Now Playing:

Multiply
Jamie Lidell
from *Multiply*
Released June 13, 2005

Guinness 'Noitulove'

Danny Kleinman
Framestore CFC SIGGRAPH 2006

Available online:
http://www.metacafe.com/watch/35459/guinness_evolution/

Lighting and Shading in OpenGL / Artistic Shading

Rick Skarbez, Instructor
COMP 575
September 25, 2007

Announcements

- Programming Assignment 1 is due Thursday at 11:59pm

Submitting Programs

- Upload source and executable(s) (Windows or Mac) to digital dropbox on Blackboard
  - blackboard.unc.edu
- Include a document that lists
  - What optional components you did
  - Instructions for use
  - Any problems that you had, or components that do not work properly
- Please submit as a zip file with your name in the filename

Last Time

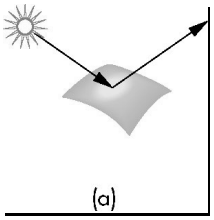
- Began our discussion of lighting and shading
- Discussed some of the simplifications of the lighting model made in OpenGL

Today

- Review lighting and shading
- Talk about how to implement lighting in OpenGL
- Demo of rendering with different lights and shading in OpenGL
- Discussion of artistic/non-photorealistic shading techniques

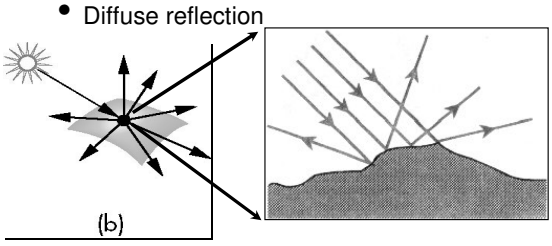
Light and Matter

- Specular reflection



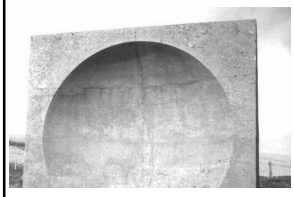
Light and Matter

- Diffuse reflection



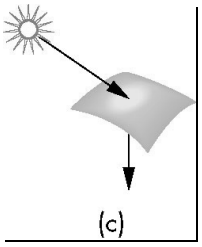
Light and Matter

- Diffuse Reflection



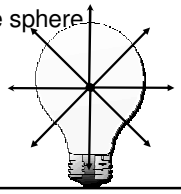
Light and Matter

- Translucency / Transparer



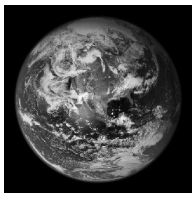
Point Lights

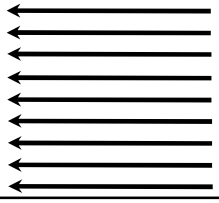
- Emit light evenly in all directions
  - That is, photons (rays) from a point light all originate from the same point, and have directions evenly distributed over the sphere



Directional Lights

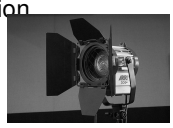
- Point light sources at an infinite (or near infinite) distance
  - Can assume that all rays are parallel

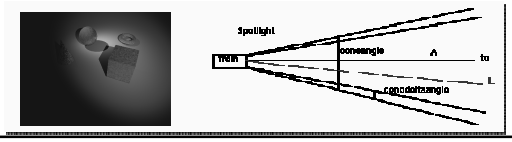




Spot Lights

- Similar to point lights, but intensity of emitted light varies by direction
  - Can think of it as only emitting rays over a patch on the sphere





Phong Reflection Model

- A simplification of the rendering equation
- Divided into 3 parts
  - Ambient
  - Diffuse
  - Specular
- The sum of these components describes the color at a point

Phong Reflection Model

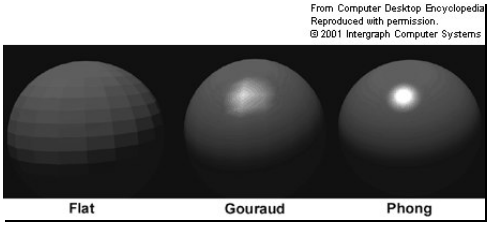
Ambient Diffuse Specular

- $I = I_a(R_a, L_a) + I_d(\mathbf{n}, \mathbf{l}, R_d, L_d, a, b, c, d) + I_s(\mathbf{r}, \mathbf{v}, R_s, L_s, n, a, b, c, d)$
- $R_{something}$ represents how reflective the surface is
- $L_{something}$ represents the intensity of the light
- In practice, these are each 3-vectors
- One each for R, G, and B

Types of Shading

- There are several well-known / commonly-used shading methods
  - Flat shading
  - Gouraud shading
  - Phong shading

Shading Comparison



Done with Review

- Different types of light interaction
  - Ambient (pretends to be indirect)
  - Diffuse
  - Specular
- Different types of lights
  - Point
  - Spot
  - Directional
- So how do we use lights in OpenGL?

Adding Lighting to Your Scene

- These are the steps required to add lighting to a scene:
  1. Define normals for all vertices of all objects
  2. Create, select, and position one or more lights
  3. Create and select a *lighting model*
  4. Define material properties of the objects in the scene

Defining Normals

- We discussed a method for computing normals last lecture
- Can assign your own normals by using glNormal3{bsidf}{v}
  - The normal is applied to all subsequent glVertex calls, until a new glNormal is assigned
  - Need to compute the normals yourself
- Helper functions light glutSolidSphere() define normals for you

Setting up Lights

- glLight{if}{v}(GLenum *light*, GLenum *pname*, *TYPE param*)
  - Creates the light named *light*, which can be GL_LIGHT0, GL_LIGHT1, ..., GL_LIGHT7
    - The API only guarantees support for 8 lights
  - Sets the parameter *pname* to the value *param*

glLight Parameters

- GL_AMBIENT (L_a in the Phong model)
- GL_DIFFUSE (L_d)
- GL_SPECULAR (L_s)
  - These are all colors (vectors of 4 floats)
- GL_POSITION
  - 3D osition of the light (vector of 4 floats)

glLight Parameters

- GL_CONSTANT_ATTENUATION (a)
- GL_LINEAR_ATTENUATION (b)
- GL_QUADRATIC_ATTENUATION (c)
  - These are all single floating point values

glLight Parameters

- GL_SPOT_DIRECTION
  - Direction (for light used as a spotlight)
  - Vector of 3 floats
- GL_SPOT_EXPONENT
  - Controls how “focused” the spot is
- GL_SPOT_CUTOFF
  - Controls the angle of the spotlight

Selecting a Lighting Model

- An OpenGL lighting model has 4 components
  - Global ambient light intensity
  - Whether the viewer is local or infinitely far away
  - Whether to light backfaces
  - When to apply specular color

Setting up the Lighting Model

- glLightModel{if}{v}(GLenum *pname*, *TYPE param*)
  - Sets the specified lighting model parameter to the specified value

glLightModel Parameters

- GL_LIGHT_MODEL_AMBIENT
  - The ambient RGBA intensity of the entire scene
  - Defaults to (0.2, 0.2, 0.2, 1.0)
    - There is always *some* light
- GL_LIGHT_MODEL_TWO_SIDE
  - Whether to compute lighting for back faces of polygons
  - GL_TRUE or GL_FALSE

glLightModel Parameters

- GL_LIGHT_MODEL_LOCAL_VIEWER
  - Whether to consider the viewer local to the scene
    - Why would / wouldn’t we want to do this?
  - GL_TRUE or GL_FALSE

glLightModel Parameters

- GL_LIGHT_MODEL_COLOR_CONTROL
  - Whether to apply specular highlights before or after texturing
  - GL_SINGLE_COLOR or GL_SEPARATE_SPECULAR_COLOR

Turning on the Lights

- Need to enable lighting
  - glEnable(GL_LIGHTING);
- Need to enable the lights you've set up
  - glEnable(GL_LIGHT0);
  - glEnable(GL_LIGHT1);
  - ...

Defining Material Properties

- glMaterial{if}{v}(GLenum *face*, GLenum *pname*, GLenum *param*)
  - Sets the parameter *pname* to the value *param* for the face(s) specified by *face*
  - *face* can be GL_FRONT, GL_BACK, or GL_FRONT_AND_BACK

glMaterial Parameters

- GL_AMBIENT (R_a in the Phong model)
- GL_DIFFUSE (R_d)
- GL_SPECULAR (R_s)
  - All RGBA colors
- GL_AMBIENT_AND_DIFFUSE
  - Used to set ambient and diffuse to the same color (just for convenience)

glMaterial Parameters

- GL_SHININESS (n in the Phong model)
  - Exponent on the specular lighting component
- GL_EMISSION
  - RGBA color of the materials "emitted" light
    - Doesn't actually emit light
    - Makes the material appear to glow

Choosing a Shading Model

- OpenGL has flat and Gouraud shading models built in
- glShadeModel(GLenum *mode*)
  - *mode* can be either GL_FLAT or GL_SMOOTH (for Gouraud shading)

Nate Robins' OpenGL Demos

- Many of the demos I showed today were modified versions of Nate Robins' GL Demos
  - Download from http://www.xmission.com/~nate/tutors.html

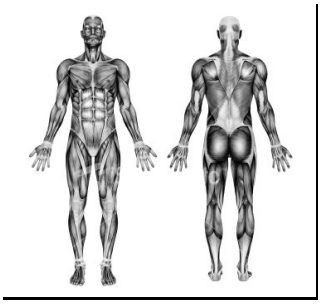
Summing up OpenGL Lighting and Shading

- Uses the Phong lighting model to compute color at vertices
- Can use flat or Gouraud shading
- Any questions about lighting and shading in OpenGL?

Realism in Shading

- We've discussed flat, Gouraud, and Phong shading
  - These all attempt to imitate the appearance of objects in the real world
- This may not be desirable for all applications
  - Consider biological drawings

Example



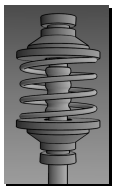
Non-Photorealistic Rendering

- Non-photorealistic rendering (NPR) is a broad term for rendering techniques that do not attempt to simulate the real world
  - Also referred to as artistic rendering or artistic shading

Why NPR?

- Can be easier for a user to understand
  - *i.e.* Enhancement of edge lines and highlights for technical illustrations

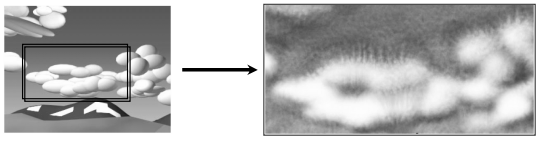
Phong Shading



NPR

Why NPR?

- Can simulate artistic media
  - *i.e.* Create an image that looks like watercolor paint on canvas



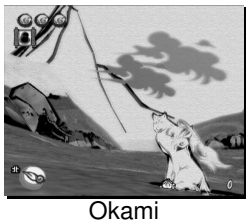
Why NPR?

- Can display the data that underlies the model
  - *i.e.* scientific visualization

Why NPR?

- Because it looks cool
  - *i.e.* movies, video games, etc.

Viewtiful Joe



Okami

NPR on Graphics Hardware

- Some of these techniques can now be used in real-time on modern graphics cards
  - esp. Toon shading
- Doing so usually takes advantage of programmable shaders and texturing
  - I may revisit toon shading as an example later in the semester

Next Time

- Moving on to vertex processing
  - Projection
  - Clipping
- Reminder: Programming assignment 1 due Thursday by 11:59pm
  - Upload to blackboard.unc.edu