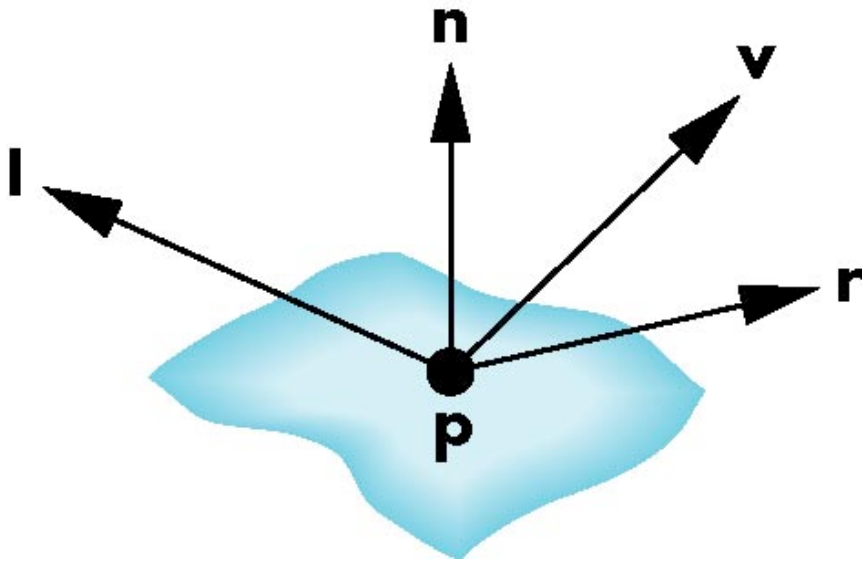
Supplement to Lecture 14

Illumination Model in OpenGL

CS 354 Computer Graphics
http://www.cs.utexas.edu/~bajaj/
Department of Computer Science
University of Texas at Austin

Phong Illumination

- A simple model that can be computed rapidly
- Has three components
  - Diffuse
  - Specular
  - Ambient
- Uses four vectors
  - To source
  - To viewer
  - Normal
  - Perfect reflector



Ambient Light

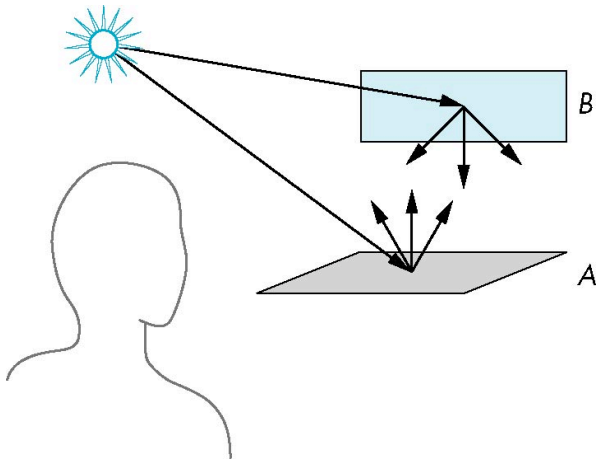
- Ambient light is the result of multiple interactions between (large) light sources and the objects in the environment
- Amount and color depend on both the color of the light(s) and the material properties of the object
- Add $k_a I_a$ to diffuse and specular terms

reflection coef

intensity of ambient light

Distance Terms

- The light from a point source that reaches a surface is inversely proportional to the square of the distance between them
- We can add a factor of the form $1/(ad + bd + cd^2)$ to the diffuse and specular terms



- The constant and linear terms soften the effect of the point source

Light Sources

- In the Phong Model, we add the results from each light source
- Each light source has separate diffuse, specular, and ambient terms to allow for maximum flexibility even though this form does not have a physical justification
- Separate red, green and blue components
- Hence, 9 coefficients for each point source
  - I_{dr}, I_{dg}, I_{db}, I_{sr}, I_{sg}, I_{sb}, I_{ar}, I_{ag}, I_{ab}

Material Properties

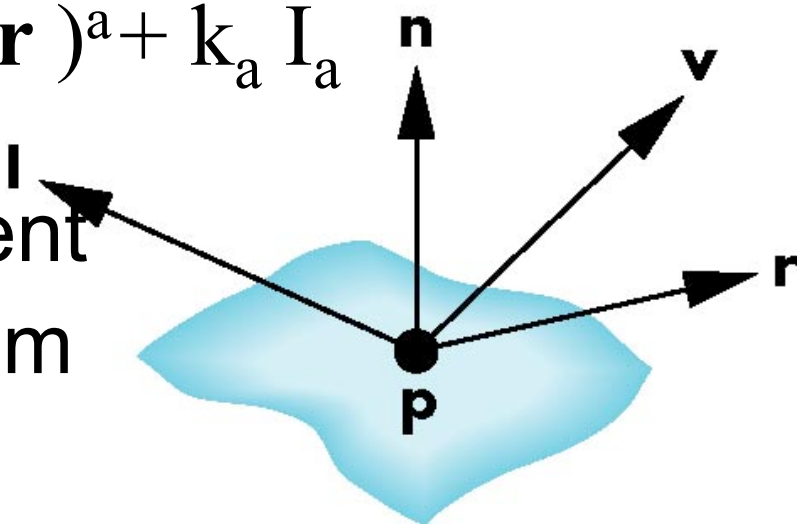
- Material properties match light source properties
  - Nine absorbtion coefficients
    - k_{dr}, k_{dg}, k_{db}, k_{sr}, k_{sg}, k_{sb}, k_{ar}, k_{ag}, k_{ab}
  - Shininess coefficient a

Adding up the Components

For each light source and each color component, the Phong model can be written (without the distance terms) as

$$I = k_d I_d \, \mathbf{l} \cdot \mathbf{n} + k_s I_s (\mathbf{v} \cdot \mathbf{r})^a + k_a I_a$$

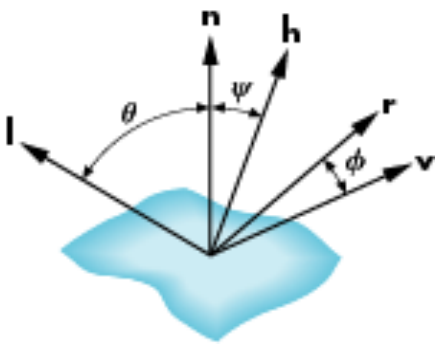
For each color component we add contributions from all sources



Modified Phong Model

- The specular term in the Phong model is problematic because it requires the calculation of a new reflection vector and view vector for each vertex
- An approximation using the halfway vector that is more efficient
  - **h** is normalized vector halfway between **l** and **v**

$$\mathbf{h} = (\mathbf{l} + \mathbf{v}) / |\mathbf{l} + \mathbf{v}|$$



Using the Half-Way Vector

- Replace $(\mathbf{v} \cdot \mathbf{r})^a$ by $(\mathbf{n} \cdot \mathbf{h})^b$
- b is chosen to match shineness
- Note that halway angle is half of angle between $\mathbf{r}$ and $\mathbf{v}$ if vectors are coplanar
- Resulting model is known as the modified Phong lighting model
  - Specified in OpenGL standard

Computation of Vectors

- **l** and **v** are specified by the application
- Can computer **r** from **l** and **n**
- Problem is determining **n**
- For simple surfaces is can be determined but how we determine **n** differs depending on underlying representation of surface
- OpenGL leaves determination of normal to application
  - Exception for GLU quadrics and Bezier surfaces