Lighting and Shading wrapup

Texture Mapping

Textbook Appendix A4, Chapter 15

Today

- Lighting and Shading, odds and ends
  - Phong shading vs. Gouraud shading
  - Phong reflection/illumination vs. Phong shading
  - Global illumination and ambient
  - Blinn-Phong reflection and the halfway vector
  - Color in OpenGL
  - Toon shading
- Texture mapping introduction
Lighting and Shading
odds and ends

- Phong shading vs. Gouraud shading
  - Gouraud == per-vertex normals and illumination. Interpolate vertex colors to fragments
  - Phong == Interpolate vertex normals, per-fragment illumination

- Phong reflection vs. Phong shading
  - P. reflection == an approximation of BRDF, into specular + diffuse + …

- Global illumination and ambient
  - Ambient term is a crude approximation of global illumination

Global Illumination
Cornell box

http://en.wikipedia.org/wiki/Cornell_box
C³ Review: Normal Transform

- When is the normal matrix the same as the model-view matrix?
  a) When the model-view matrix is a pure translation
  b) When the model-view matrix is a pure rotation
  c) When the model-view matrix is a scaling along only one axis
  d) (a) and (b)
  e) (a) and (c)
C³ Review: Normal Transform

- To transform the normal, we need to pass a normal mapping matrix normalMatrix to the vertex shader. Which of the following creates the correct normalMatrix?

\[ \text{mv} = V \times M \]
\[ \text{mvp} = P \times V \times M \]

a) \[ \text{normalMatrix} = \text{transpose}(\text{mv}) \]

b) \[ \text{normalMatrix} = \text{transpose}(\text{inverse}(\text{mv})) \]

c) \[ \text{normalMatrix} = \text{transpose}(\text{inverse}(\text{mvp})) \]

d) \[ \text{normalMatrix} = \text{transpose}(\text{mvp}) \]

Lighting and Shading
odds and ends

- Switch to pen for
  - Blinn-Phong reflection and the halfway vector
  - Color in OpenGL
  - Toon shading
- Half-way vector (Blinn-Phong Reflection)

This angle is actually $\phi$, not $\theta$ inside a plane.

$\vec{h}$ is half-way vector (between $\vec{l}$ and $\vec{n}$)

Blinn:

$$(\vec{l} \cdot \vec{n})$$

$\vec{h}$ is computed easily: normal $(\vec{l} + \vec{n})$

($\vec{l}$, $\vec{n}$ are unit vectors)

This produces a similar effect, but need a larger $\omega$ (shininess) exponent.

§ Color in OpenGL

Red
Green
Blue
Alpha - need to represent transparency, compositing
§ Toon Shading (Assignment 3)

Similar to computing the diffuse ($\alpha$ ambient)

$\begin{array}{|c|c|}
\hline
\text{color} & \text{output} \\
\hline
1 & 1 \\
0.75 & 0.75 \\
0.5 & 0.5 \\
0.25 & 0.25 \\
0 & 0 \\
\hline
\end{array}$

intensity

computed by your shading
Texture Mapping

An example scene from Pixar’s Bolt

Figure 12: A real production still from “Bolt” using Ptex for all models. © Disney/Pixar Animation Studios

http://ptex.us/ptexpaper.html
Texture Mapping in Doom ca. 1993!

http://en.wikipedia.org/wiki/Doom_%28video_game%29

Example: Pixar’s ptex

Source: http://ptex.us/ptexpaper.html
Another Example

Source: (result of random web search)

First example
How to model the earth?