Lecture 8:
Shading 1
(Illumination and Blinn-Phong Model)
Last Lecture

• Antialiasing
  - Sampling theory
  - Antialiasing in practice

• Visibility / occlusion
  - Z-buffering
Today

- Shading
  - Illumination & Shading
  - Blinn-Phong reflectance model
    - Specular and ambient terms
  - Shading frequencies
  - Graphics Pipeline
What We’ve Covered So Far

Position objects and the camera in the world

Compute position of objects relative to the camera

Project objects onto the screen

Sample triangle coverage
Rotating Cubes (Now You Can Do)
Rotating Cubes (Expected)
What Else Are We Missing?

Credit: Bertrand Benoit. “Sweet Feast,” 2009. [Blender /VRay]
Shading
Shading: Definition

In Merriam-Webster Dictionary

shading, [ˈʃeɪdɪŋ], noun
The darkening or coloring of an illustration or diagram with parallel lines or a block of color.

In this course

The process of applying a material to an object.
A Simple Shading Model
(Blinn-Phong Reflectance Model)
Perceptual Observations

- Specular highlights
- Diffuse reflection
- Ambient lighting

Photo credit: Jessica Andrews, flickr
Shading is Local

Compute light reflected toward camera at a specific shading point

Inputs:
- Viewer direction, \( v \)
- Surface normal, \( n \)
- Light direction, \( l \) (for each of many lights)
- Surface parameters (color, shininess, …)
Shading is Local

No shadows will be generated! (shading ≠ shadow)
Diffuse Reflection

- Light is scattered uniformly in all directions
  - Surface color is the same for all viewing directions
Diffuse Reflection

• But how much light (energy) is received?
  - 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 $\cos \theta = l \cdot n$
Light Falloff

intensity here: \( I \)

\( r \)

intensity here: \( I/r^2 \)
Lambertian (Diffuse) Shading

Shading independent of view direction

Energy arrived at the shading point

\[ L_d = k_d \left( \frac{I}{r^2} \right) \max(0, n \cdot l) \]

Energy received by the shading point

diffusely reflected light

diffuse coefficient (color)

Max function ensures only non-negative values

Distance between light and shading point

\( r \)

Unit vectors

\( n \) and \( l \)

\( \theta \)
Lambertian (Diffuse) Shading

Produces diffuse appearance

$k_d$
Specular Term (Blinn-Phong)

Intensity **depends** on view direction

- Bright near mirror reflection direction
Specular Term (Blinn-Phong)

V close to mirror direction ⇔ **half vector** near normal

- Measure “near” by dot product of unit vectors

\[ h = \text{bisector}(v, l) \]
\[ = \frac{v + l}{\|v + l\|} \]

\[ L_s = k_s \left( \frac{I}{r^2} \right) \max(0, \cos \alpha)^p \]
\[ = k_s \left( \frac{I}{r^2} \right) \max(0, n \cdot h)^p \]
Cosine Power Plots

Increasing $p$ narrows the reflection lobe
Specular Term (Blinn-Phong)

Blinn-Phong

\[ L_s = k_s \left( \frac{I}{r^2} \right) \max(0, n \cdot h)^p \]

Note: showing \( L_d + L_s \) together

[ Foley et al. ]
 Ambient Term

Shading that does not depend on anything
  • Add constant color to account for disregarded illumination and fill in black shadows
  • This is approximate / fake!

\[ L_a = k_a I_a \]

- ambient coefficient
- reflected ambient light
Blinn-Phong Reflection Model

\[
L = L_a + L_d + L_s \\
= k_a I_a + k_d \left( \frac{I}{r^2} \right) \max(0, \mathbf{n} \cdot \mathbf{l}) + k_s \left( \frac{I}{r^2} \right) \max(0, \mathbf{n} \cdot \mathbf{h})^p
\]
Shading Frequencies
Shading Frequencies

What caused the shading difference?
Shade each triangle (flat shading)

**Flat shading**

- Triangle face is flat — one normal vector
- Not good for smooth surfaces
Shade each vertex (Gouraud shading)

**Gouraud shading**
- Interpolate colors from vertices across triangle
- Each vertex has a normal vector (how?)
Shade each pixel (Phong shading)

**Phong shading**

- Interpolate normal vectors across each triangle
- Compute full shading model at each pixel
- Not the Blinn-Phong Reflectance Model
Shading Frequency: Face, Vertex or Pixel

Shading freq. : Face  Vertex  Pixel
Shading type : Flat  Gouraud  Phong

Defining Per-Vertex Normal Vectors

Best to get vertex normals from the underlying geometry

- e.g. consider a sphere

Otherwise have to infer vertex normals from triangle faces

- Simple scheme: average surrounding face normals

\[ N_v = \frac{\sum_i N_i}{\| \sum_i N_i \|} \]
Defining Per-Pixel Normal Vectors

Barycentric interpolation (introducing soon) of vertex normals

Don’t forget to normalize the interpolated directions
Thank you!