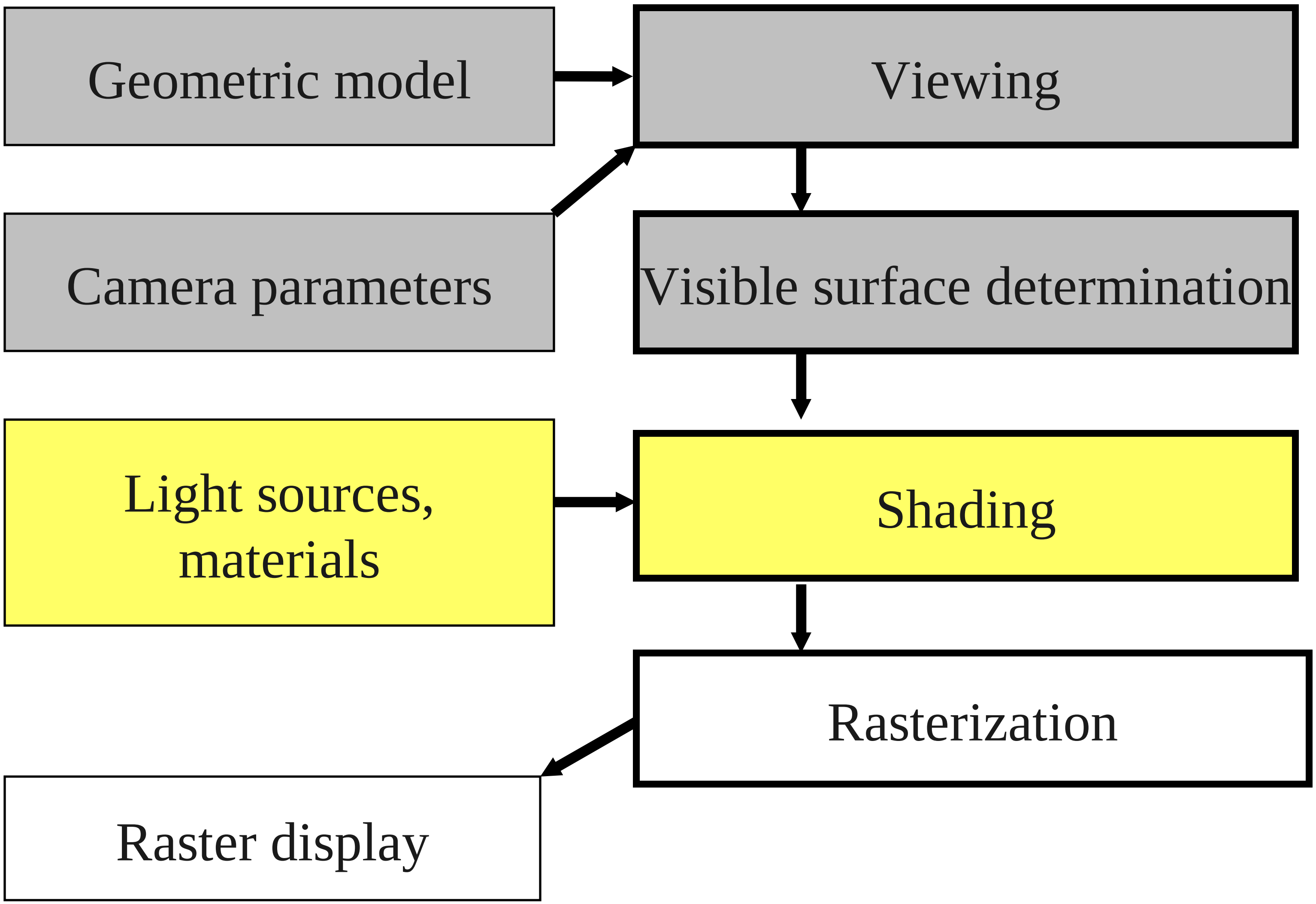
7M836 *Animation & Rendering*

Illumination models, light sources, reflection shading

Arjan Kok, Kees Huizing, Huub van de Wetering
h.v.d.wetering@tue.nl

Graphics pipeline



Illumination

- Illumination model
  - determines Illumination (= color) of a point on a surface by simulation of light in a scene
  - is an approximation of real light transport

- Shading method
  - determines for which points illumination is *computed*
  - and *how* it is approximated for other points.

Illumination models

- Model(s) needed for
  - Emission of light
  - Scattering light at surfaces
  - Reception on camera

- Desired model requirements
  - Accurate
  - Efficient to compute

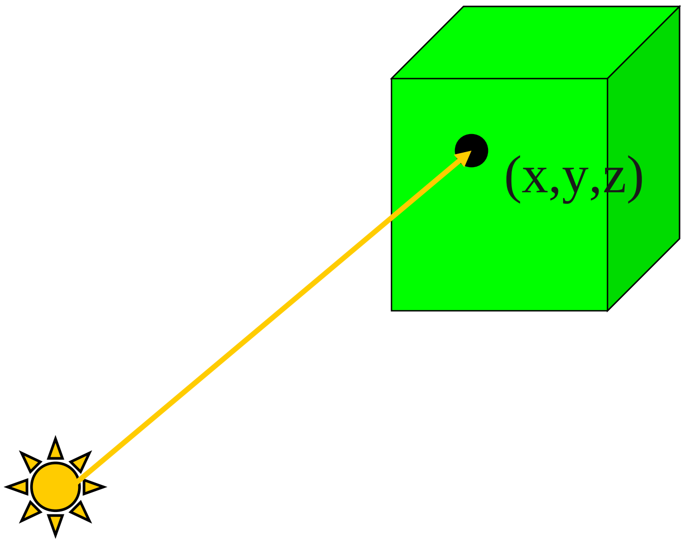
Illumination models

- Local illumination
  - Emission of light sources
  - Direct illumination
  - Scattering at surfaces

- Global illumination
  - Shadows
  - Refraction
  - Interreflection

Light source

- I_L (x, y, z, θ, φ, λ)
  - Intensity of energy
  - of light source L
  - arriving at point (x, y, z)
  - from direction (θ, φ)
  - with wavelength λ

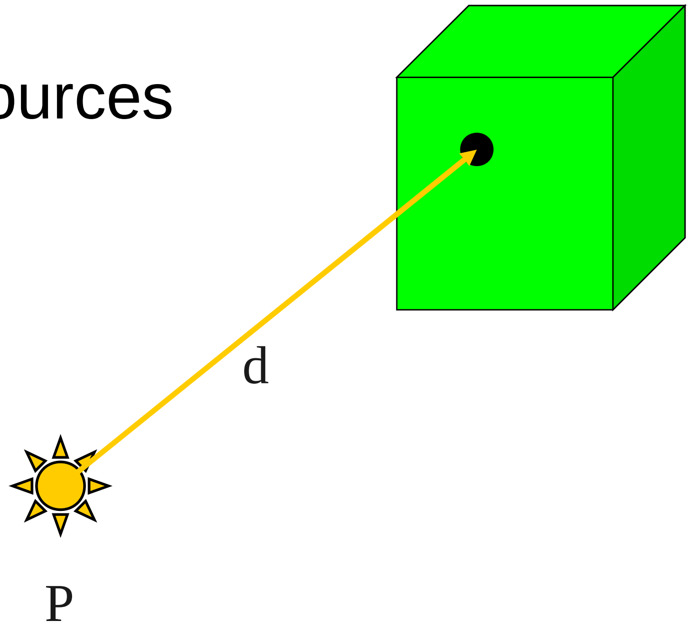


- Complex
  - Simpler model needed

Point light source

- Emits light equally in all directions
- Parameters
  - Intensity I_0
  - Position P (p_x, p_y, p_z)
- Approximation for small light sources

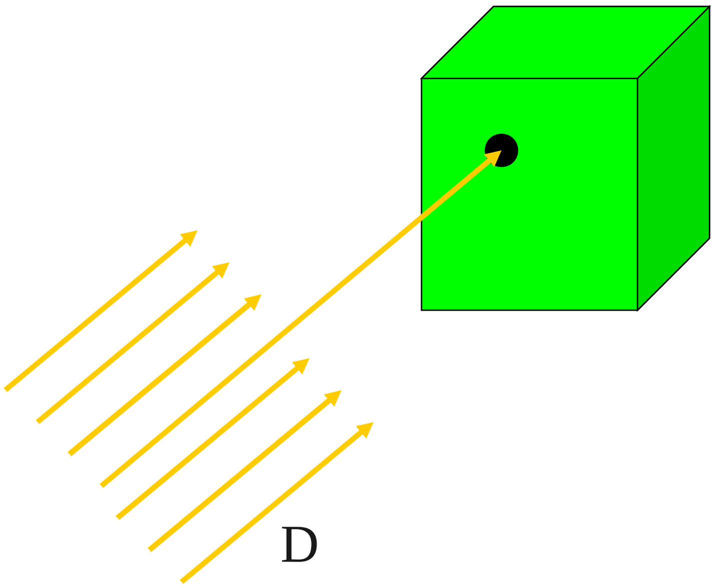
- $I_L = I_0$



Point light source

Direction light source

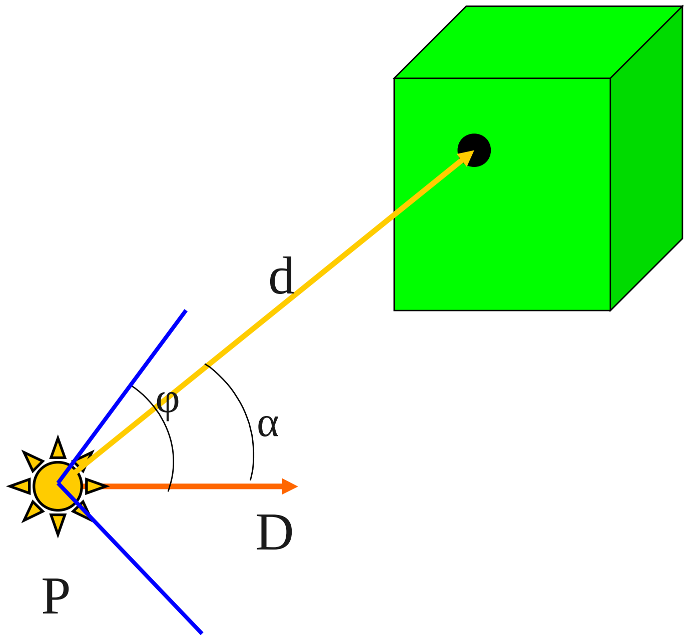
- Emits light in one direction
  - Can be regarded as point light source at infinity
  - E.g. sun
- Parameters
  - Intensity I_0
  - Direction D (d_x,d_y,d_z)
- $I_L = I_0$



Directional light source

Spot light source

- Point light source with direction
  - E.g. Spot light
- Parameters
  - Intensity I_0
  - Position P (p_x,p_y,p_z)
  - Direction D (d_x,d_y,d_z)
  - Maximum angle φ

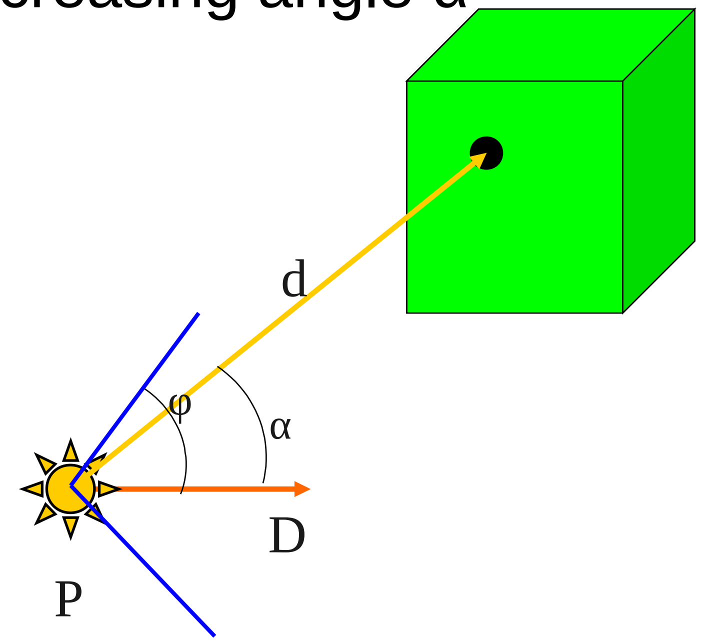


- $I_L = I_0$ if $\cos \alpha > \cos \varphi$

Point light source

Spot light source

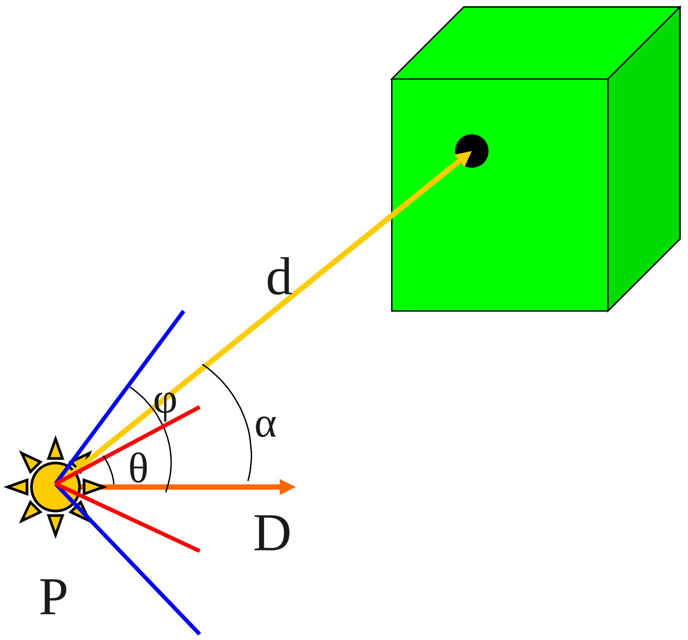
- Point light source with direction
  - E.g. Spot light
  - Decreasing intensity with increasing angle α
- Parameters
  - Intensity I_0
  - Position P (p_x,p_y,p_z)
  - Direction D (d_x,d_y,d_z)
  - Maximum angle φ
  - Exponent n



- $I_L = \cos^n\alpha \; I_0 \qquad \text{if } \cos\alpha > \cos\varphi$

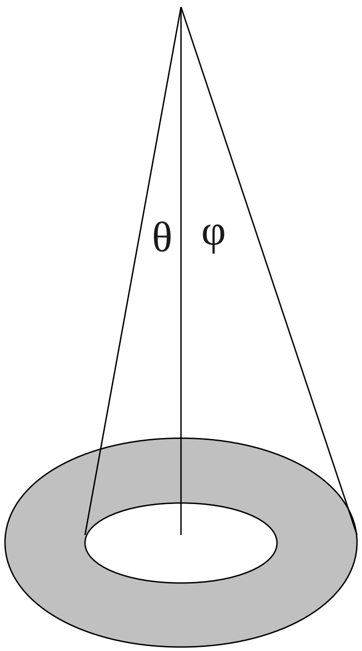
Spot light source

- Point light source with direction
  - E.g. Spot light
  - Decreasing intensity with increasing angle α
- Parameters
  - Intensity I_0
  - Position P (p_x,p_y,p_z)
  - Direction D (d_x,d_y,d_z)
  - "Hotspot" angle θ
  - Maximum angle φ
  - Exponent n



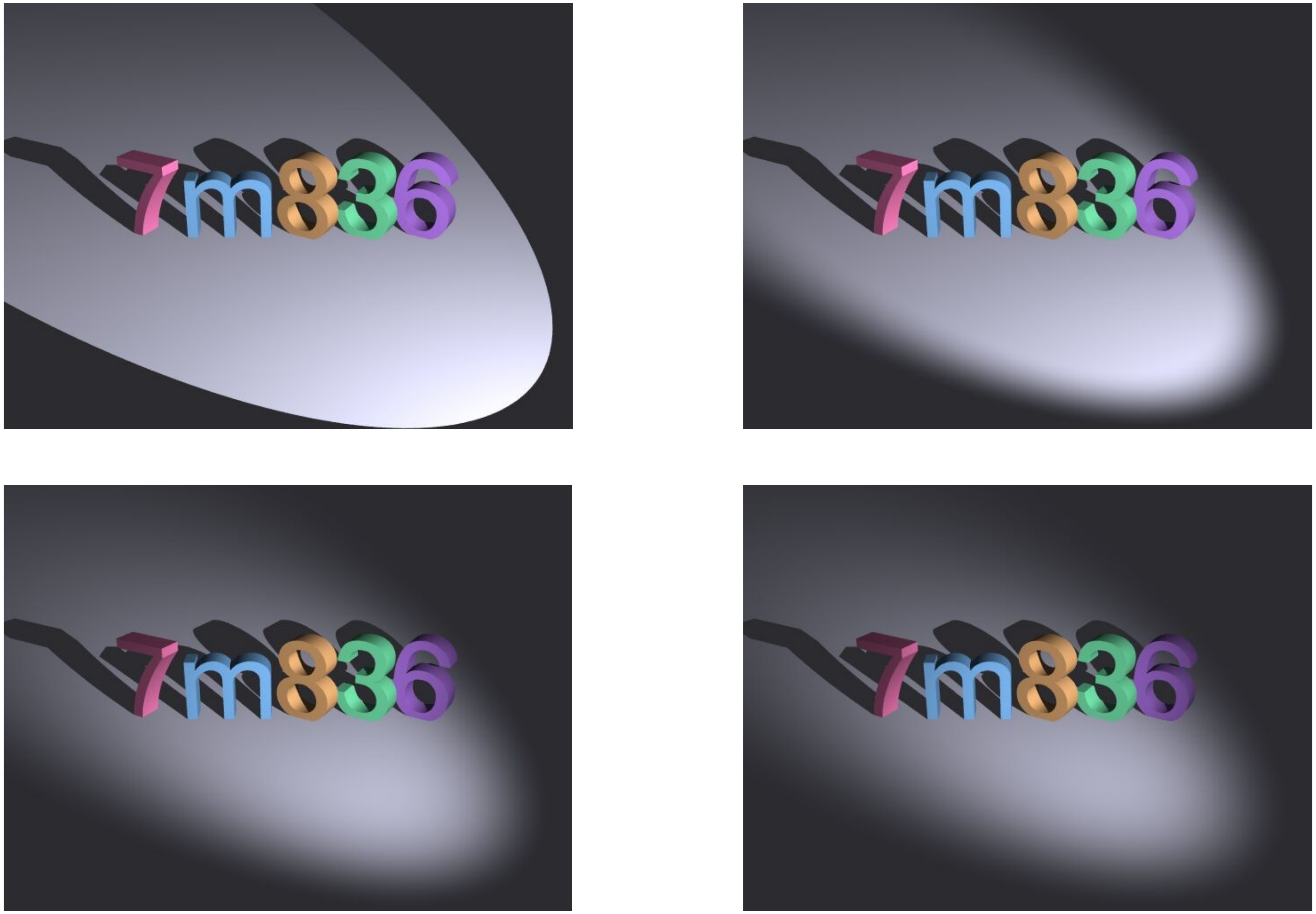
Spot light source

- $I_L = I_0$ if $\alpha \le \theta$
- $I_L = I_0 \cos^n\left(\frac{\alpha - \theta}{\varphi - \theta} * \frac{\pi}{2}\right)$ if $\theta < \alpha \le \varphi$
- $I_L = 0$ if $\varphi < \alpha$



Spot light source

Spot light source



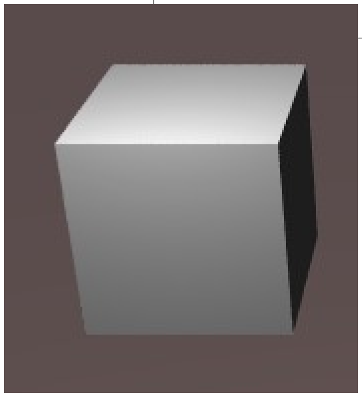
Other light sources

- Linear light sources
- Area sources
- Spherical lights
- …

Povray – light sources

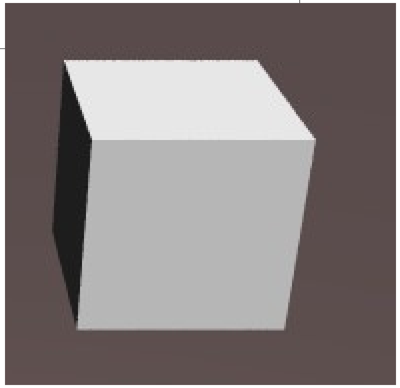
Point light

```
#declare pos=<-2,1,0>;
light_source {
        pos+<0,2,-1.5>
        color White
}
```



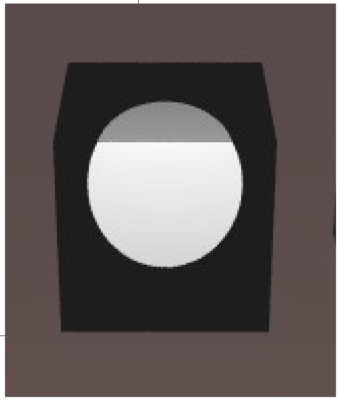
Directional light

```
light_source {
   <0,2,-1.5>
   color White
   parallel
   point_at <0,0,0>
}
```



Spot light

```
#declare pos=<0,1,0>;
light_source {
   pos+<0,2,-3>
   color White
   spotlight
   radius 10
   falloff 10
   tightness 0
   point_at pos
}
```



Attenuation

- Intensity of light decreases with squared distance:

  $I_L = I_0 / d^2$

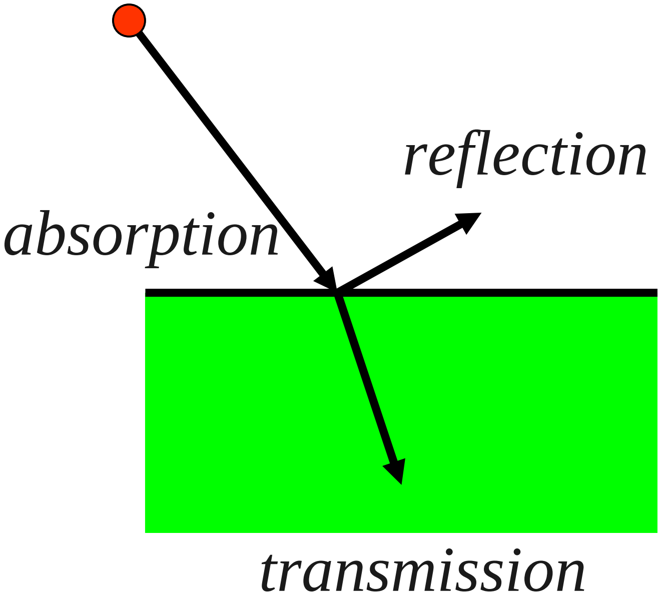
  - d is distance to light source
  - scenes are often to dark
- A more practical solution:

$$I_L = \frac{I_0}{a_c + a_l d + a_q d^2}$$

- (a_c, a_l, a_q) : the attenuation factors

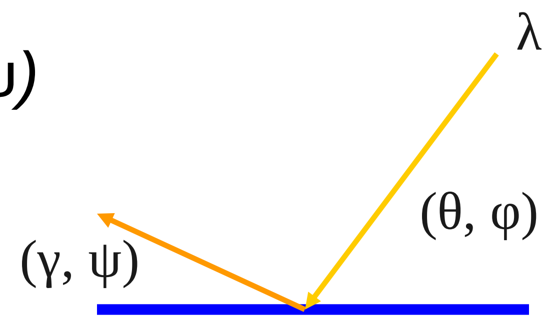
Surface illumination

- When light arrives at a surface, it can be
  - Absorbed
  - Reflected
  - Transmitted



Reflection

- R_s (θ, φ, γ, ψ, λ)
  - Amount of energy,
  - arriving from direction (θ, φ),
  - that is reflected in direction (γ, ψ)
  - with wavelength λ



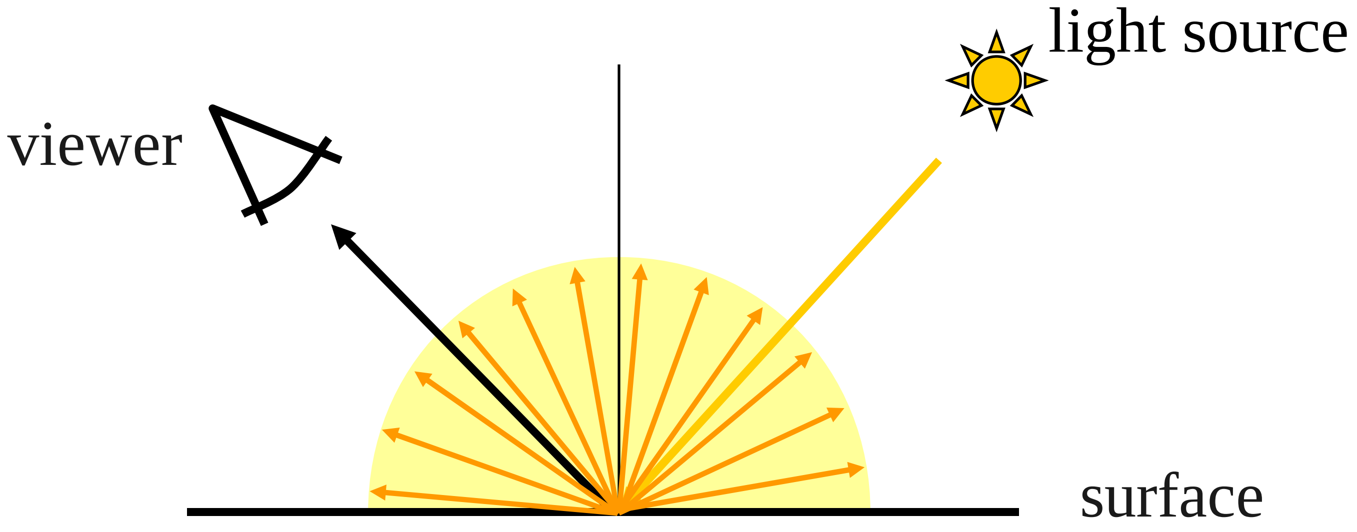
- Ideally
  - Describe this function for all combinations of θ, φ, γ, ψ, and λ
  - Impossible, simpler model(s) needed

Reflection model

- Total intensity on point “reflected” in certain direction is determined by:
  - Diffusely reflected light
  - Specularly reflected light
  - Emission (for light sources)
  - Reflection of ambient light
- Intensity not computed for all wavelengths λ, only for R, G and B

Diffuse reflection

- Incoming light is reflected equally in all directions
- Amount of reflected light depends only on angle of incident light

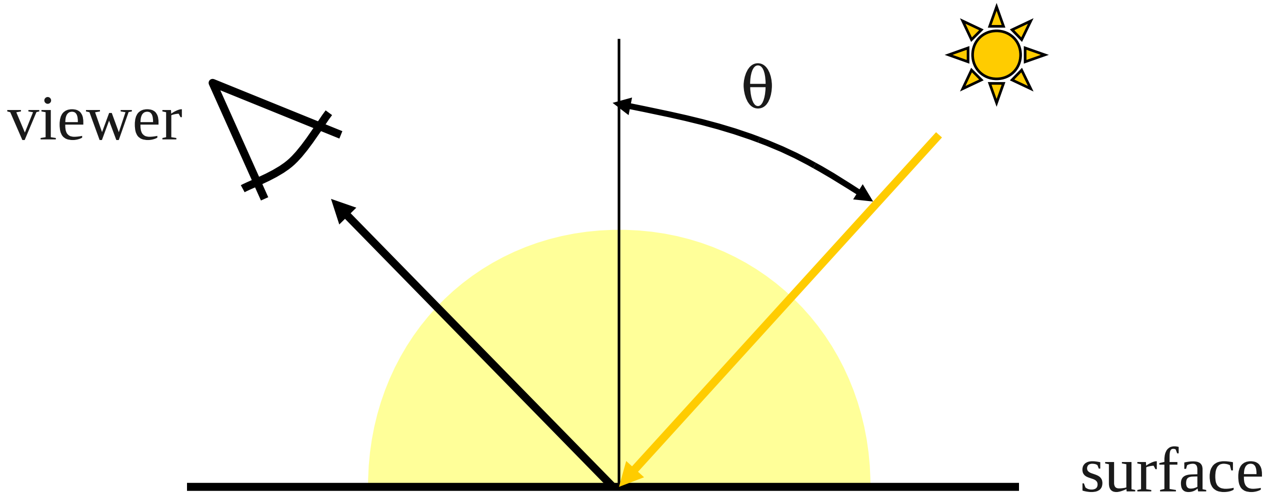


Diffuse reflection

Lambert' law: *Reflected energy from a surface is proportional to the cosine of the angle of direction of incoming light and normal of surface:*

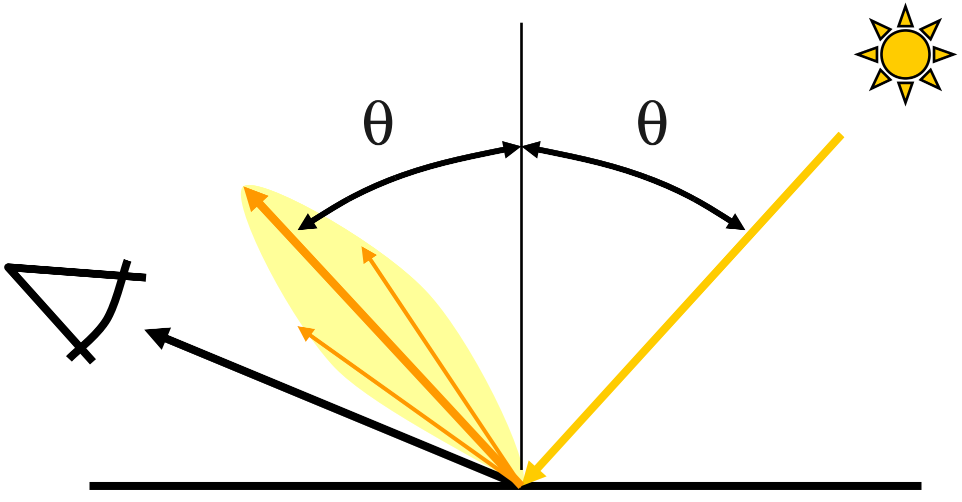
$$I_d = k_d I_L \cos(\theta)$$

where k_d is the diffuse reflection coefficient.



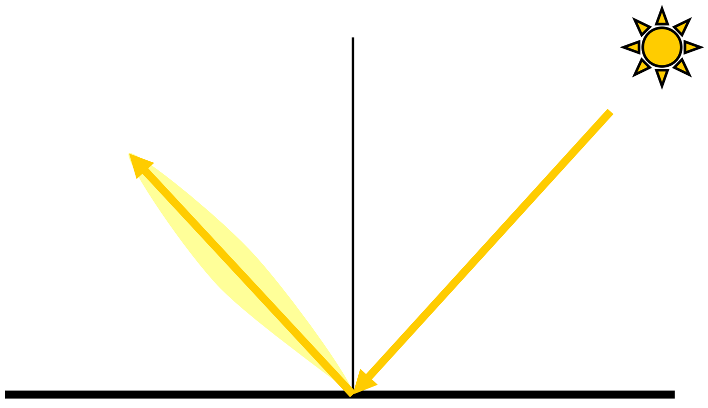
Specular reflection

- Models reflections of shiny surfaces

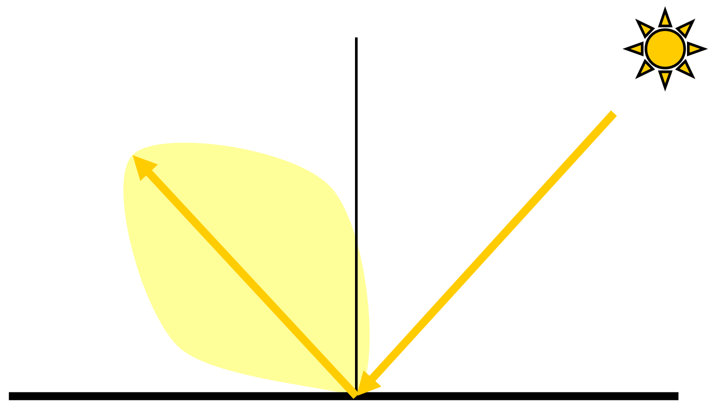


Specular reflection

- Shininess depends on roughness of the surface
- Incident light is not reflected equally in all directions
- Reflection strongest near mirror angle



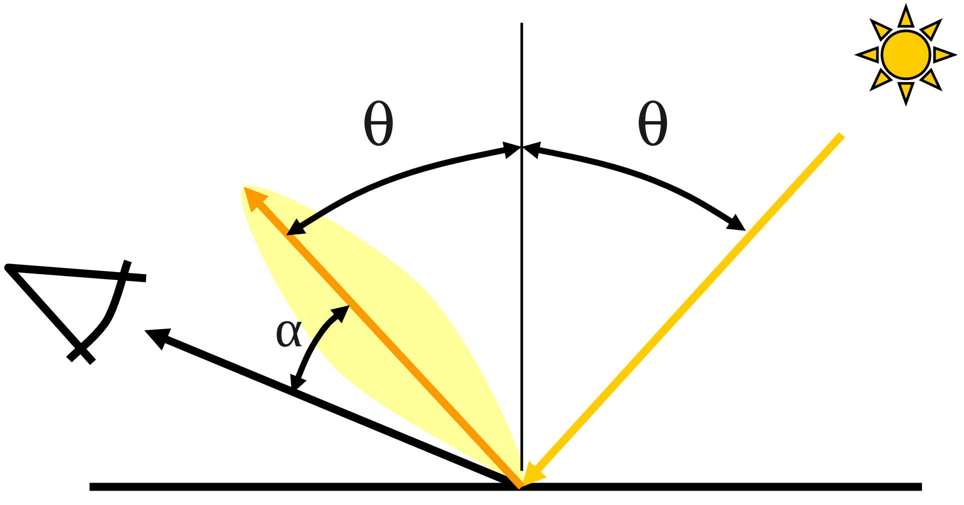
very shiny



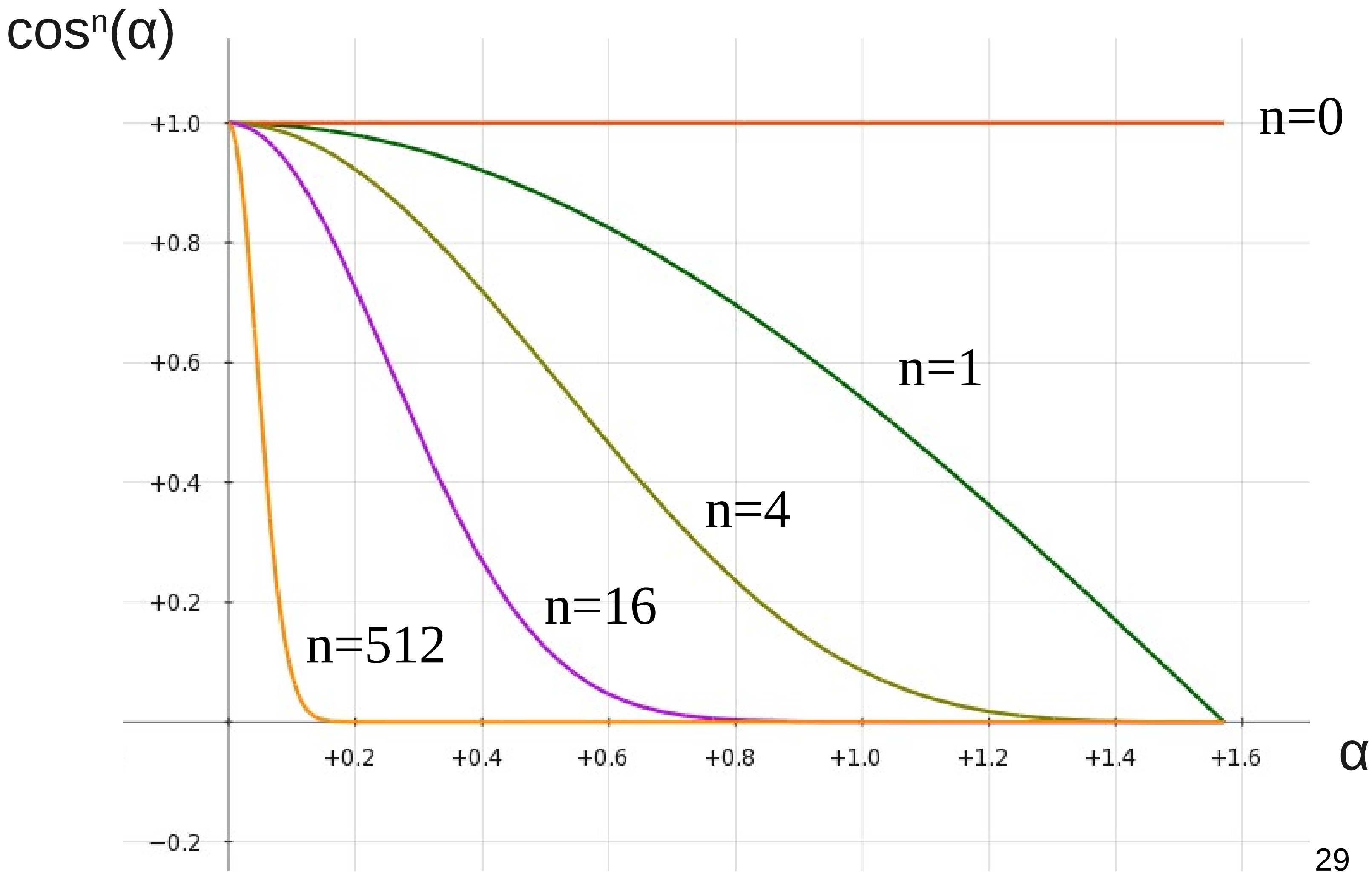
less shiny

Specular reflection / Phong

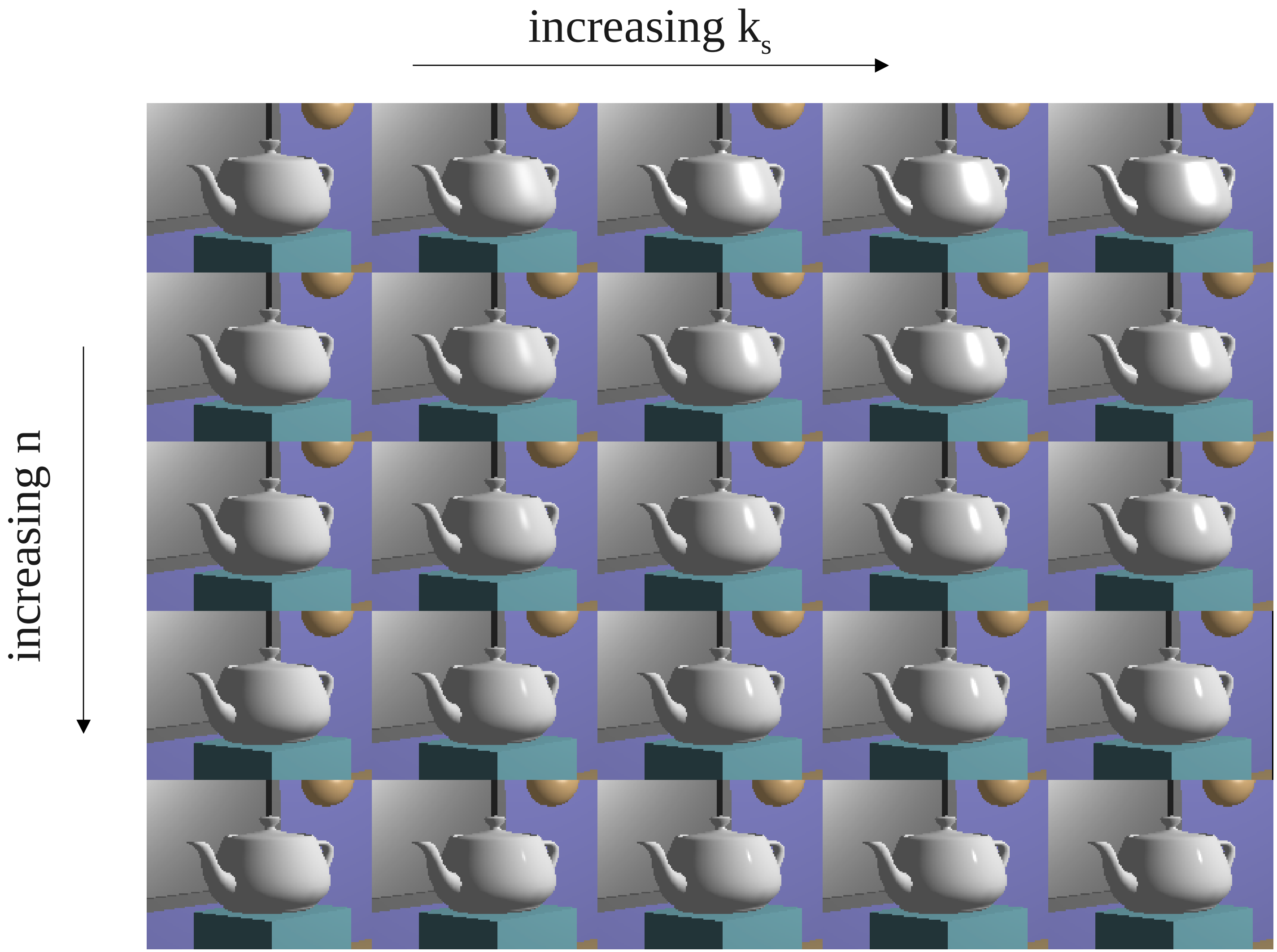
- Light intensity observed by viewer depends on angle of incident light and angle to viewer
- Phong model: $I_s = k_s I_L \cos^n(\alpha)$
  - k_s is specular reflection coefficient
  - n is specular reflection exponent



Specular reflection

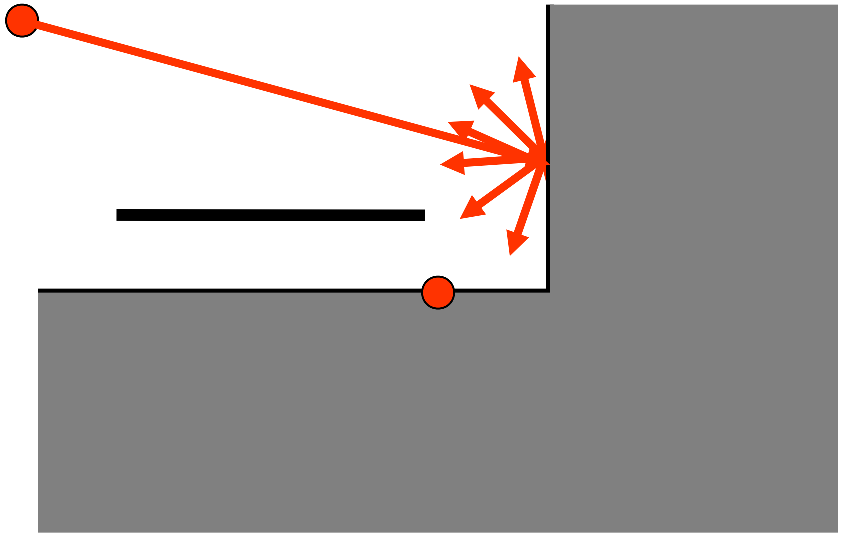


Specular reflection / Phong



Ambient light

Even though parts of the scene are not directly lit by light sources, they can still be visible because of indirect illumination.



Ambient light

- “Ambient” light I_A is an approach to this indirect illumination
- Ambient light is independent of light sources and viewer position
- Contribution of ambient light $I = k_a I_a$
  - k_a is ambient reflection coefficient
  - I_a is ambient light intensity (constant over scene)

- Indirect illumination can be computed much better: e.g. *radiosity*

Emission

Emission I_e represents light directly emitted by a surface

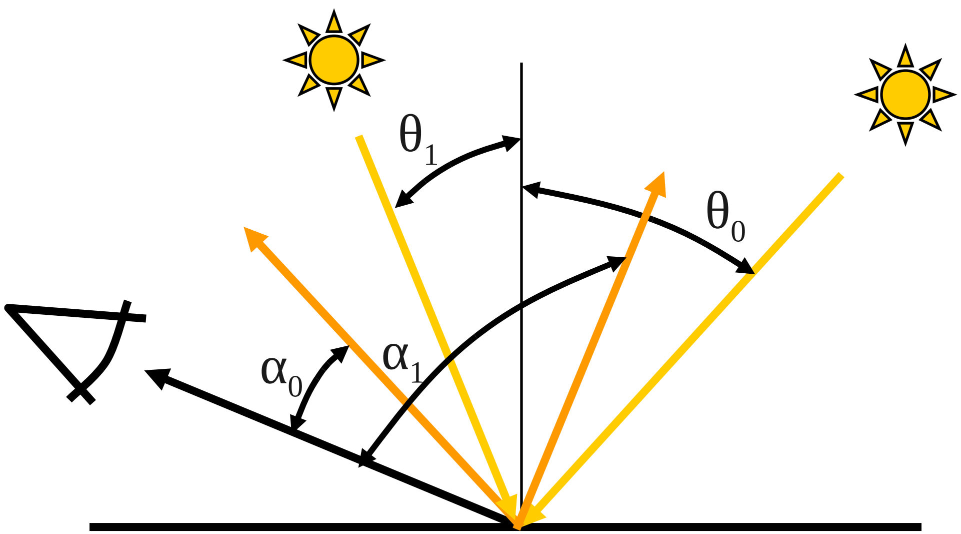
- Used for light sources

Emission ≠ 0

Total illumination Phong model

- Total intensity I_{total} at point P seen by viewer is

$$I_{total} = I_e + k_a I_a + \sum_i I_i \left(k_d \cos(\theta_i) + k_s \cos^n(\alpha_i) \right)$$



Total illumination Phong model

- In previous formula k_a, k_d dependent on wavelength (different values for R, G and B)
- Often division into color and reflection coefficient of surface

$$I_{totaal} = I_e + k_a C_a I_a + \sum_i I_i \left(k_d C_d \cos(\theta_i) + k_s C_s \cos^n(\alpha_i) \right)$$

where:

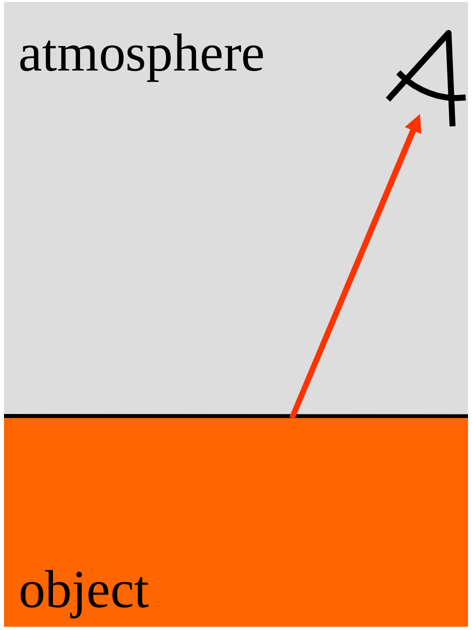
- $0 \leq k_a, k_s, k_s \leq 1$
- C_a is ambient reflected color of surface
- C_d is diffuse reflected color of surface
- C_s is specular reflected color of surface

Only emission and ambient

Including diffuse reflection

Including specular reflection

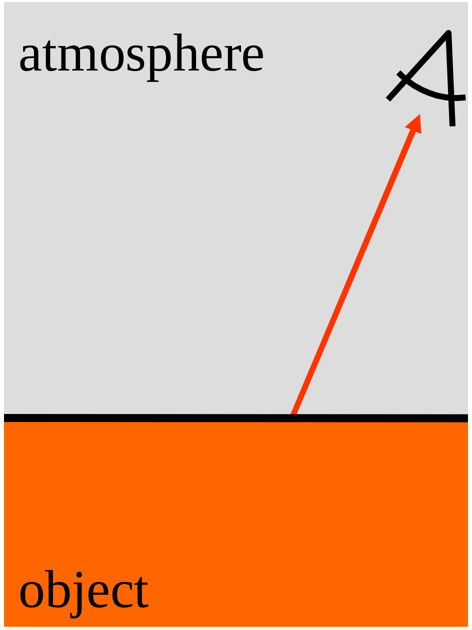
Atmospheric effects



Atmospheric effects:

- Dust, smoke, ..
- Colors are dimmed
- Objects less visible

Atmospheric effects



Perceived intensity I:

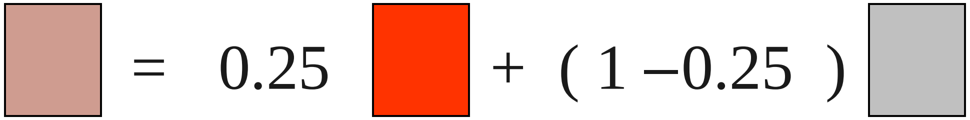
$$I = f(d)\, I_{object} + (1 - f(d))\, I_{atmosphere}$$

with d = distance travelled through medium and

$$f(d) = e^{-\rho d}$$

where ρ = attenuation factor, or

$$f(d) = (d - d_{min})/(d_{max} - d_{min})$$



Illumination model - summary

- Given model:
  - Simple
  - Effective
  - Not real world

- More advanced models:
  - Allows for angle between incident light and surface normal for specular reflection (Fresnel's law)
  - Better models for surface roughness
  - Better model for wavelength dependent reflection

Shading

- Illumination model computes illumination for a point on a surface
- But how do we compute the illumination for all points on all (visible) surfaces?

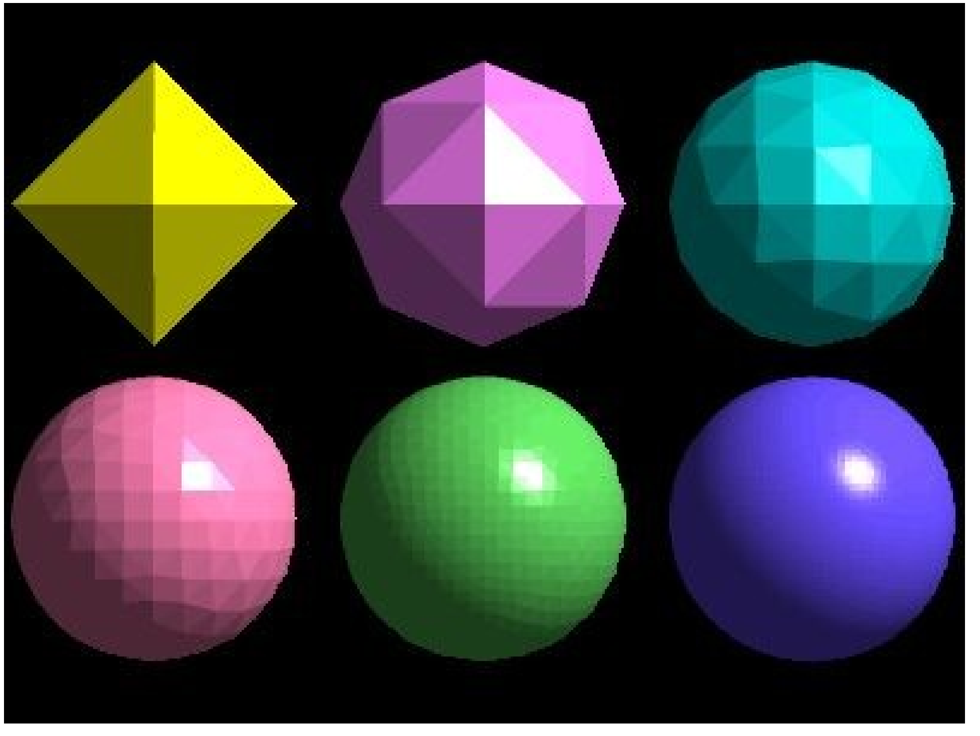
- Shading methods:
  - Flat shading
  - Gouraud shading
  - Phong shading

Flat shading

- Determine illumination for one point on polygon
- Use this illumination for all points on polygon

- Disadvantages:
  - Does not account for changing direction to light source over polygon
  - Does not account for changing direction to eye over polygon
  - Discontinuity at polygon boundaries (when a curved surface is approximated by a polygon mesh

Flat shading

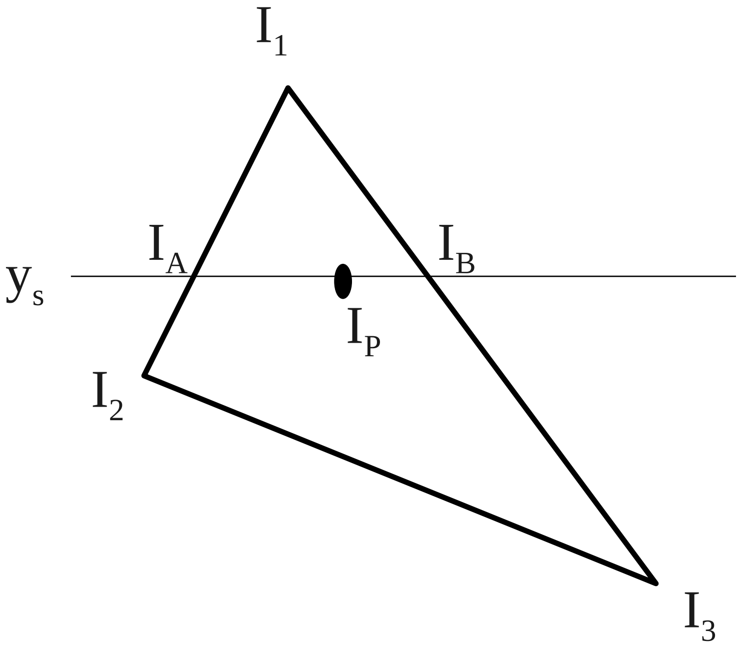


Gouraud shading

- Based on interpolation of illumination values
- Computes illumination for the vertices of a polygon

- Algorithm
  - for all vertices of a polygon
    - get vertex normal
    - compute vertex illumination using this normal
  - for all pixels in the projection of the polygon
    - compute pixel illumination by interpolation of vertex illumination

Gouraud shading



$$I_A = (1-\alpha)I_1 + \alpha I_2$$

$$\alpha = \frac{y_1 - y_s}{y_1 - y_2}$$

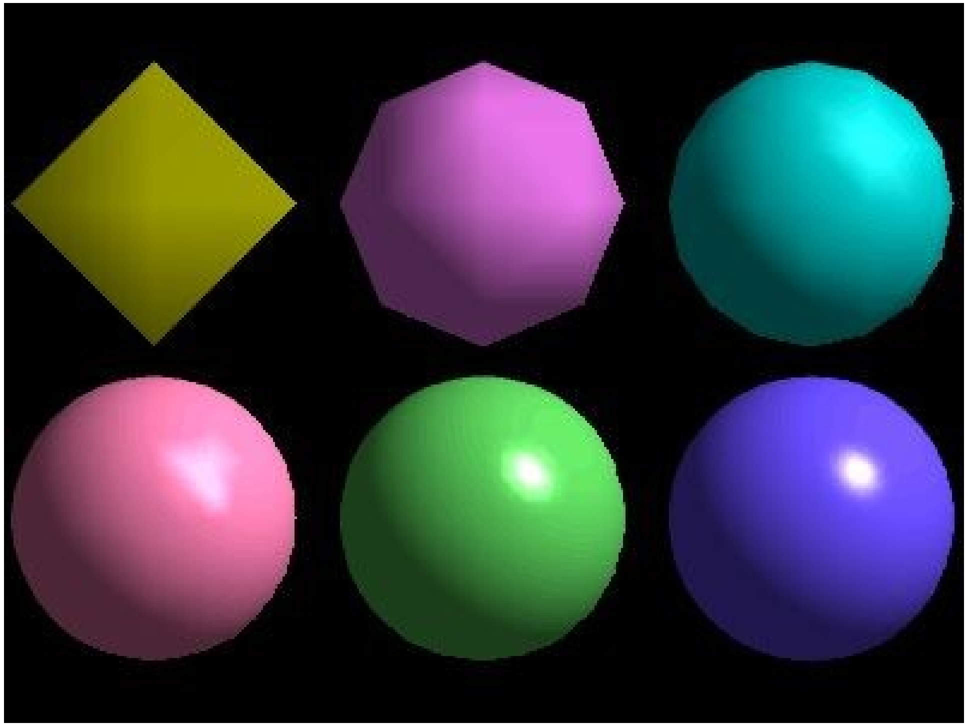
$$I_B = (1-\beta)I_1 + \beta I_2$$

$$\beta = \frac{y_1 - y_s}{y_1 - y_3}$$

$$I_P = (1-\gamma)I_A + \gamma I_B$$

$$\gamma = \frac{x_A - x_P}{x_A - x_B}$$

Gouraud shading



Gouraud shading

- Advantages
  - Interpolation is simple, hardware implementation
  - Continuous shading
  - Better results for curved surfaces

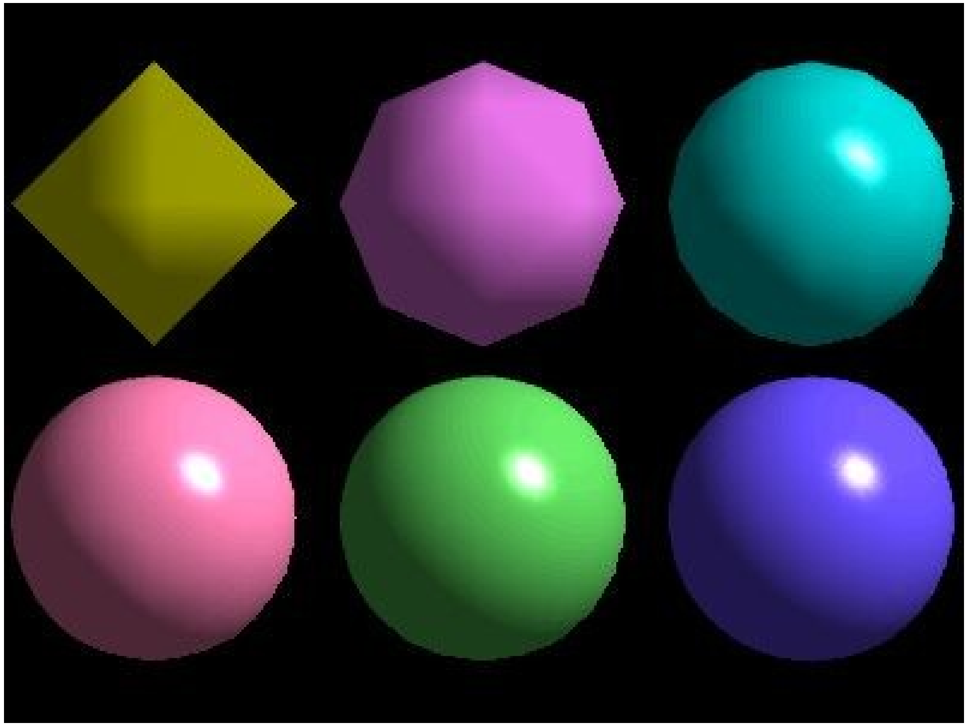
- Disadvantages
  - Subtle illumination effects (e.g. highlights) require high subdivision of surface in very small polygons
  - Mach banding

Phong shading

- Based on normal interpolation
- Computes illumination for each point of polygon

- Algorithm
  - for each vertex of polygon
    - get vertex normal
  - for each pixel in projection of polygon
    - compute point normal by interpolation of vertex normals
    - compute illumination using interpolated normal

Phong shading

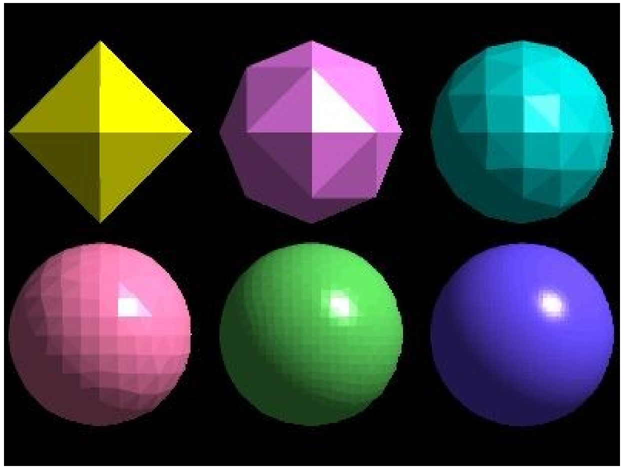


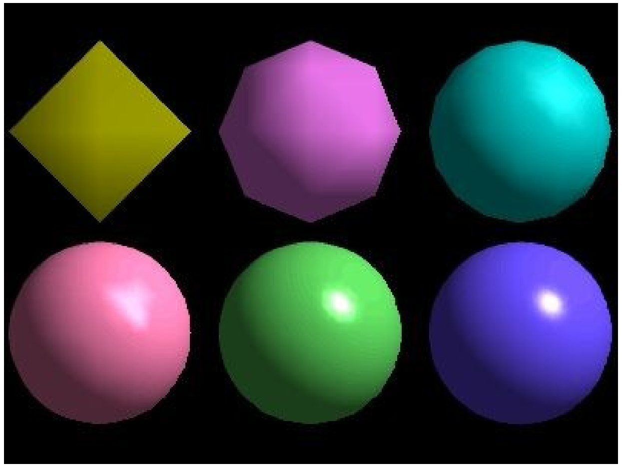
Phong shading

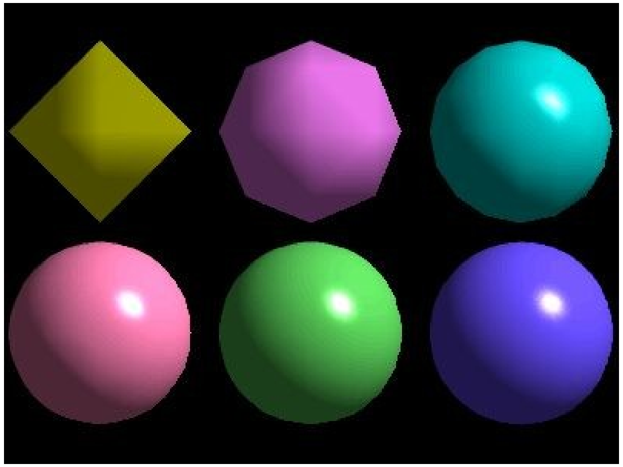
- Advantages
  - Much better results for curved surfaces
  - Nice highlights
  - No Mach banding

- Disadvantages
  - Computational expensive
    - Normal interpolation and application of illumination model for each pixel

Shading summarized







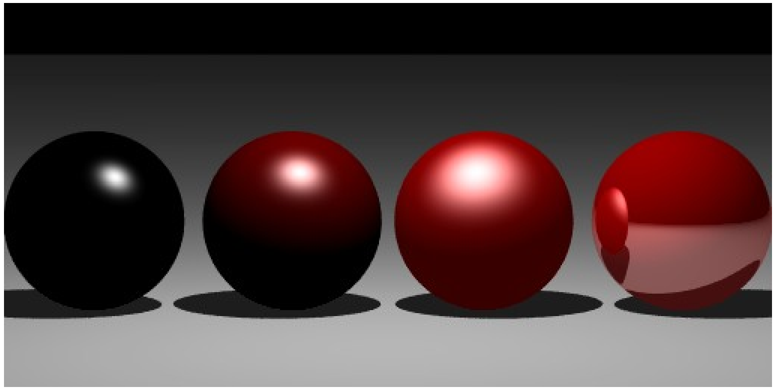
Shading

- Flat shading
  - 1x application of illumination model per polygon
  - 1 color per polygon
- Gouraud shading
  - 1x application of illumination model per vertex
  - Interpolated colors
- Phong shading
  - 1x application of illumination model per pixel
  - Nice highlights

Increase computation time

Increase quality

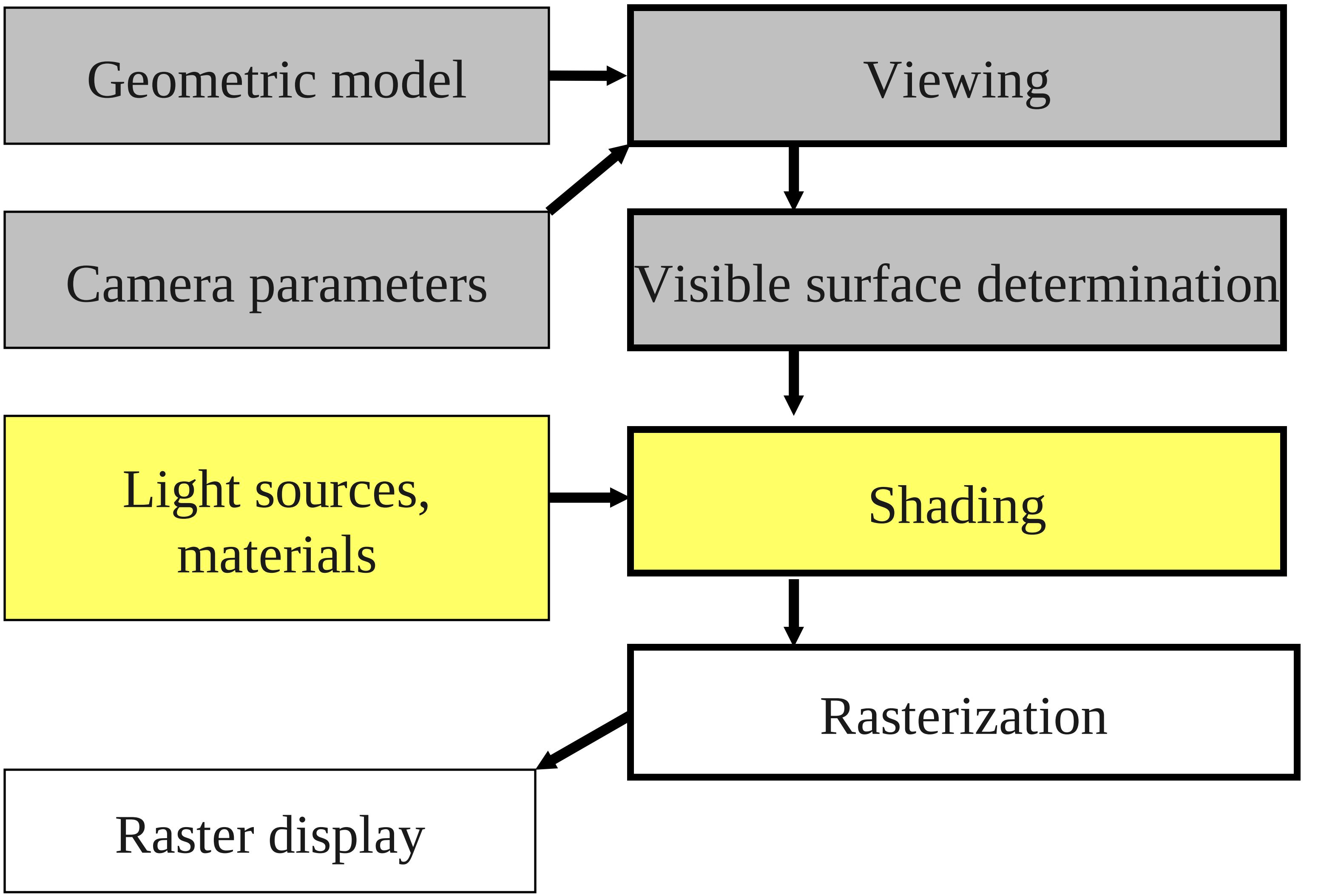
Povray – material properties



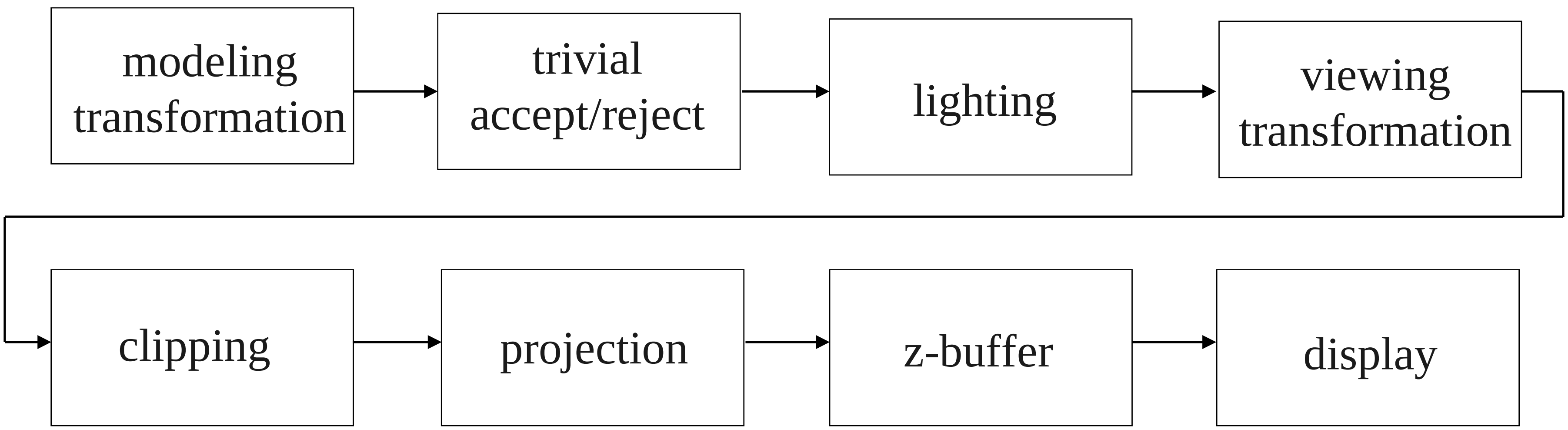
```
 ...
light_source {
   <0,5,-2>
   color White
 }
sphere { <-2.30,.7,0>, .7 texture { pigment {Red}
    finish { ambient 0.0 diffuse 0.0 phong 1 phong_size 20 }}}
sphere { <-0.75,.7,0>, .7 texture { pigment {Red}
    finish { ambient 0.0 diffuse 0.4 phong 1 phong_size 10 }}}
sphere { < 0.75,.7,0>, .7 texture { pigment {Red}
    finish { ambient 0.2 diffuse 0.4 phong 1 phong_size 3  }}}
sphere { < 2.30,.7,0>, .7 texture { pigment {Red}
    finish { ambient 0.2 diffuse 0.4 reflection 0.5       }}}
plane { <0,1,0>, 0 texture { pigment {White }}}
```

ambient= k_a, **diffuse**= k_d, **phong** = k_s, **phong_size** = n

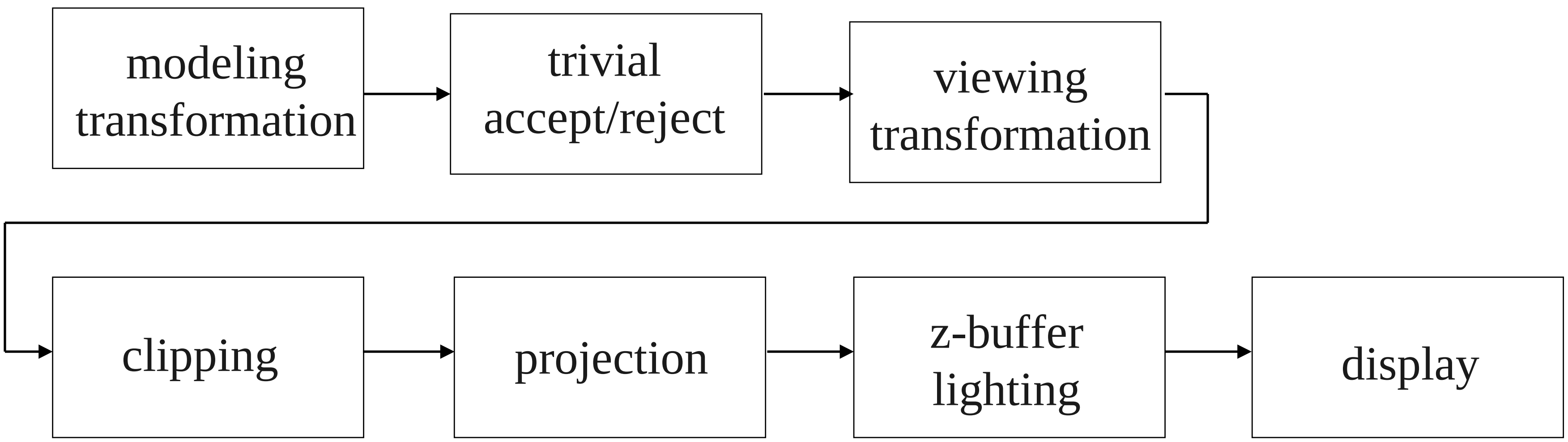
Grafische pijplijn



Z-buffer & Gouraud shading



Z-buffer & Phong shading



What is still missing?

- Shadows
- Area light sources
- “Real” mirroring
- Transparency
- Indirect diffuse reflection

Shadow

- Illumination with shadows

$$I_{totaal} = I_e + k_a C_a I_a + \sum_i I_i S_i \left(k_d C_d \cos(\theta_i) + k_s C_s \cos^n(\alpha_i) \right)$$

- S_i = 0 if light of source *i* is blocked
  = 1 if light of source *i* is not blocked

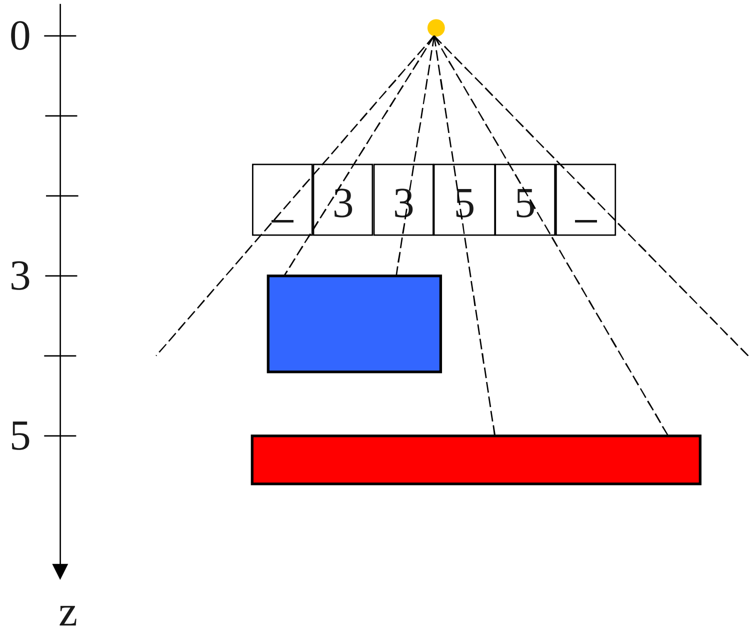
- Several methods to compute shadow
  - Shadow buffer
  - Ray casting
  - Shadow volumes
  - ..

Shadow buffer

- Based on z-buffer algorithm
- Rendering in 2 steps
  - Compute depth information
    - Z-buffer from light source (= shadow buffer) to compute shadows
  - During “normal” rendering, when computing illumination of point read from shadow buffer if point is lit by light source

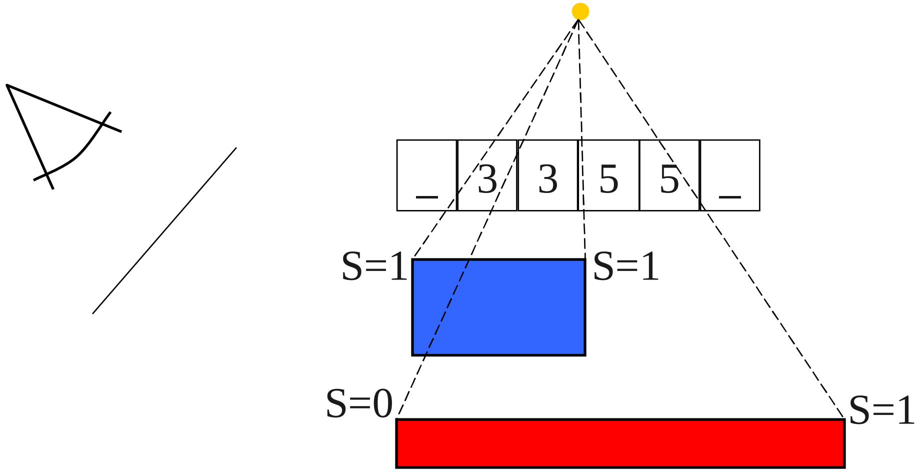
Shadow buffer – step 1

- "Render" scene with light source position as viewpoint and store depth results in depth buffer (no illumination)
- sbuffer(*x*,*y*) = minimum z-value after projection of all polygons on (*x*,*y*), i,e. distance to object nearest to light source

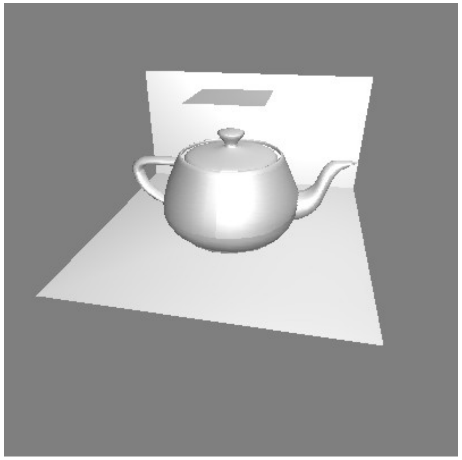


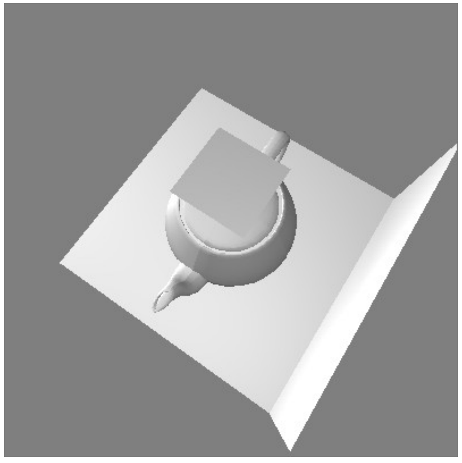
Shadow buffer – step 2

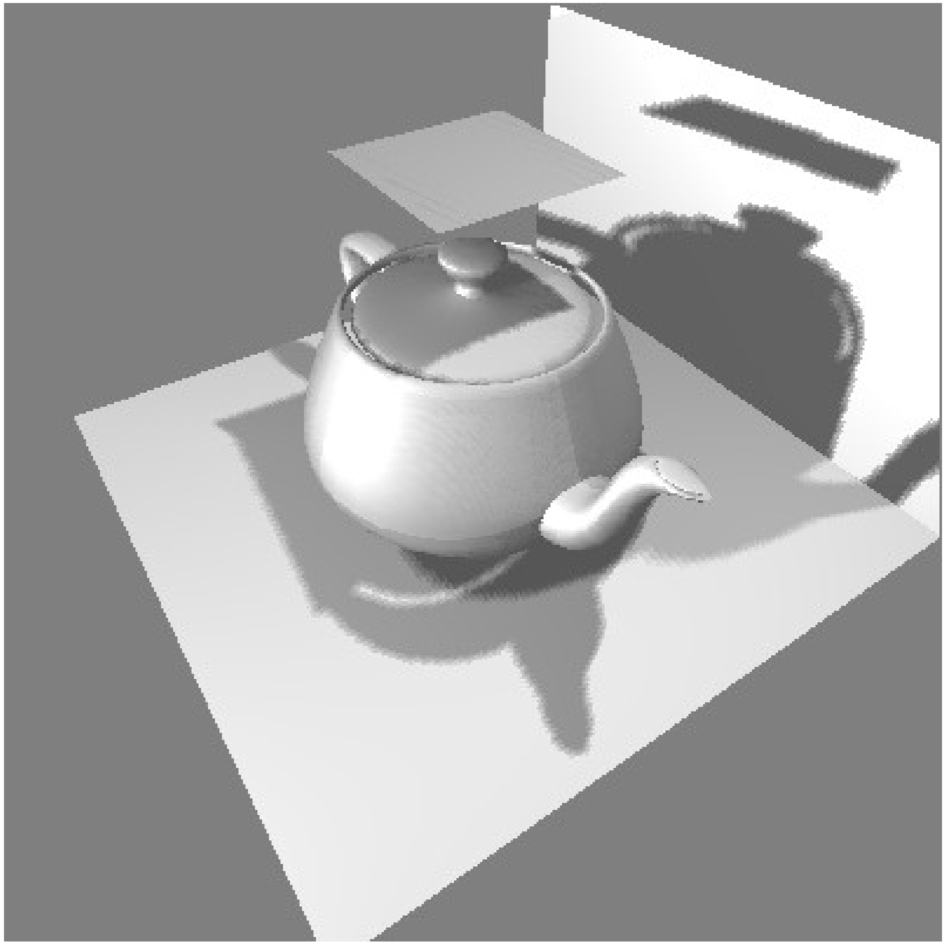
- At rendering, when computing illumination of point (x,y,z), transform point into light source coordinate system, resulting in point (x’,y’,z’)
- S_i = 1 als z’ <= sbuffer(x’,y’)
  = 0 als z’ > sbuffer(x’,y’)



Shadow buffer







What is still missing?

- Area light sources
- “Real” mirroring
- Transparency
- Indirect diffuse reflection
- Influence of atmosphere