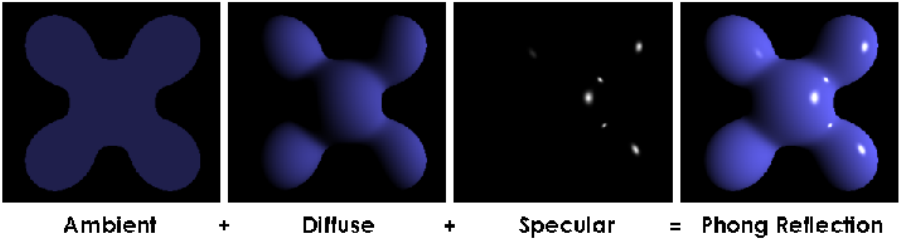
Ray Tracing, Part II

Geoffrey Matthews

Department of Computer Science
Western Washington University

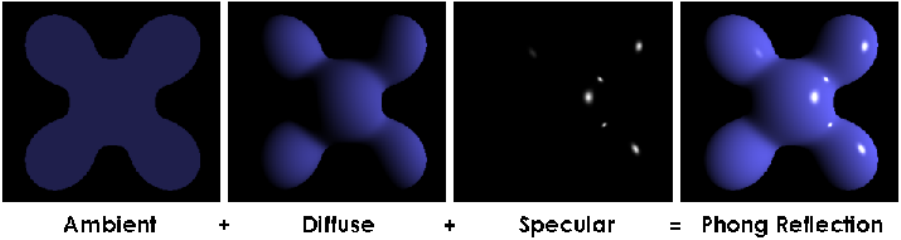
Fall 2015

Shading: Phong Reflection



- Phong reflection model, a combination of three simple shaders
- A *phenomenological* model, not a *physical* one.

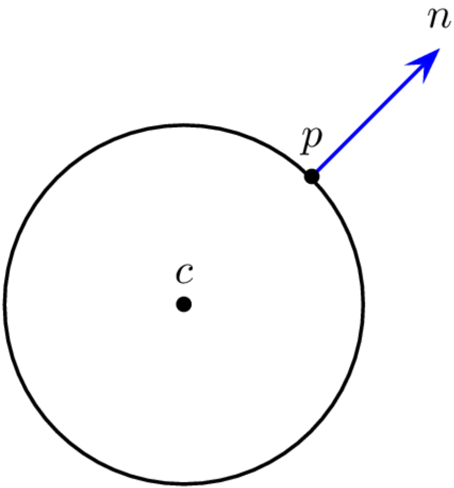
Phong reflection



Colors calculated using three vectors:

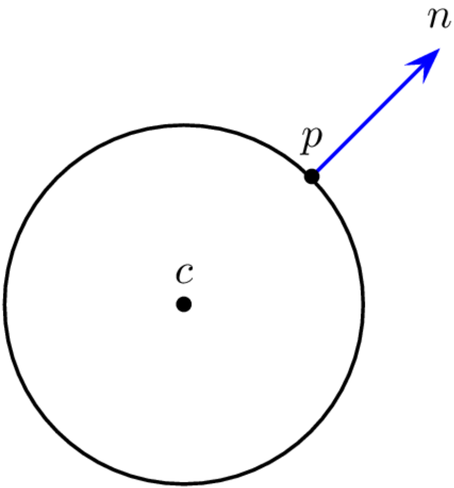
- Orientation of the surface (the *normal*)
- Direction toward *light*
- Direction toward camera (or *eye*)

Spheres: finding the normal at a point



- How do we find the normal?

Spheres: finding the normal at a point



- How do we find the normal?
- $n = p - c$ (may need to normalize)

Spheres: finding the eye vector and the light vector

- How do we find the vector pointing toward the camera?

Spheres: finding the eye vector and the light vector

- How do we find the vector pointing toward the camera?
  camera_position - point_on_sphere

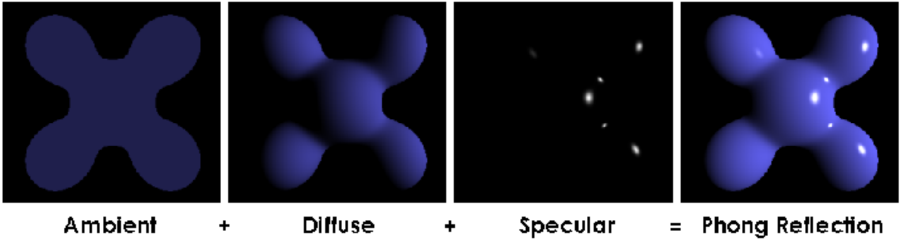
Spheres: finding the eye vector and the light vector

- How do we find the vector pointing toward the camera?
  camera_position - *point_on_sphere*
- How do we find the vector pointing toward the light?

Spheres: finding the eye vector and the light vector

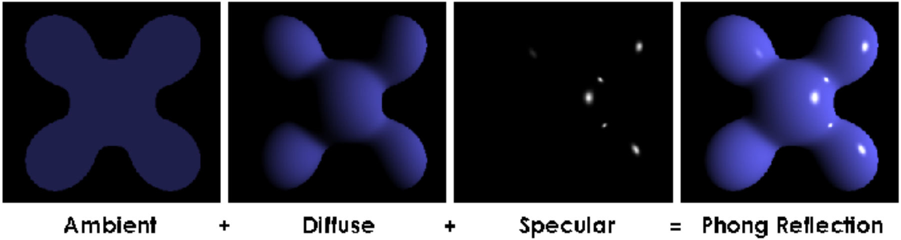
- How do we find the vector pointing toward the camera?
  camera_position - *point_on_sphere*
- How do we find the vector pointing toward the light?
- Depends on the kind of light.
  - *Distant lights*, like the sun, are a fixed direction.
  - *Point lights*, are located at a point: *light_position* - *point_on_sphere*

Shading: Phong Reflection



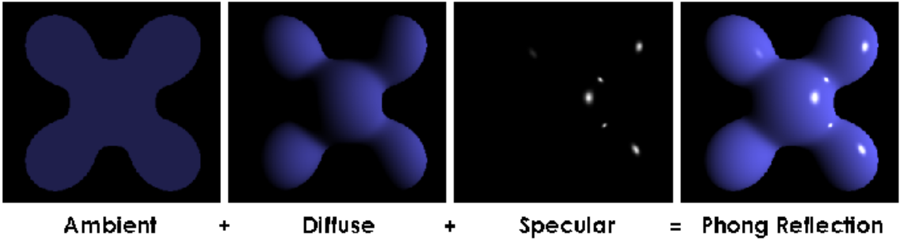
- Given **normal**, **light**, and **eye** vectors, how do we compute each of these factors?

The Ambient Term



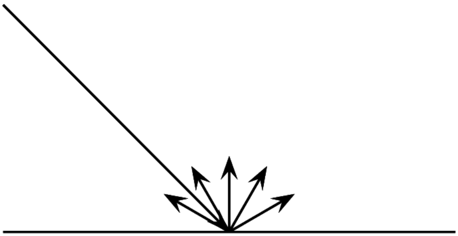
- Light comes reflected and mixed from all objects in the environment.
- Approximate this with a small amount of white light.
- Without this we would get totally black shadows.
- Doesn't use any of the vectors—totally flat.

The Diffuse Term



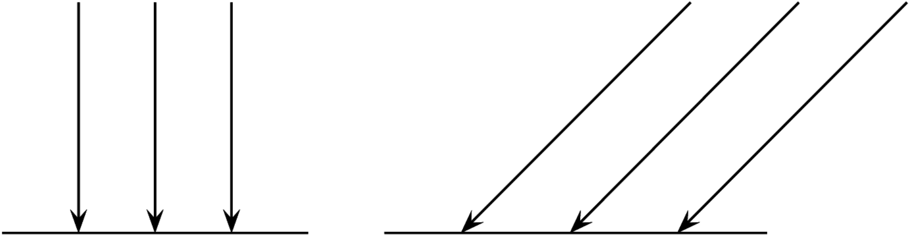
- Gives the shading falloff of light.
- Side oriented toward light: full object color.
- Side oriented away from light: black.

The Diffuse Term: Lambertian Reflection



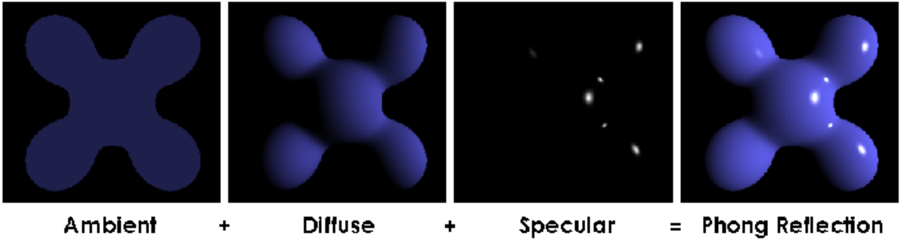
- Objects with rough surfaces reflect light equally in all directions.
- Light energy **coming off** surface in any direction will be proportional to the amount of light **falling on** the surface.

The Diffuse Term: Lambertian Reflection



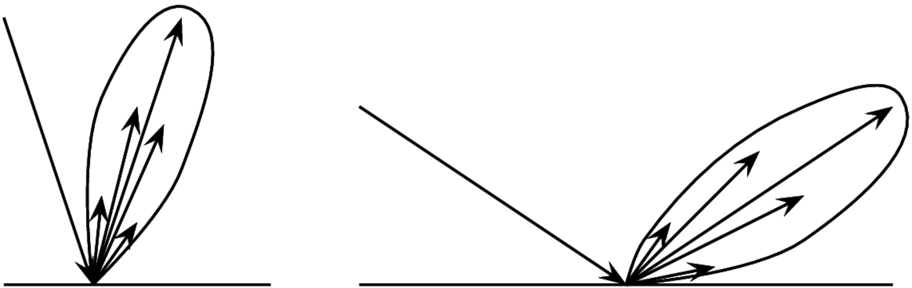
- Light energy falling on a surface is proportional to the cosine of the angle of incidence of the light source.
- Therefore the diffuse term will be proportional to the cosine of the angle of incidence of the light source.
- Depends only on the **light** and **normal** vectors.

The Specular Term



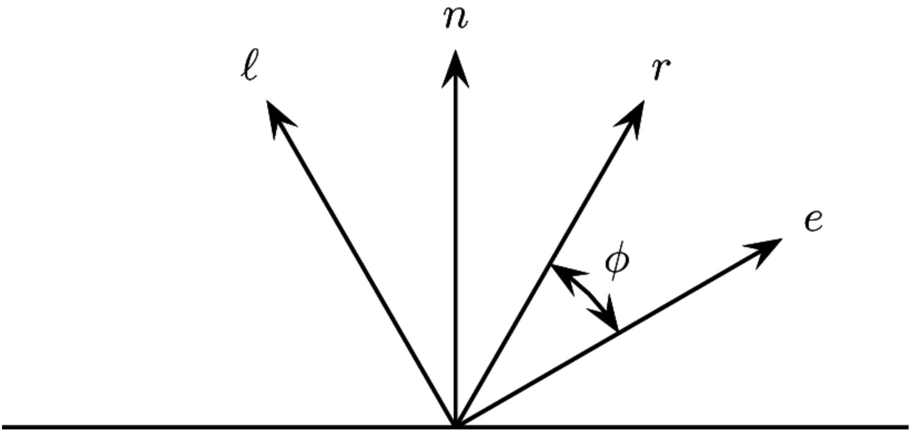
- Shiny spots are tiny blurry images of the light source.
- They depend on **normal**, **light**, and **eye** vectors.
- If you move your head, the shiny spots will move, but the lambertian shade will stay the same.

The Specular Term



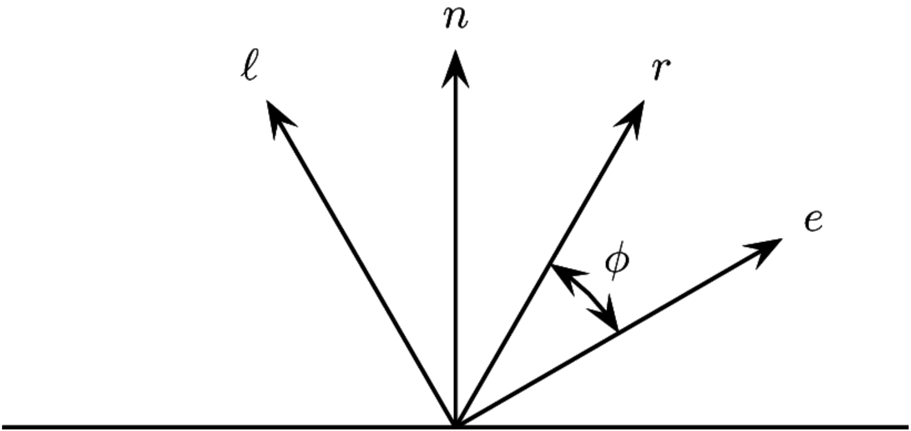
- Smooth surfaces act a bit like mirrors.
- Intensity of light will fall off more or less rapidly from the ideal (mirror) reflection vector.
- How do you calculate the reflection vector?

The Reflection Vector



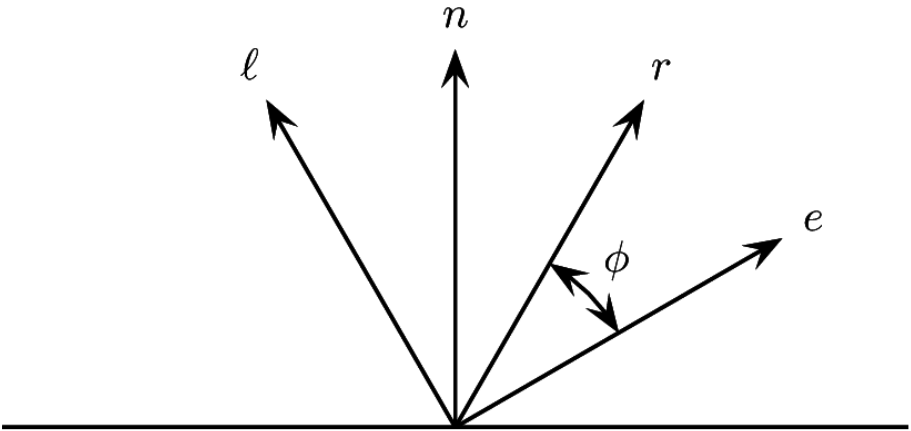
- Assume all vectors are normalized, find r

The Reflection Vector



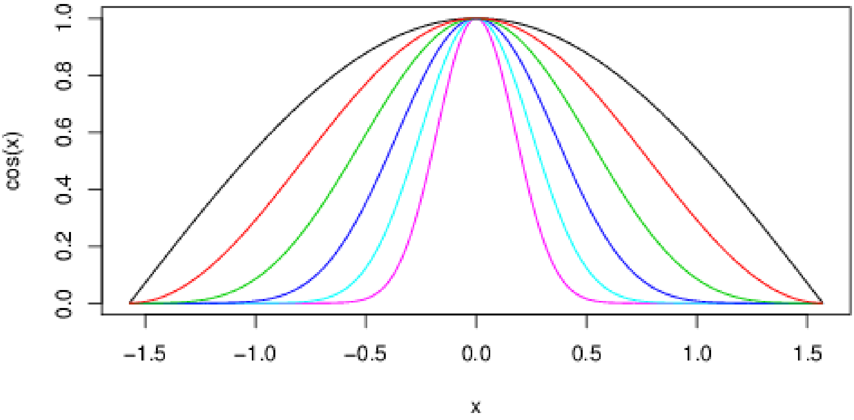
- Assume all vectors are normalized, find r
- $r = \ell - 2(\ell - (n \cdot \ell)n)$

The Reflection Vector



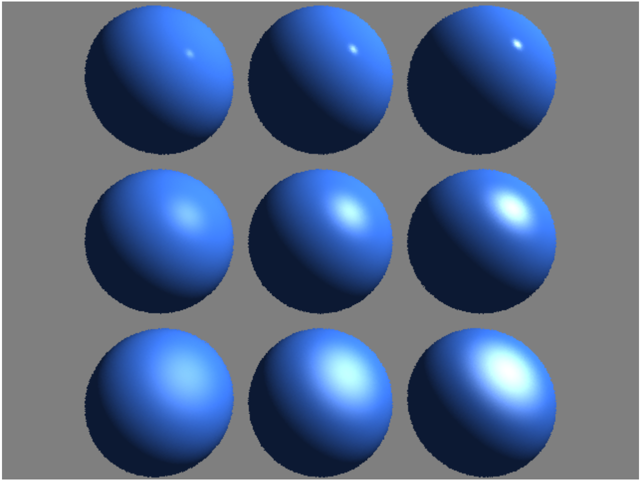
- Assume all vectors are normalized, find r
- $r = \ell - 2(\ell - (n \cdot \ell)n)$
- Use $\cos(\phi) = r \cdot e$

Shininess



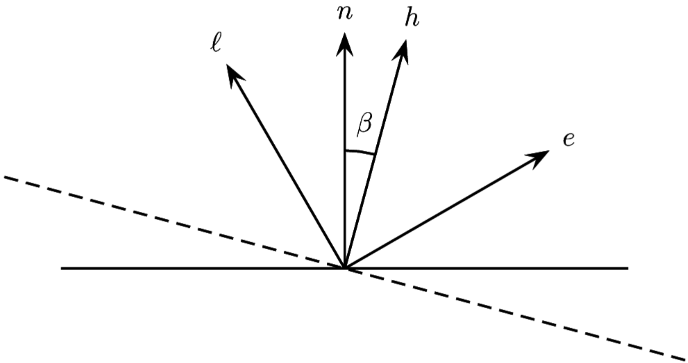
- $\cos(x)^i$ for $i \in \{1, 2, 4, 8, 16, 32\}$

Specular reflection



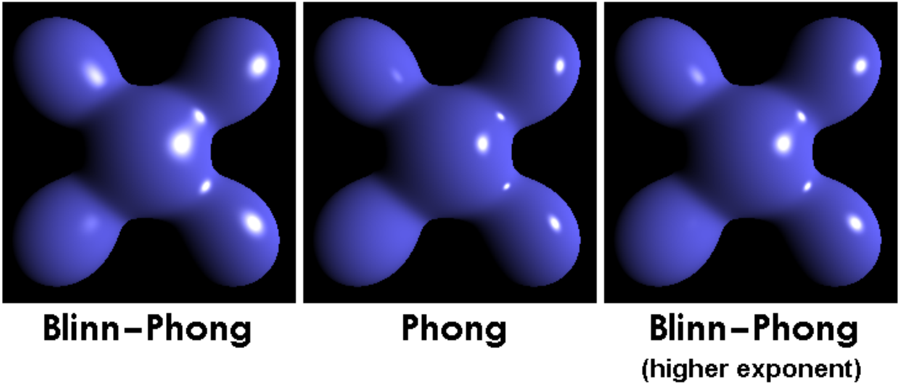
- Specular coefficient in $(0.25, 0.5, 0.75)$
- Shininess in $(3, 9, 200)$

The Halfway Vector: a speed hack



- Use vector halfway between e and ℓ
- If $n = h$, we get the brightest possible reflection.
- Use $\cos(\beta) = h \cdot n$ for the falloff.
- This angle β is about half the angle ϕ found before.
- We can adjust the shininess to handle that.

The Halfway Vector

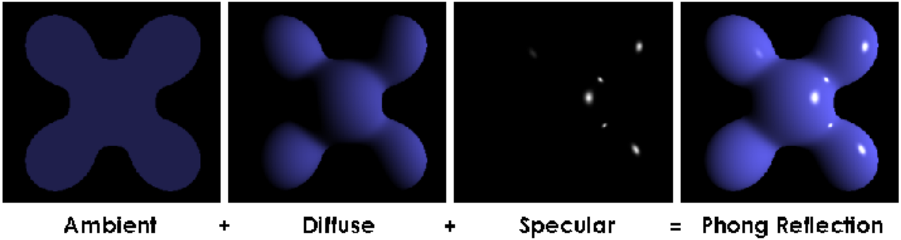


Add up all the terms

$$k_a C_a + \sum_{m \in \text{lights}} (k_d (L_m \cdot N) C_{m,d} + k_s (R_m \cdot E)^\alpha C_{m,s})$$

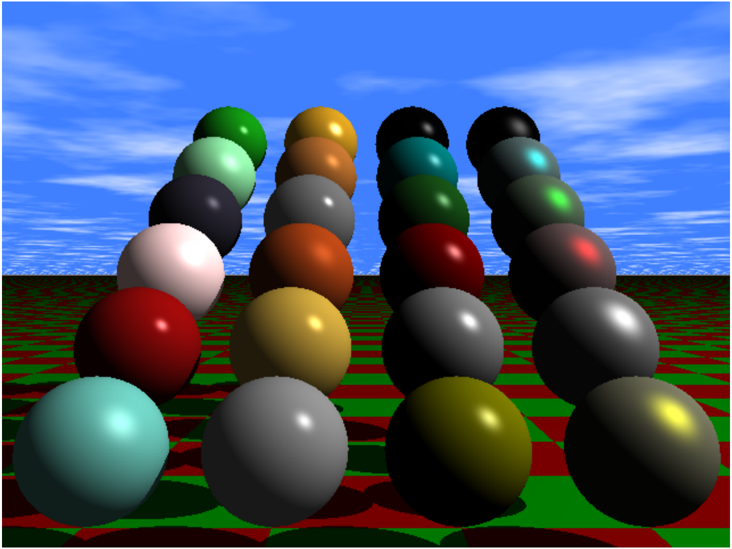
$$\begin{aligned}
C_a, C_s, C_d &= \text{ambient, specular, diffuse colors} \\
k_a, k_s, k_d &= \text{ambient, specular, diffuse reflection constants} \\
\alpha &= \text{shininess} \\
L_m &= \text{light vector} \\
N &= \text{surface normal} \\
E &= \text{eye vector} \\
R_m &= \text{light vector reflected about normal} \\
&= 2(L_m \cdot N)N - L_m
\end{aligned}$$

Phong reflection

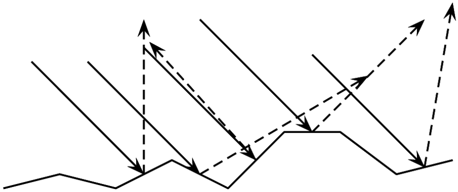


$$k_a C_a + \sum_{m \in \text{lights}} (k_d (L_m \cdot N) C_{m,d} + k_s (R_m \cdot E)^{\alpha} C_{m,s})$$

Phong examples



Microfacets



- Assume a surface is made up of tiny mirrors.
- The statistical distribution of these facets will determine the reflection in each direction.
- More sophisticated stochastic models such as these give better approximations to some surfaces than Phong reflection.