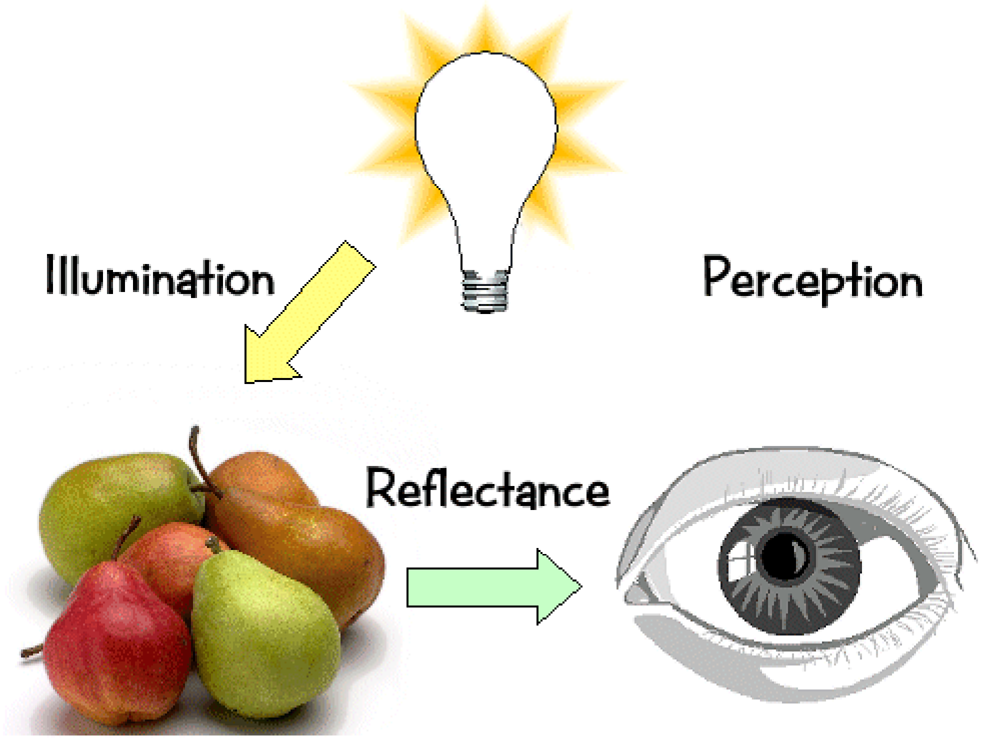
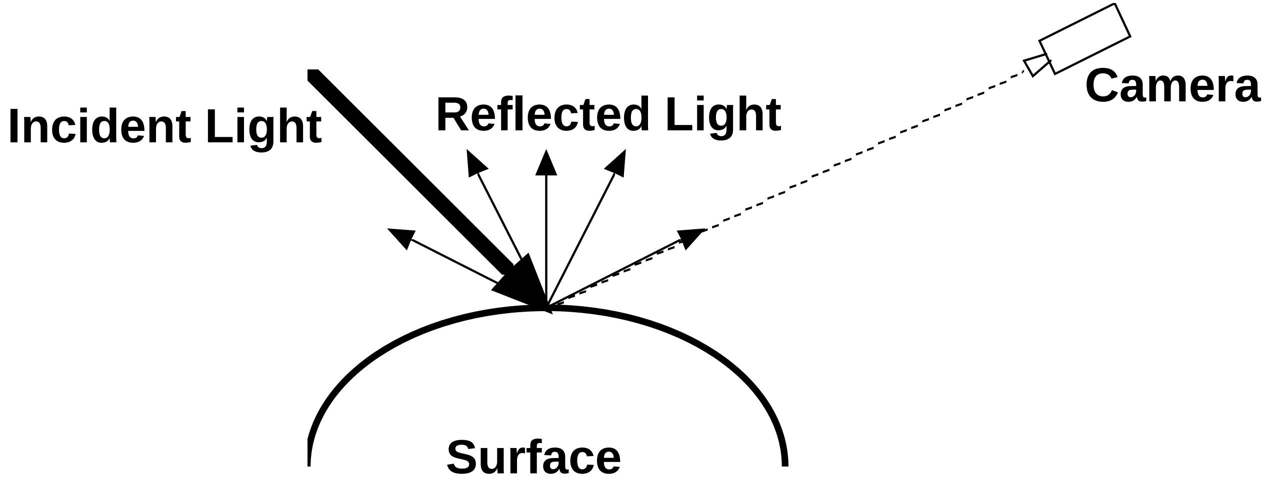
Light Transport



Surface Reflection

- When light hits a surface some is absorbed, the rest is reflected and transmitted (never mind for now)
- The reflected light is what we see
- Reflection is not simple and varies with material
  - the surface's micro structure defines the details of reflection
  - variations produce anything from bright specular reflection (mirrors) to dull matte finish (chalk)

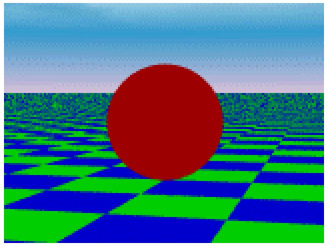


Illumination

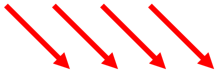
- **Light Sources emit light**
  - EM spectrum (color)
  - Position and direction
- **Surfaces reflect light**
  - Geometry (position, orientation, micro-structure)
  - Absorption
  - Transmission
  - Reflectance
- **Illumination determined by the interactions between light sources and surfaces**

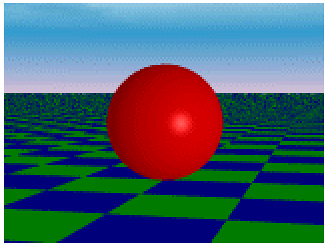
Types of Light Sources

- Ambient: equal light in all directions
  - a hack to model inter-reflections

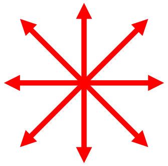


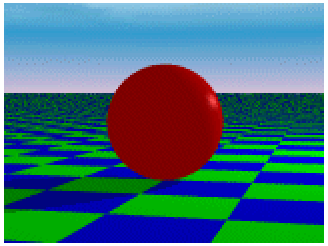
- Directional: light rays oriented in same direction
  - good for distance light sources (sunlight)



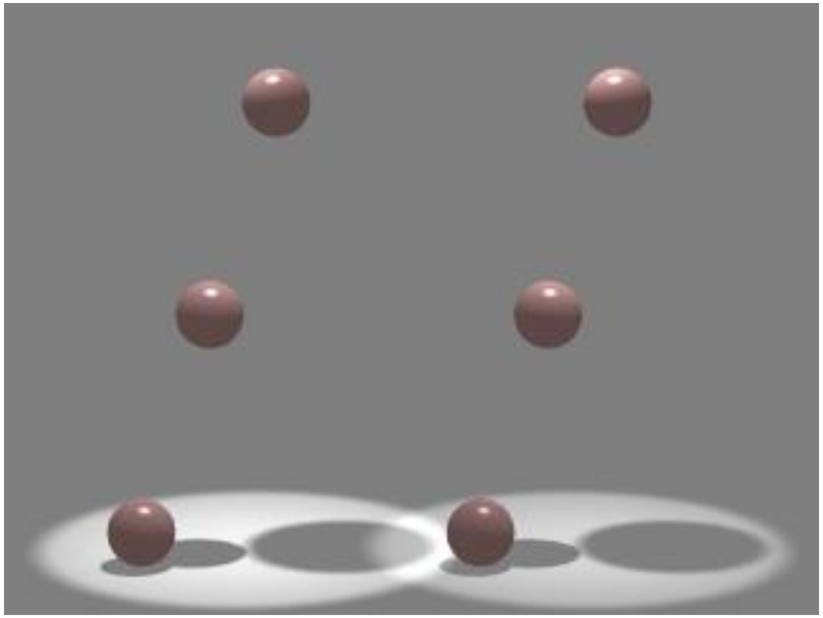


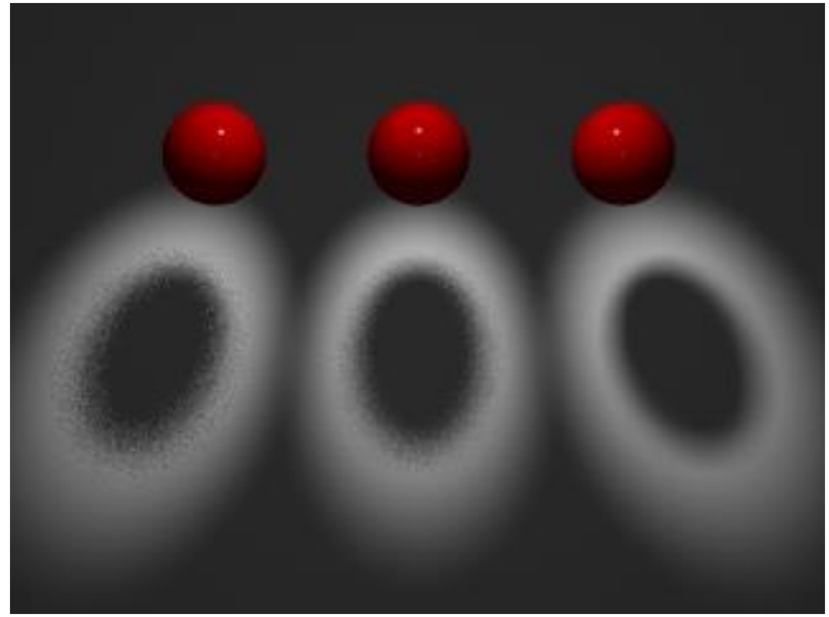
- Point: light rays diverge from a single point
  - approximation to a light bulb





More Light Sources

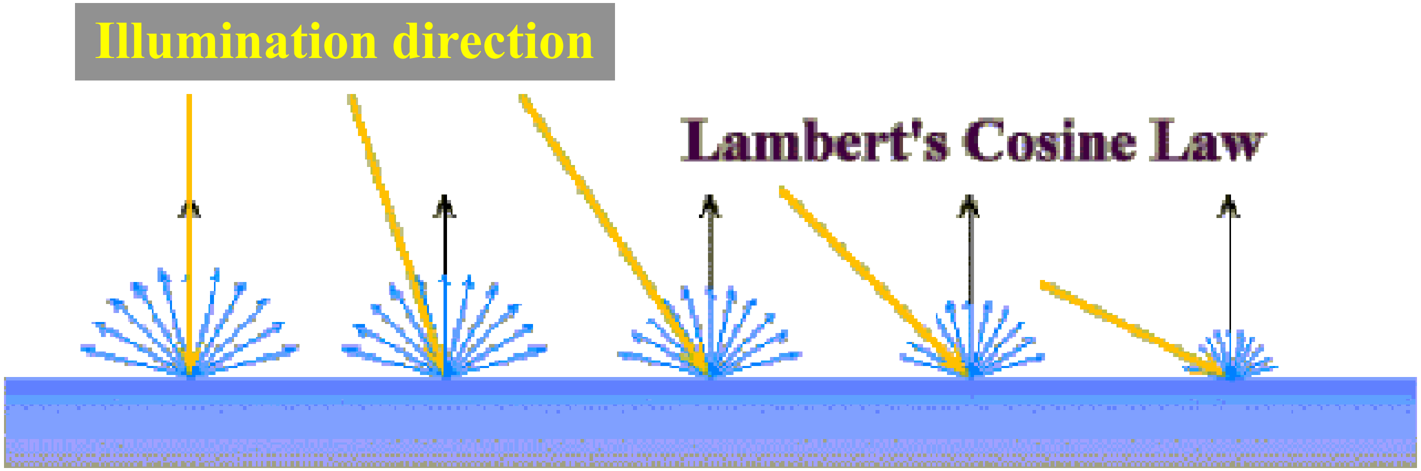




- **Spotlight: point source with directional fall-off**
  - **intensity is maximal along some direction D, falls off away from D**
  - **specified by color, point, direction, fall-off parameters**
- **Area Source: Luminous 2D surface**
  - **radiates light from all points on its surface**
  - **generates soft shadows**

Diffuse Reflection

- Simplest kind of reflector (also known as *Lambertian Reflection*)
- Models a matte surface -- rough at the microscopic level
- Ideal diffuse reflector
  - incoming light is scattered equally in all directions
  - viewed brightness does not depend on viewing direction
  - brightness *does* depend on direction of illumination



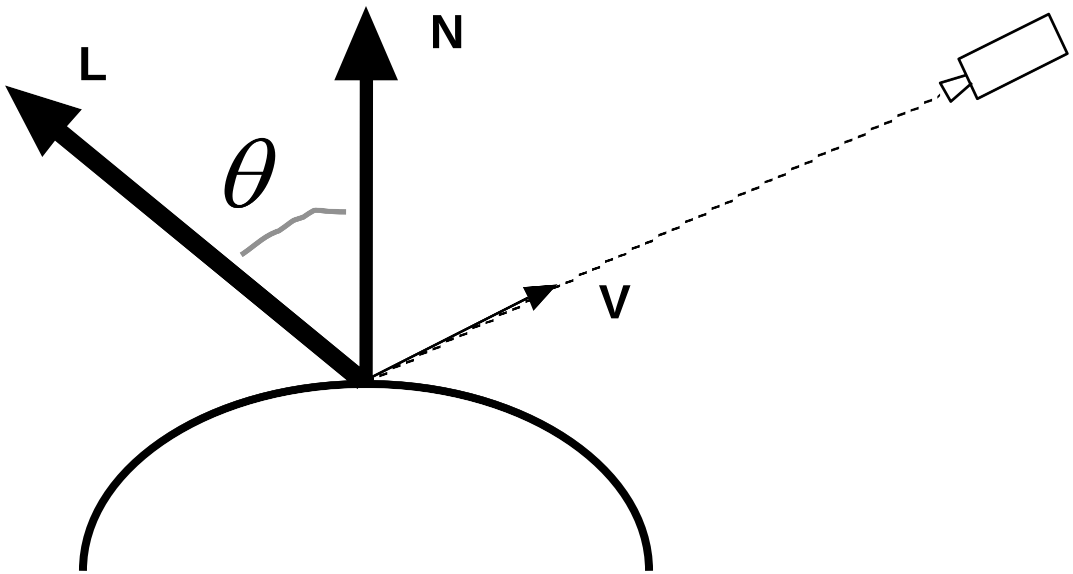
- Lambert's Law

$$I_{diffuse} = k_d I_{light} \cos\theta$$
$$= k_d I_{light} (N \bullet L)$$

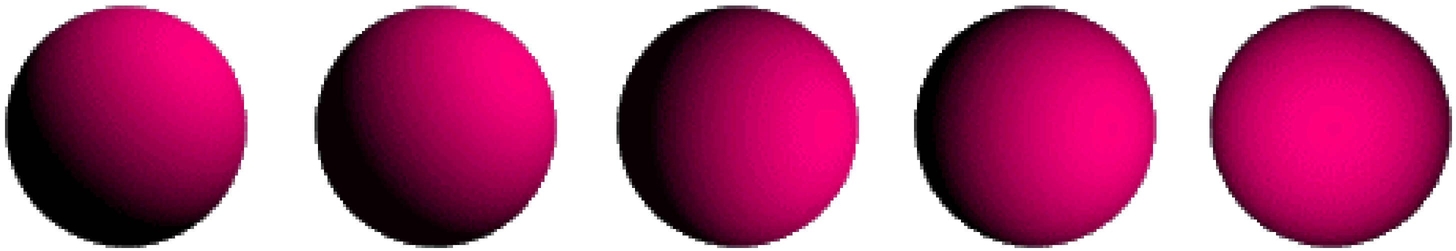
I_{light} : Light Source Intensity

k_d : Surface reflectance coefficient in [0,1]

θ : Light/Normal angle



Examples of Diffuse Illumination

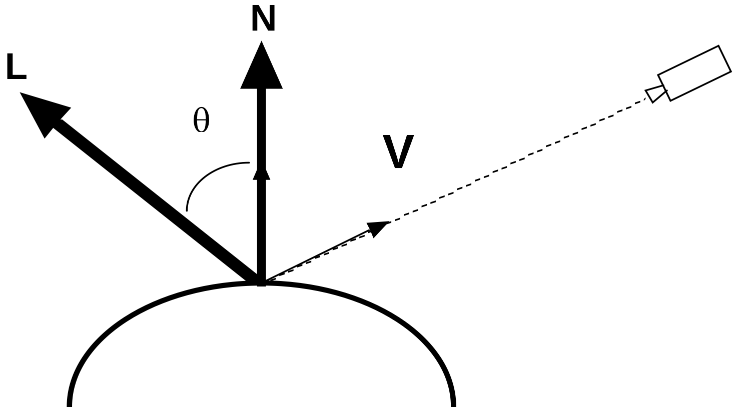


Above is the same sphere lit diffusely from different lighting angles

Ambient + Diffuse Reflection

- CG started using the Lambertian model and then added more terms as extra effects were required

$$I_{d+a} = k_a I_a + k_d I_{light} (N \bullet L)$$



I_a : Ambient light intensity (global)

k_a : Ambient reflectance (local)

- This is diffuse illumination plus a simple ambient light term
  - a trick to account for a background light level caused by multiple reflections from all objects in the scene

Further Simple Illumination Effects

- **Light attenuation:**
  - light intensity falls off with the square of the distance from the source – so we add an extra term for this

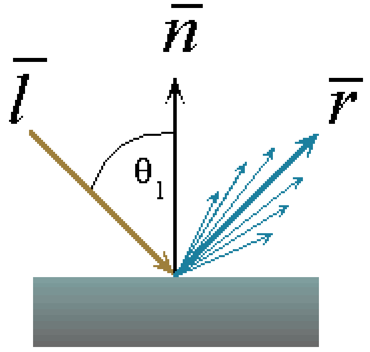
$$I_{d+a} = k_a I_a + f_{att} k_d I_{light} (N \bullet L) \qquad \textbf{where} \qquad f_{att} = \frac{1}{d^2}$$

with d the light source to surface distance

- **Colored lights and surfaces:**
  - just have three separate equations for RGB (or CIE, YIQ, etc.)
- **Atmospheric attenuation:**
  - use viewer-to-surface distance to give extra effects
  - the distance is used to blend the object's radiant color with a "far" color (e.g., a nice hazy gray)

Specular Reflection

- Shiny surfaces change appearance when viewpoint is varied
  - specularities (highlights) are view-dependent
  - caused by surfaces that are microscopically smooth
- For shiny surfaces part of the incident light reflects coherently
  - an incoming ray is reflected in a single direction (or narrow beam)
  - direction is defined by the incoming direction and the surface normal
- A mirror is a perfect specular reflector
  - approximate specular reflectors give fuzzy highlights



Phong Illumination

- One function that approximates specular falloff is called the *Phong Illumination* model

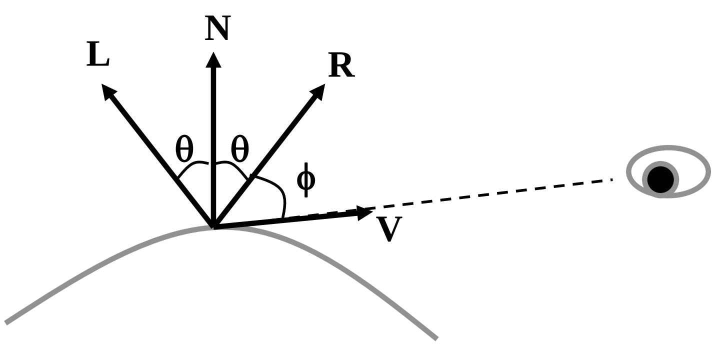
$$I_{specular} = k_s I_{light} (\cos\phi)^{n_{shiny}}$$

ϕ : Angle between reflected light ray **R** and viewer **V**

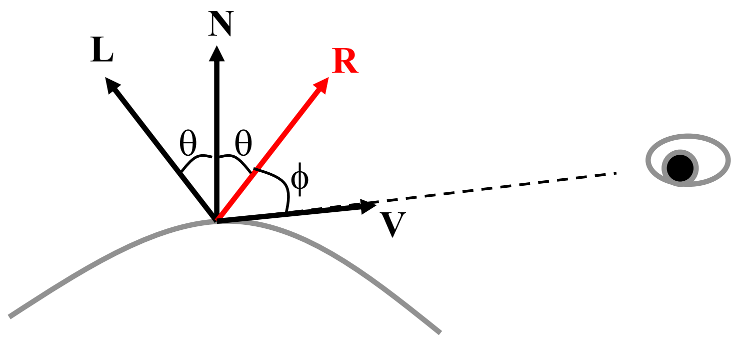
k_s : Specular reflectance

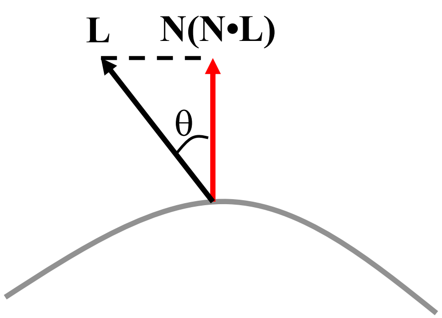
n_{shiny} : Rate of specular falloff

  - no physical basis, yet widespread use in computer graphics

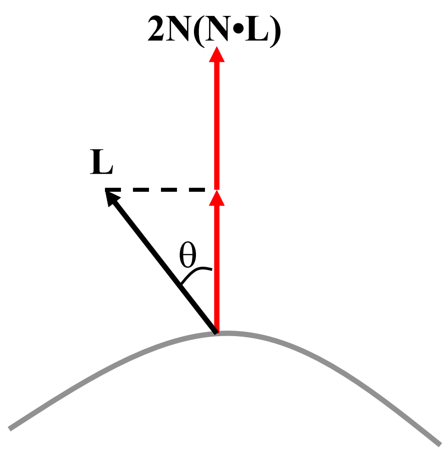


Computing the Reflected Ray

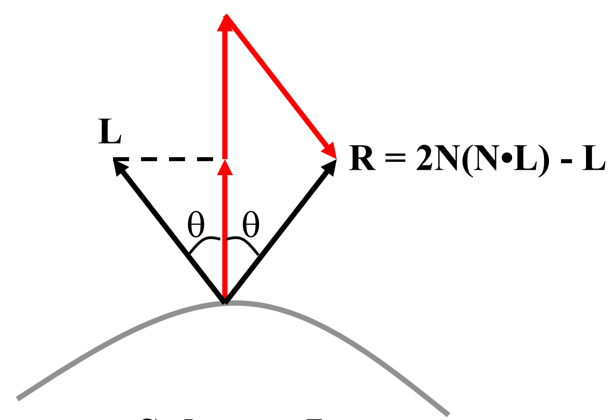




Project L onto N



Double length of vector



Subtract L

Phong Illumination Curves

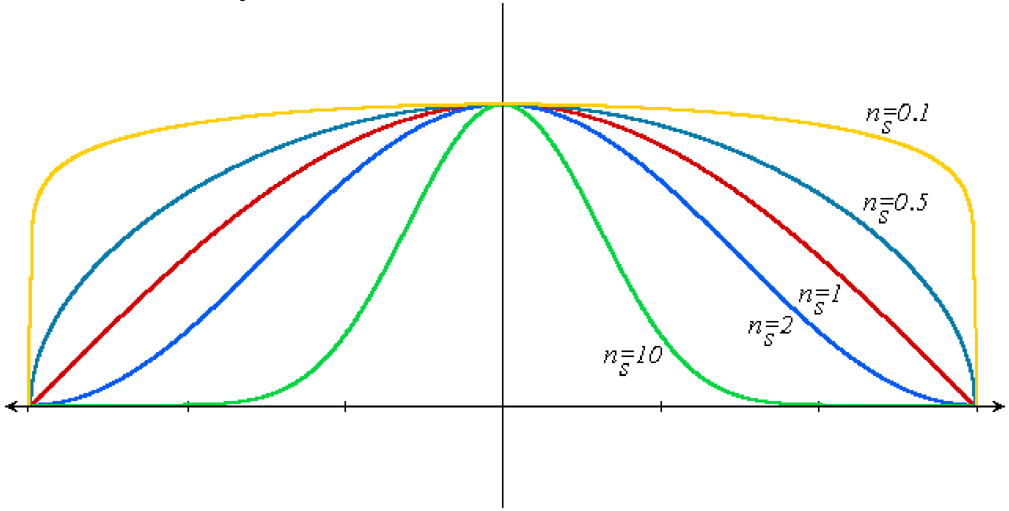
- The specular exponents are often much larger than 1; values of 100 are not uncommon.

$$I_{specular} = k_s I_{light} (\cos\phi)^{n_{shiny}}$$

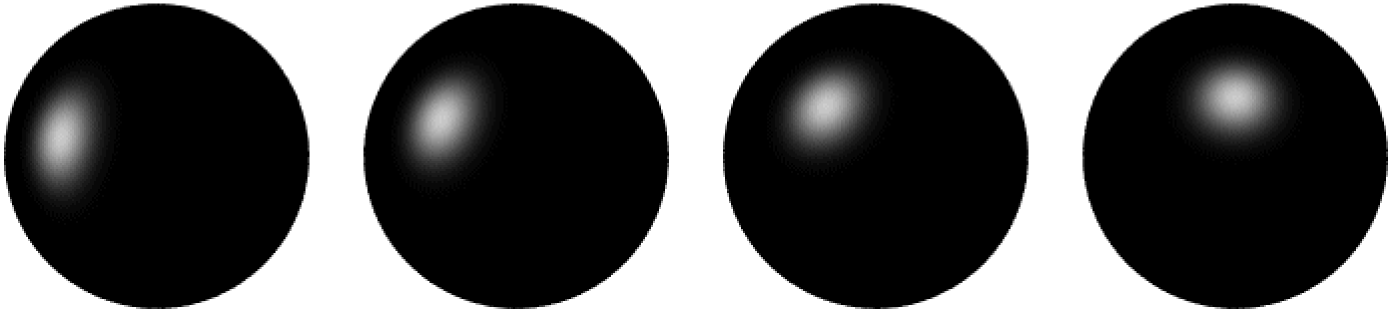
ϕ : angle between line of sight and perfect reflection

k_s : Specular reflectance

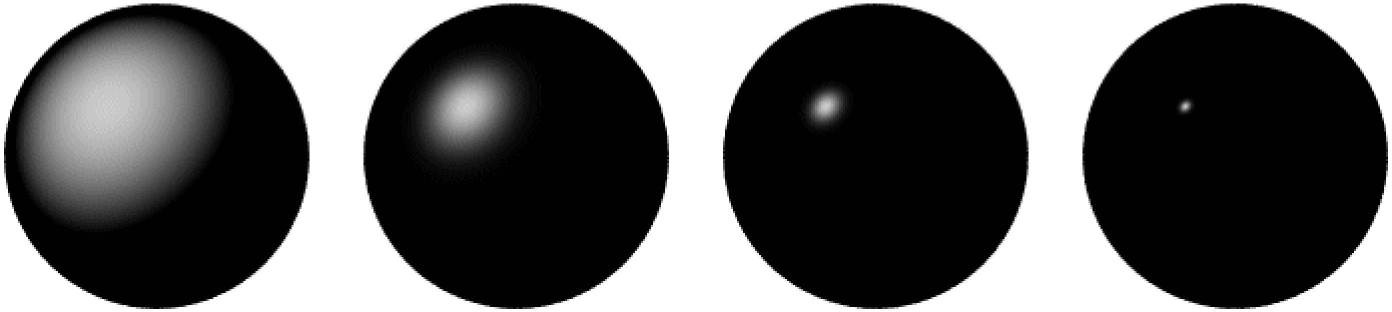
n_{shiny} : Rate of specular falloff



Phong Illumination



Moving the light source



Changing n_{shiny}

Putting It All Together

$$I = k_a I_a + f_{att} I_{light} \left[k_d \cos\theta + k_s (\cos\phi)^{n_{shiny}} \right]$$

- Combining ambient, diffuse, and specular illumination
- For multiple light sources
  - Repeat the diffuse and specular calculations for each light source
  - Add the components from all light sources
  - The ambient term contributes only once
- The different reflectance coefficients can differ.
  - Simple "metal": k_a and k_d share material color, k_s is white
  - Simple plastic: k_s also includes material color
- Remember, when cosine is negative lighting term is zero!

Where do we Illuminate ?

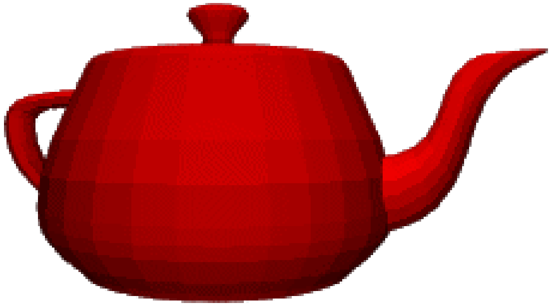
To this point we have discussed how to compute an illumination model at a *single* point on a surface. But, at which points on the surface is the illumination model applied? Where and how often it is applied has a noticeable effect on the result.

Illuminating can be a costly process involving the computation of and normalizing of vectors to multiple light sources and the viewer.

Flat Shading

The simplest shading method applies only one illumination calculation at one point for each primitive. Which one? Usually the centroid. For a convex facet the centriod is given as follows:

$$centroid = \frac{1}{vertices} \sum_{i=1}^{vertices} \overline{p}_i$$



This technique is called *constant or flat shading.* It is often used on polygonal primitives.

Facts:

- For point light sources the direction to the light source varies over the facet
- For specular reflections the direction to the eye varies over the facet
- Facet facts are clearly visible

Gouraud Shading

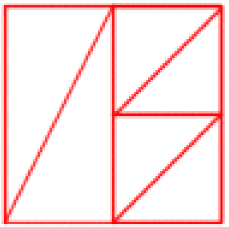
The Gouraud Shading method applies the illumination model on a subset of surface points and interpolates the intensity of the remaining points on the surface. In the case of a polygonal mesh the illumination model is usually applied at each vertex and the colors in the triangles interior are linearly interpolated from these vertex values during polygon rasterization.

Facet artifacts are still visible.

Gouraud Shading

- Gouraud shading interpolates illumination values at each vertex using *screen-space interpolation.*
- Screen-space interpolation is ***wrong***!
- However, you usually will not notice because the transition in colors is very smooth.
- There are some cases where the errors in Gourand shading become obvious.
  - When switching between different levels-of-detail representations
  - At "T" joints.

A "T" joint



Phong Shading

In Phong shading (not to be confused with Phong's illumination model), the surface normal is linearly interpolated across polygonal facets, and the Illumination model is applied at every point.

A Phong shader assumes the same input as a Gouraud shader, which means that it expects a normal for every vertex. The illumination model is applied at every point on the surface being rendered, where the normal at each point is the result of linearly interpolating the vertex normals defined at each vertex of the triangle.

Phong shading will usually result in a very smooth appearance, facet artifacts can only be seen along silhouettes.

Comparison

faceted shading shading   Gouraud shading   Phong