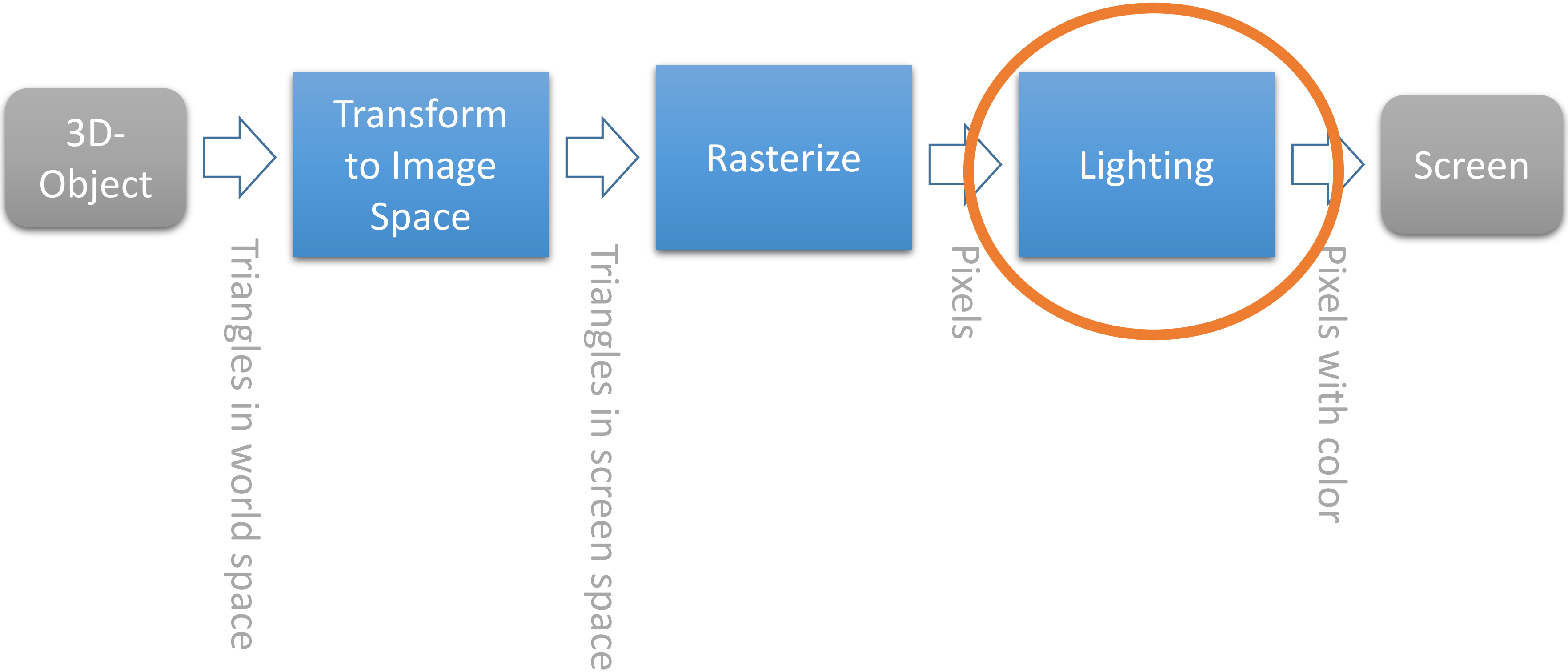
Lecture #9

Lighting

Computer Graphics
Winter Term 2016/17

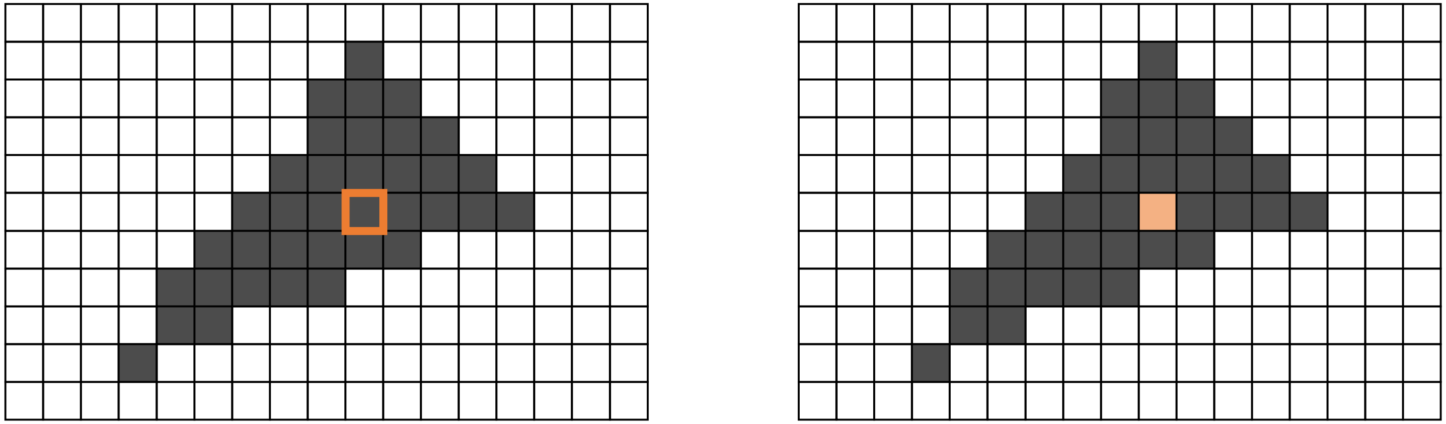
Marc Stamminger / Roberto Grosso

Lighting


3D-Object
Triangles in world space
Transform to Image Space
Triangles in screen space
Rasterize
Pixels
Lighting
Pixels with color
Screen

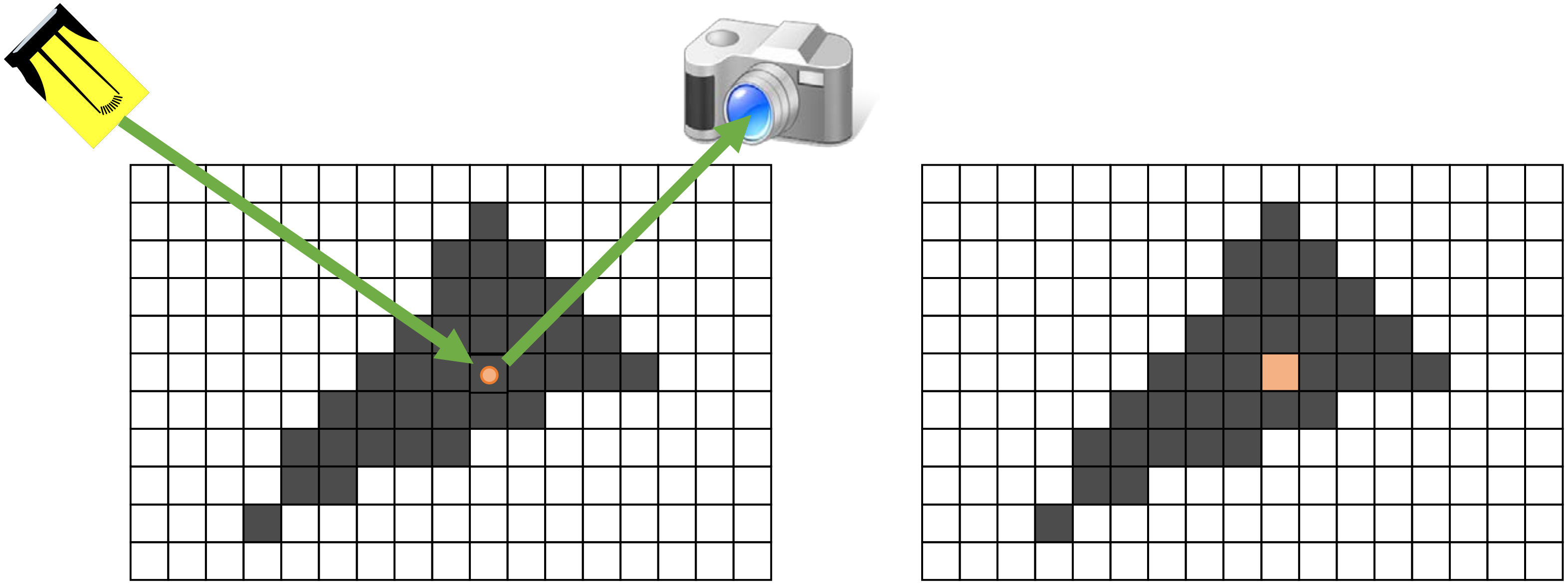
Lighting

- After rasterization: compute color of a particular pixel
(= Shading, later we will learn difference between shading and lighting)



Lighting

- Find position of pixel's center in 3D world
- Simulate lighting of this point → color of pixel



Lighting

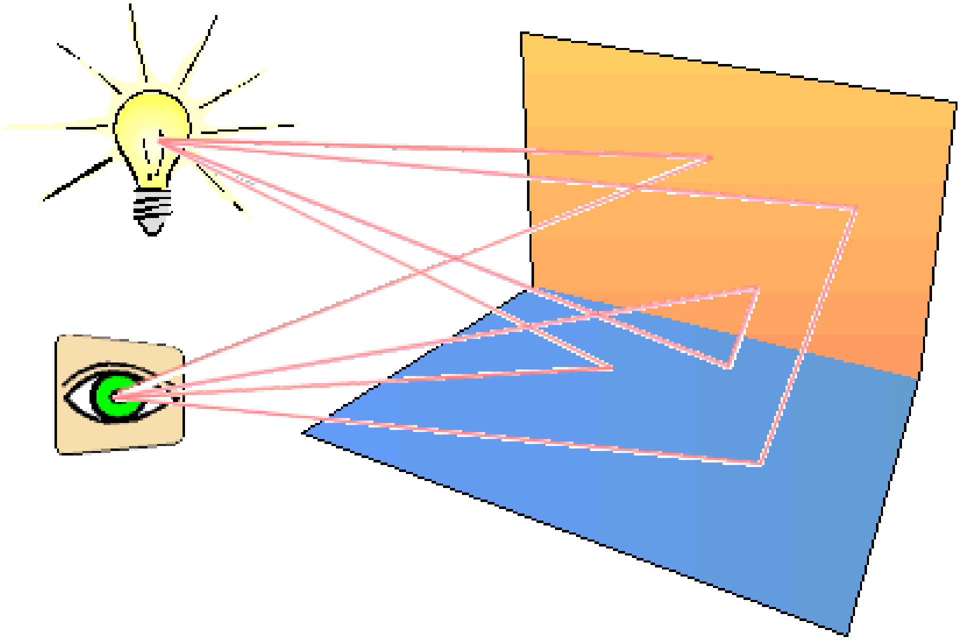
- Lighting of pixel is computed in the **Vertex and/or Fragment Shader**
- Input:
  - Light source positions, colors, material parameters
    → uniforms
  - Surface position, surface normal (= orientation of surface)
    → vertex attributes → vertex shader
    → varying → fragment shader

```
uniform vec3 kdiff;
uniform vec3 l;
varying vec3 n;

void main (void)
{
   gl_FragColor.rgb = kdiff * max(0,dot(n,l));
}
```

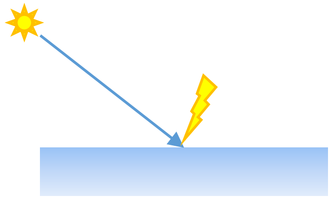
Introduction

- Lighting
  - Simulation of light propagation: Complex mathematical problem, computation intensive
  - Use approximations or heuristic models to make the scene look as real as possible, but still achieve reasonable rendering times.

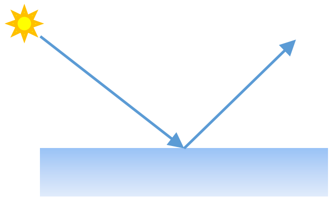


Introduction

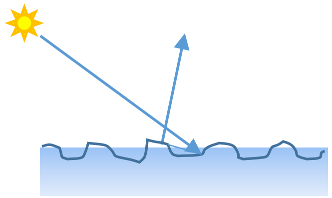
- Physics:
  When light hits a surface it may either be
  - ***Absorbed***



  - ***Reflected***: input angle = output angle

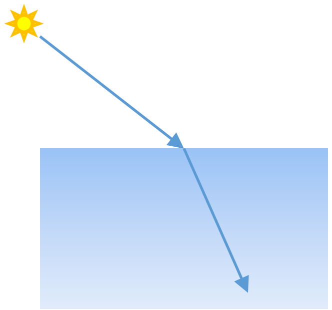


  - ***Scattered***

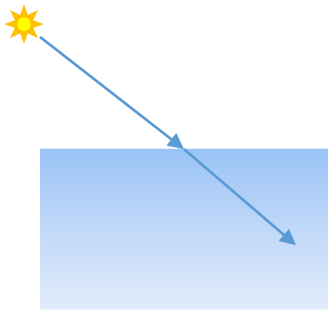


Introduction

- Physics: When light hits a surface it may either be (cont.)



  - ***Refracted**:* changing direction according to Snell's law



  - ***Transmitted***: passing through without changing direction

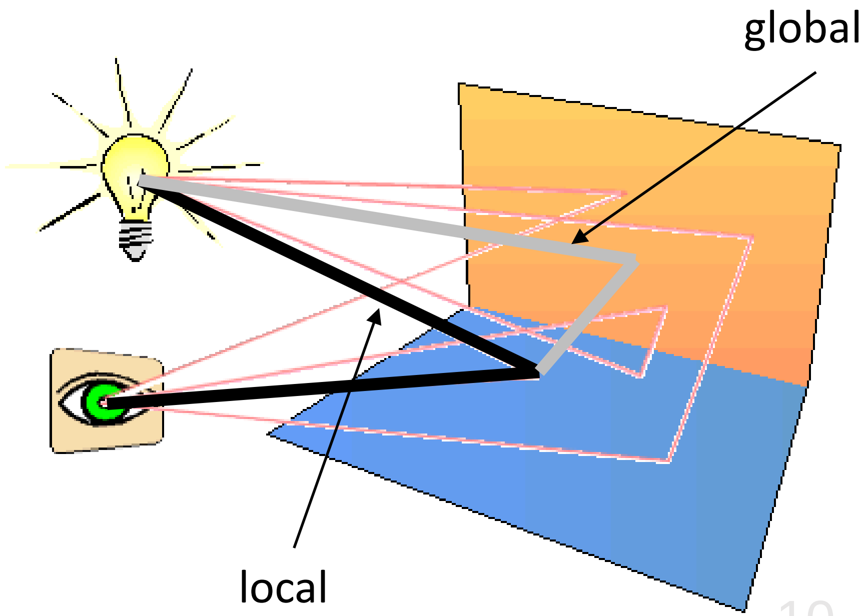
Lighting

- More general: given
  - A point x on an object
    - 3D-position $x = (x_1, x_2, x_3) \in \mathbb{R}^3$
    - Surface normal $n = (n_1, n_2, n_3) \in \mathbb{R}^3$, $\|n\| = 1$
  - A number of light sources L_i with intensities l_i
    - Point Lights with positions $p_i \in \mathbb{R}^3$
    - Or parallel lights with directions $d_i \in \mathbb{R}$^3
  - And a viewer position v

- → Determine color of point x lit by lights L_i when looked at from viewer position v

Lighting

- **Local illumination:**
  Calculate lighting by just considering the influence of the light sources and neglecting the influence of other scene objects:
  - no shadows
  - no indirect (reflected from other objects) light
  - Fast but lacks important effects
  - Example: OpenGL based rendering



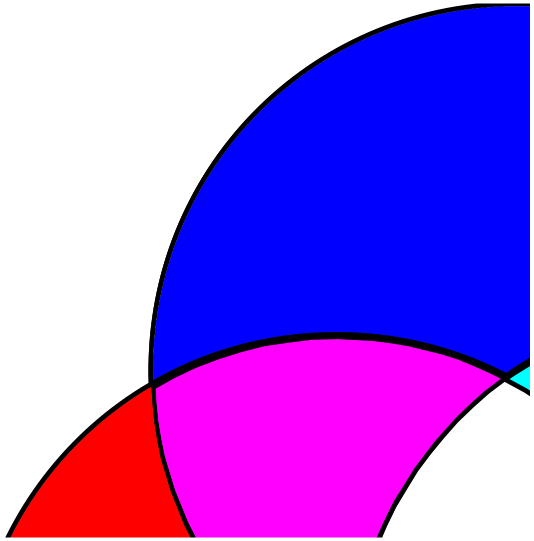
Lighting

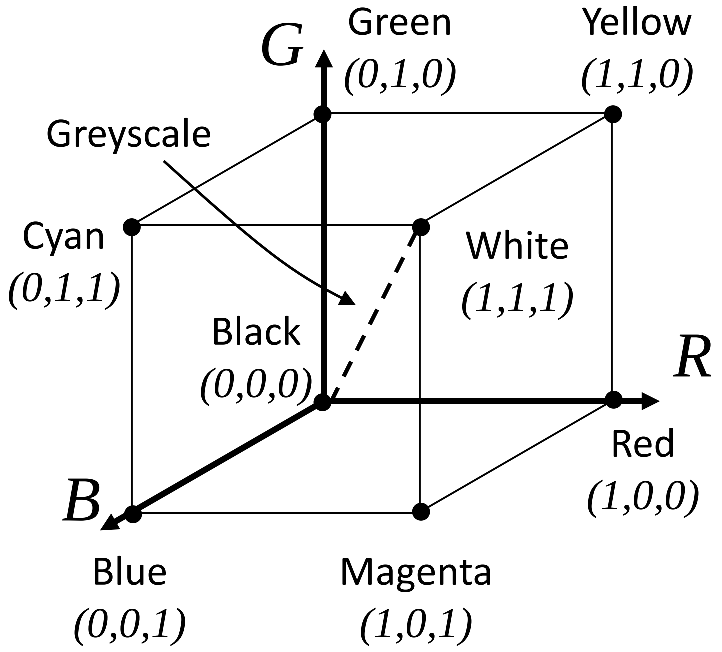
- **Global illumination**
  - Light exchange between objects and light sources
  - Light exchange between objects themselves, indirect illumination.
  - Area light sources
  - Shadows and soft-shadows
  - Slow but higher quality

- → later: ray tracing

Colors

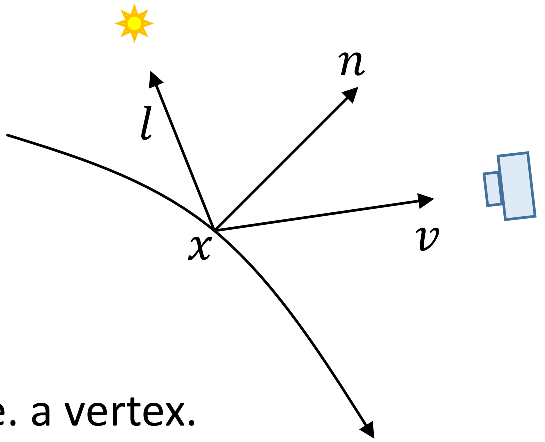
- For now, we describe color using the RGB color model (remember lecture #2)
  → all light values are (r, g, b)-triples





Local Illumination

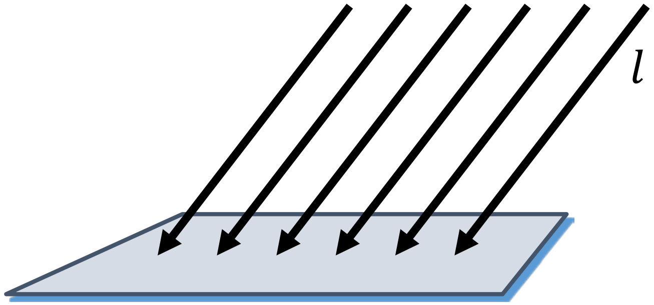
- In this chapter
  - Local illumination
  - Mostly used model: **Phong lighting**



- Problem statement
  - Compute light from light source to a point on a surface, i.e. a vertex.
  - Compute reflected light from this point to the camera

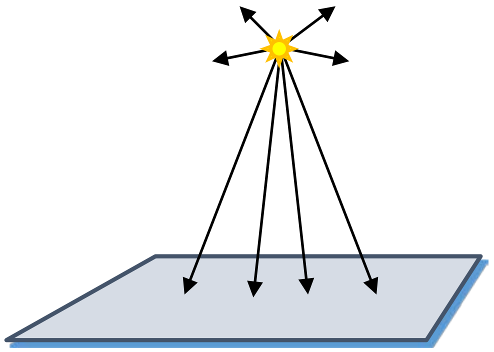
Directional Light Sources

- Directional light source
  - Simplifying assumption of only parallel rays from source
    - As if the source is infinitely far away from the surfaces in the scene
    - A good approximation to sunlight
  - Direction is constant for all surfaces in the scene
  - Specify light source by giving the direction l of the light rays.



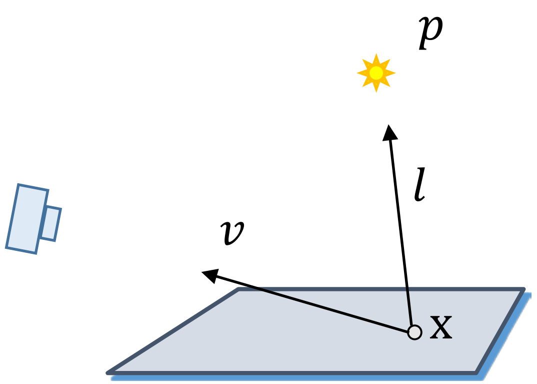
Point Light Sources

- Point light source
  - Emits light equally in all directions from a single point p
  - Direction to the light from a point on a surface differs for different points



  - Requires to calculate a normalized vector l to the light source for every lit point:

$$l = \frac{p - x}{\|p - x\|}$$



  - Distance to light also important → later
  - Essentially, the same is true for the camera: v is vector to the camera

Other Light Sources

- Spotlights
  - Point sources: intensity falls off directionally

- Area light sources
  - Define a 2D emissive surface
    - usually a disc or polygon
  - Good example: light panels
  - Capable of generating soft shadows
  - Not supported by OpenGL → later: ray tracing

Phong Lighting

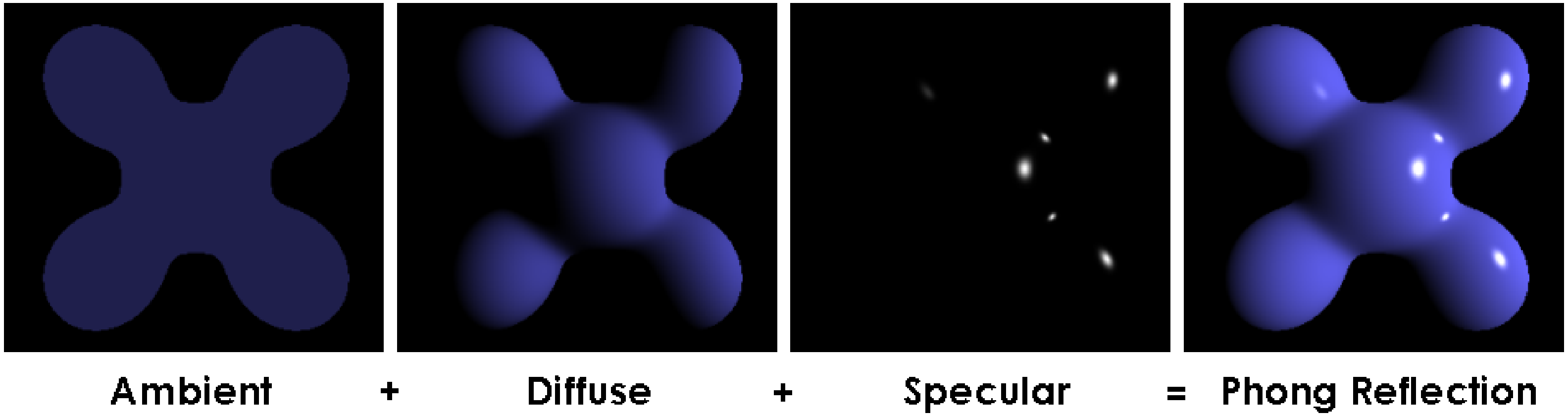
- Phong Lighting Model
  - Heuristic, not physical
  - Fast to evaluate and simple parameters
  - De-facto standard in CG

- Reflected light composed from 3 components
  - Ambient light/reflection L_{amb}
  - Diffuse reflection L_{diff}
  - Specular reflection L_{spec}

$$L_{total} = L_{amb} + L_{diff} + L_{spec}$$

Phong Lighting

- Ambient light
  - Simulates indirect light from surrounding
  - Modeled by constant
- Diffuse reflection
  - Reflection from rough surfaces
  - Uniform in all viewing directions
- Specular reflection
  - Reflection from glossy but not perfectly mirroring surfaces
  - Something "between" a diffuse and a perfect surface.



Source: http://en.wikipedia.org/wiki/Phong_shading

Ambient Light

- Objects not directly lit but still visible
  - Example: ceiling, undersides of desks
  - Result of indirect illumination: From emitters, bouncing off intermediate surfaces
  - Too expensive to calculate exactly (in real time)
  - Workaround (hack) called ambient light source
    - No spatial or directional characteristics
    - Illuminates all surfaces equally

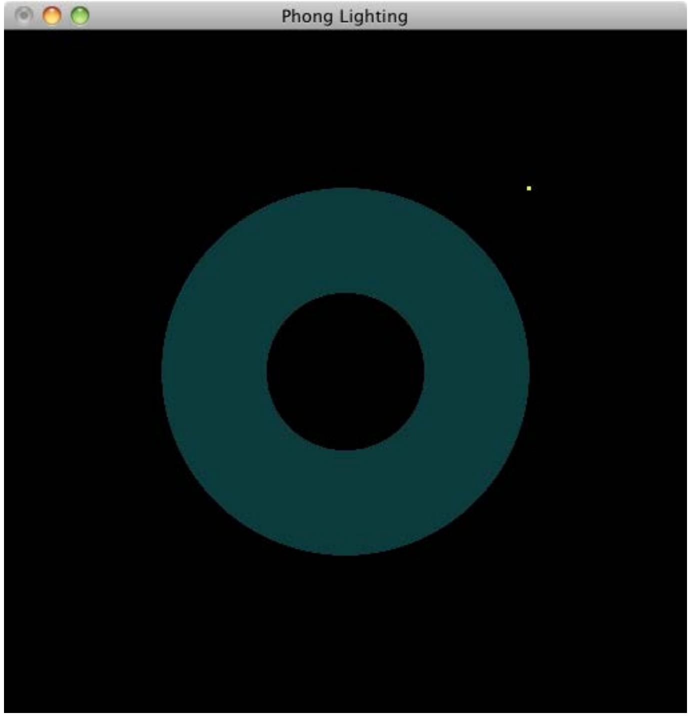
- Ambient light reflected from a surface
  - Surface properties
  - Intensity of the ambient light source (constant for all points on all surfaces)

$$L_{amb} = k_{amb} I_{amb}$$

- I_{amb}: intensity of the ambient light / light of surrounding
- k_{amb}: objects ambient reflection coefficient

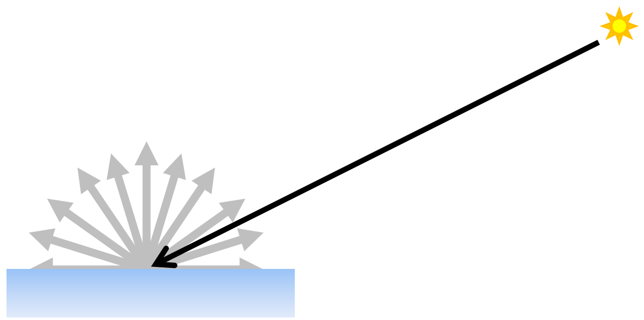
Ambient Light

- Scene lit only with an ambient light source



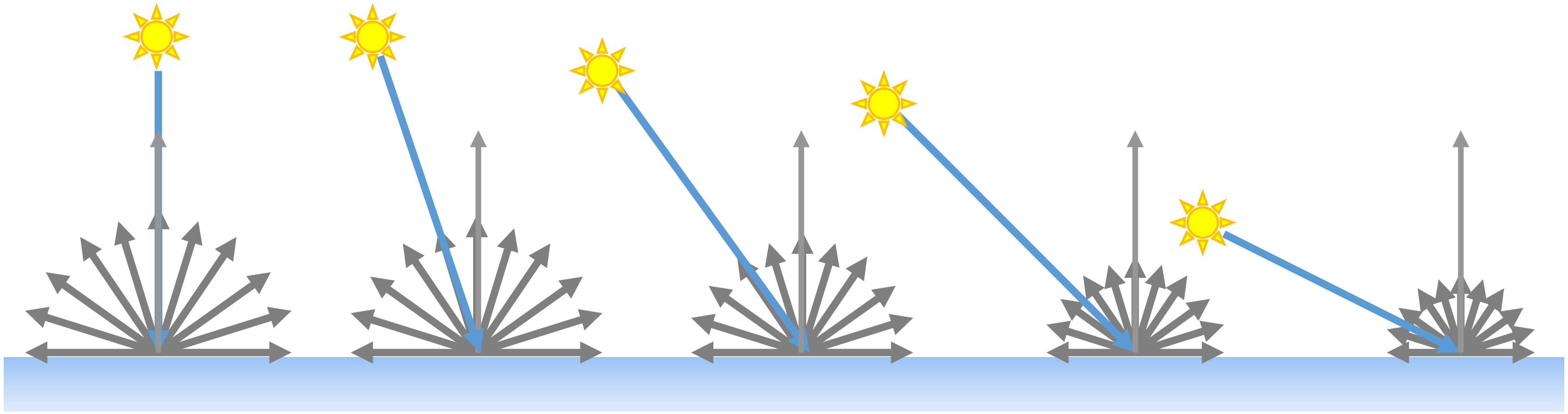
Diffuse Reflection

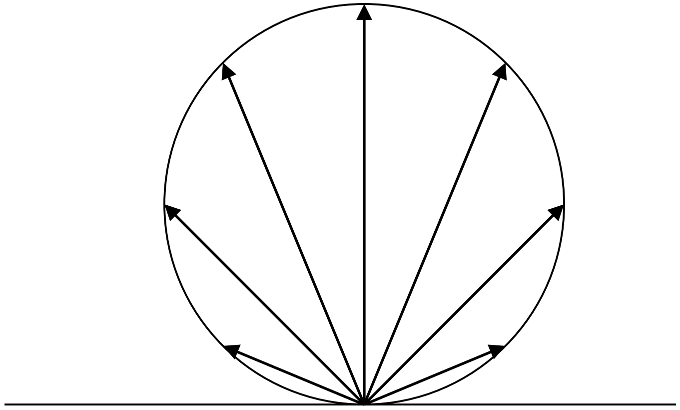
- Ideal diffuse reflection
  - Ideal diffuse reflector
    - Has very rough surface at microscopic level (e.g.: chalk, matte paint, cloth, unfinished wood, etc.)
    - Often called Lambertian surfaces, obeys Lambert's cosine law
  - Effect on incoming ray of light: Equally likely to be reflected in any direction over the hemisphere due to microscopic variations
  - Reflected light independent of viewing angle



Diffuse Reflection

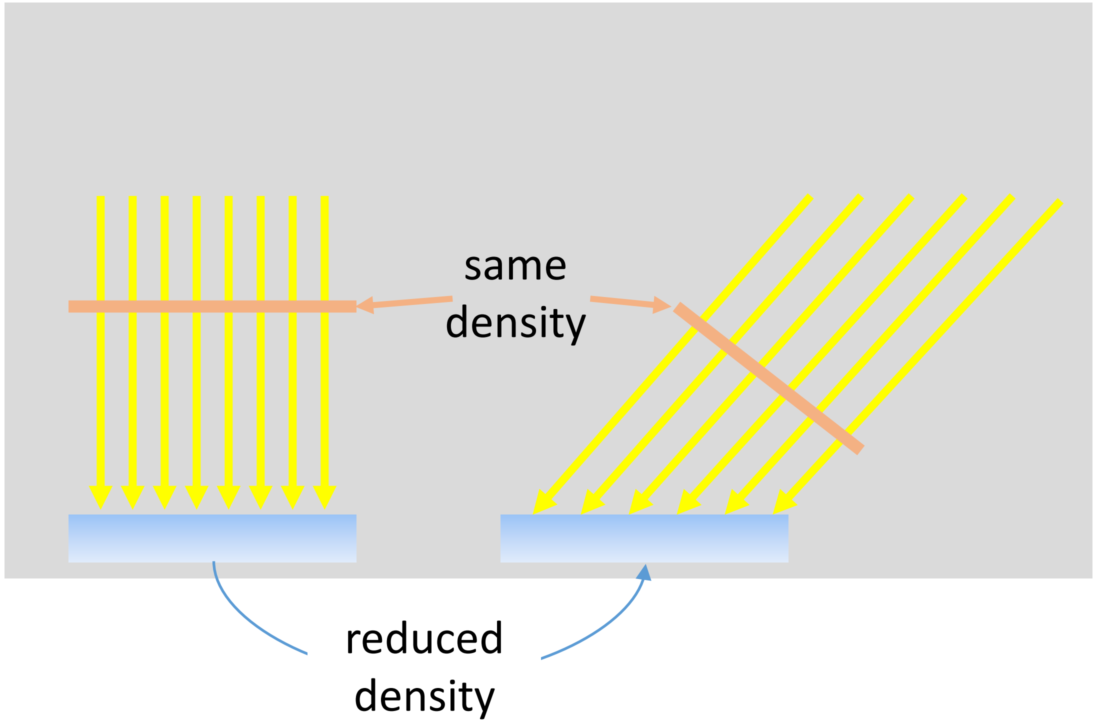
- Lambert’s Cosine Law





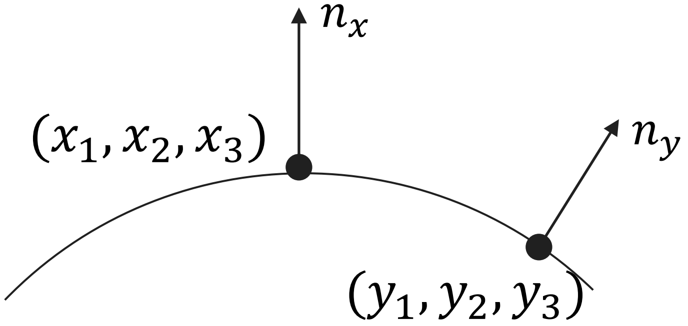
Diffuse Reflection

- Conservation of energy – energy flux

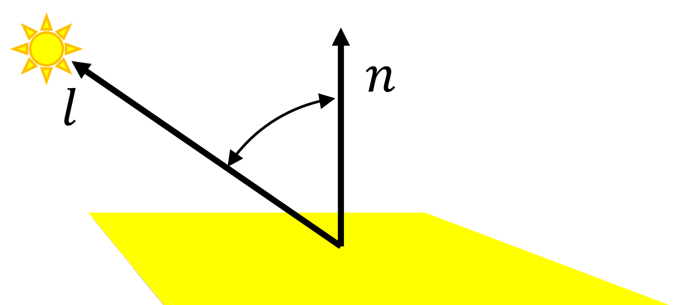


Diffuse Reflection

- Surface
  - position & a normal at every position



- Light Vector
  - l = vector to the light source
  - light position minus surface point position (normalized)



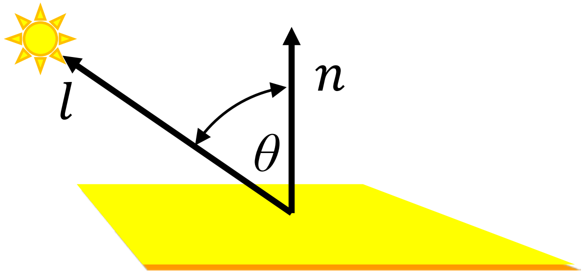
Diffuse Reflection

- Angle of incidence
  - Angle between surface normal and incoming light

$$L_{\text{diff}} = k_{\text{diff}} \, I_{\text{in}} \cos\theta$$

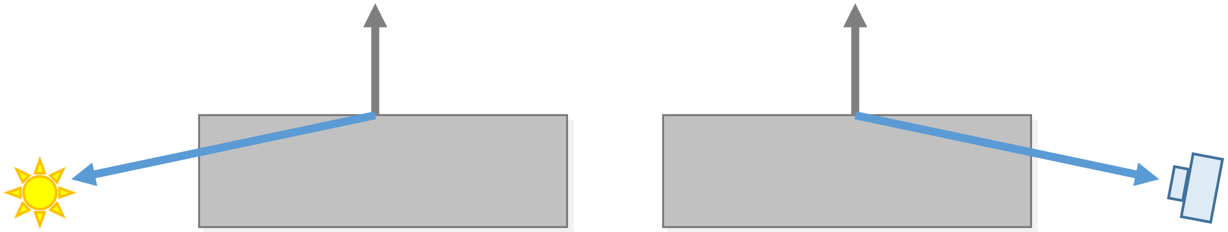
$$L_{\text{diff}} = k_{\text{diff}} \, I_{\text{in}} \, (n \circ l)$$

  - I_{in} : intensity of light source (can be RGB triple)
  - k_{diff} : surface color (can be RGB triple)
  - Use vector dot product to compute cosine.



Diffuse Reflection

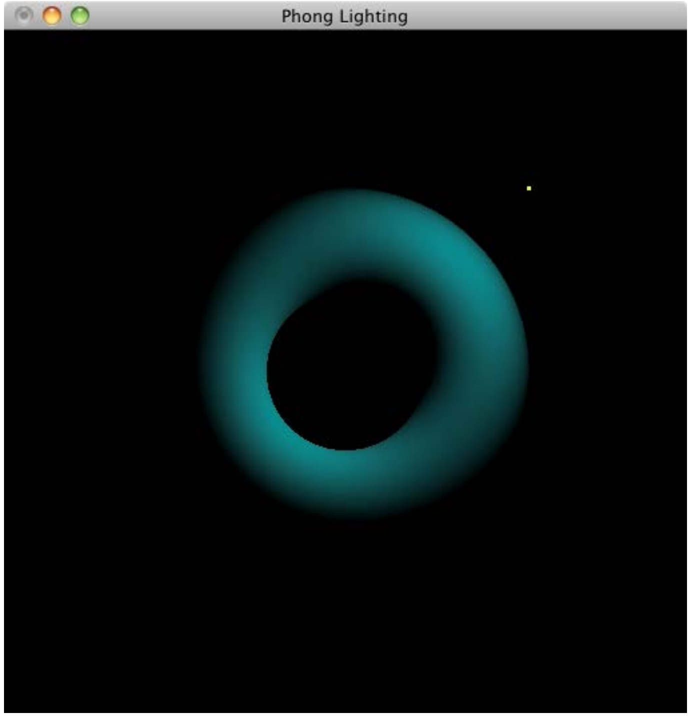
- Meaning of negative dot products
  - If $n \circ l < 0$, then the light is behind the surface and cannot illuminate it
  - If $n \circ v < 0$, then the viewer is looking at the underside of the surface and cannot see its front-face.



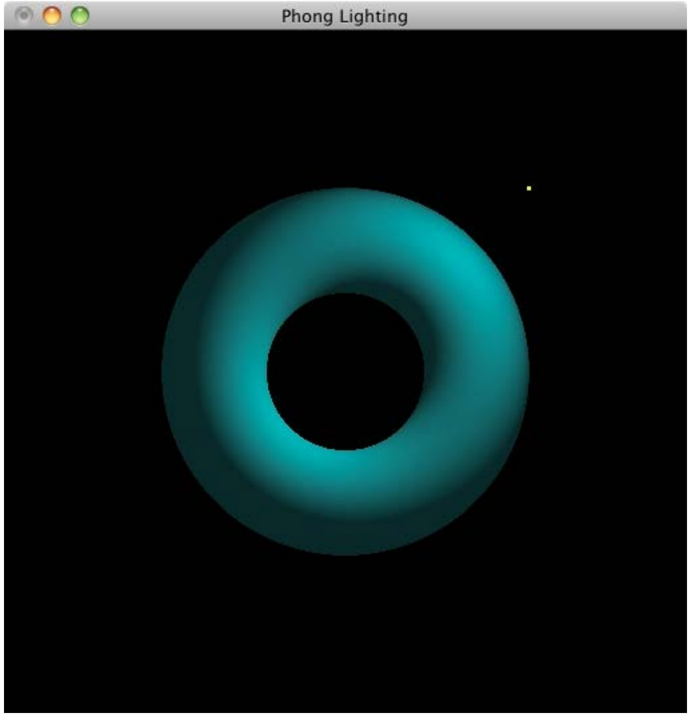
  - In both cases, the dot product is clamped to zero.

Diffuse Reflection

- Scene with diffuse material and lit with a point and an ambient light source.



Only diffuse



Diffuse material + ambient light

Specular Reflection

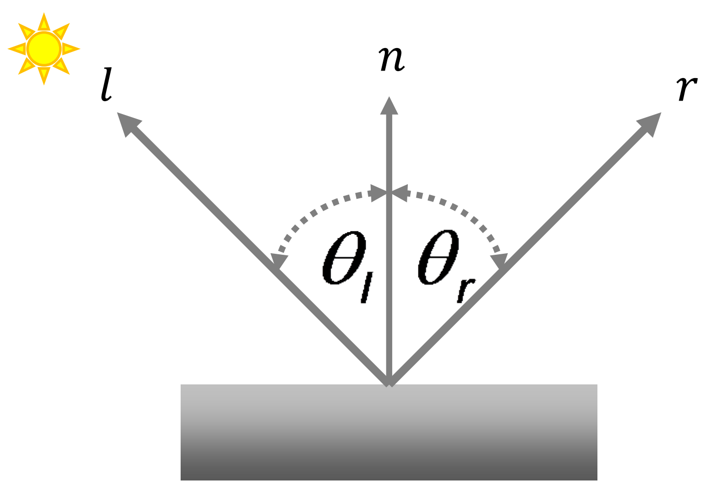
- Shiny surfaces
  - Exhibit specular reflection
  - Examples: polished metal, glossy car finish
- Light shining on specular surface causes a bright spot known as a **specular highlight**
- Occurrence of these highlights
  - Function of the viewer's position
  - specular reflectance is **view-dependent**

Specular Reflection

- At microscopic level
  - Specular reflecting surface is very smooth
- What happens to a ray of light
  - Bounced off the micro-geometry in mirror-like fashion
- General principle
  - The smoother the surface, the closer it gets to a perfect mirror

Ideal Specular Reflection

- Reflection following geometric optics
  - Incoming ray and reflected ray lie in a plane with the surface normal
  - Angle θ_r of reflected ray formed with surface normal equals angle θ_l of incoming ray formed with surface normal
- → Mirror



$$\theta_r = \theta_l$$

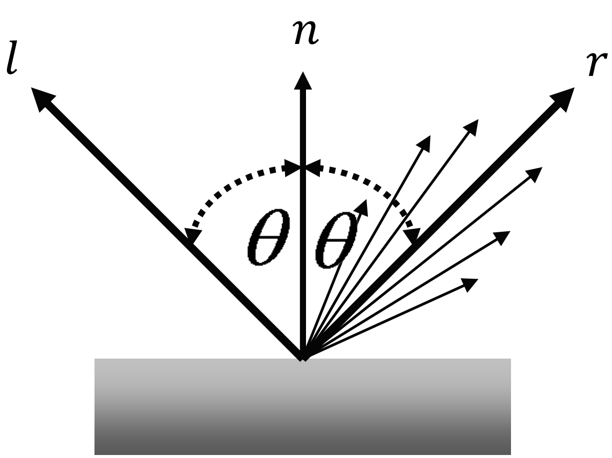
Non-Ideal Specular Reflectance

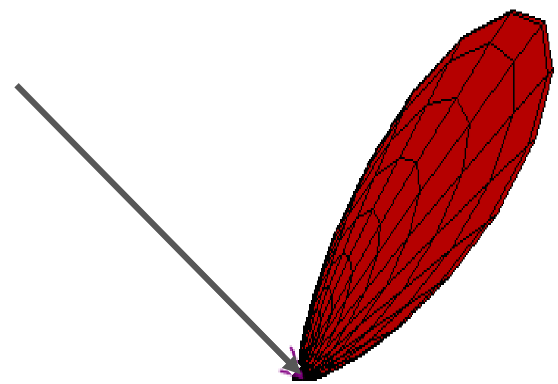
- Perfectly specular objects
  - Only with perfect mirror-like surfaces (and chrome)
  - Few surfaces exhibit perfect specular reflexion
  - with a point light: no highlight visible (infinitly small)

- Non-ideal ("softer") reflections
  - Surface is not perfectly smooth, but has some roughness
  - Glossy rather than mirror-like surface
  - Direct approach: Model the micro-geometry of the surface
  - Empirical approximation: use heuristic, simple and efficient.

Non-Ideal Specular Reflectance

- Empirical Approximation
  - General assumption: Most reflected light in direction according to ideal reflection.
  - Influence of microscopic surface variations
    - Some light reflected in direction slightly off ideal reflected ray.
    - Less reflected light is expected with increasing angle from ideal reflected ray.



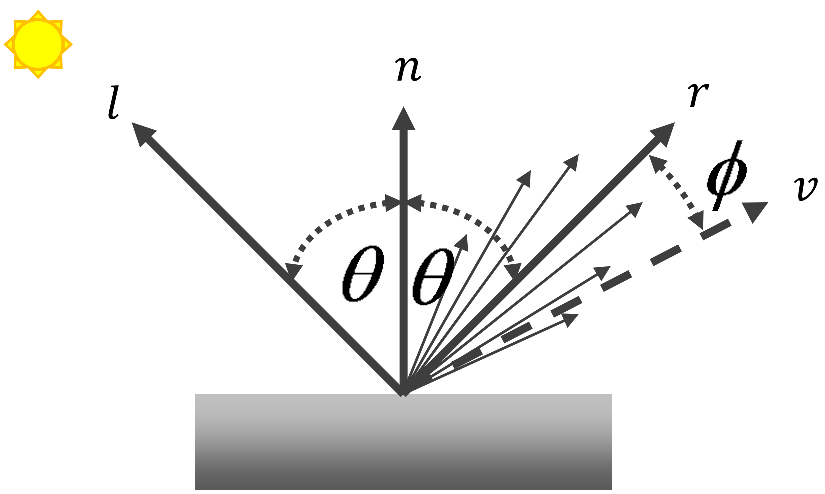


Specular Reflection

- Phong Lighting
  - Most common lighting model in CG

$$L_{spec} = k_{spec}\ I_{in}\ (\cos\phi)^{\mathrm{n_s}}$$

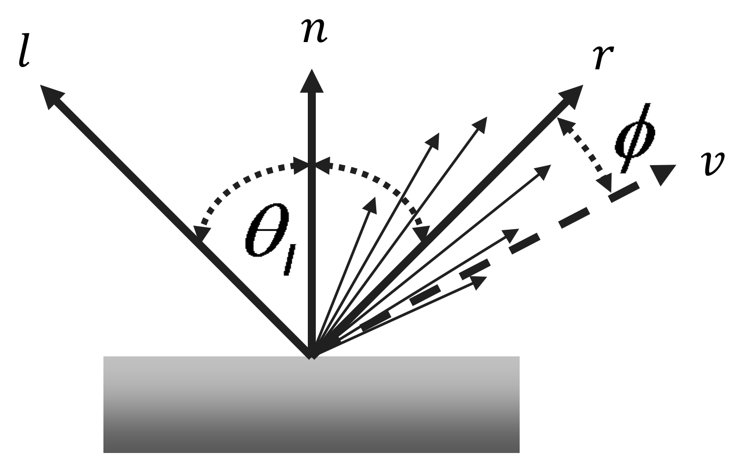
  - Phong reflectance term n_s (purely empirical constant – varies the rate of falloff)
  - Model has no physical basis, it works (sort of) in practice



Specular Reflection

- Calculating Phong Lighting
  - The **cos** term computed using vector arithmetic
  - v: unit vector towards the viewer
  - r: ideal reflectance direction

$$I_{spec} = k_{spec}\, I_{in}\, (v \circ r)^{n_s}$$

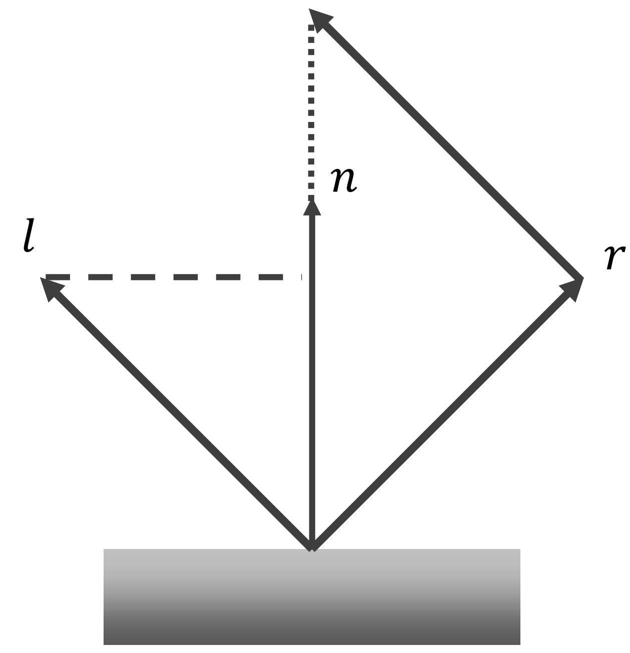


Phong Lighting

- Calculating the vector $\boldsymbol{r}$
  - Remember that n and l are normalized

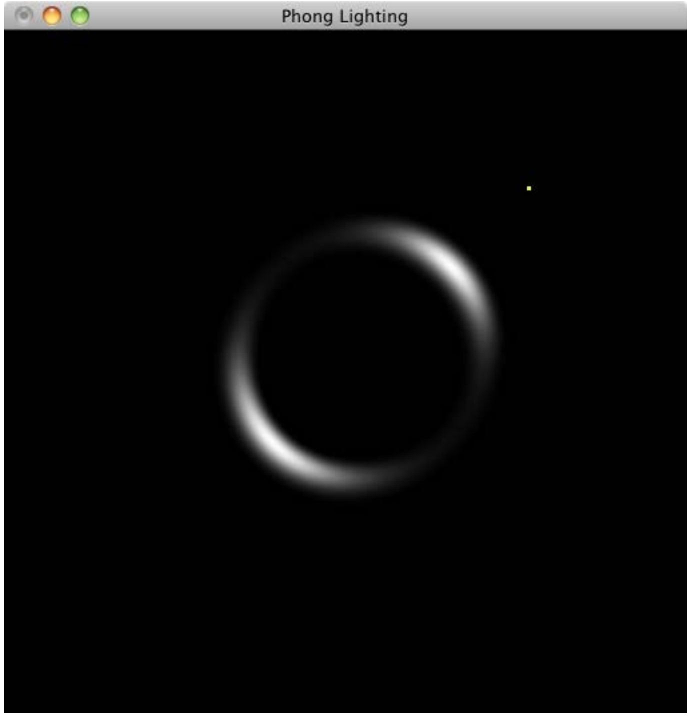
  - $r + l = 2(n \circ l)n$

$$r = 2(n \circ l)n - l$$

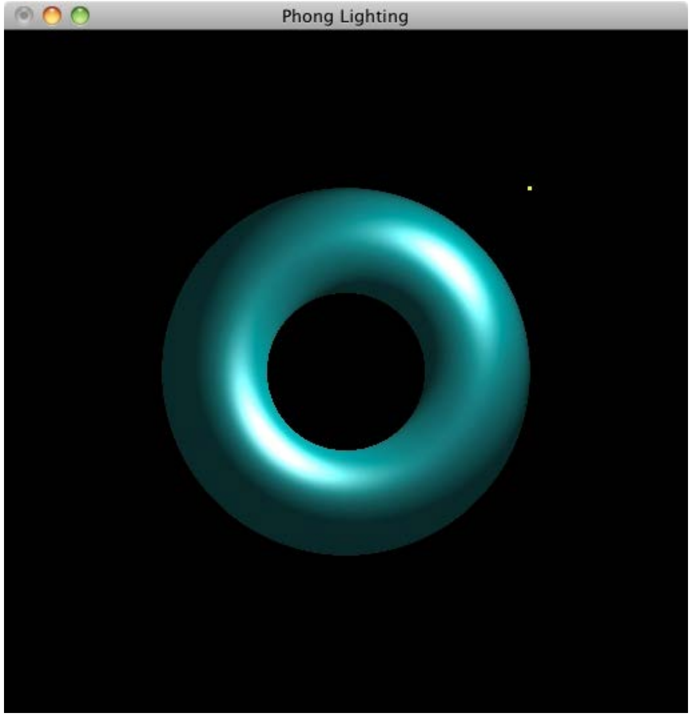


Specular Reflection

- Scene with specular surfaces.



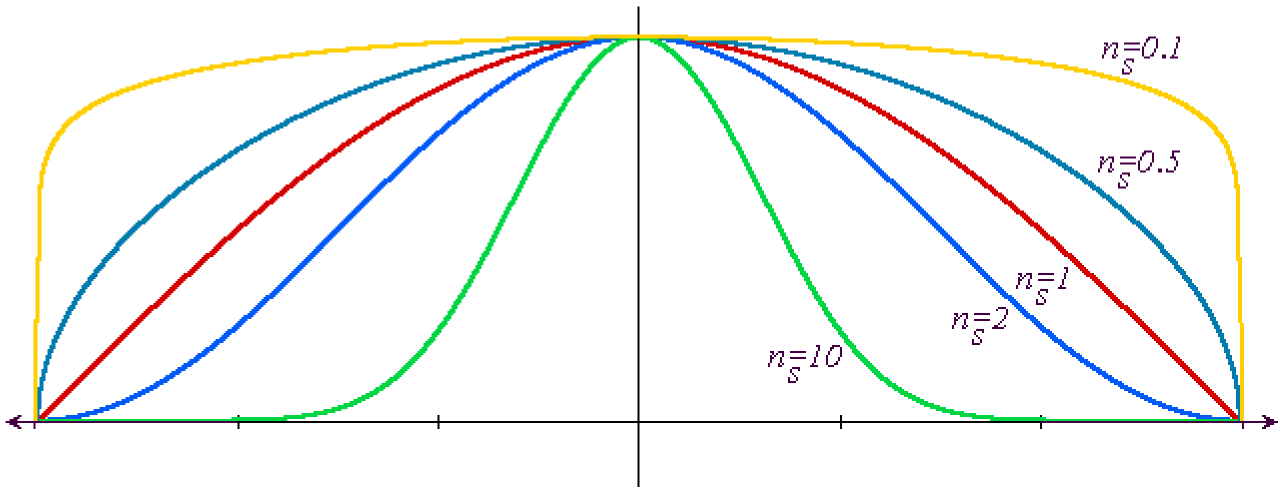
Only specular reflection



Ambient light + diffuse and specular reflection

Specular Reflection

- Phong reflectance term
  - Diagram: how Phong reflectance term drops off with divergence of viewing angle from ideal reflected ray



Specular Reflection

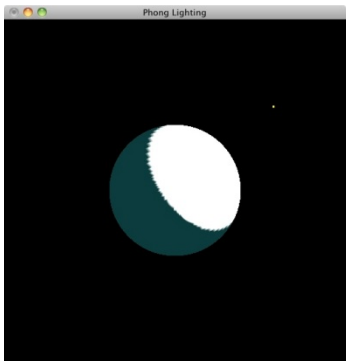
- What does this term control, visually?

n_s small $\rightarrow$ rather diffuse
n_s large $\rightarrow$ glossy
$\Rightarrow n_s$ controls roughness

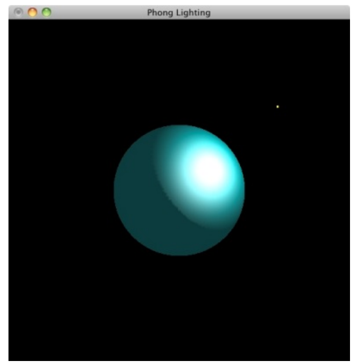
- Remark: the specular term is usually white.

Specular Reflection

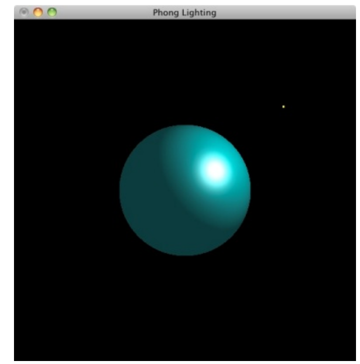
- Effect of shiny coefficient



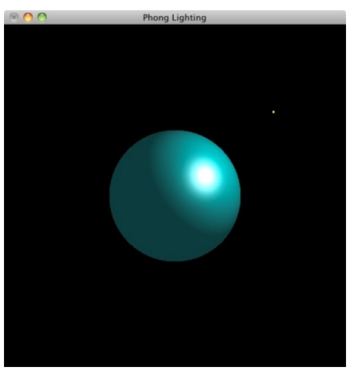
Shiny = 0



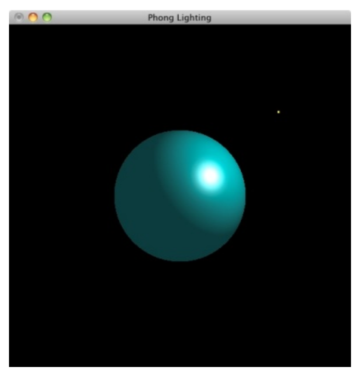
Shiny = 5



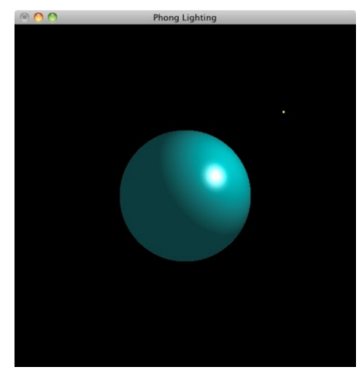
Shiny = 10



Shiny = 20



Shiny = 30



Shiny = 50

Phong Lighting

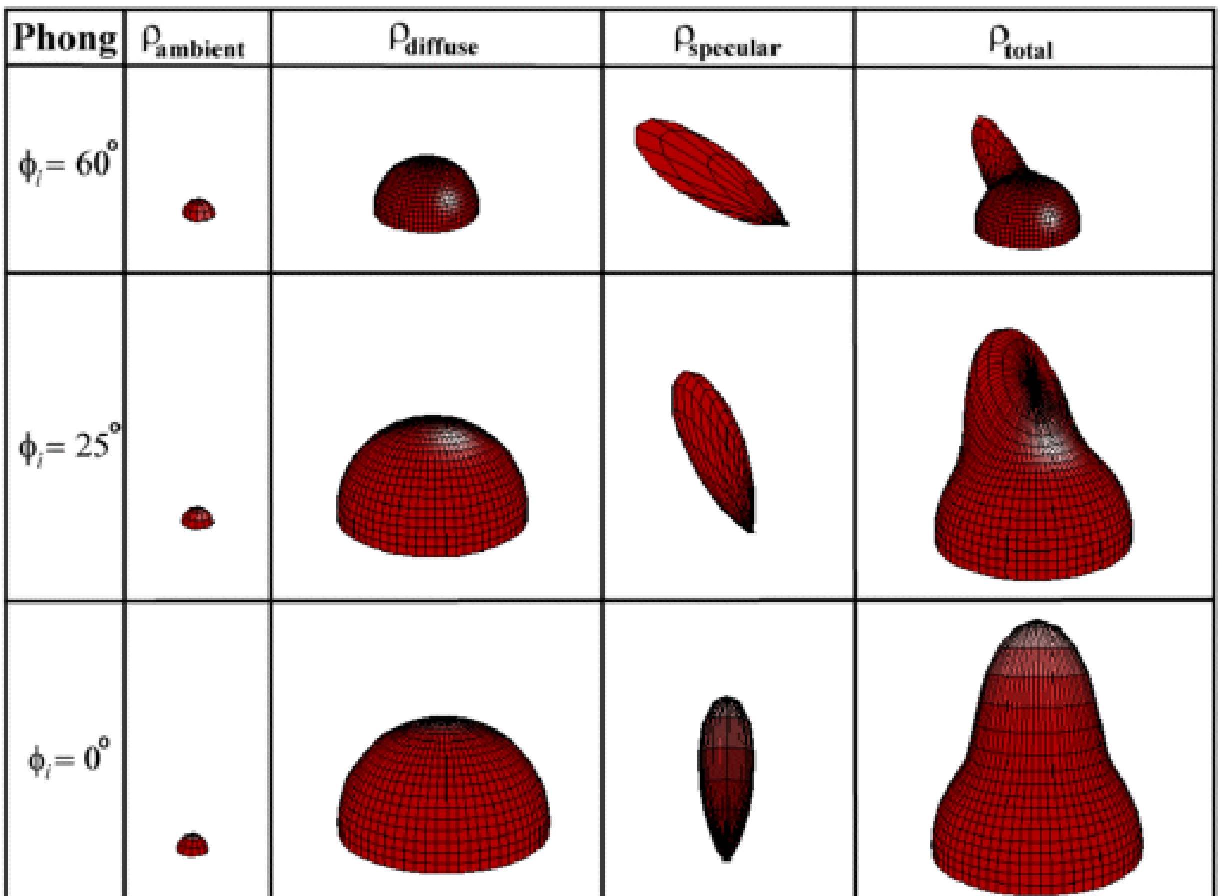
- Phong Lighting Model
  - Altogether: $L = L_{amb} + L_{diff} + L\_spec$
  - For multiple lights:

$$L = k_{amb} I_{amb} + \sum_{i=1}^{\#lights} I_i^{in} \left(k_{diff}(n \circ l_i) + k_{spec}(v \circ r_i)^{n_s}\right)$$

  - $k_{amb}, k_{diff}, k_{spec}$ are color triples

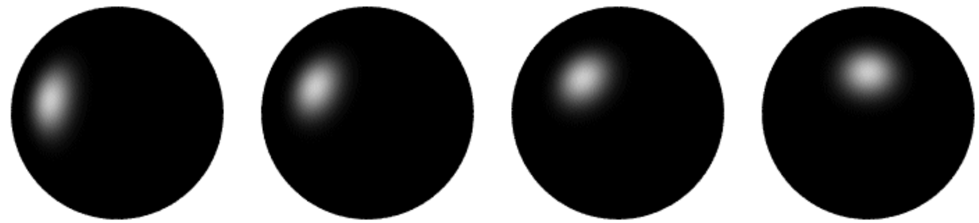
Phong Lighting

- Intensity Plots

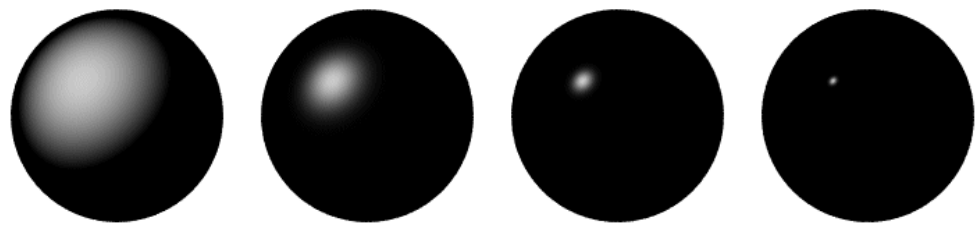


Phong Lighting

- Spheres illustrating the Phong model
  - Varying l



  - Varying n_s



  - specular reflections are usually white
  - Colored plastic: k_{spec} = white and k_{diff}= plastic color

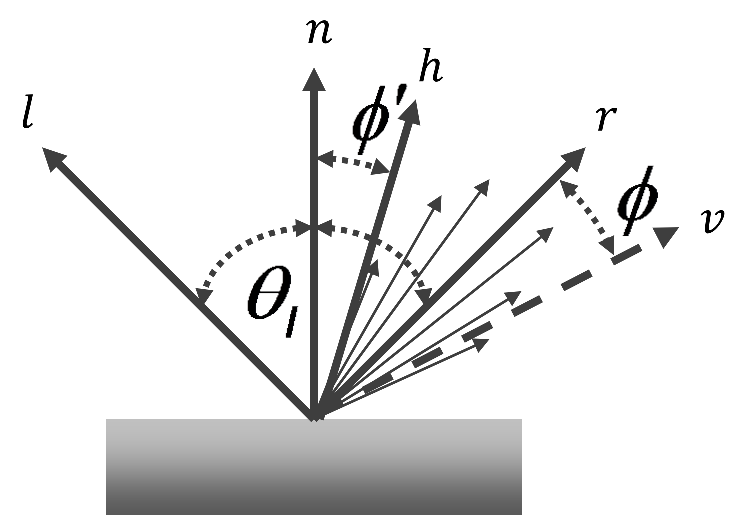
Phong Lighting

- For $n_s = 1$, the specular term is *not* the same as the diffuse
  - Dot product with reflection direction, not with normal

- When n_s gets larger, highlights get smaller, but not brighter
  - Less light is reflected
  - $n_s \rightarrow \infty$ does not generate mirror !
    - This would require reflection computations (see later)
  - Amount of reflected light varies with n_s→ not energy-conserving

Phong Lighting

- Blinn-Phong Lighting Model: Halfway optimization
  - It's a bit faster to use angle ϕ' instead of ϕ between normal n and halfway vector h
  - If n, l and v are coplanar, then $\phi' = \phi/2$
  - $h = \frac{l+v}{\|l+v\|}$, $\cos\phi' = h \circ n$
  - $I_{spec} = k_{spec}\, I_{in}(n \circ h)^{n_s}$

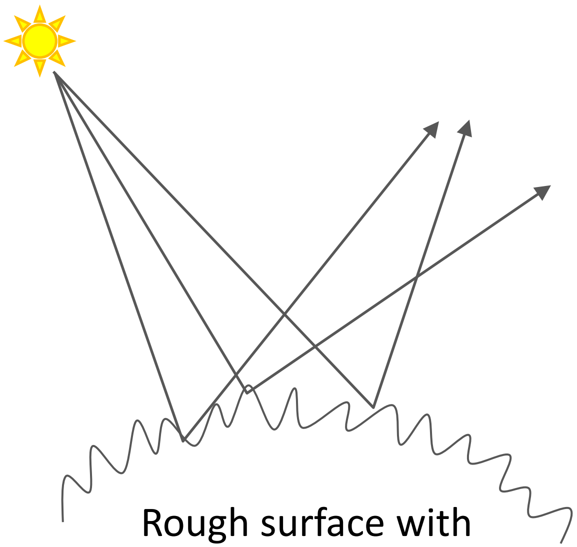


Phong Lighting

- Attenuation: Consider distance d from light source to object
  - Parallel (directional) light with intensity I
    - $I_{in} = I = const$
  - Point light at p with intensity I
    - According to physics: $I_{in} = \frac{I}{\|p-x\|^2}$
    - Look implausible for very close and very distant light sources
    - Instead use second-order polynomial
      $I_{in} = \frac{I}{c_0+c_1 d+c_2 d^2}$ with $d = \|p - x\|^2$
    - c_0, c_1, c_2 are parameters to be chosen by user

Torrance-Sparrow Light Model

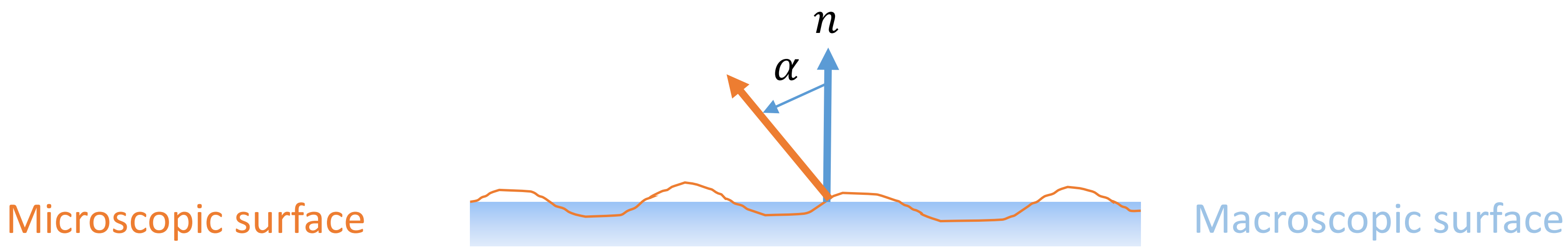
- Physically based light model –
  - Torrance-Sparrow, 1967
  - Cook and Torrance, 1982: model for computer graphics
  - Also known as Blinn-Cook-Torrance,
  - Models the specular term as a perfect reflection off a bumpy surface



Rough surface with mirror reflection

Torrance-Sparrow Light Model

- Major assumptions
  - Surface is assumed to be composed of a collection of mirror like micro facets that are oriented in random directions on the surface.
  - The macroscopic normal n to the surface is the mean of the normals of all micro facets.
  - Each micro facet has an inclination α with respect to n which is assumed to be rotation symmetric around n.



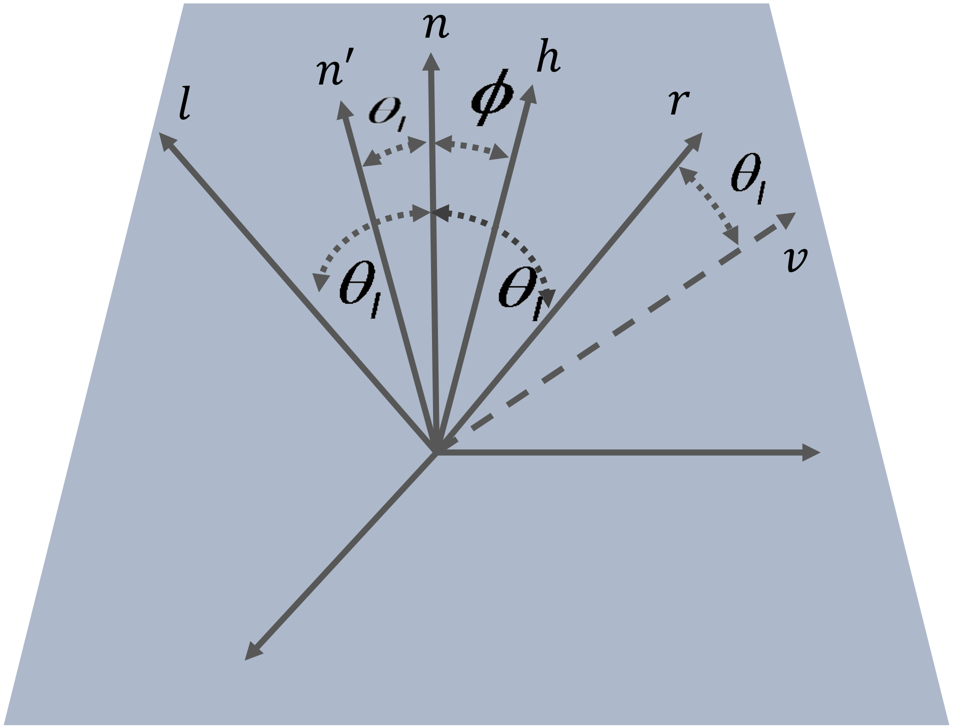
Torrance-Sparrow Light Model

- Macroscopic reflection described using three terms:
  - Distribution of normals D
    - Probability distribution of inclination angles
  - Fresnel term F
    - Describes reflection off specular surface
    - More light is reflected for grazin angles
  - Geometry term G
    - For very rough surfaces (high variation of normals), self-occlusion appears, in particular for grazing angles

$$L_{spec} = I_{in} \frac{D \cdot F \cdot G}{\pi (n \circ v)(n \circ l)}$$

Torrance-Sparrow Light Model

- n': normal to the microscopic surface
- α: inclination
- l: direction to light
- r: direction of reflection
- h: halfway vector



Torrance-Sparrow Light Model

- Torrance-Sparrow specular component

$$L_{spec} = I_{in} \frac{D(h \circ n) \cdot F(l \circ n) \cdot G(l \circ n, v \circ n)}{\pi(n \circ v)(n \circ l)}$$

  - D: reflection coefficient due to micro faces
  - F: Fresnel coefficient
  - G: geometrical attenuation factor
  - Proportional to surface area that
    - Viewer's foreshortened area: $n \circ v$
    - Light's foreshortened area: $n \circ l$

Torrance-Sparrow Light Model

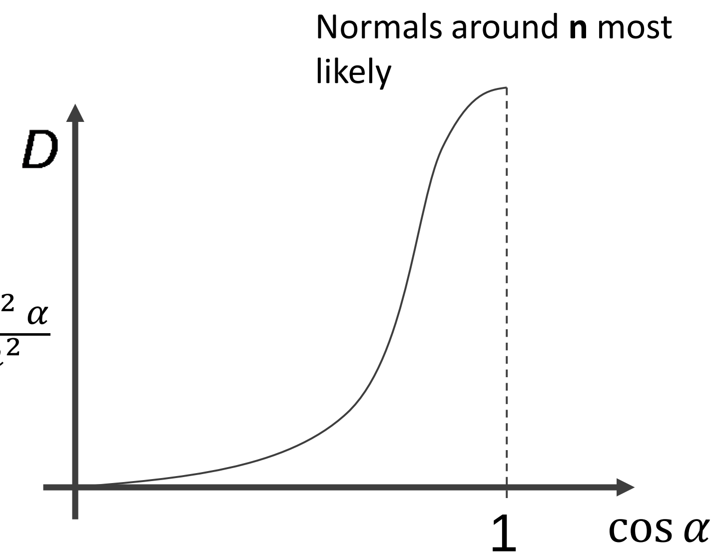
- Term $\mathrm{D}(\cos\alpha) = D(n \circ n')$
  - Distribution function of the micro facets orientation (probability)
  - Assumption: the distribution of the micro facets is isotropic around the normal n of the surface, i.e. it depends only of the inclination angle α.

- Possible terms D(α)
  - Gaussian distribution

$$D(\alpha) = ce^{-\frac{\alpha^2}{m^2}}$$

  - Beckmann distribution

$$\mathrm{D}(\alpha) = \frac{1}{m^2 \cos^4 \alpha} e^{-\frac{\tan^2 \alpha}{m^2}}$$



Torrance-Sparrow Light Model

- Fresnel term $F(l, v)$
  - Relates incident to reflected light for planar, dielectric surface
  - For such surface, a fraction of the incident light is reflected, the remainder is refracted
  - Fraction varies with angle of incidence θ_i

$$R_s = \left|\frac{n_1 \cos\theta_i - n_2 \cos\theta_t}{n_1 \cos\theta_i + n_2 \cos\theta_t}\right|^2 = \left|\frac{n_1 \cos\theta_i - n_2\sqrt{1-\left(\frac{n_1}{n_2}\sin\theta_i\right)^2}}{n_1 \cos\theta_i + n_2\sqrt{1-\left(\frac{n_1}{n_2}\sin\theta_i\right)^2}}\right|^2$$

$$R_p = \left|\frac{n_1 \cos\theta_t - n_2 \cos\theta_i}{n_1 \cos\theta_t + n_2 \cos\theta_i}\right|^2 = \left|\frac{n_1\sqrt{1-\left(\frac{n_1}{n_2}\sin\theta_i\right)^2} - n_2 \cos\theta_i}{n_1\sqrt{1-\left(\frac{n_1}{n_2}\sin\theta_i\right)^2} + n_2 \cos\theta_i}\right|^2$$

Different polarization directions

From Wikipedia

$$R = \frac{R_s + R_p}{2}$$

Fresnel-Term for unpolarized light

Torrance-Sparrow Light Model

- Schlick-Approximation
  - $F(\alpha) = F_0 + (1 - f_0)(1 - \cos\alpha)^5$

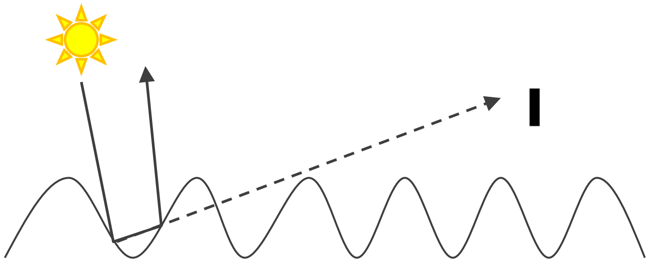
- For common glass, 4% of perpendicular incident light is reflected, at 80° about 50%

Torrance-Sparrow Light Model

- Term $G(n, l, v)$
  - Geometrical attenuation factor
  - Represents masking and self-shadowing of micro facets.
  - Reflection in direction l is unlikely because the photons reflected in direction l will be scattered out by the surface

$$G(n, l, v) = \min(1, G_{msk}, G_{shdw})$$

$$G_{msk} = \frac{2(n \circ h)(n \circ v)}{v \circ h}, G_{shdw} = \frac{2(n \circ h)(n \circ l)}{v \circ h}, h = \frac{v + l}{\|v + l\|}$$

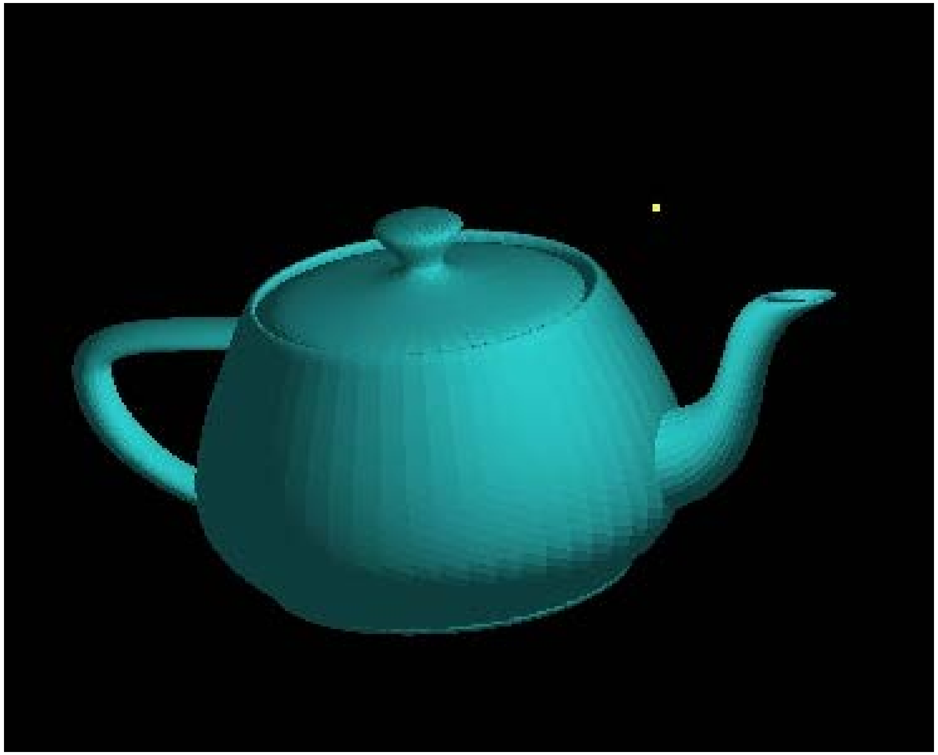


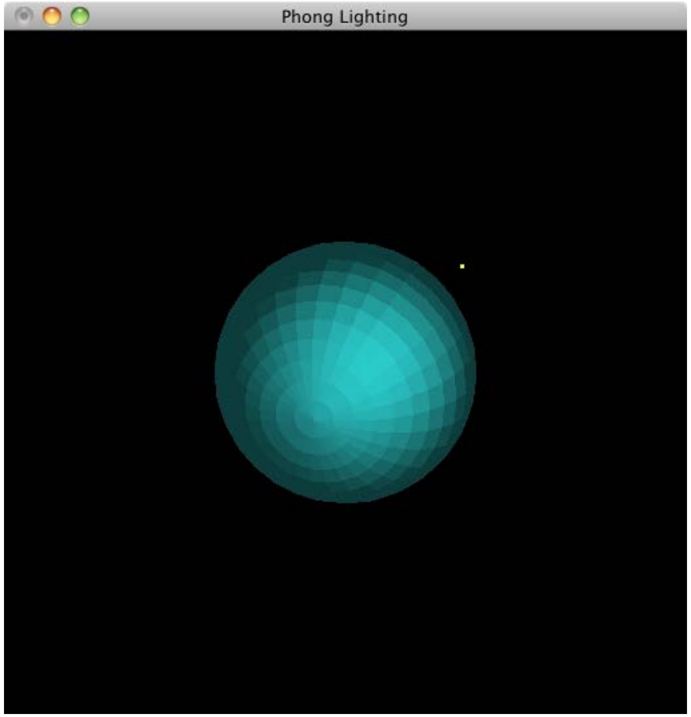
Flat Shading

- Flat Shading
  - Each face has a constant surface normal
  - Lighting mainly depends on normal
    -> a single color per face is okay
  - Calculate a single lighting value for each polygon
  - But only approximate
    - For point sources: direction to light varies across the facet
    - For specular reflectance: direction to eye varies across the facet
  - And the single faces become visible, so curved surface get facetted

Flat Shading

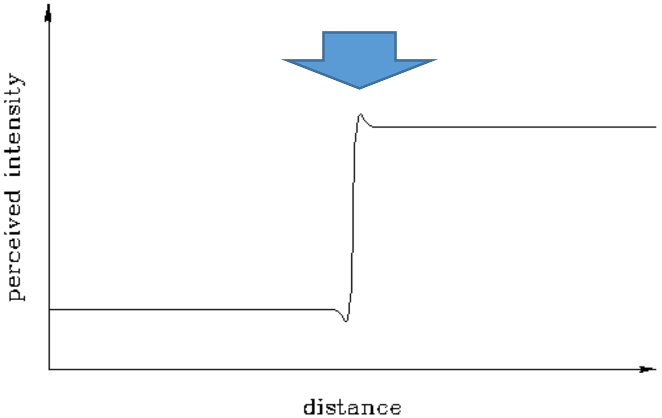
- Facets get *very* visible due to **Mach band effect**





Mach Band Effect

- Mach band effect
  - Human mind subconsciously increases contrast between two surfaces with different luminance.



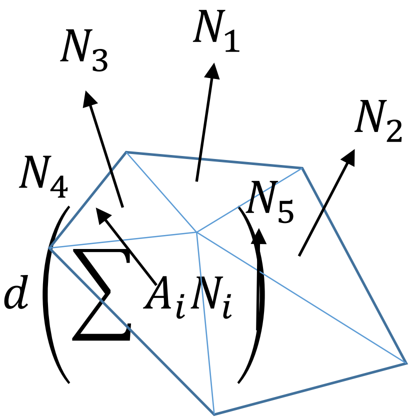
Gouraud Shading

- Gouraud Shading
  - Avoid discontinuities of flat shading
  - Perform Phong lighting once per vertex
    → use this color for all aligning triangles
    → requires a single normal per vertex
    → normally less vertices than faces !
  - Linear interpolation of resulting colors over faces during rasterization
    → see lecture **rasterization**

Gouraud Shading

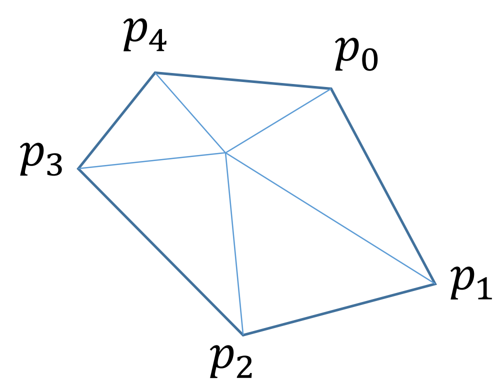
- Vertex normal averaged by area

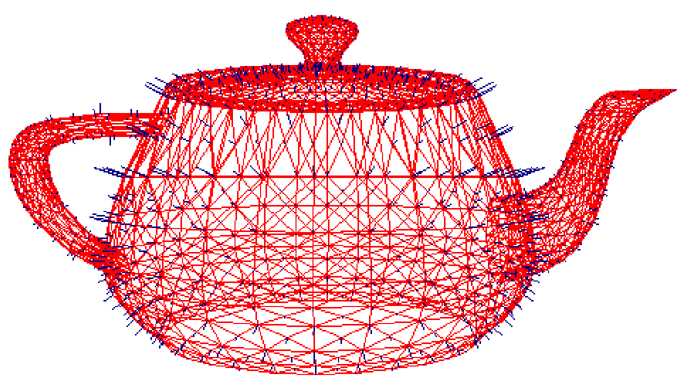
$$N = normalized\left(\sum A_i N_i\right)$$



- Same result, but directly from fan around vertex:

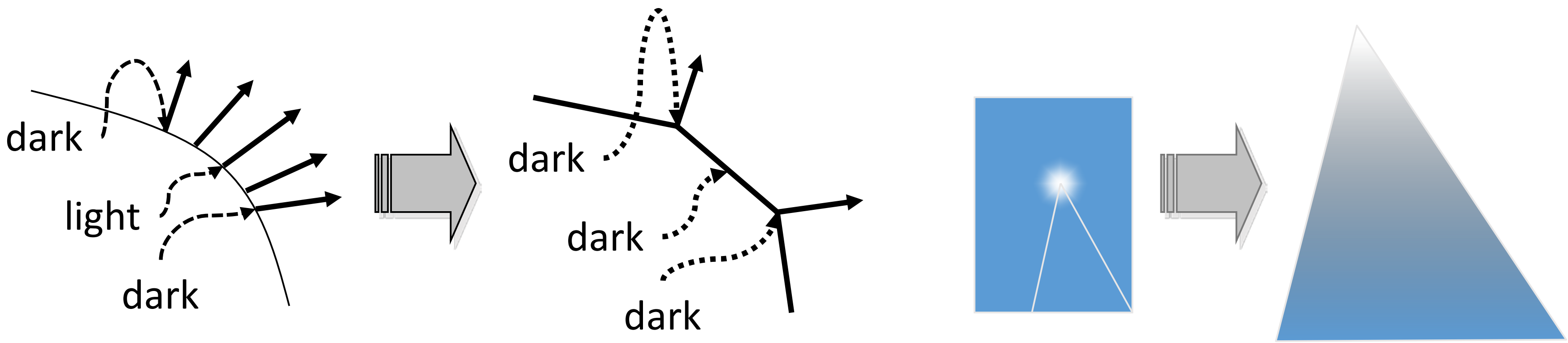
$$N = normalized\left(\sum p_i \times p_{i+1}\right)$$





Gouraud Shading

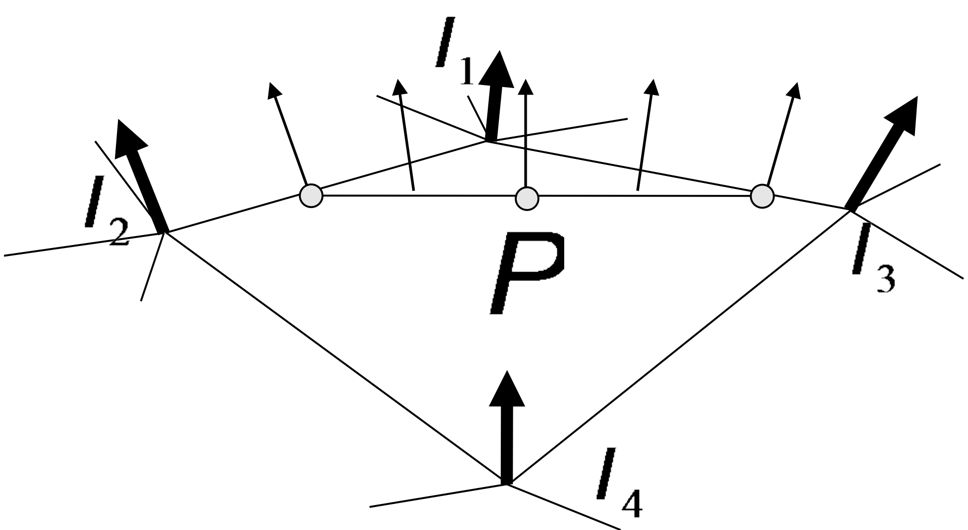
- Problem with highlights
  - Get lost when in between vertices
  - And get enlarged when at vertices



- → Phong shading as better alternative

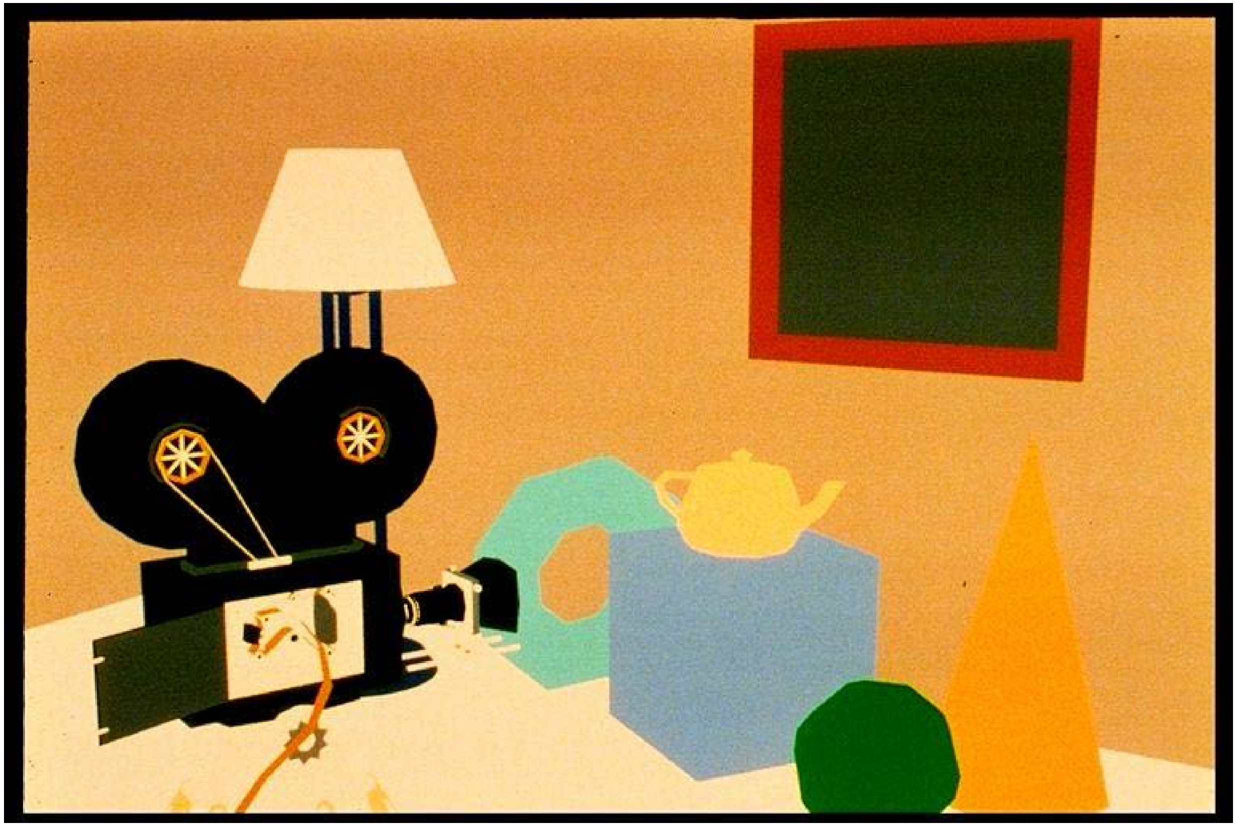
Phong Shading

- Normal vectors at vertices
- Interpolate normal vectors during rasterization
  → must be renormalized (interpolated normals no longer unit vectors)
- do lighting computation in fragment shader **per pixel**
  → per pixel lighting
- considerably more expensive than Gouraud Shading (= vertex lighting)
- but much better quality, in particular for glossy objects



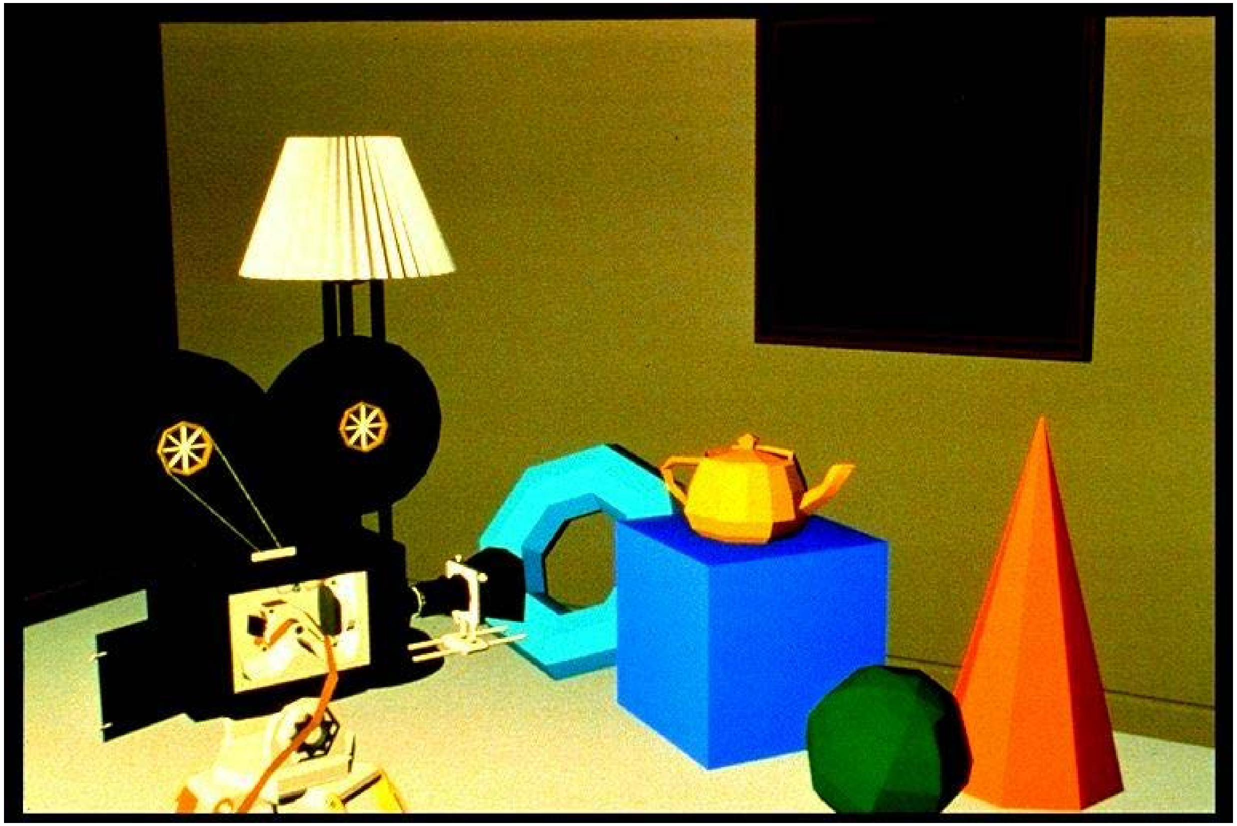
Overview

- Constant Shading



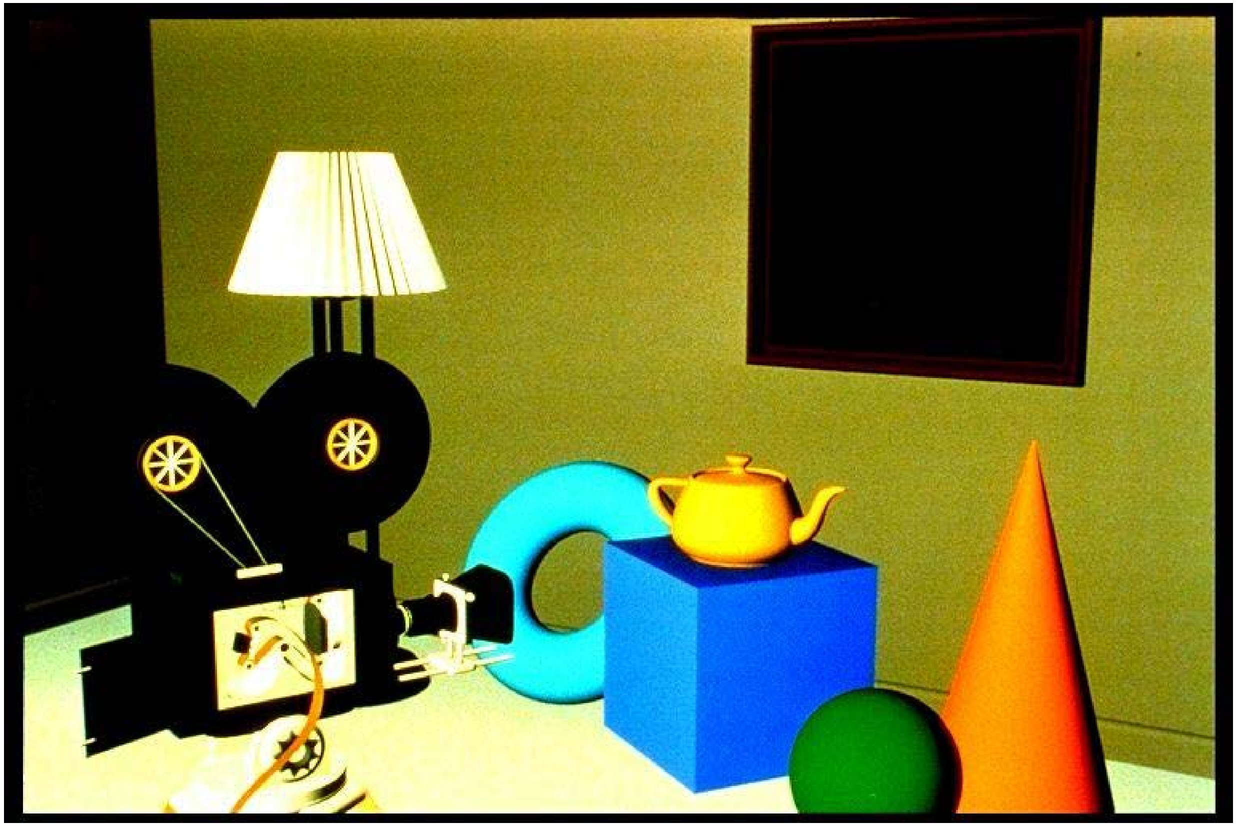
Overview

- Flat shading



Overview

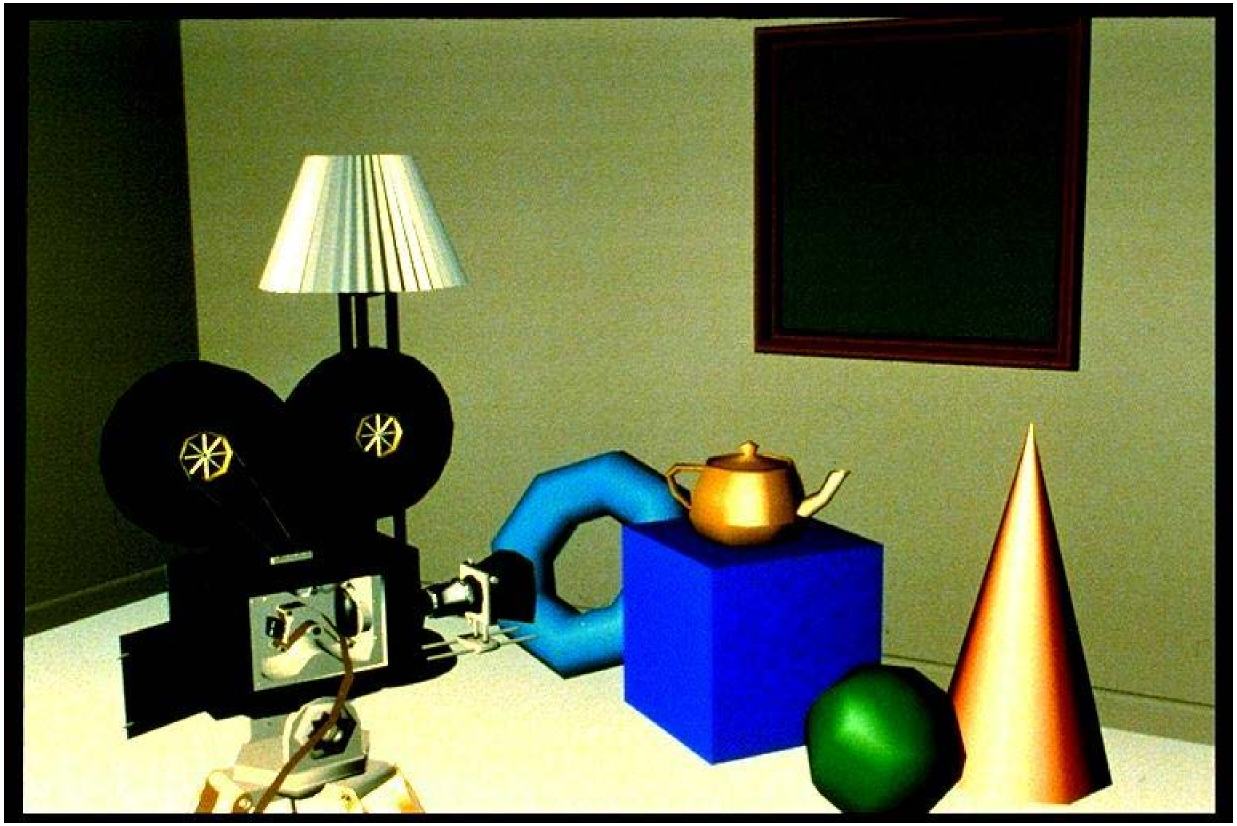
- Gouraud shading and no specular highlights



A Pixar Shutterbug example

Overview

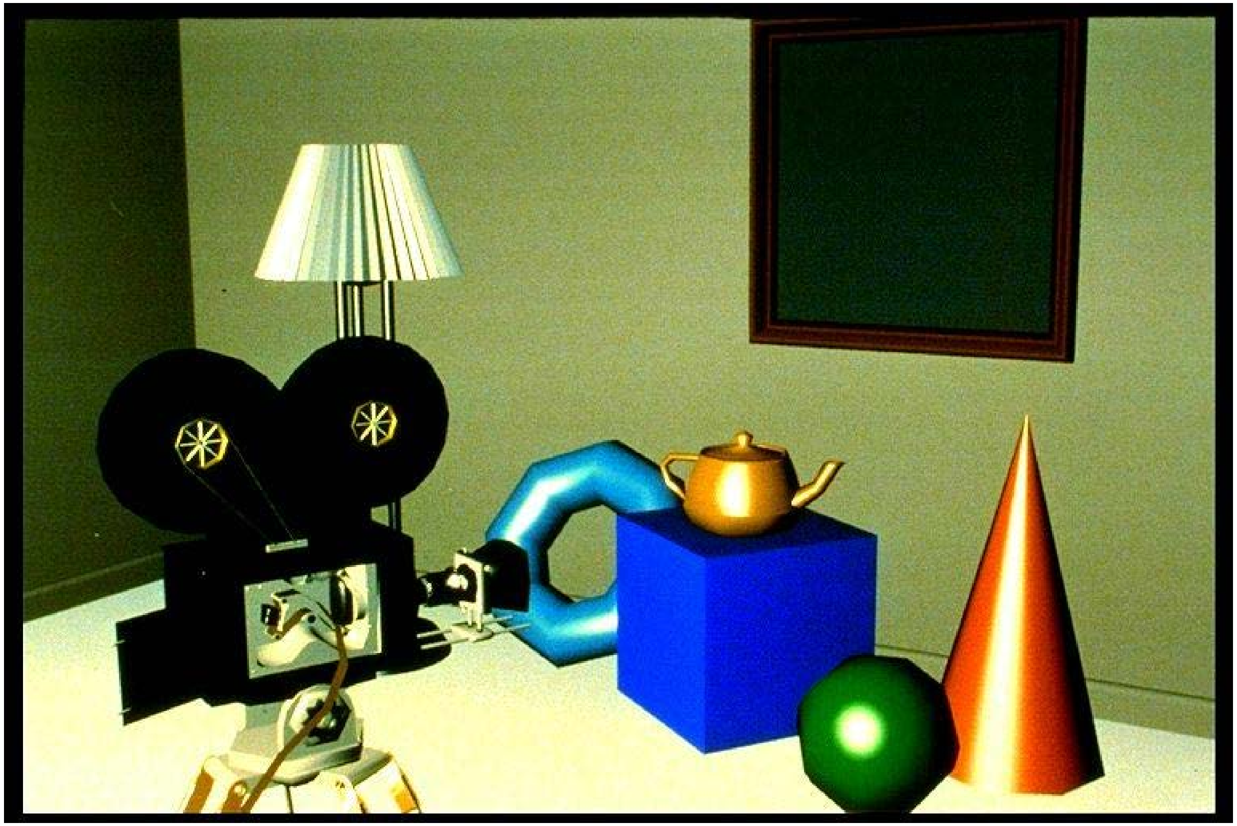
- Gouraud shading and specular highlights



A Pixar Shutterbug example

Overview

- Phong Shading Model



A Pixar Shutterbug example

Let’s Play

- Phong Lighting – Gouraud Shading – Phong Shading