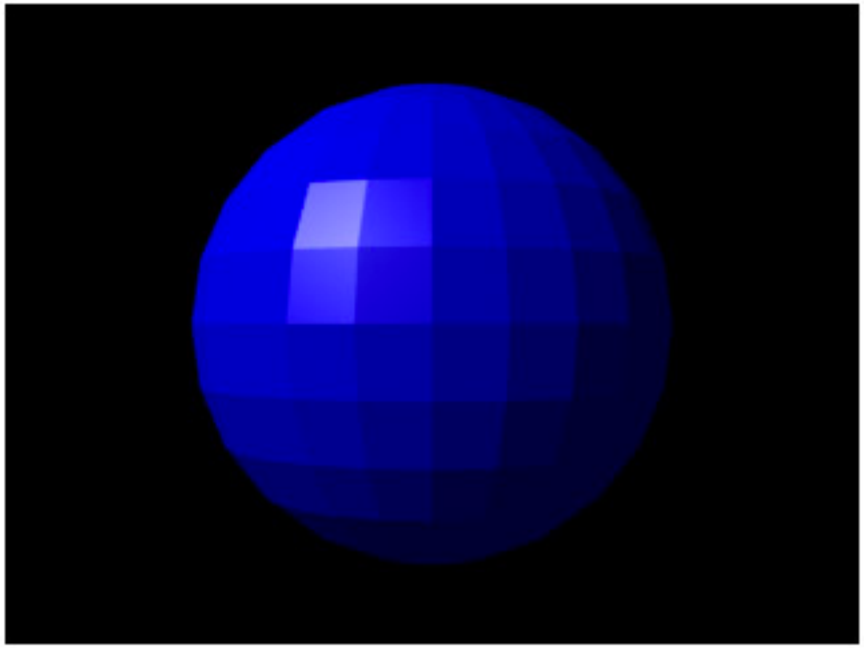
CS 418: Interactive Computer Graphics

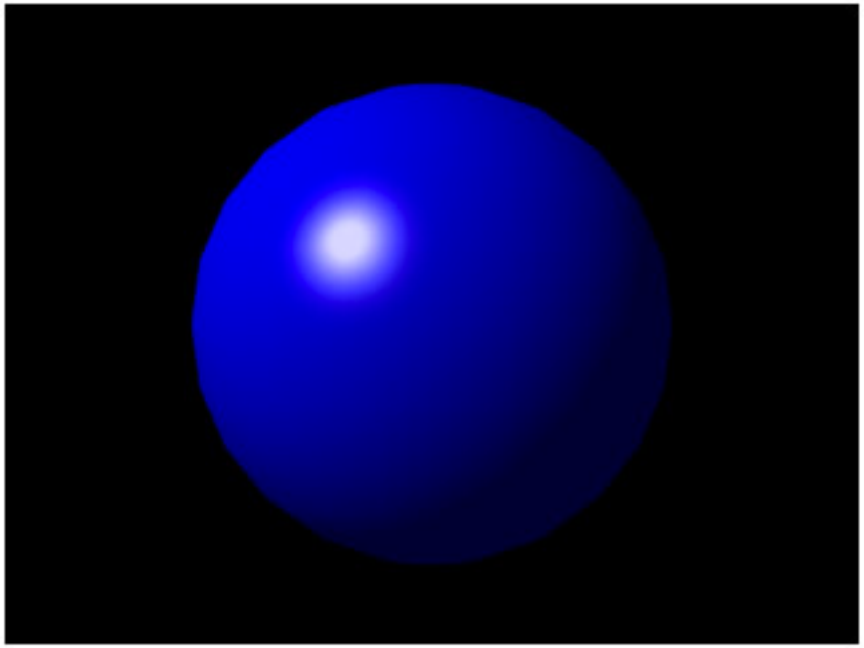
Basic Shading

Eric Shaffer

Shading

Shading refers to the process of determining the color for a pixel (or vertex…or polygon) during the rendering process

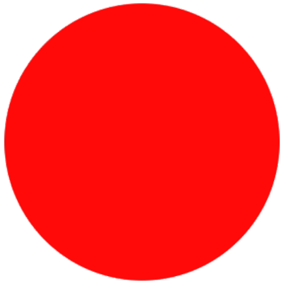




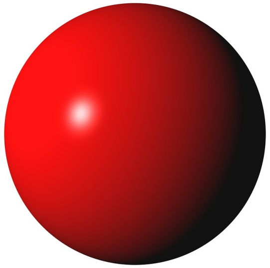
What is the difference between the two images?

Why we need shading

- Suppose we build a model of a sphere using many polygons and color it with a single color as we have done so far. We get something like

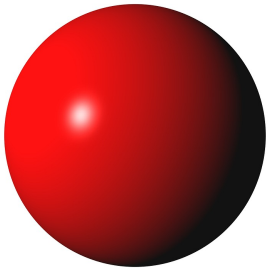


- But we want



Shading

- Why does the image of a real sphere look like



- Light-material interactions cause each point to have a different color or shade
- Need to consider
  - Light sources
  - Material properties
  - Location of viewer
  - Surface orientation

Shading

We will create a ***simple*** mathematical model for shading

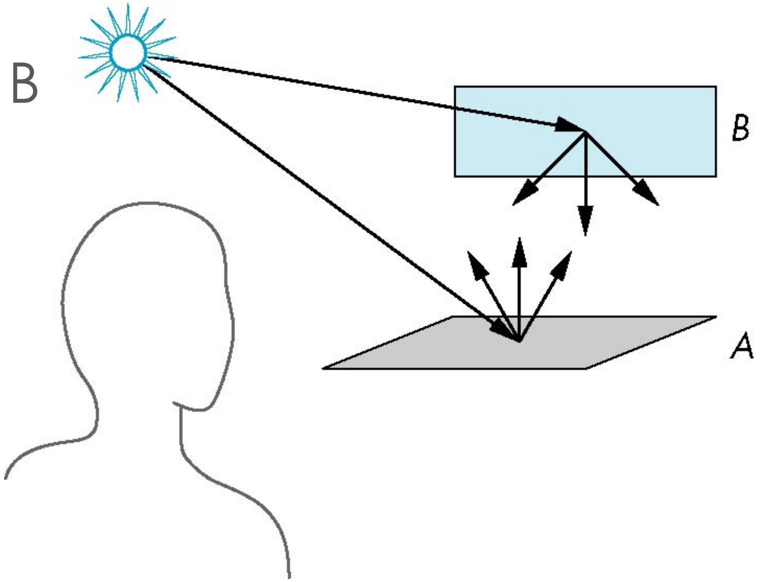
It will be function that takes in:

- Light sources
- Material properties
- Location of viewer
- Surface orientation

And returns a color….

Scattering

- Light strikes A
  - Some scattered
  - Some absorbed
- Some of scattered light strikes B
  - Some scattered
  - Some absorbed
- Some of this scattered light strikes A and so on



Rendering Equation

- Scattering and absorption of light described by the *rendering equation*
  - Scattering and absorption are infinite phenomena
  - Cannot be solved in general analytically
  - Describes global lighting effects
- CS 419 discusses solutions to the rendering equation
  - Ray-tracing, path tracing, etc.

$$\underbrace{L_r(x,\omega_r)}_{\text{Light reflected towards eye}} = \int_{H^2} \underbrace{f_r(x,\omega_i \to \omega_r)}_{\text{Surface reflectance}} \underbrace{L_i(x,\omega_i)}_{\text{Incoming light}} \underbrace{\cos\theta_i'}_{\text{Angle weighting}} \, d\omega_i$$

Global Effects

Local vs Global Illumintaion

- Correct shading requires a global calculation involving all objects and light sources
  - Incompatible with pipeline model which shades each polygon independently (local rendering)
  - Why do we want to shade each polygon independently?

- However, in computer graphics, especially real time graphics, we are happy if things "look right"
  - There exist many techniques for approximating global effects
  - Can anyone guess what you need to do to approximate global illumination in rasterization?

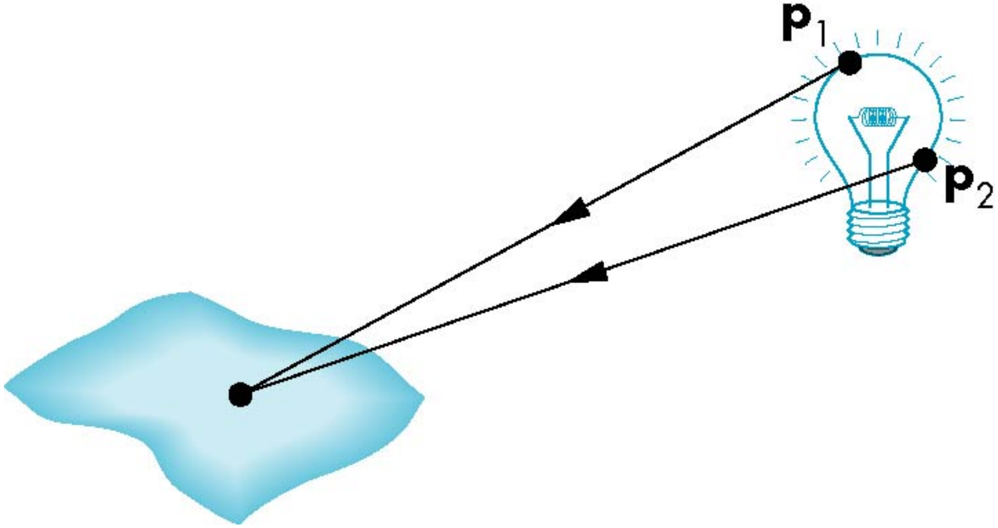
Light-Material Interaction

- Light that strikes an object is
  - partially absorbed and
  - partially scattered (reflected)
- The amount reflected determines
  - color of the object
  - brightness of the object
  - A surface appears red under white light because the red component of the light is reflected and the rest is absorbed
- How the reflected light is scattered depends on
  - the smoothness and orientation of the surface

Light Sources

General light sources are complex to model

Would need to integrate light coming from all points on the source

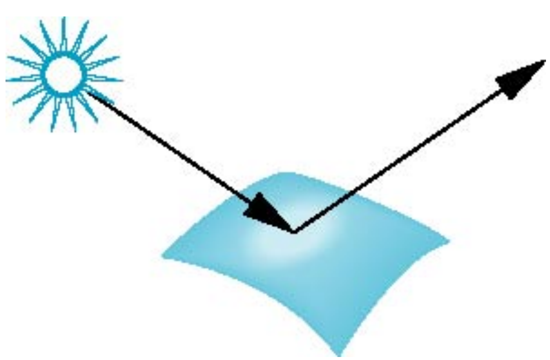


Simple Light Source Models

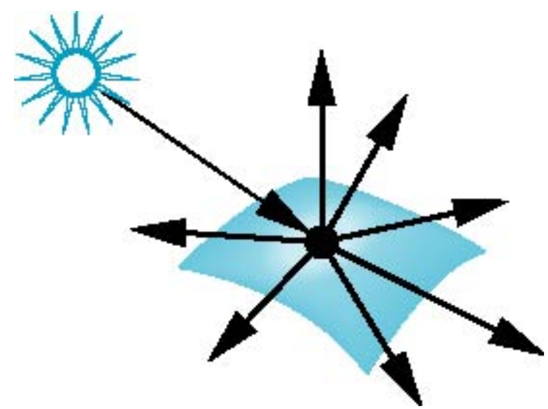
- Point source
  - Model with position and color
- Directional source
  - Distant source = infinite distance away (parallel)
- Ambient light
  - Same amount of light everywhere in scene
  - Can model contribution of many sources and reflecting surfaces

Surface Types

- Consider light traveling along a specific ray
- The smoother a surface, the more reflected light is concentrated in a single direction
  - Perfect mirror reflects perfectly in a single direction
- A very rough surface scatters light in all directions



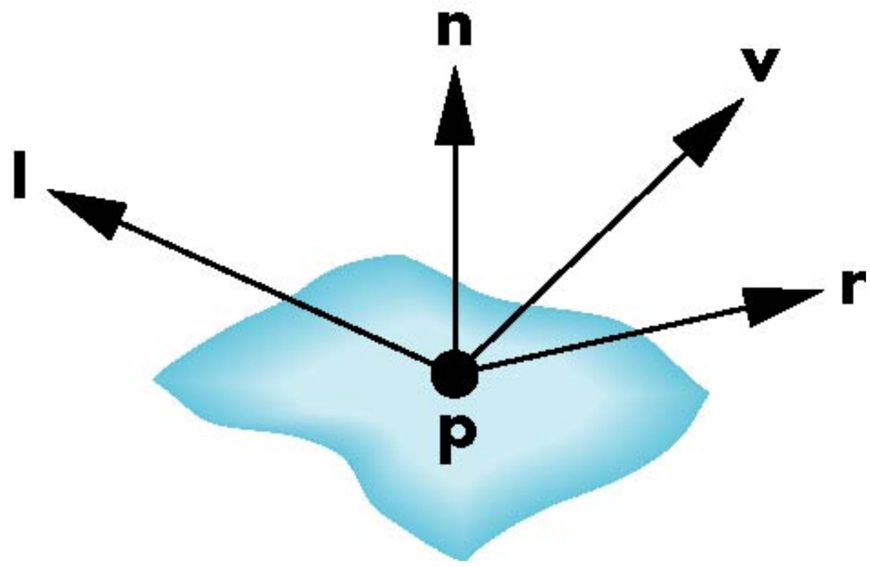
smooth surface



rough surface

The Phong Shading Model

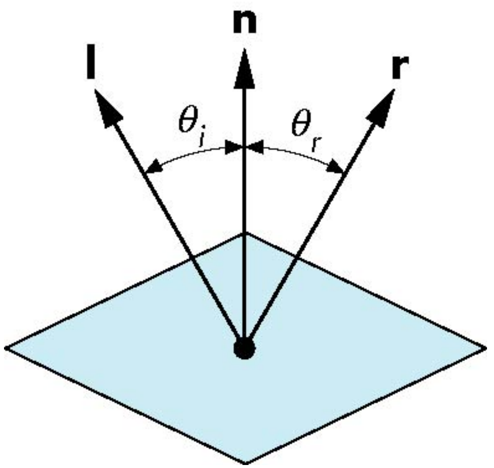
- A simple model that can be computed rapidly
- Has three components
  - Diffuse
  - Specular
  - Ambient
- Uses four vectors
  - To source
  - To viewer
  - Normal
  - Perfect reflector



Modleing a Ideal Reflector

- Incoming light ray is reflected in a single
- Normal is determined by local orientation
- Given the direction of incoming light…we can find r
  - l and n are unit vectors
  - Angle of incidence = angle of reflection
  - The three vectors must be coplanar
- What is another name for an ideal reflector?

$$\mathbf{r} = 2\,(\mathbf{l} \cdot \mathbf{n}\,)\,\mathbf{n} - \mathbf{l}$$

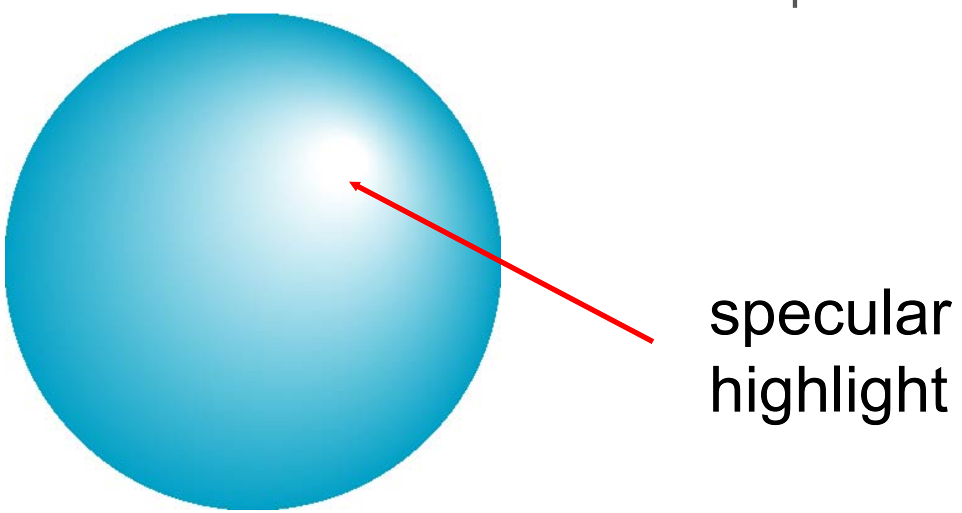


Modeling a Lambertian Surface

- Perfectly diffuse reflector
- Light scattered equally in all directions
- Amount of light reflected is affected by the angle of incidence
  - reflected light proportional to cos q_i
  - cos q_i = **l** · **n** if vectors normalized
  - There are also three coefficients, k_r, k_b, k_g that specify how much of each color component is reflected

Specular Surfaces

- Most surfaces are neither ideal diffusers nor perfectly specular (ideal reflectors)
- Smooth surfaces show specular highlights due to incoming light being reflected in directions concentrated close to the direction of a perfect reflection



Modeling Specular Relections

- Bui Tuong Phong [1973] proposed using a term that dropped off as the angle between the viewer and the ideal reflection increased

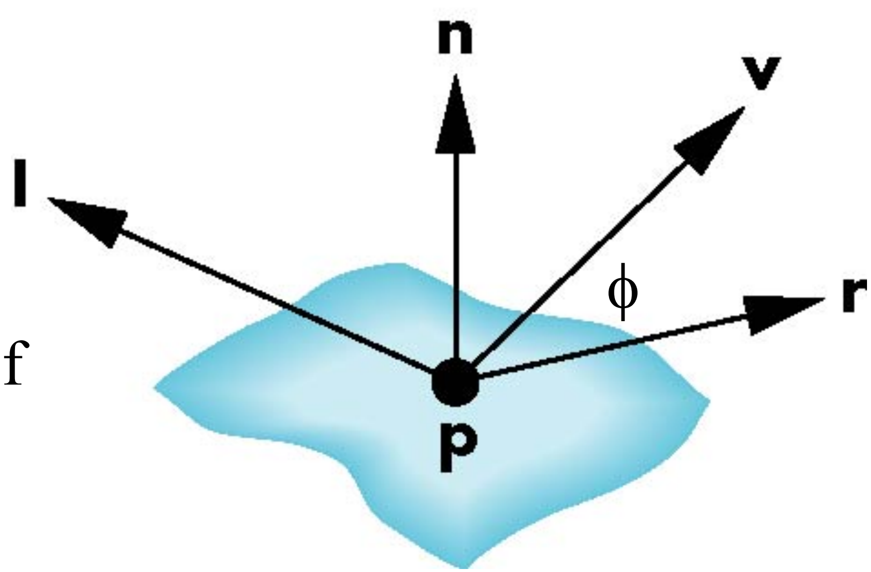
$$\mathbf{I}_r \sim k_s\, \mathbf{I} \cos^{\alpha}\phi$$

reflected intensity

absorption coef

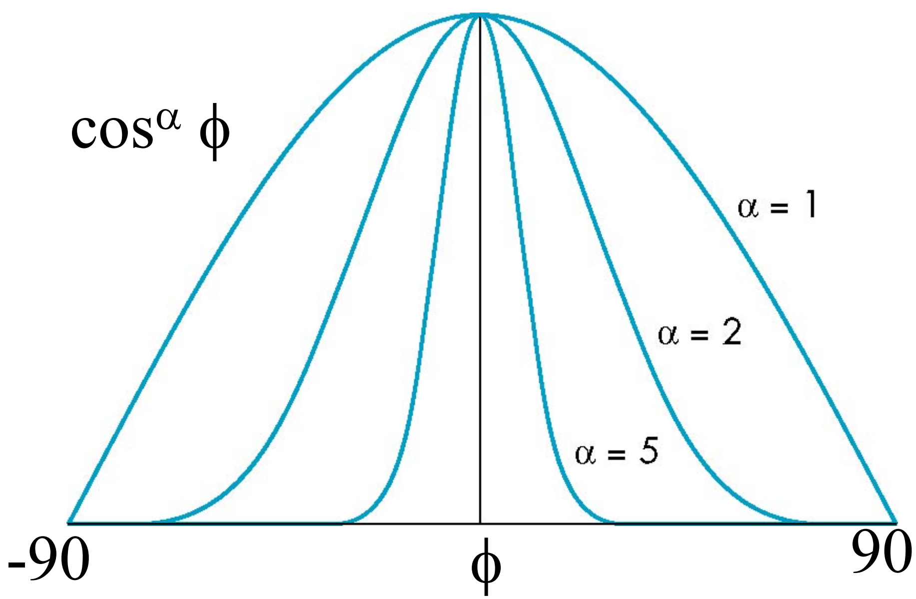
incoming intensity

shininess coef



The Shininess Coefficient

- Values of a between 100 and 200 correspond to metals
- Values between 5 and 10 give surface that look like plastic



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

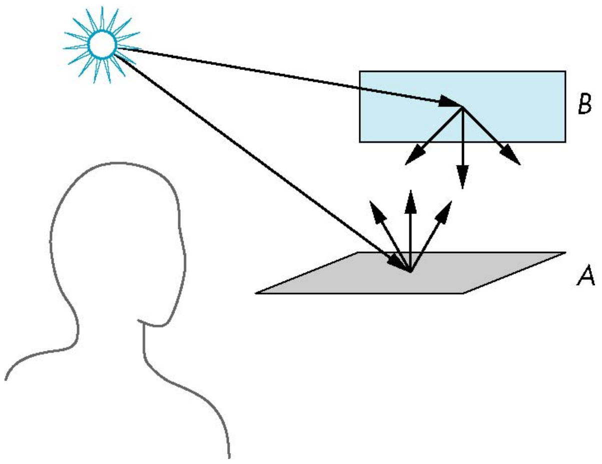
Distance Terms

- 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 $1/(a + bd + cd^2)$ to

the diffuse and specular terms



- d is the distance from the light to surface
- a,b,c are constants you choose to get different effects

Light Sources

- In the Phong Model, we add the results from each light source

- Each light source has separate terms for
  - diffuse
  - specular
  - ambient terms

- 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 reflection coefficients
    - k_{dr}, k_{dg}, k_{db}, k_{sr}, k_{sg}, k_{sb}, k_{ar}, k_{ag}, k_{ab}
    - These vary from 0 to 1

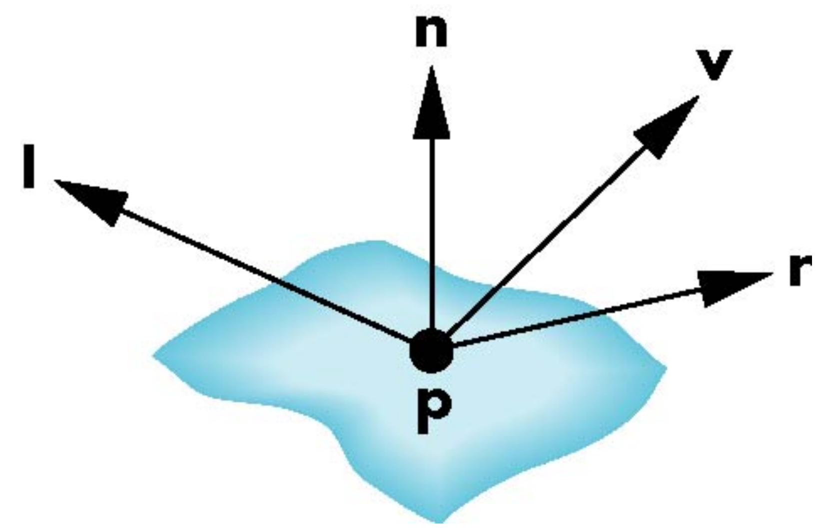
- Materials also have shininess coefficient α

Adding up the Components

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

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

For each color component we add contributions from all sources



Modified Phong Model

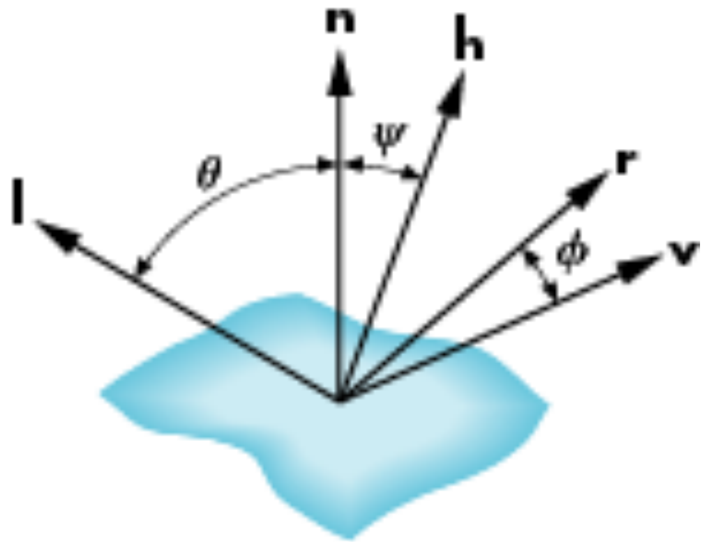
- The specular term in the Phong model was considered problematic because it requires the calculation of a new reflection vector and view vector for each vertex

- Jim Blinn suggested an approximation using the halfway vector that is more efficient under certain conditions
  - Specifically directional lights and orthographic cameras
  - It also produces images closer to those produced by more physically motivated reflection models

The Halfway Vector

- **h** is normalized vector halfway between **l** and **v**

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

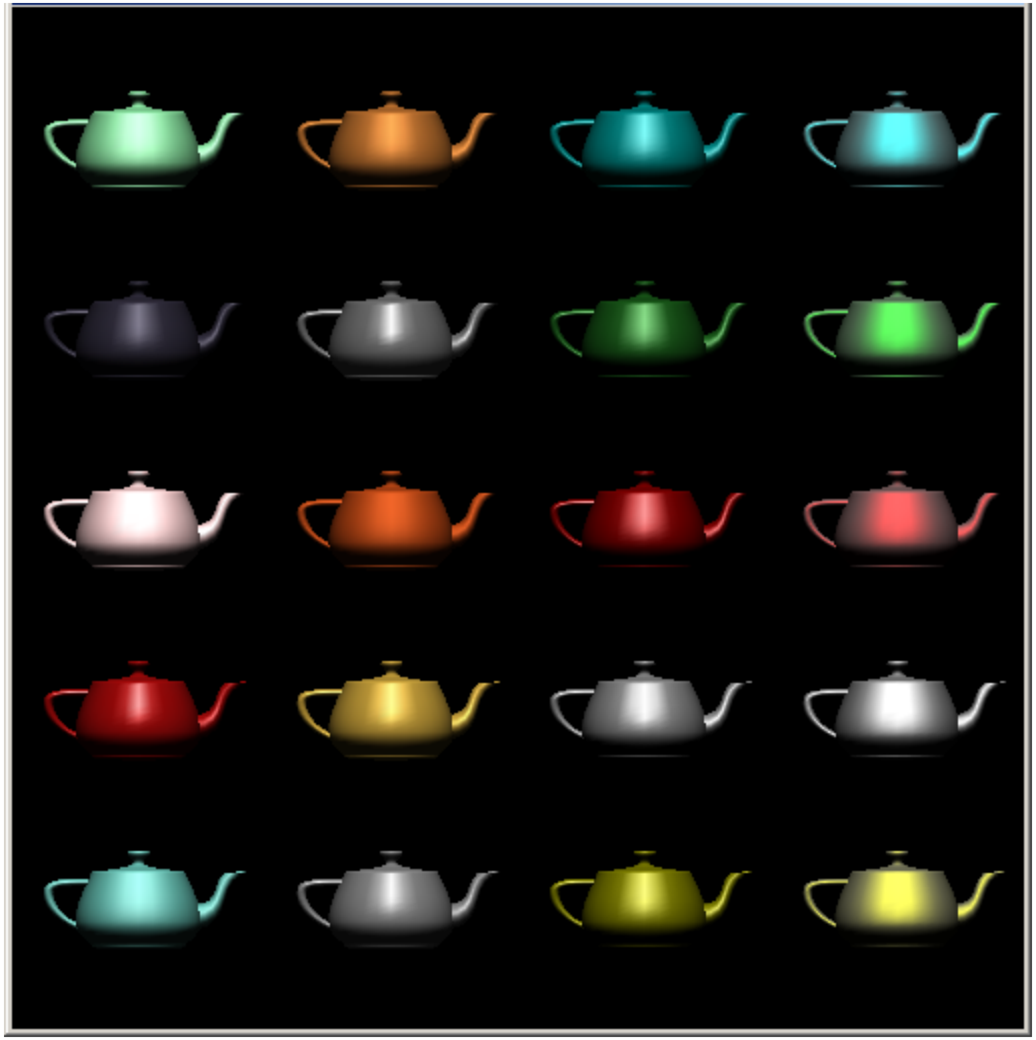


Using the Halfway Vector

- Replace $(\mathbf{v} \cdot \mathbf{r})^{\alpha}$ by $(\mathbf{n} \cdot \mathbf{h})^{\beta}$
- β is chosen to match shininess
- Halfway angle is half of angle between **r** and **v**
  - if vectors are coplanar
- Model is known as the Phong-Blinn lighting model

Example

Only differences in these teapots are the parameters in the modified Phong model



Computation of Vectors

- **l** and **v** are specified by the application
- Can computer **r** from **l** and **n**
- How we determine **n** differs
  - depends on underlying representation of surface