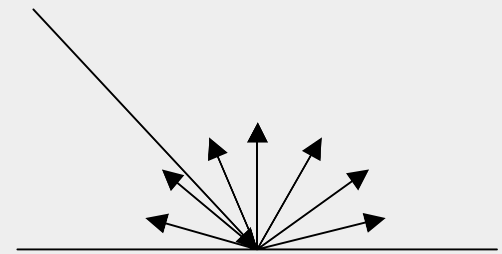
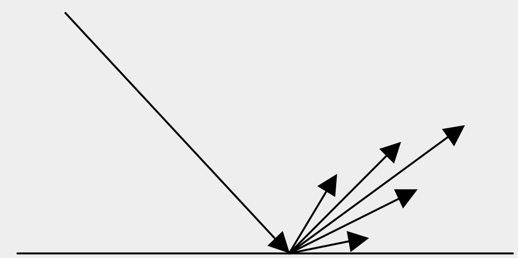
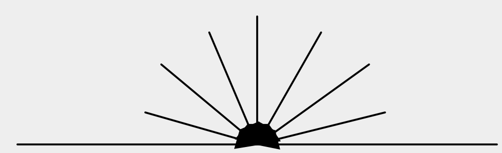
“Standard” Lighting Model

- Consists of three terms linearly combined:
  - *Diffuse component* for the amount of incoming light reflected equally in all directions
  - *Specular component* for the amount of light reflected in a mirror-like fashion
  - *Ambient term* to approximate light arriving via other surfaces





- This is very simple approximation
  - particularly good for plastic
  - particularly good for metal
  - That’s why CG images tend to look like plastic and metal

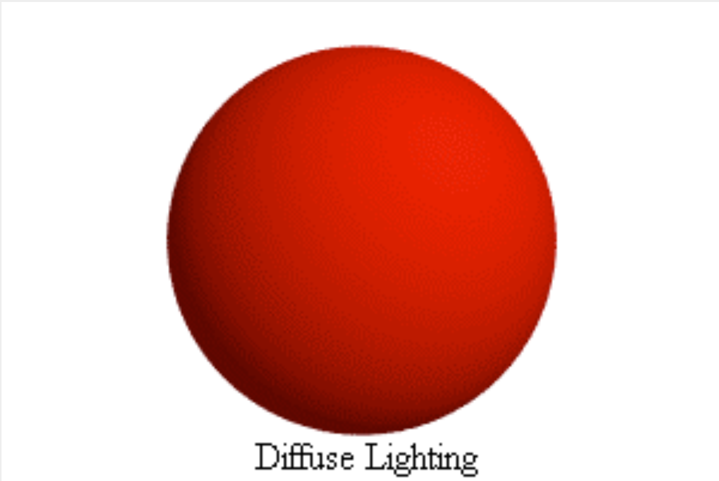


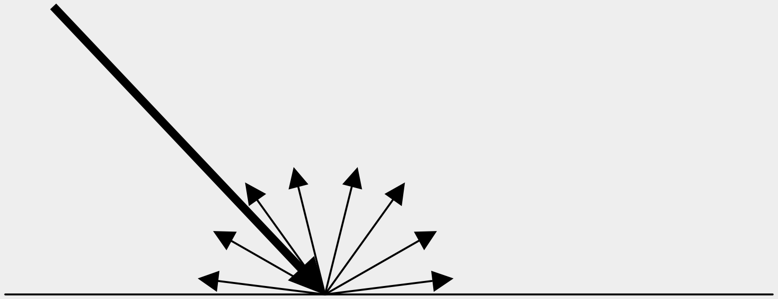
Reflectivity

- White sheet of paper might reflect 95% of incident light
- A mirror might reflect 95% of incedent light
- Yet, these two things look completely different:
  - They reflect light in different directions
  - The paper is a *diffuse* reflector
  - The mirror is a *specular* reflector

Diffuse Reflection

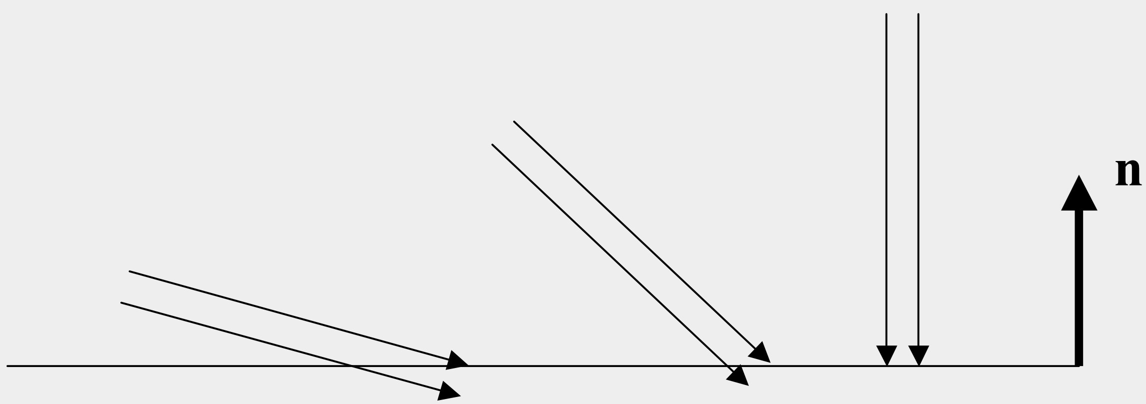
- An ideal diffuse reflector receives light from some direction and bounces it uniformly in all directions
  - very rough at microscopic level
- Diffuse materials have a dull or matte appearance
  - example: chalk



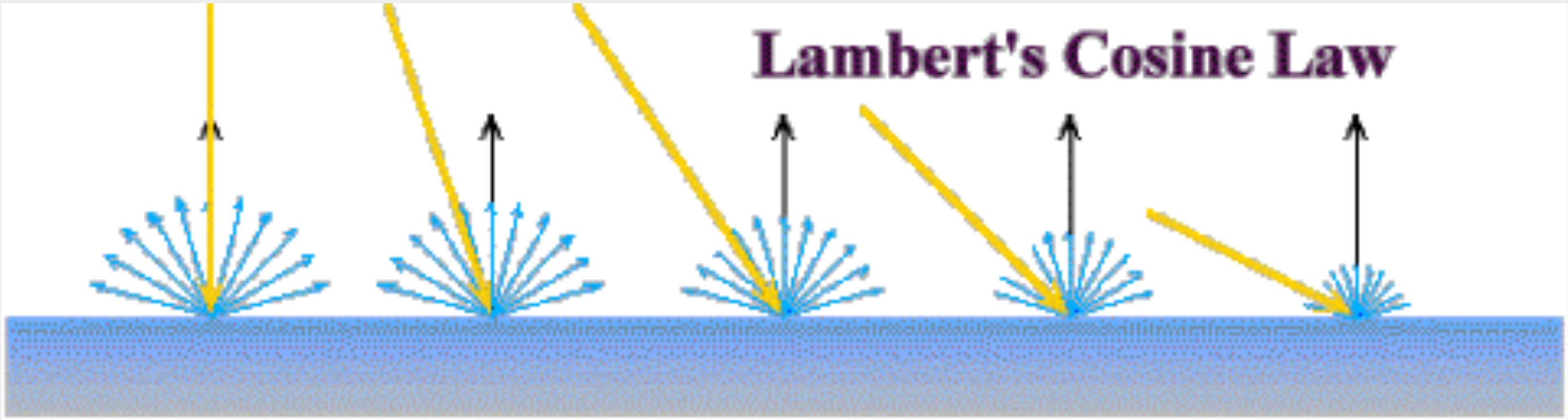


Diffuse Reflection

- Assume a beam of parallel rays shining on the surface
- Consider area of the surface covered by the beam
  - varies based on the angle between the beam and the normal
  - The larger this area, the less incident light per area
  - The incident light per unit area is proportional to the cosine of the angle between the normal and the light rays
- Object darkens as normal turns away from light
  - This is known as *Lambert's cosine law*
  - Diffuse surfaces AKA *Lambertian* surfaces



Lambert’s Cosine Law



Diffuse Reflection

Given:

$\vec{\mathbf{n}}$ = surface unit normal

$\vec{\mathbf{L}}$ = unit vector pointing towards the light

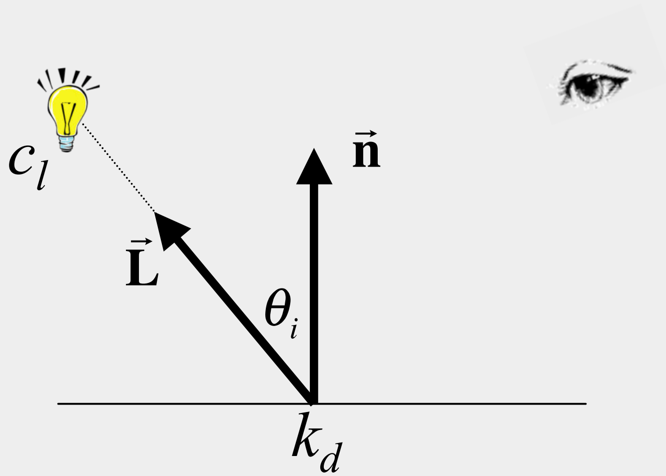
θ_i = angle of incidence, between $\vec{\mathbf{L}}$ and $\vec{\mathbf{n}}$

k_d = material diffuse reflectance coefficient (0.0 to 1.0)

c_l = color (intensity) of the light source

Diffuse color:

$$c_d = k_d c_l \cos(\theta_i) = k_d c_l (\vec{\mathbf{L}} \cdot \vec{\mathbf{n}})$$



- Notes:
  - Depends on light and normal directions
  - Doesn't depend on eye position
    - diffuse reflection is same in every direction
  - Don't want to illuminate from rear
    - use $k_d c_l \max(\vec{\mathbf{L}} \cdot \vec{\mathbf{n}}, 0)$

Diffuse Lighting Examples

- A Lambertian sphere at several different lighting angles:

- Diffuse lighting provides visual cues
  - indicates 3D depth
  - indicates surface curvature

Multiple Lights

- Can have many light sources in a scene
  - Light (generally) behaves additively
  - Add up the contribution of each light

$$c = k_d \sum c_{l_i} (\vec{\mathbf{L}}_i \cdot \vec{\mathbf{n}})$$

Ambient Light

- In the real world, light gets bounced all around the environment
  - Resulting light illuminates surface from every direction.
  - Global illumination techniques attempt to compute this. Complex.
- Simple approximation (hack): *Ambient light*
  - Assume net effect is a constant color shining from every direction
  - Add to the net color, attenuated by reflectance coefficient

$$c = k_a c_a + \sum k_d c_{l_i} \left(\vec{\mathbf{L}}_i \cdot \vec{\mathbf{n}} \right)$$

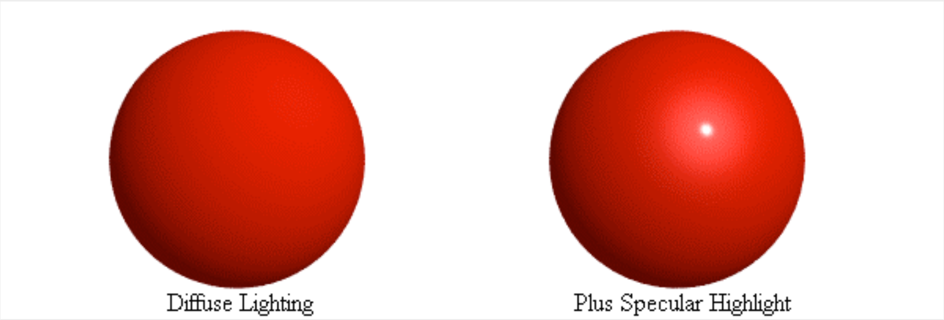
- Effect of ambient light:
  - Keeps unlit areas from going completely black
  - Makes things look flatter
    - with ambient and no diffuse: object has solid color, is completely 2D
  - k_a or c_a usually small (.1 or less)

Specular Reflection

- Shiny surfaces exhibit *specular reflection*
  - Polished metal
  - Glossy car finish
  - Plastics

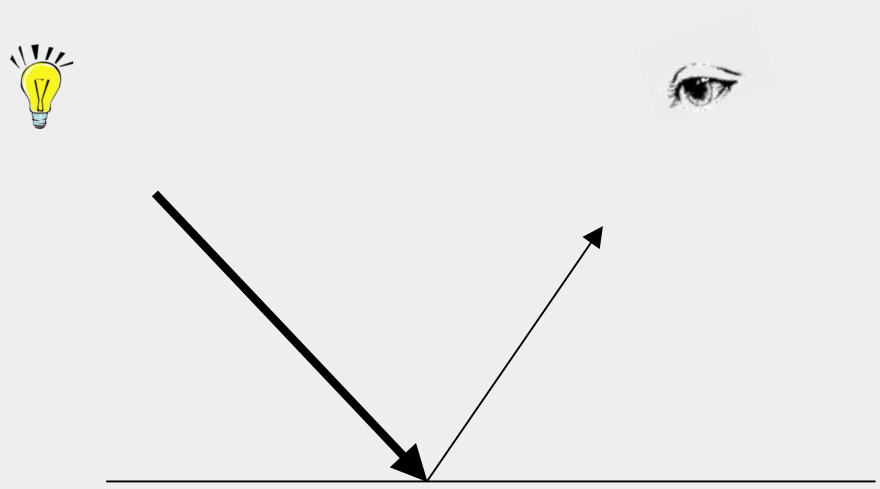
- A light shining on a specular surface causes a bright spot:
  - known as a *specular highlight*
  - essentially, a rough reflection of the light source

- Highlight location depends viewer position relative to surface & lights



Specular Reflection

- An ideal specular reflector = mirror
  - perfectly smooth surface
  - bounces an incoming light ray in a single direction
  - angle of incidence equals the angle of reflection

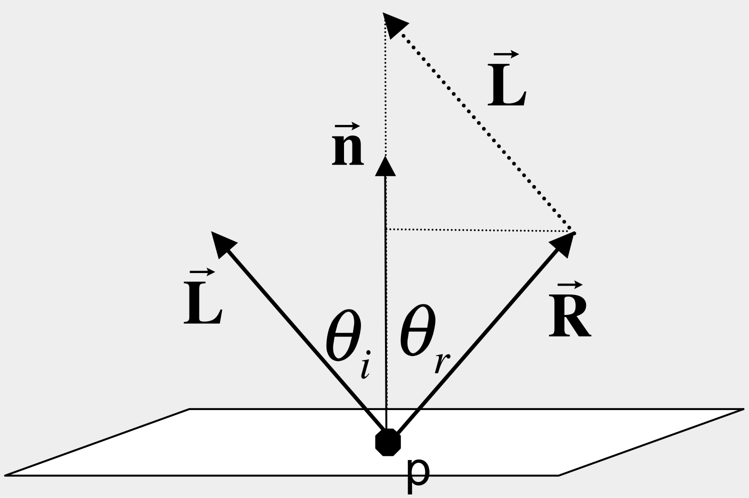


Law of Reflection

- Angle of reflectance = angle of incidence

$$\vec{\mathbf{R}} + \vec{\mathbf{L}} = 2\cos\theta\ \vec{\mathbf{n}} = 2\left(\vec{\mathbf{L}} \cdot \vec{\mathbf{n}}\right)\vec{\mathbf{n}}$$

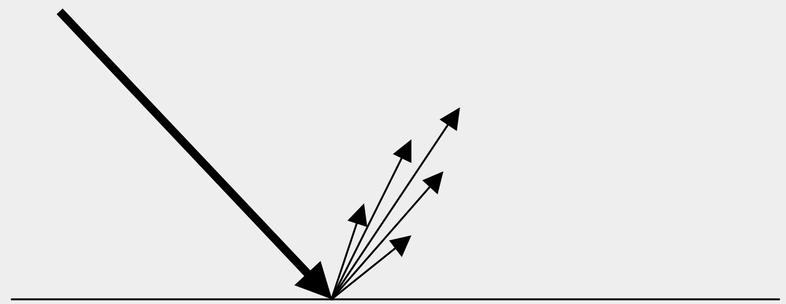
$$\vec{\mathbf{R}} = 2\left(\vec{\mathbf{L}} \cdot \vec{\mathbf{n}}\right)\vec{\mathbf{n}} - \vec{\mathbf{L}}$$



$$\theta_r = \theta_i$$

Specular (Glossy) Reflection

- Many materials not quite perfect mirrors
  - Glossy materials look shiny and will show specular highlights
  - In CG, this is sometimes referred to as glossy reflection

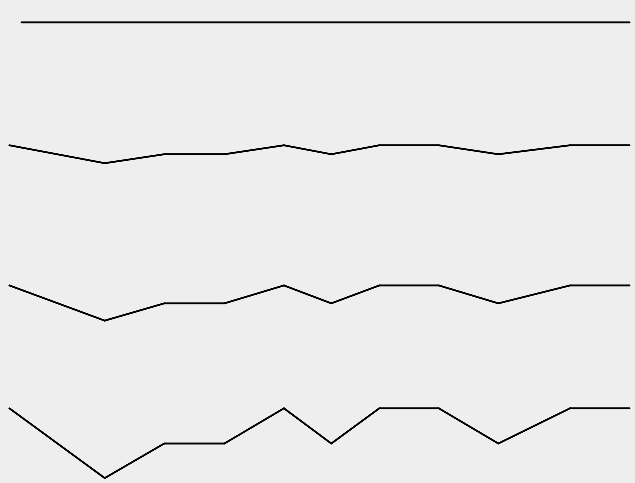


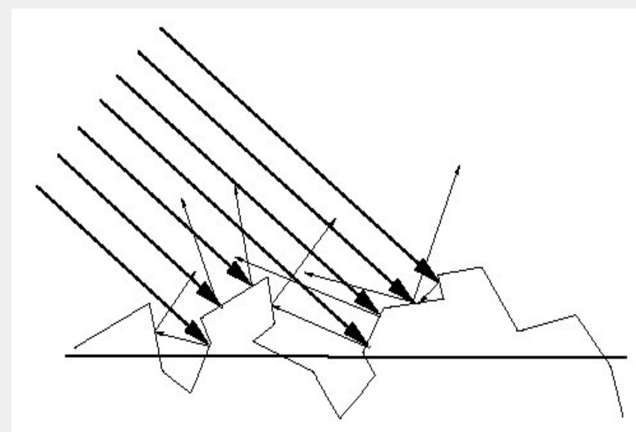
- Many formulations for this
  - First: most basic and famous: Phong lighting model (1973)
  - Then: most popular: Blinn lighting model (1977)

Shiny materials

- The surface roughness will vary from material to material
  - Smooth surfaces have sharp highlights
  - Rougher surfaces have larger, more blurry highlights
- Assume surface composed of microfacets with random orientation
  - Smooth surfaces: microfacet normals very close to surface normal
  - Rough surfaces: microfacet normals are spread around more
    - on average, microfacet normals close to surface normal

- Polished:
- Smooth:
- Rough:
- Very rough:





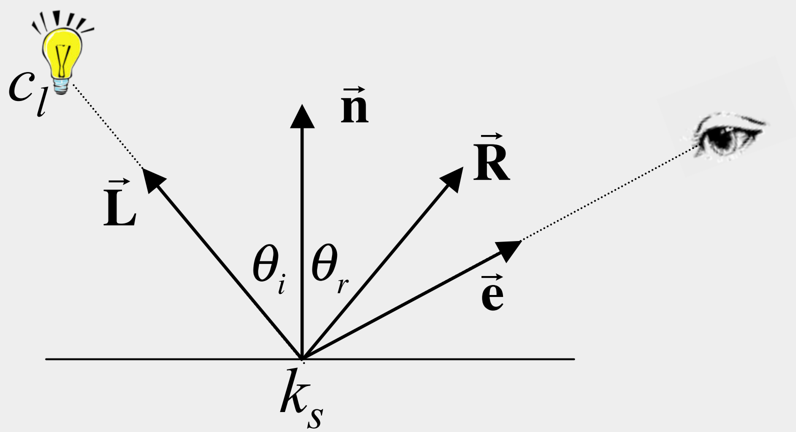
Empirical observation

- In general, we expect most reflected light to travel in direction of exact reflection
- But because of microscopic surface variations, some light may be reflected in a direction slightly off the ideal reflected ray
- So:
  - Most reflected light in direction of ideal reflection
  - Brightest when eye vector (view vector) is aligned with reflection
  - Intensity decreases as eye vector angle from reflection increases
  - Use dot product of eye vector with reflection vector

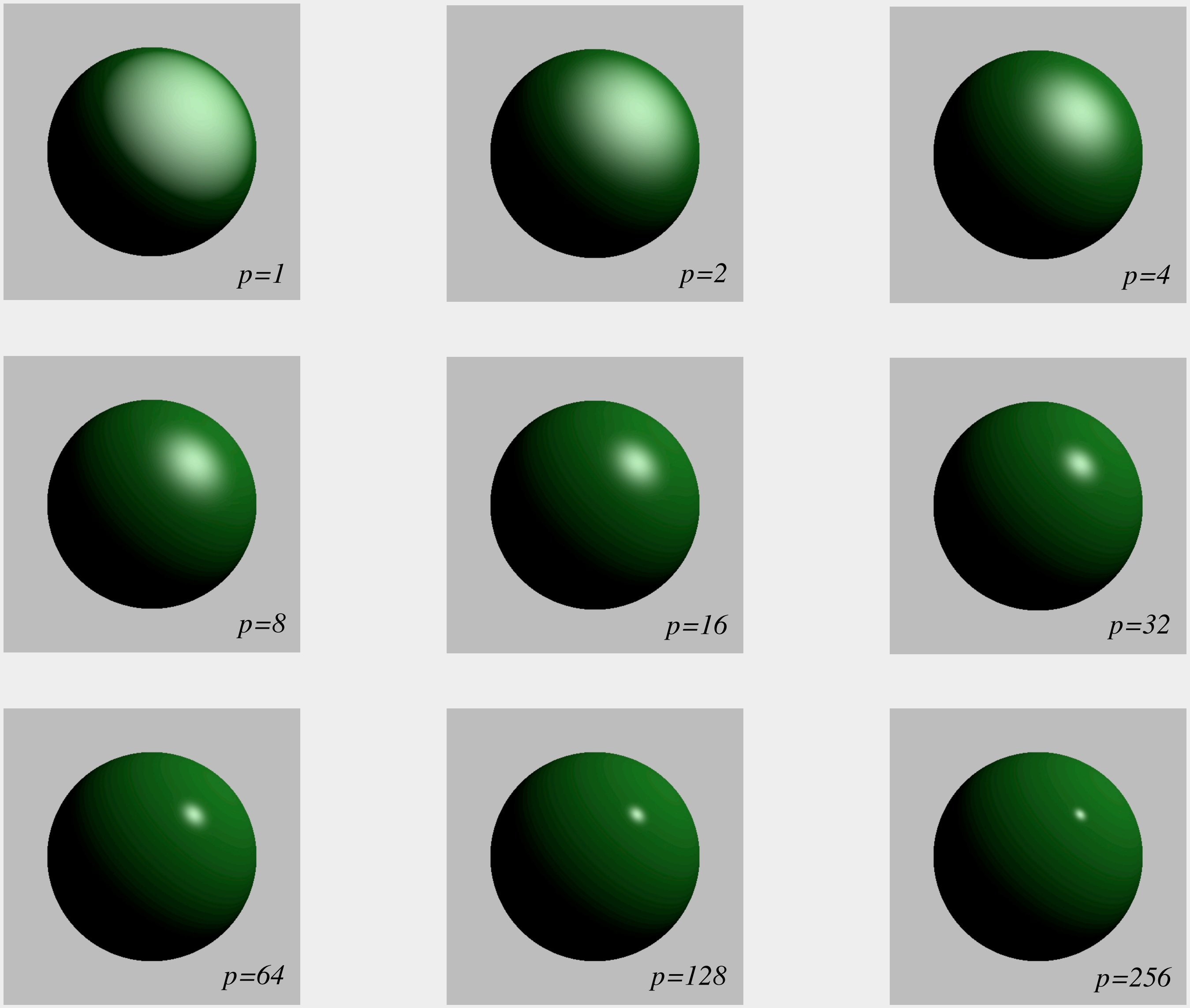
Phong Lighting Model

$$c_s = k_s c_l \left(\vec{\mathbf{R}} \cdot \vec{\mathbf{e}}\right)^p$$

- parameters:
  - specular reflectance coefficient, k_s
  - Phong Exponent p controls the apparent size of the specularity
    - Higher p, smaller highlight



Phong Lighting Model Examples



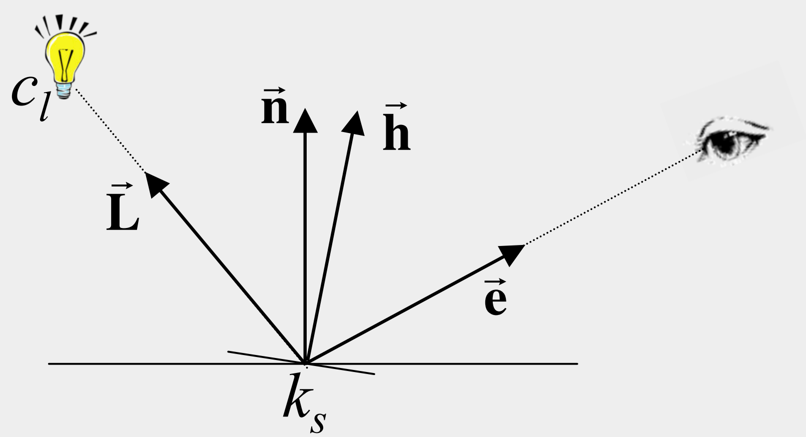
Blinn Lighting Model

- Define the unit-length "halfway" vector $\vec{\mathbf{h}}$,

  halfway between $\vec{\mathbf{L}}$ and $\vec{\mathbf{e}}$:

$$\vec{\mathbf{h}} = \frac{\vec{\mathbf{L}} + \vec{\mathbf{e}}}{\left|\vec{\mathbf{L}} + \vec{\mathbf{e}}\right|}$$

- $\vec{\mathbf{h}}$ represents the normal of a microfacet we would need to have oriented so as to reflect the light to the eye



Blinn Lighting Model

- Microfacets roughly line up with surface
- The farther $\vec{\mathbf{h}}$ is from $\vec{\mathbf{n}}$, the less we expect microfact to align
  - So use cosine of angle between them:

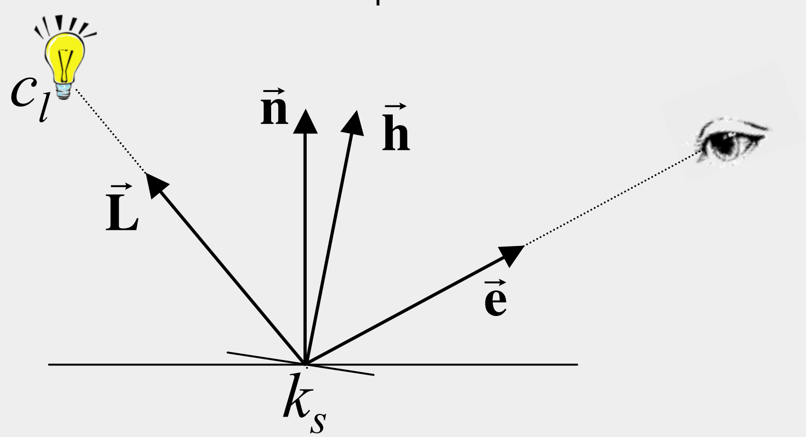
$$c = k_s c_l \left(\vec{\mathbf{h}} \cdot \vec{\mathbf{n}}\right)^s$$

- Very similar in effect to Phong model

- Optimization due to Schick (1994):

  replace t^s with $\dfrac{t}{s - st + t}$

  - Faster than exponent. Result is different but acceptable

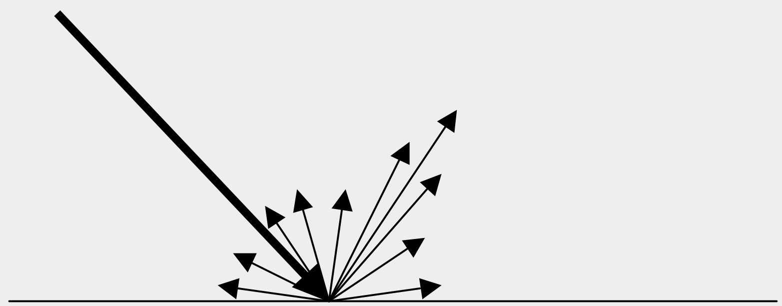


Complete Blinn Lighting Model

- Add to ambient and diffuse
- Add specular contribution for each light

$$c = k_a c_a + \sum c_{l_i} \left(k_d \left(\vec{\mathbf{L}}_i \cdot \vec{\mathbf{n}} \right) + k_s \left(\vec{\mathbf{h}}_i \cdot \vec{\mathbf{n}} \right)^s \right)$$

- It appears in a few slightly different forms and in a wide variety of notations…



Note on color

$$c = k_a c_a + \sum c_{l_i} \left(k_d \left(\vec{\mathbf{L}}_i \cdot \vec{\mathbf{n}} \right) + k_s \left(\vec{\mathbf{h}}_i \cdot \vec{\mathbf{n}} \right)^s \right)$$

- Do this in parallel for R,G,B
- Coefficients ka, kd, ks can be different for each of R,G,B
  - This defines the material ambient color, *diffuse color*, and *specular color*.
  - Other (expensive) terms in expression are shared for each of R,G,B
- Generally, use ambient color = diffuse color
- For metals, specular color = diffuse color
  - highlight is color of the material
- For plastics, specular color = white
  - highlight is the color of the light

Note on normals and spaces

- Lighting depends on angles between normals, vectors
- Must be in space that preserves angles
  - World Space or Camera Space
  - *Not* normalized view space: perspective doesn't preserve angles
- Conveniently, we can put world-space normals as per-vertex data
  - Doesn't get transformed by projection.

- But remember, when taking normals from object to world space:
  - if world transform has nonuniform scale, normals must use inverse-transpose
  - if world transform has uniform scale, normals must be renormalized
  - if world transform has no scales, normals transform like vectors