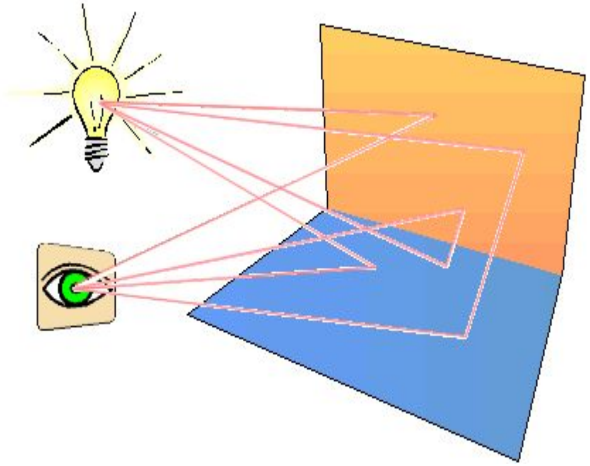
Lighting and Shading

Why we need?
- a model of a sphere using many polygons with a single color (left: default settings, we want the right one)
- light-material interactions cause each point to have a different color or shade
- right one is more realistic, color differences between the points (some parts get more, some parts get less light)

Lighting and Shading

To calculate the color consider:
- Light sources
- Material properties
- Location of viewer
- Surface orientation

Lighting and Shading



Purpose: realistic visualization. Concepts:

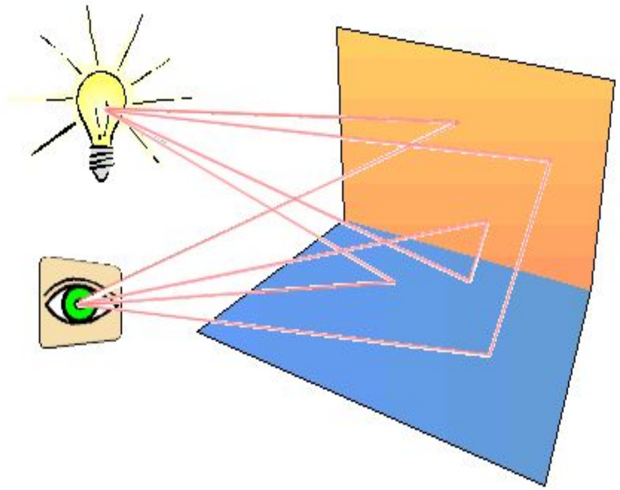
Illumination: transport of energy from light sources to points on surfaces via direct & indirect paths

Lighting: computing intensity reflected from 3D points in scene

Shading: assigning (determining) pixel colours, the final stage after illumination and lighting

* lighting and illumination may be used interchangeably, but we have basically made their definitions

Lighting and Shading

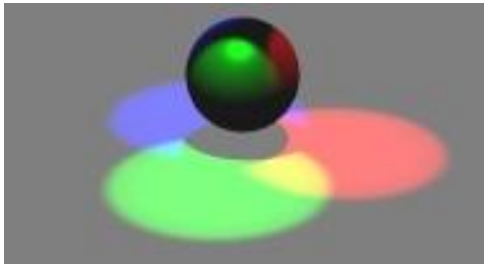


two types of illumination approaches (the types of models used for light calculations):

- Empirical models: are simple approximations to observed phenomena.

- Physically-based models: model actual physics of light interactions.

Components of Illumination



- Light sources:

– Spectrum of emittance (color of the light)

– Geometric attributes

Position, Direction, Shape

– Attenuation (with respect to distance or direction)

increase as the distance increases, occur as the object moves angularly away from direction of light

- Surface (reflecting light) properties:

– Reflectance spectrum (color of the surface)

– Subsurface reflectance (how and how much the surface reflects the light)

– Geometric attributes

Position, Orientation, Microstructure (internal structure: rough or smooth)

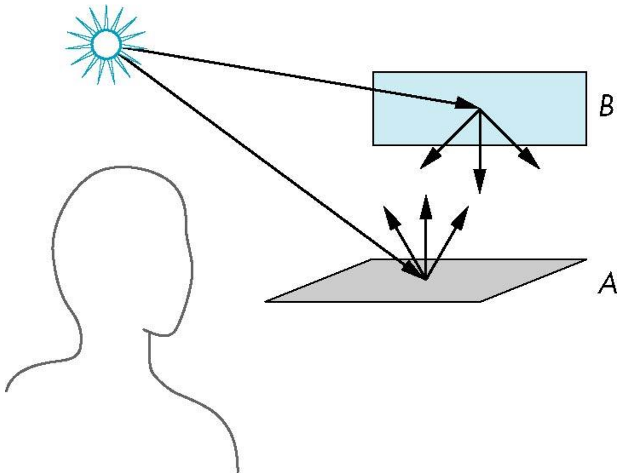
Scattering

Surfaces A, B.

w.r.t. light source and surface properties, some of the rays are scattered, some are absorbed.

- 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 (till energy of light is completely extinguished)

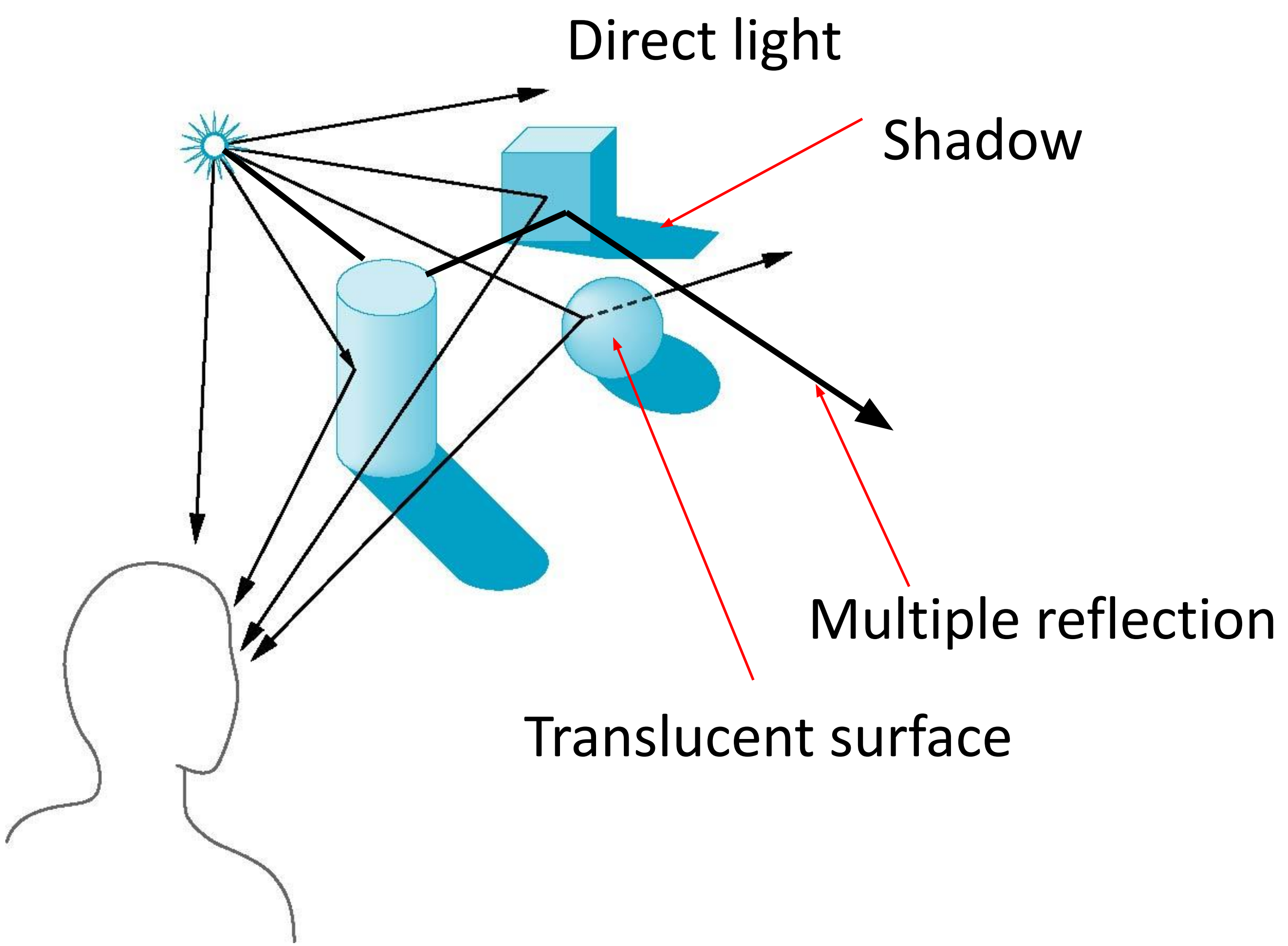


render equation: make all these calculations and model this situation

Rendering Equation

- The infinite scattering and absorption of light can be described by the *rendering equation*
  - Cannot be solved in general
  - Ray tracing is a special case for perfectly reflecting surfaces (mirrors)
  - Possible when the angle of incidence of the light on the surface and the angle of reflection are equal
- Rendering equation is global/universal (All direct rays and all other surfaces where light reflects, have an effect on a pixel color) and includes:
  - Shadows
  - Multiple scattering from object to object

Global Effects

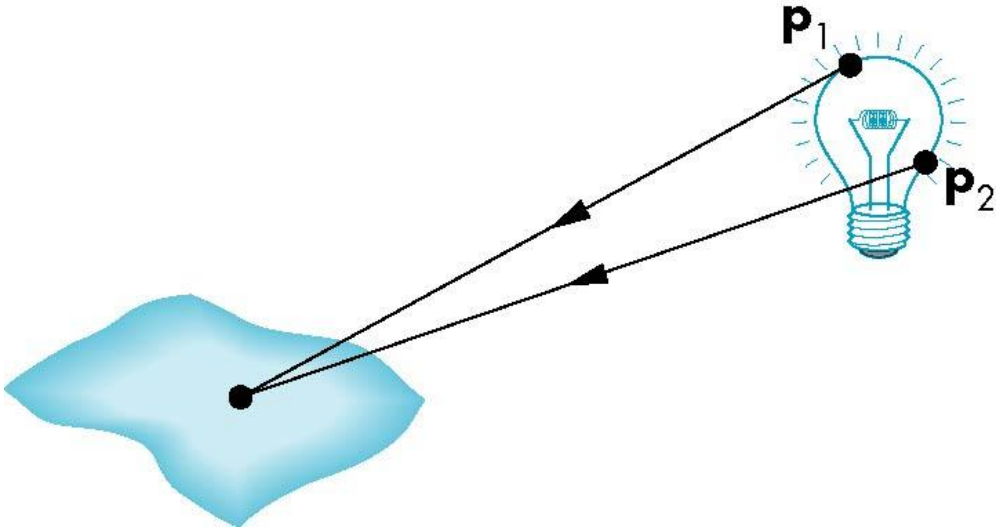


Local vs. Global Rendering

- Correct shading requires a global calculation involving all objects and light sources
  - Incompatible with pipeline model which shades each polygon independently (local rendering)
- However, in computer graphics, especially real time graphics, we are happy if things simply "look right". We do not use global calculations in the graphics pipeline.
  - Many techniques approximate global effects (empirical models)

Light Sources

Light sources are difficult to work with (difficult to model) because we must integrate light coming from all points on the source



Light is emitted from every point of the filament of the light bulb

Simple Light Sources

- Since real light sources are difficult to model, in theory, simple light sources are defined: *point source, spotlight, and ambient.*

Simple Light Sources

- Point source
  - No shape, no size, light is emitted from a single point
  - Location of the light source is important for the calculations
  - Modeled with position and color
  - If source is in an infinite distance, (instead of its location or distance to the object) its direction (the direction of the light) becomes important
  - e.g. sun light (directional light source)
- Spotlight
  - Restrict light from ideal point source angularly
  - e.g. table lamp (rays illuminate up to a certain angle)
- Ambient light

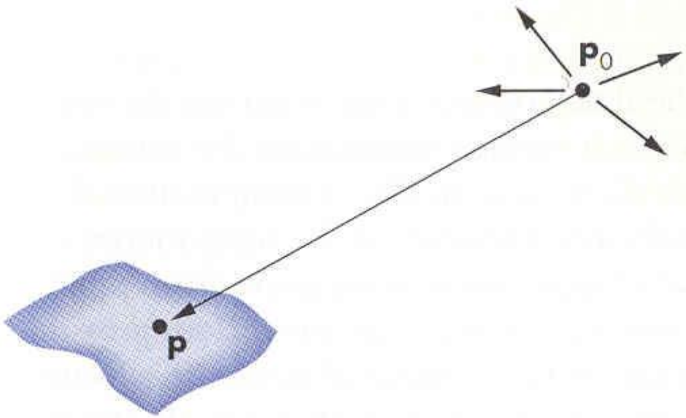
–Same amount of light everywhere in scene, every point on the surface (background light)

–Can model contribution of many light sources and light reflecting surfaces (global effects)

Light Intensity Attenuation - Distance

- p: object location, p_0: point light location
- $I(p_0)$: initial intensity of light

$$\mathbf{I}(\mathbf{p}_0) = \begin{bmatrix} I_r(\mathbf{p}_0) \\ I_g(\mathbf{p}_0) \\ I_b(\mathbf{p}_0) \end{bmatrix}$$



- $I(p,p_0)$: intensity of light reflected from the object surface. Inversely proportional to the squared distance:

$$\mathbf{I}(\mathbf{p}, \mathbf{p}_0) = \frac{1}{|\mathbf{p} - \mathbf{p}_0|^2} \mathbf{I}(\mathbf{p}_0)$$

$$f_{att} = 1/(d_{light})^2$$

d: distance between the light source and the object

Light Intensity Attenuation - Distance

- Point source with distance d isn't realistic (too much intensity variation for near objects to little variation for distant objects)

$f_{att} = 1/(d_{light})^2$

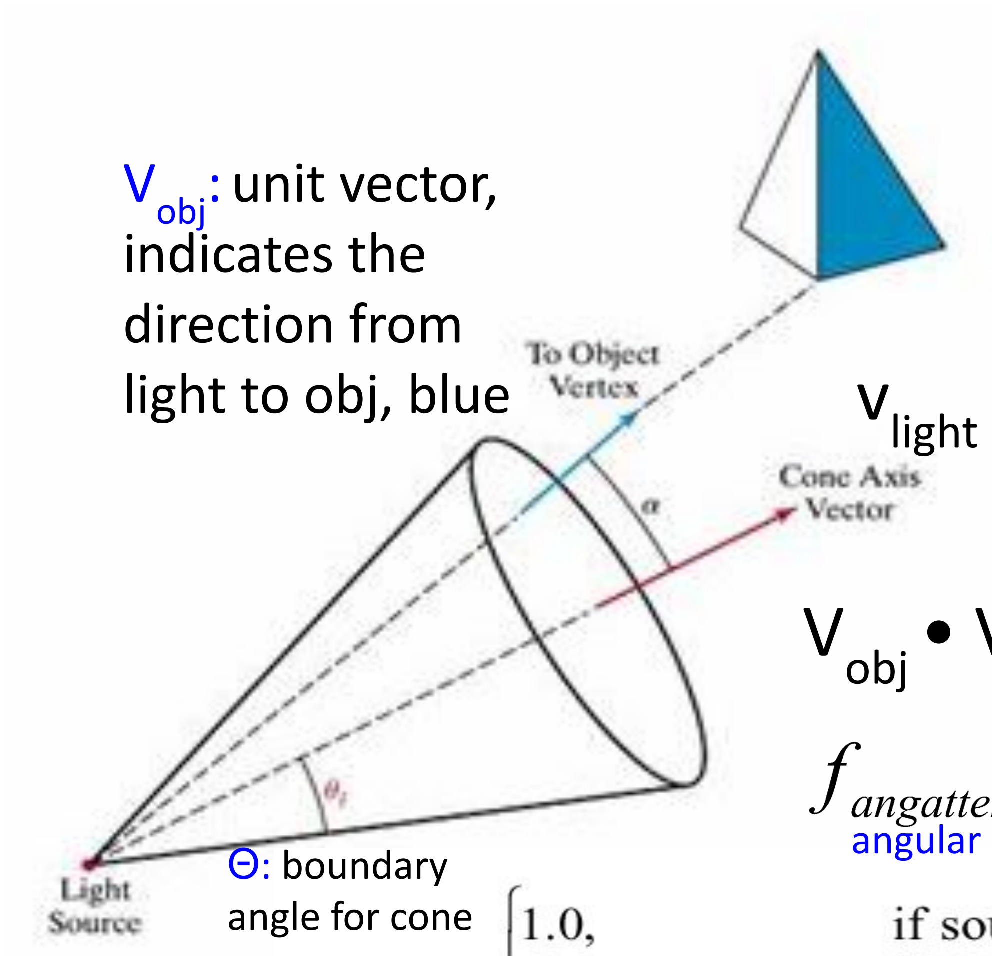
$$\mathbf{I}(\mathbf{p}, \mathbf{p}_0) = \frac{1}{|\mathbf{p} - \mathbf{p}_0|^2} \mathbf{I}(\mathbf{p}_0)$$

- We use this equation for light attenuation (for control over attenuation using k constants)
- For a very distant light source (e.g. sun) attenuation is ignored, f_{att} is accepted as 1

$f_{att} = 1/(k_0 + k_1 d_{light} + k_2 d_{light}^{\ 2})$

$$I(p, p_0) = \frac{1}{k_0 + k_1 d + k_2 d^2} I(p_0)$$

Light Intensity Attenuation - Direction



V_{obj}: unit vector, indicates the direction from light to obj, blue

V_{light}

- we set an angular limit on the point light source (spotlight), conical structure
- Objects (or pixels of the object inside the cone) will be affected by the rays, the area outside the cone will not be illuminated

$$V_{obj} \bullet V_{light} = \cos \alpha$$

$$f_{angatten}(\phi) = \cos^{a_l} \phi \quad 0^o \le \phi \le \theta$$

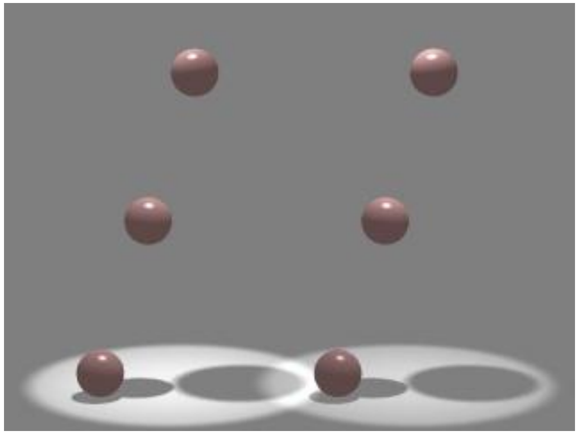
angular attenuation

Θ: boundary angle for cone

$$f_{l,angatten} = \begin{cases} 1.0, & \text{if source is not a spotlight} \\ 0.0, & \text{if } \mathbf{V}_{\mathbf{obj}} \cdot \mathbf{V}_{\mathbf{light}} = \cos\alpha < \cos\theta_l \\ & \text{(object is outside the spotlight cone)} \\ (\mathbf{V}_{\mathbf{obj}} \cdot \mathbf{V}_{\mathbf{light}})^{a_l}, & \text{otherwise} \end{cases}$$

a_l coefficient is added

Other Light Sources

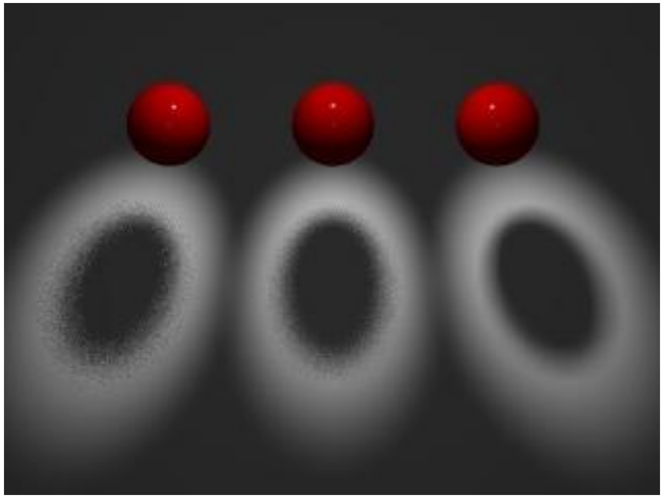


- point lights create sharp shadows.

- in reality (since the light does not radiate from a single point as in fluorescent) softer shadows are formed.

- area light sources were also modeled.

- Area Light Sources
  - Light source occupies a 2-D area (usually a polygon or disk)
  - Generates *soft* shadows
- Extended Light Sources
  - Spherical Light Source
  - Generates *soft* shadows



Surface Effects

- Surface properties:
  - Reflectance spectrum (color of the surface)
  - Subsurface reflectance (how and how much the surface reflects the light)
  - Geometric attributes

    Position, Orientation, Micro-structure (internal structure, e.g. rough or smooth)

Like the attenuation of the light, these surface properties will be taken into account in the lighting models

Basic (Phong) Illumination Model

- the easiest model for pipeline
- results are approximate but not too bad
- to shade an object, other objects in the scene are not fully taken into account

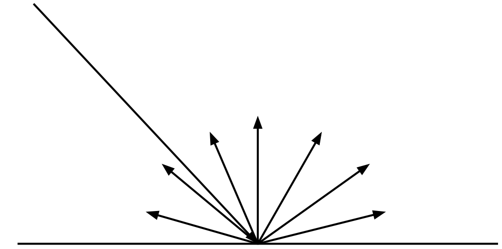
3 components:

- *Diffuse* component
- *Specular* component
- *Ambient* term

Basic (Phong) Illumination Model

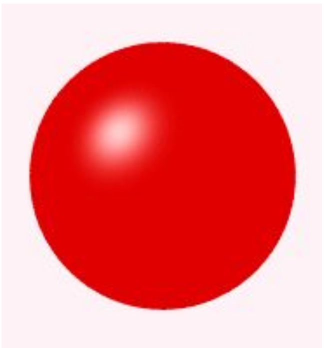
– ***Diffuse term:***

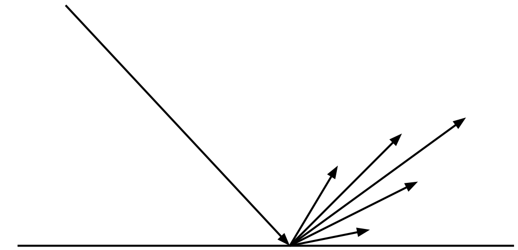
for the amount of incoming light, reflected equally in all directions



– ***Specular term:***

for the amount of light reflected in a mirror-like fashion

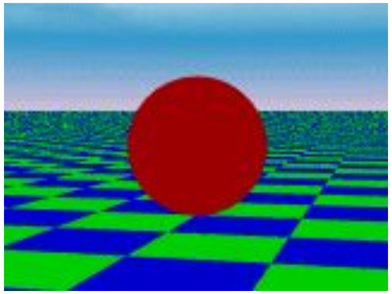


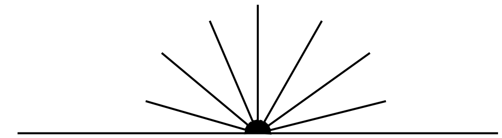


– ***Ambient term:***

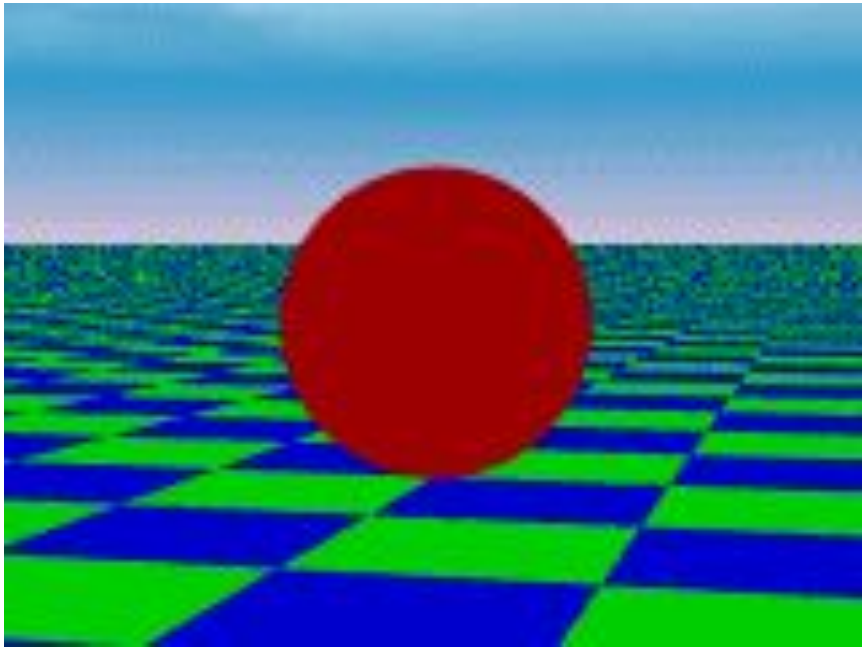
have equal light on every object, everywhere on the scene

approximates light arriving via other surface





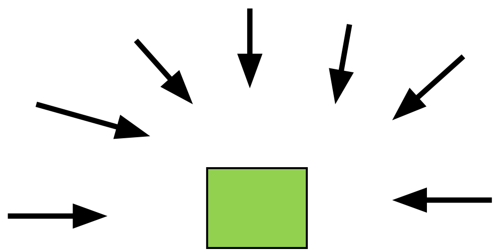
Ambient Light



sphere or circle?

- if only ambient light is used, the forms of the objects cannot be understood.
- determines how bright a surface will appear when there is no direct light reaching the surface.

Modeling Ambient Light



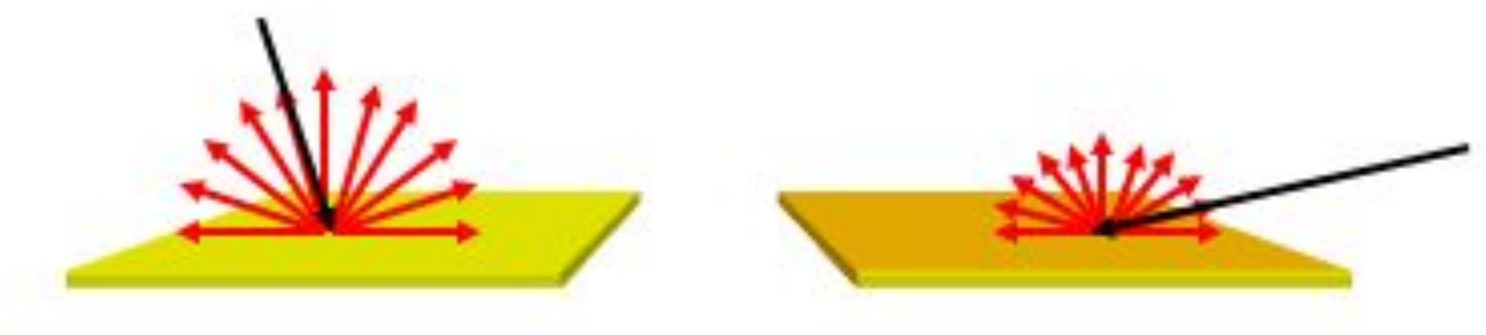
$$I_{reflected} = k_{ambient} I_a$$

$$I_a = \begin{bmatrix} I_{ar} \\ I_{ag} \\ I_{ab} \end{bmatrix}$$

I: intensity, I_a: intensity of ambient light
k_a: object spesific ambient reflection coefficient (between 0.0 and 1.0 for R, G, B)

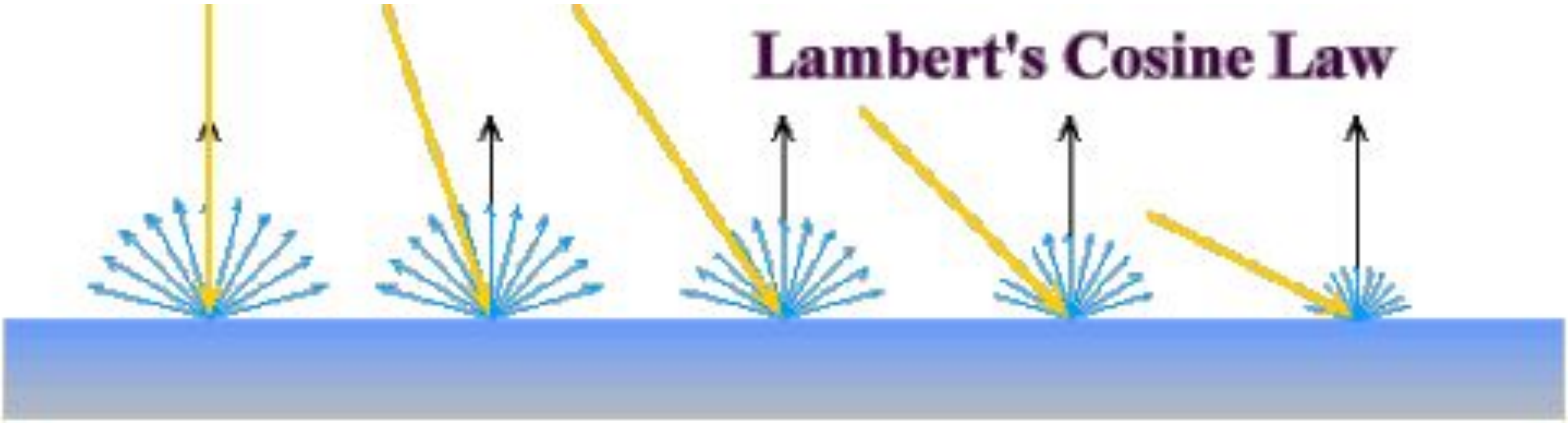
position, direction, and distance of the light source and the camera are not important for calculation of ambient light

Diffuse Reflection



- reflection in all directions
- the more perpendicular the light to the surface, the brighter the surface appears
- angle of incidence of the light on the surface is important
- if the angle increases, there is less reflection in the direction
- an ideal diffuse reflector, at the microscopic level, is a very rough surface (an incoming ray of light is equally likely to be reflected in any direction over the hemisphere)

Lambert’s Cosine Law



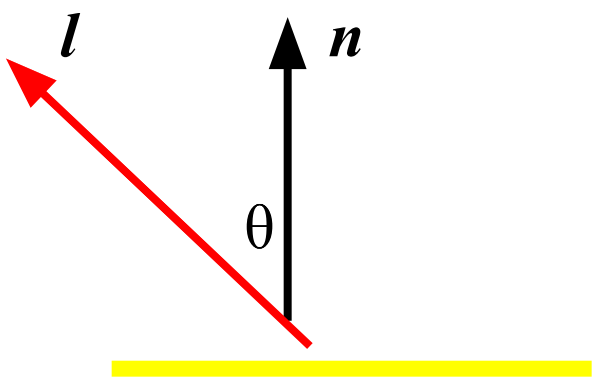
- an ideal diffuse surface scatters the beam (Lambert's law of cosine)

"In a given direction, the reflection of the energy incident on a small part of the surface is proportional to the cosine of the angle between the surface normal and the direction of the light"

- these type of surfaces are referred to as Lambert surfaces
- intensity of the reflected beam is independent of where the camera is looking from, is directly related to the position of the surface relative to the light

Modeling Diffuse Reflection

- angle between the surface normal (n) [vector perpendicular to the surface] and incident light (l): incidence angle, **θ**

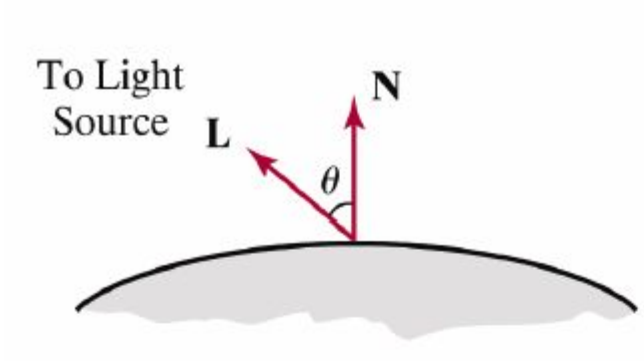


$$\mathbf{I_{diffuse} = k_d\, I_{light}\, \cos\theta}$$

$$I_{diffuse} = k_d\, I_{light}\, (n \bullet l)$$ substitute cos term here

k_d: coefficient (diffuse reflectivity of the surface at that wavelength), between 0 and 1, close to 1 for a highly reflecting surface and close to 0 for more light absorbing surfaces

Combining Ambient and Diffuse



Ambient & diffuse reflection for single point-source illumination

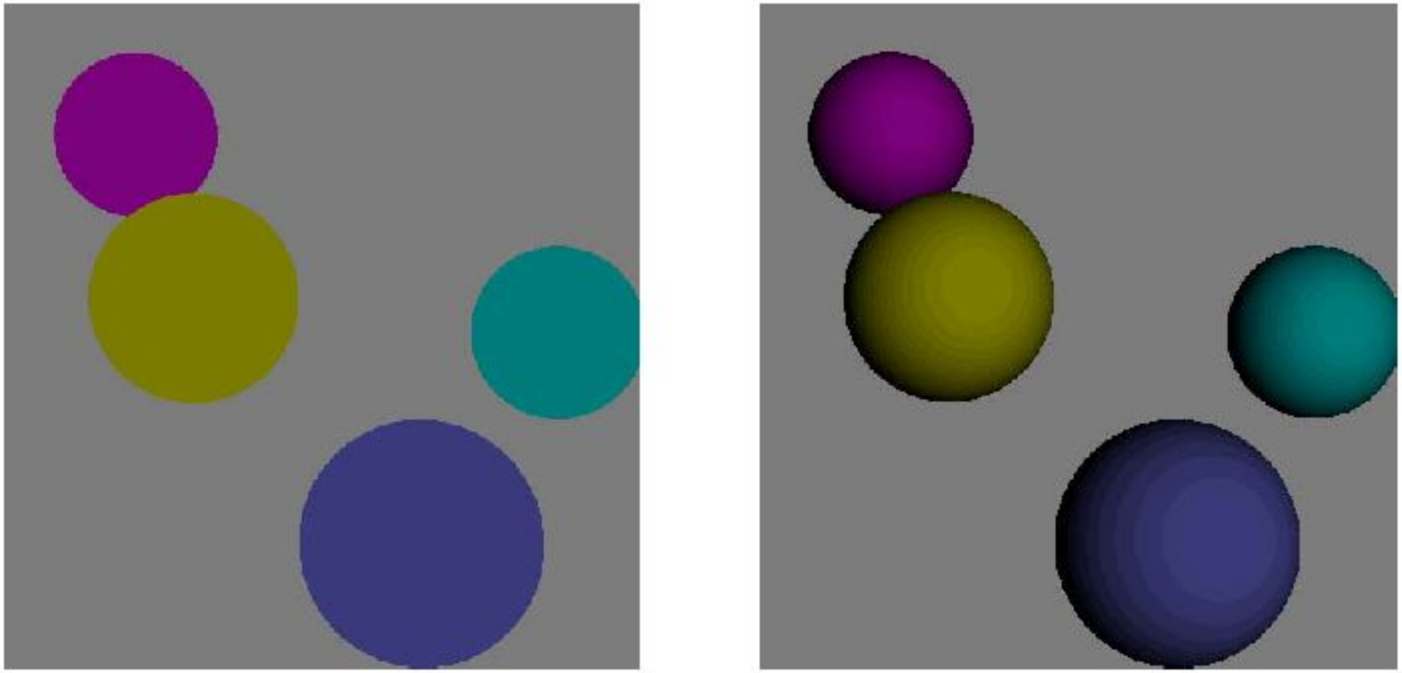
$$I_{l,diff} = \begin{cases} k_a I_a + k_d I_l (\mathbf{N} \cdot \mathbf{L}), & \text{if } \mathbf{N} \cdot \mathbf{L} > 0 \\ k_a I_a, & \text{if } \mathbf{N} \cdot \mathbf{L} \leq 0 \end{cases}$$

if Θ>=90°, diffuse reflection has no effect on the illumination of the object

L: unit direction vector to light source

$$\mathbf{L} = \frac{\mathbf{P}_{source} - \mathbf{P}_{surface}}{\left|\mathbf{P}_{source} - \mathbf{P}_{surface}\right|}$$

Ambient Light Only – Ambient Light + Diffuse Reflection



left: only ambient light.

right: image with ambient + diffuse (a little more 3D feeling)

Specular Reflection

- takes place on glossy surfaces (e.g. polished metals: car body, hard plastics: billiard ball)
- smoothness of the surface and the size of the “bright spot" are inversely proportional (the smoother the surface, the smaller the circular bright area-bright spot)
- dependent on the viewer/camera position

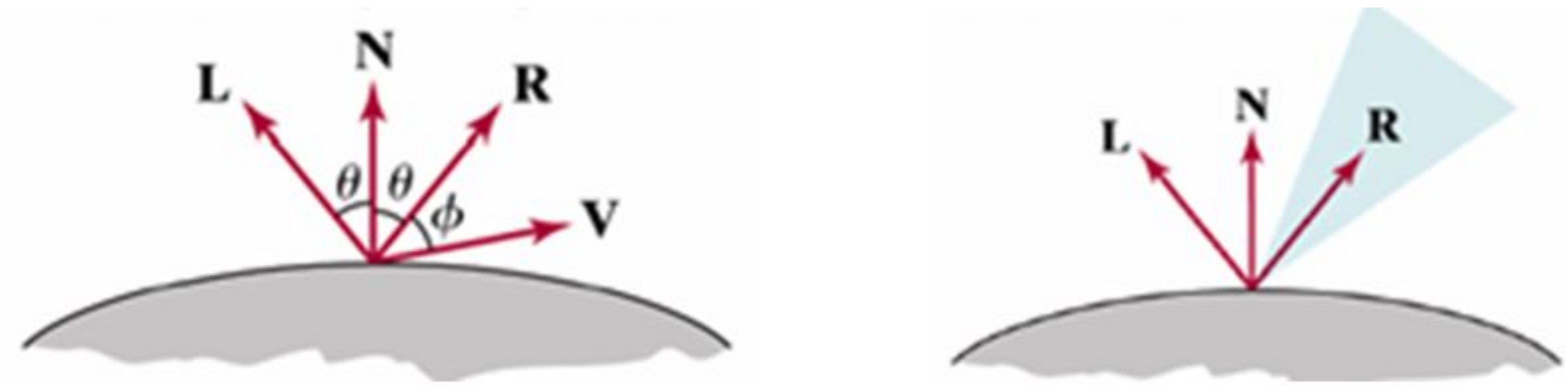
Summary:

ambient: doesn’t depend on the location of the light source or the object or the camera

diffuse: depends on the direction of the light and how perpendicular it is to the surface

specular: depends on both the direction of the light and the position of the camera

Modeling Specular Reflection - Ideal Case



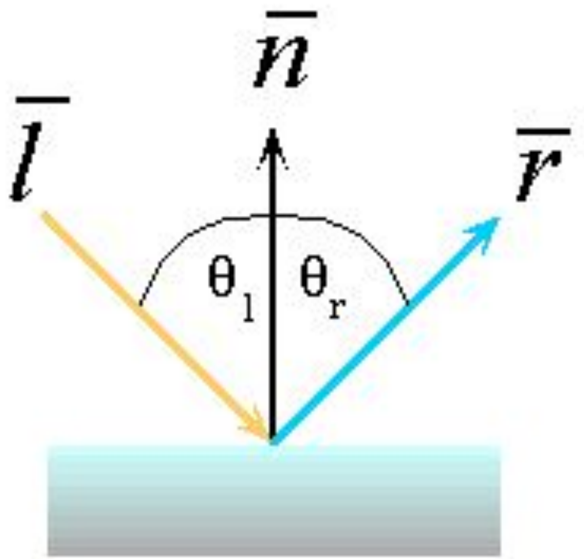
L: light direction (direction to the light source),
N: surface normal,
R: reflection direction,
V: camera direction, the direction from the surface to the camera.

On an ideal (mirror-like) surface, the angles between L and N, N and R are equal (incidence and reflection angles)

As V moves away from R the brightness decreases, as V gets closer the brightness increases (brighter object)

Modeling Specular Reflection - Ideal Case

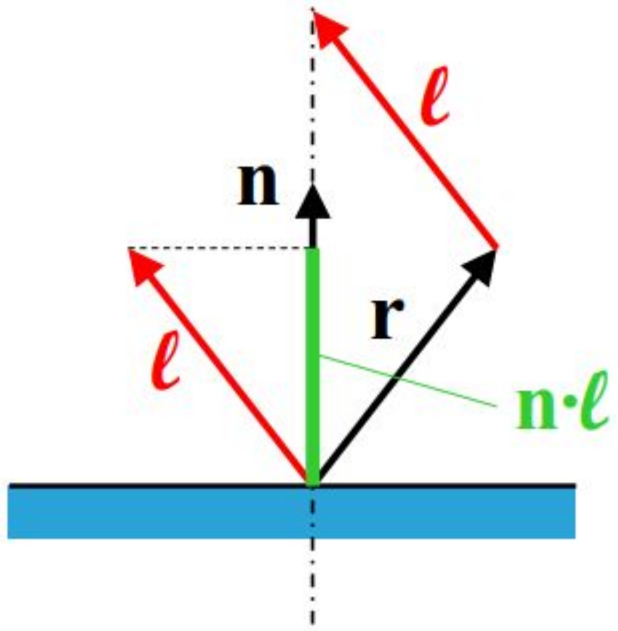
- Snell's Law (applies to specular reflection)
  - The angle that the incident light makes with the surface normal and the angle that the reflected light makes with the surface normal are opposite but of the same value and form a plane:



$$\theta_{(l)ight} = \theta_{(r)eflection}$$

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

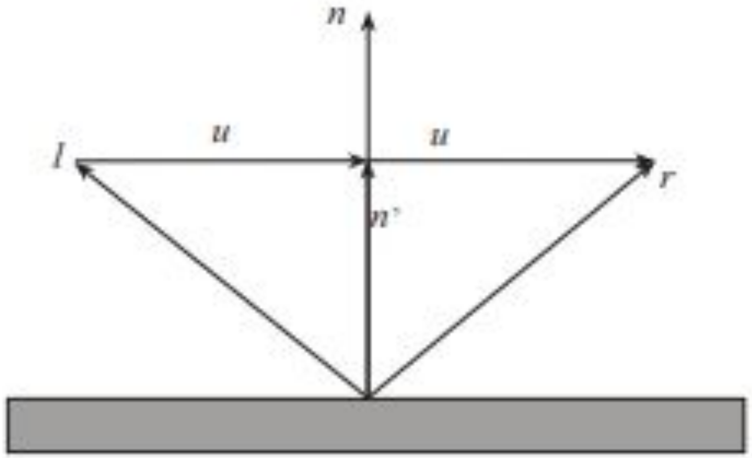
Calculation of r



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

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

Calculation of r



$$\vec{n}' = (\vec{n} \cdot \vec{l})\vec{n}$$

$$\vec{u} = \vec{n}' - \vec{l}$$

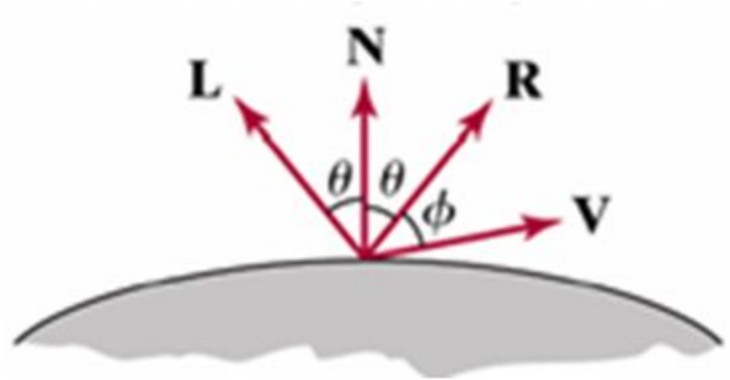
$$\vec{r} = \vec{l} + 2\vec{u} = \vec{l} + 2(\vec{n}' - \vec{l}) = 2(\vec{n} \cdot \vec{l})\vec{n} - \vec{l}$$

Calculating r with a slightly different approach

Modeling Specular Reflection - Ideal Case

$$I_{l,spec} = k_s I_l \cos^{n_s} \varphi$$

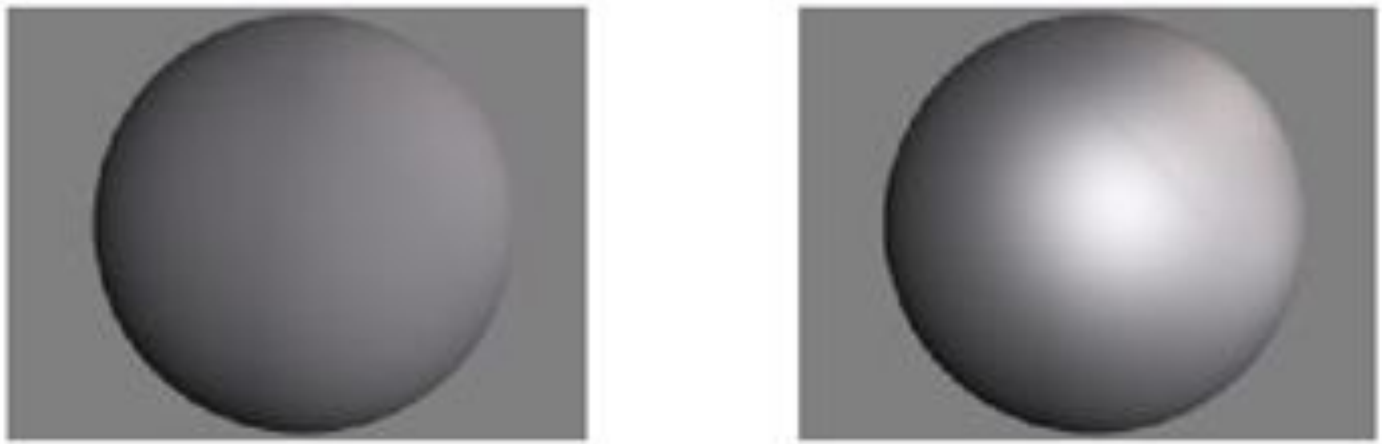
$$\cos\varphi = V \bullet R$$



k_s: expresses the surface property (like k_a for ambient, k_d for diffuse)
n_s: shows how much specular reflectance the surface has, determines whether the increase and decrease of brightness will be fast or slow, as the Φ angle changes

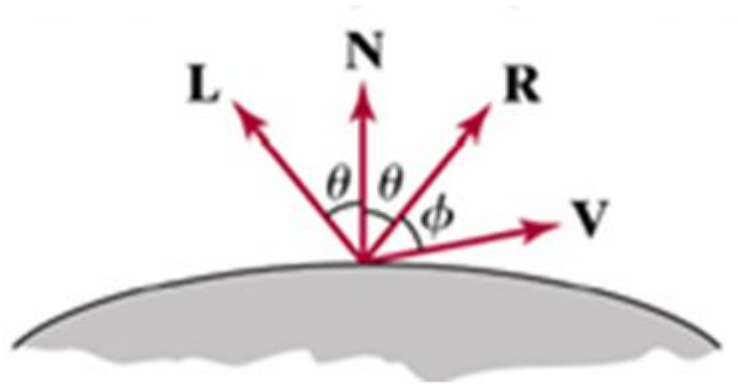
$$I_{l,spec} = \begin{cases} k_s I_l (\mathbf{V}\cdot\mathbf{R})^{n_s}, & \text{if} \quad \mathbf{V}\cdot\mathbf{R} > 0 \quad \text{and} \quad \mathbf{N}\cdot\mathbf{L} > 0 \\ 0.0, & \text{if} \quad \mathbf{V}\cdot\mathbf{R} < 0 \quad \text{and} \quad \mathbf{N}\cdot\mathbf{L} \leq 0 \end{cases}$$

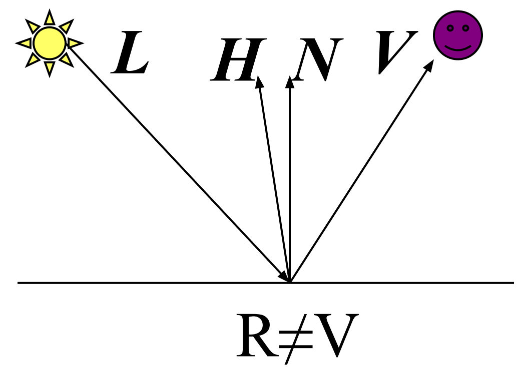
Diffuse On – Diffuse + Specular



left: diffuse reflection only
right: effect of the bright reflection. Image with diffuse reflection + bright reflection (specularity: bright spot)

Modified Phong (Blinn) Model





$$I_{l,spec} = k_s I_l (V \bullet R)^{n_s}$$

$$\mathbf{H} = (\mathbf{L} + \mathbf{V}) / |\mathbf{L} + \mathbf{V}|$$

$$k_s I_l (\mathbf{H} \bullet \mathbf{N})^n$$

- simplifies specular model
- instead of R (reflected vector), H (the halfway vector) is calculated
- the closer H to N, the closer V to R
- use angle between H and the surface normal N (or scalar product of these two unit vectors)

Putting Them Together

- bring the ambient, diffuse and specular components together linearly (according to the Blinn lighting model)
- k coefficients can be taken differently for each color channel
- L.N and H.N will remain the same for each color channel
- below formula is used for a light source, if there is more than one light source in the environment, we make these calculations for each light source (then the results are summed up)
- these calculations are made for each point of the object (for acceleration purposes, if the light sources or the camera are far away, we keep them constant even if the points change)

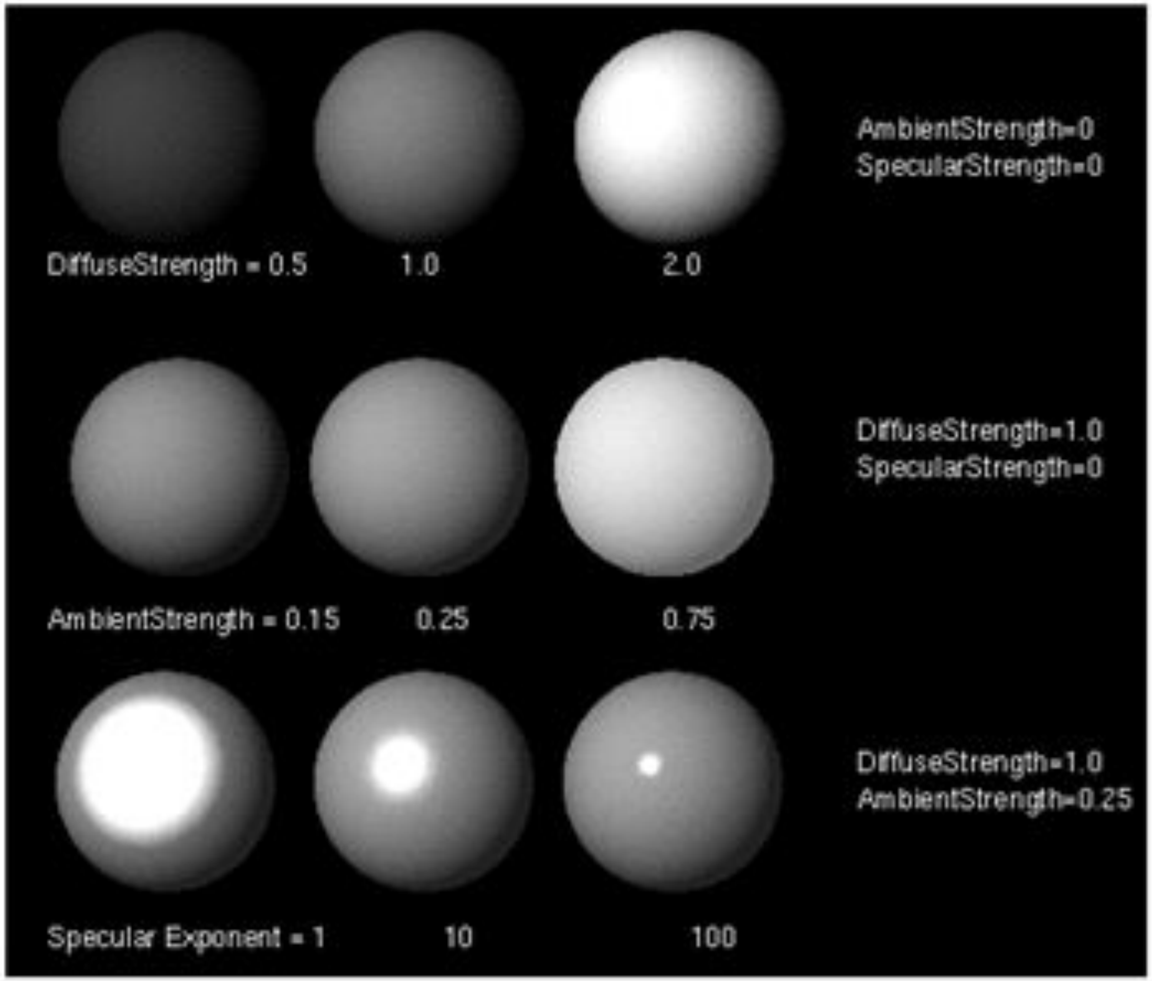
$$I = k_a I_a + I_l \left(k_d (\mathbf{L} \bullet \mathbf{N}) + k_s (\mathbf{H} \bullet \mathbf{N})^n \right)$$

$$I_r = k_{a,r} I_{a,r} + I_{l,r} \left(k_{d,r} (\mathbf{L} \bullet \mathbf{N}) + k_{s,r} (\mathbf{H} \bullet \mathbf{N})^n \right)$$

- Results for different diffuse coefficients with constant ambient and specular coefficients (top)

- diffuse and specular are constant, ambient coefficients have changed (midline)

- diffuse and ambient coefficients are constant, specular coefficients have changed (bottom)



Summary:
- light reflected consists of ambient, diffuse and specular components
- ambient component is independent of surface and camera position
- diffuse is independent of the camera but depends on the angle of incidence of the light on the surface (position of the light source and the surface relative to each other is important)
- specular is like diffuse and the position of the camera is also important

Polygon Rendering Methods

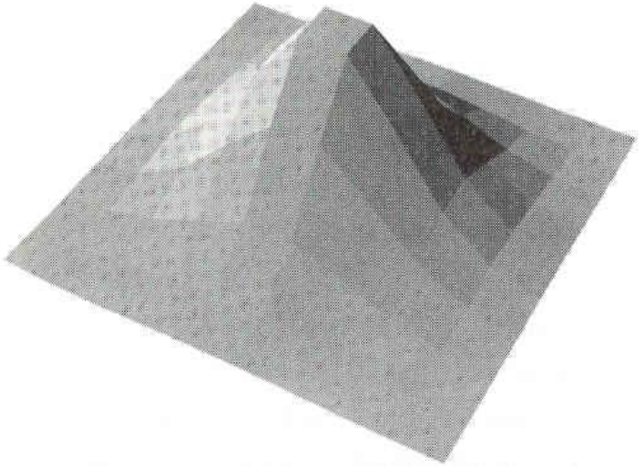
- objects are made up of polygons
- lighting calculations are made for points and vertices (pixel color value is calculated)
- for which points in the polygons are we going to make these calculations?
- three methods below require less to more computation, respectively.

- Constant Intensity/Flat Shading
- Gouraud Shading (Intensity Interpolation)
- Phong Shading (Normal Vector Interpolation)

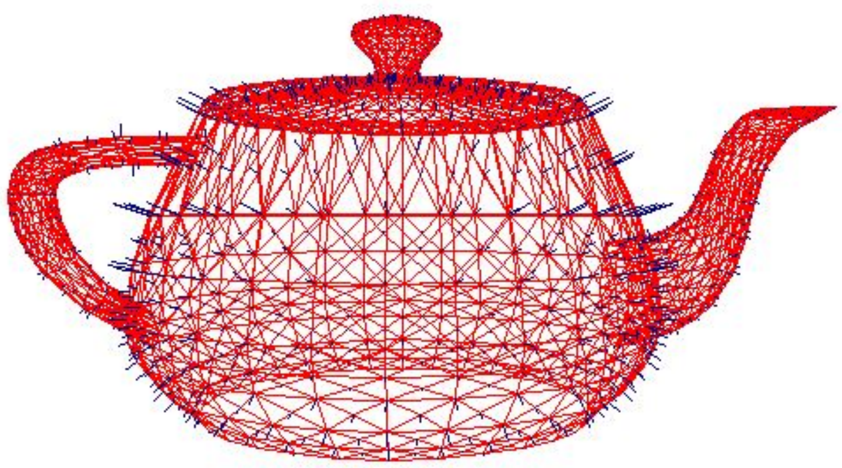
Flat Shading

- the same color is always used throughout the polygon (no smooth apperance, sharp lines)
- it is assumed that the light source and camera are at infinite distances
- throughout a polygon, all lighting parameters are constant
- the simplest method (calculation of the surface normal and other calculations are done once for a polygon)

Same color for poygon surfaces

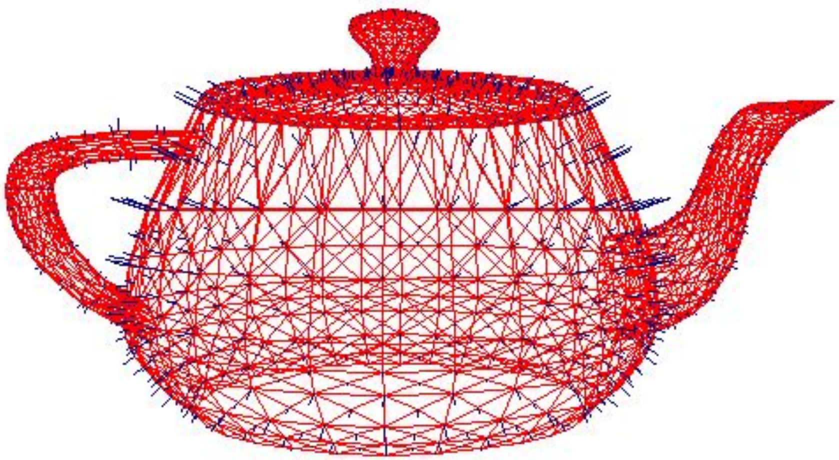


Gouraud Shading (Intensity Interpolation)



- roughly, the color at some points of a surface is calculated, and at other points of the surface, the colors are interpolated from the found colors
- final image is softer, but there are still artifacts

Phong Shading
(Normal Vector Interpolation)



- vertex normals are interpolated, color calculation is made for each point
- most costly method, there are many operations
- phong shading is not phong illumination model
- all of these methods are approaches, they don’t exactly reflect reality
- fixed function pipeline in OpenGL: the Blinn-Phong model was standard for a long time
- now it is possible to be more comfortable and flexible in choosing the model to be used
- Light sources, surface coefficients, exponential coefficient of specular reflection are defined and light calculations are made in shaders