CS195V Week 8

Materials and BRDFs

Overview

- Materials are a big part of rendering
  - Even if you have a great rendering algorithm, you need good material representations to create a good result
- Today we'll be discussing common BRDF approximations and specific applications
- After spring break, we will try to shift more towards compute-related topics and examples
  - If you have a graphics-related case study to present, next week is a good time to do it...

Ok Maybe Not

- The first CUDA project will probably involve rendering, so it might be helpful to review some rendering concepts and seed some ideas

Our Friend, the BRDF

- Let us remind you of the old BRDF equation:

$$f_r(\omega_i, \omega_o) = \frac{dL_r(\omega_o)}{dE_i(\omega_i)} = \frac{dL_r(\omega_o)}{L_i(\omega_i)\cos\theta_i \, d\omega_i}$$

- Given some incident and exitant directions, find the relationship between the incoming and outgoing light (radiance)
  - The ratio of the reflected radiation to the incident radation is called the *albedo*
- Let's look at some common models to give you an idea of what's available

http://www.cs.ucl.ac.uk/staff/j.kautz/GameCourse/04_PointLights.pdf

Good Ol' Phong

- Ambient, Diffuse, Specular
  - Not really physically based, but some relation to physical parameters
    - kd can be thought of as diffuse albedo
    - ks is related to the Fresnel reflectance along the normal
    - Fresnel Reflectance is derived from Fresnel equations with use refractive indices, etc.
- For realism, must be normalized
  - kd + ks <= 1
  - Ensures energy conservation

Specular Term

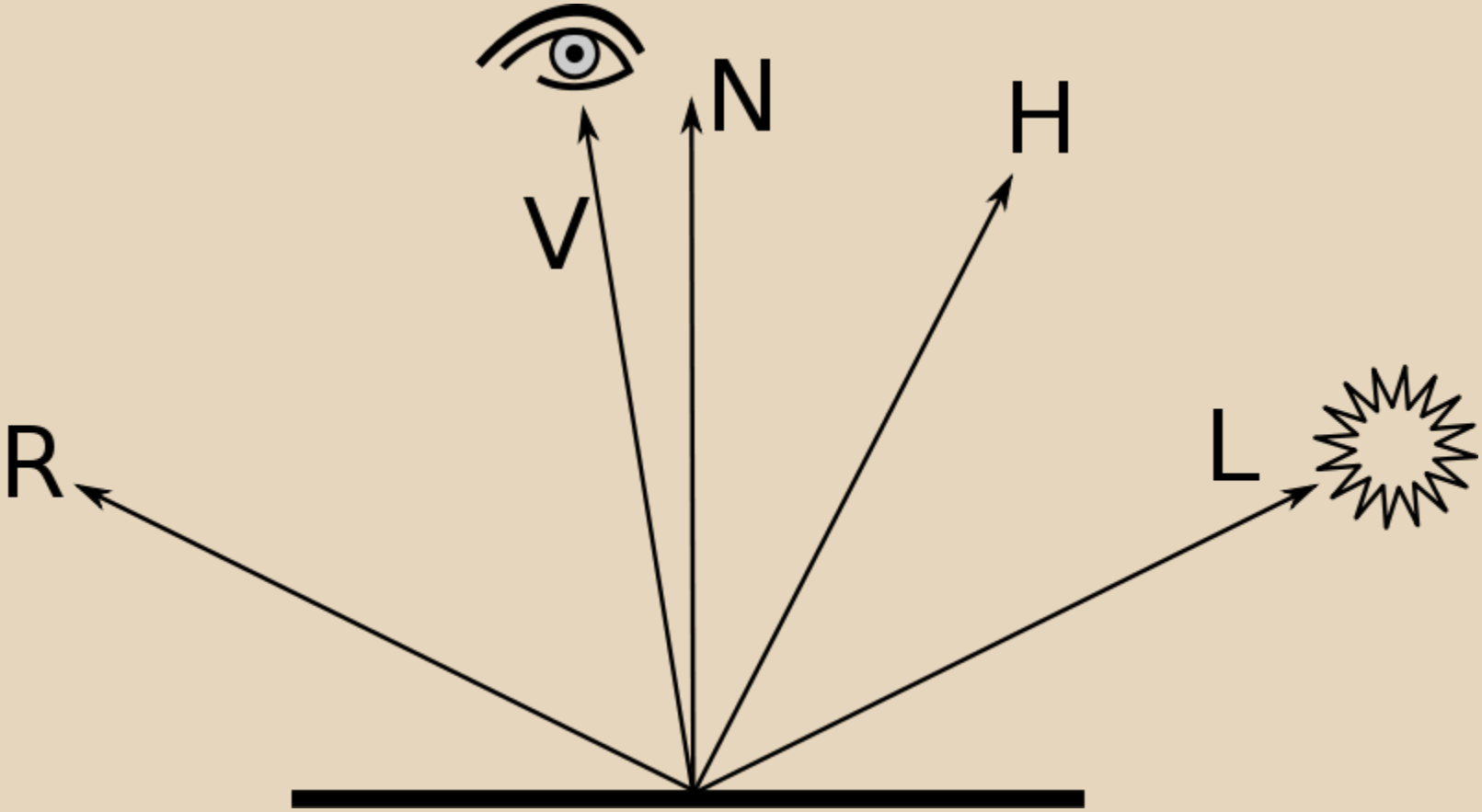
- Consider the specular term, what does it mean?

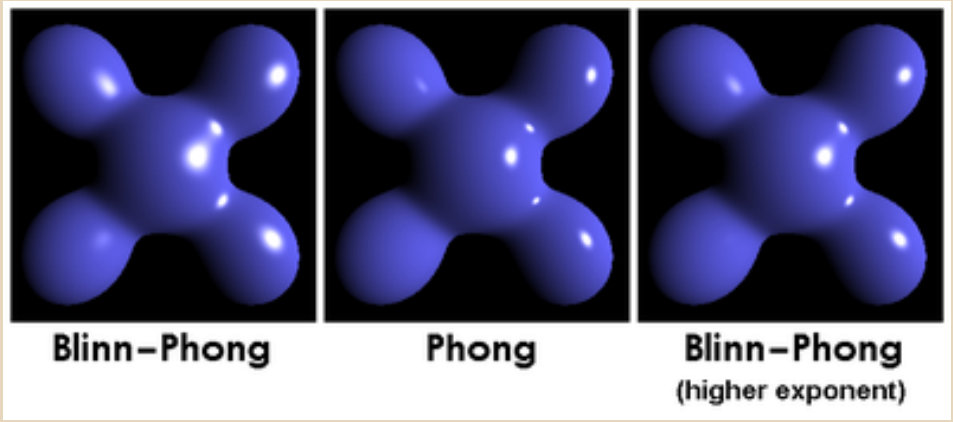
$$I_p = k_a i_a + \sum_{\text{m} \in \text{lights}} (k_d(\hat{L}_m \cdot \hat{N}) i_{m,d} + k_s(\hat{R}_m \cdot \hat{V})^{\alpha} i_{m,s}).$$

- What happens when the dot product is negative?
- What does the specular exponent represent anyways?

Blinn-Phong

- The main modification is to the specular term to prevent this strange behavior
- Instead of comparing the reflected light vector and the view vector, it calculates the half angle vector between the light and the eye and compares that to the surface normal


N
H
V
R
L


Blinn–Phong
Phong
Blinn–Phong
(higher exponent)

ks and Fresnel Reflectance

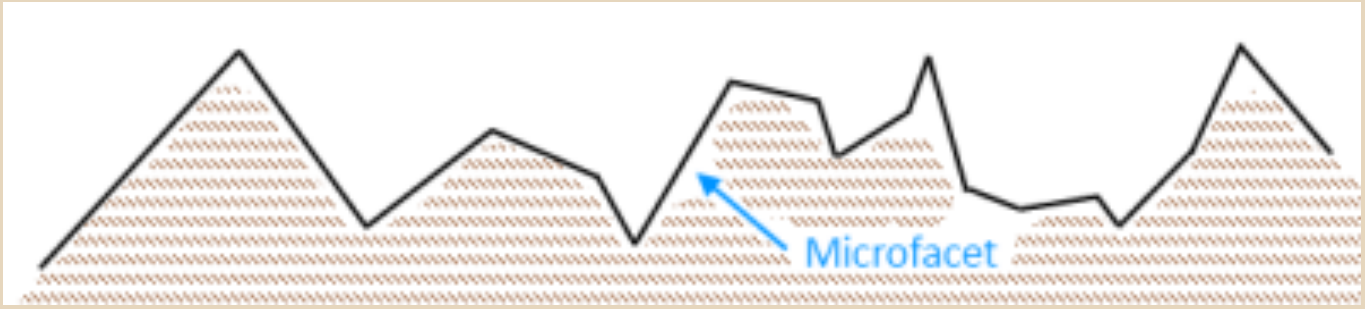
- For some materials, it may make sense for ks to vary
  - That is, the material becomes very specular at an oblique angle
- This can be particularly important for dieletric materials
- However, this means you need some function for ks, which would require evaluating Fresnel equations or some other approximation

Beyond...

- Most of what you've implemented is probably some variation on Phong or Blinn-Phong
  - Pretty good for most materials
- Some specific phenomena and materials might benefit from more complex BRDFs

Smooth vs Microfacet

- When we think of reflecting light off of a surface, we treat the geometry as a smooth surface
- The following models instead treat the surface as a set of microfacets, small irregularities in the geometry which affect how it reflects light



Microfacets

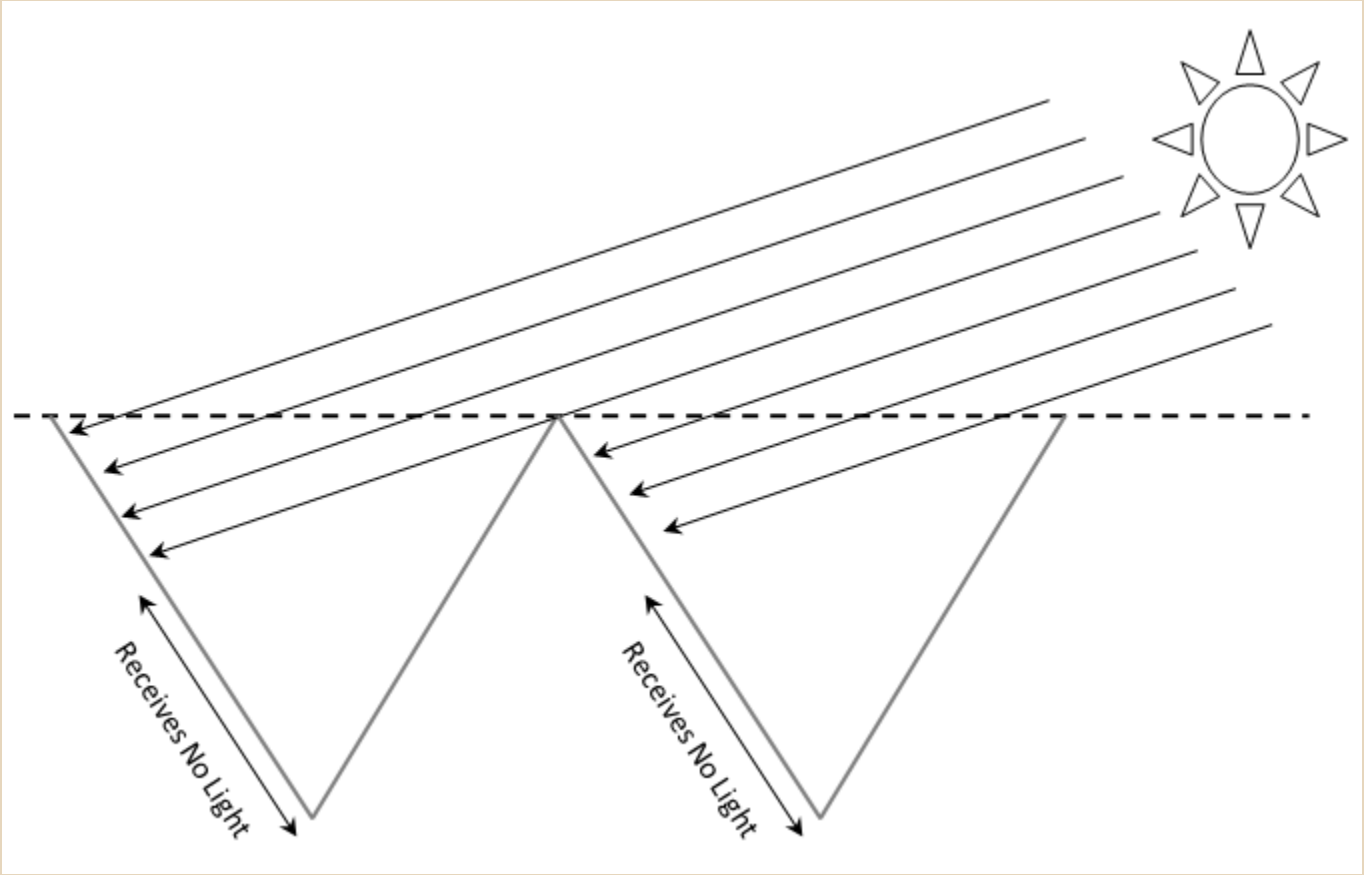
- Say that there is some random distribution of directions that these microfacets are pointing
- This leads to a number of consequences, each modeled with a different term
  - Roughness
  - Shadowing/Masking
    - 2 phenomena, one term

Roughness

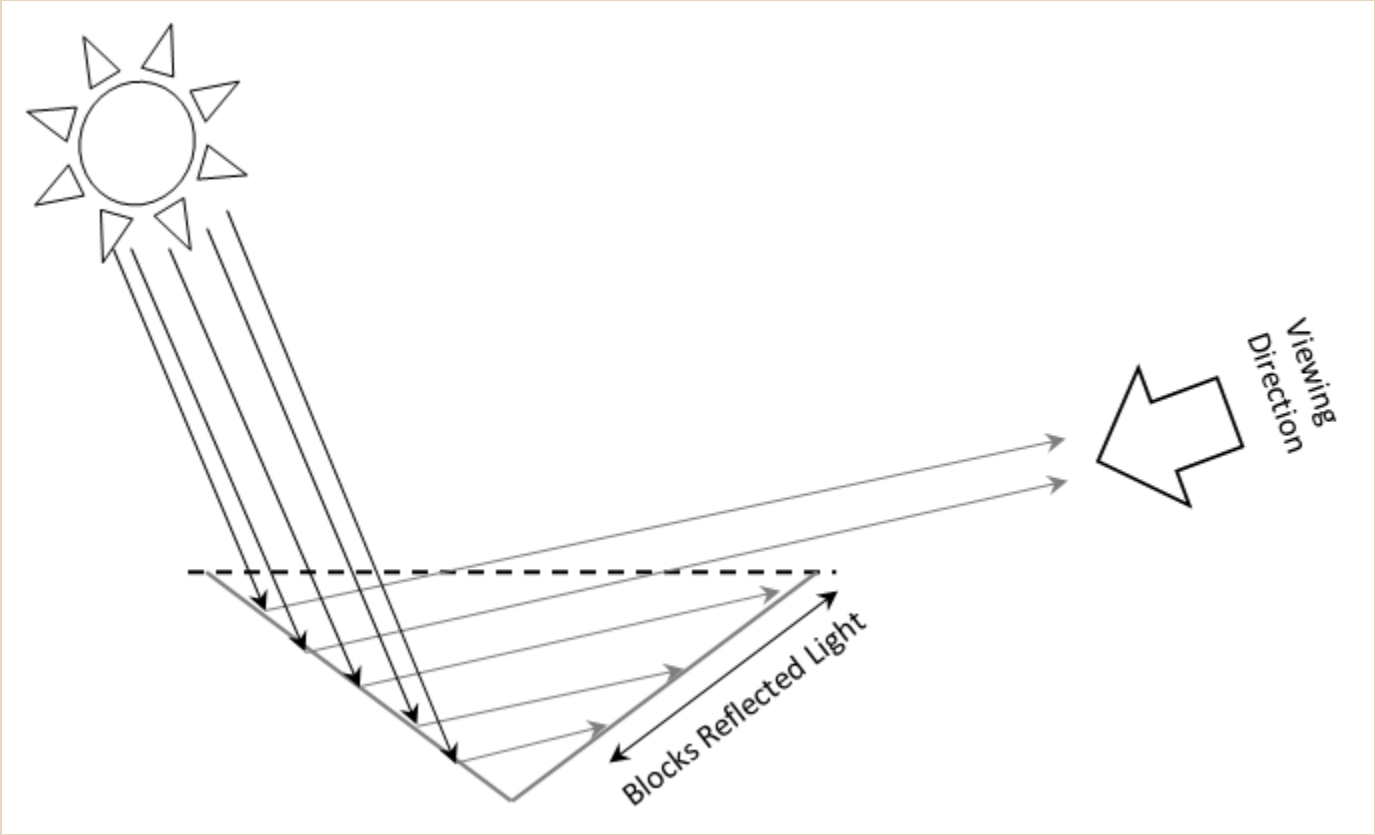
- With this surface of random microfacets, how much light will exit in a given direction?
  - Find out what percentage of microfacets mirror light to go in the correct direction
- The distribution of microfacet normals defined as roughness
  - Usually some variation on the normal distribution
  - A smoother surface has more microfacets pointed in similar or same direction as surface normal

Shadowing/Masking

- For a given microfacet, some of the incident light may be occluded by other microfacets
  - This is called shadowing
- Also, even if light reaches the facet, the reflected light may be blocked by another
  - This is masking



Shadowing



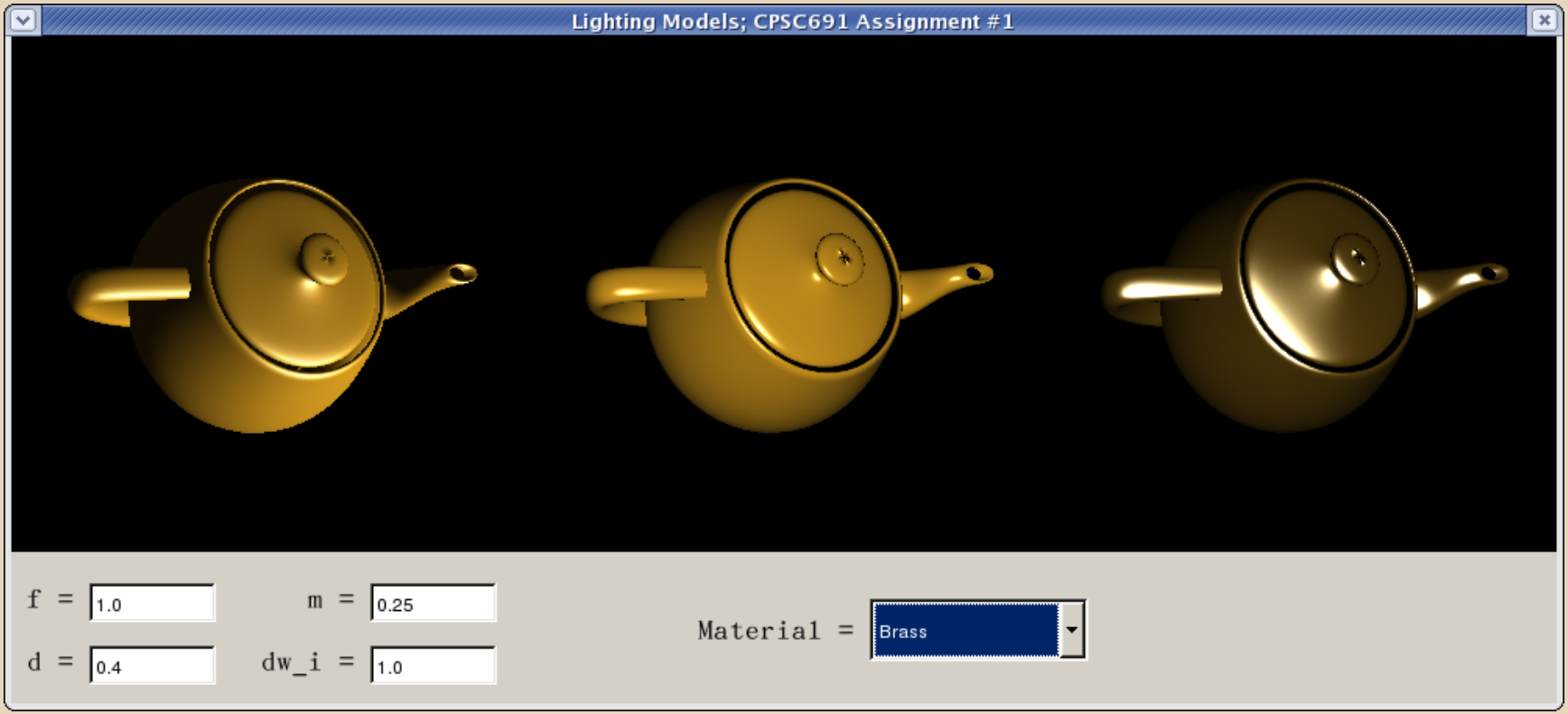
Masking

Geometry Term

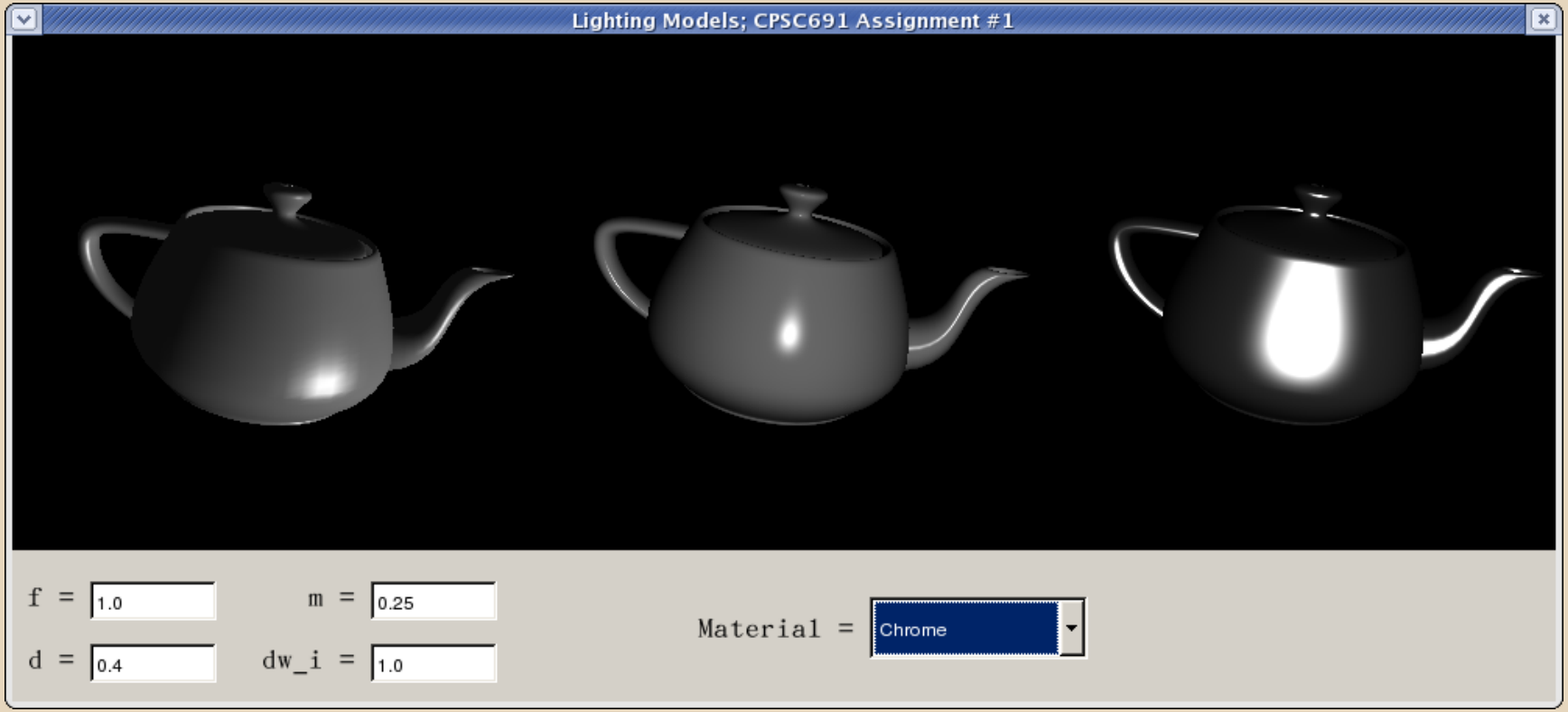
- Practically, shadowing and masking are accounted for by a geometry term
  - Varies between 0 and 1

This equation is perhaps the most commonly used (derived from some geometric analysis)

$$G = \min\left(1, \frac{2(H \cdot N)(E \cdot N)}{E \cdot H}, \frac{2(H \cdot N)(L \cdot N)}{E \cdot H}\right)$$



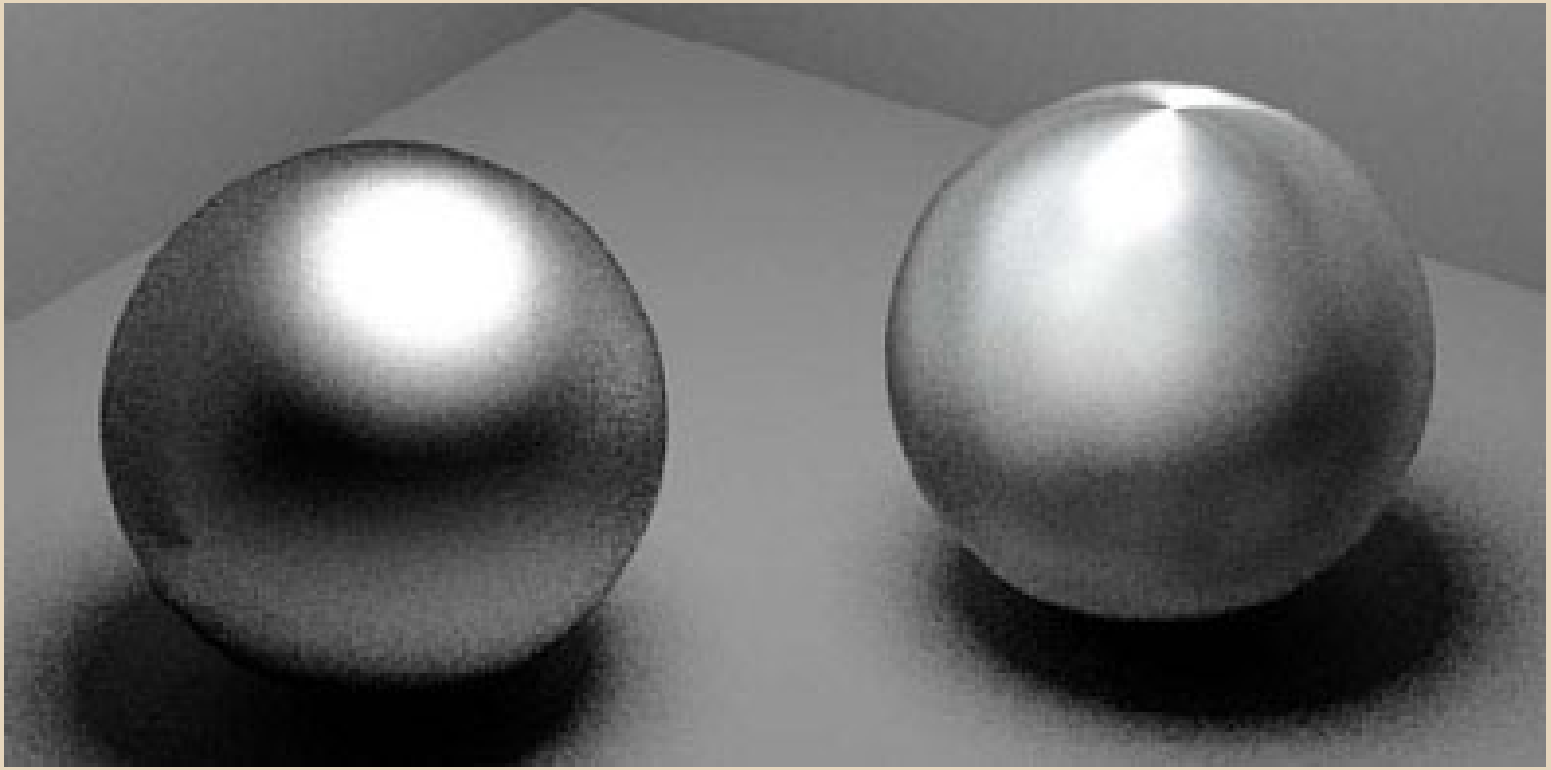
Comparison - Phong, Blinn-Phong, Cook-Torrance



More comparison

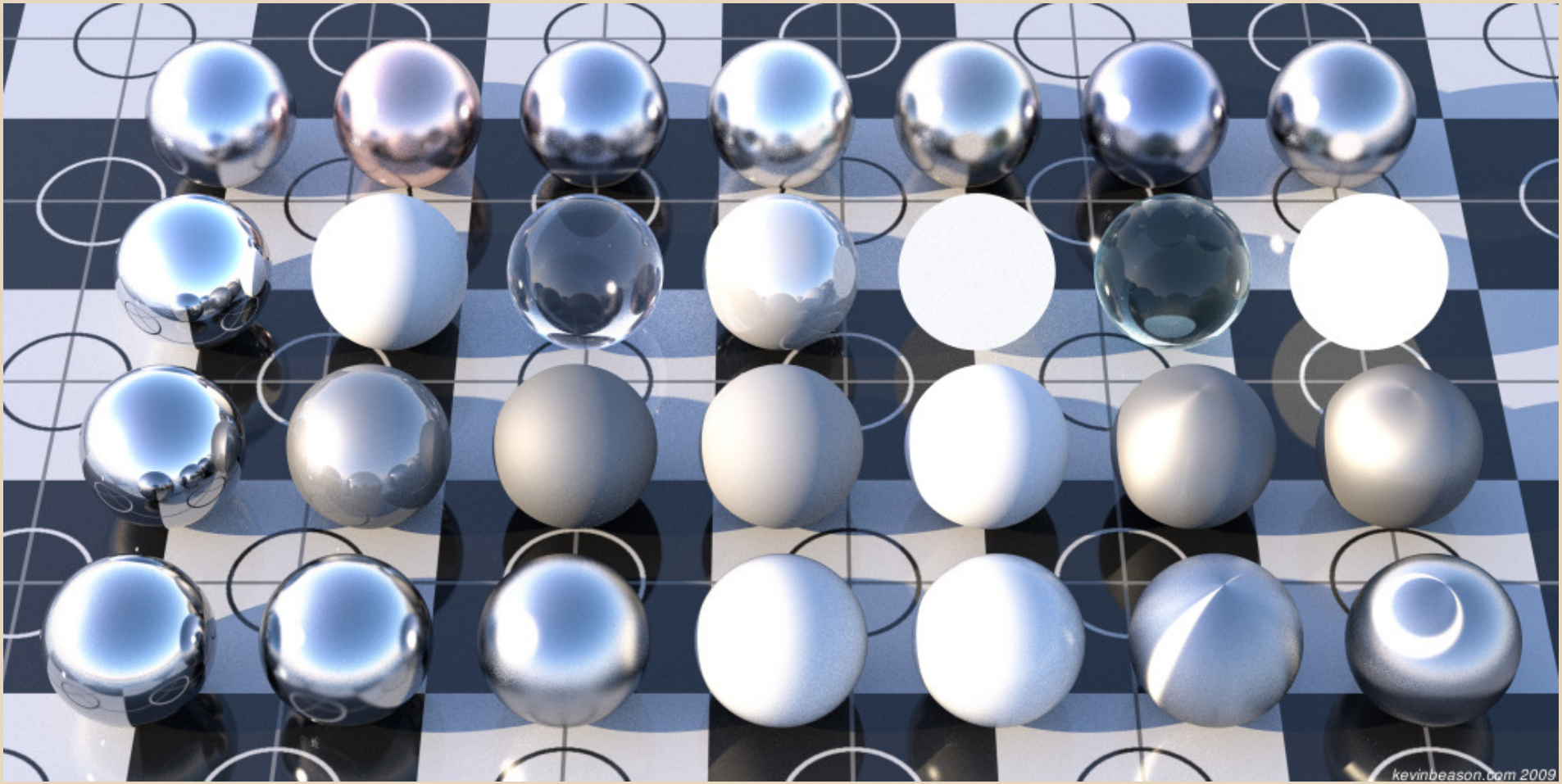
Anisotropy

- Reflections can be anisotropic, that is, preferring a particular direction
- Brushed metal for example, or other surfaces with regular streaks or irregularities
  - If these patterns are in a particular direction, will affect how light reflects off of them
- Maybe those funky hairy Christmas ornaments...



Isotropic vs Anisotropic

LuxRender!



Different Types of Anisotropic Reflections (Different BRDFs)

Hairy Christmas Ornament

Banks BRDF

- Modified Blinn-Phong to account for anisotropic reflection
- Using knowledge of the material's grain, modify calcuations for the diffuse and specular terms (T is along the grain)

$$\textbf{Diffuse} = k_d * \left(\sqrt{1 - (L \bullet T)^2} \right)^p$$

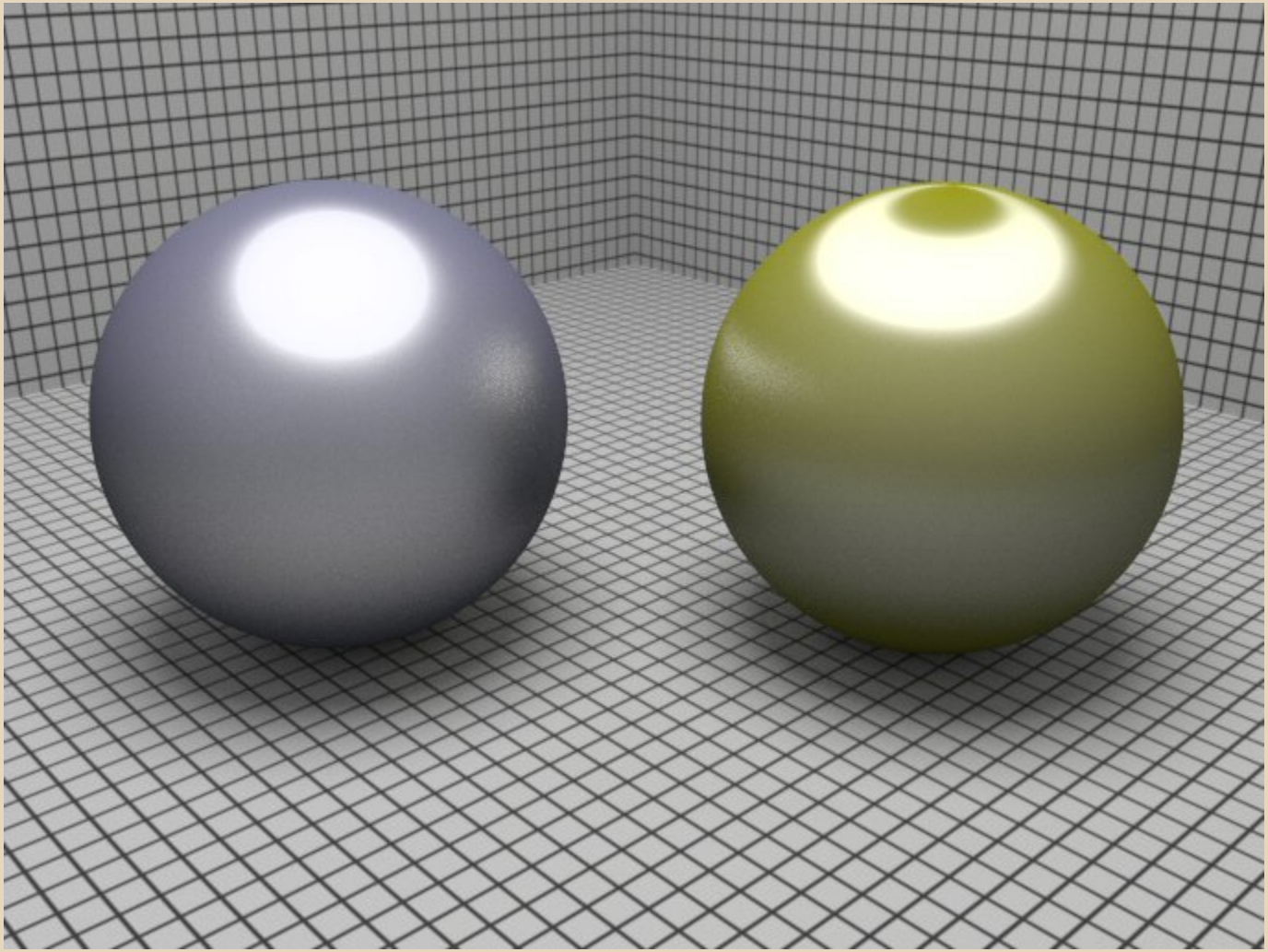
$$\textbf{Specular} = k_s * \left(\sqrt{1 - (L \bullet T)^2} \sqrt{1 - (V \bullet T)^2} - (L \bullet T)(V \bullet T) \right)^q$$

Ward Anisotropic BRDF

- Designed to give fine control over anisotropic parameters

$$\rho_{bd}(\theta_i, \phi_i, \theta_r, \phi_r) = \frac{\rho_d}{\pi} + \rho_s \cdot \frac{1}{\sqrt{\cos\theta_i \cos\theta_r}} \cdot \frac{1}{4\pi\alpha_x\alpha_y} \cdot \exp\left[-2\frac{\left[\frac{H \bullet T}{\alpha_x}\right]^2 + \left[\frac{H \bullet B}{\alpha_y}\right]^2}{1 + H \bullet N}\right]$$

- ax and ay are the "lobe size" in the principal directions of anisotropy
  - If they are equal, reflection is isotropic
- But no shadowing or masking!



Ward Example

Ashikhmin-Shirley BRDF

- One of the few BRDFs which doesn't use Lambert's cosine law
- Instead, trades off directly between diffuse and specular terms based on the viewing angle
- Based off physical observations
  - Conservation of energy, etc.
- No shadowing or masking

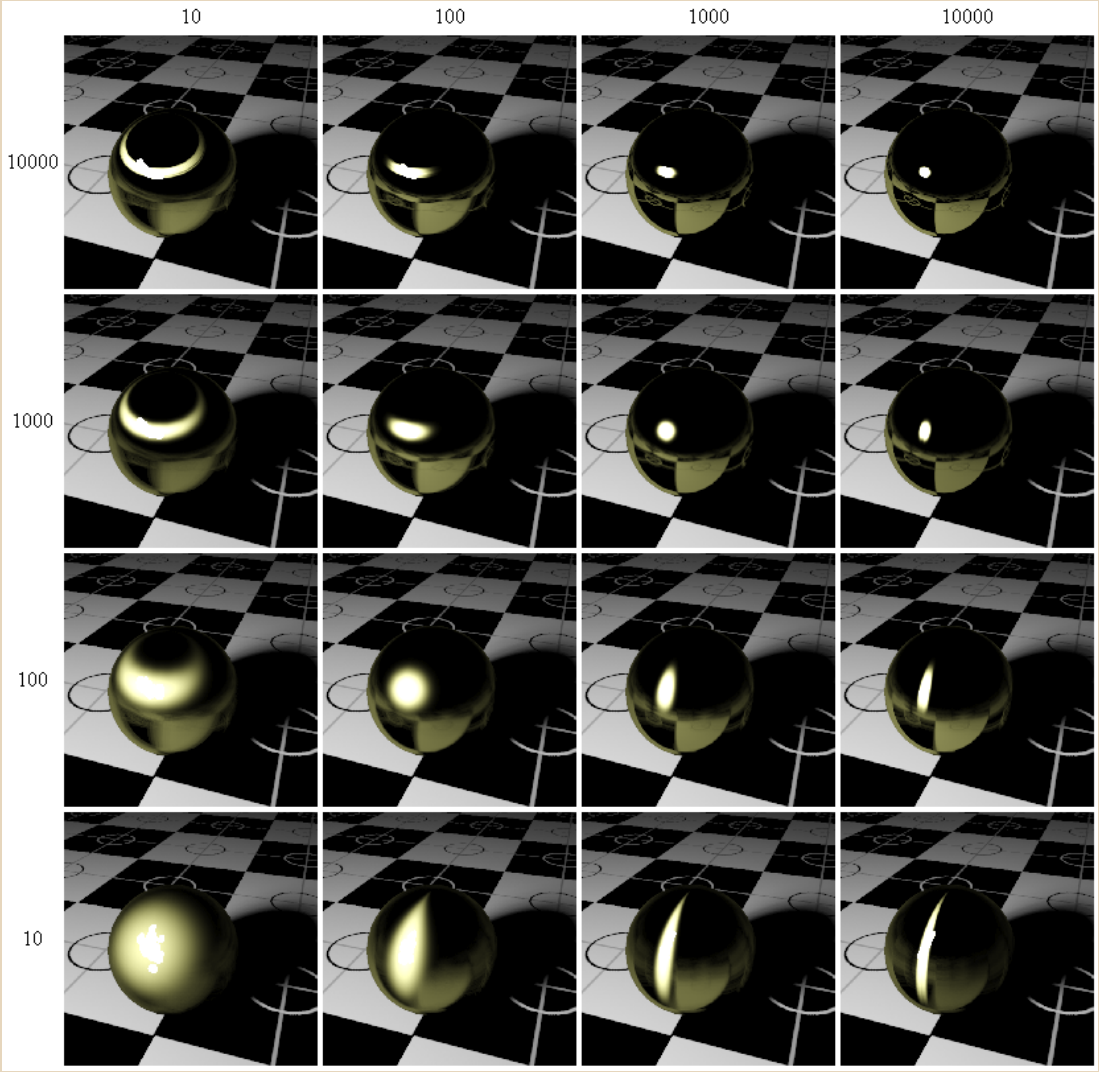
Math

$$\rho(Light, View) = \rho_d(Light, View) + \rho_s(Light, View)$$

$$\rho_d = \frac{28 \times R_d}{23 \times \pi} \times (1 - R_s) \times \left(1 - \left(1 - \frac{Normal \bullet Light}{2}\right)^5\right) \times \left(1 - \left(1 - \frac{Normal \bullet View}{2}\right)^5\right)$$

$$\rho_s = \frac{\sqrt{(n_u + 1) \times (n_v + 1)} \times (Normal \bullet Half)^{n_u \times \cos^2\phi + n_v \times \sin^2\phi}}{8 \times \pi \times (Half \bullet Light) \times max((Normal \bullet Light), (Normal \bullet View))} \times F(Half \bullet Light)$$

$$F(x) = R_s + (1 - R_s) \times (1 - x)^5$$



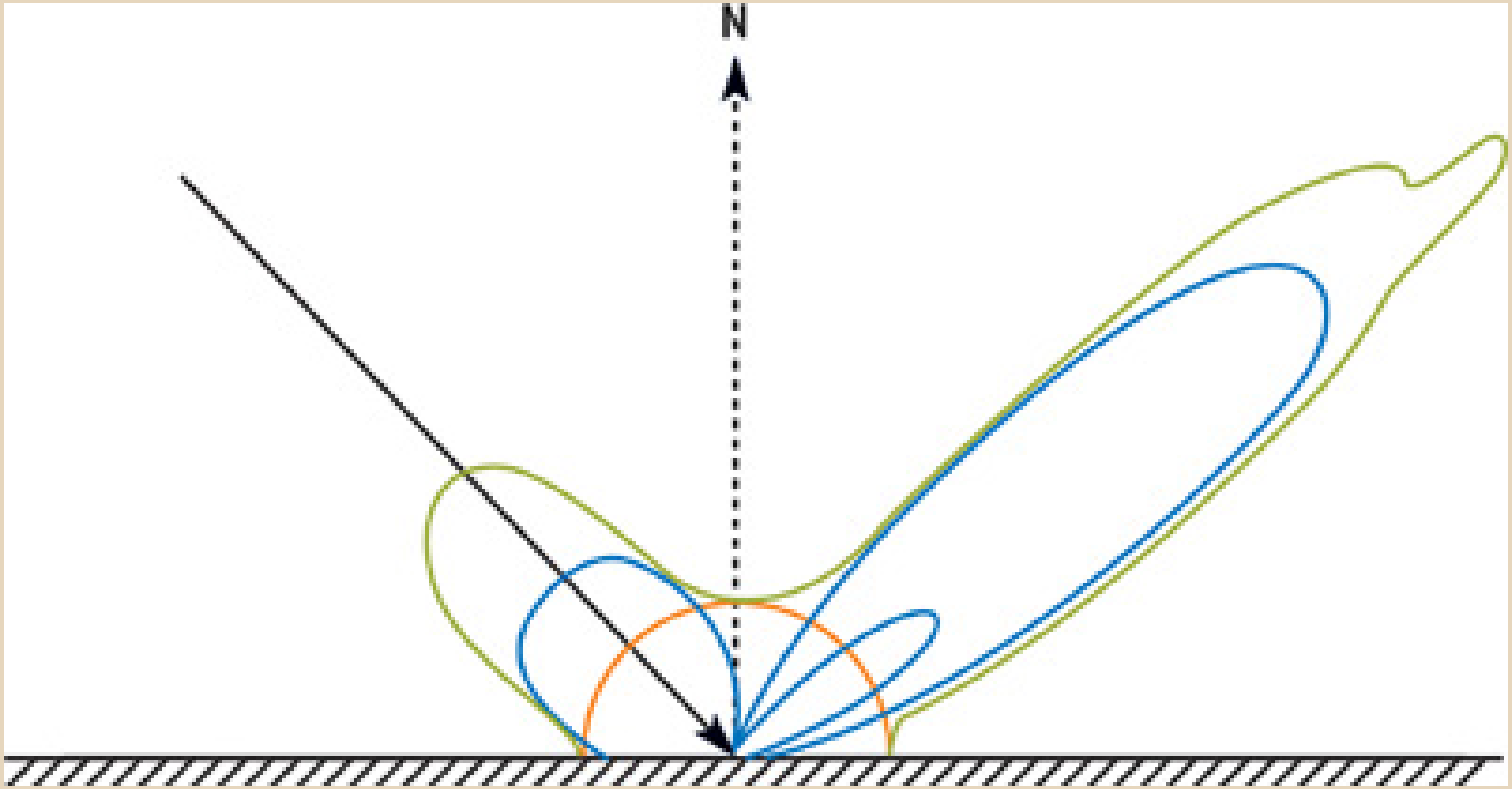
Fine control of anisotropic reflection

Lafortune BRDF

- Modification on Phong (again) to give fine control over anisotropic reflections
- Specular term is multiplied by

$$s(\omega_i, \omega_r) = \left(C_x \omega_{i,x} \omega_{r,x} + C_y \omega_{i,y} \omega_{r,y} + C_z \omega_{i,z} \omega_{r,z}\right)^n .$$

- This sum of "lobes" allows for fine control of BRDF



Visualization of Lobes
Orange - Diffuse, Blue - Lobes, Green - BRDF

Lafortune Example

Finally, Oren-Nayar

- Microfacet BRDF, but uses lambertian scattering rather than mirror scattering off microfacets
- Very useful for diffuse materials

In Reality...

- Most games use Blinn-Phong because of its computational ease
- Other applications like Blender and Renderman will give you multiple choices for BRDFs
- And of course, high end renderers support as much as they can

Use on the GPU

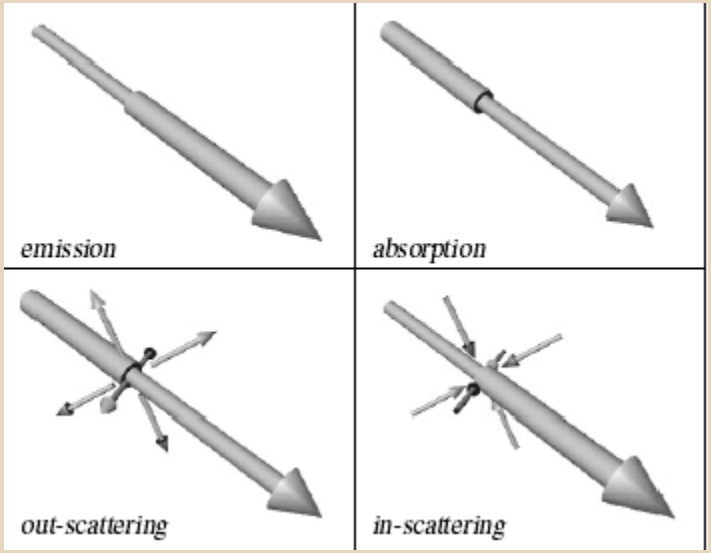
- All of the BRDFs are well and good, but how do we use them in GPU applications?
- SBRDF: Spatial Bi-Directional Reflectance Distribution Function
  - BRDF that varies over space, say, texture space
  - At each spatial point, define the parameters of the BRDF
  - Easy to evaluate as long as you have texture coordinates

More on Materials

- Using these BRDFs will allow you to render a wide range of reflective materials
- Transmission can be added with some hassle
- What are things you think of as difficult to render well?
  - Fire, water, smoke, perhaps?
  - Subsurface effects like skin?

Participating Media

- Sometimes light scatters as it passes through a medium
- These phenomena can be difficult to render using traditional techniques
  - Now dealing more with volumes as opposed to surfaces



Complex Media

- There is still research being done on these rendering topics
- Instead of further boring you with inevitable math and such, let's look at some specific GPU-based examples of specific materials

Fluids/Water

- We discussed a little bit last lecture
- Overall, if we assume a mostly transmissive fluid, we can model as a combination of reflection, refraction, and diffuse color

Water by Insomniac Games

- Perhaps best known recently for the Resistance series
- Cube map for cheap environment reflections
- Downsample the current frame buffer for get scene-specific reflections/refractions
  - Perturb and bleed depending on waves
- Use depth data to apply fogging to simulate murky water
- Particle effects for foam!

Video

Fire/Smoke

- These are volume effects, and thus we need to consider the scattering (and in fire's case, emission) of light in the volume
- As of now, most games seem to still use precomputed sprites or particles to do these materials
  - Somehow hack in transmission using alpha blending
- Still can look pretty good

Far Cry 2

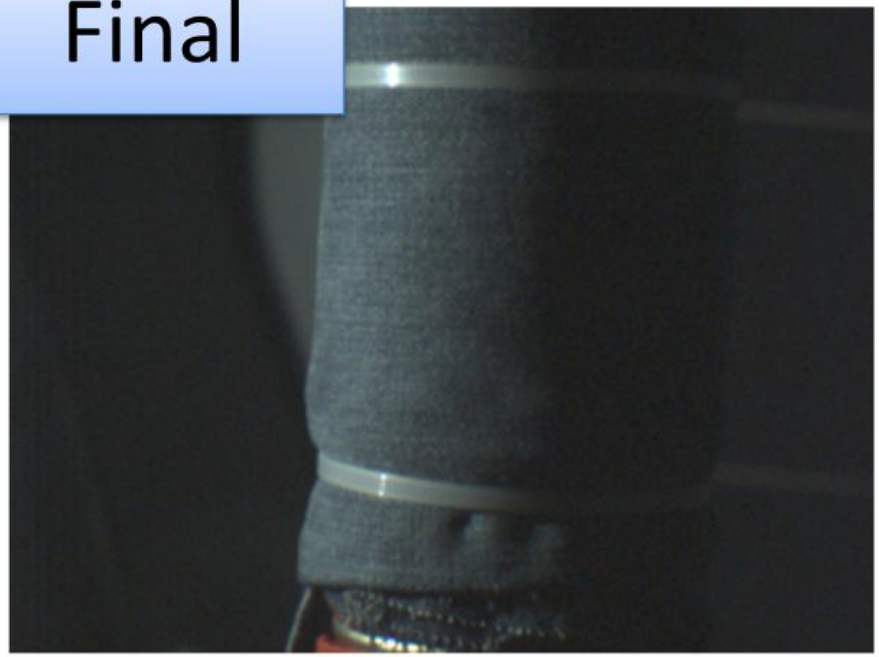
Alone in the Dark

Cloth

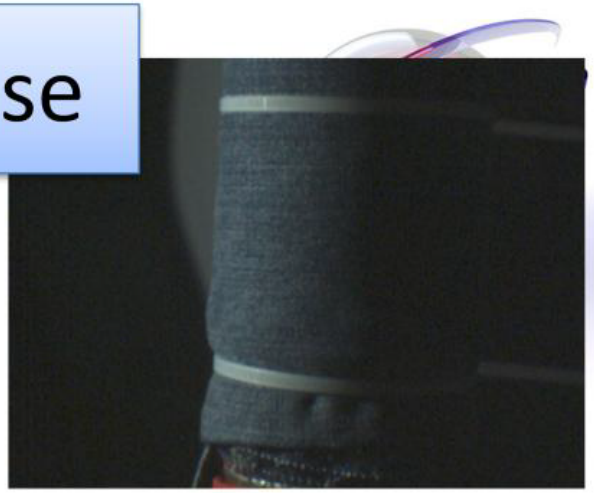
- We want some of the anisotropy from the hairy BRDFs, plus some subsurface effects
- However, most games still hack it a bit
- You would think that it's mostly diffuse, but even cloth has a significant specular component

Separated

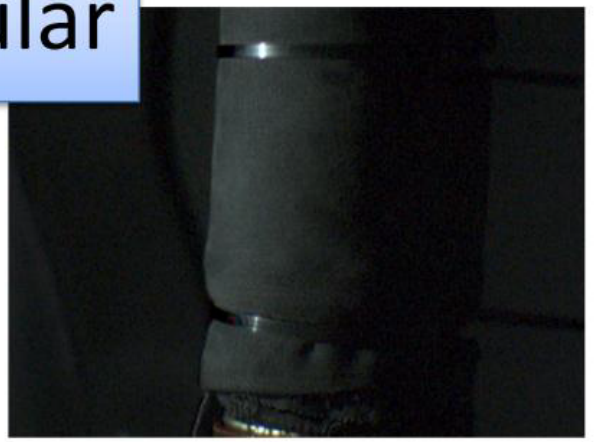
Final



Diffuse



Specular



Subsurface Effects

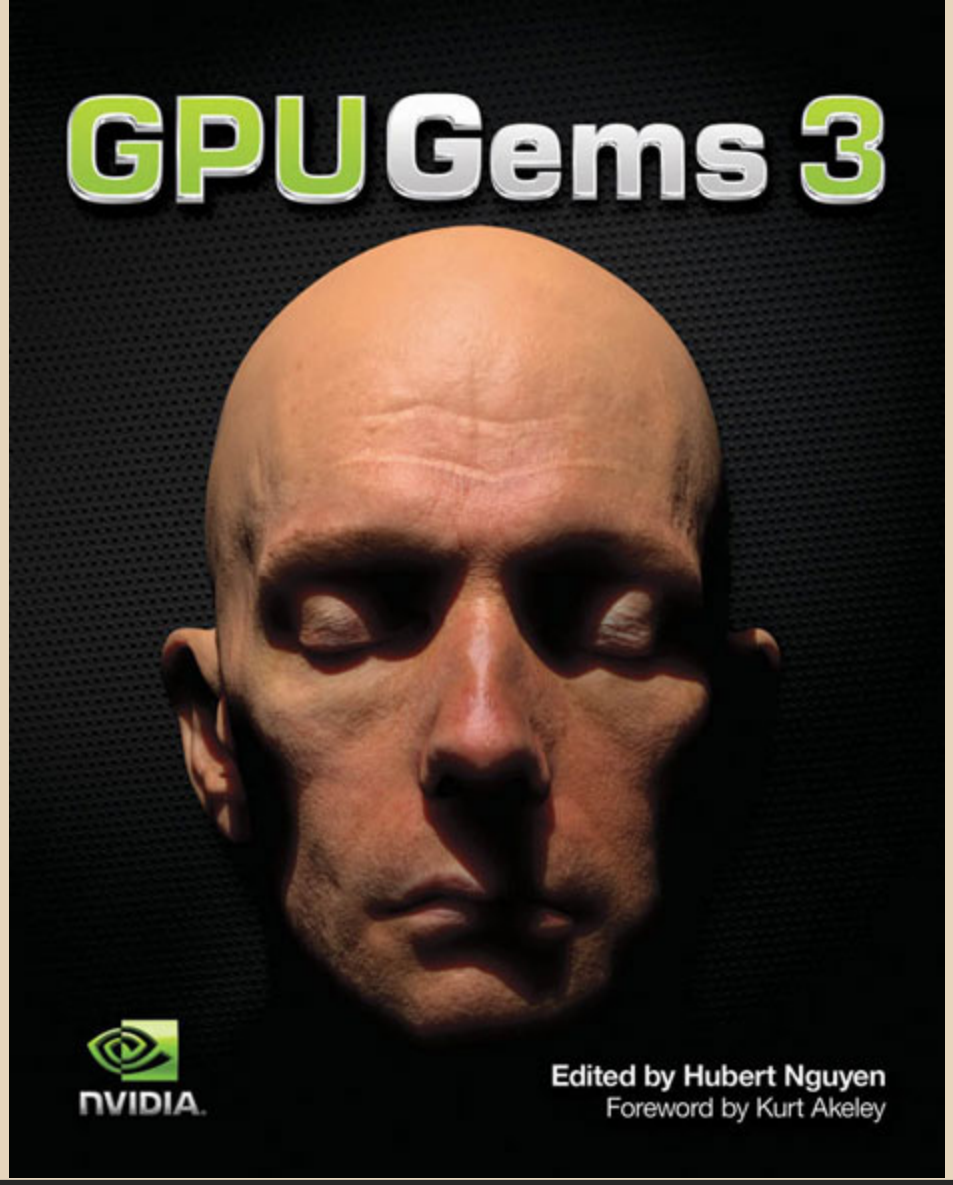
- Subsurface scattering happens when light enters the surface and scatters inside of it before being emitted
  - The emitted point is often in a different location than the incident point!
- Overall, we're going for a certain softness...

Screen Space Subsurface Scattering

Case Study

Hardcore

- This thing is pretty ridiculous
- Released in 2007, still the gold standard for real-time skin
- It's the cover of GPU Gems 3!
- Uncharted uses a toned-down version of this technique


GPU Gems 3
NVIDIA
Edited by Hubert Nguyen
Foreword by Kurt Akeley

Wat

Ok mister

I'm tired and it's 6AM so here's the short version

1. Evaluate specular BRDF using Kelemen/Szirmay-Kalos model
   a. Just another BRDF, don't freak out
   b. Precompute the micropolygon distribution function and store it in a texture
   c. Specular parameters actually vary across the surface, store in a texture map

...

2. Subsurface scattering
    a. There is some diffusion profile which measures how likely light will transmit a certain distance away from the incident point
    b. Diffusion profile is generated based on observed data and modeling the skin in three main layers
    c. You can just convolve the skin texture map with the diffusion profile to precompute the approximate subsurface scattering
        i. Have to correct for texture space stretching

Bells and Whistles

- Additional texture space computations allow for transmission through thin areas such as nostrils or ears
- Finally a bloom filter because why not

Demo if I can get it to run...