  - add constant color to account for disregarded illumination and fill in black shadows

\[ L_a = k_a I_a \]

Putting it together

- Usually include ambient, diffuse, Phong in one model

\[ L = L_a + L_d + L_s = k_a I_a + k_d I \max(0, \mathbf{n} \cdot \mathbf{l}) + k_s I \max(0, \mathbf{n} \cdot \mathbf{h})^n \]

- The final result is the sum over many lights

\[ L = L_a + \sum_{i=1}^{N} [(L_d)_i + (L_s)_i] \]

\[ L = k_a I_a + \sum_{i=1}^{N} [k_d I_i \max(0, \mathbf{n} \cdot \mathbf{l}_i) + k_s I_i \max(0, \mathbf{n} \cdot \mathbf{h}_i)^n] \]