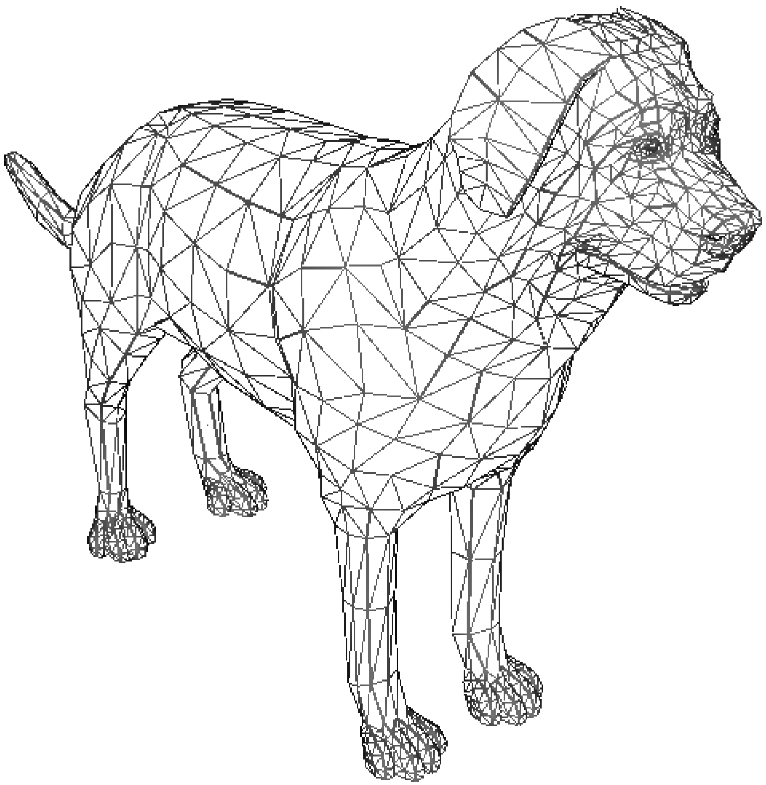
Ray-Tracing

Misha Kazhdan

Ray-Tracing

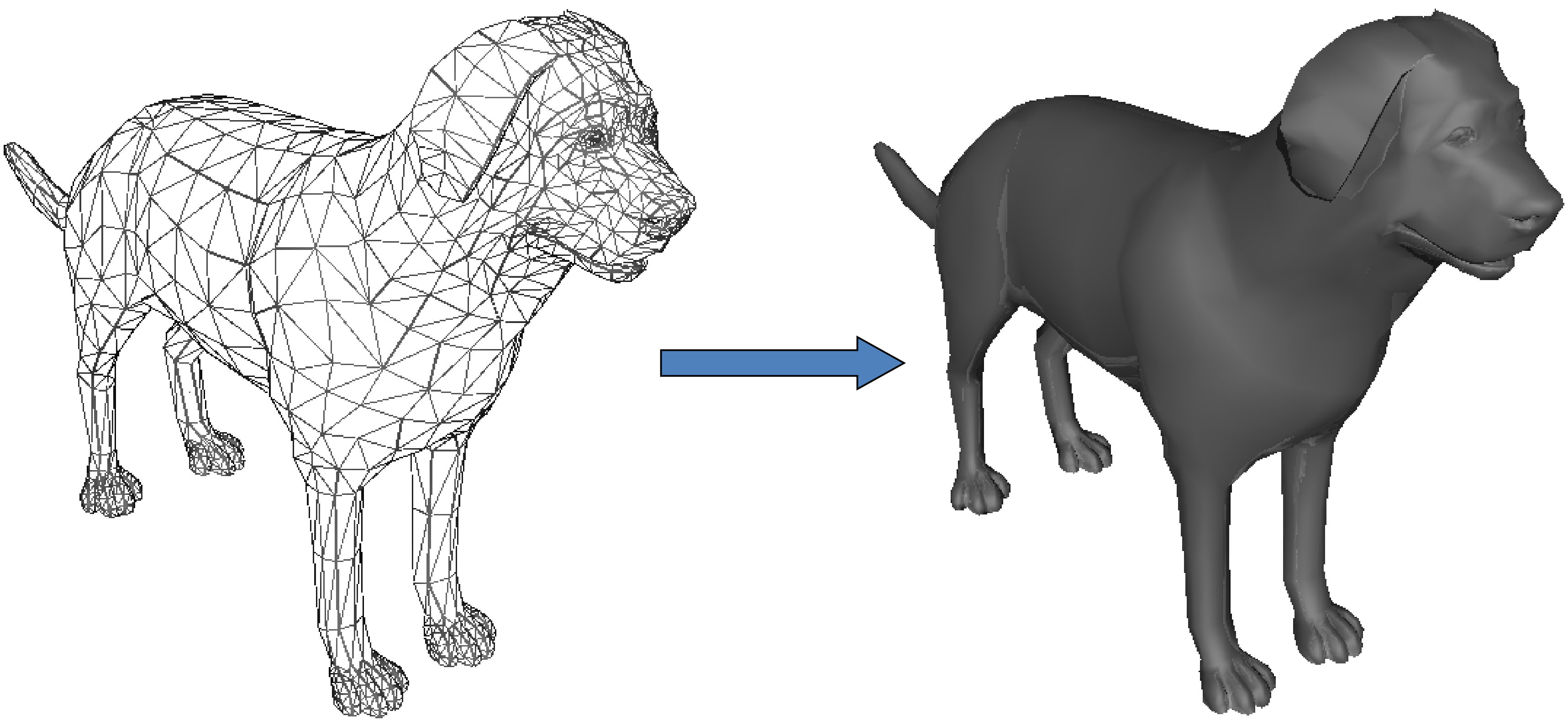
In graphics, we often represent the surface of a 3D shape by a set of triangles.



Ray-Tracing

Goal:

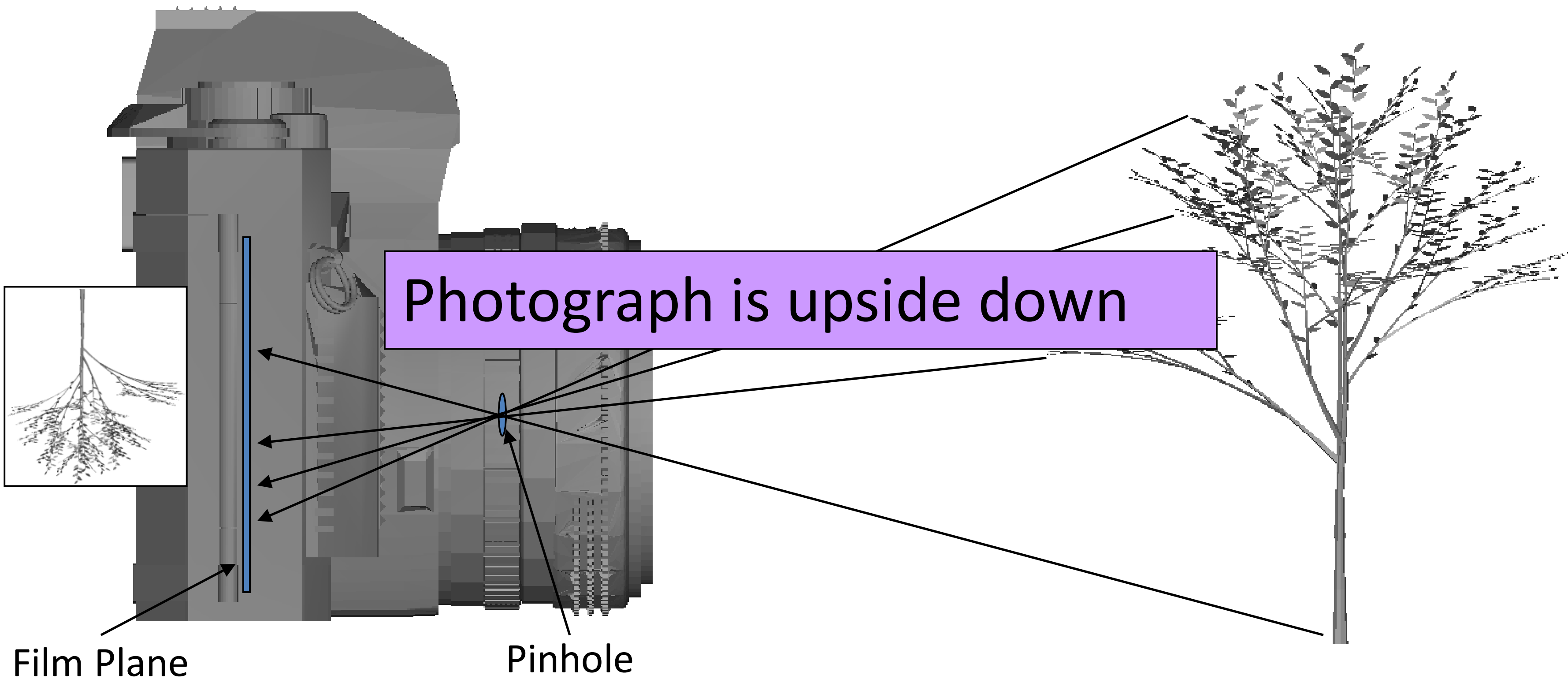
Take a collection of triangles representing a 3D scene and render a detailed image.



Physical Pinhole Camera

The film sits behind the pinhole of the camera.

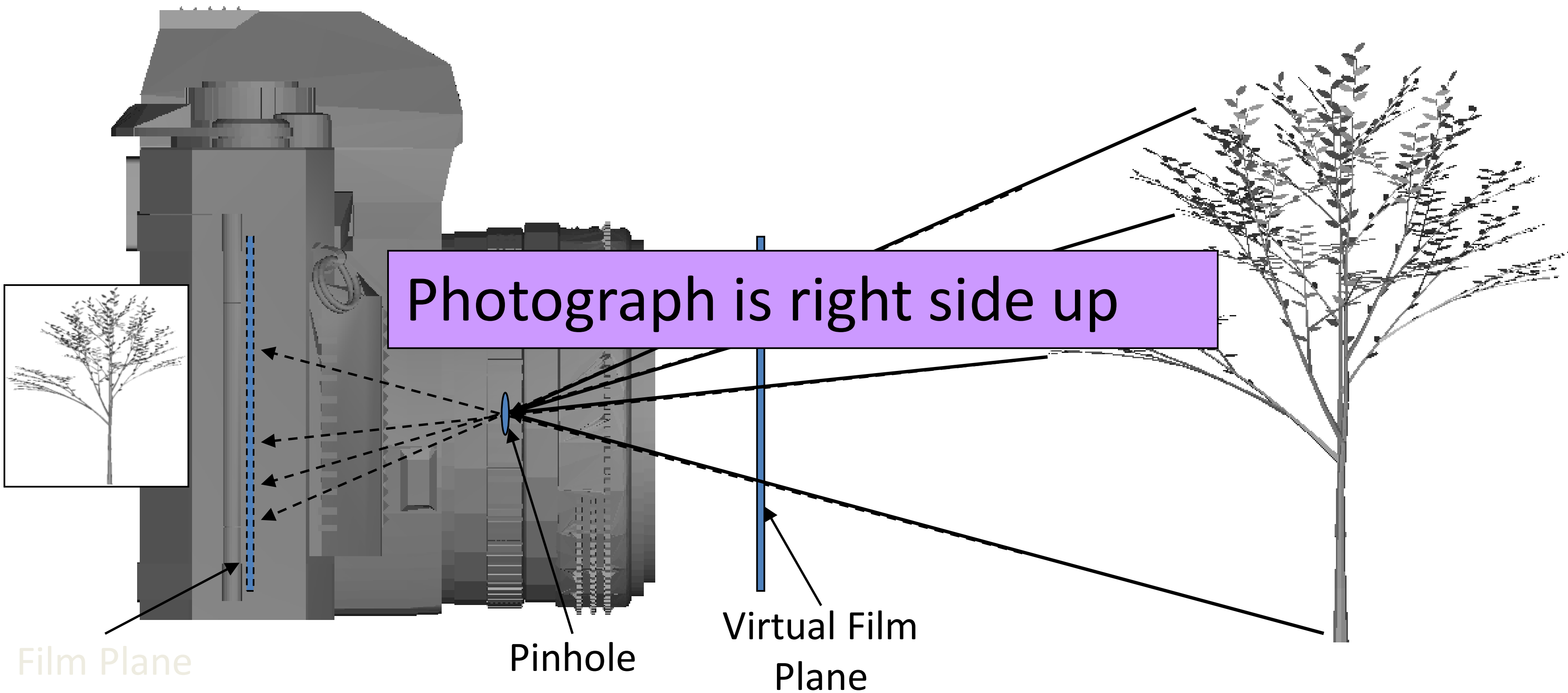
Rays come in from the outside, pass through the pinhole, and hit the film plane.



Virtual Camera

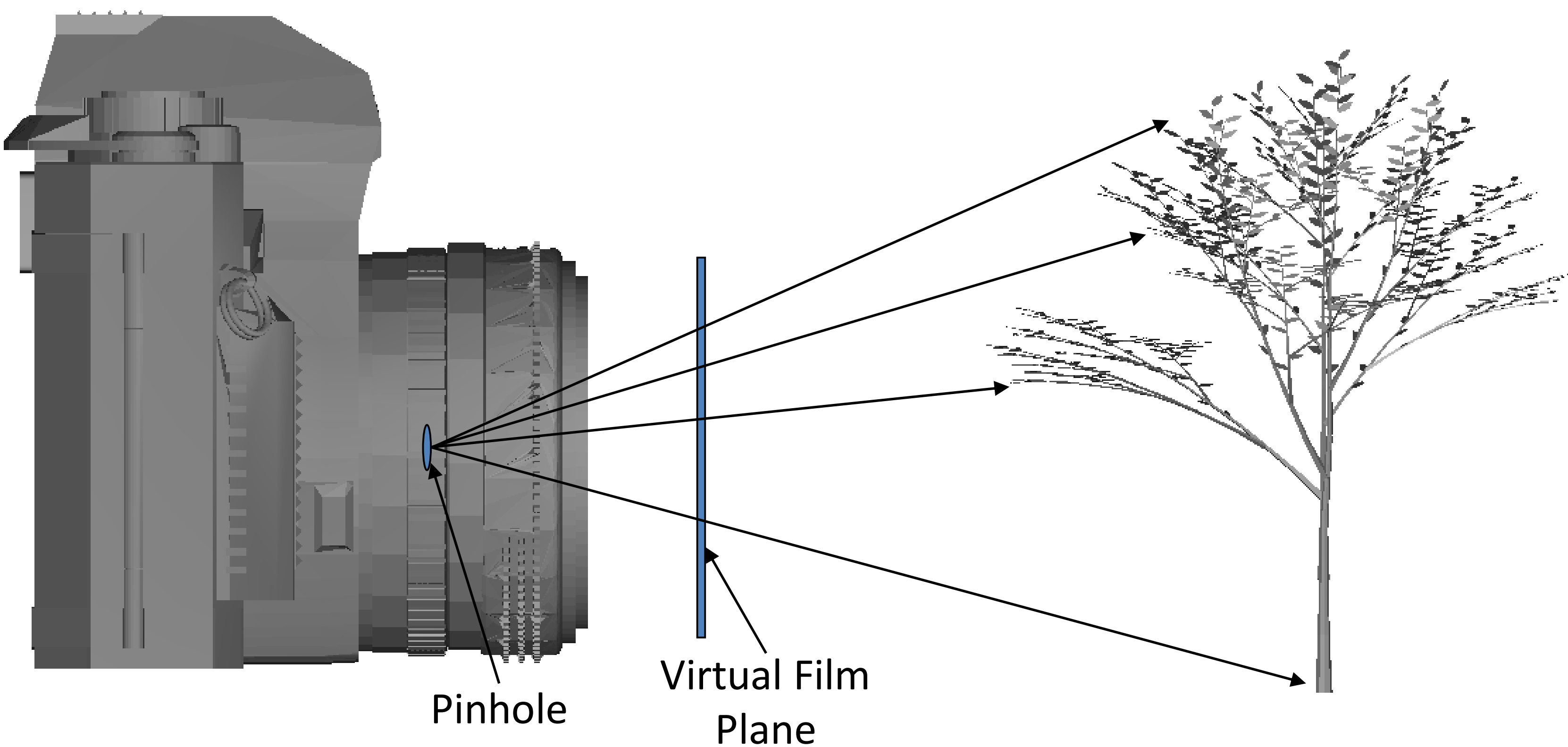
The film sits in front of the pinhole of the camera.

Rays come in from the outside, pass through the film plane, and hit the pinhole.



Ray-Casting

We invert the process of image generation by sending rays out from the pinhole.

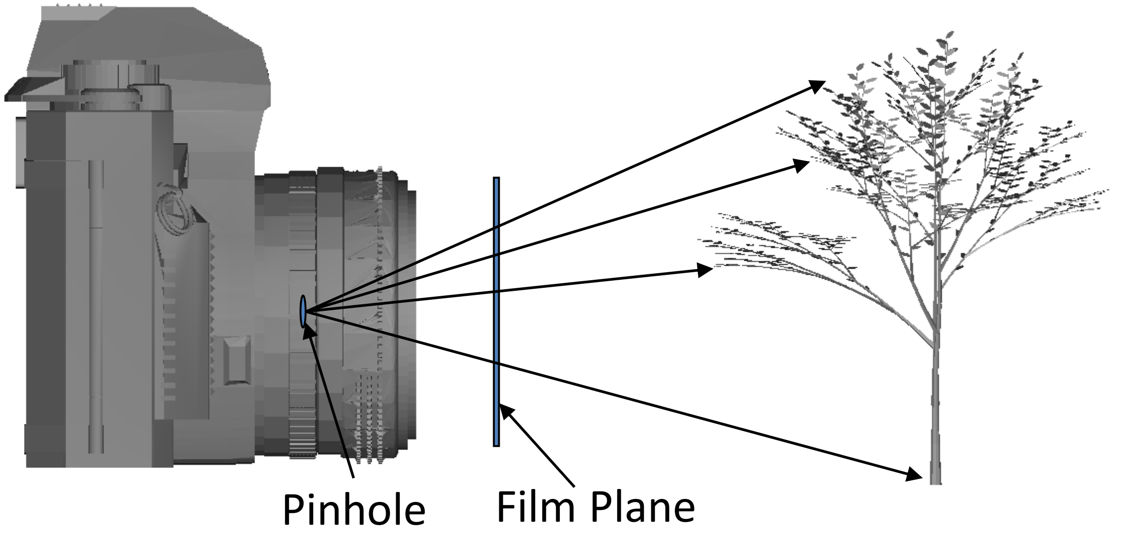


Ray-Casting

We invert the process of image generation by sending rays out from the pinhole.

For each pixel in the virtual file plane, we:

- Compute the ray: pinhole → pixel
- Figure out what object in the scene is hit first
- Set the pixel to the color of the object

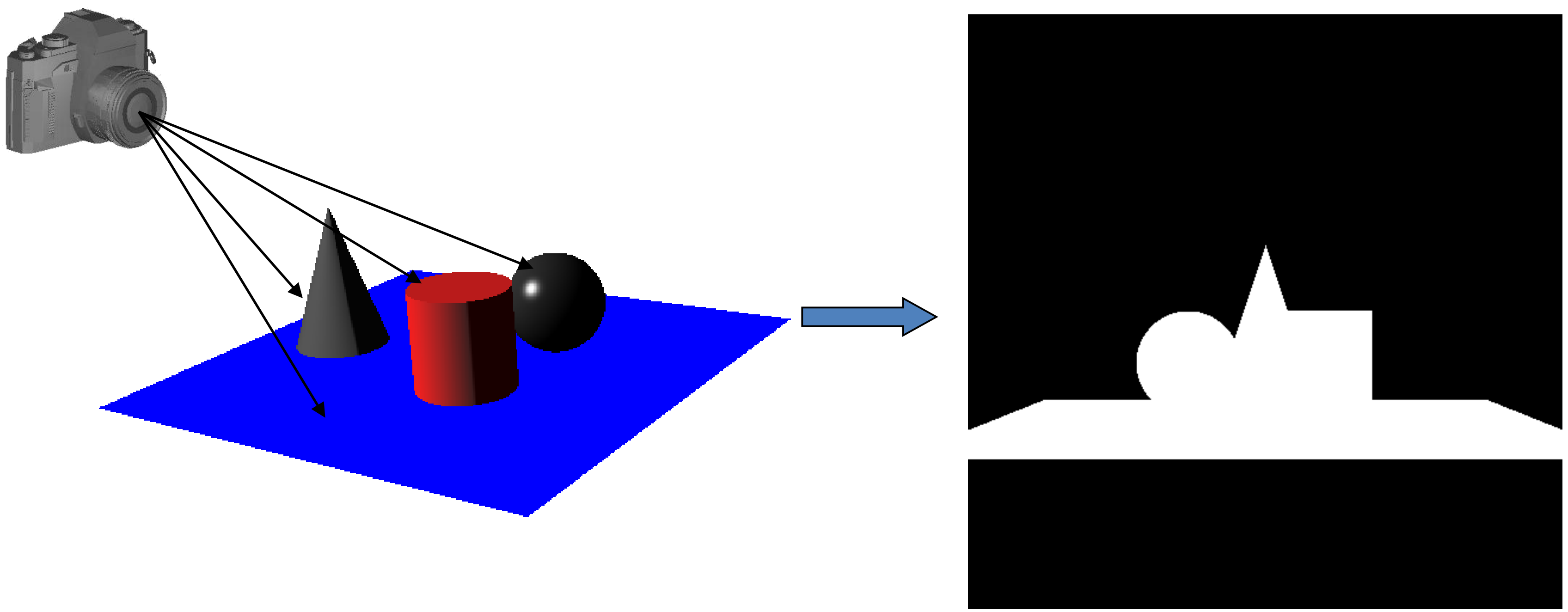


Ray-Casting

```
Image RayCast( Camera camera , Scene scene , int width , int height )
{
	Image image = new Image( width , height );
	for( int i=0 ; i<width ; i++ ) for ( int j=0 ; j<height ; j++ )
	{
		Ray ray = ConstructRayThroughPixel( camera , i , j );
		Intersection hit = FindIntersection( ray , scene );
		image[i][j] = GetColor( hit );
	}
	return image;
}
```

Ray-Casting

If we ignore the color computation, we get the silhouettes of the scene:



Outline

Ray-Tracing

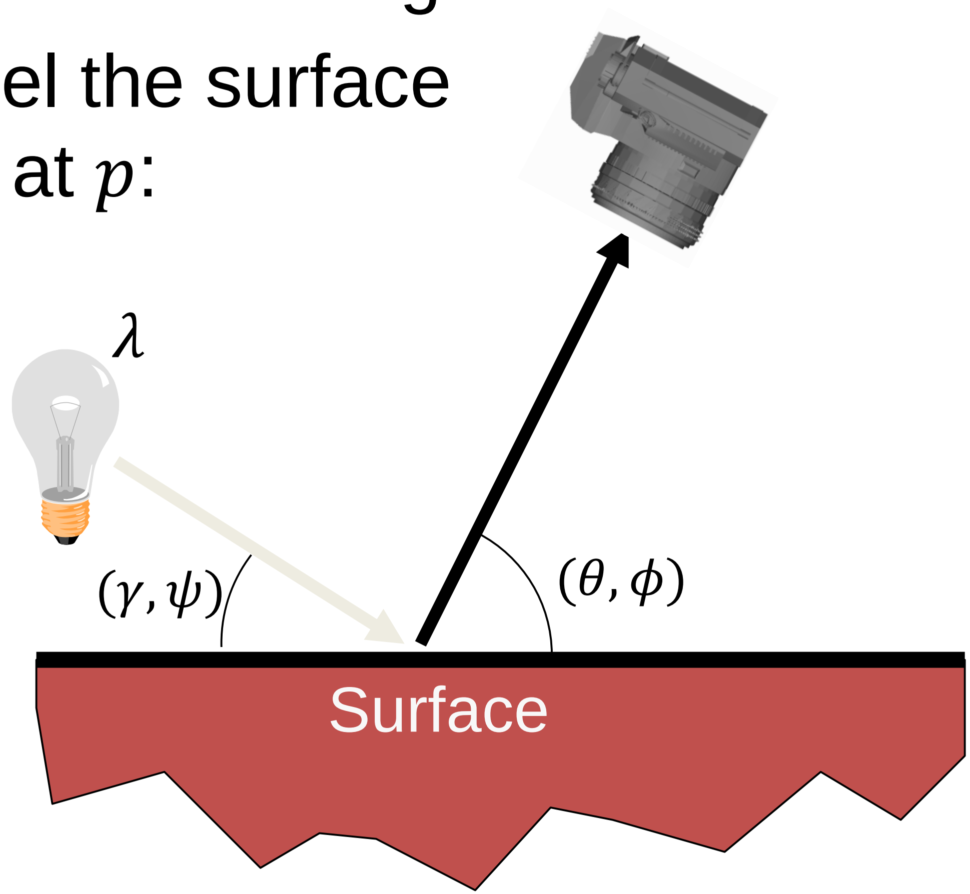
- Overview
- Direct Illumination
- Global Illumination

Modeling Surface Reflectance

Surface color is determined by the lights and the way different surfaces reflect the light.

Ideally, we would model the surface reflectance properties at p:

$$R_p(\theta, \phi, \lambda, \gamma, \psi)$$



R_p is the fraction of incident light:

- arriving from direction (γ, ψ)
- with wavelength λ
- leaving in direction (θ, ϕ)

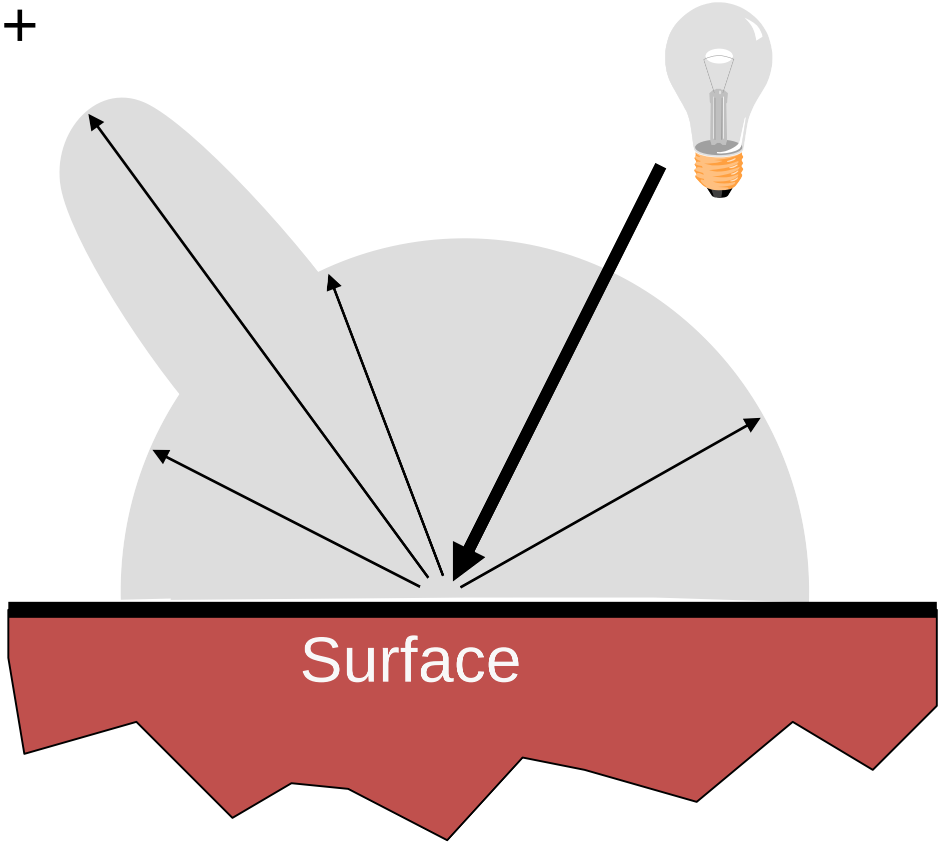
- Too much storage
- Difficult in practice

Simple Reflectance Model

Simple model:

- diffuse reflection +
- specular reflection +
- emission +
- “ambient”

Based on model proposed by Phong

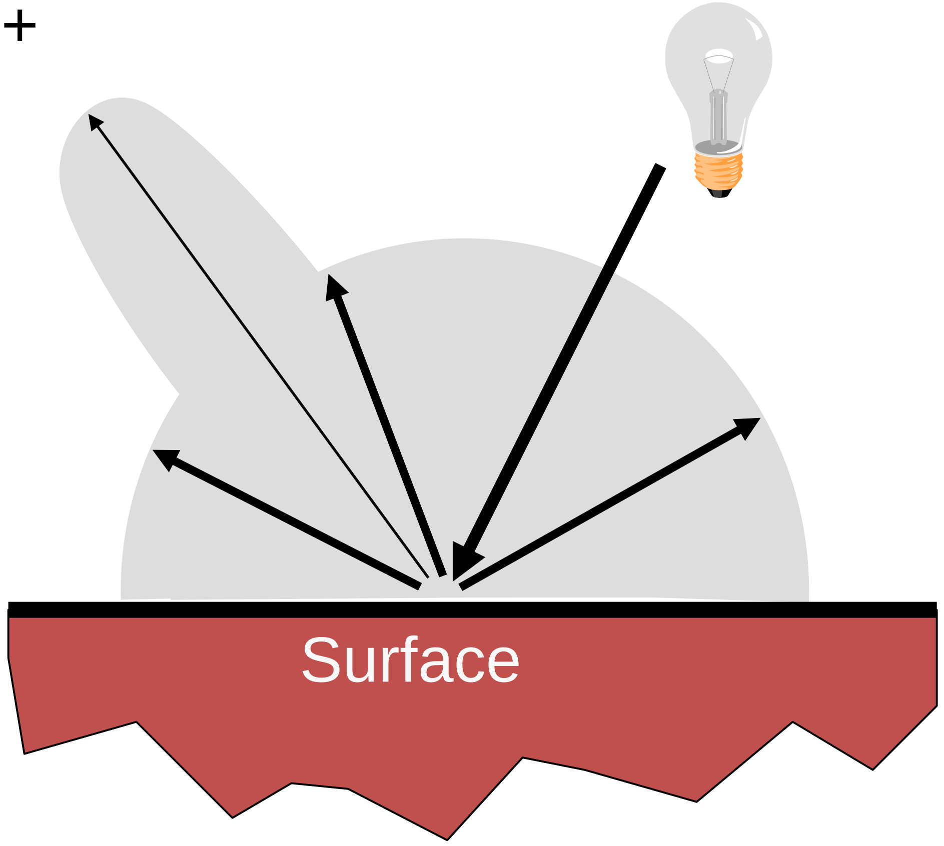


Simple Reflectance Model

Simple model:

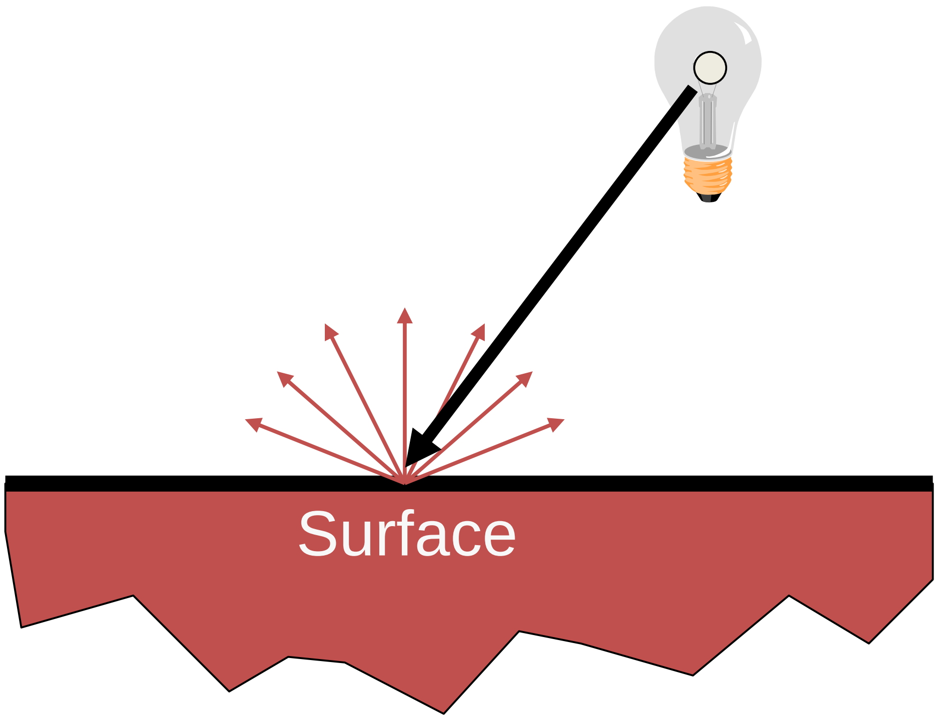
- diffuse reflection +
- specular reflection +
- emission +
- “ambient”

Based on model proposed by Phong



Diffuse Reflection

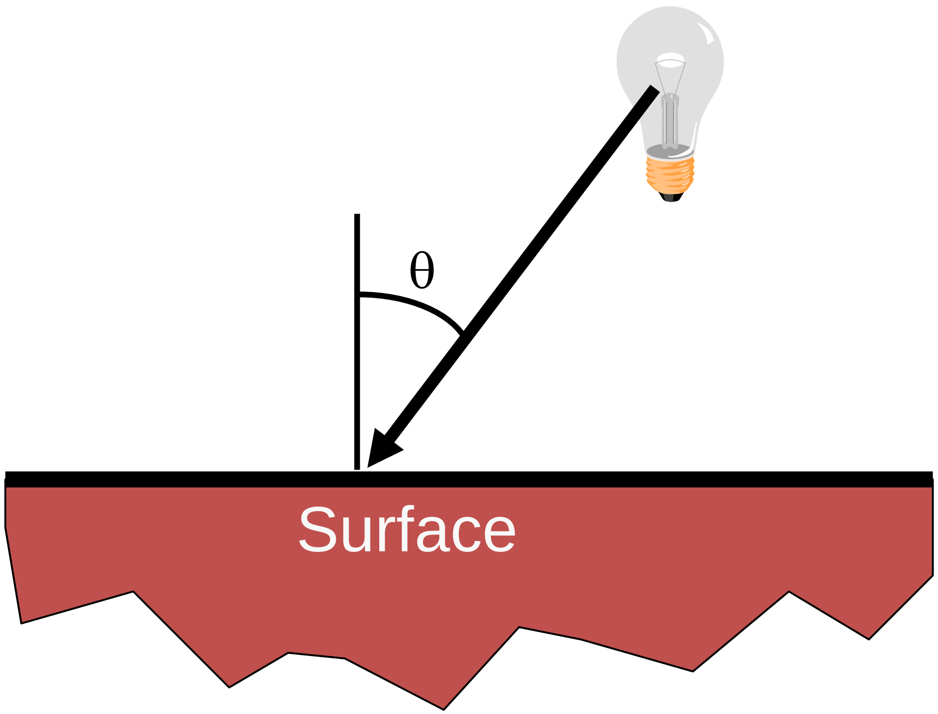
- Assume surface reflects equally in all directions
  - Examples: chalk, clay



Diffuse Reflection

How much light is reflected?

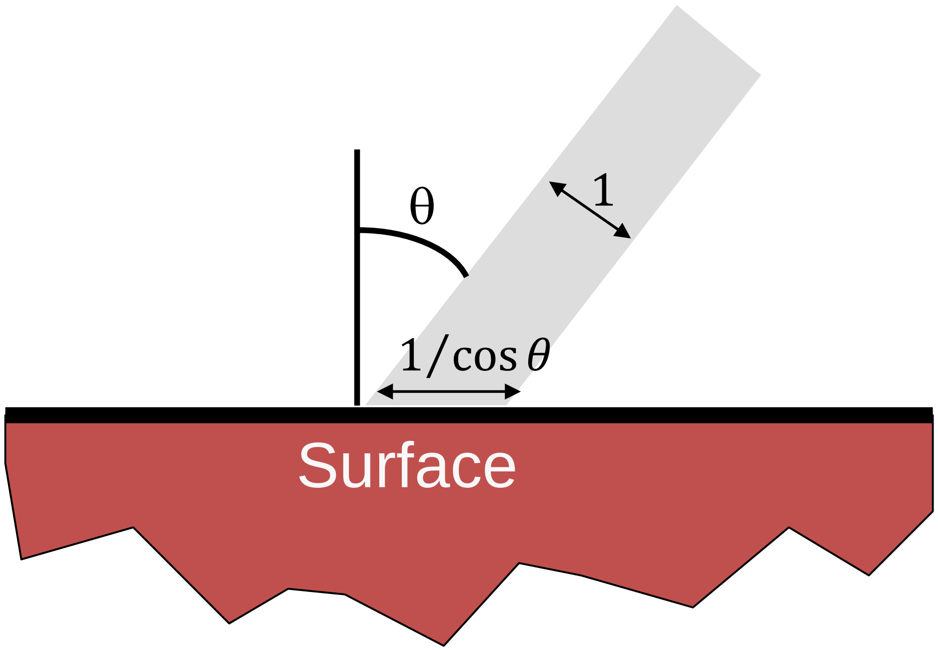
– Depends on angle of incident light



Diffuse Reflection

How much light is reflected?

– Depends on angle of incident light



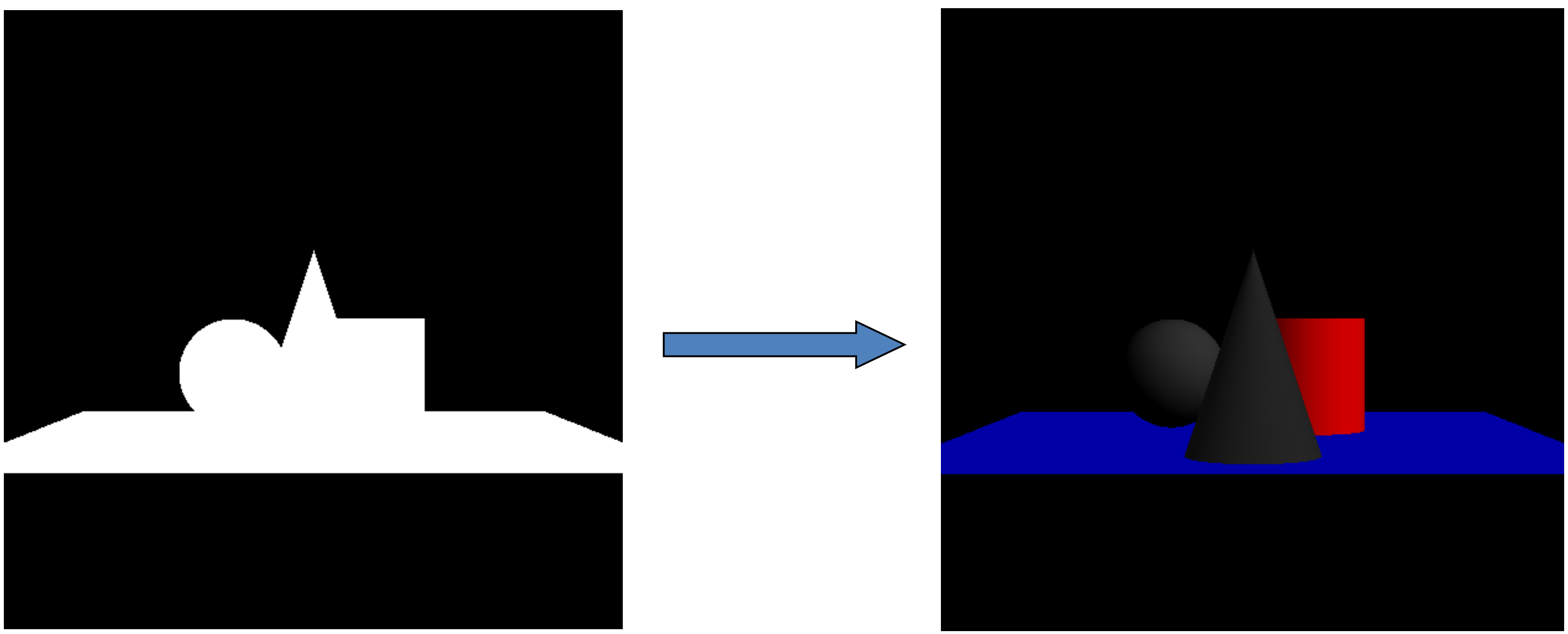
Think of a flashlight!

Physically motivated:

(Surface color) = (Light color) * $\cos\theta$ * (Diffuse)

Diffuse Reflection

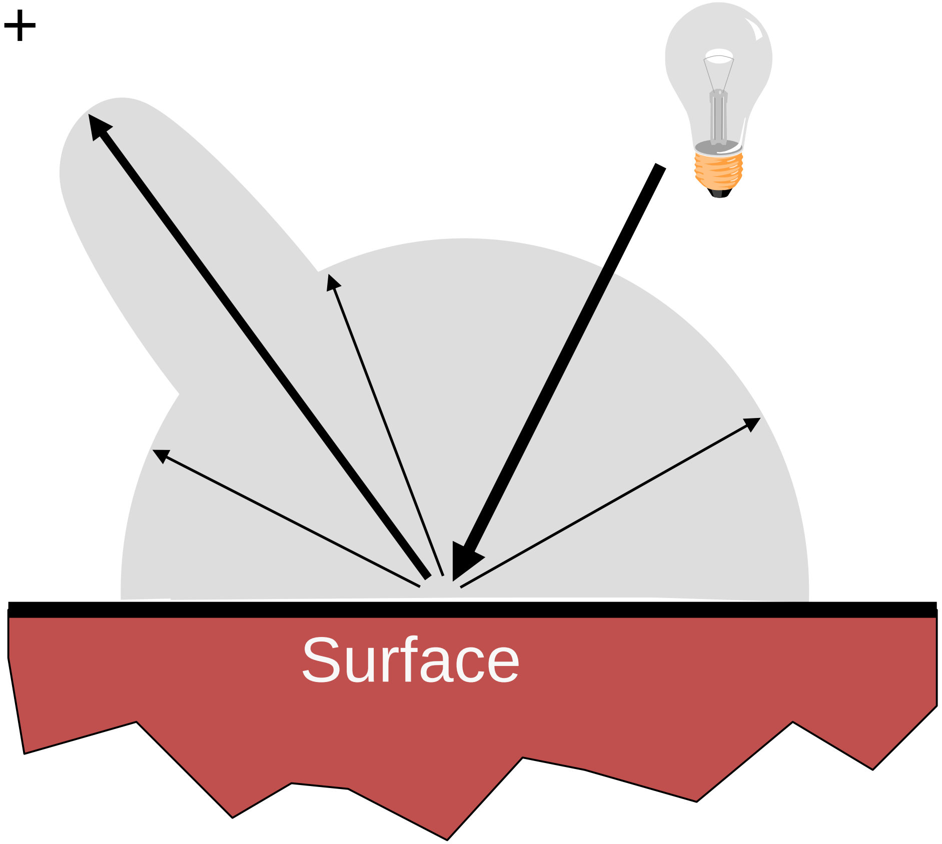
- Assume surface reflects equally in all directions
  - Examples: chalk, clay



Simple Reflectance Model

Simple analytic model:

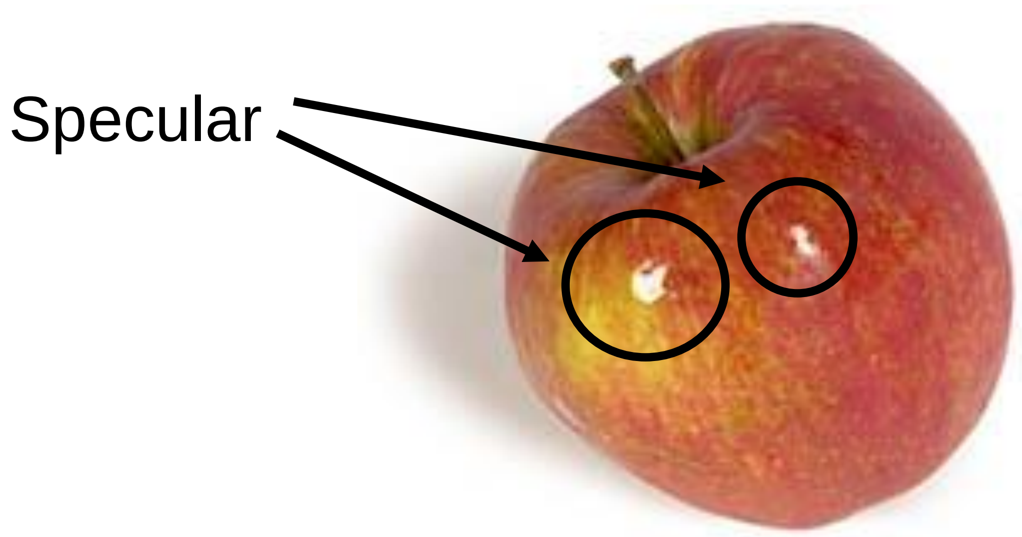
- diffuse reflection +
- specular reflection +
- emission +
- "ambient"



Specular Reflection

Reflection is strongest near mirror angle

- Examples: metals, shiny apples

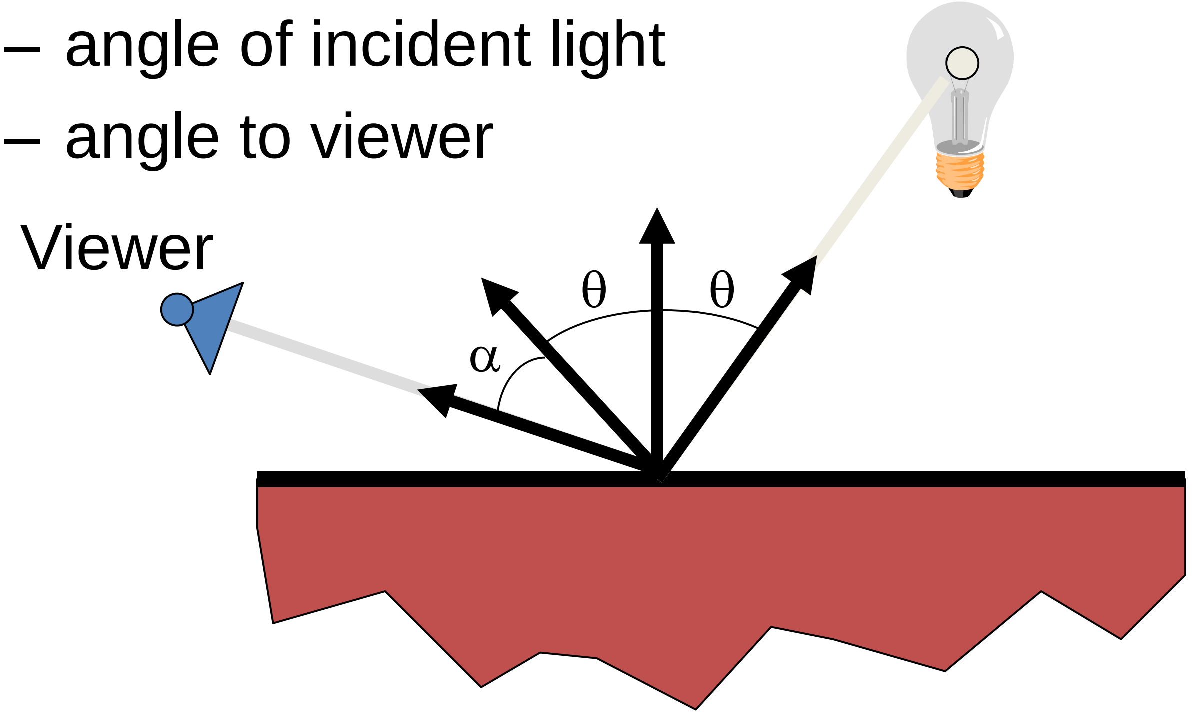


Specular Reflection

How much light is seen?

Depends on:

- angle of incident light
- angle to viewer



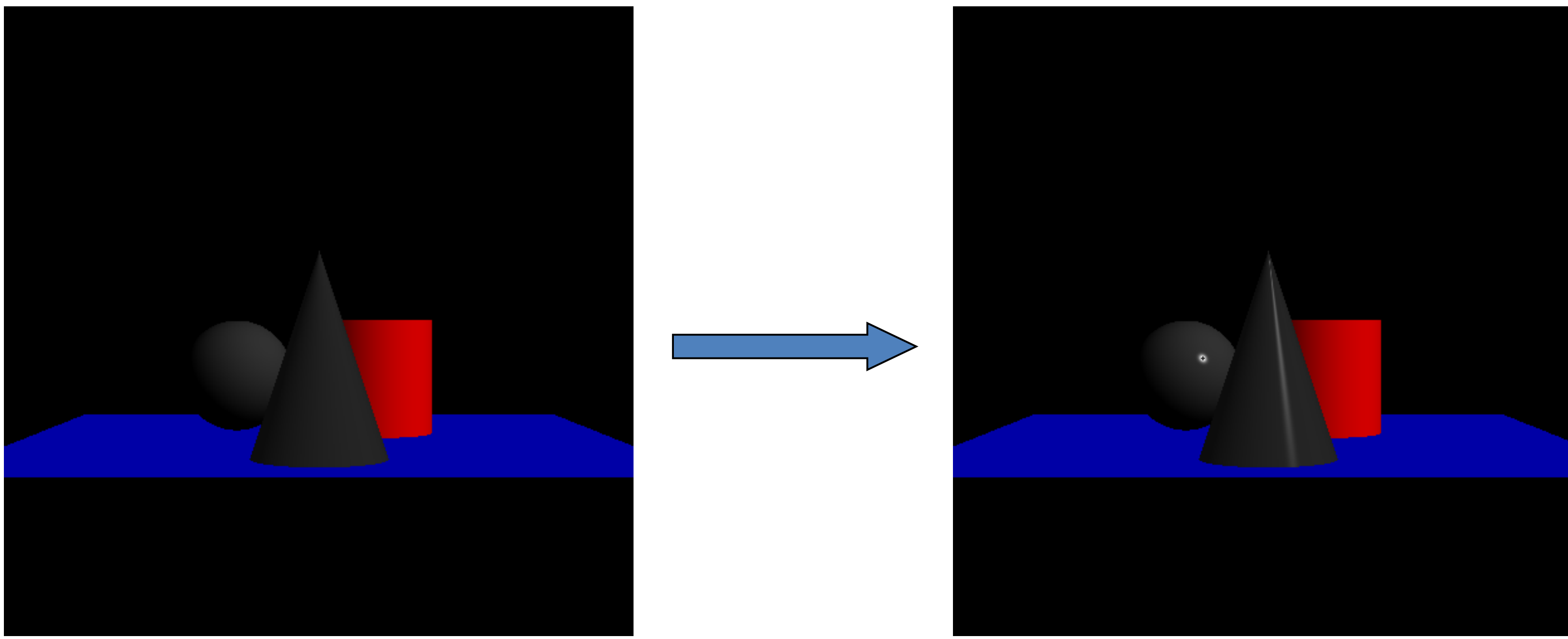
Works well in practice:

(Surface color) = (Light color) * $\cos^k \alpha$ * (Specular)

Specular Reflection

Reflection is strongest near mirror angle

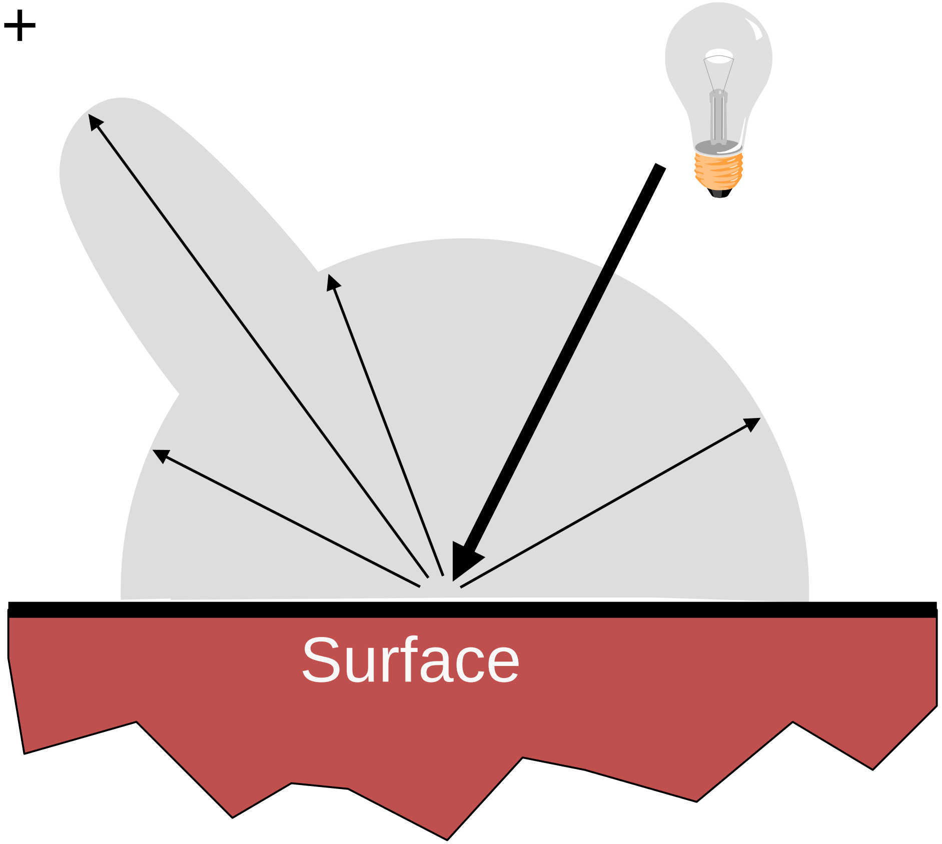
- Examples: metals, shiny apples



Simple Reflectance Model

Simple analytic model:

- diffuse reflection +
- specular reflection +
- emission +
- “ambient”



Emission

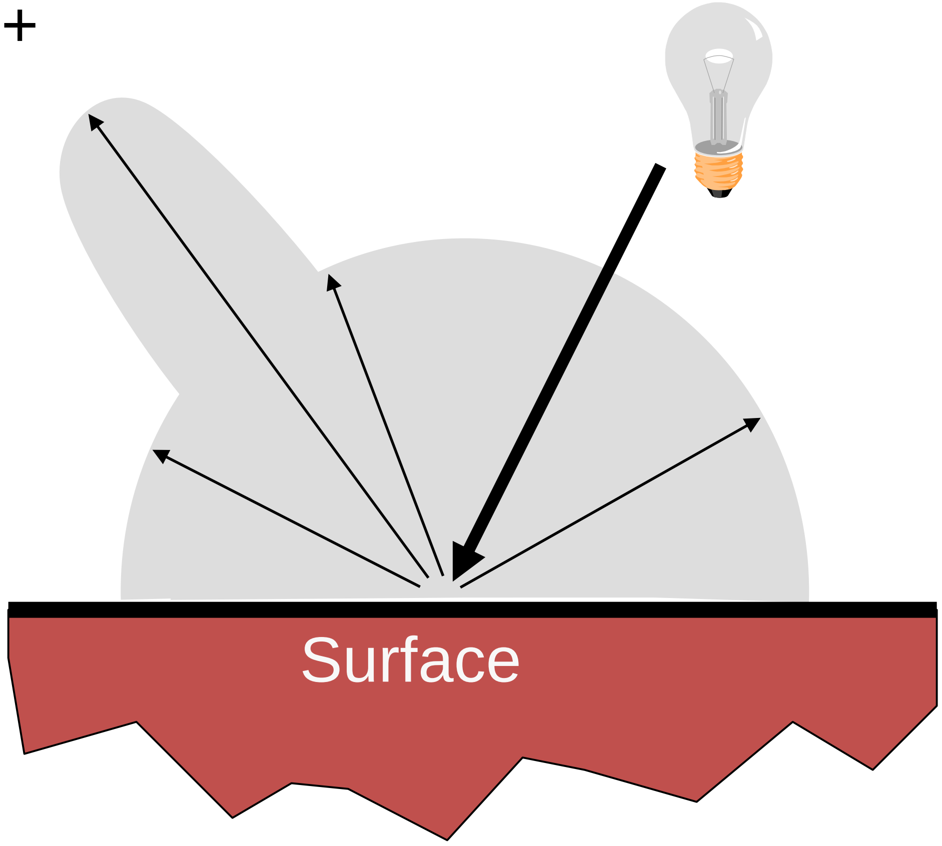
Represents light emanating directly from polygon

Emission ≠ 0

Simple Reflectance Model

Simple analytic model:

- diffuse reflection +
- specular reflection +
- emission +
- “ambient”



Ambient Term

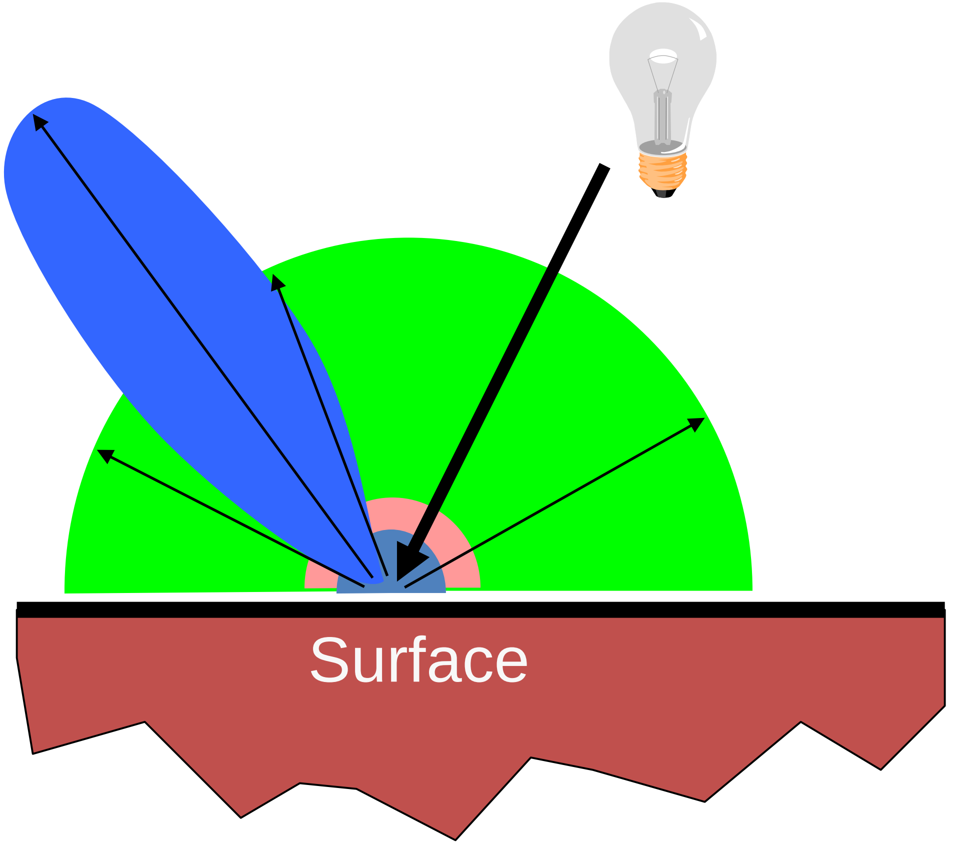
Represents accumulation of indirect illumination

Locations that are not directly illuminated are still not black because of indirect illumination.

Simple Reflectance Model

Simple analytic model:

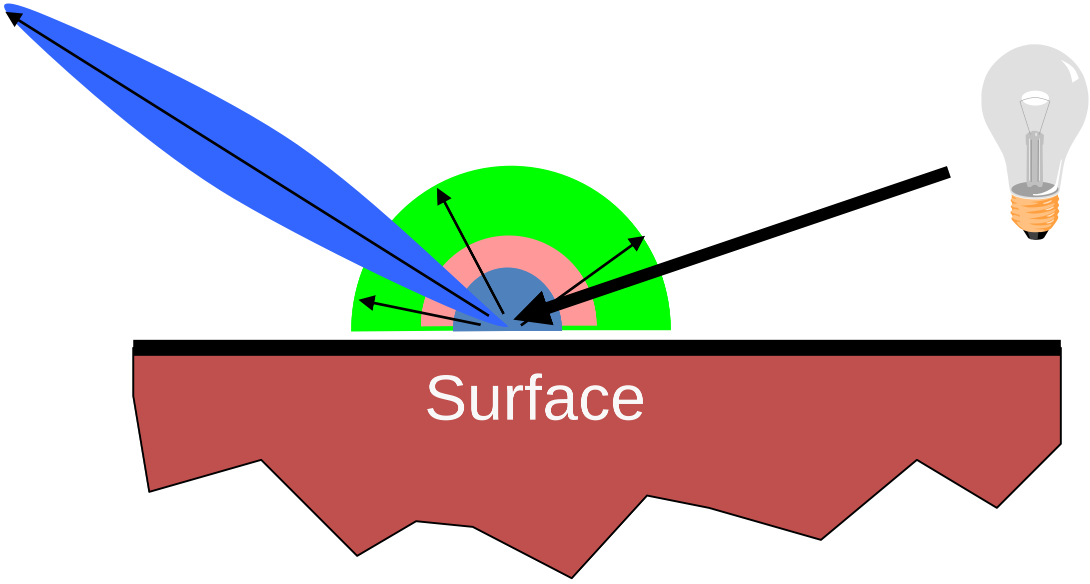
- diffuse reflection +
- specular reflection +
- emission +
- “ambient”



Simple Reflectance Model

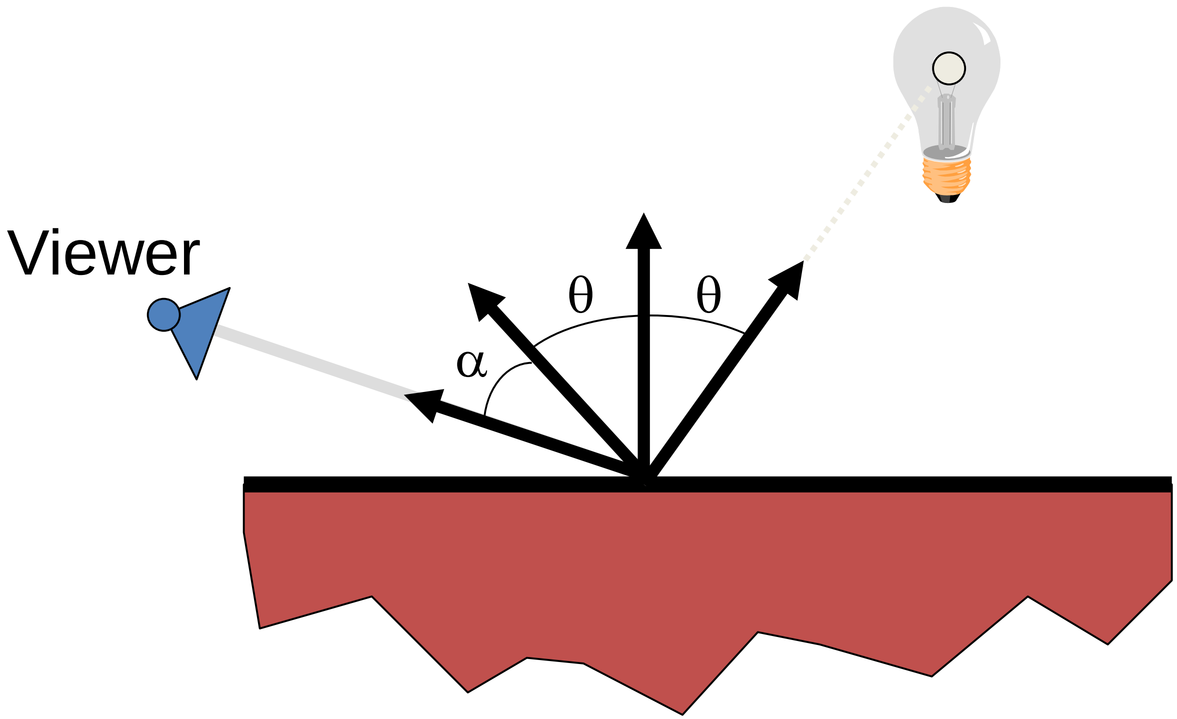
Simple analytic model:

- diffuse reflection +
- specular reflection +
- emission +
- “ambient”



Surface Illumination Calculation

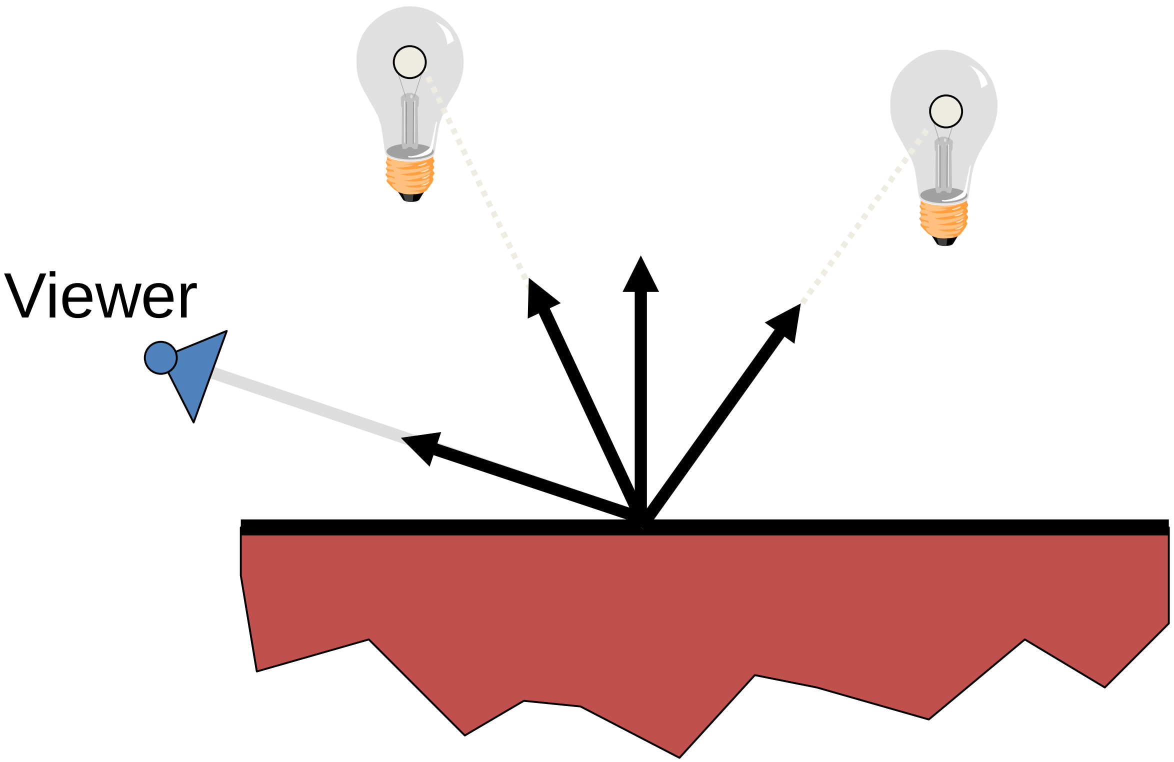
Single light source:



$$I = K_E + K_A + \left(\cos\theta \cdot K_D + \cos^k \alpha \cdot K_S\right) \cdot I_L$$

Surface Illumination Calculation

Multiple light sources:



$$I = K_E + K_A + \sum_{L \in Lights} \left(\cos\theta_L \cdot K_D + \cos^k \alpha_L \cdot K_S\right) \cdot I_L$$

Outline

Ray-Tracing

- Overview
- Direct Illumination
- Global Illumination

Shadows

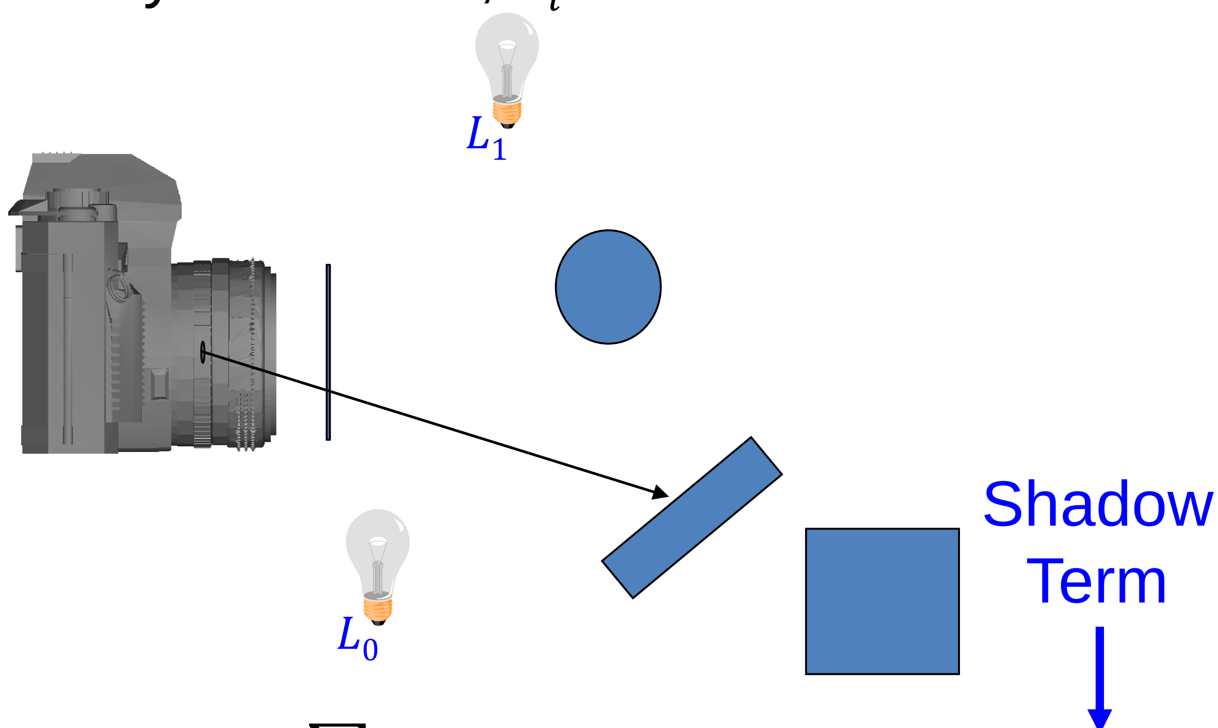
Shadow term tells if light sources are blocked

- Cast ray towards each light.
  If the ray is blocked, ignore the light's contribution.

Shadows

Shadow term tells if light sources are blocked

– Cast ray towards each light.
  $S_i = 0$ if ray is blocked, $S_i = 1$ otherwise

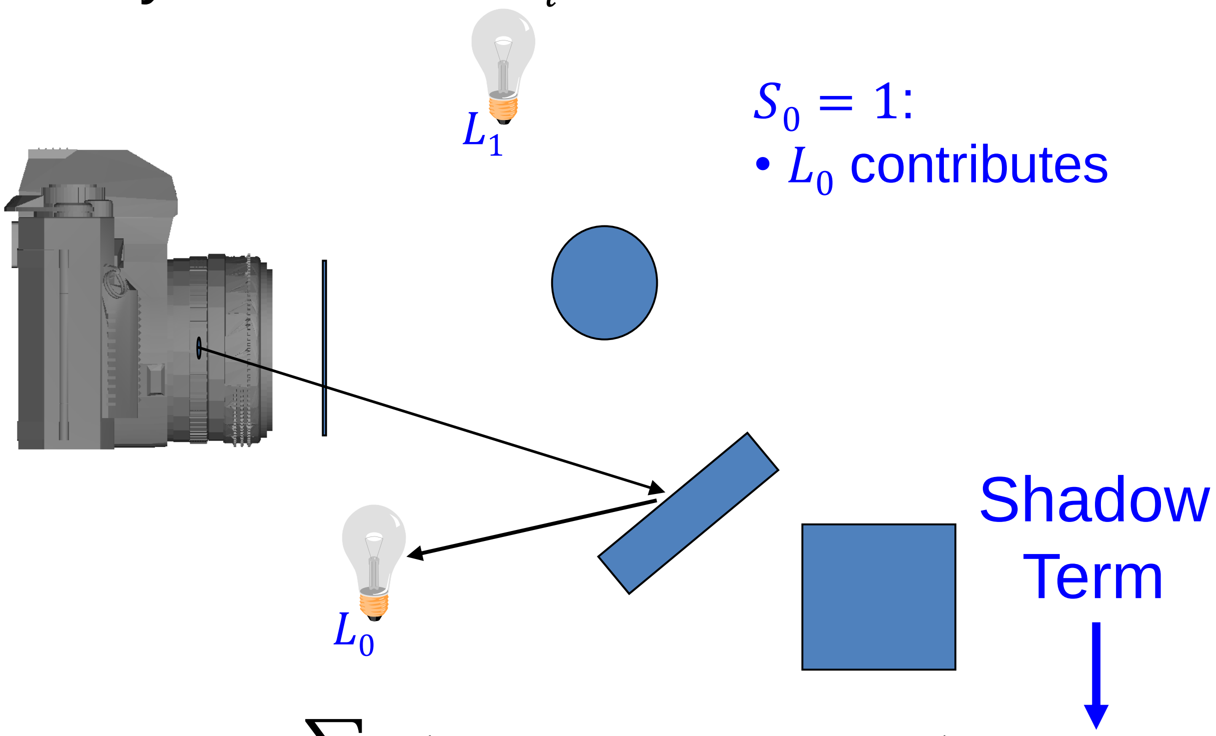


$$I = K_E + K_A + \sum_{L \in Lights} \left(\cos\theta_L \cdot K_D + \cos^k \alpha_L \cdot K_S\right) \cdot I_L \cdot S_L$$

Shadows

Shadow term tells if light sources are blocked

– Cast ray towards each light.
  $S_i = 0$ if ray is blocked, $S_i = 1$ otherwise

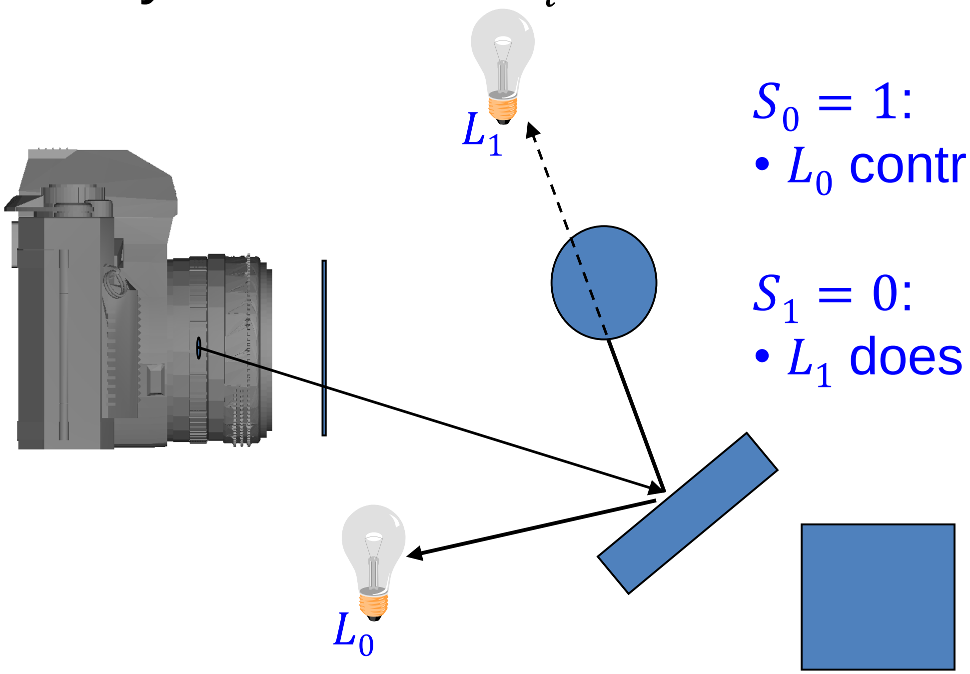


$$I = K_E + K_A + \sum_{L \in Lights} \left(\cos\theta_L \cdot K_D + \cos^k \alpha_L \cdot K_S\right) \cdot I_L \cdot S_L$$

Shadows

Shadow term tells if light sources are blocked

– Cast ray towards each light.
  $S_i = 0$ if ray is blocked, $S_i = 1$ otherwise



$S_0 = 1$:
- L_0 contributes

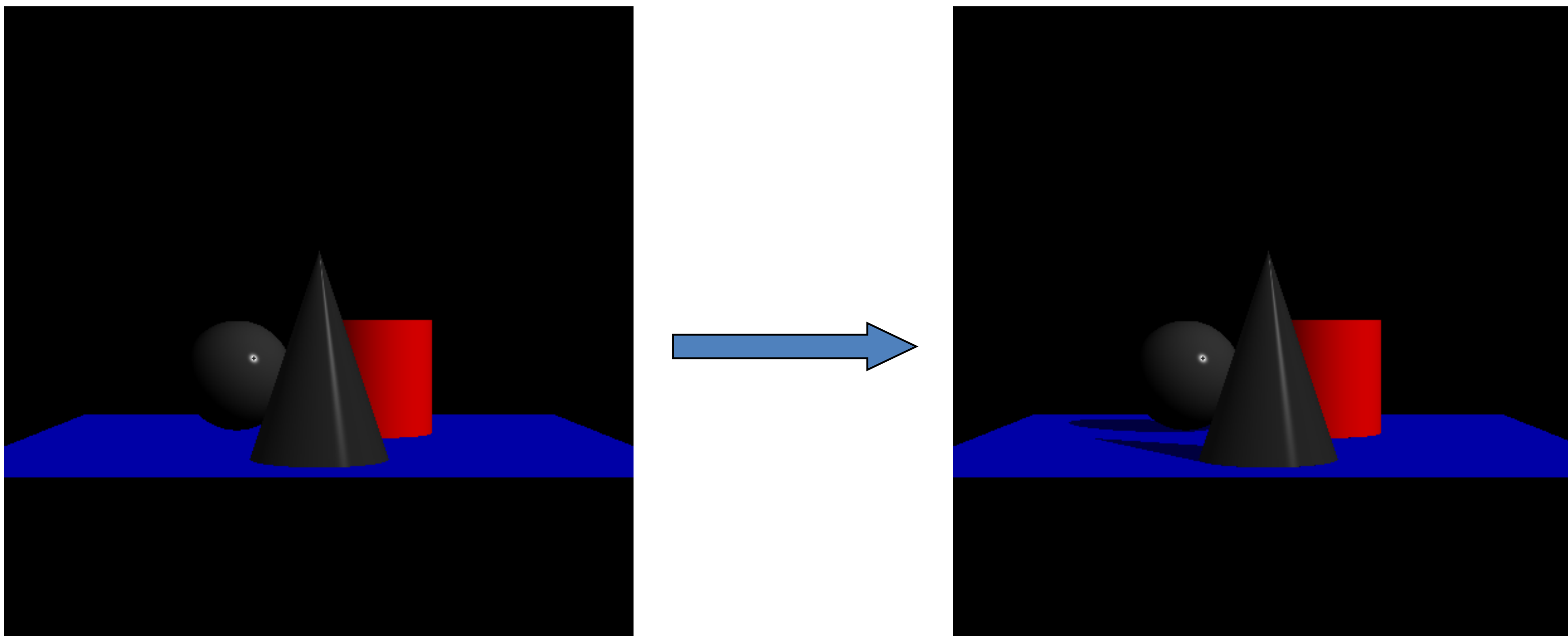
$S_1 = 0$:
- L_1 does not contribute

Shadow Term

$$I = K_E + K_A + \sum_{L \in Lights} \left(\cos\theta_L \cdot K_D + \cos^k \alpha_L \cdot K_S\right) \cdot I_L \cdot S_L$$

Ray Casting

- Trace primary rays from camera
  - Direct illumination from unblocked lights only



Recursive Ray Tracing

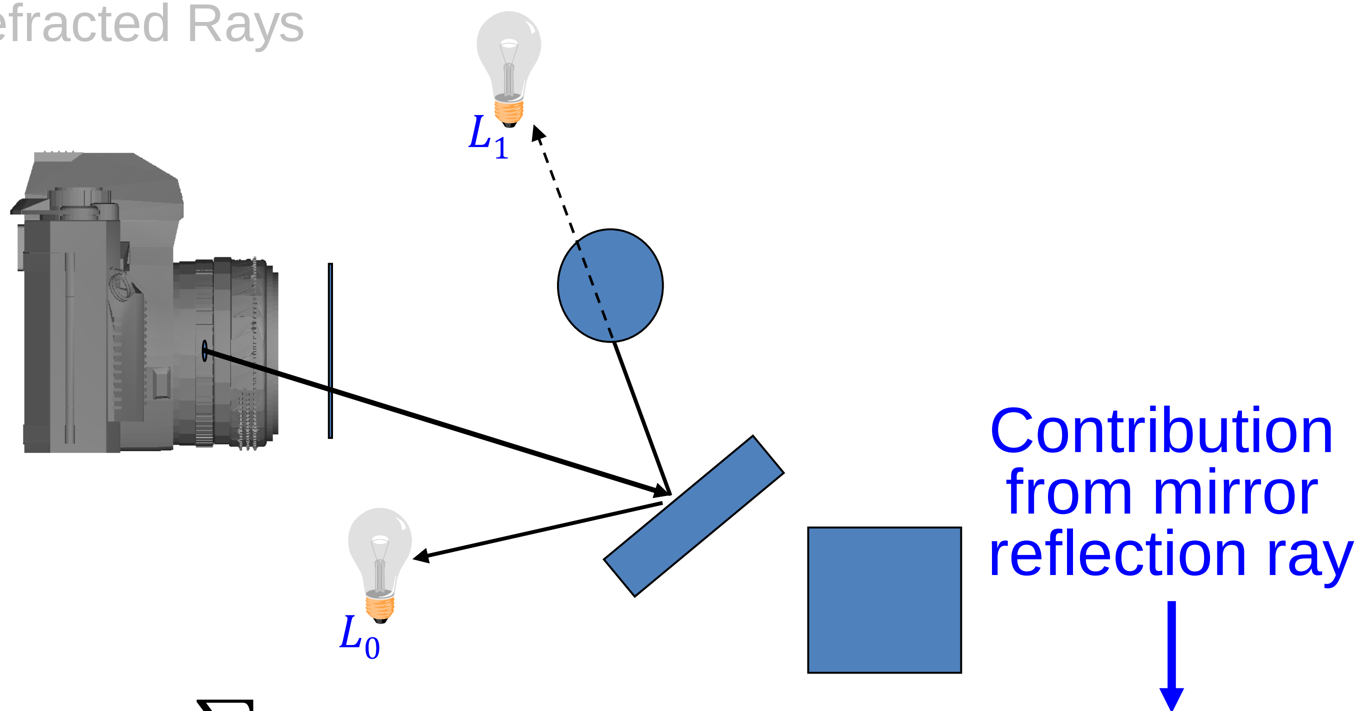
Also trace secondary rays from hit surfaces

- Consider contributions from:
  1. Reflected Rays
  2. Refracted Rays

Mirror Reflections

Also trace secondary rays from hit surfaces

- Consider contributions from:
  1. Reflected Rays
  2. Refracted Rays

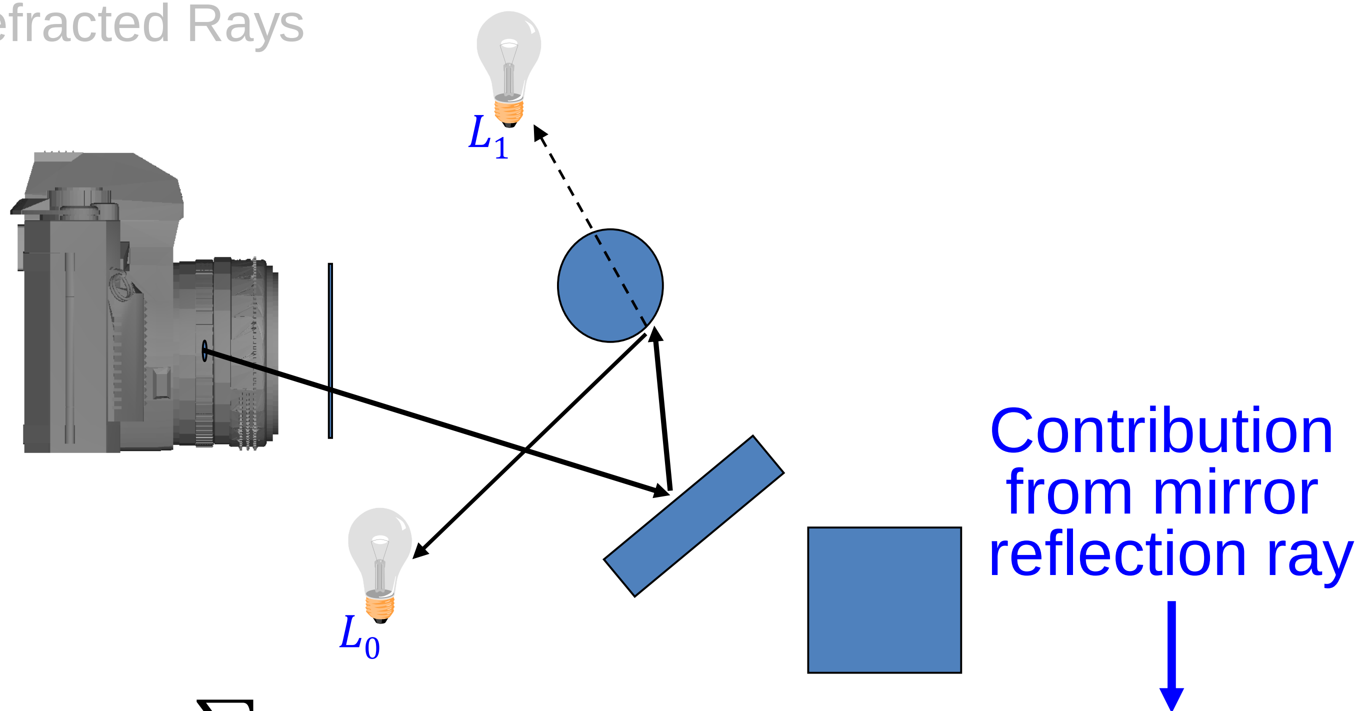


$$I = K_E + K_A + \sum_{L \in Lights} \left(\cos\theta_L \cdot K_D + \cos^k \alpha_L \cdot K_S\right) \cdot I_L \cdot S_L + K_S \cdot R$$

Mirror Reflections

Also trace secondary rays from hit surfaces

– Consider contributions from:
  1. Reflected Rays
  2. Refracted Rays

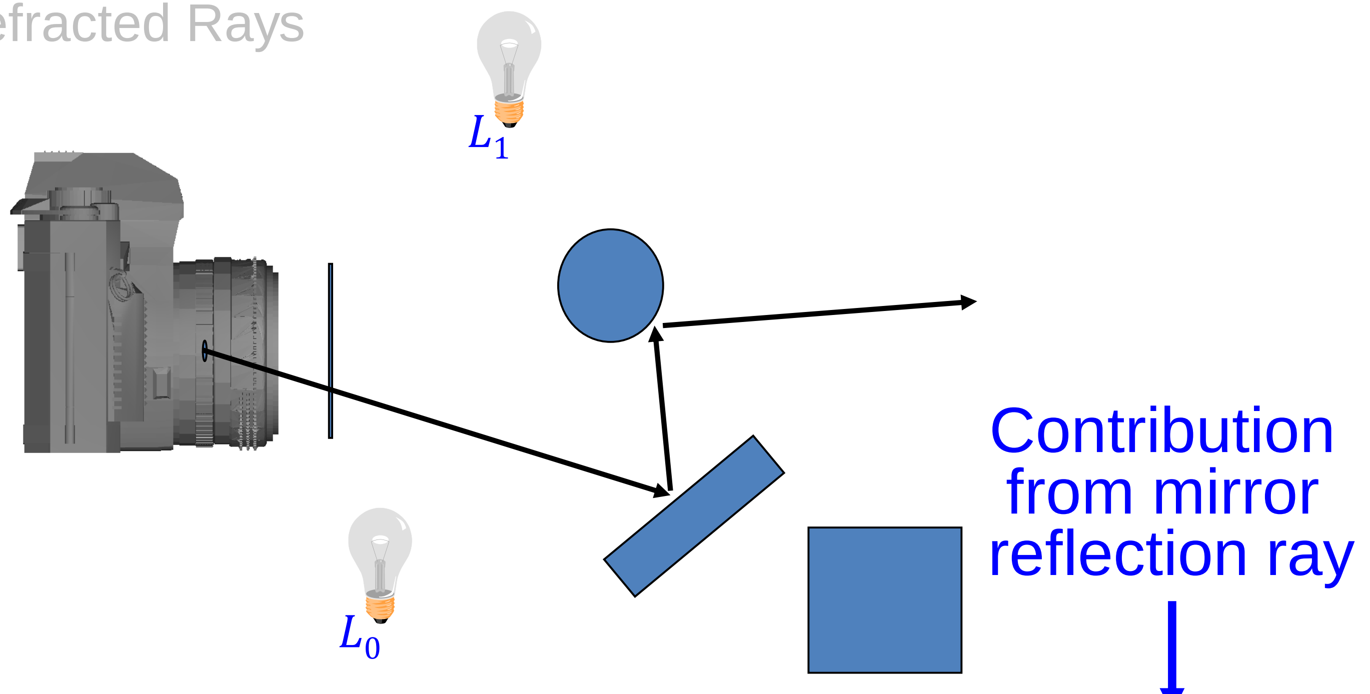


$$I = K_E + K_A + \sum_{L \in Lights} \left(\cos\theta_L \cdot K_D + \cos^k \alpha_L \cdot K_S\right) \cdot I_L \cdot S_L + K_S \cdot R$$

Mirror Reflections

Also trace secondary rays from hit surfaces

- Consider contributions from:
  1. Reflected Rays
  2. Refracted Rays

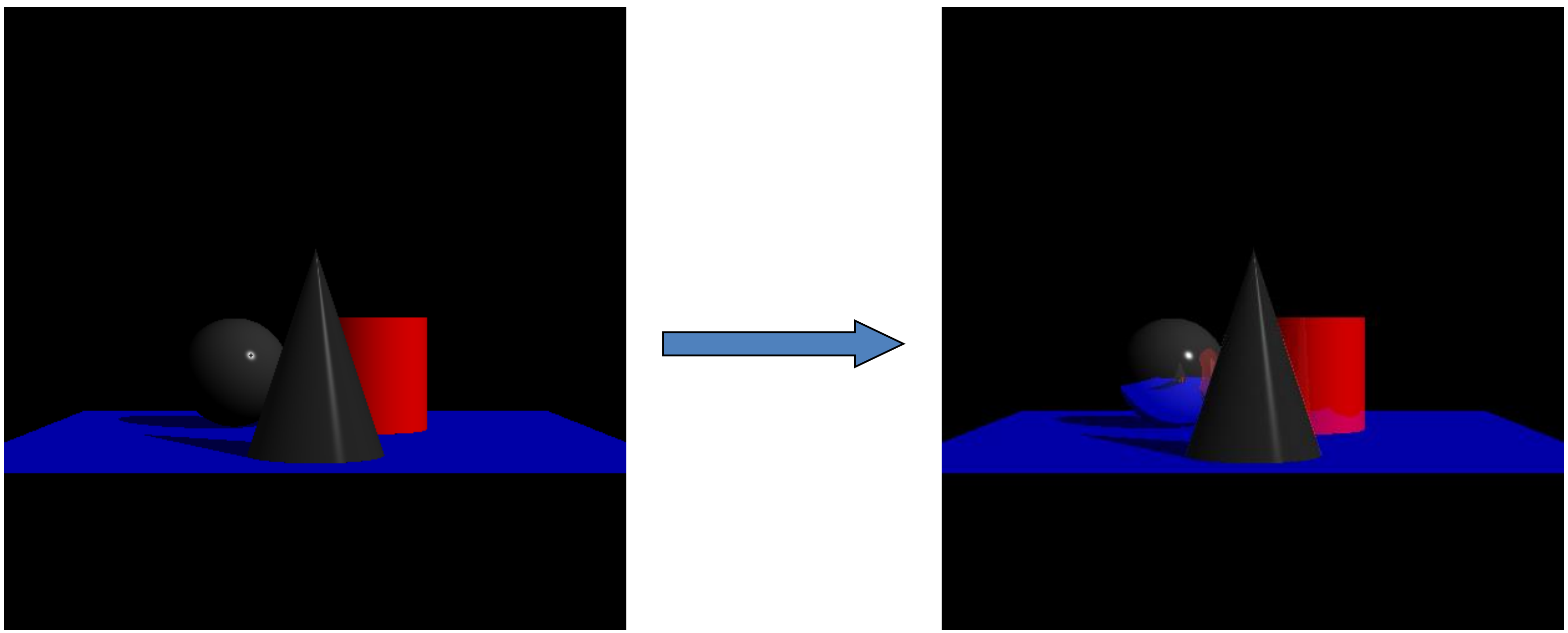


$$I = K_E + K_A + \sum_{L \in Lights} \left(\cos\theta_L \cdot K_D + \cos^k \alpha_L \cdot K_S\right) \cdot I_L \cdot S_L + K_S \cdot R$$

Mirror Reflections

Also trace secondary rays from hit surfaces

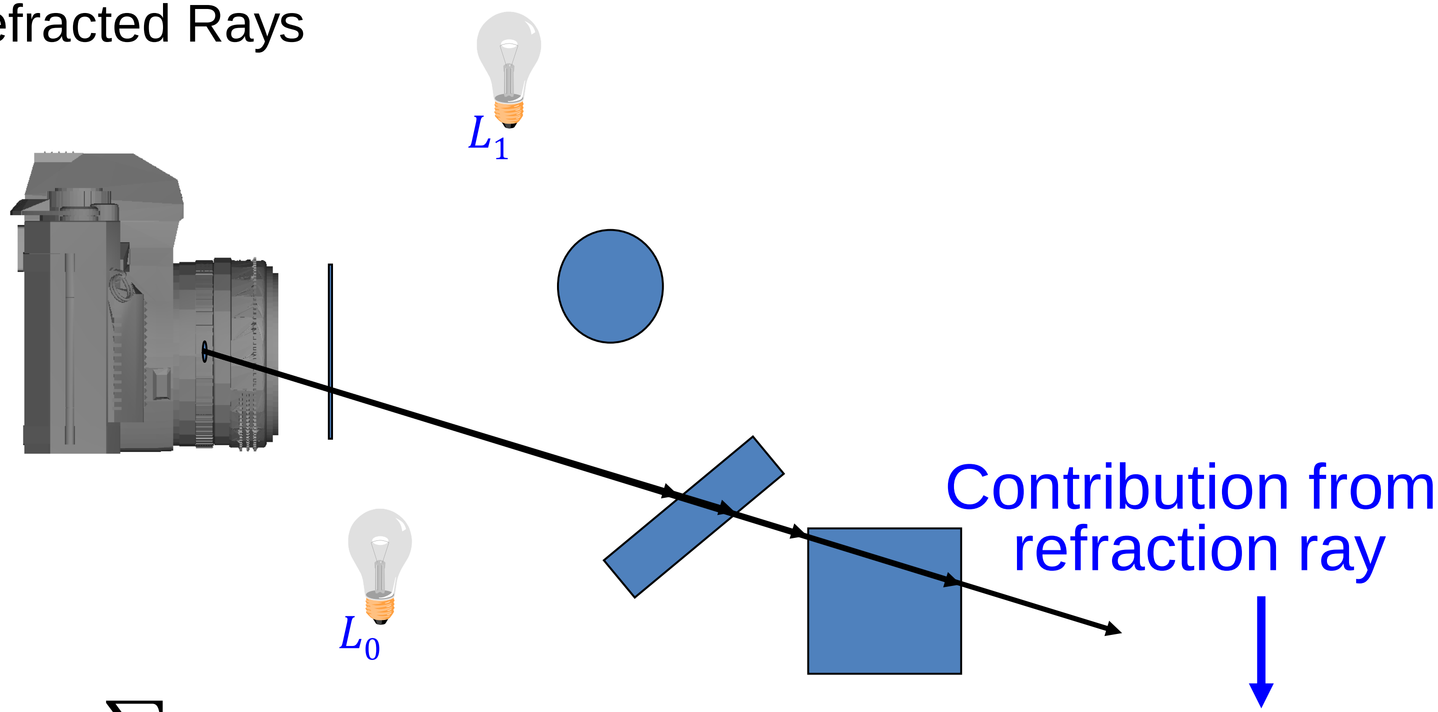
- Consider contributions from:
  1. Reflected Rays
  2. Refracted Rays



Transparency

Also trace secondary rays from hit surfaces

- Consider contributions from:
  1. Reflected Rays
  2. Refracted Rays

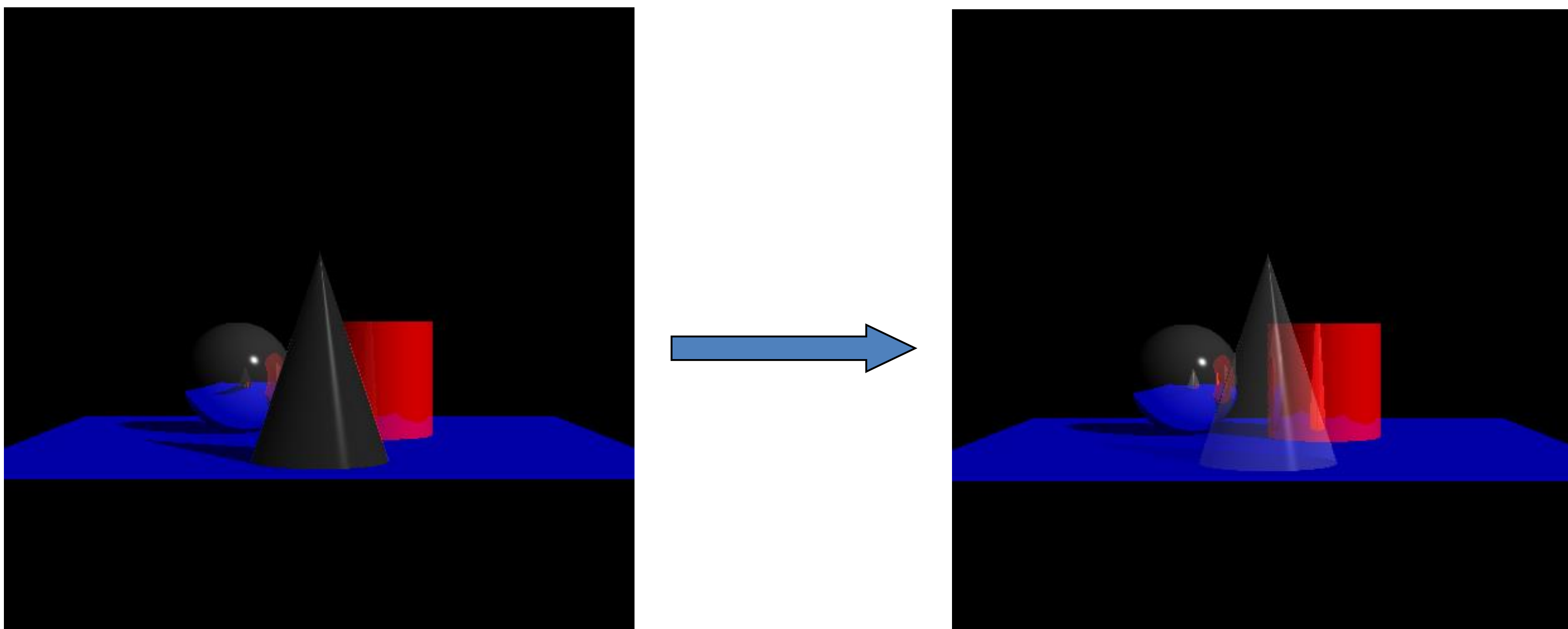


$$I = K_E + K_A + \sum_{L \in Lights} \left(\cos\theta_L \cdot K_D + \cos^k \alpha_L \cdot K_S\right) \cdot I_L \cdot S_L + K_S \cdot R + K_T \cdot T$$

Transparency

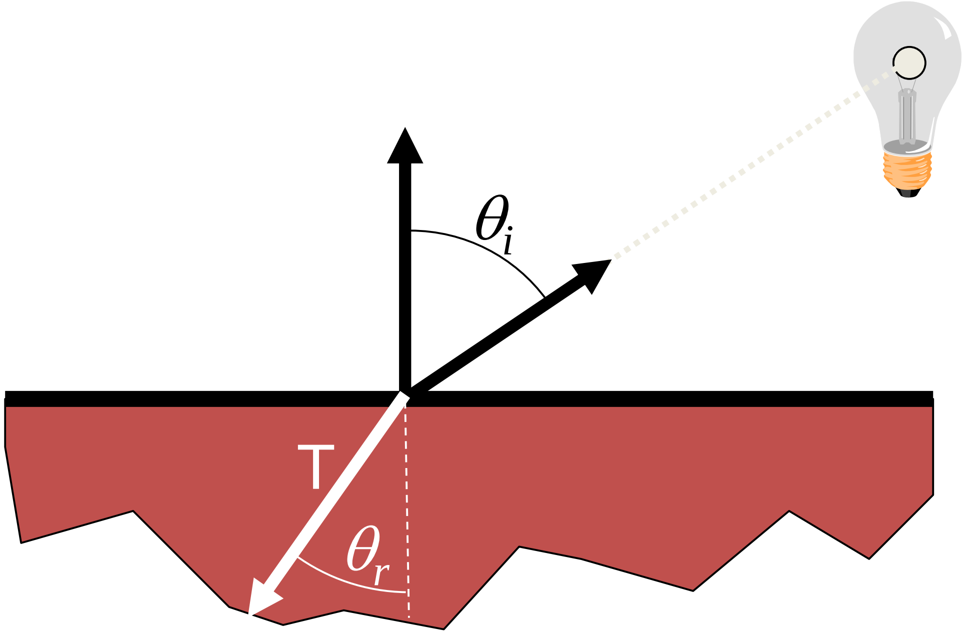
Also trace secondary rays from hit surfaces

- Consider contributions from:
  1. Reflected Rays
  2. Refracted Rays



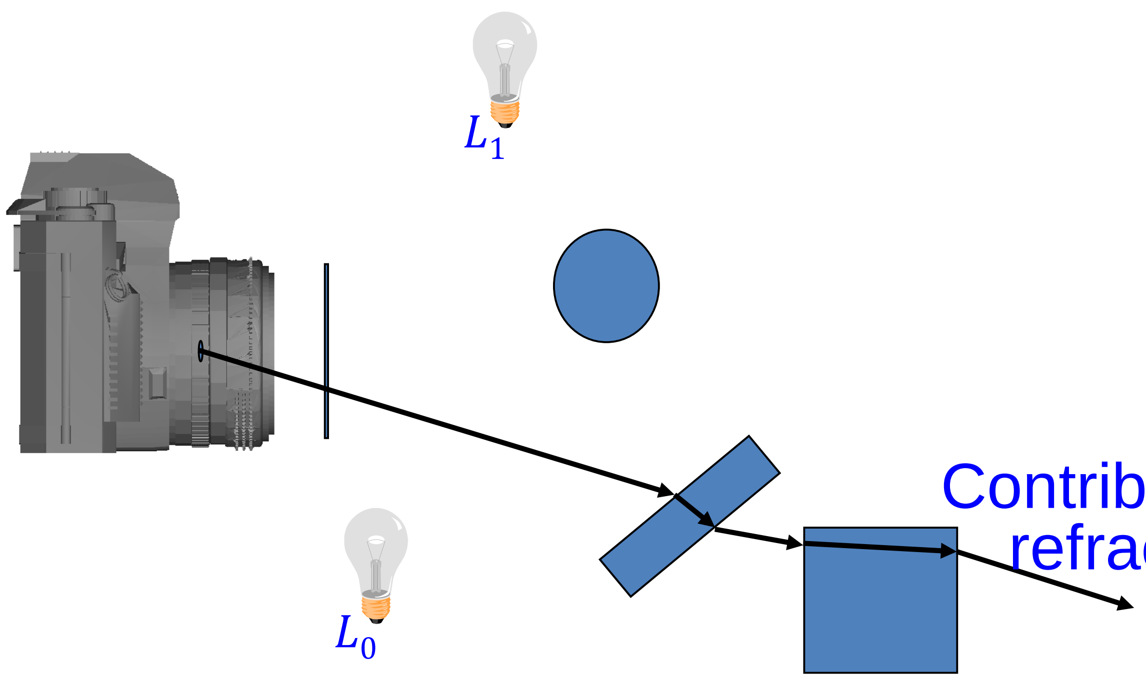
Refraction (Snell's Law)

Light bends as it passes through a transparent object ($\theta_i \neq \theta_r$).



Refraction (Snell's Law)

Light bends as it passes through a transparent object ($\theta_i \neq \theta_r$).

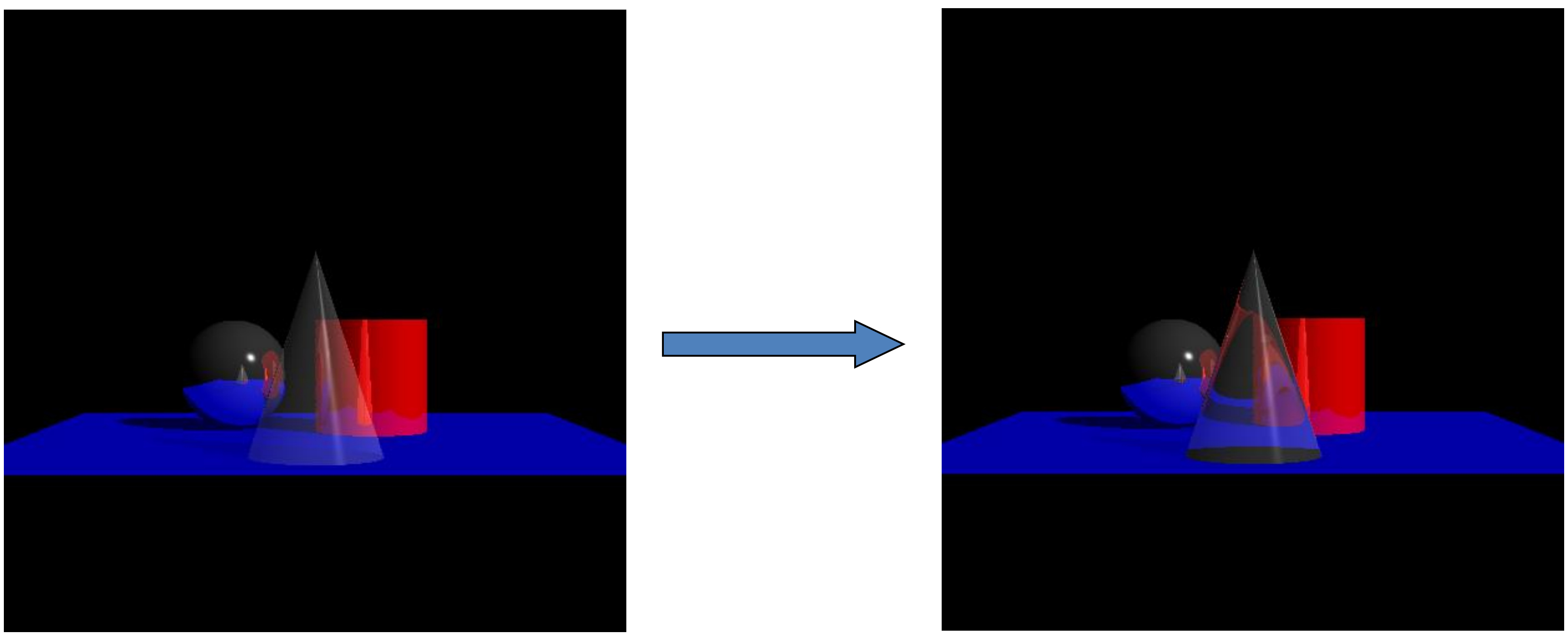


Contribution from refraction ray

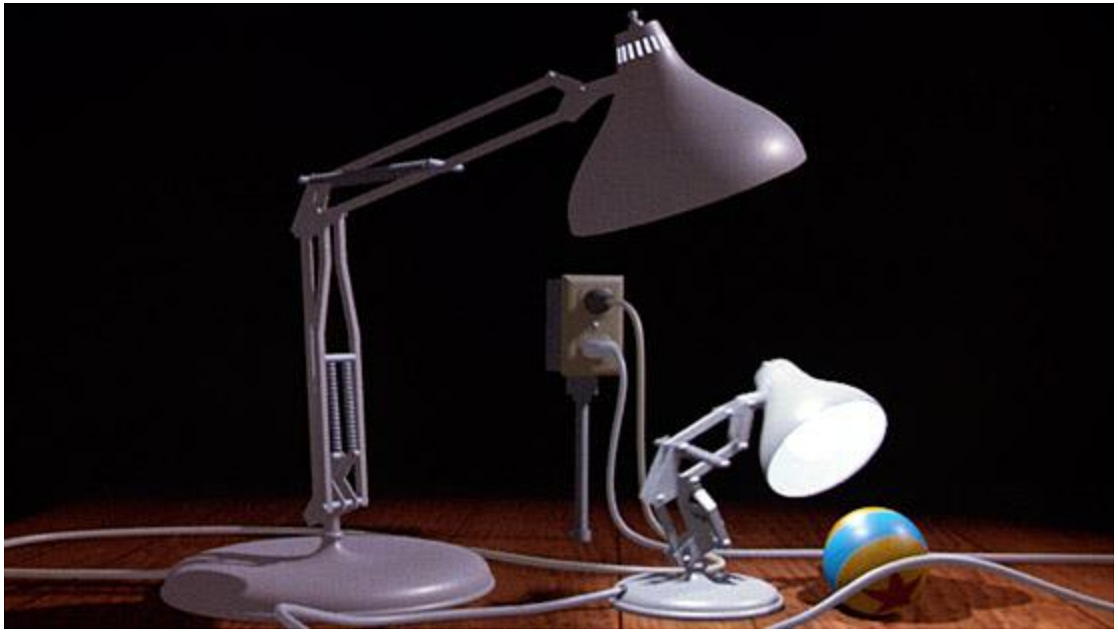
$$I = K_E + K_A + \sum_{L \in Lights} \left(\cos\theta_L \cdot K_D + \cos^k \alpha_L \cdot K_S\right) \cdot I_L \cdot S_L + K_S \cdot R + K_T \cdot T$$

Refraction (Snell's Law)

Light bends as it passes through a transparent object ($\theta_i \neq \theta_r$).



Summary



Pixar

Discussion

- How do we make ray-tracing fast?
- What does this model fail to capture?