Computer Graphics

Lighting - Local

Mark Pauly

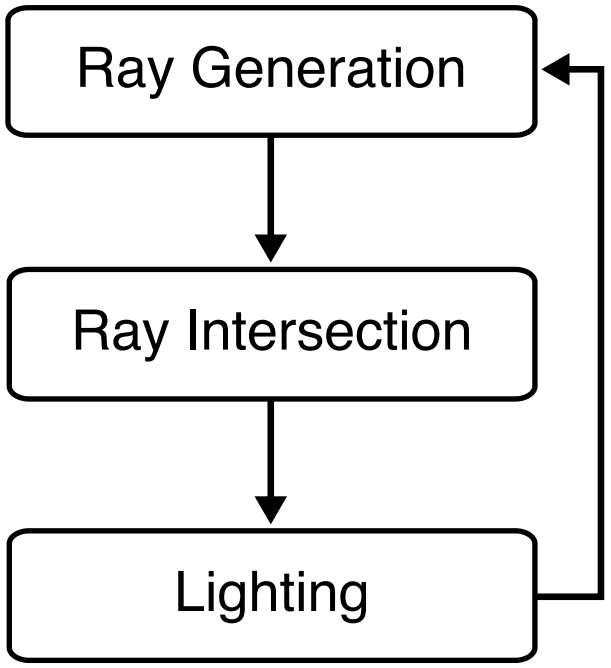
Geometric Computing Laboratory

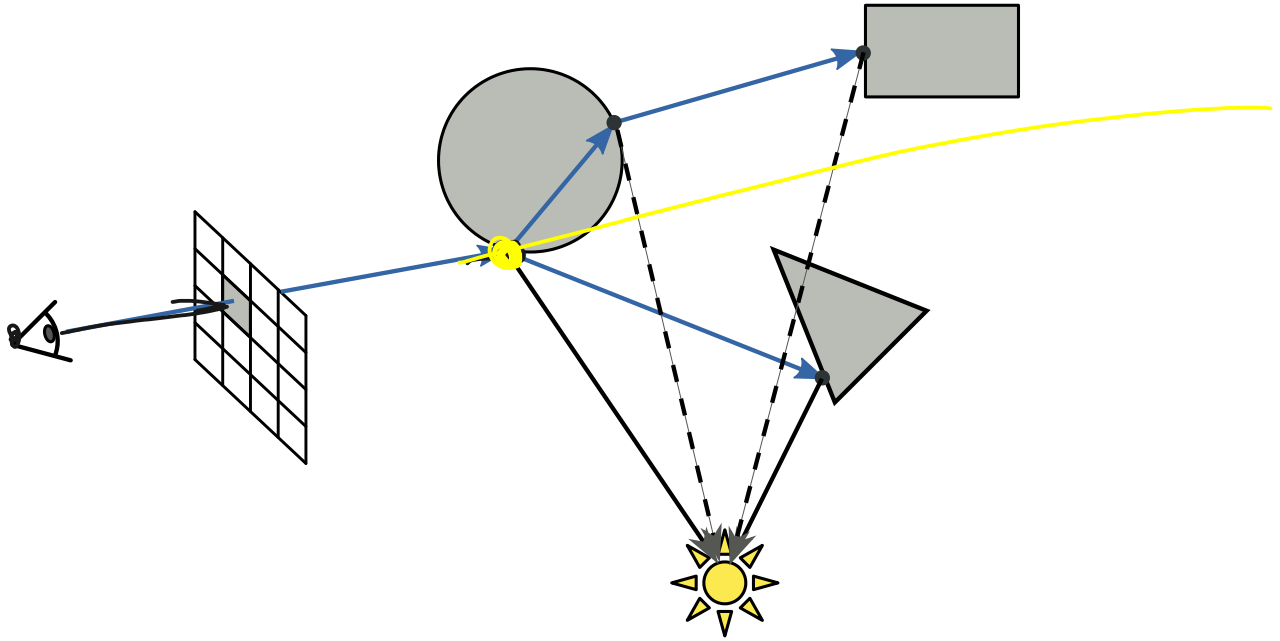
And the Oscar goes to …

Blender!

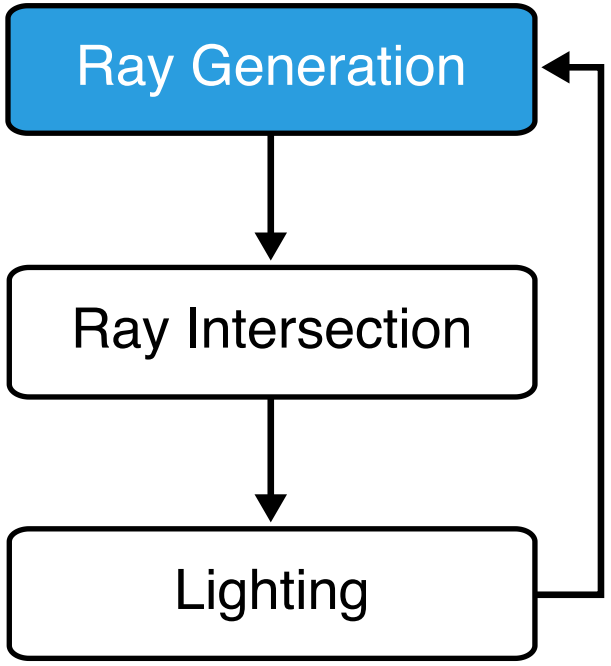
Source: The Guardian

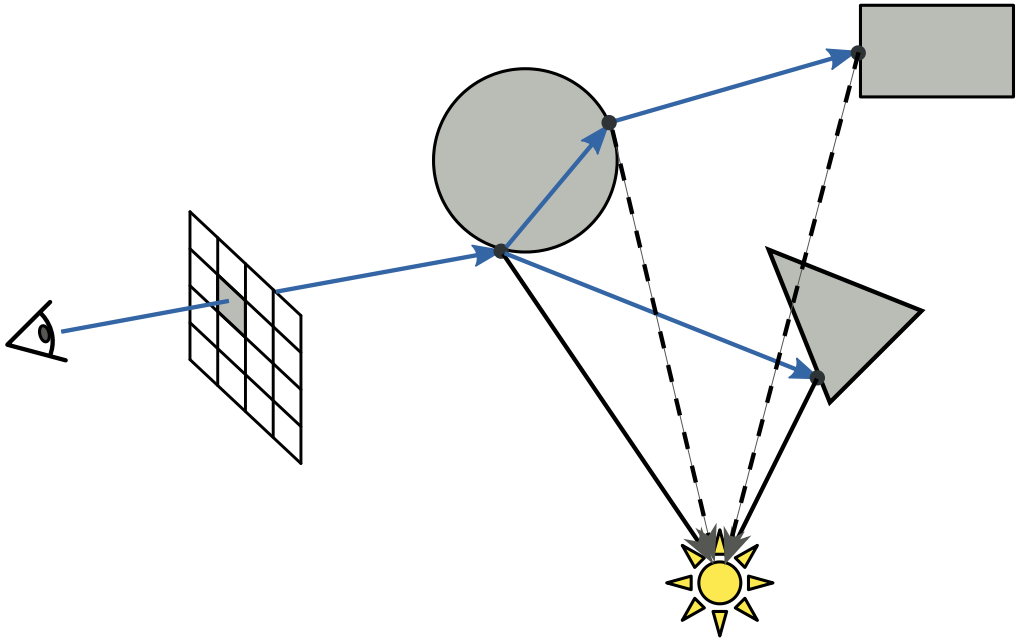
Ray Tracing Operators





Ray Tracing Operators

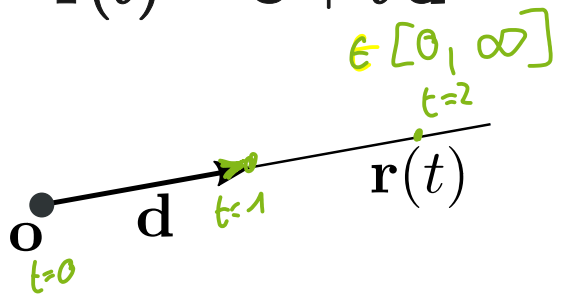




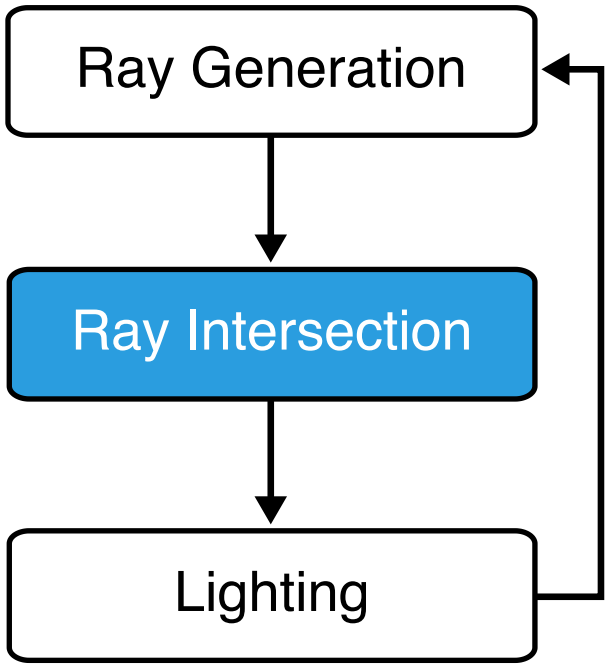
Ray Equation

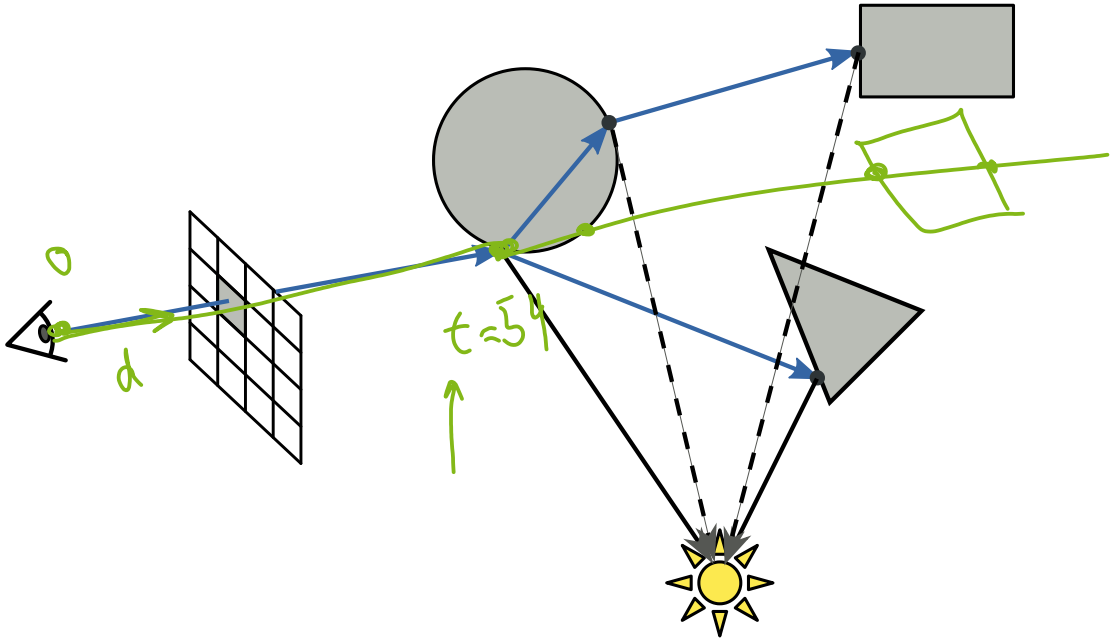
Explicit equation for ray $\mathbf{r}(t)$ starting at origin $\mathbf{o}$ and going in (normalized) direction $\mathbf{d}$:

$$\mathbf{r}(t) = \mathbf{o} + t\,\mathbf{d}$$



Ray Tracing Operators





Ray-Sphere Intersection

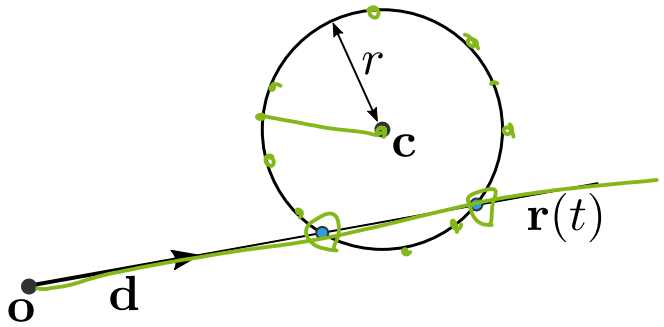
- Let us use the *implicit* representation of a sphere with center $\mathbf{c}$ and radius r:

$$\| \mathbf{x} - \mathbf{c} \| - r = 0$$

- Insert *explicit* ray equation into *implicit* sphere equation:

$$\| \mathbf{o} + t\mathbf{d} - \mathbf{c} \| - r = 0$$

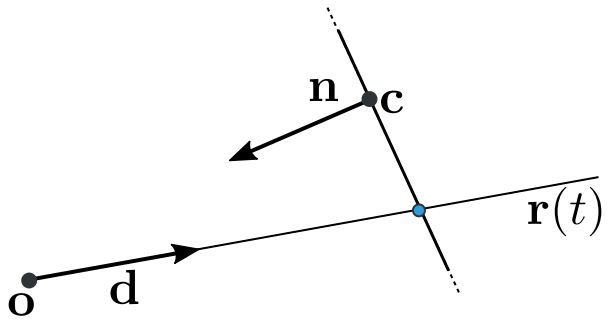
and solve for t.



Ray-Plane Equation

- Implicit equation for a plane with normal $\mathbf{n}$ centered at point $\mathbf{c}$:

$$\mathbf{n}^\mathsf{T}(\mathbf{x} - \mathbf{c}) = 0$$



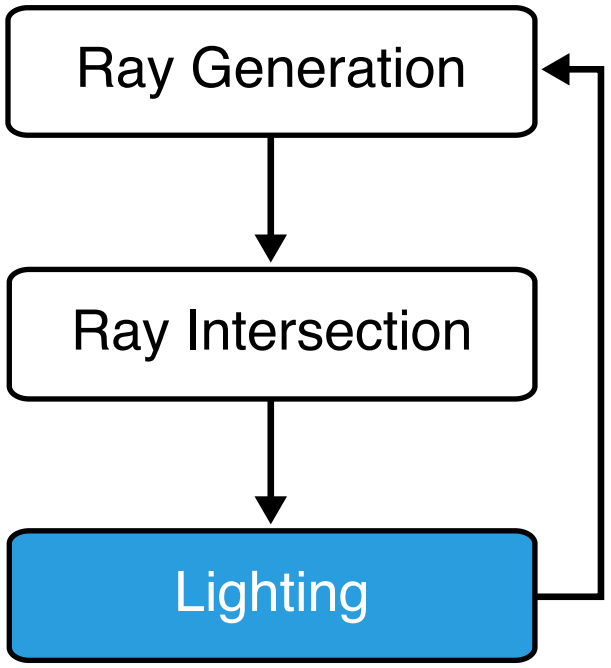
- Or use plane distance from origin $d = \mathbf{n}^\mathsf{T}\mathbf{c}$:

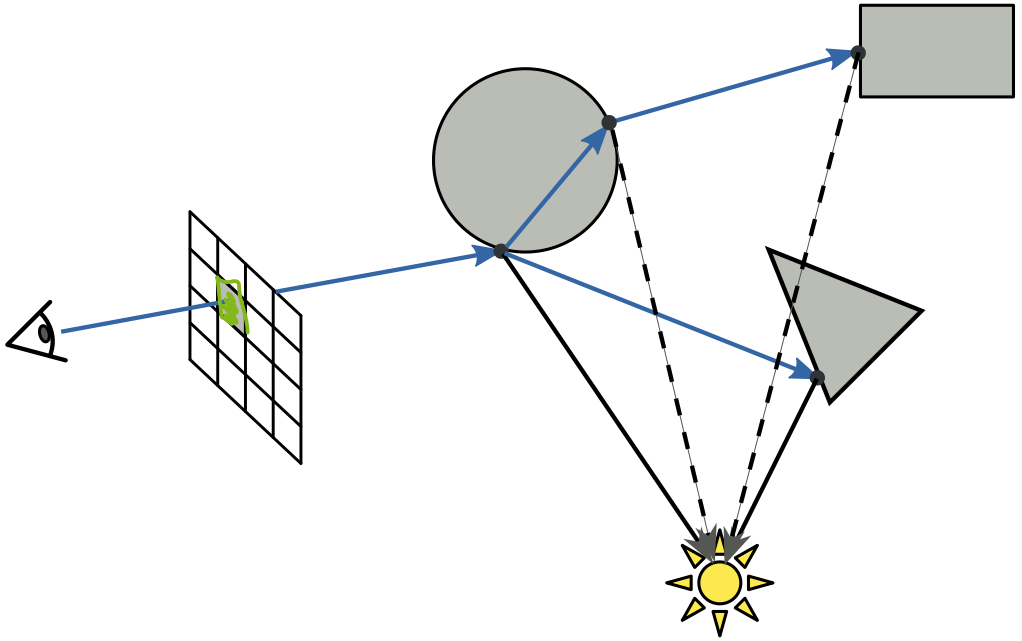
$$\mathbf{n}^\mathsf{T}\mathbf{x} - d = 0$$

- Insert *explicit* ray equation into *implicit* plane equation and solve for t:

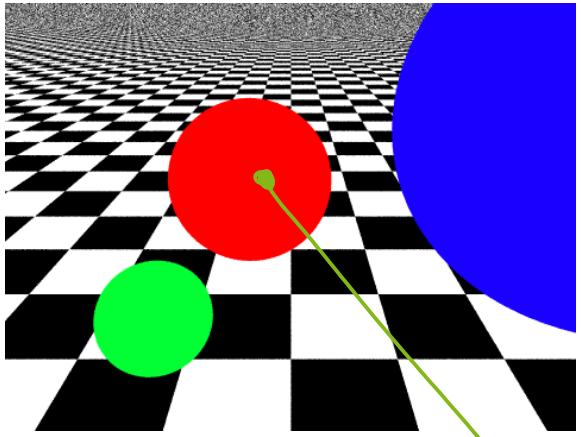
$$\mathbf{n}^\mathsf{T}(\mathbf{o} + t\,\mathbf{d}) - d = 0$$

Ray Tracing Operators

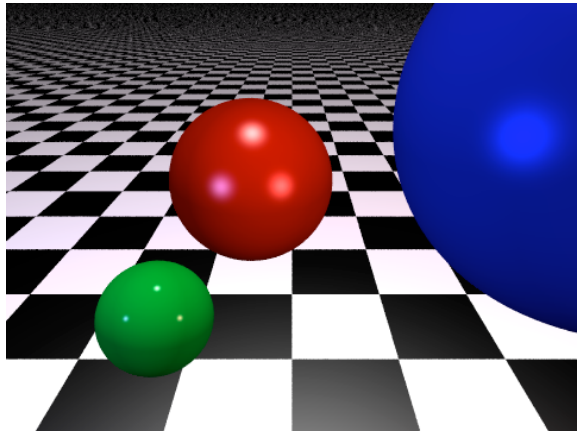




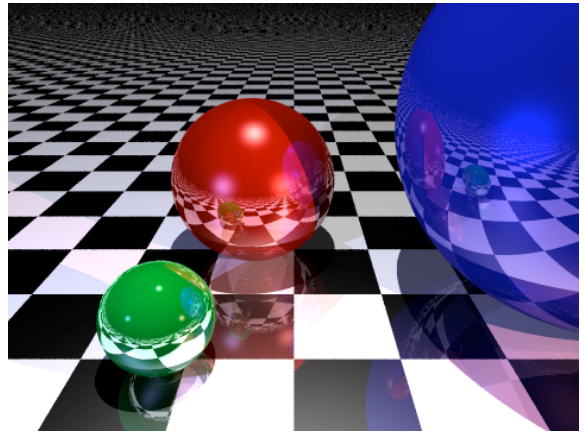
Lighting is important!



no illumination



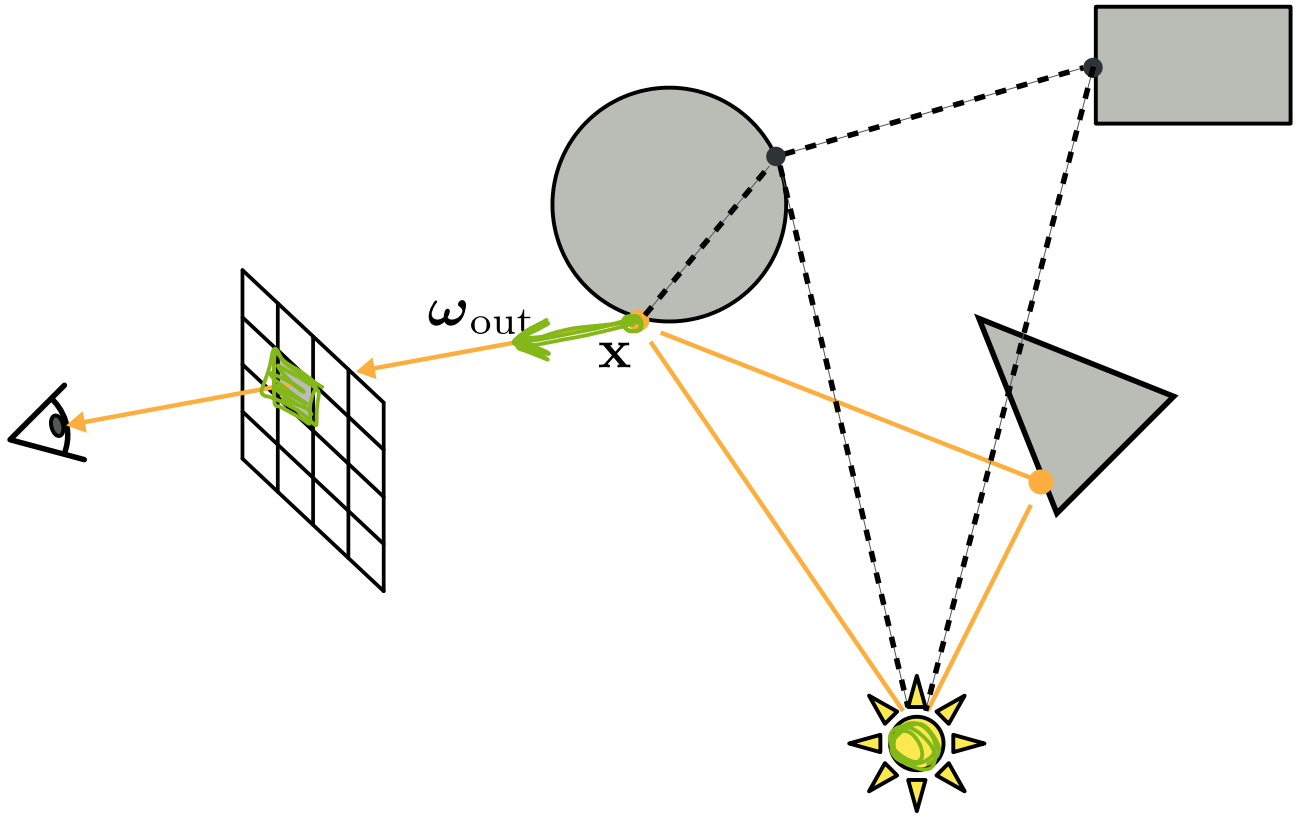
local illumination



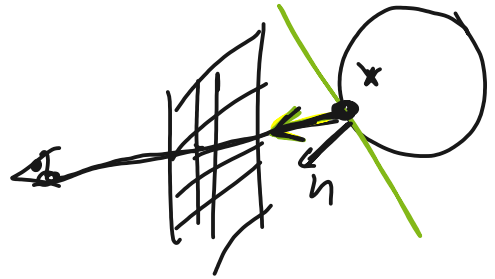
global illumination

Surface Reflectance

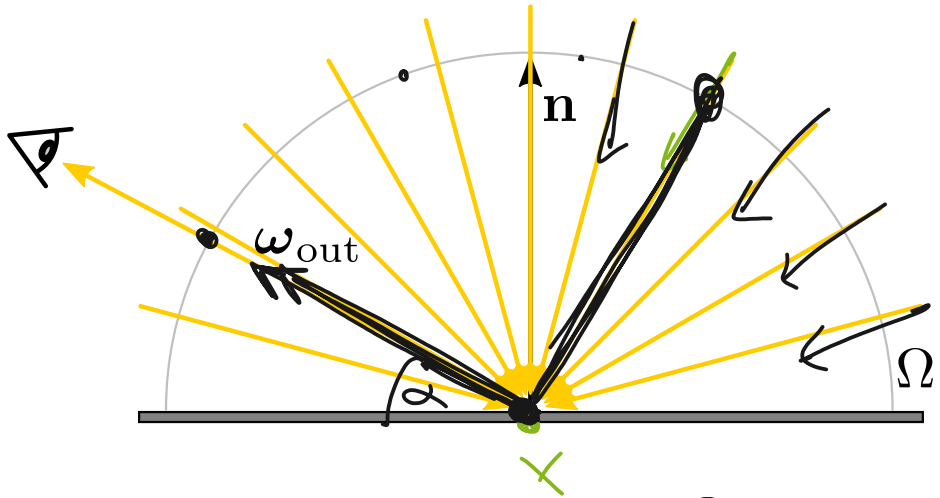
- How much light is leaving point $\mathbf{x}$ in direction ω_{out}?



Surface Reflectance



- How much light is leaving point $\mathbf{x}$ in direction ω_{out}?

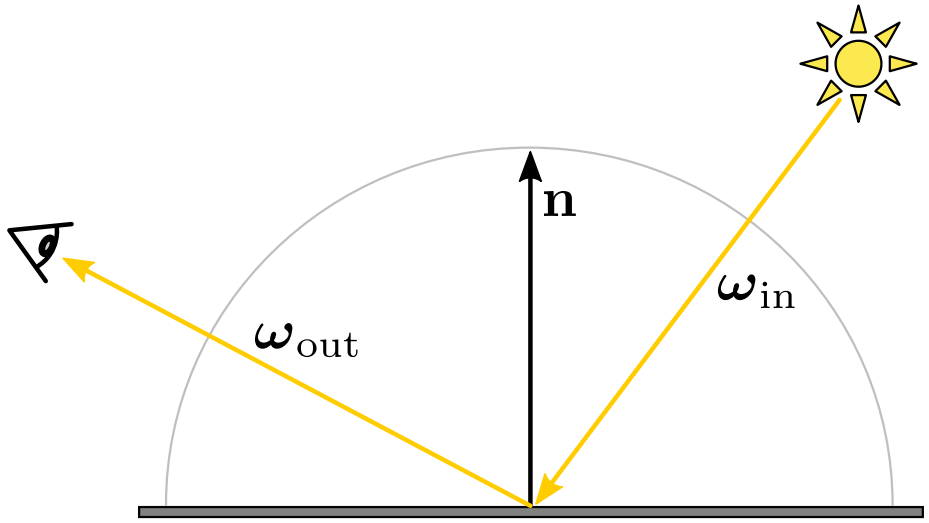


- Collect incoming light L_{in} from all directions $\omega_{\text{in}} \in \Omega$

$$L_{\text{out}}(\omega_{\text{out}}) = \int_{\Omega} f(\omega_{\text{in}}, \omega_{\text{out}}) L_{\text{in}}(\omega_{\text{in}}) \cos(\theta_{\text{in}}) \, \mathrm{d}\omega_{\text{in}}$$

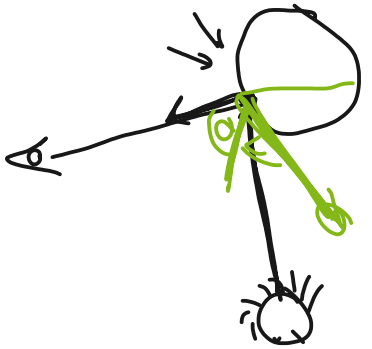
Bidirectional Reflectance Distribution Function

- How much light coming in from direction ω_{in} is reflected out in direction ω_{out}?



- Determined by the object's *BRDF* $f(\omega_{\text{in}}, \omega_{\text{out}})$
  - Bidirectional Reflectance Distribution Function
  - General description of an object's material

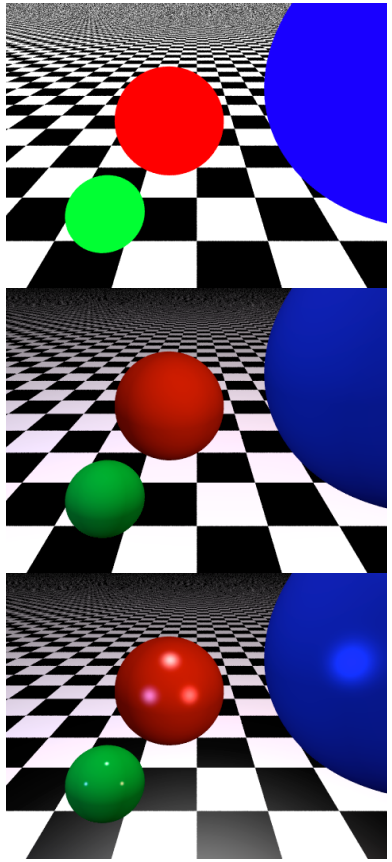
Ray Tracing Approximations



$$L_{\text{out}}(\omega_{\text{out}}) = \int_{\Omega} f(\omega_{\text{in}}, \omega_{\text{out}})\, L_{\text{in}}(\omega_{\text{in}})\, \cos(\theta_{\text{in}})\, \mathrm{d}\omega_{\text{in}}$$

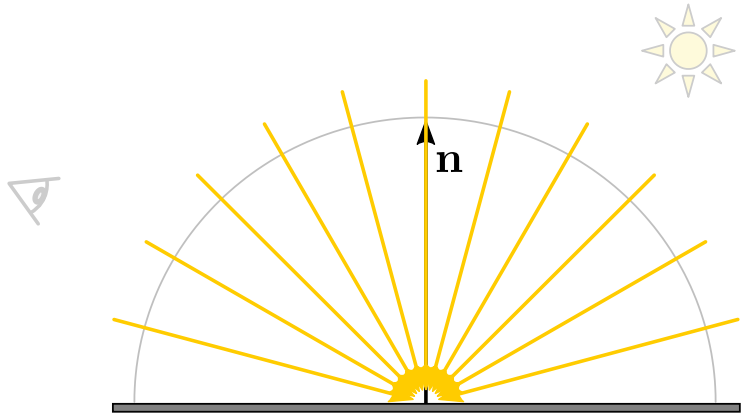
- **Ray Tracing** approximates integral by three directions
  - directions toward light sources, directions of perfect reflection and perfect refraction
- **Phong Lighting** approximates BRDF by three components
  - ambient light, diffuse reflection, specular reflection
  - *Phong Lighting handles direct, local illumination only.*

Phong Lighting Model



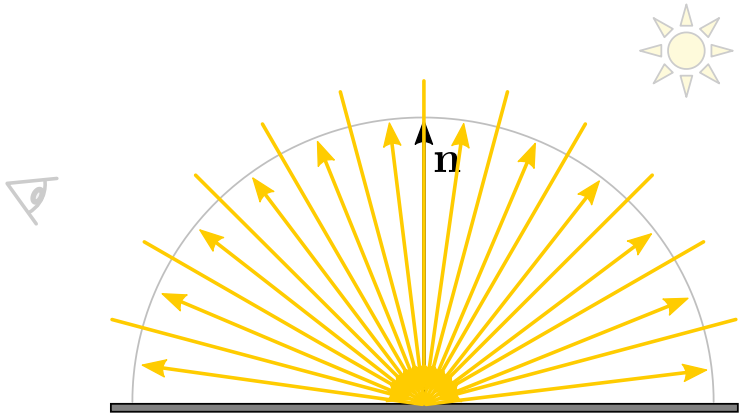
- **Ambient lighting**
  - approximate global light transport / exchange
  - uniform in, uniform out
- **Diffuse lighting**
  - dull / mat surfaces
  - directed in, uniform out
- **Specular lighting**
  - shiny surfaces
  - directed in, directed out

Ambient Light



uniformly incoming light

Ambient Light

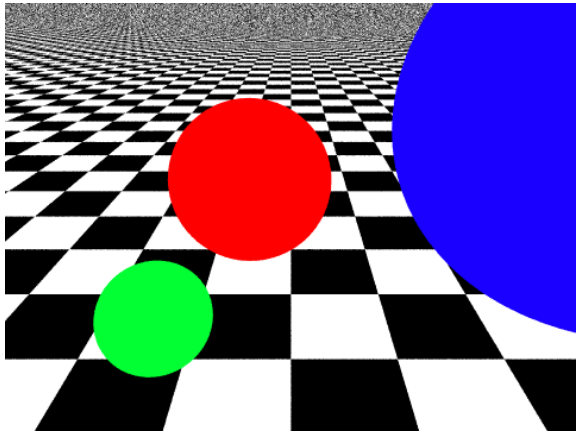


uniformly outgoing light

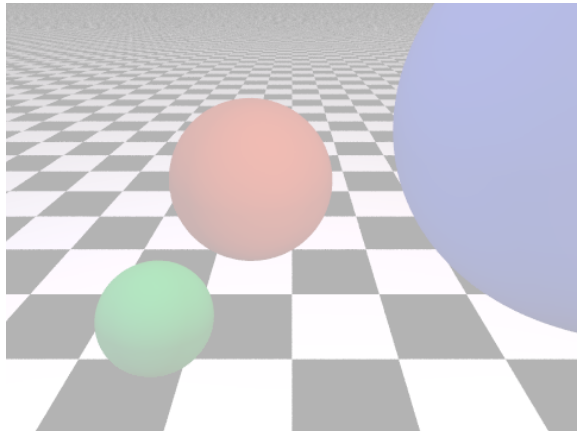
$$I = I_a m_a$$

- I_a: ambient light intensity in the scene
- m_a: material's ambient reflection coefficient

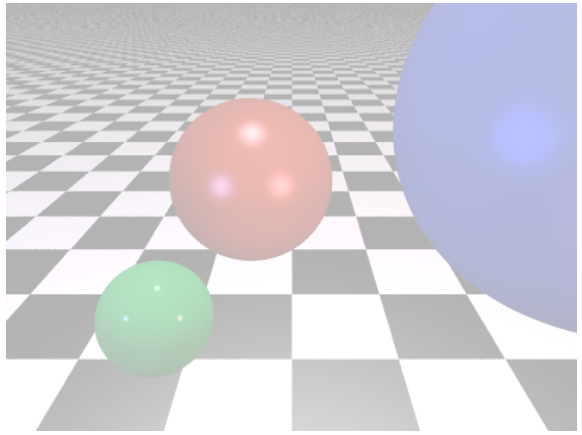
Lighting Computations



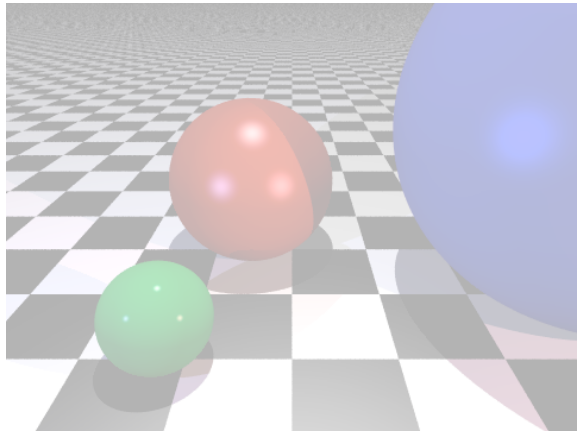
ambient



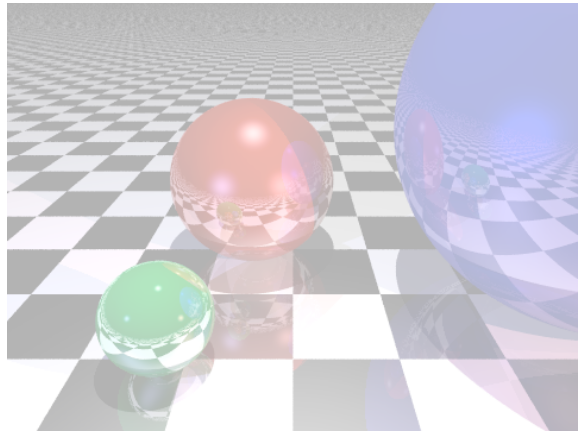
+diffuse



+specular

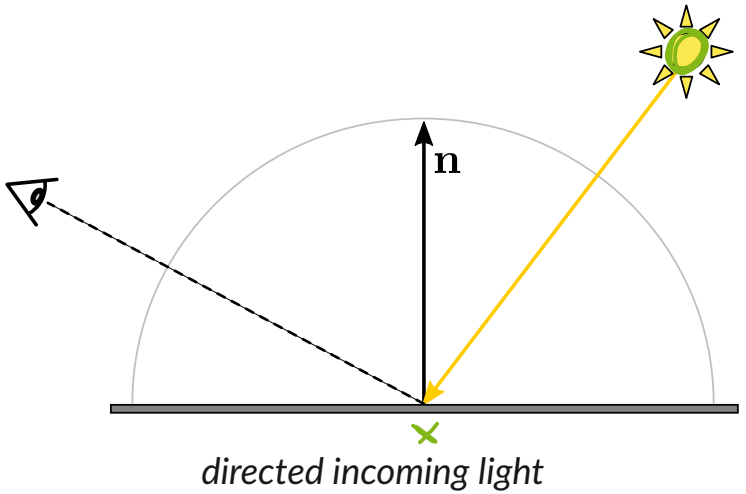


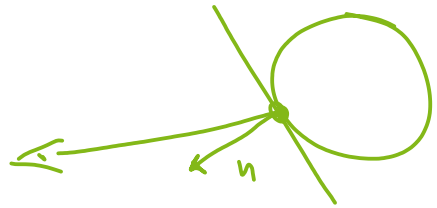
+shadows



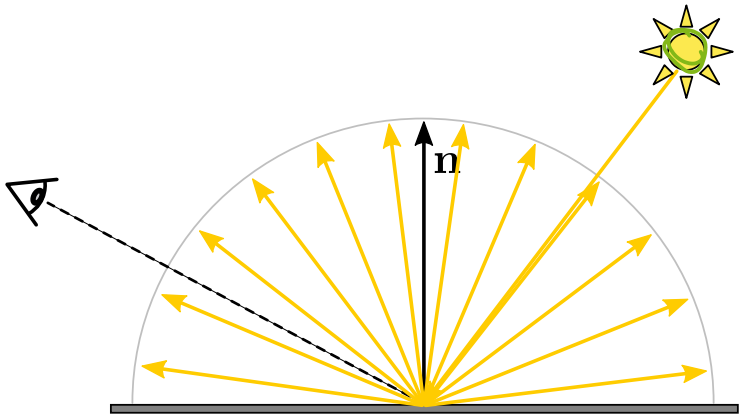
+reflections

Diffuse Reflection





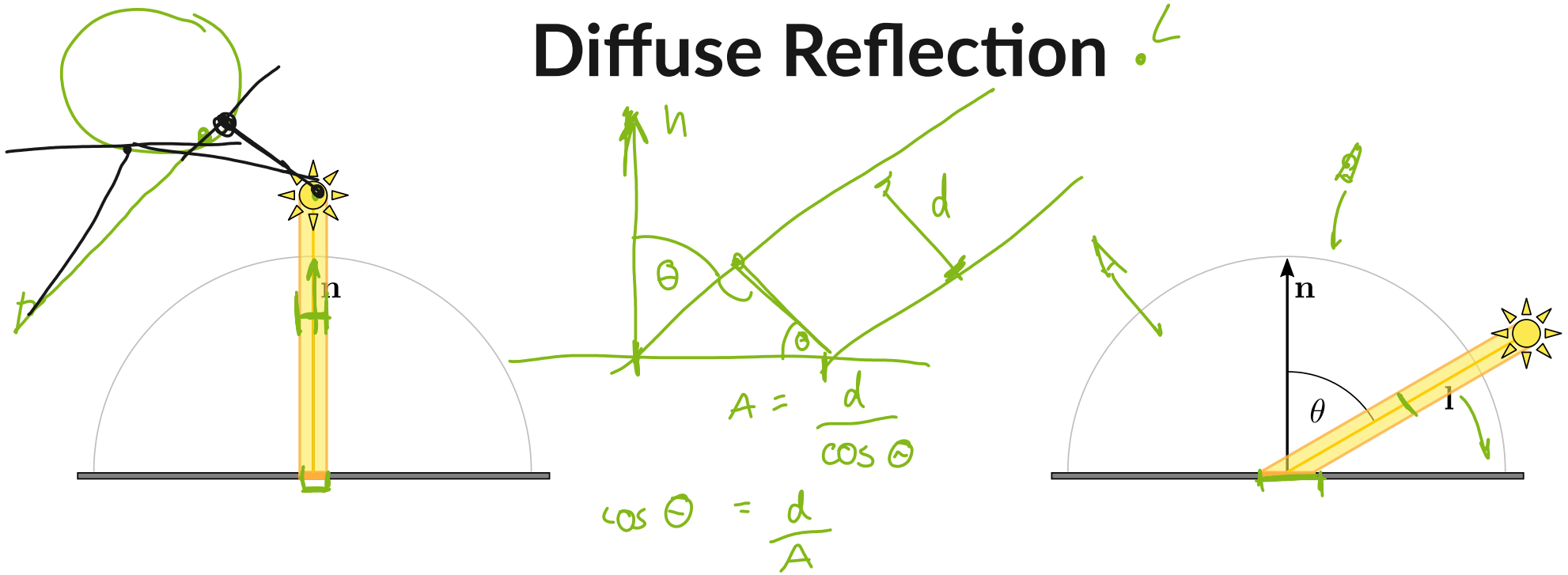
Diffuse Reflection



uniformly outgoing light

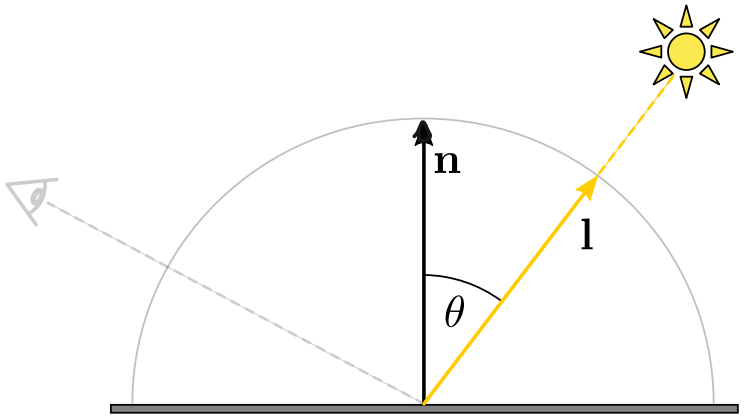
Brightness depends on how much light (density) comes in!

Diffuse Reflection



- Intensity inversely proportional to illuminated area
- Illuminated area inversely proportional to $\cos(\theta)$
- Therefore intensity proportional to $\cos(\theta)$
- So-called *Lambertian reflection*

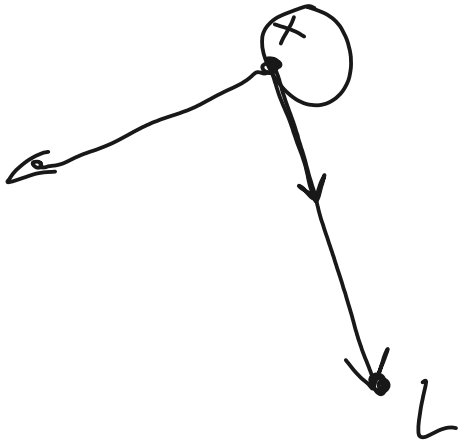
Diffuse Reflection



$$I = \underbrace{I_l m_d \cos\theta}$$

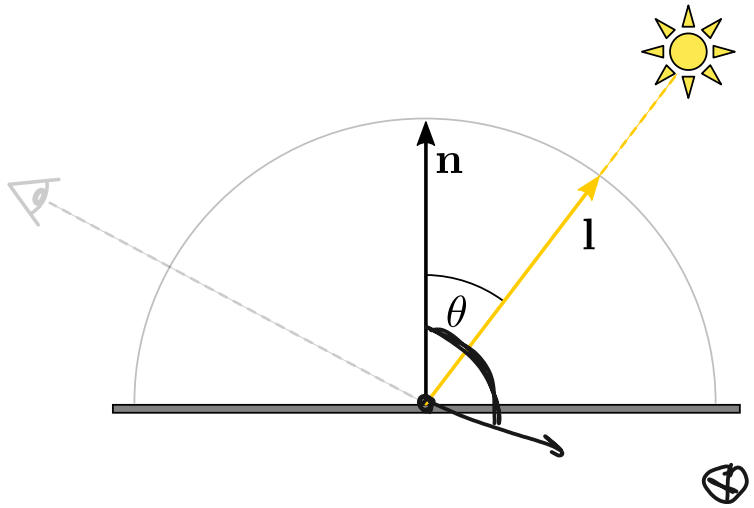
- I_l: intensity of light source l
- m_d: material's diffuse reflection coefficient

How can we compute $\cos\theta$ efficiently?



Diffuse Reflection

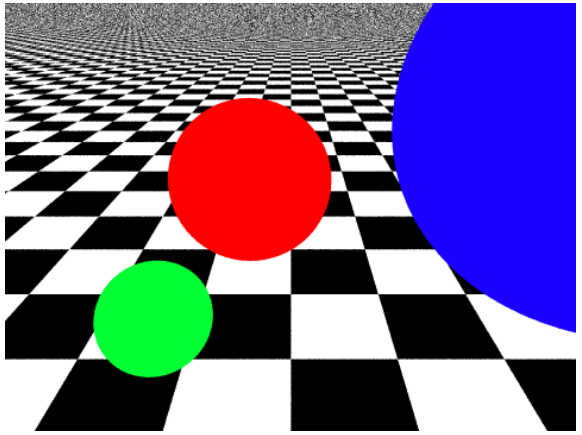
$$\cos\theta = \frac{n \cdot l}{\|n\| \, \|l\|}$$



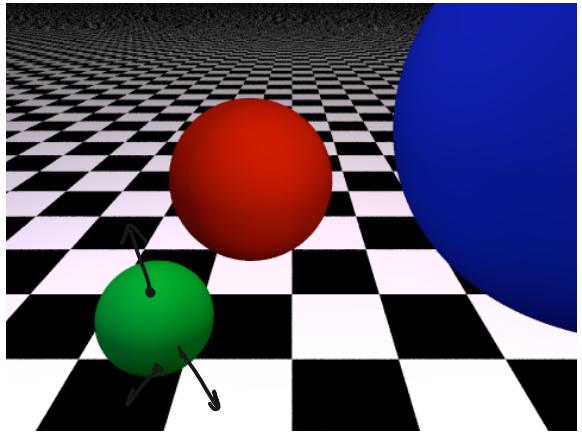
$$I = I_l \, m_d \cos\theta = I_l \, m_d \, (\mathbf{n} \cdot \mathbf{l})$$

- I_l: intensity of light source l
- m_d: material's diffuse reflection coefficient
- directions $\mathbf{n}$ and $\mathbf{l}$ assumed to be normalized
- no illumination if $\mathbf{n} \cdot \mathbf{l} < 0$ (why?)

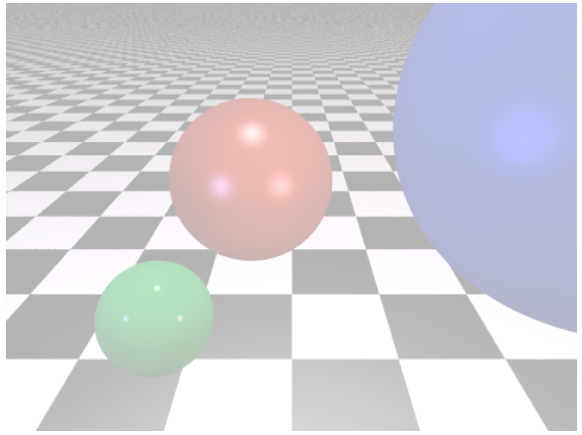
Lighting Computations



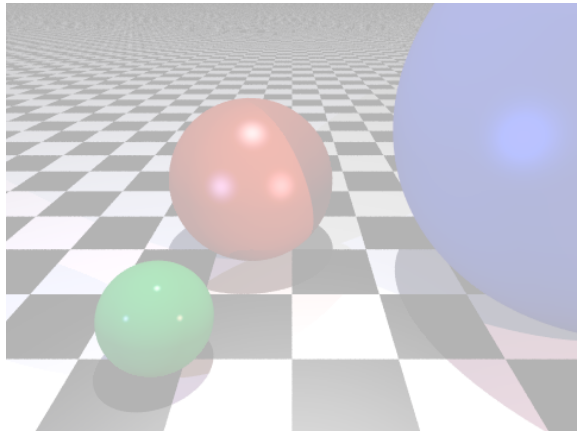
ambient



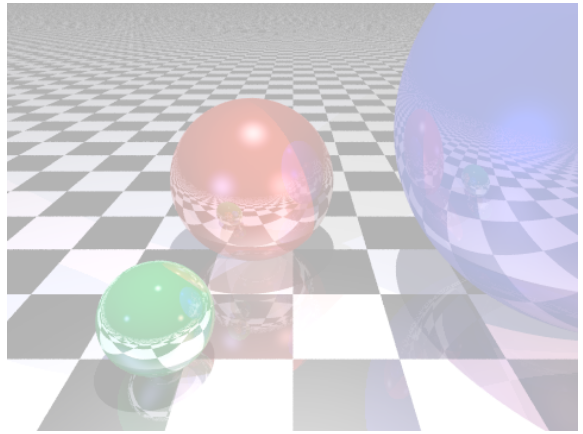
+diffuse



+specular

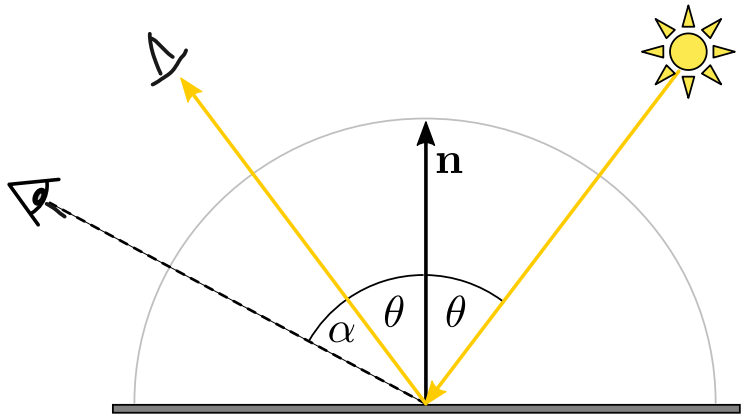


+shadows



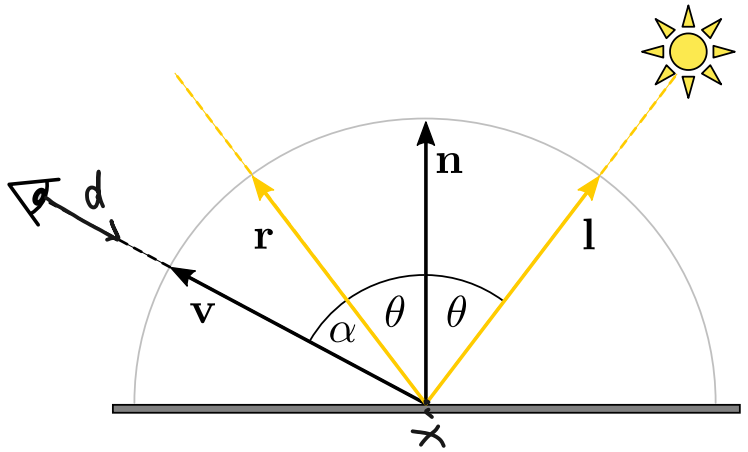
+reflections

Specular Reflection

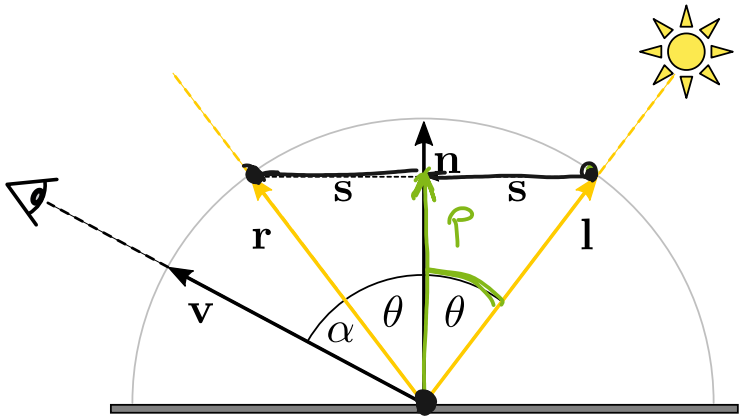


directed incoming, directed outgoing

Specular Reflection



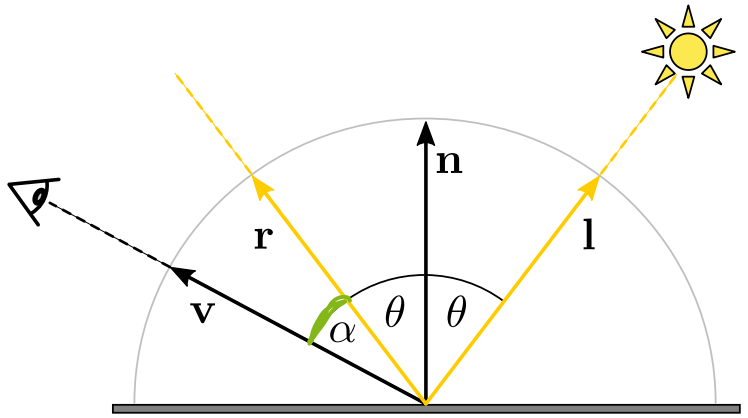
Specular Reflection



How to compute reflected ray $\mathbf{r}$?

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

Specular Reflection

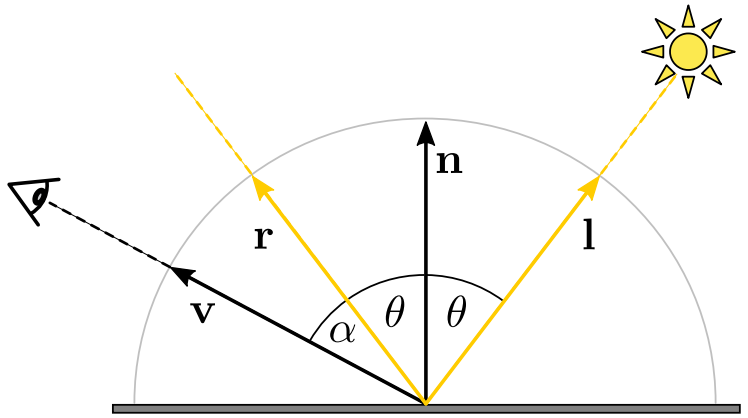


$$I = I_l m_s \cos(\alpha) = I_l m_s (\mathbf{r} \cdot \mathbf{v})$$

- I_l: intensity of light source l
- m_s: material's specular reflection coefficient
- all directions assumed to be normalized
- no illumination if $\mathbf{n} \cdot \mathbf{l} < 0$ or $\mathbf{r} \cdot \mathbf{v} < 0$

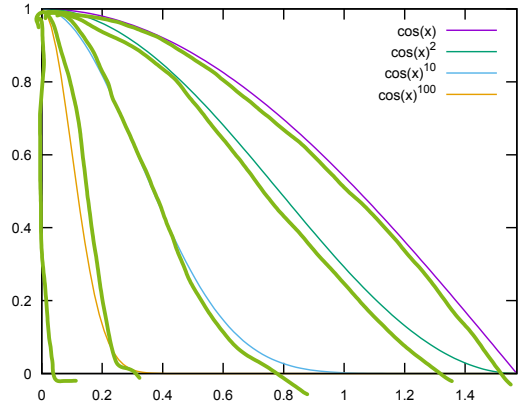
How can we control the surface's shininess?

Specular Reflection

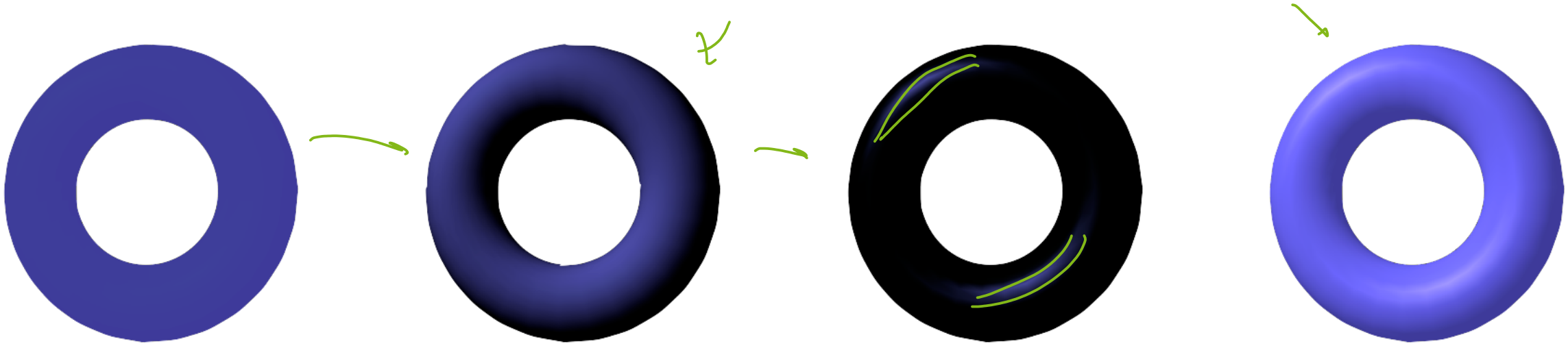


$$I = I_l m_s \cos^s(\alpha) = I_l m_s (\mathbf{r} \cdot \mathbf{v})^s$$

- I_l: intensity of light source l
- m_s: material's specular reflection coefficient
- all directions assumed to be normalized
- no illumination if $\mathbf{n} \cdot \mathbf{l} < 0$ or $\mathbf{r} \cdot \mathbf{v} < 0$
- s: cosine exponent controls shininess



Phong Lighting Model



ambient: $I_a m_a$

diffuse: $I_l m_d (\mathbf{n} \cdot \mathbf{l})$

specular: $I_l m_s (\mathbf{r} \cdot \mathbf{v})^s$

ambient+diffuse+specular

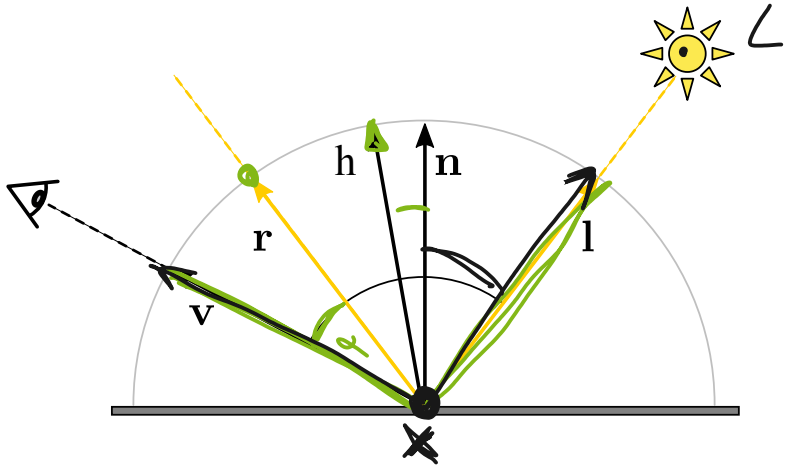
Blinn-Phong Model

- Simplification and improvement of Phong model
  - Half-way vector:

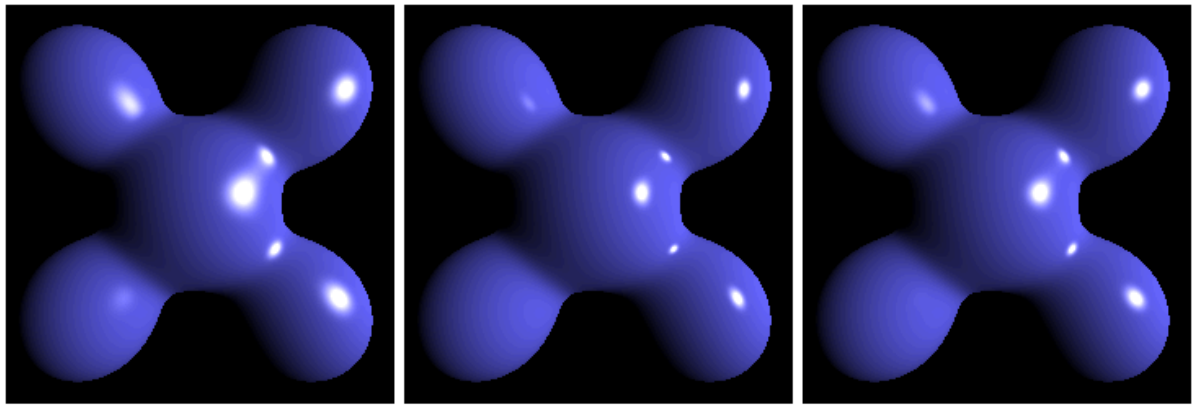
$$\mathbf{h} = \frac{\mathbf{l} + \mathbf{v}}{\|\mathbf{l} + \mathbf{v}\|}$$

  - Specular component

$$I = I_l m_s (\mathbf{n} \cdot \mathbf{h})^s$$

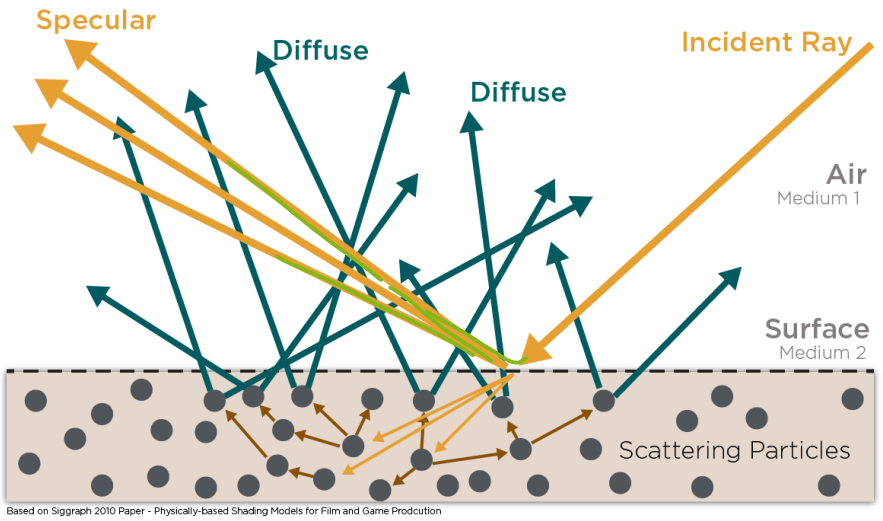


- Can be more efficient when viewer and light are far. Why?
- More info: Wikipedia



Advanced Models

- (Blinn-)Phong lighting is crude empirical model that does not capture physical light scattering accurately.



source

- Some reading for more advanced shading models: PBR Guide, Disney

Colors & Lighting

- Lighting depends on wavelength λ

$$I_\lambda \;=\; I_{a,\lambda} m_{a,\lambda} + I_{l,\lambda} \left(m_{d,\lambda} \left(\mathbf{n} \cdot \mathbf{l} \right) + m_{s,\lambda} (\mathbf{r} \cdot \mathbf{v})^s \right)$$

- We approximate it by RBG components

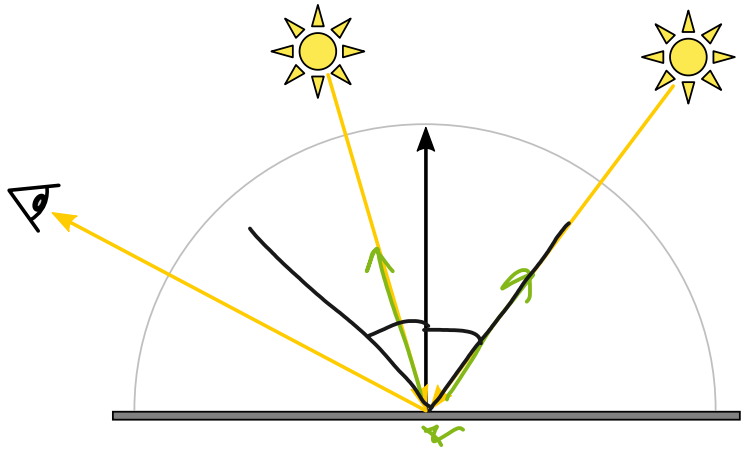
$$\begin{aligned}
I_R &= I_{a,R} m_{a,R} + I_{l,R} \left(m_{d,R} \left(\mathbf{n} \cdot \mathbf{l} \right) + m_{s,R} (\mathbf{r} \cdot \mathbf{v})^s \right) \\
I_G &= I_{a,G} m_{a,G} + I_{l,G} \left(m_{d,G} \left(\mathbf{n} \cdot \mathbf{l} \right) + m_{s,G} (\mathbf{r} \cdot \mathbf{v})^s \right) \\
I_B &= I_{a,B} m_{a,B} + I_{l,B} \left(m_{d,B} \left(\mathbf{n} \cdot \mathbf{l} \right) + m_{s,B} (\mathbf{r} \cdot \mathbf{v})^s \right)
\end{aligned}$$

- For RGB light colors/intensities $\mathbf{I}$ and RGB material colors/coefficients $\mathbf{m}$ and with component-wise product "$*$"

$$\mathbf{I} \;=\; \mathbf{I}_a * \mathbf{m}_a + \mathbf{I}_l * \left(\mathbf{m}_d \left(\mathbf{n} \cdot \mathbf{l} \right) + \mathbf{m}_s (\mathbf{r} \cdot \mathbf{v})^s \right)$$

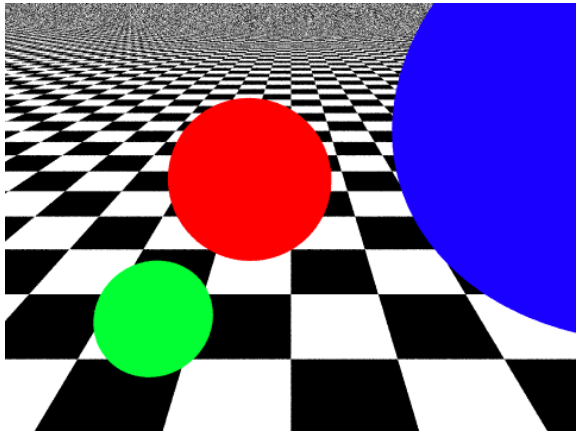
Multiple Light Sources

- We assumed linear superposition of light contributions and therefore can simply sum over all light sources

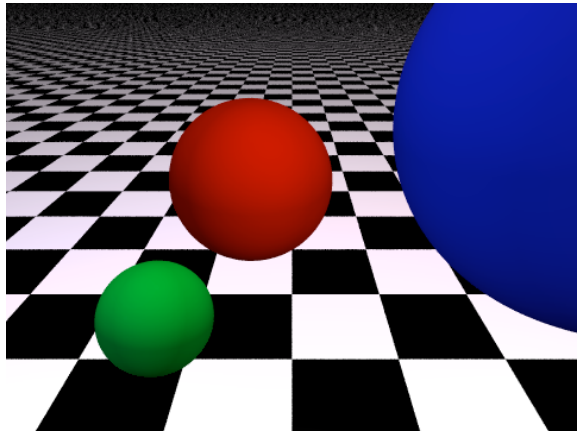


$$I = I_a m_a + \sum_l I_l \left(m_d (\mathbf{n} \cdot \mathbf{l}_l) + m_s (\mathbf{r}_l \cdot \mathbf{v})^s \right)$$

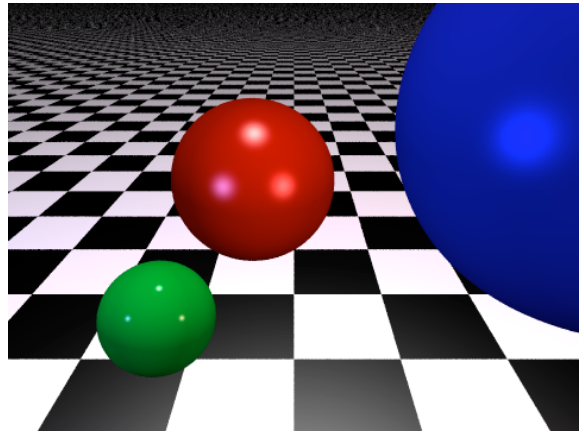
Lighting Computations



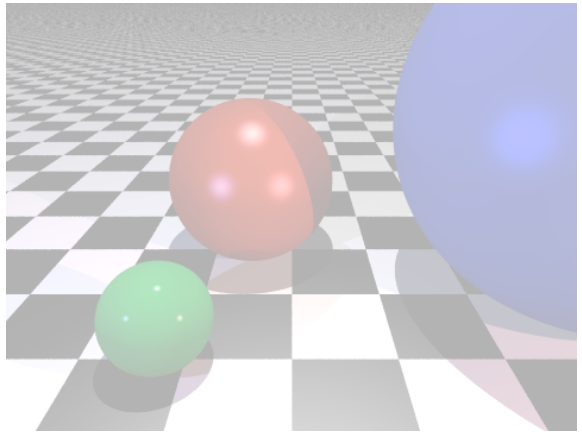
ambient



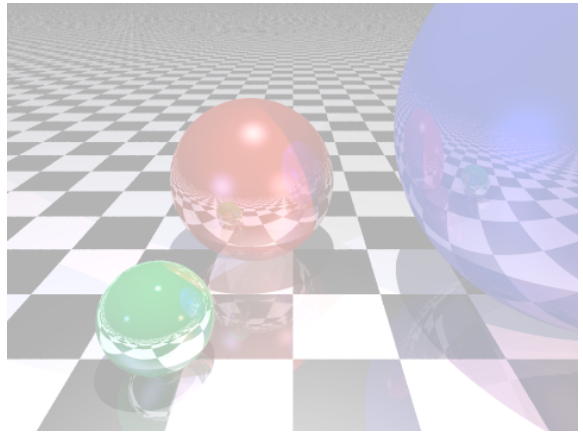
+diffuse



+specular



+shadows



+reflections

Try it yourself!