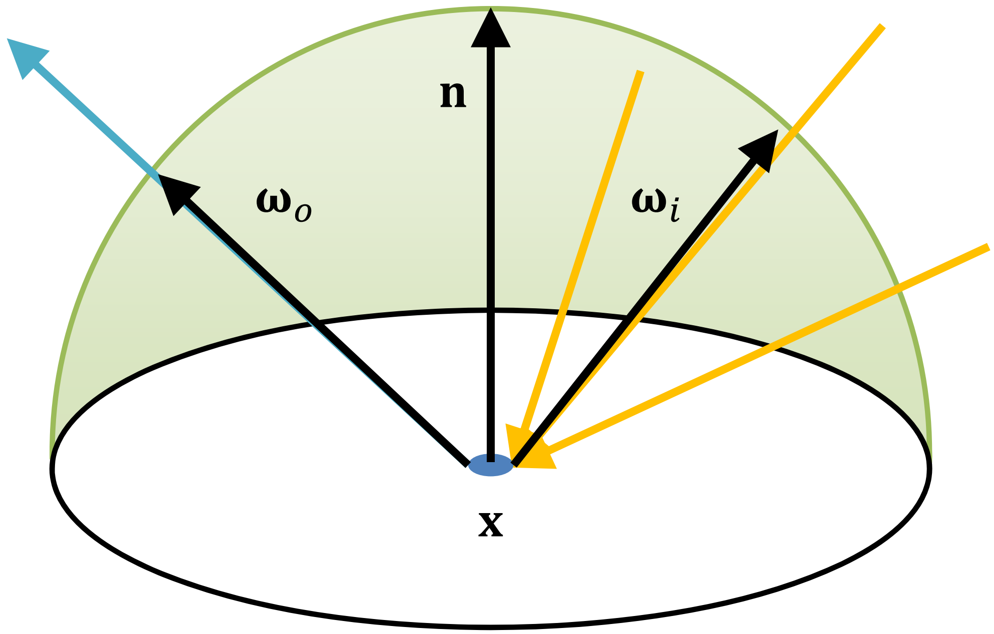

Image source: Epic Games
Shading Models

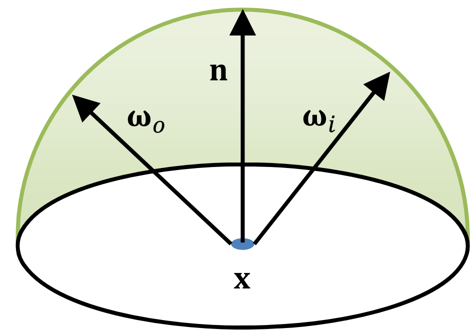
Shading

- We know which pixels are occupied by which object
  - E. g., from rasterization
- What color should the pixels have?
  - How is light reflected by an object?

What Happens at a Surface?



Rendering Equation



indirect illumination

BRDF

$$L(\mathbf{x},\boldsymbol{\omega}_o) = \underbrace{L_e(\mathbf{x},\boldsymbol{\omega}_o)}_{\text{emitted light}} + \underbrace{\int_\Omega L(\mathbf{x},\boldsymbol{\omega}_i) f_r(\mathbf{x},\boldsymbol{\omega}_i,\boldsymbol{\omega}_o)\, d\boldsymbol{\omega}_i}_{\text{reflected light}}$$

BRDF

- **B**idirectional **R**eflectance **D**istribution **F**unction
- Describes how a surface reflects light
  - At location $\mathbf{x}$
  - From incoming direction $\boldsymbol{\omega}_i$
  - Into outgoing direction $\boldsymbol{\omega}_o$

$$f_r(\mathbf{x}, \boldsymbol{\omega}_i, \boldsymbol{\omega}_o)$$

BRDF Properties

- Reciprocity

$$f_r(\mathbf{x}, \boldsymbol{\omega}_1, \boldsymbol{\omega}_2) = f_r(\mathbf{x}, \boldsymbol{\omega}_2, \boldsymbol{\omega}_1) \;\; \forall \boldsymbol{\omega}_1, \boldsymbol{\omega}_2$$

- Energy conservation

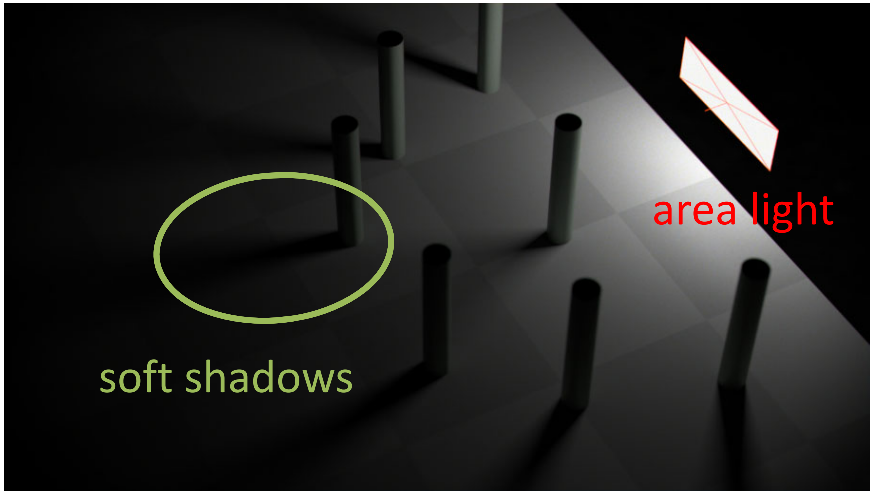
$$\int_{\Omega} f_r(\mathbf{x}, \boldsymbol{\omega}_i, \boldsymbol{\omega}_o)\, d\boldsymbol{\omega}_i \leq 1 \;\; \forall \boldsymbol{\omega}_o$$

- Positivity

$$f_r(\mathbf{x}, \boldsymbol{\omega}_1, \boldsymbol{\omega}_2) \geq 0$$

Light Sources

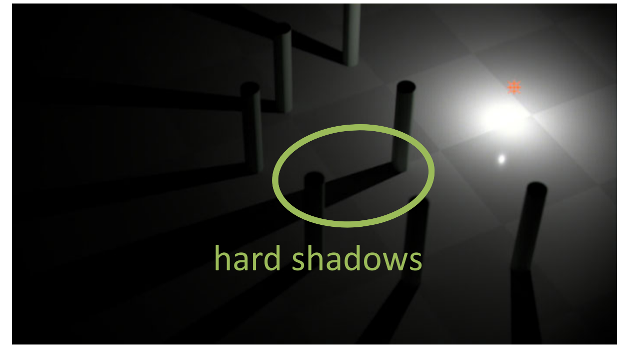
- Most general light source: area light
- Problem: at every location, light is coming from a range of directions
  - – Integration needed
  - – Analytic solution only for extremely trivial cases
  - → Simplification needed



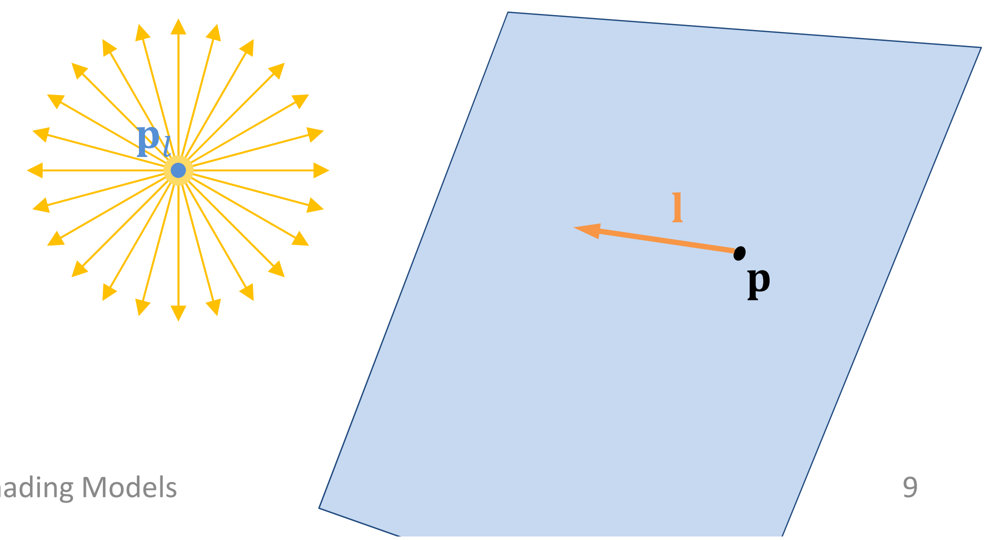
Light Sources in Real-Time

- Local shading
  - Direct illumination only
  - $O(\mathrm{n}^{\infty})$ reduces to $O(n)$
- Consider analytical light sources only
  - Point light (infinitely small)
  - Directional light (infinitely far away)
  - Light from a single direction $\mathbf{l}$ → integral vanishes

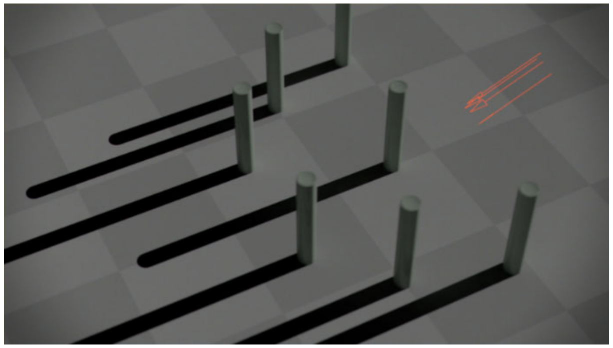
Point Light



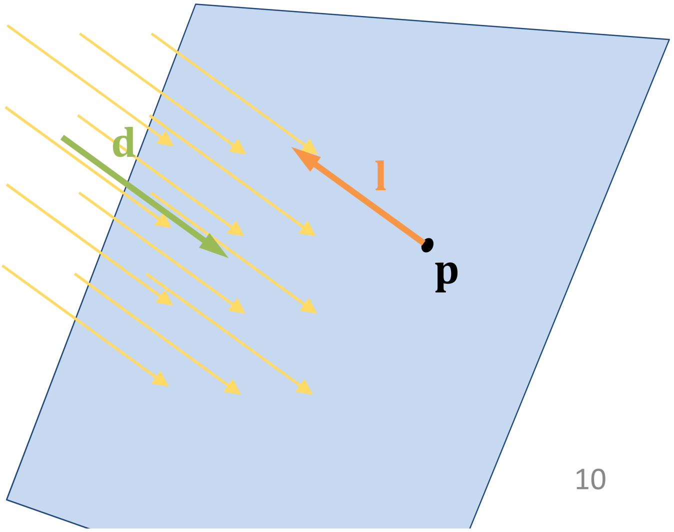
- Simplification
  - Light source infinitesimally small
  - At every location $\mathbf{p}$, light incoming from a single direction only
- Parameters:
  - $\mathbf{p}_l$ light position



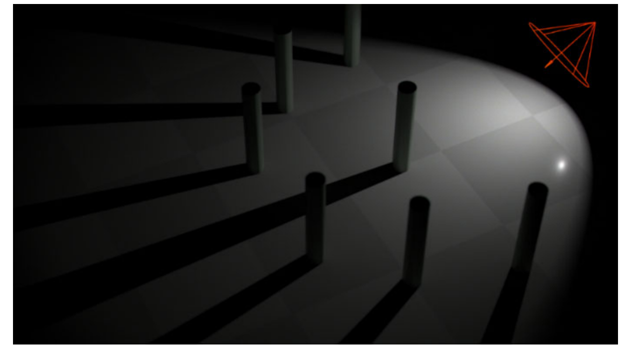
Directional Light



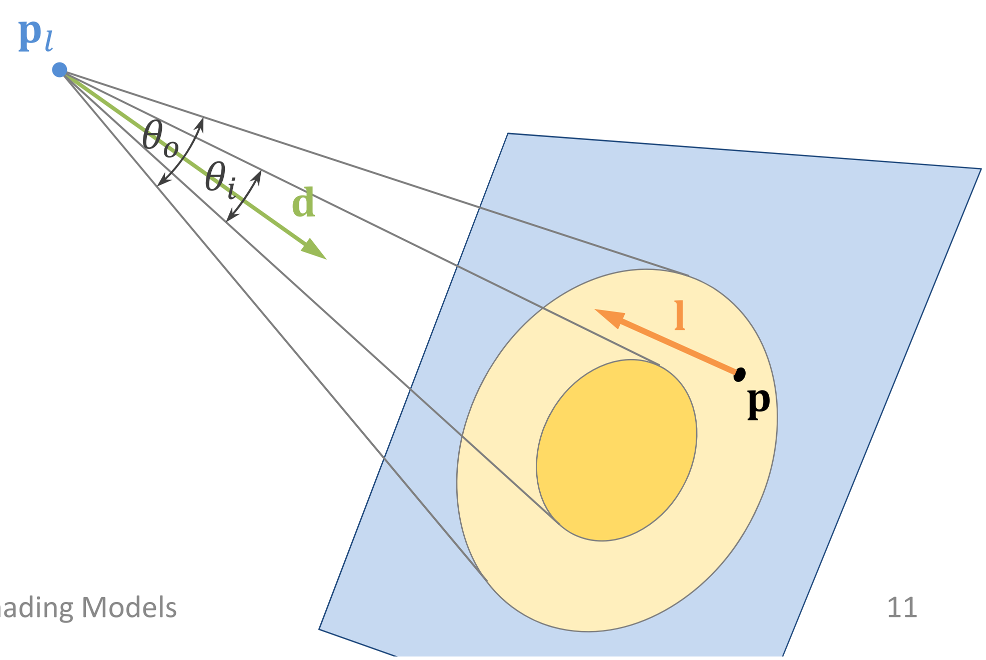
- Simplification
  - Light source infinitely far away
  - At every location $\mathbf{p}$, light from the same direction only
- Parameters:
  - $\mathbf{d}$ light direction



Spot Light



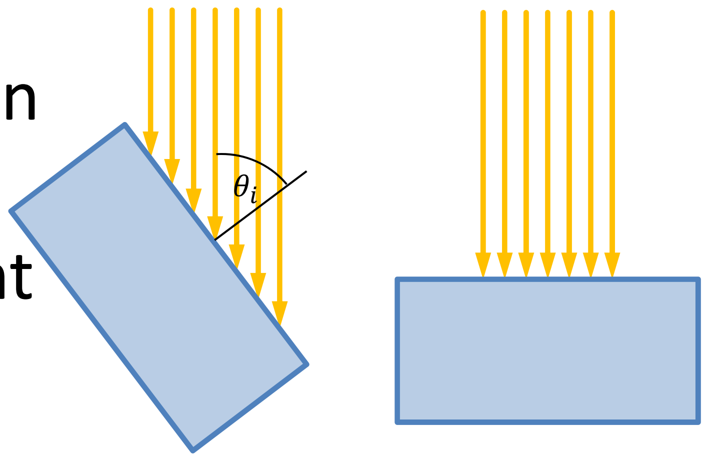
- Like point light
- Volume constrained to a cone
- Parameters:
  - $\mathbf{p}_l$ light position
  - $\mathbf{d}$ cone direction
  - θ_i inner cone angle
  - θ_o outer cone angle



Lambert Model

- Assumption of a *perfectly diffuse* reflector
- Scatters light evenly in all directions
  - View independent
  - Depends only on orientation of surface towards light
  - Surface irradiance from light

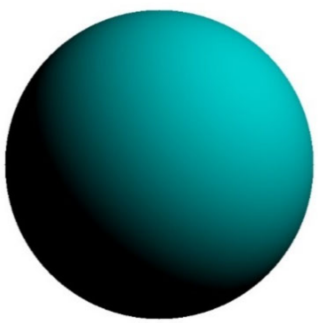
$$E_L \propto \cos\theta_i$$

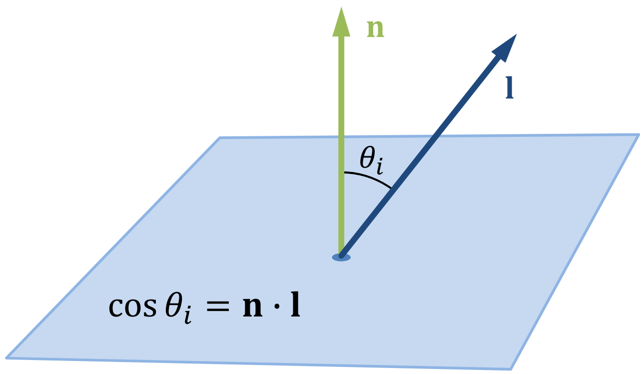


Lambert Shading Computation

$$L_o = c_d \circ \max(\mathbf{n} \cdot \mathbf{l}, 0) \circ I_L$$

- c_d Diffuse reflectance („albedo“)
- I_L Irradiance from light





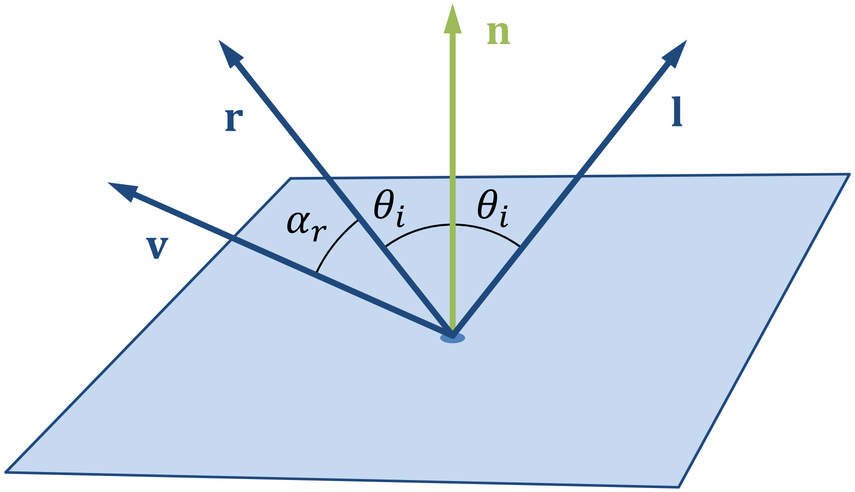
Diffuse vs Specular

Diffuse + specular

Diffuse

Phong Model

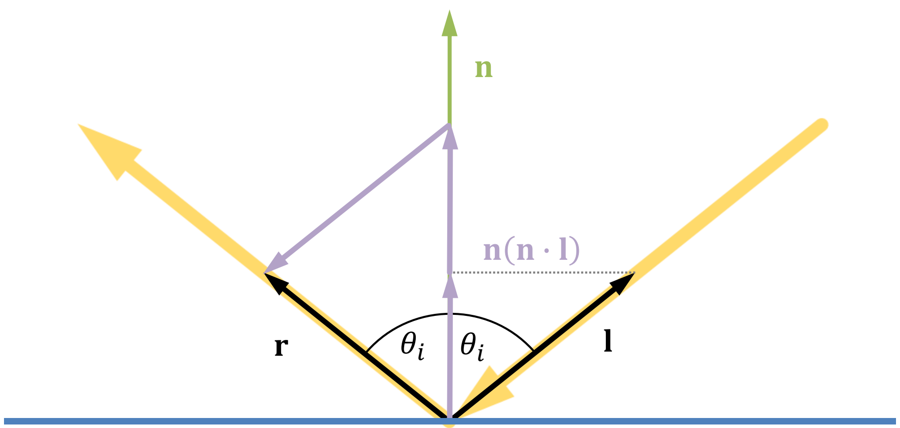
- $\mathbf{L}_o = \mathbf{c}_e + (\mathbf{c}_d \circ \cos\theta_i + \mathbf{c}_s \circ (\cos\alpha_r)^m) \circ \mathbf{B}_L$
  - $\mathbf{c}_e$ Emissive color
  - $\mathbf{c}_d$ Diffuse color
  - $\mathbf{c}_s$ Specular color
  - m Specular power
  - $\mathbf{B}_L$ Light color



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

All vectors assumed to be normalized!

Reflection



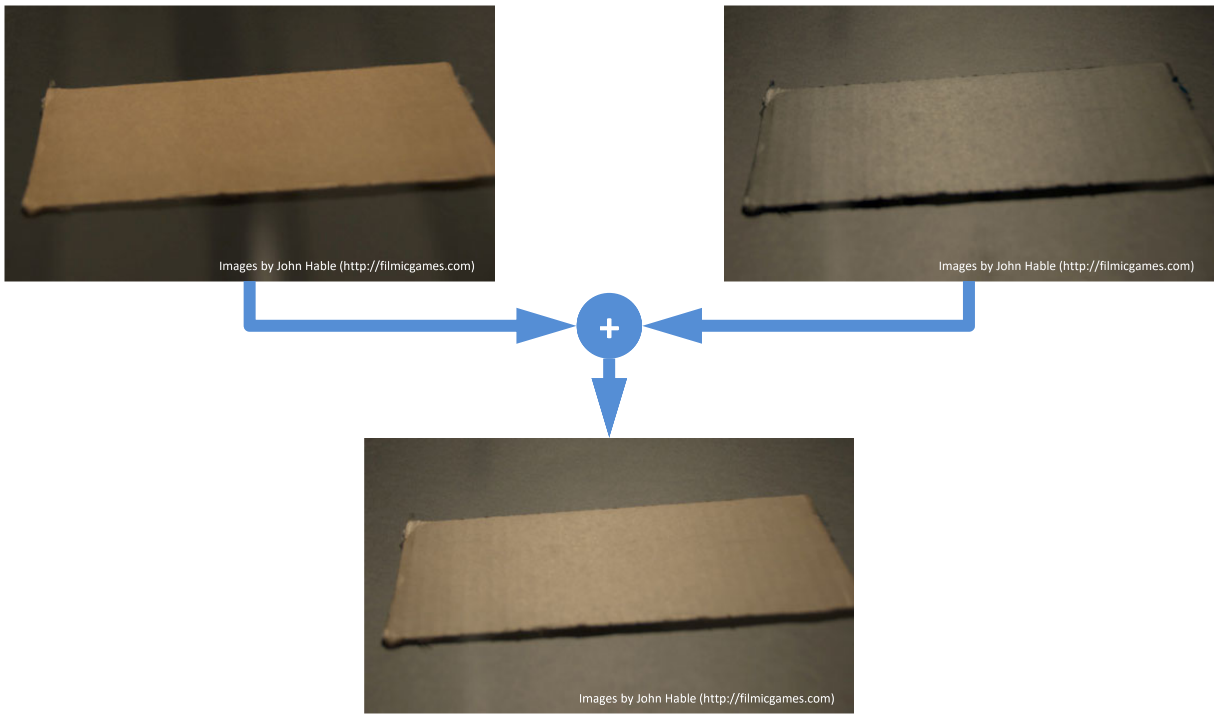
$$\mathbf{r} = 2\mathbf{n}(\mathbf{n} \cdot \mathbf{l}) - \mathbf{l}$$

All vectors assumed to be normalized!

Diffuse and Specular Combined 1

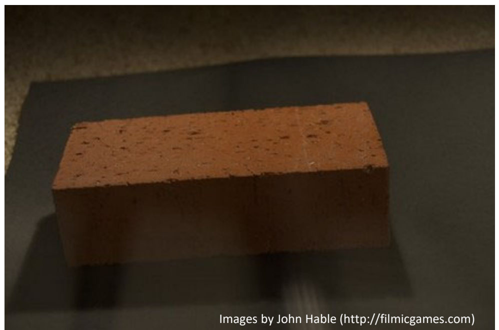
Diffuse

Specular



Diffuse and Specular Combined 2

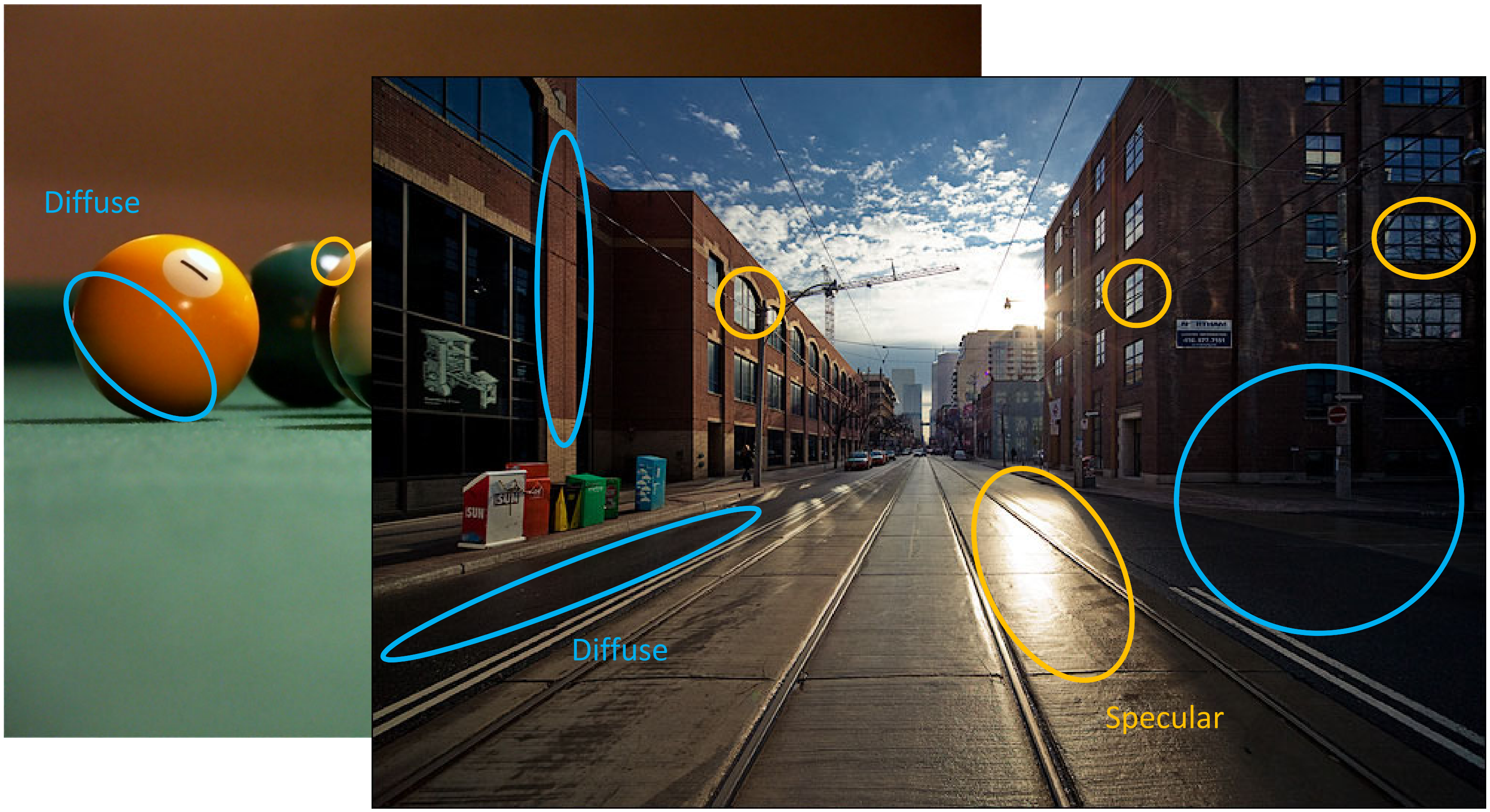
Diffuse



Specular

Specular reflection
on brick is rather white
(from light), not red

Example Images

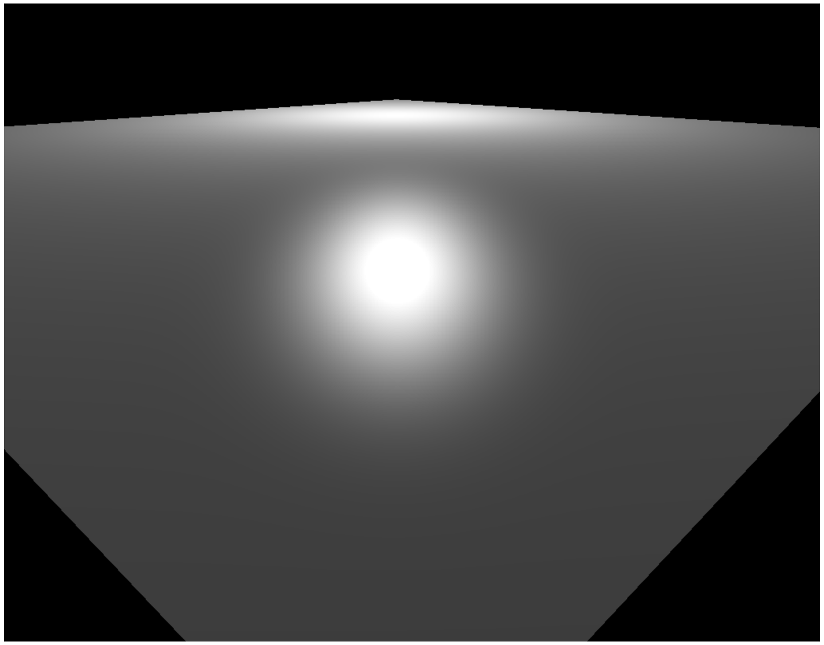


Phong Shading Properties

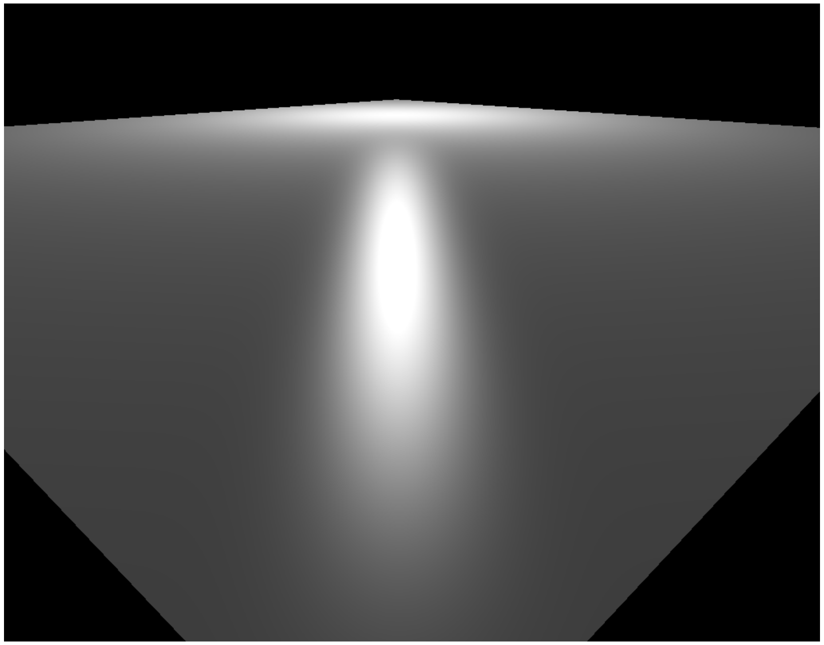
- Phenomenological model
  - Not a physically meaningful BRDF
- Highlights always circular
- Energy conservation ignored
  - Large m should lead to smaller, but stronger highlight
  - Normalization would be needed

Highlight Shape

Phong



Blinn-Phong

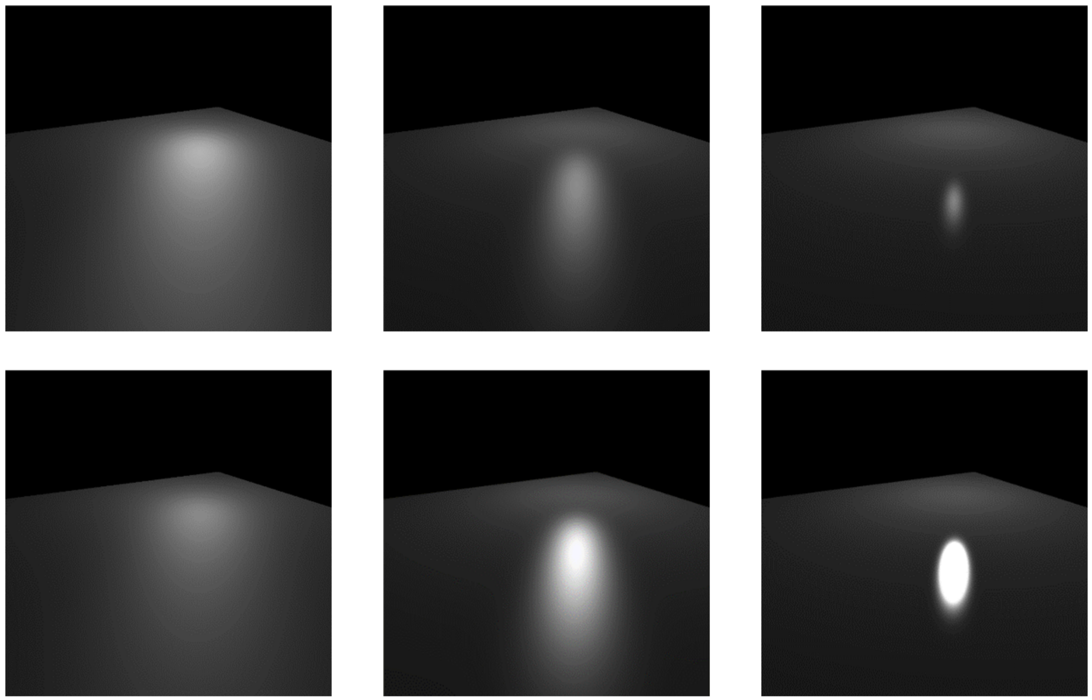


Normalization

$m = 8$ $m = 48$ $m = 512$

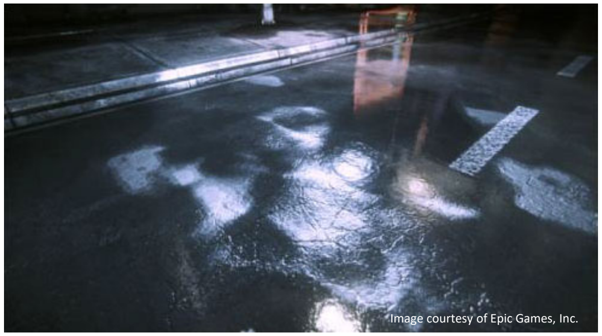
Unnormalized

Normalized



Energy Conservation

Specular tweaked for glossy

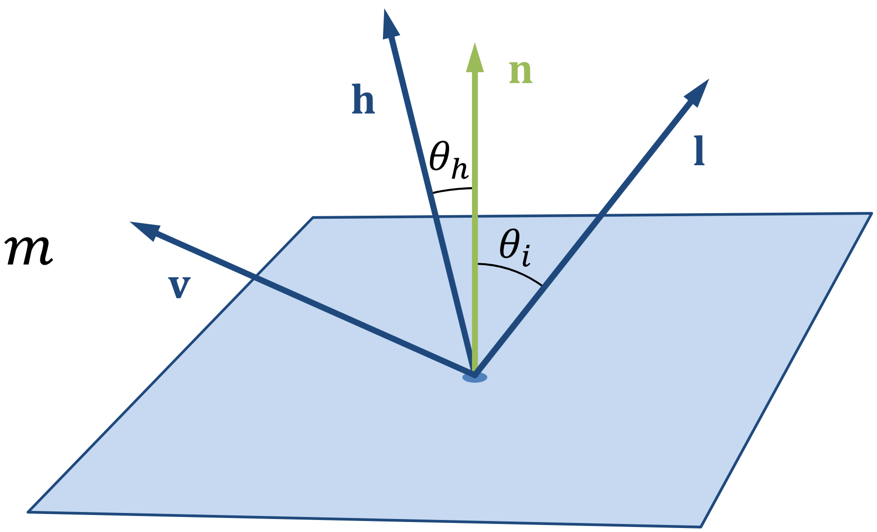


Specular tweaked for dull

Energy conserving

Blinn-Phong Model

- Approximates energy conservation
- $f_r = \frac{\mathbf{c}_d}{\pi} + \frac{m+8}{8\pi} \mathbf{c}_s (\cos\theta_h)^m$
  - $\mathbf{c}_d$ Diffuse color
  - $\mathbf{c}_s$ Specular color
  - m Specular power

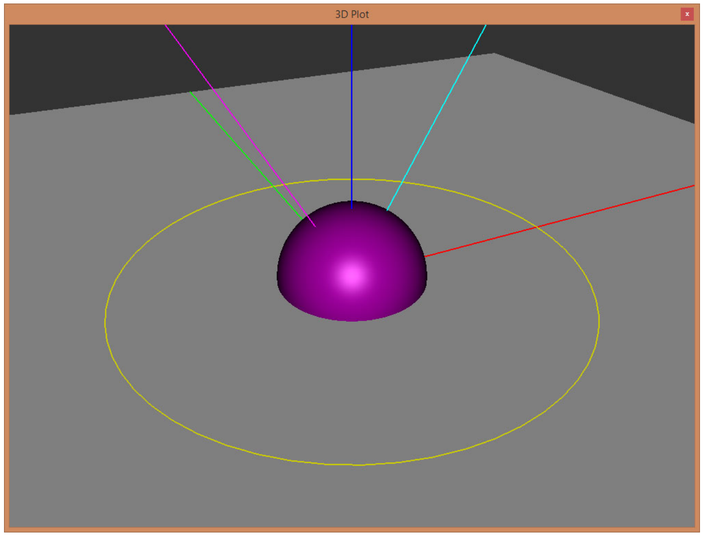


$$\mathbf{h} = \frac{\mathbf{v} + \mathbf{l}}{\|\mathbf{v} + \mathbf{l}\|}$$

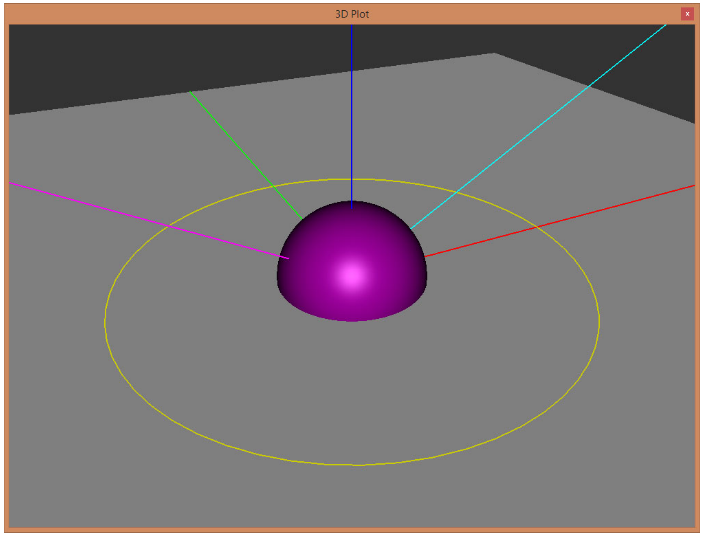
All vectors assumed to be normalized!

Lambert BRDF

$\theta_i = 30°$

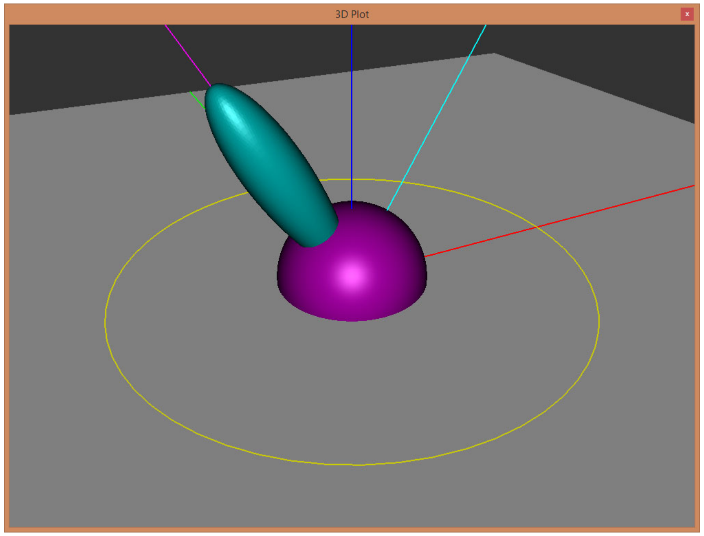


$\theta_i = 60°$

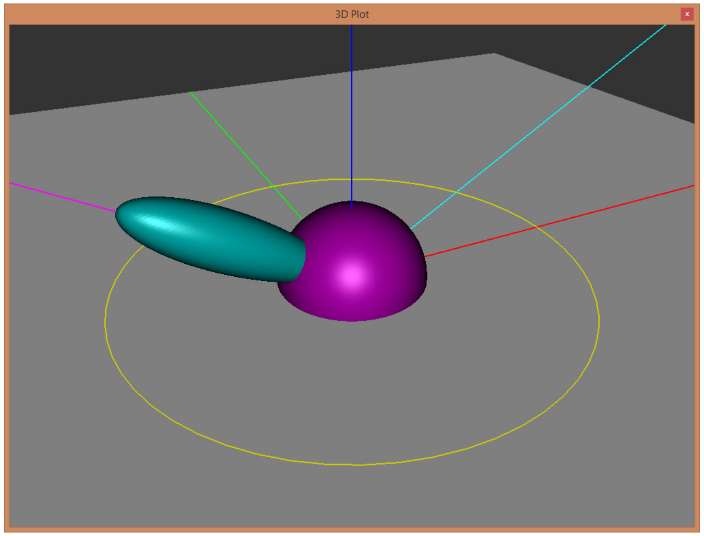


Phong BRDF 1

$\theta_i = 30°$

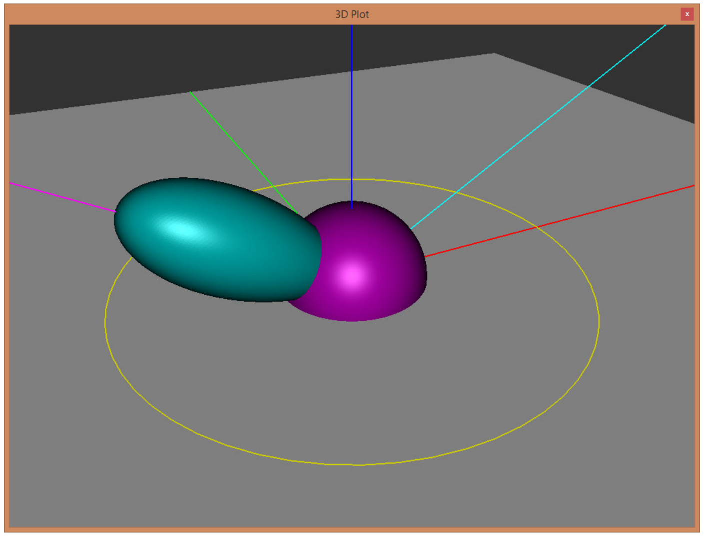


$\theta_i = 60°$

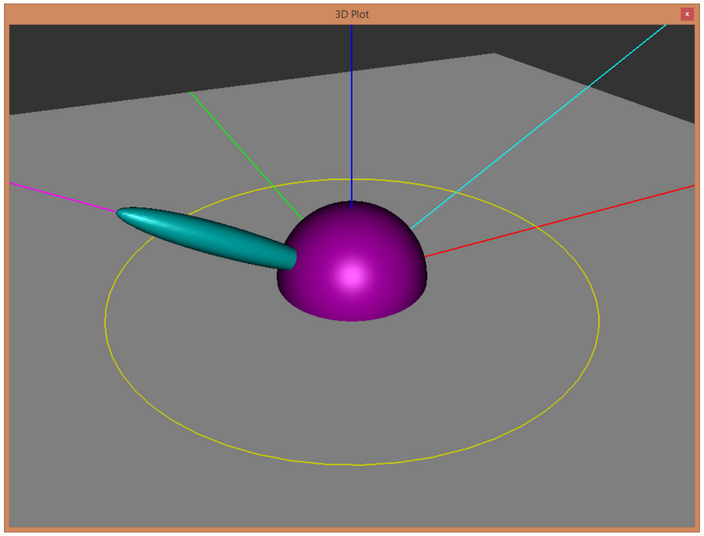


Phong BRDF 2

$m = 8$

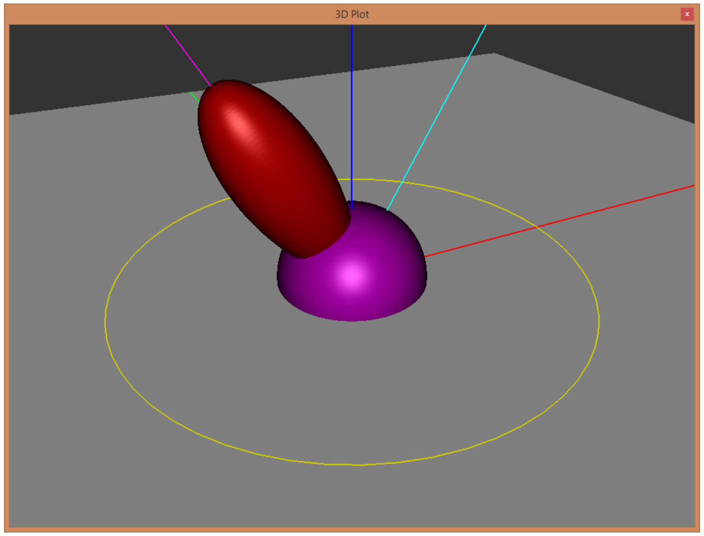


$m = 100$

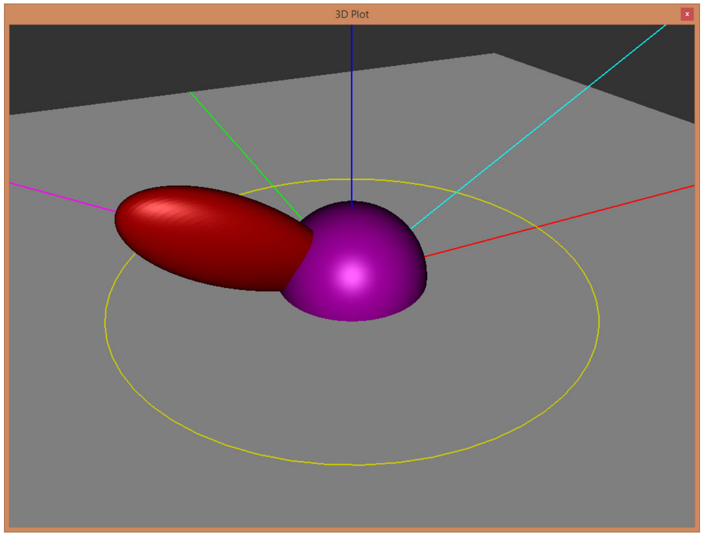


Blinn-Phong BRDF 1

$\theta_i = 30°$

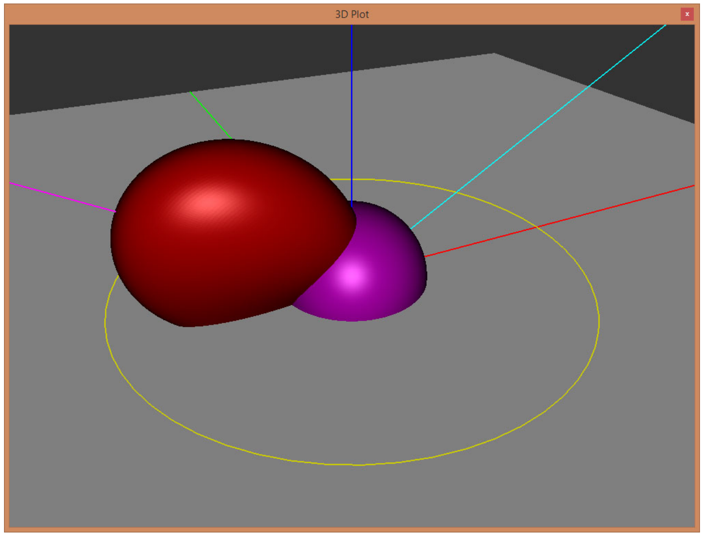


$\theta_i = 60°$

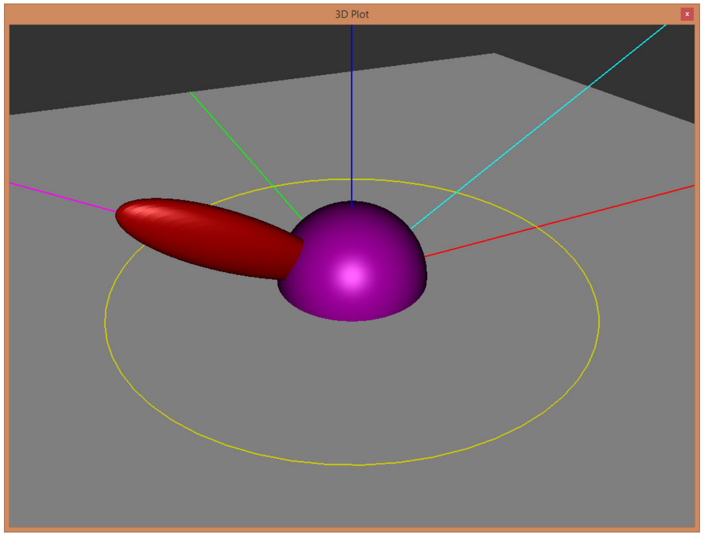


Blinn-Phong BRDF 2

$m = 8$



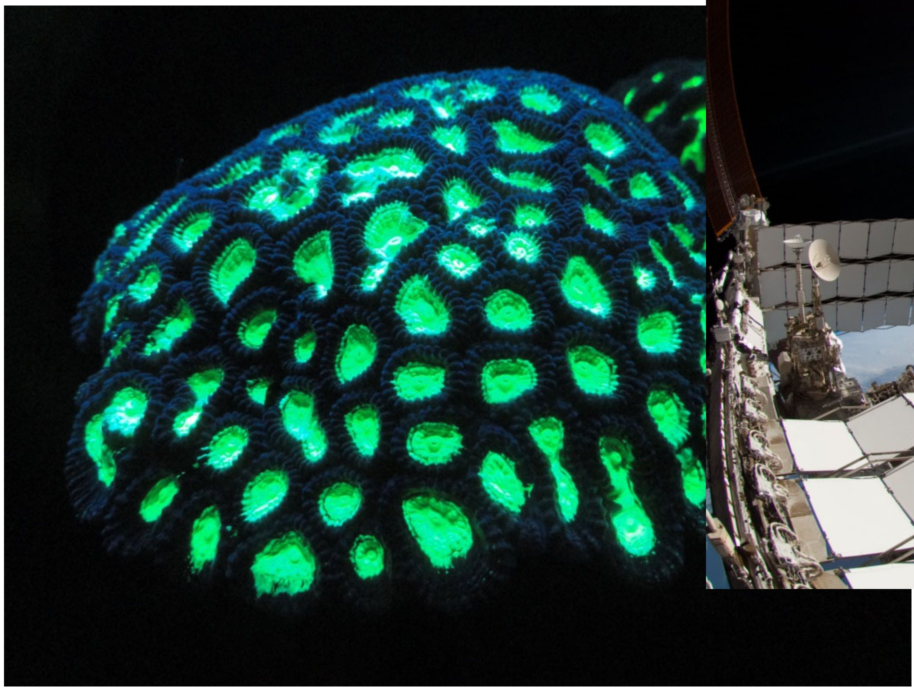
$m = 100$



Physically-Based Rendering

- Derive shading from (simplified) physics instead of just re-creating the phenomena
- All light-matter interaction boils down to
  - Emission
  - Scattering
  - Absorption

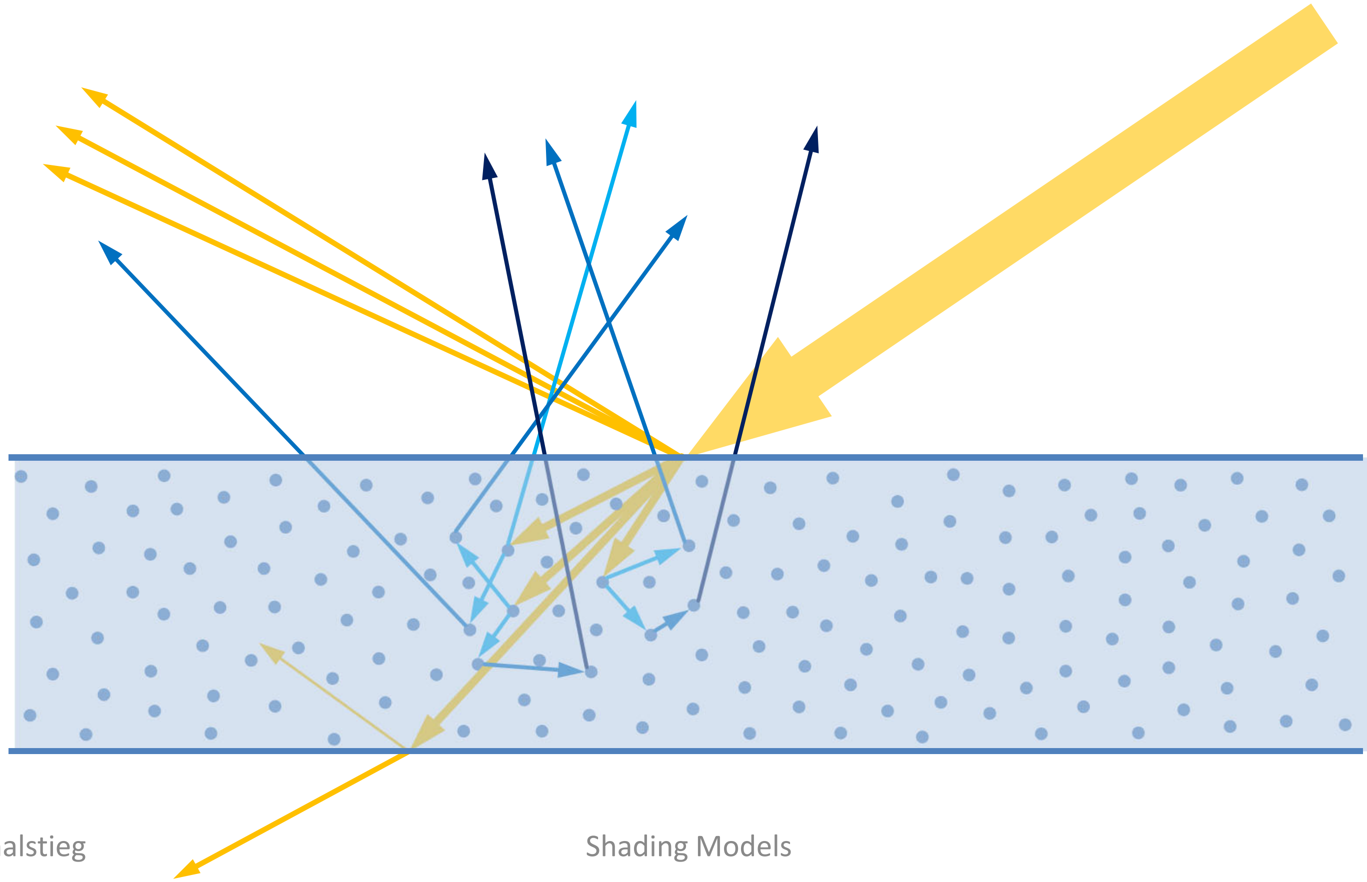
Emission



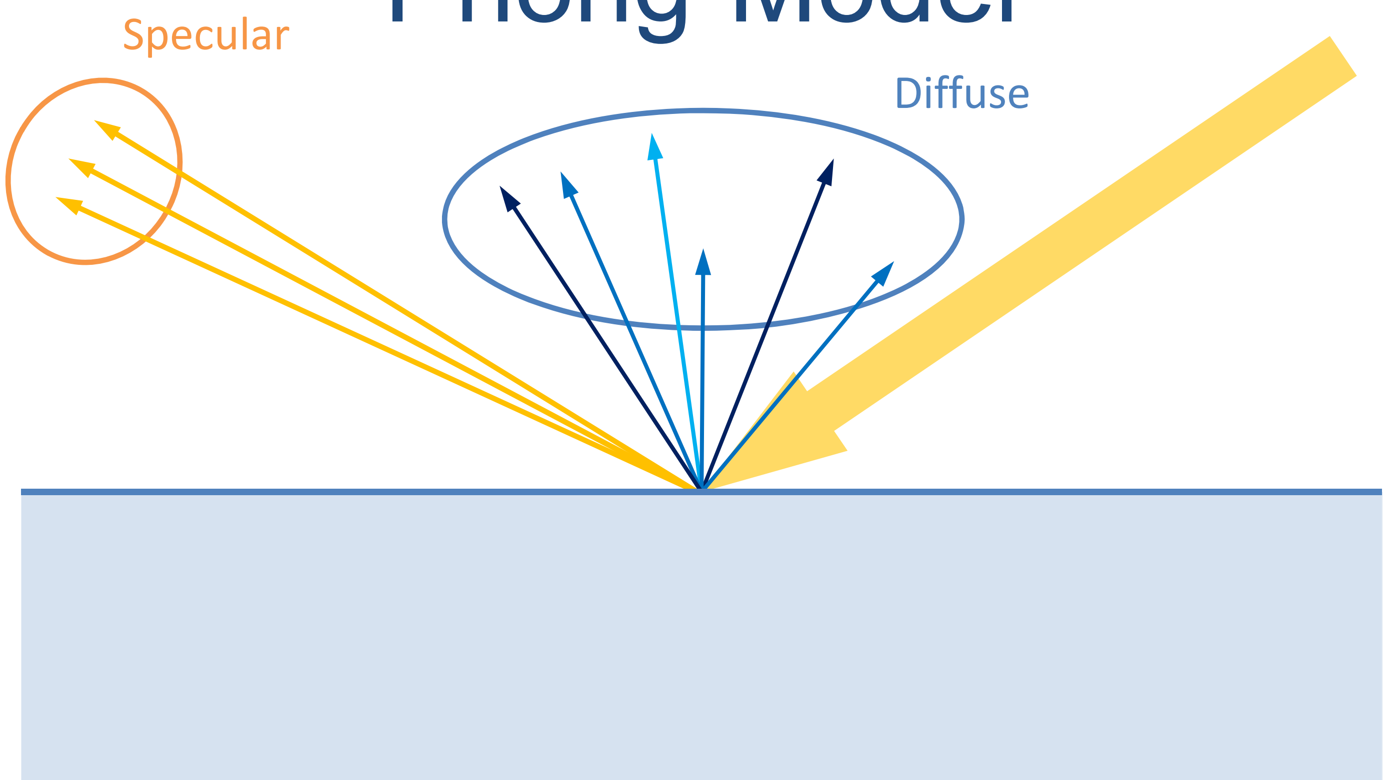
Absorption vs. Scattering

Images by Naty Hoffman

Nature



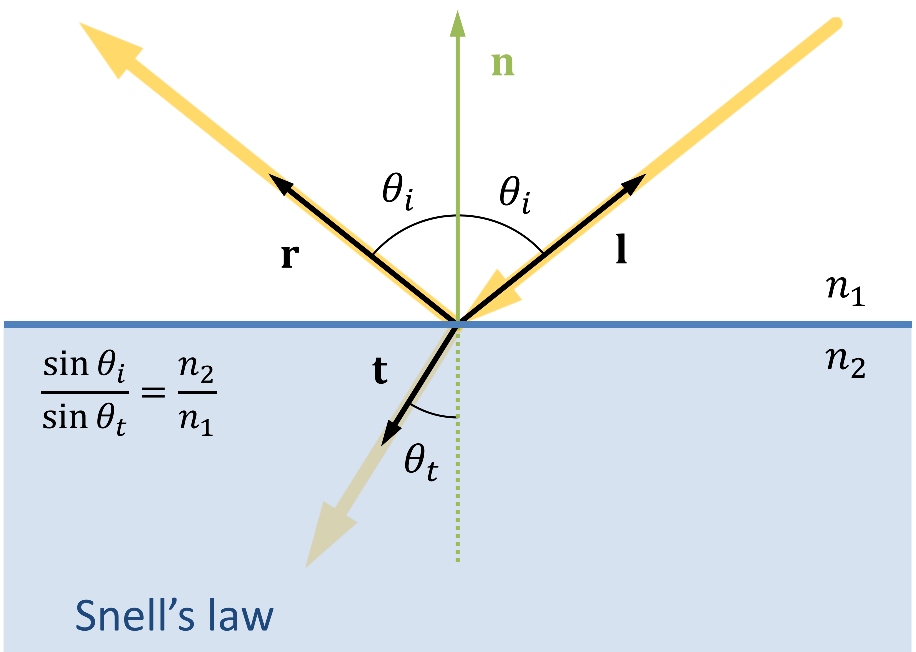
Phong-Model



Fresnel Laws

- What happens at boundary between two media
  - Phase speed of light different in each medium
- Part of the light wave is directly *specularly* reflected
- Part of the light wave is transmitted
  - We assume the material is not transparent
  - The transmitted part becomes the *diffuse* reflection
- What is the exact ratio?
  - *Augustin-Jean Fresnel knows.*

Fresnel Equations



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

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

R, *T* depend only on n_1, n_2!

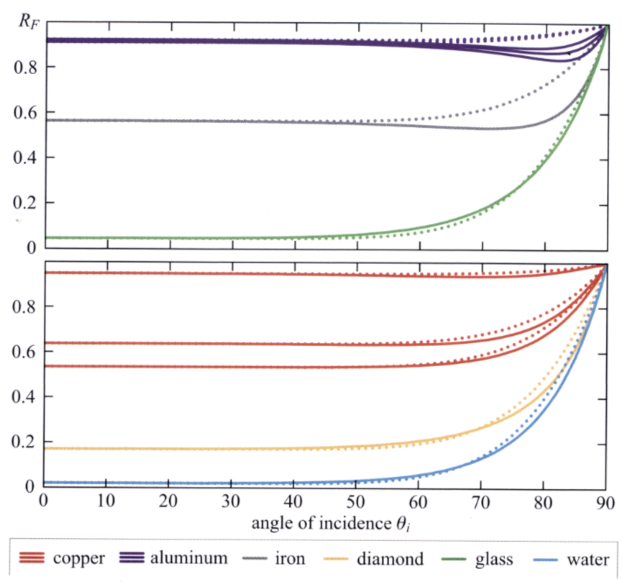
Specular $R = \frac{R_s + R_p}{2}$

$T = 1 - R$ Diffuse

Schlick's Approximation of Reflection

Easier (faster) to compute than original Fresnel equations

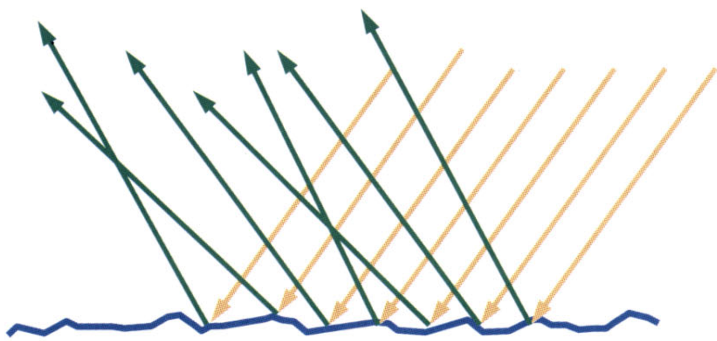
$$R = R_0 + (1 - R_0)(1 - \cos\theta_i)^5$$

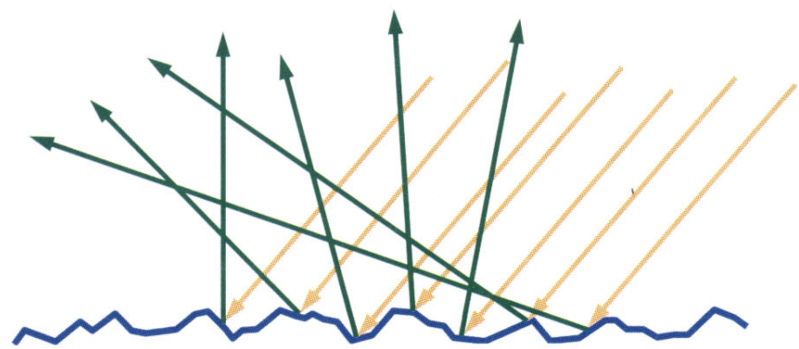


[Schlick 1994]

Image from [Akenine-Möller et al. 2008]

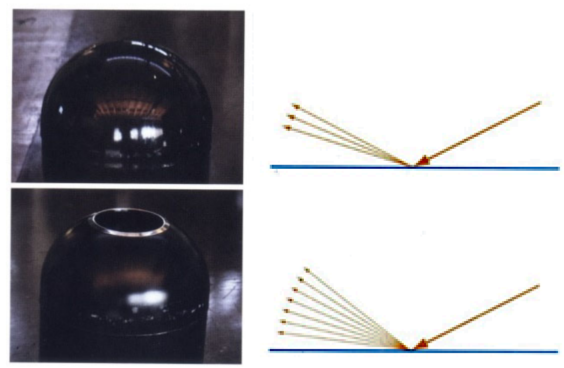
Surface Roughness





Metallic surfaces: diffuse model not really working

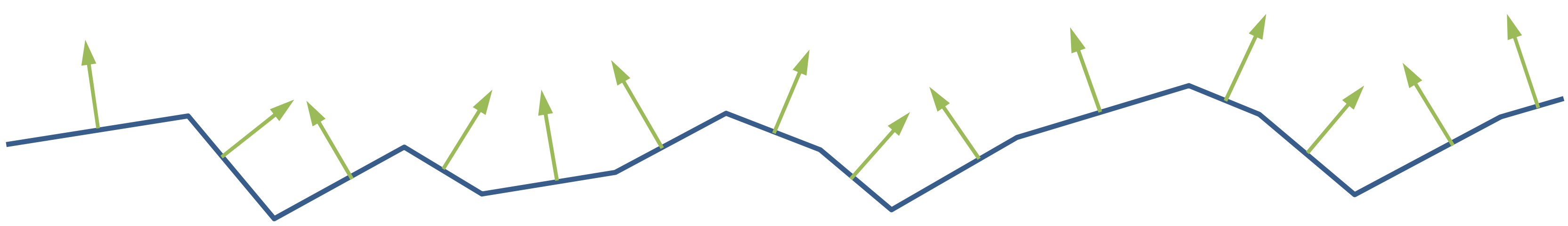
Model surface roughness instead



Images from [Akenine-Möller et al. 2008]

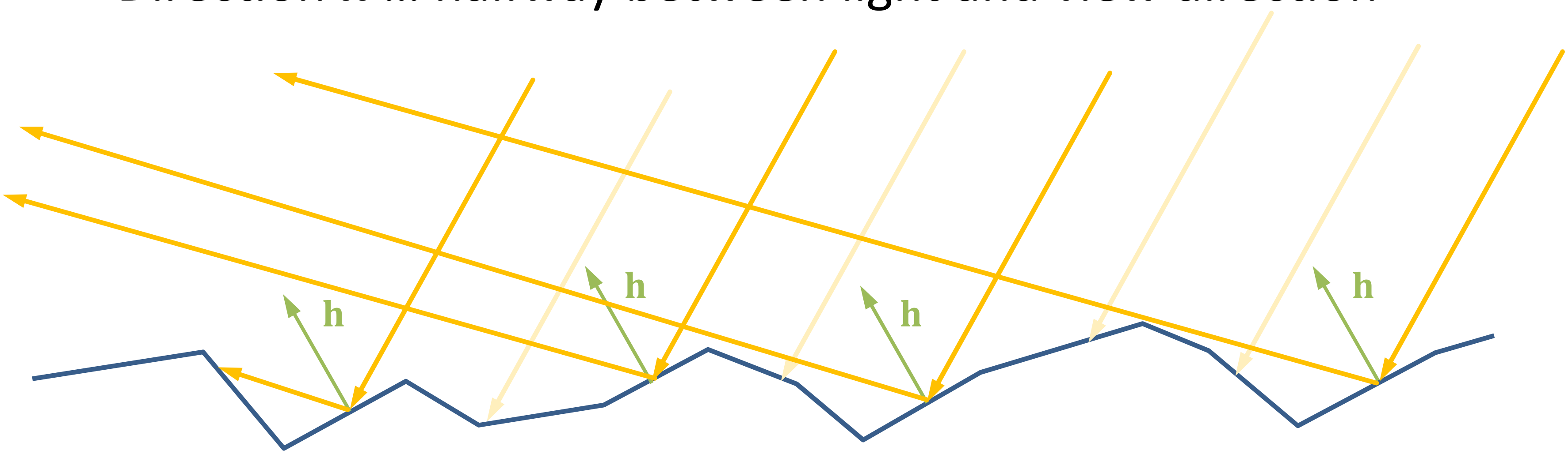
Microfacets

- Surface assumed to be made up of tiny mirrors
  – Lots of tiny mirrors
- Need to model the distribution of mirrors



Half-Vector

- Only mirrors oriented halfway between light and view direction reflect light into camera
- Direction **h** … halfway between light and view direction



Microfacet Distribution

- Fraction of microfacets oriented in direction $\mathbf{h}$
- Example: Beckmann Distribution

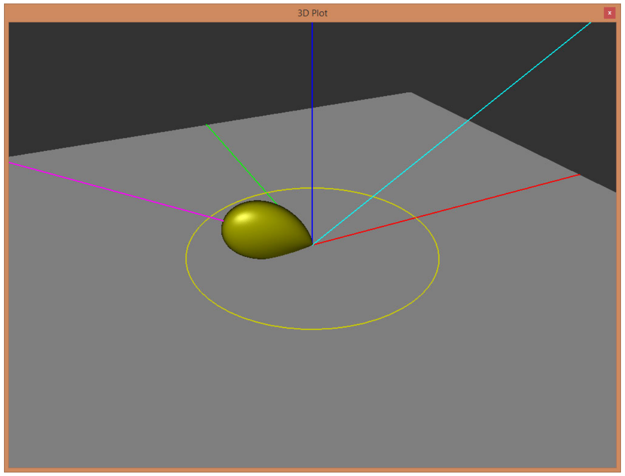
$$D(\mathbf{n},\mathbf{h},m)=\frac{1}{4m^2(\mathbf{n}\cdot\mathbf{h})^4}\exp\left(\frac{(\mathbf{n}\cdot\mathbf{h})^2-1}{m^2(\mathbf{n}\cdot\mathbf{h})^2}\right)$$

m Root mean squared slope of microfacets (corresponds to roughness)

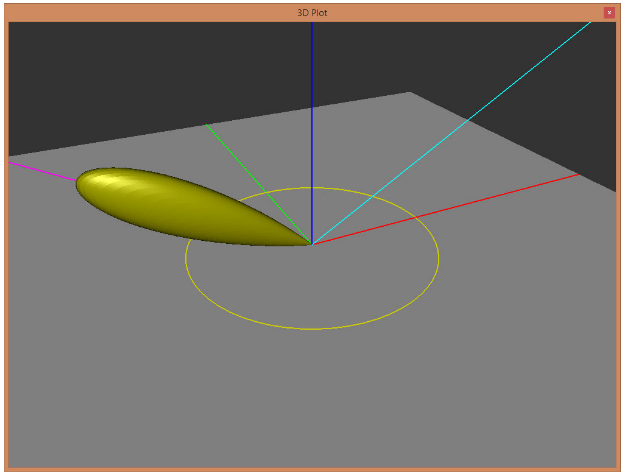
[Beckmann, Spizzichino 1963]

Beckmann Distribution

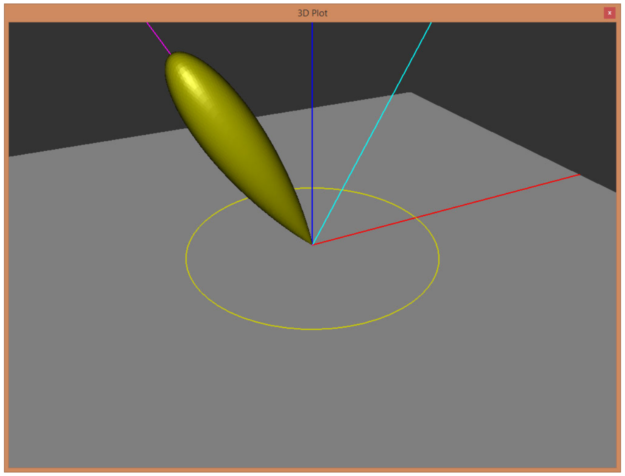
$\theta_i = 60°,\ m = 0.24$



$\theta_i = 60°,\ m = 0.069$



$\theta_i = 30°,\ m = 0.069$



Geometric Attenuation

- Accounts for shadowing
  - Light blocked from reaching microfacet by other microfacets
- Accounts for masking
  - Reflected light blocked from reaching camera by other microfacets
  - Example: Torrance-Sparrow model

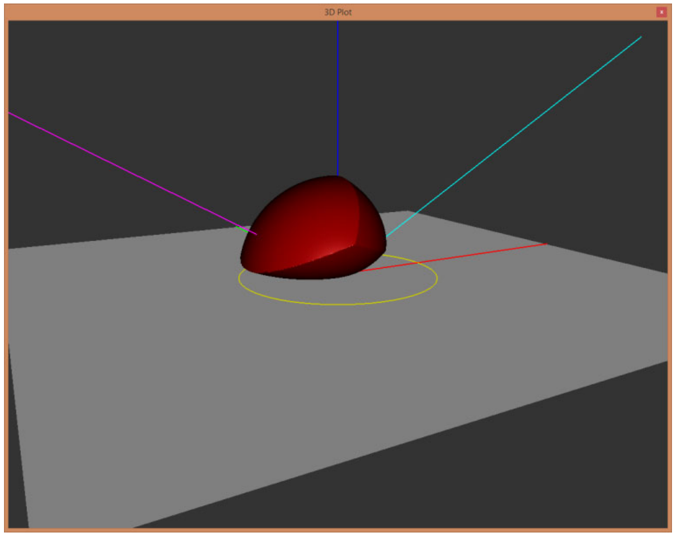
$\cos\theta_h = \mathbf{h}\cdot\mathbf{n}$
$\cos\theta_o = \mathbf{v}\cdot\mathbf{n}$
$\cos\alpha_h = \mathbf{v}\cdot\mathbf{h}$

$$G(\mathbf{l},\mathbf{v}) = \min\left(1, \frac{2\cos\theta_h\cos\theta_o}{\cos\alpha_h}, \frac{2\cos\theta_h\cos\theta_i}{\cos\alpha_h}\right)$$

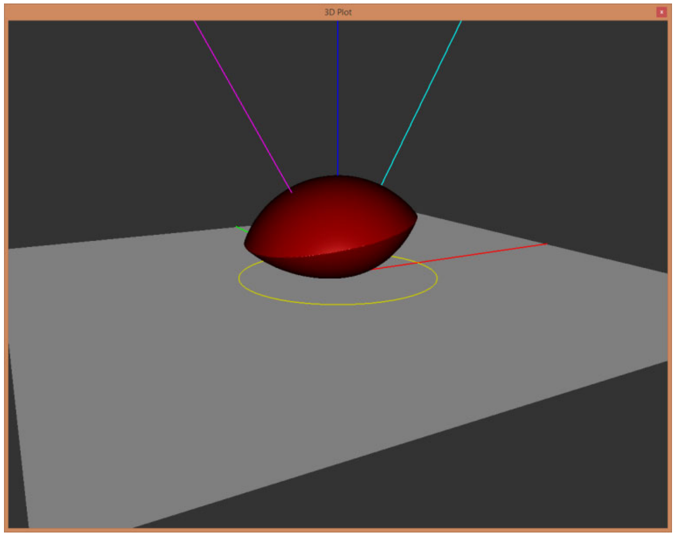
[Torrance, Sparrow 1967]

Torrance-Sparrow Geometric Attenuation

$\theta_i = 60°$



$\theta_i = 30°$



Cook-Torrance Model

Popular shading model which is putting all these components together

Microfacet distribution
(accounts for surface roughness)

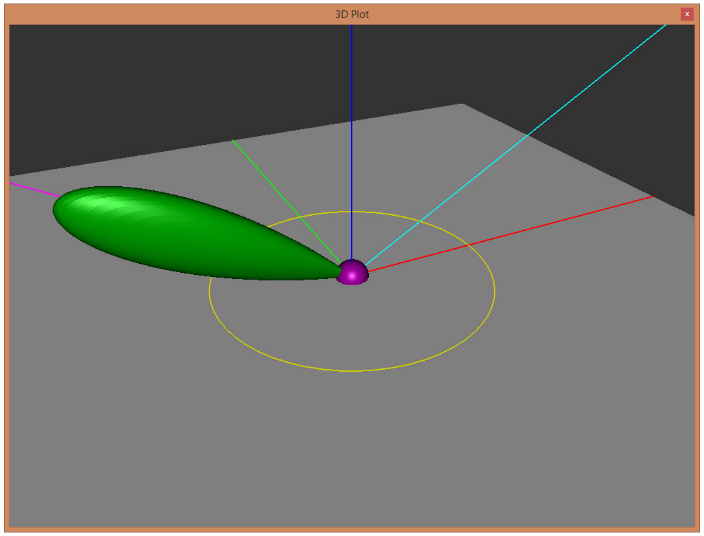
Fresnel
reflectance

Geometric attenuation
(amount of self-shadowing)

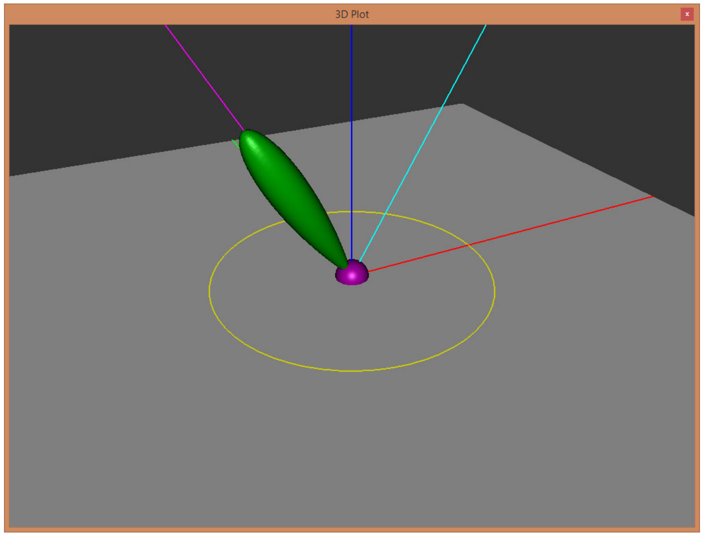
$$f_r(\mathbf{l},\mathbf{v}) = \frac{D(\mathbf{h})F(\mathbf{l},\mathbf{v})G(\mathbf{l},\mathbf{v})}{4(\mathbf{n}\cdot\mathbf{v})(\mathbf{n}\cdot\mathbf{l})}$$

Cook-Torrance BRDF 1

$\theta_i = 60°,\ m = 0.069$

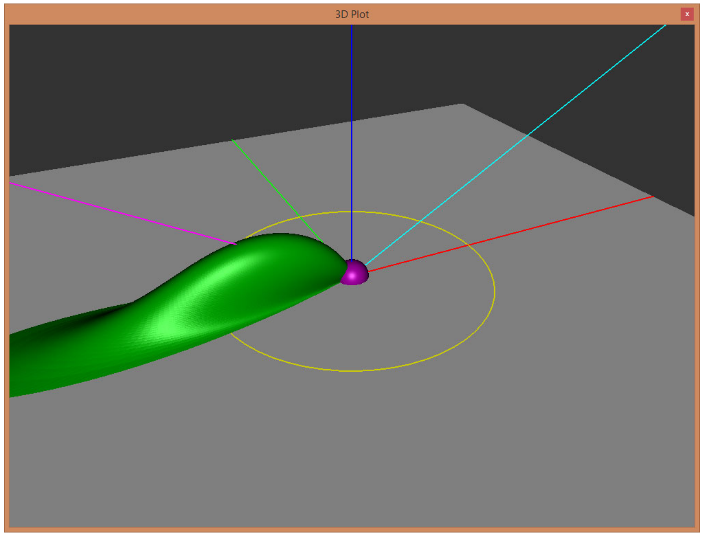


$\theta_i = 30°,\ m = 0.069$

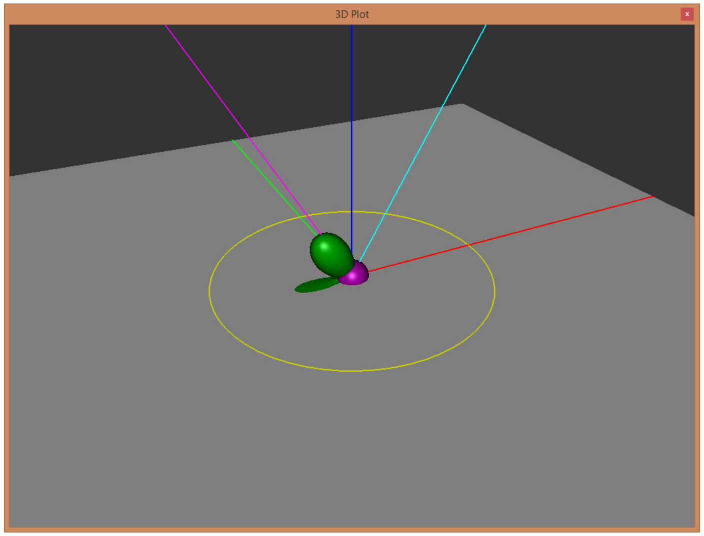


Cook-Torrance BRDF 2

$\theta_i = 60°,\ m = 0.24$



$\theta_i = 30°,\ m = 0.24$



Examples

Phong diffuse

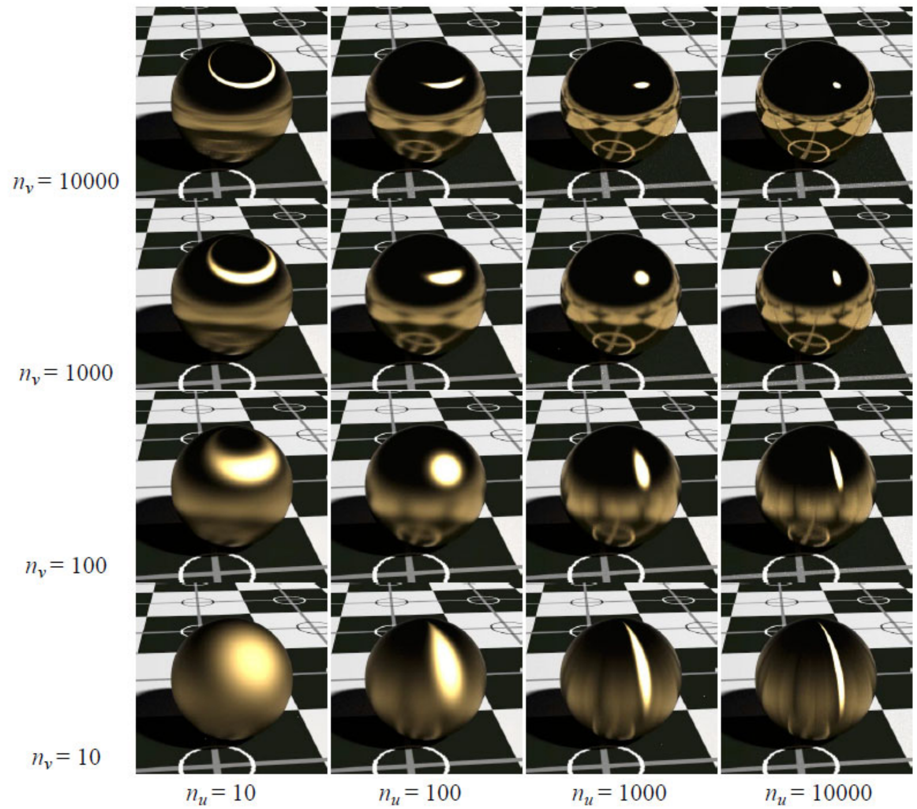
Phong specular

Cook-Torrance

https://www.youtube.com/watch?v=iVOfKlVtuVg

Anisotropic Reflection

- Viewing direction considered
  - Not just inclination
- Example
  - Brushed metal

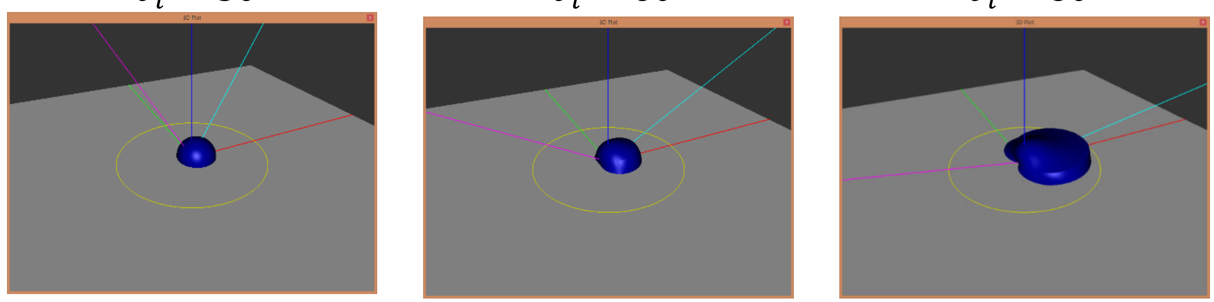


[Ashikhmin, Shirley 2000]

Retro-Reflection

- Deep cavities on surface reflect light back to light source
- Brighter at steep angles (near silhouette of object)
- Example: pottery

$\theta_i = 30°$ $\theta_i = 60°$ $\theta_i = 80°$



Photograph Lambertian Oren-Nayar

[Oren, Nayar 1994]

Implementation in GLSL

- BRDF:

$$f(\mathbf{l}, \mathbf{v}) = \frac{dL_o(\mathbf{v})}{dE_i(\mathbf{l})}$$

outgoing radiance

irradiance incident on surface

- Irradiance from light source:

$$E_L = \frac{I_L}{r^2}$$

light source intensity

squared distance to light source

- Radiance towards viewer:

incoming irradiance

$$L_o(\mathbf{v}) = \sum_{k=1}^{n} f(\mathbf{l}_k, \mathbf{v}) \circ E_{L_k} \cos\theta_{i_k}$$

sum up contributions of each light source

Example: Blinn-Phong Vertex Shader

```
#version 330

uniform mat4x4 PV;  // view projection matrix

layout(location = 0) in vec3 vertex_position;
layout(location = 1) in vec3 vertex_normal;

out vec3 p;
out vec3 normal;

void main()
{
    gl_Position = PV * vec4(vertex_position, 1.0f);
    p = vertex_position;
    normal = vertex_normal;
}
```

Example: Phong Fragment Shader

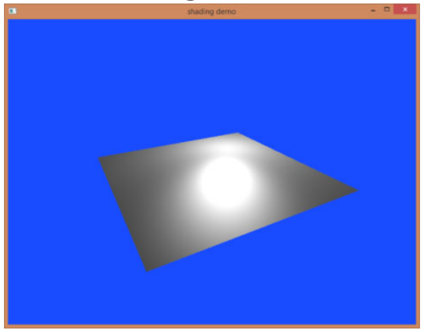
```
#version 330

uniform vec3 camera_position;
uniform vec3 light_direction;
uniform vec3 B_L;
uniform vec3 c_d;
uniform vec3 c_s;
uniform float m;
in vec3 p;
in vec3 normal;

layout(location=0) out vec4 fragment_color;

void main()
{
      vec3 n = normalize(normal);
      vec3 v = normalize(camera_position - p);
      vec3 l =-light_direction;
      vec3 r = 2.0f * n * dot(n, l) - l;

      float lambert = max(dot(n, l), 0.0f);
      float specular = pow(max(dot(v, r), 0.0f), m);

      fragment_color.rgb = (c_d *lambert + c_s * specular) * B_L;
      fragment_color.a = 1.0f;
}
```

Example: Blinn-Phong Fragment Shader

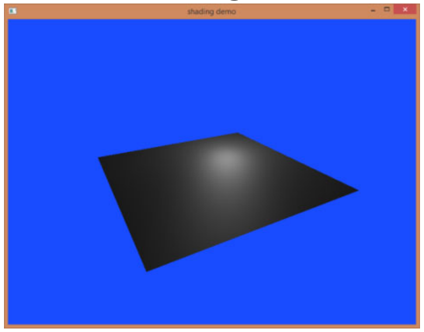
```glsl
#version 330

const float pi = 3.14159265358979f;

uniform vec3 camera_position;
uniform vec3 light_direction;
uniform vec3 I_L;
uniform vec3 c_d;
uniform vec3 c_s;
uniform float m;

in vec3 p;
in vec3 normal;

layout(location=0) out vec4 fragment_color;

void main()
{
      vec3 n = normalize(normal);  // renormalize interpolated normal
      vec3 v = normalize(camera_position – p);
      vec3 l =-light_direction;
      vec3 h = normalize(v + l);
      float lambert = max(dot(n, l), 0.0f);
      float specular = (m + 8) / (8 * pi) * pow(max(dot(n, h), 0.0f), m);
      fragment_color.rgb = (c_d / pi + c_s * specular) * lambert * I_L;
      fragment_color.a = 1.0f;
}
```

Results

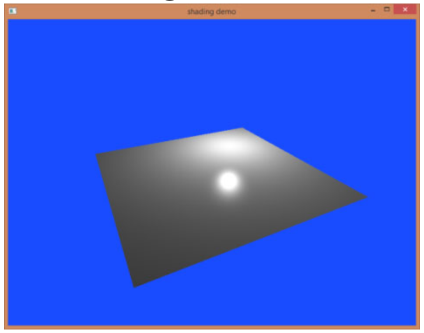
Phong, $m = 8$



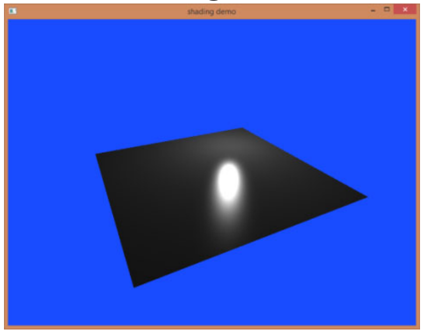
Blinn-Phong, $m = 8$



Phong, $m = 100$



Blinn-Phong, $m = 100$



References

[Akenine-Möller et al. 2008] Tomas Akenine-Möller, Eric Haines, Naty Hoffman (2008). Real-Time Rendering. 3rd ed. AK Peters.

[Lengyel 2004] Eric Lengyel (2004). Mathematics for 3D Game Programming & Computer Graphics. 2nd ed. Charles River Media.

[Phong 1975] Bui Tuong Phong (1975). "Illumination for computer generated pictures". *Communications of the ACM*, vol. 18, no. 5.

[Blinn 1977] James F. Blinn (1977). "Models of light reflection for computer synthesized pictures". Proc. 4th annual conference on computer graphics and interactive techniques: 192–198.

[Schlick 1994] Christophe Schlick. (1994). "An Inexpensive BRDF Model for Physically-based Rendering". Computer Graphics Forum, vol. 13, no. 3: 233–246.

[Torrance, Sparrow 1967] K. E. Torrance, E. M. Sparrow (1967). "Theory for Off-specular Reflection From Roughened Surfaces". *Journal of the Optical Society of America*, vol. 57, no. 9: 32–41.

[Beckmann, Spizzichino 1963] Petr Beckmann, André Spizzichino (1963). The scattering of electromagnetic waves from rough surfaces, Pergamon Press.

[Ashikhmin, Shirley 2000] Michael Ashikhmin, Peter Shirley (2000). "An Anisotropic Phong BRDF Model". Journal of Graphics Tools, vol. 7, no. 4: 61–68.

[Oren, Nayar 1994] Michael Oren, Shree K. Nayar (1994). "Generalization of Lambert's reflectance model". Proc. 21st annual conference on computer graphics and interactive techniques: 239–246.