Shader Lighting and Textures

CSCI 4830/7000
Advanced Computer Graphics
Spring 2009

Shader Lighting

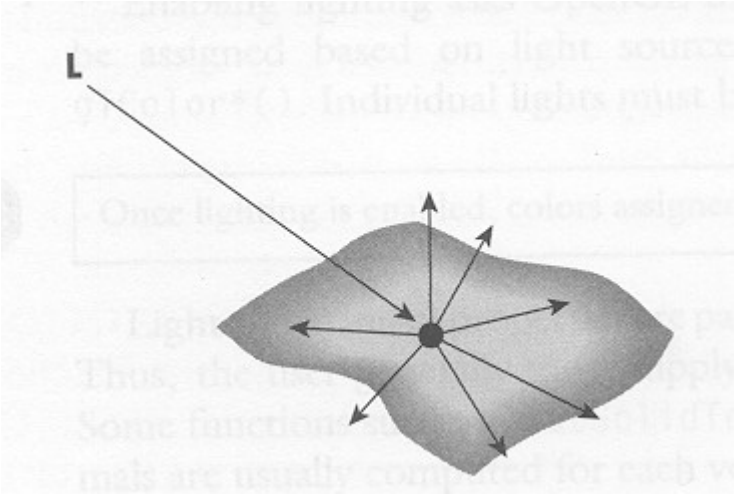
- Ultimate flexibility
  - Lighting method
    - Phong reflection
    - Blinn-Phong reflection
  - Lighting
    - Per vertex with Gouraud shading
    - Per pixel lighting
  - Special effects
    - High Dynamic Range lighting
- Ultimate responsibility
  - Nothing happens automatically

OpenGL Lighting Components

- $C = M_E + M_A(C_A + C_G) + (N \cdot L)M_D C_D + (N \cdot H)^S M_S C_S$
- C_X are light components
- M_X are material components
- Components
  - Emission
  - Ambient (also Global Ambient)
  - Diffuse
  - Specular
- Calculated for each light, vertex, RGBA
- Assumes values in the range 0-1

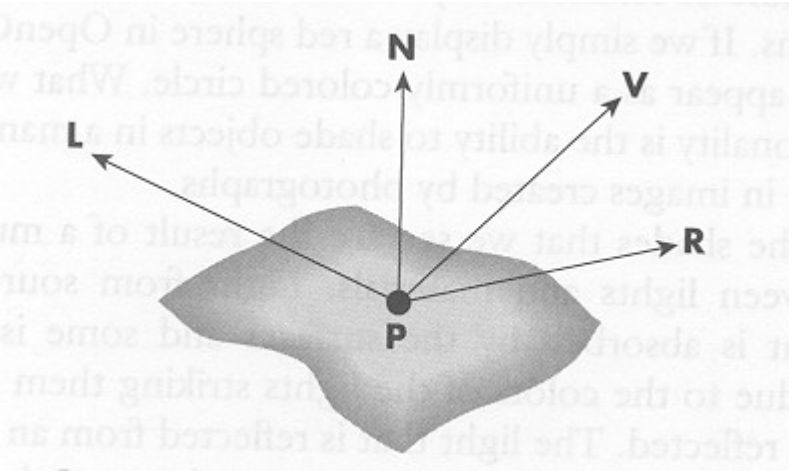
Diffuse Reflections

- Diffuse light scatters in all directions
  - Lambertian reflection
- Intensity depends on cosine of the angle of incidence
- Intensity $(N \cdot L) M C_D$



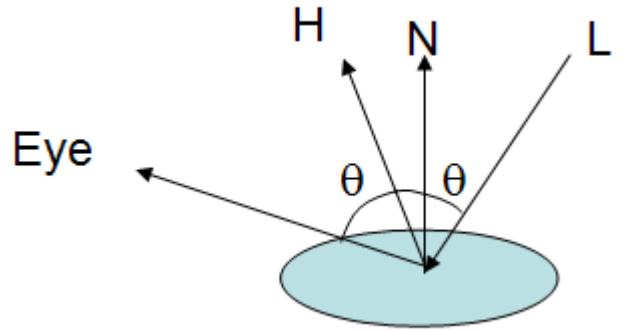
Phong Reflection Model

- *L* light source
- *N* normal vector for surface
- *R* reflected light
  - $R = 2(L \bullet N)N - L$
- *V* viewer (eye)
- Intensity $(V \bullet R)^S MC$
  - *S* shininess
  - *M* material reflection coefficient
  - *C* color if light source
- Calculated independently for R,G,B



Blinn-Phong Reflection Model

- Also called modified Phong or Fast Phong
- Simpler and faster
- Half angle $H = L+V$ (renormalize)
- Intensity $(N \bullet H)^S MC$



Per Vertex Lighting

- Calculate lighting at vertex
- Linearly interpolate across polygon
  - This is often called Gouraud shading
  - Real Gouraud shading averages normals at vertexes and then interpolates
- Effort proportional to number of vertexes
- May miss important effects for large polygons

Per pixel lighting

- Calculate lighting at pixel
- Calculate ambient and emission by vertex
  - Set L,P,V,H for use in frag shader
- Calculate diffuse and specular by pixel
- Effort proportional to number of pixels

Shader Textures

- Pointer to texture
  - sampler2D *name*;
  - Points to current texture from glBindTexture()
- Extract pixel at vec2 texture coordinate *pos*
  - texture2D(*name*,*pos*);
- Different sampler/function for 1D,2D,LOD,...
- Returns vec4 (RGBA)