

# Reflection models

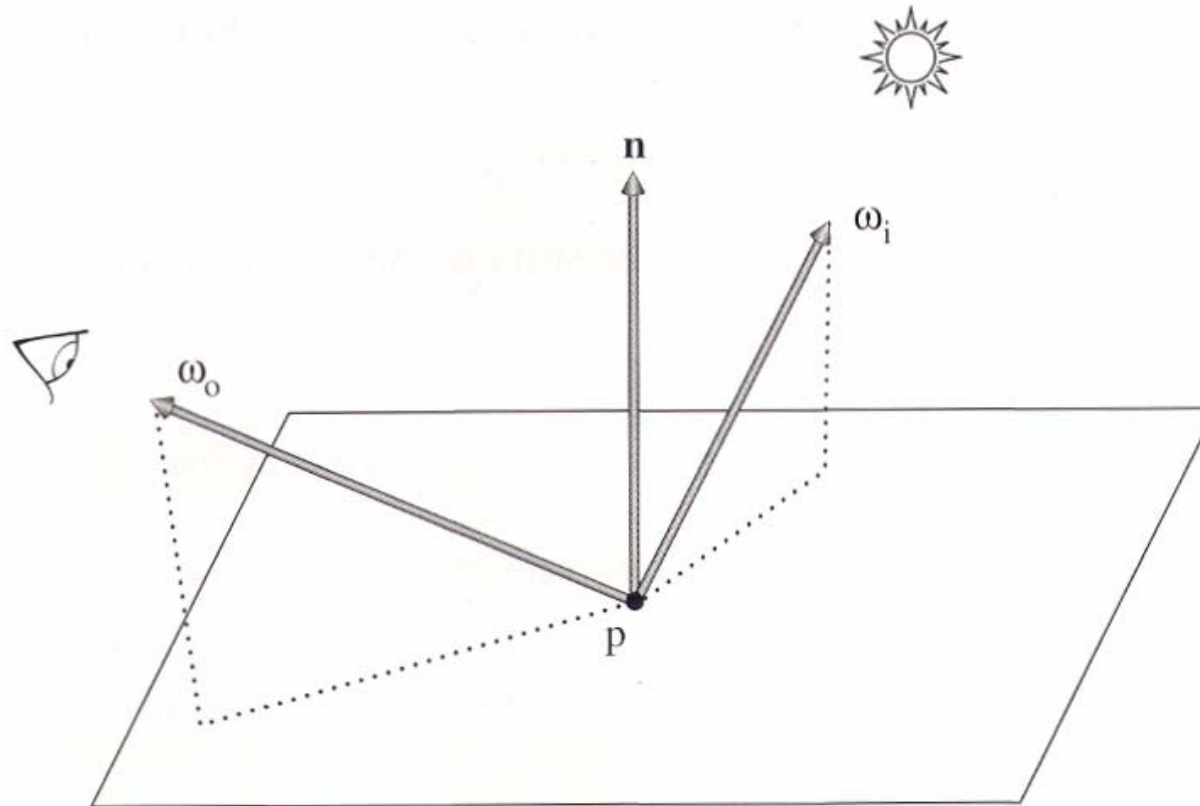
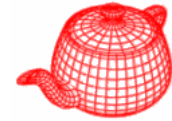
Digital Image Synthesis

*Yung-Yu Chuang*

11/16/2006

*with slides by Pat Hanrahan and Matt Pharr*

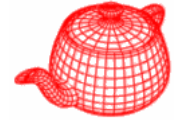
# Rendering equation



$$L(\omega_o) = \int_{\Omega} f(\omega_i \rightarrow \omega_o) L(\omega_i) \cos \theta_i d\omega_i$$

# Taxonomy 1

---



$$(x, y, t, \theta, \phi, \lambda)_{in} \rightarrow (x, y, t, \theta, \phi, \lambda)_{out}$$

General function = 12D

↓ assume time doesn't matter (no phosphorescence)  
↓ assume wavelengths are equal (no fluorescence)

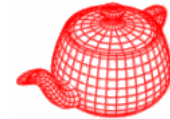
Scattering function = 9D

↓ assume wavelength is discretized or integrated into RGB  
(This is a common assumption for computer graphics)

Single-wavelength Scattering function = 8D

$$(x, y, \theta, \phi)_{in} \rightarrow (x, y, \theta, \phi)_{out}$$

# Taxonomy 2



$(x, y, \theta, \phi)_{in} \rightarrow (x, y, \theta, \phi)_{out}$   
Single-wavelength Scattering function = 8D

ignore subsurface scattering  $(x, y)_{in} = (x, y)_{out}$

ignore dependence on position

Bidirectional Texture Function (BTF)  
Spatially-varying BRDF (SVBRDF) = 6D

Bidirectional Subsurface Scattering  
Distribution Function (BSSRDF) = 6D

ignore direction of incident light

ignore dependence  
on position

ignore subsurface scattering

Light Fields, Surface LFs = 4D

BRDF = 4D

$(x, y, \theta, \phi)_{out}$

$(\theta, \phi)_{in} \rightarrow (\theta, \phi)_{out}$

assume Lambertian

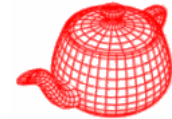
assume isotropy

Texture Maps = 2D

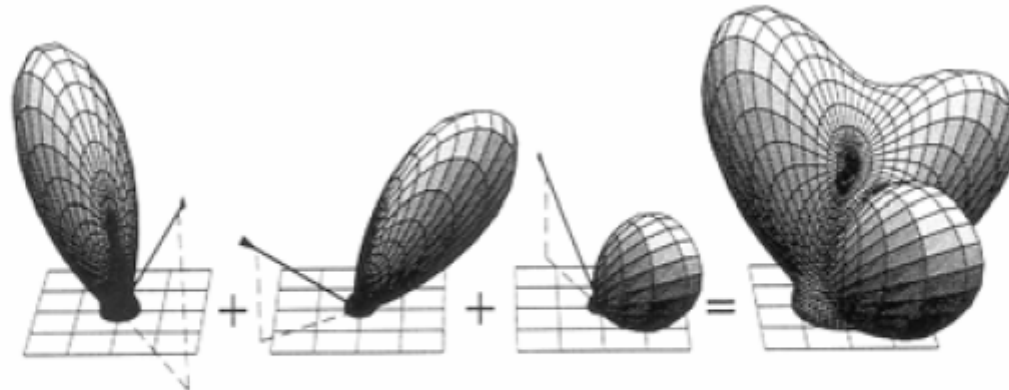
3D

$(x, y)_{out}$

# Properties of BRDFs

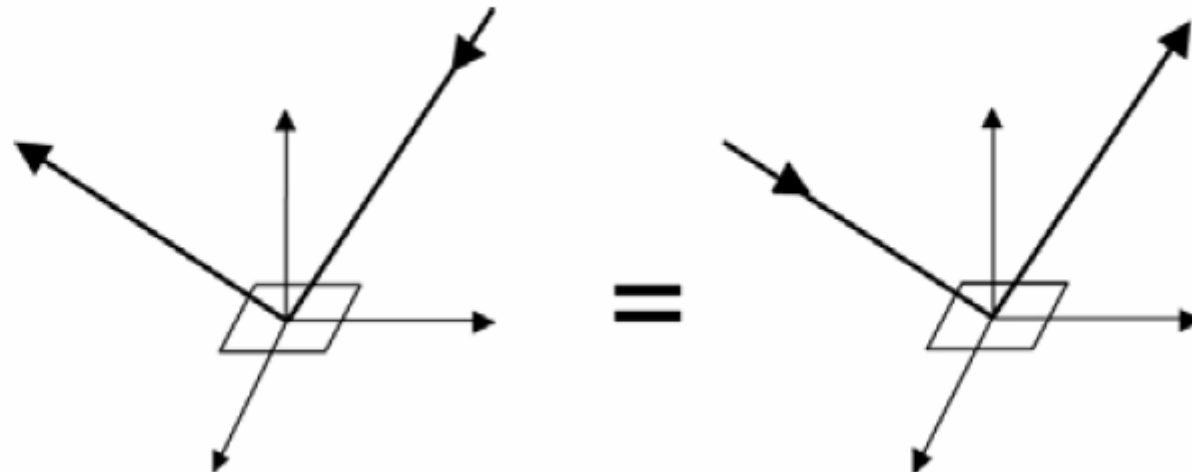


## 1. Linear

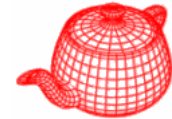


From Sillion, Arvo, Westin, Greenberg

## 2. Reciprocity principle $f_r(\omega_r \rightarrow \omega_i) = f_r(\omega_i \rightarrow \omega_r)$

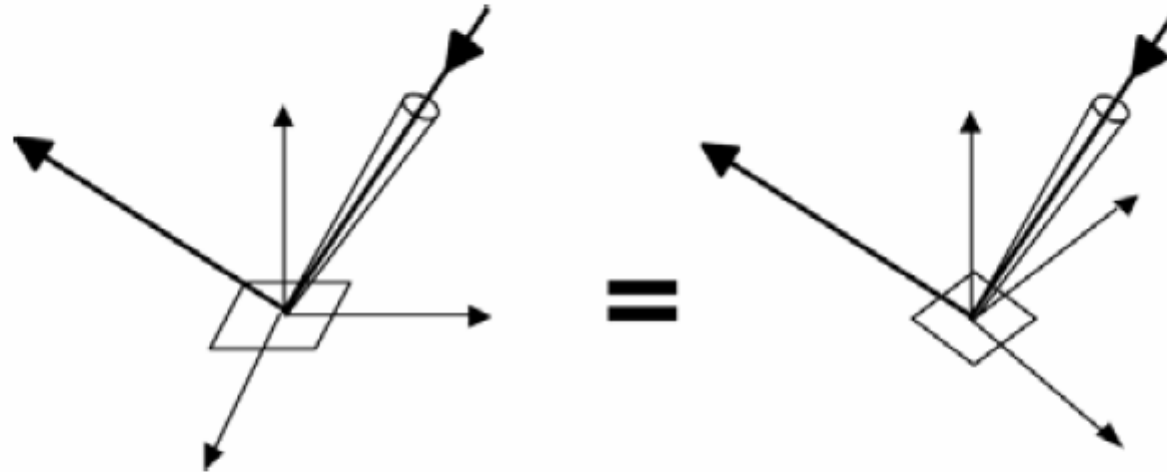


# Properties of BRDFs



## 3. Isotropic vs. anisotropic

$$f_r(\theta_i, \varphi_i; \theta_r, \varphi_r) = f_r(\theta_i, \theta_r, \varphi_r - \varphi_i)$$



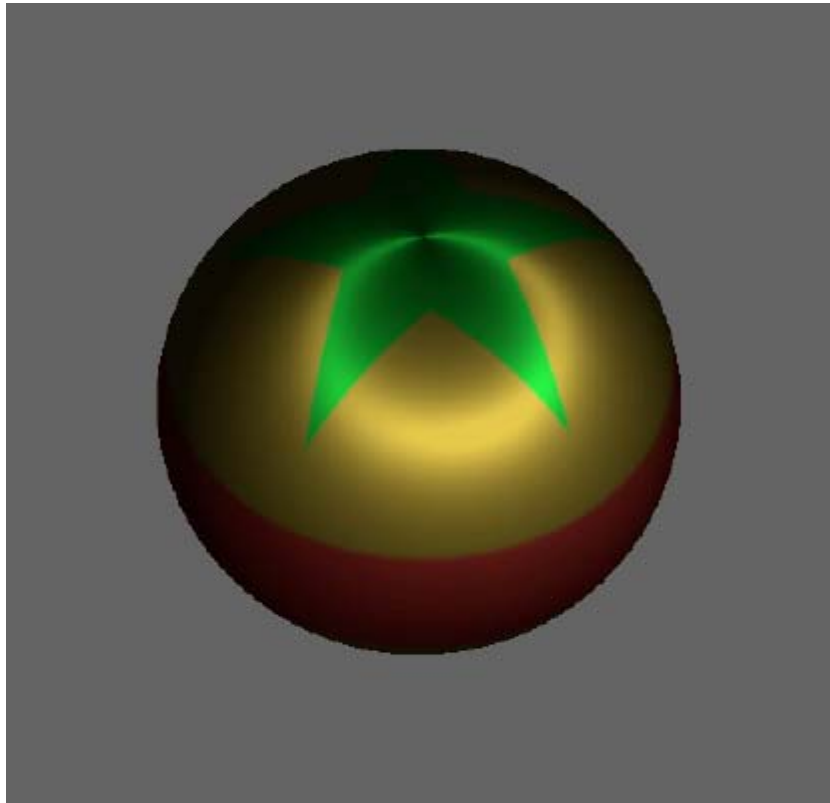
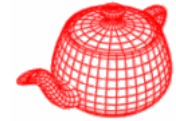
## Reciprocity and isotropy

$$f_r(\theta_i, \theta_r, \varphi_r - \varphi_i) = f_r(\theta_r, \theta_i, \varphi_i - \varphi_r) = f_r(\theta_i, \theta_r, |\varphi_r - \varphi_i|)$$

## 4. Energy conservation $\int_{\Omega} f_r(\omega_o, \omega_i) \cos \theta_i d\omega_i \leq 1$

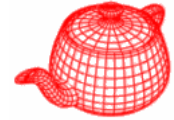
# Isotropic and anisotropic

---



# Reflection models

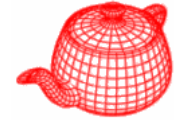
---



- BRDF/BTDF/BSDF
- Scattering from realistic surfaces is best described as a mixture of multiple BRDFs and BSDFs.
- **core/reflection.\***
- Material = BSDF that combines multiple BRDFs and BSDFs. (chap. 10)
- Textures = reflection and transmission properties that vary over the surface. (chap. 11)

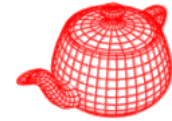
# Surface reflection models

---

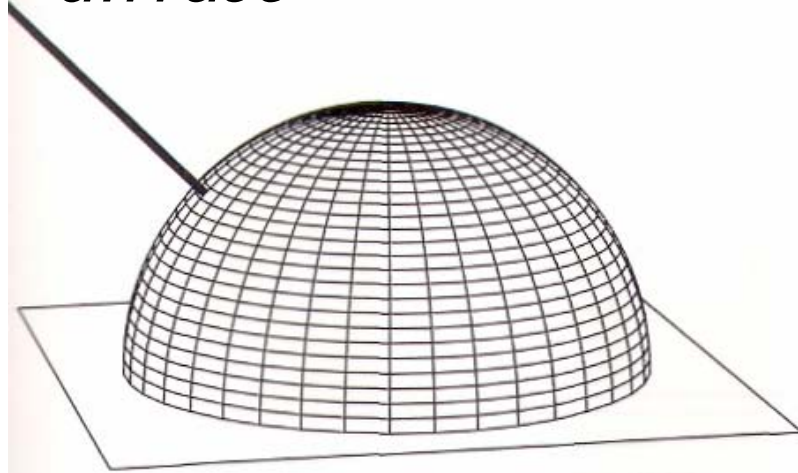


- Measured data: usually described in tabular form or coefficients of a set of basis functions
- Phenomenological models: *qualitative* approach; models with intuitive parameters
- Simulation: simulates light scattering from microgeometry and known reflectance properties
- Physical optics: solve Maxwell's equation
- Geometric optics: microfacet models

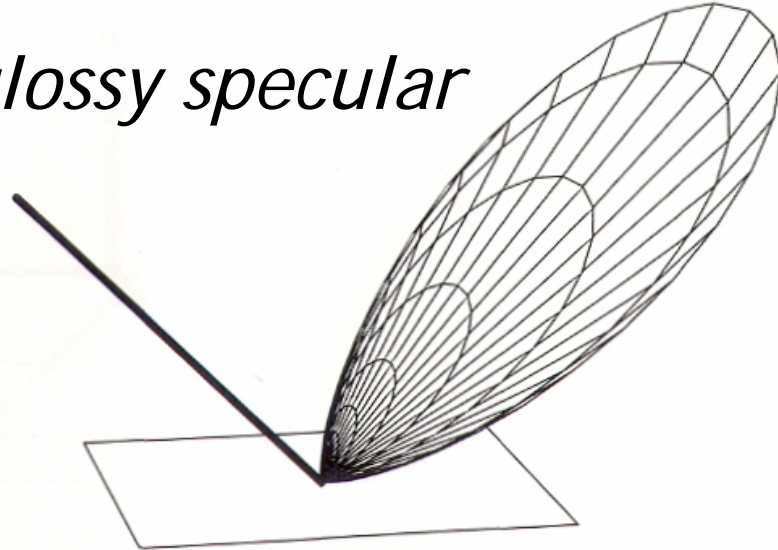
# Reflection categories



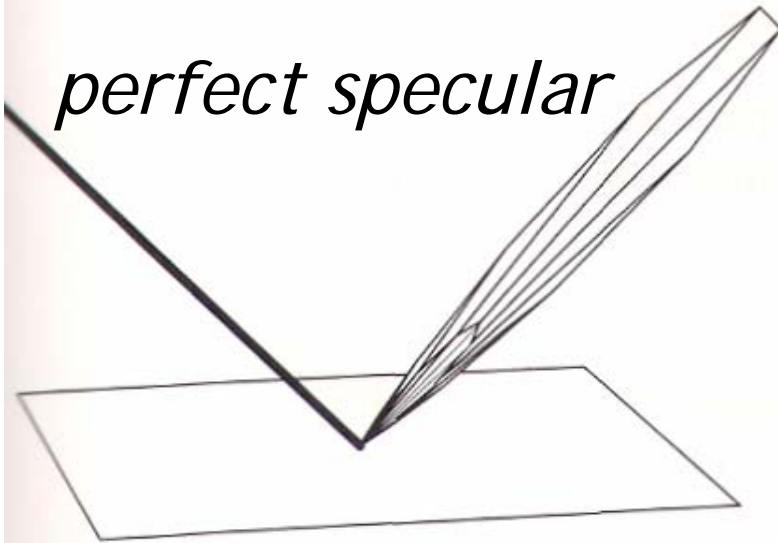
*diffuse*



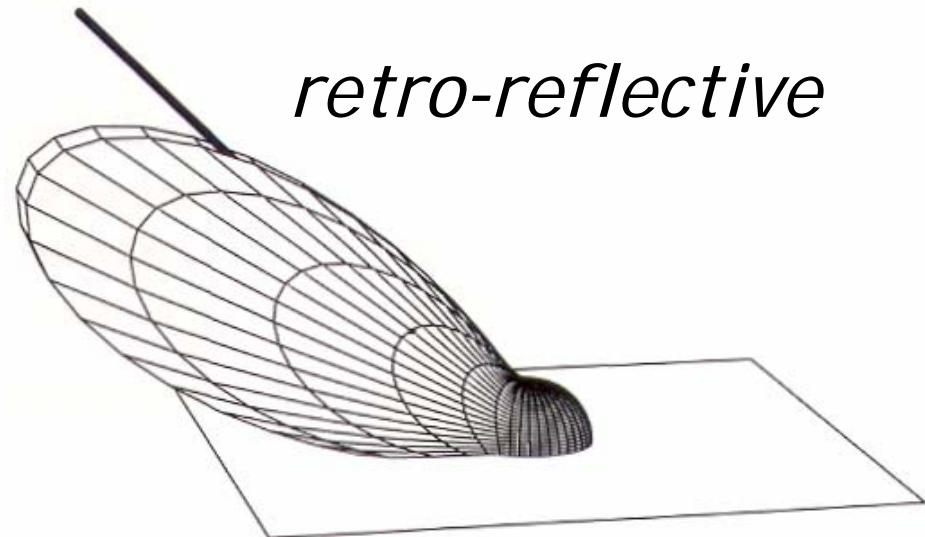
*glossy specular*



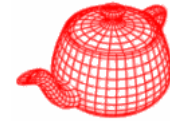
*perfect specular*



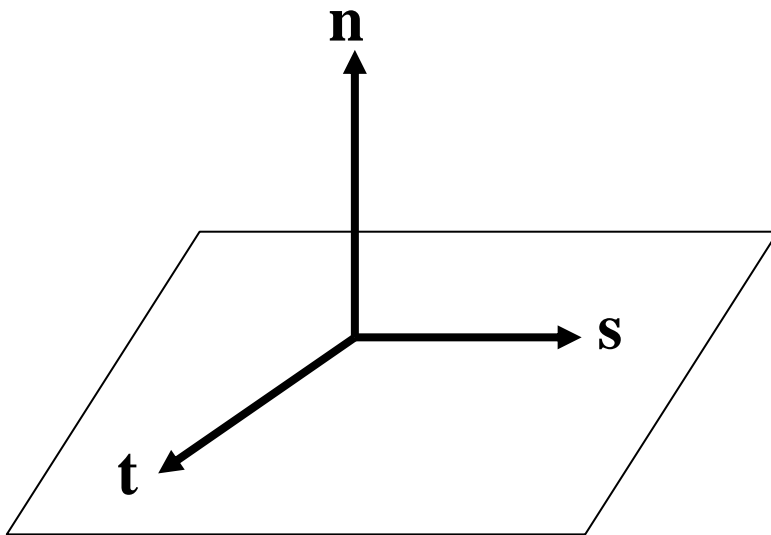
*retro-reflective*



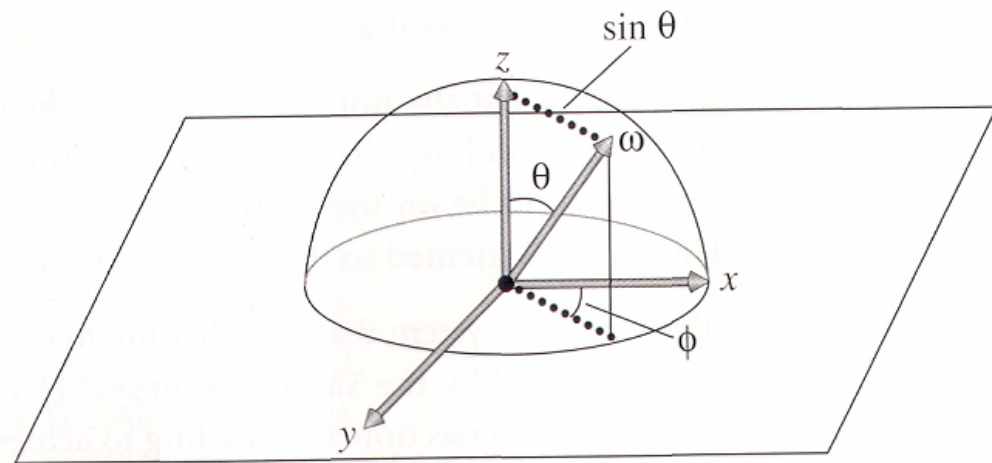
# Geometric setting



incident and outgoing directions are normalized and outward facing after being transformed into the local frame



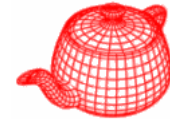
local frame



$$\cos \theta = \omega_z, \quad \sin \theta = \sqrt{1 - \omega_z^2}$$

$$\cos \phi = \frac{\omega_x}{\sin \theta}, \quad \sin \phi = \frac{\omega_y}{\sin \theta}$$

# BxDF



- **BxDFType**
  - BSDF\_REFLECTION, BSDF\_TRANSMISSION
  - BSDF\_DIFFUSE, BSDF\_GLOSSY (retro-reflective), BSDF\_SPECULAR
- Spectrum **f**(Vector &wo, Vector &wi)=0;
- Spectrum **Sample\_f**(Vector &wo, Vector \*wi, float u1, float u2, float \*pdf);  
used to find an incident direction for an outgoing direction;  
especially useful for reflection with a delta distribution
- Spectrum **rho**(Vector &wo, int nSamples=16, float \*samples=NULL);

hemispherical-directional  
reflectance; computed  
analytically or by sampling

$$\rho_{hd}(\omega_o) = \int_{\Omega} f_r(p, \omega_o, \omega_i) |\cos \theta_i| d\omega_i$$

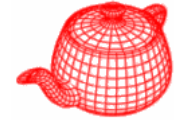
- Spectrum **rho**(int nSamples, float \*samples);

hemispherical-hemispherical  
reflectance

$$\rho_{hh} = \frac{1}{\pi} \int_{\Omega} \int_{\Omega} f_r(p, \omega_o, \omega_i) |\cos \theta_i \cos \theta_o| d\omega_i d\omega_o$$



# Fresnel reflectance

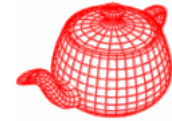


- Reflectivity and transmissiveness: fraction of incoming light that is reflected or transmitted; they are usually uniform over the surface but **view dependent**. Hence, the reflectivity should be corrected by *Fresnel equation*
- *Fresnel reflectance* for dielectrics

$$r_{\parallel} = \frac{\eta_t \cos \theta_i - \eta_i \cos \theta_t}{\eta_t \cos \theta_i + \eta_i \cos \theta_t} \quad F_r(\omega_i) = \frac{1}{2} (r_{\parallel}^2 + r_{\perp}^2)$$
$$r_{\perp} = \frac{\eta_i \cos \theta_i - \eta_t \cos \theta_t}{\eta_i \cos \theta_i + \eta_t \cos \theta_t} \quad F_t(\omega_i) = (1 - F_r(\omega_i))$$

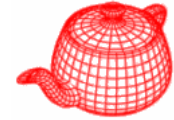
# Indices of refraction

---



medium	Index of refraction
Vaccum	1.0
Air at sea level	1.00029
Ice	1.31
Water (20°C)	1.333
Fused quartz	1.46
Glass	1.5~1.6
Sapphire	1.77
Diamond	2.42

# Fresnel reflectance



- *Fresnel reflectance* for conductors (no transmission)

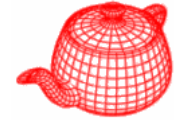
index of refraction  $\swarrow$       absorption coefficient  $\swarrow$

$$r_{\parallel}^2 = \frac{(\eta^2 + k^2) \cos^2 \theta_i - 2\eta \cos \theta_i + 1}{(\eta^2 + k^2) \cos^2 \theta_i + 2\eta \cos \theta_i + 1}$$

$$r_{\perp}^2 = \frac{(\eta^2 + k^2) - 2\eta \cos \theta_i + \cos^2 \theta_i}{(\eta^2 + k^2) + 2\eta \cos \theta_i + \cos^2 \theta_i}$$

$$F_r(\omega_i) = \frac{1}{2} (r_{\parallel}^2 + r_{\perp}^2)$$

# $\eta$ and $k$ for a few conductors



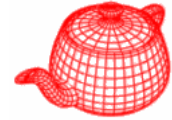
---

Object	$\eta$	$k$
Gold	0.370	2.820
Silver	0.177	3.638
Copper	0.617	2.630
Steel	2.485	3.433

- However, for most conductors, these coefficients are unknown. Approximations are used to find plausible values for these quantities if reflectance at the normal incidence is known.

# Approximation

---



- Measure  $F_r$  for  $\theta_{i=0}$

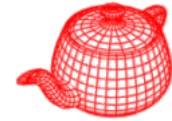
1. Assume  $k = 0$

$$r_{\perp}^2 = r_{\parallel}^2 = \frac{(\eta - 1)^2}{(\eta + 1)^2} \quad \eta = \frac{1 + \sqrt{F_r(0)}}{1 - \sqrt{F_r(0)}}$$

2. Assume  $\eta = 1$

$$r_{\perp}^2 = r_{\parallel}^2 = \frac{k^2}{k^2 + 4} \quad k = 2\sqrt{\frac{F_r(0)}{1 - F_r(0)}}$$

# Fresnel class

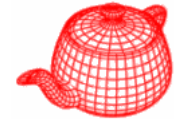


```
class Fresnel {
public:
    virtual Spectrum Evaluate(float cosi) const = 0;
};
class FresnelConductor : public Fresnel {
public:
    FresnelConductor(Spectrum &e, Spectrum &kk)
        : eta(e), k(kk) {}
private:
    Spectrum eta, k;
};
class FresnelDielectric : public Fresnel {
public:
    FresnelDielectric(float ei, float et) {
        eta_i = ei; eta_t = et; }
private:
    float eta_i, eta_t;
};
```

**Evaluate** directly implements  
Fresnel formula for conductor

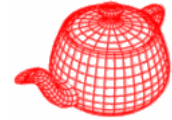
**Evaluate** directly implements  
Fresnel formula for dielectric

# Specular reflection



```
class SpecularReflection : public BxDF {
public:
    SpecularReflection(const Spectrum &r, Fresnel *f)
        : BxDF(BxDFType(BSDF_REFLECTION | BSDF_SPECULAR)),
          R(r), fresnel(f) { }
    Spectrum f(const Vector &, const Vector &) const {
        return Spectrum(0.);
    }
    Spectrum Sample_f(const Vector &wo, Vector *wi,
                      float u1, float u2, float *pdf) const;
    float Pdf(const Vector &wo, const Vector &wi) const {
        return 0.;
    }
private:
    Spectrum R;
    Fresnel *fresnel;
};
```

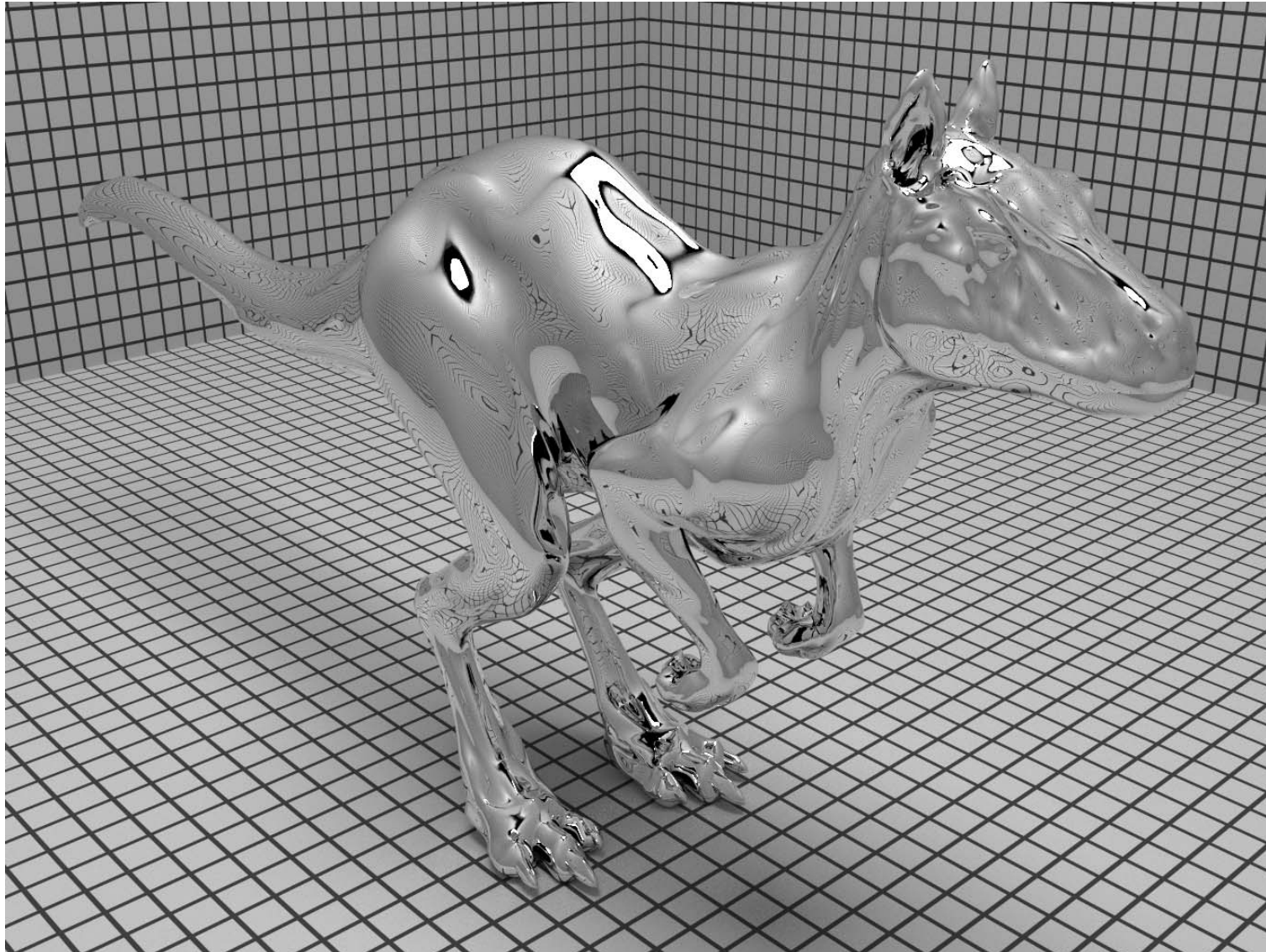
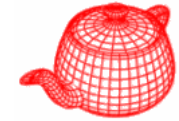
# Specular reflection



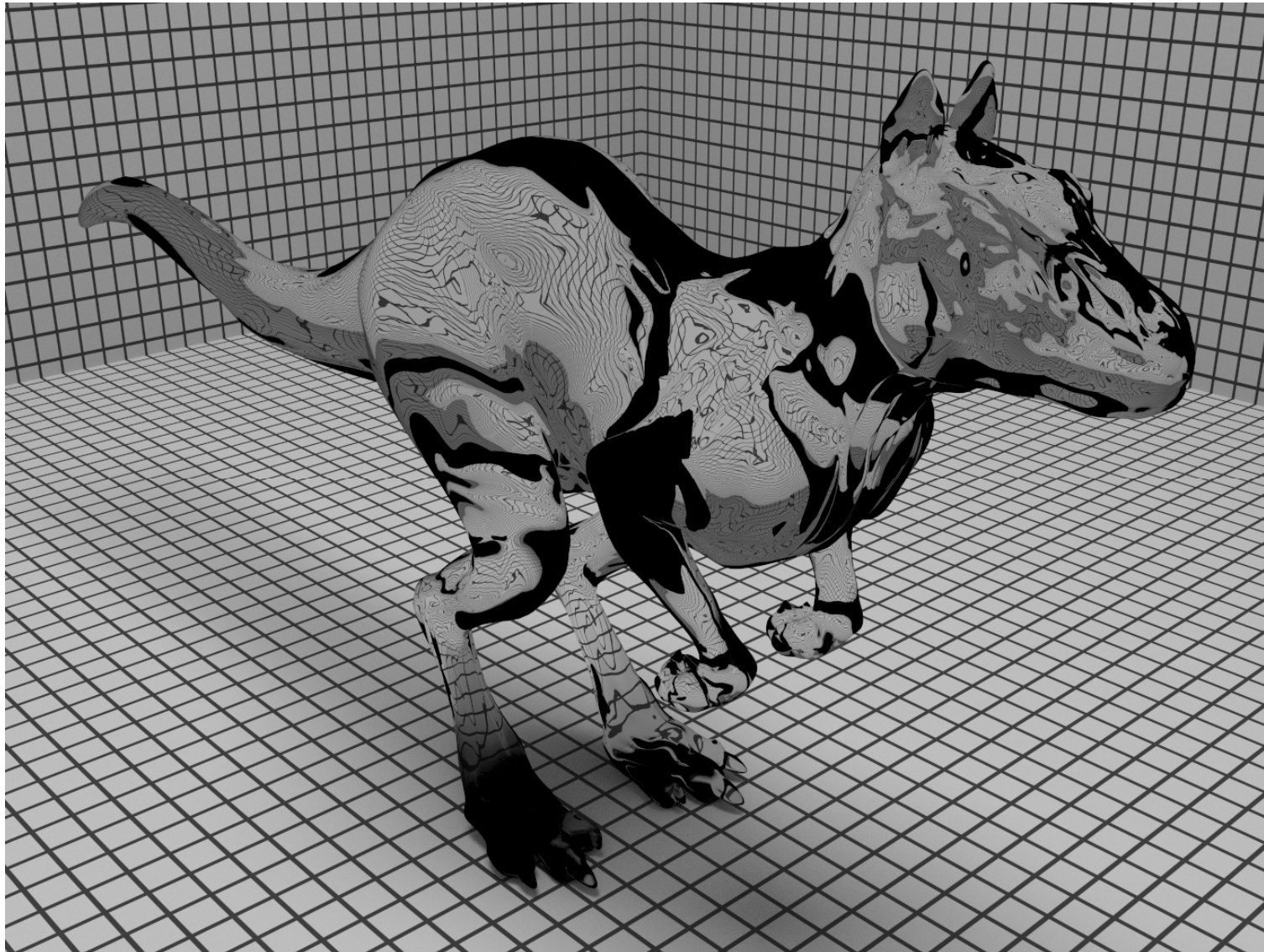
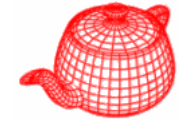
---

```
Spectrum SpecularReflection::Sample_f(Vector &wo,
    Vector *wi, float u1, float u2, float *pdf) const{
    // Compute perfect specular reflection direction
    *wi = Vector(-wo.x, -wo.y, wo.z);
    *pdf = 1.f;
    return fresnel->Evaluate(CosTheta(wo)) * R /
        fabsf(CosTheta(*wi));
}
```

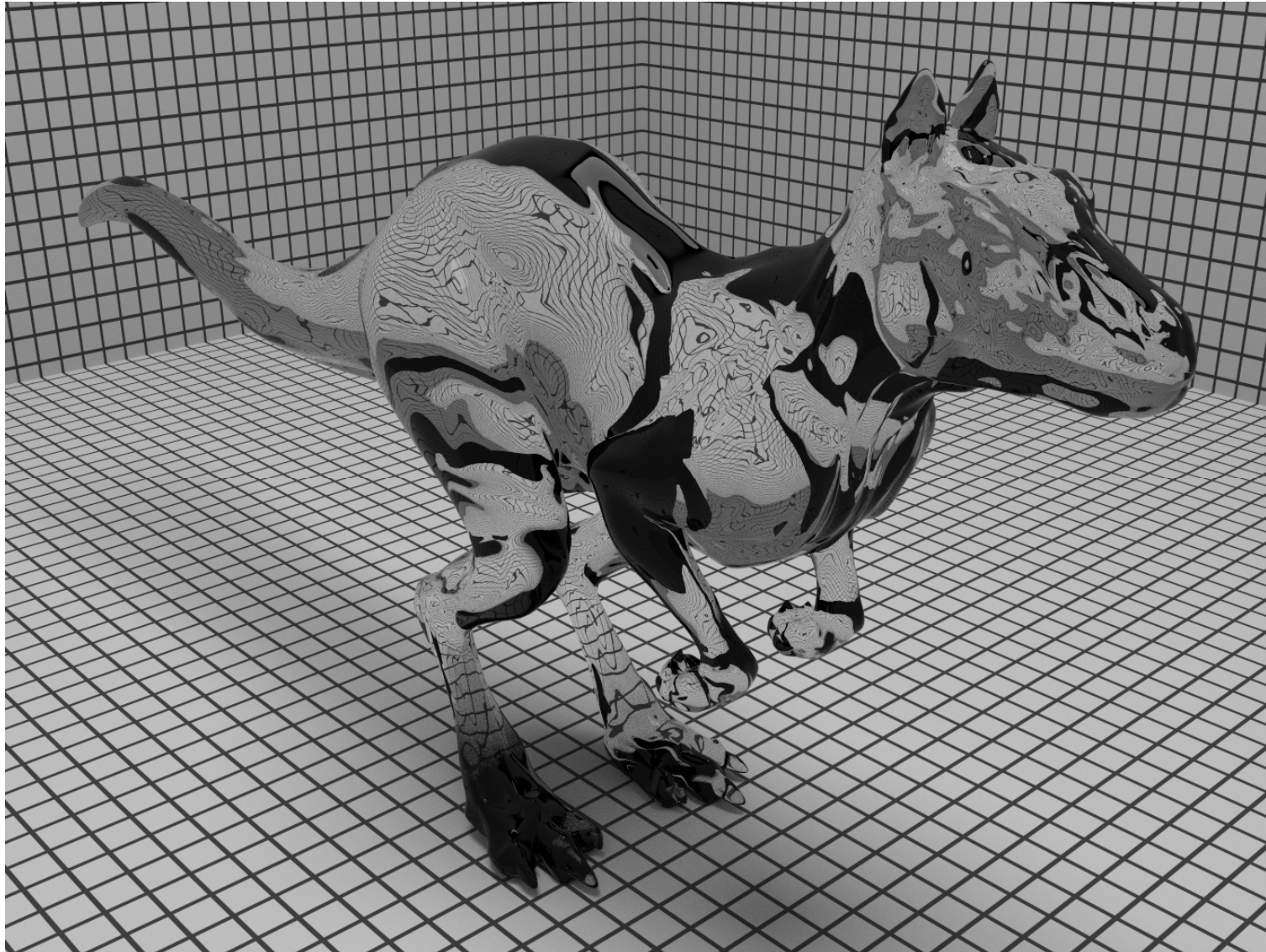
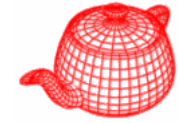
# Perfect specular reflection



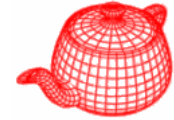
# Perfect specular transmission



# Fresnel modulation



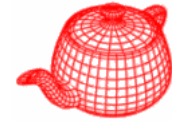
# Lambertian reflection



- It is not physically feasible, but provides a good approximation to many real-world surfaces.

```
class COREDLL Lambertian : public BxDF {
public:
    Lambertian(Spectrum &reflectance)
        : BxDF(BxDFType(BSDF_REFLECTION | BSDF_DIFFUSE)),
          R(reflectance), RoverPI(reflectance * INV_PI) {}
    Spectrum f(Vector &wo, Vector &wi) {return RoverPI}
    Spectrum rho(Vector &, int, float *) { return R; }
    Spectrum rho(int, float *) { return R; }
private:
    Spectrum R, RoverPI;
};
```

# Derivations



$$\rho_{hh} = \frac{1}{\pi} \int_{\Omega} \int_{\Omega} f_r(p, \omega_o, \omega_i) |\cos \theta_i \cos \theta_o| d\omega_i d\omega_o$$

$$R = \frac{1}{\pi} \int_{\Omega} \int_{\Omega} c |\cos \theta_i \cos \theta_o| d\omega_i d\omega_o$$

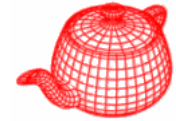
$$R = \frac{c}{\pi} \cdot \int_{\Omega} \cos \theta_i d\omega_i \cdot \int_{\Omega} \cos \theta_o d\omega_o = c\pi$$

$$c = \frac{R}{\pi}$$

$$\begin{aligned} \int_{\Omega} \cos \theta_i d\omega_i &= \int_0^{2\pi} \int_0^{\pi/2} \cos \theta_i \sin \theta_i d\theta_i d\phi_i \\ &= \int_0^{2\pi} d\phi_i \int_0^{\pi/2} \cos \theta_i \sin \theta_i d\theta_i \\ &= 2\pi \int_0^{\pi/2} \frac{1}{2} \sin(2\theta_i) \frac{1}{2} d(2\theta_i) \\ &= \frac{\pi}{2} \cdot -\cos(2\theta_i) \Big|_0^{\pi/2} = \pi \end{aligned}$$

# Derivations

---



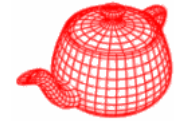
$$\rho_{hd}(\omega_o) = \int_{\Omega} f_r(p, \omega_o, \omega_i) |\cos \theta_i| d\omega_i$$

$$= \int_{\Omega} \frac{R}{\pi} \cos \theta_i d\omega_i$$

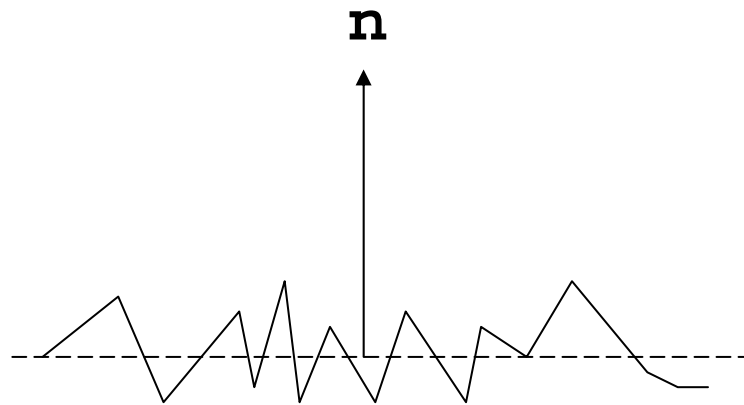
$$= \frac{R}{\pi} \int_{\Omega} \cos \theta_i d\omega_i$$

$$= \frac{R}{\pi} \cdot \pi = R$$

# Microfacet models

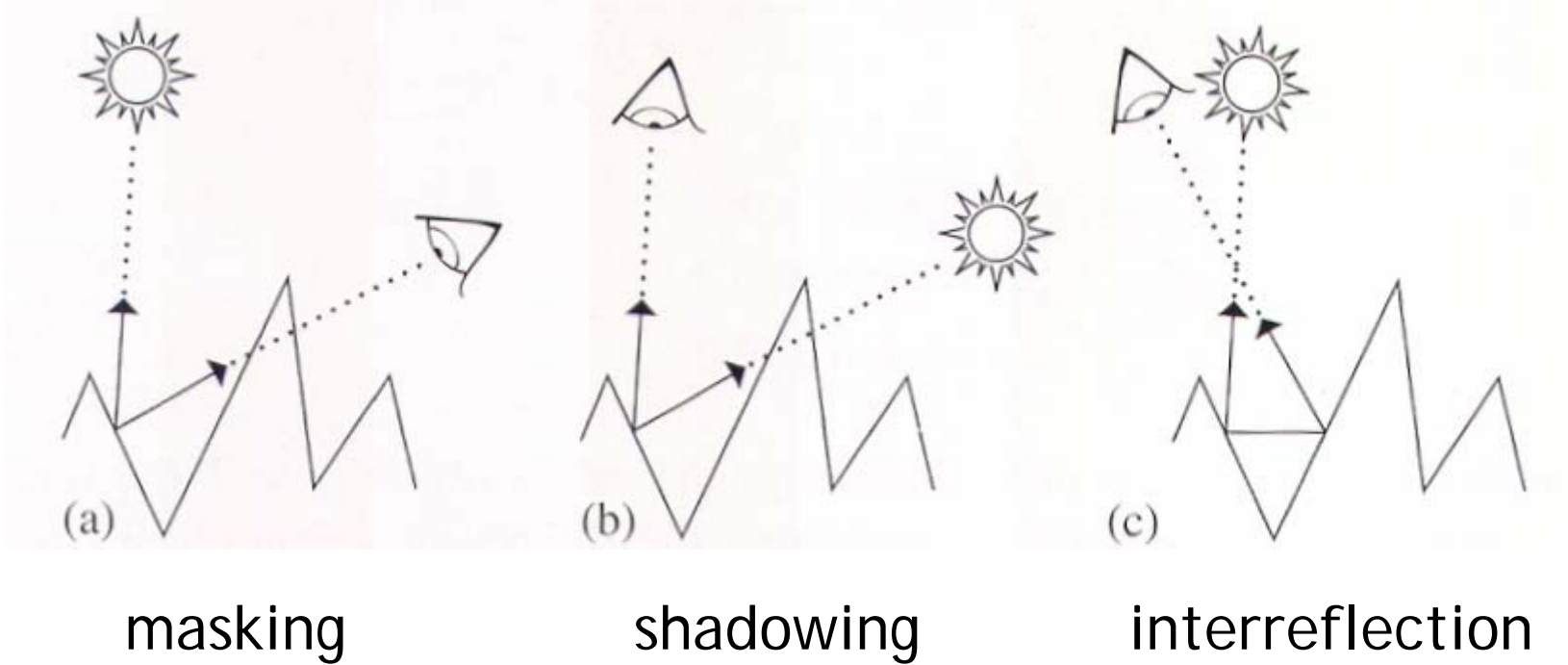


- Rough surfaces can be modeled as a collection of small microfacets. Their **aggregate behavior** determines the scattering.
- Two components: distribution of microfacets and how light scatters from individual microfacet → closed-form BRDF expression



# Important geometric effects to consider

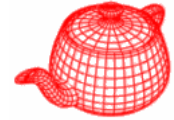
---



Most microfacet models assume that all microfacets make up symmetric V-shaped grooves so that only neighboring microfacet needs to be considered. Particular models consider these effects with varying degrees of accuracy.

# Oren-Nayar model

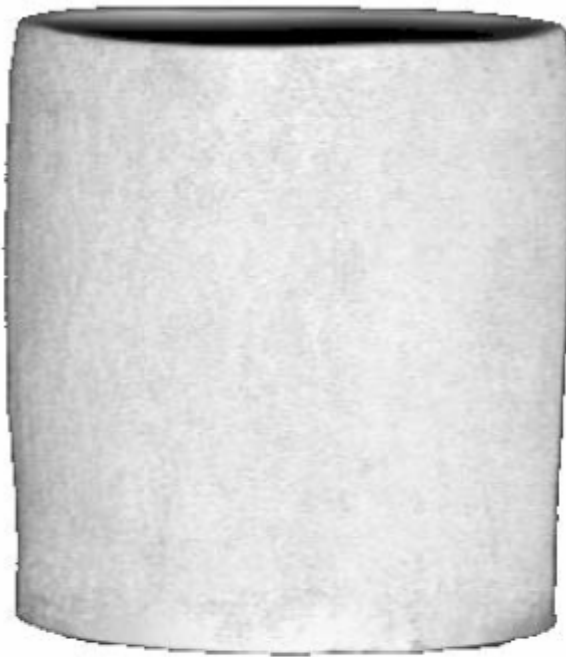
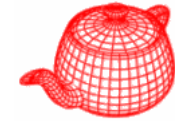
---



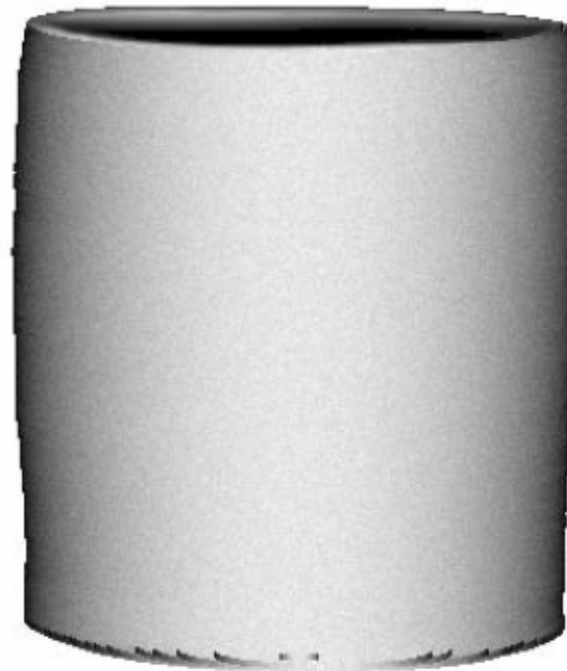
- Many real-world materials such as concrete, sand and cloth are not real Lambertian. Specifically, rough surfaces generally appear brighter as the illumination direction approaches the viewing direction.
- A collection of symmetric V-shaped perfect **Lambertian** grooves whose orientation angles follow a **Gaussian distribution**.
- Don't have a closed-form solution, instead they used an approximation

# Oren-Nayar model

---



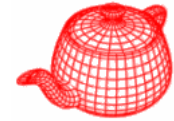
(a) Real image



(b) Lambertian model

# Oren-Nayar model

---



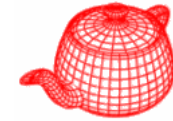
standard deviation for Gaussian

$$A = 1 - \frac{\sigma^2}{2(\sigma^2 + 0.33)}, \quad B = \frac{0.45\sigma^2}{\sigma^2 + 0.09}$$

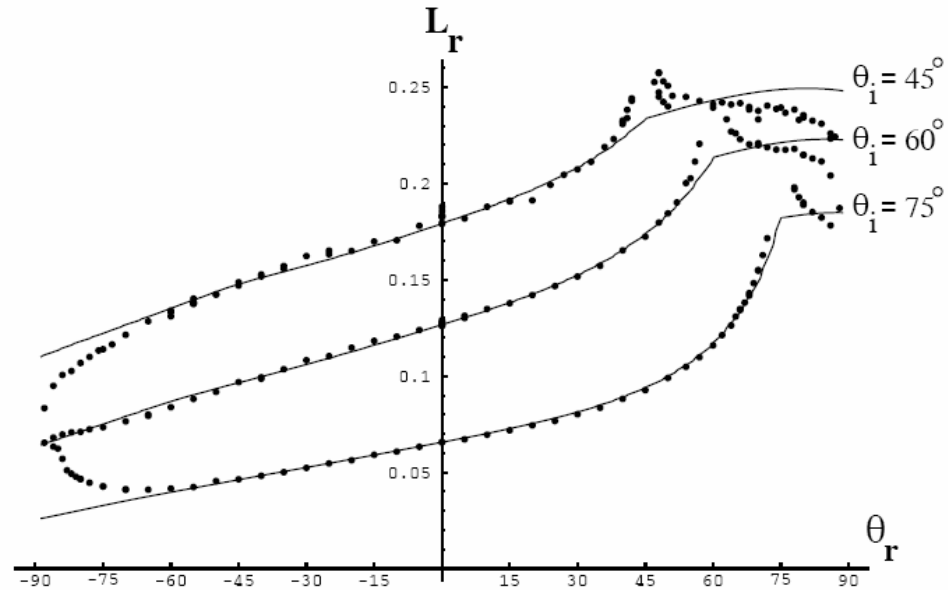
$$\alpha = \max(\theta_i, \theta_o), \quad \beta = \min(\theta_i, \theta_o)$$

$$f_r(\omega_i, \omega_o) = \frac{\rho}{\pi} (A + B \max(0, \cos(\phi_i - \phi_o))) \sin \alpha \tan \beta$$

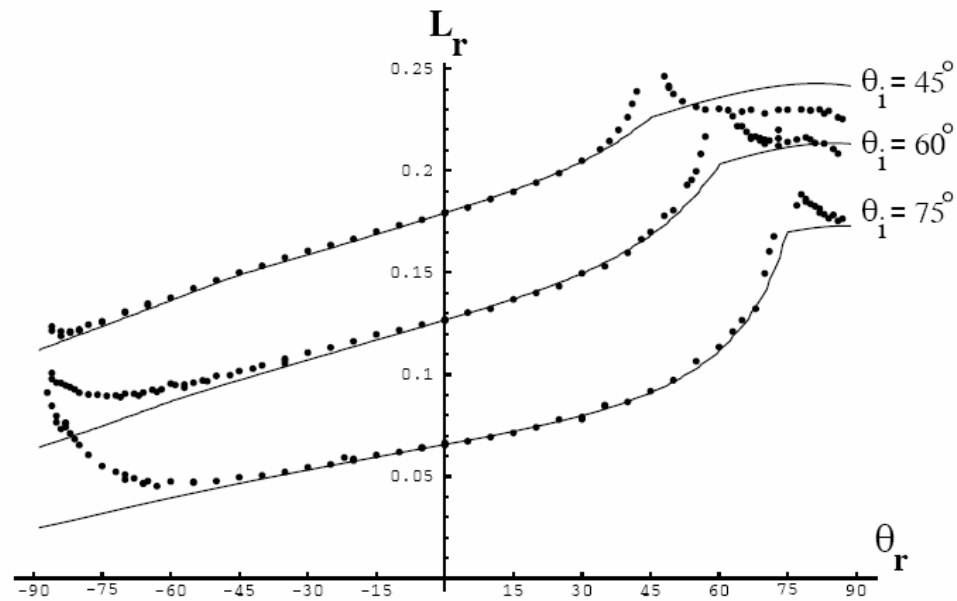
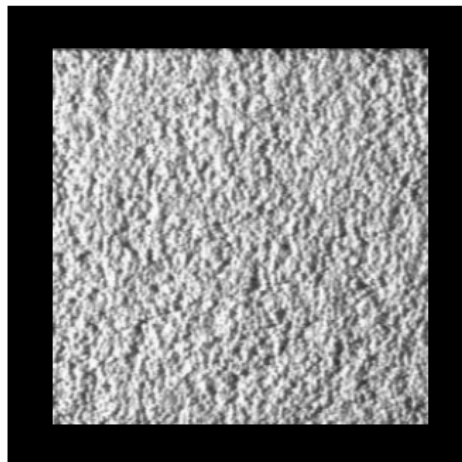
# Oren-Nayar model



Sand Paper

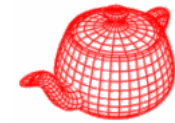


Sand

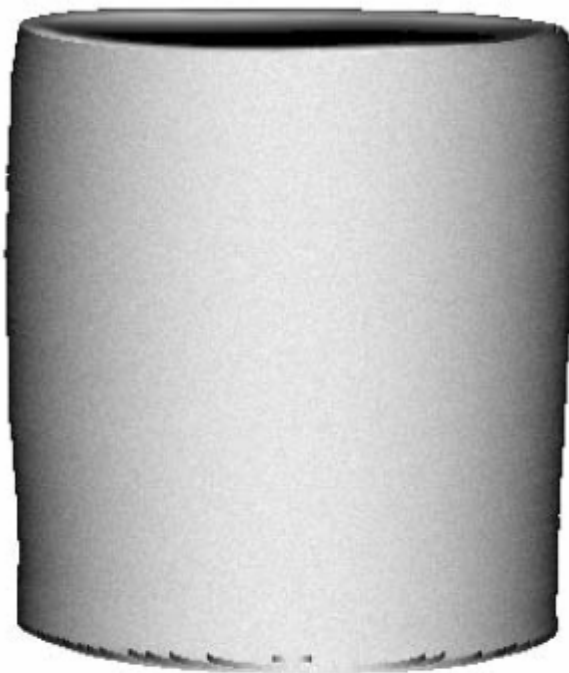


# Oren-Nayar model

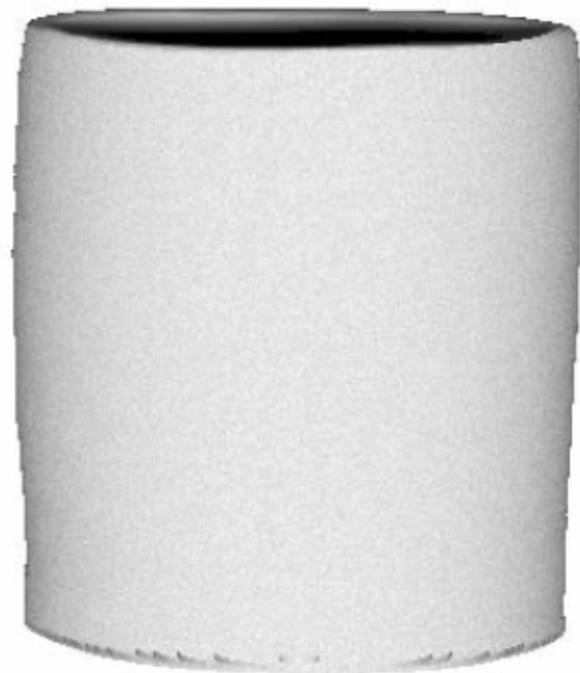
---



(a) Real image

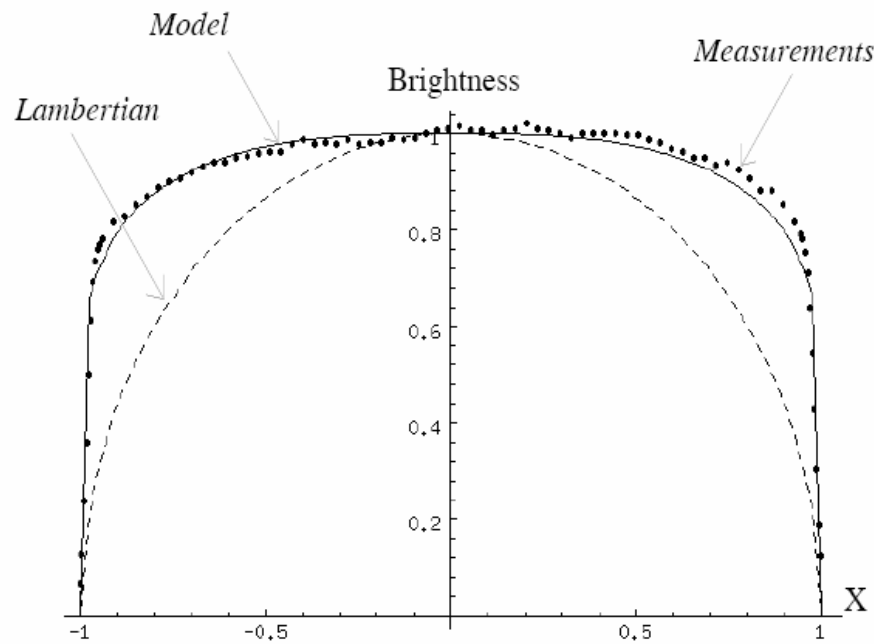
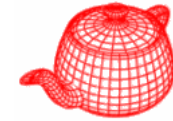


(b) Lambertian model

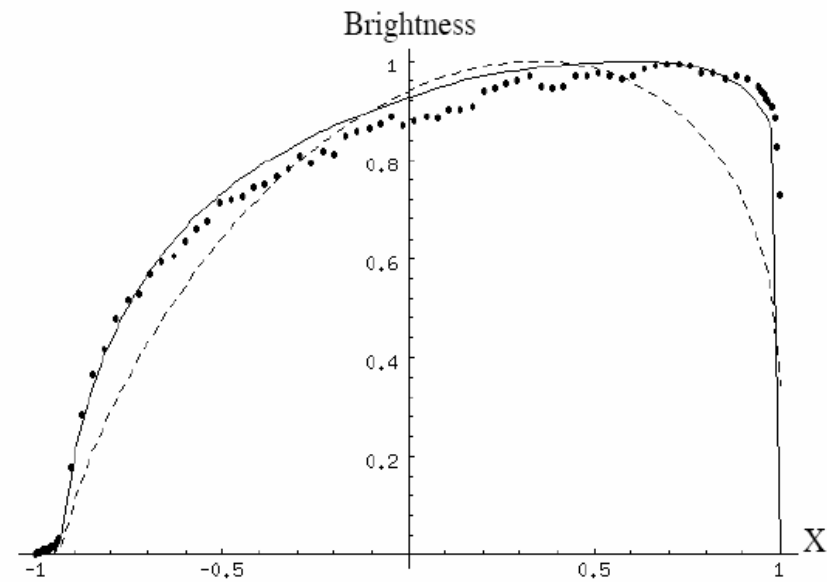


(c) Proposed model

# Oren-Nayar model

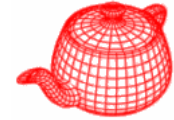


(a)  $\theta_i = 0^\circ$



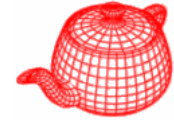
(b)  $\theta_i = 20^\circ$

# Oren-Nayar model



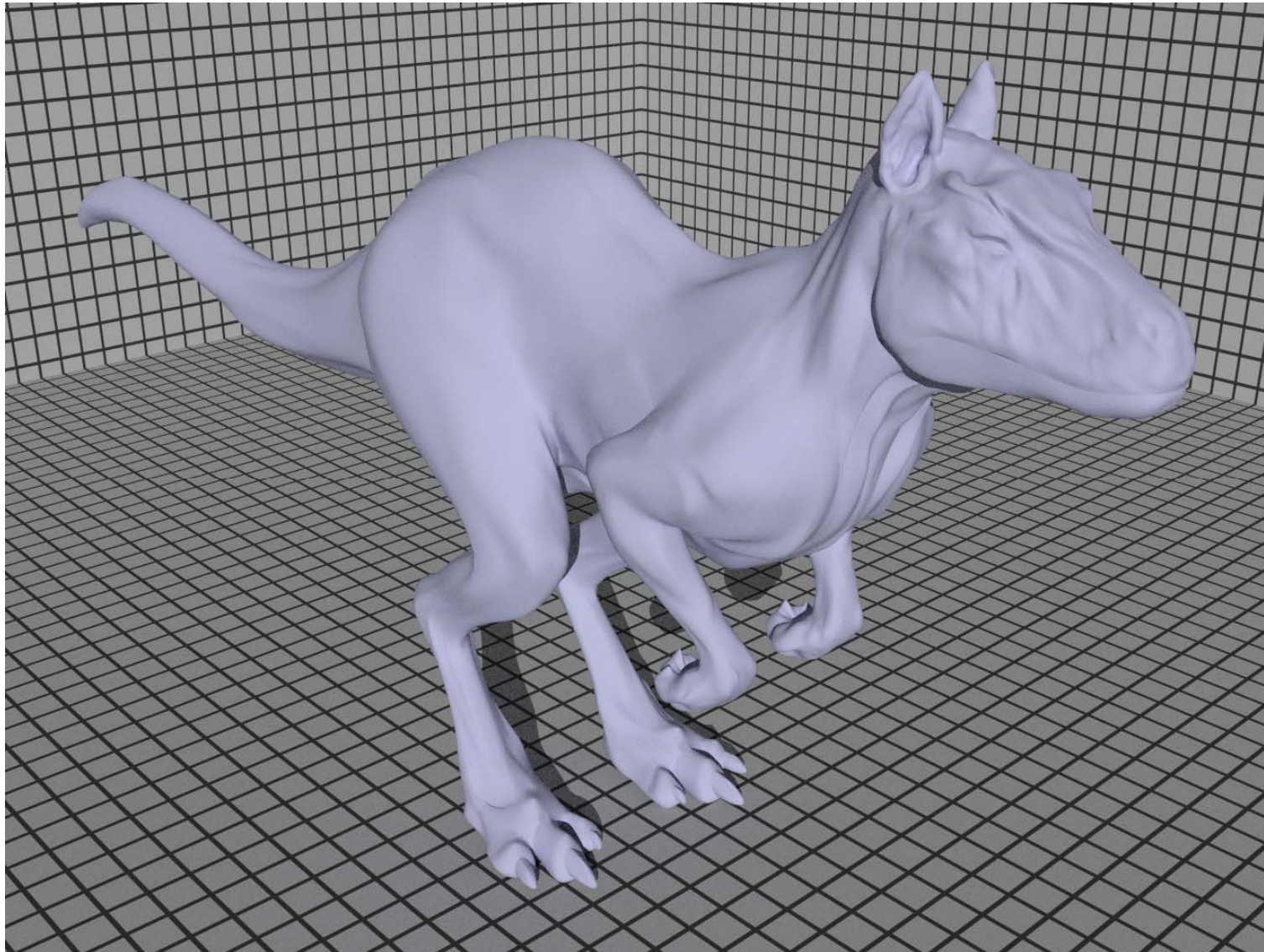
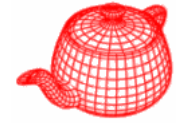
```
class OrenNayar : public BxDF {
public:
    Spectrum f(const Vector &wo, const Vector &wi) const;
    OrenNayar(const Spectrum &reflectance, float sig)
        : BxDF(BxDFType(BSDF_REFLECTION | BSDF_DIFFUSE)),
          R(reflectance) {
        float sigma = Radians(sig);
        float sigma2 = sigma*sigma;
        A = 1.f - (sigma2 / (2.f * (sigma2 + 0.33f)));
        B = 0.45f * sigma2 / (sigma2 + 0.09f);
    }
private:
    Spectrum R;
    float A, B;
};
```

# Oren-Nayar model



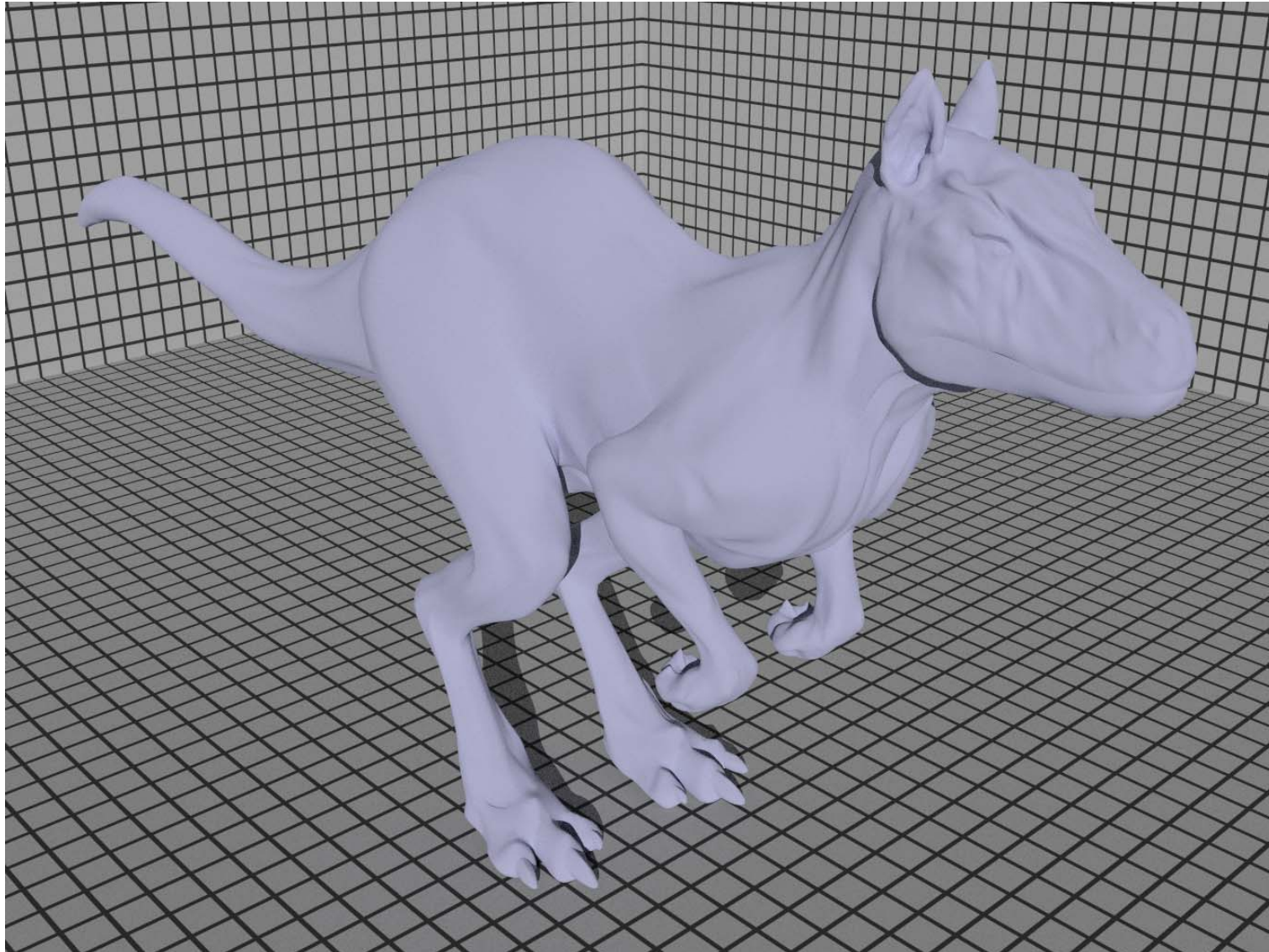
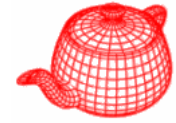
```
Spectrum OrenNayar::f(Vector &wo, Vector &wi)vconst{
    float sinthetai = SinTheta(wi);
    float sinthetao = SinTheta(wo);
    float sinphii = SinPhi(wi), cosphii = CosPhi(wi);
    float sinphio = SinPhi(wo), cosphio = CosPhi(wo);
    float dcos = cosphii * cosphio + sinphii * sinphio;
    float maxcos = max(0.f, dcos);
    float sinalpha, tanbeta;
    if (fabsf(CosTheta(wi)) > fabsf(CosTheta(wo))) {
        sinalpha = sinthetao;
        tanbeta = sinthetai / fabsf(CosTheta(wi));
    } else {
        sinalpha = sinthetai;
        tanbeta = sinthetao / fabsf(CosTheta(wo));
    }
    return R * INV_PI *
        (A + B * maxcos * sinalpha * tanbeta);
}
```

# Lambertian

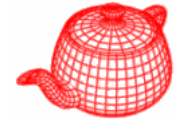


# Oren-Nayer model

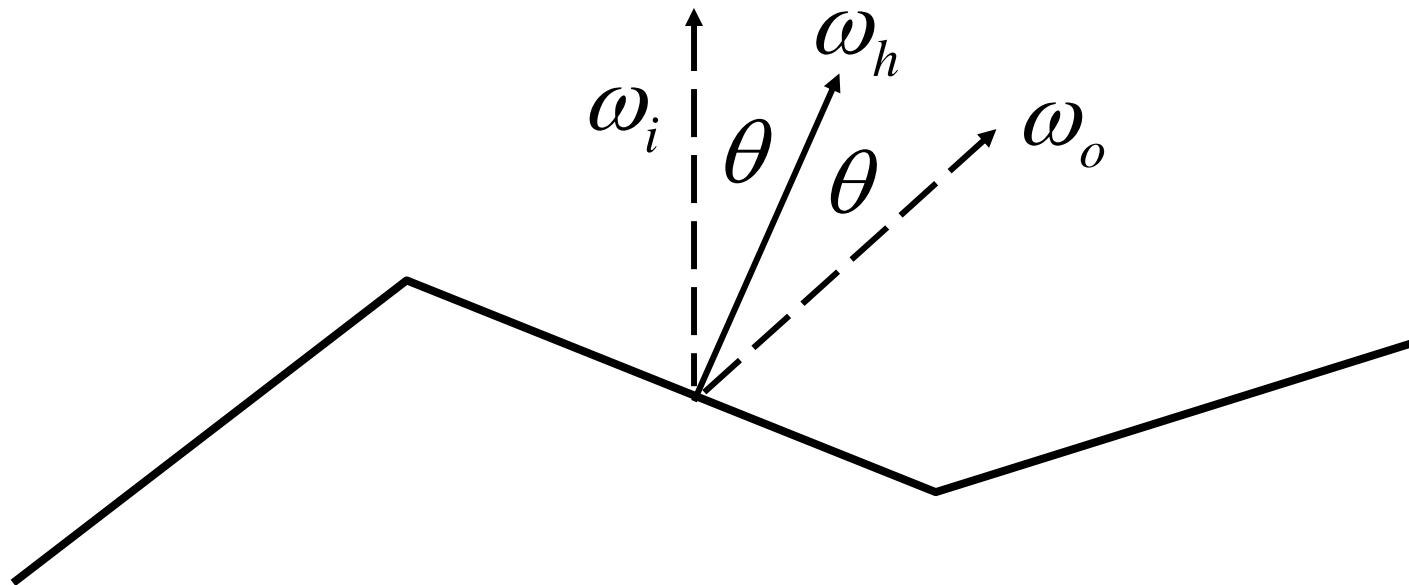
---



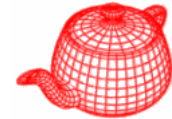
# Torrance-Sparrow model



- One of the first microfacet models, designed to model metallic surfaces
- A collection of perfectly smooth mirrored microfacets with distribution  $D(\omega_h)$

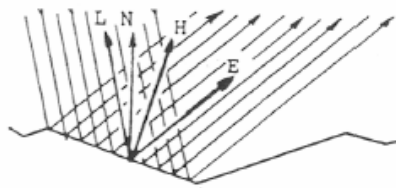


# Torrance-Sparrow model

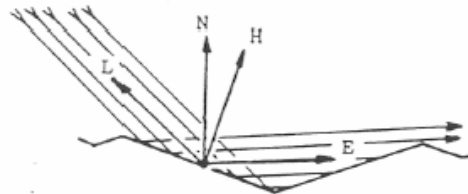


- Microfacet distribution  $D$
- Fresnel reflection  $F$
- Geometric attenuation  $G$

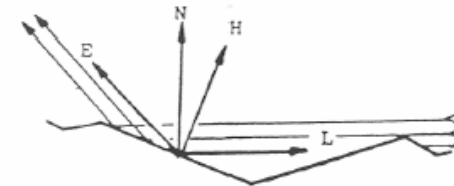
$$f_r(\omega_i, \omega_o) = \frac{D(\omega_h) G(\omega_i, \omega_o) F(\omega_i, \omega_h)}{4 \cos \theta_i \cos \theta_o}$$



$$G = 1$$

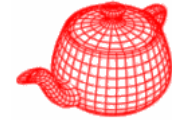


$$G = \frac{2(N \cdot H)(N \cdot \omega_i)}{(H \cdot \omega_i)}$$



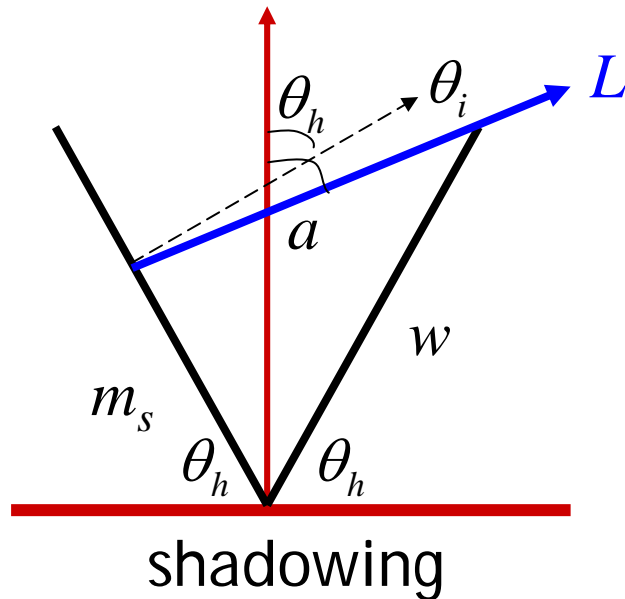
$$G = \frac{2(N \cdot H)(N \cdot \omega_o)}{(H \cdot \omega_o)}$$

# Geometry attenuation factor



$$G = \frac{\text{facet area that is both visible and illuminated}}{\text{total facet area}}$$

$$= \frac{1 \cdot \min(w - m_s, w - m_v)}{1 \cdot w} = \min\left(1 - \frac{m_s}{w}, 1 - \frac{m_v}{w}\right)$$



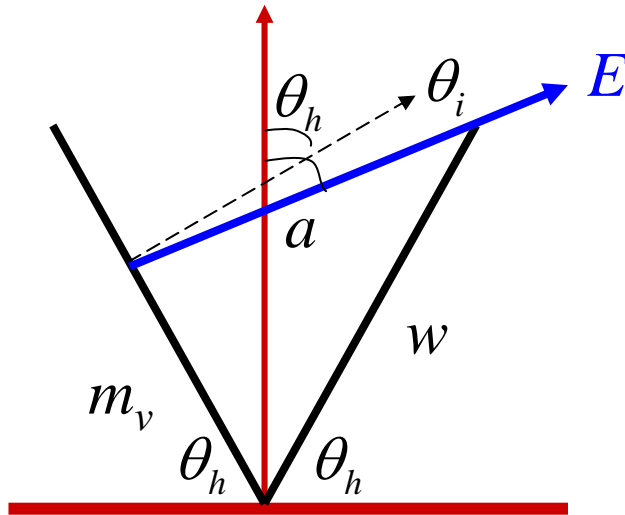
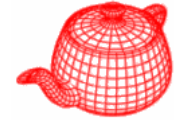
$$a \sin \theta_i = w \cos \theta_h + m_s \cos \theta_h \quad \times \cos \theta_i$$

$$a \cos \theta_i = w \sin \theta_h - m_s \sin \theta_h \quad \times -\sin \theta_i$$

$$\frac{m_s}{w} = -\frac{\cos(\theta_h + \theta_i)}{\cos(\theta_h - \theta_i)}$$

$$1 - \frac{m_s}{w} = \frac{2 \cos \theta_h \cos \theta_i}{\cos(\theta_h - \theta_i)}$$

# Geometry attenuation factor



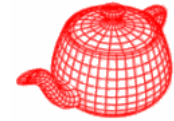
$$1 - \frac{m_v}{w} = \frac{2 \cos \theta_h \cos \theta_o}{\cos(\theta_h - \theta_o)}$$

masking

$$G = \min\left(1 - \frac{m_s}{w}, 1 - \frac{m_v}{w}\right) = \min\left(\frac{2 \cos \theta_h \cos \theta_i}{\cos(\theta_h - \theta_i)}, \frac{2 \cos \theta_h \cos \theta_o}{\cos(\theta_h - \theta_o)}\right)$$

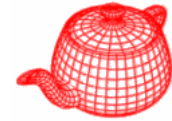
$$G(\omega_o, \omega_i) = \min\left(1, \min\left(\frac{2(n \cdot \omega_h)(n \cdot \omega_o)}{\omega_o \cdot \omega_h}, \frac{2(n \cdot \omega_h)(n \cdot \omega_i)}{\omega_o \cdot \omega_h}\right)\right)$$

# Microfacet model



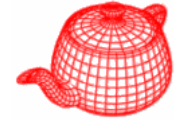
```
class COREDLL MicrofacetDistribution {
public:
    virtual ~MicrofacetDistribution() { }
    virtual float D(const Vector &wh) const=0;
    virtual void Sample_f(const Vector &wo,
        Vector *wi, float u1, float u2,
        float *pdf) const = 0;
    virtual float Pdf(const Vector &wo,
        const Vector &wi) const = 0;
};
```

# Microfacet model



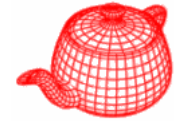
```
class Microfacet : public BxDF {
public:
    Microfacet(const Spectrum &reflectance, Fresnel *f,
               MicrofacetDistribution *d);
    Spectrum f(const Vector &wo, const Vector &wi) const;
    float G(Vector &wo, Vector &wi, Vector &wh) const {
        float NdotWh = fabsf(CosTheta(wh));
        float NdotWo = fabsf(CosTheta(wo));
        float NdotWi = fabsf(CosTheta(wi));
        float WdotWh = AbsDot(wo, wh);
        return min(1.f, min((2.f*NdotWh*NdotWo/WdotWh),
                           (2.f*NdotWh*NdotWi/WdotWh)));
    }
    ...
private:
    Spectrum R;    Fresnel *fresnel;
    MicrofacetDistribution *distribution;
};
```

# Microfacet model



```
Spectrum Microfacet::f(const Vector &wo,
                      const Vector &wi)
{
    float cosTheta0 = fabsf(CosTheta(wo));
    float cosThetaI = fabsf(CosTheta(wi));
    Vector wh = Normalize(wi + wo);
    float cosThetaH = Dot(wi, wh);
    Spectrum F = fresnel->Evaluate(cosThetaH);
    return R * distribution->D(wh)
           * G(wo, wi, wh) * F
           / (4.f * cosThetaI * cosTheta0);
}
```

# Blinn microfacet distribution



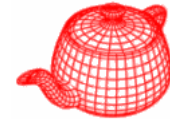
- Distribution of microfacet normals is modeled by an exponential falloff

$$D(\omega_h) \propto (\omega_h \cdot n)^e = (\cos \theta_h)^e$$

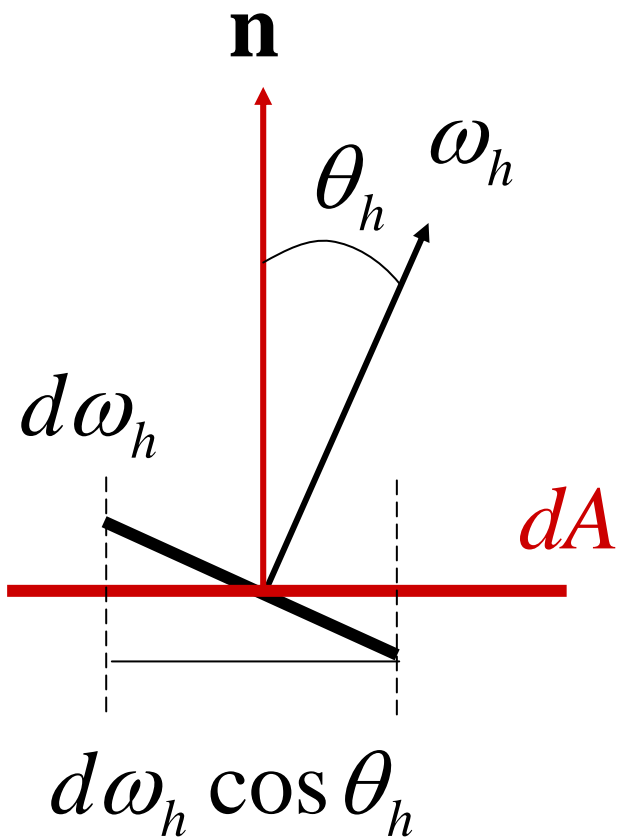
- For smooth surfaces, this falloff happens very quickly; for rough surfaces, it is more gradual.
- Microfacet distribution must be normalized to ensure that they are physically plausible. The projected area of all microfacet faces over some area  $dA$ , the sum should be  $dA$ .

$$\int_{\Omega} D(\omega_h) \cos \theta_h d\omega_h = 1$$

# Blinn microfacet distribution



$$\int_{\Omega} D(\omega_h) \cos \theta_h d\omega_h = 1 \quad \int_{\Omega} c(\omega_h \cdot \mathbf{n})^e \cos \theta_h d\omega_h = 1$$



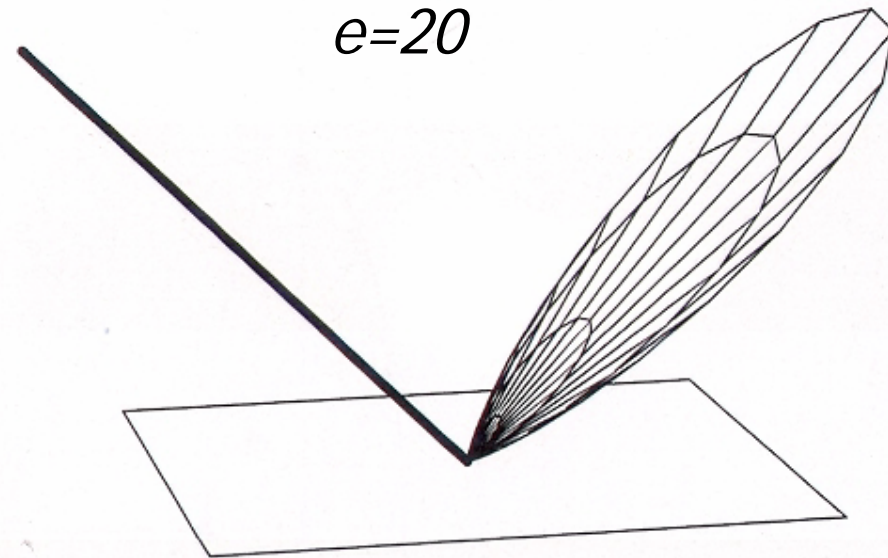
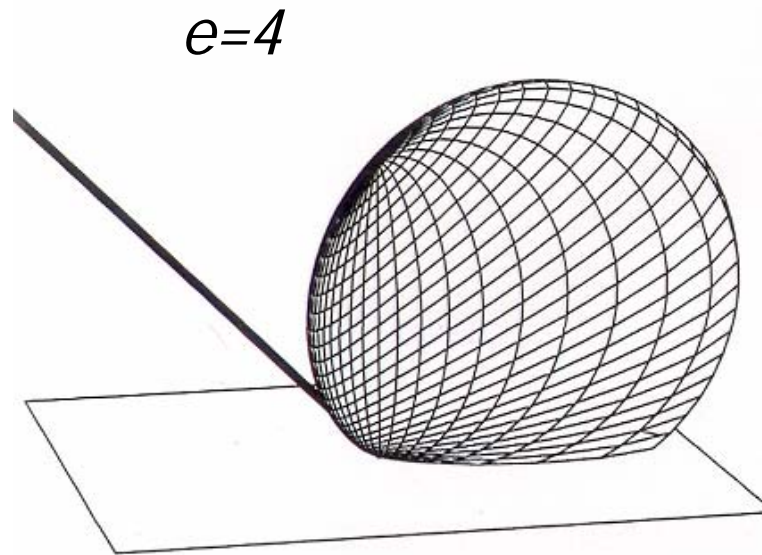
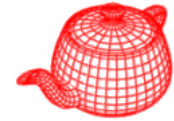
$$\int_0^{2\pi} \int_0^{\frac{\pi}{2}} c(\cos \theta_h)^{e+1} \sin \theta_h d\theta_h d\phi_h = 1$$

$$2\pi c \int_0^{\frac{\pi}{2}} (\cos \theta_h)^{e+1} (-d \cos \theta_h) = 1$$

$$-2\pi c \frac{(\cos \theta_h)^{e+2}}{e+2} \Big|_{\cos \theta_h=1}^{\cos \theta_h=0} = 1$$

$$c = \frac{e+2}{2\pi} \quad D(\omega_h) = \frac{e+2}{2\pi} (\omega_h \cdot \mathbf{n})^e$$

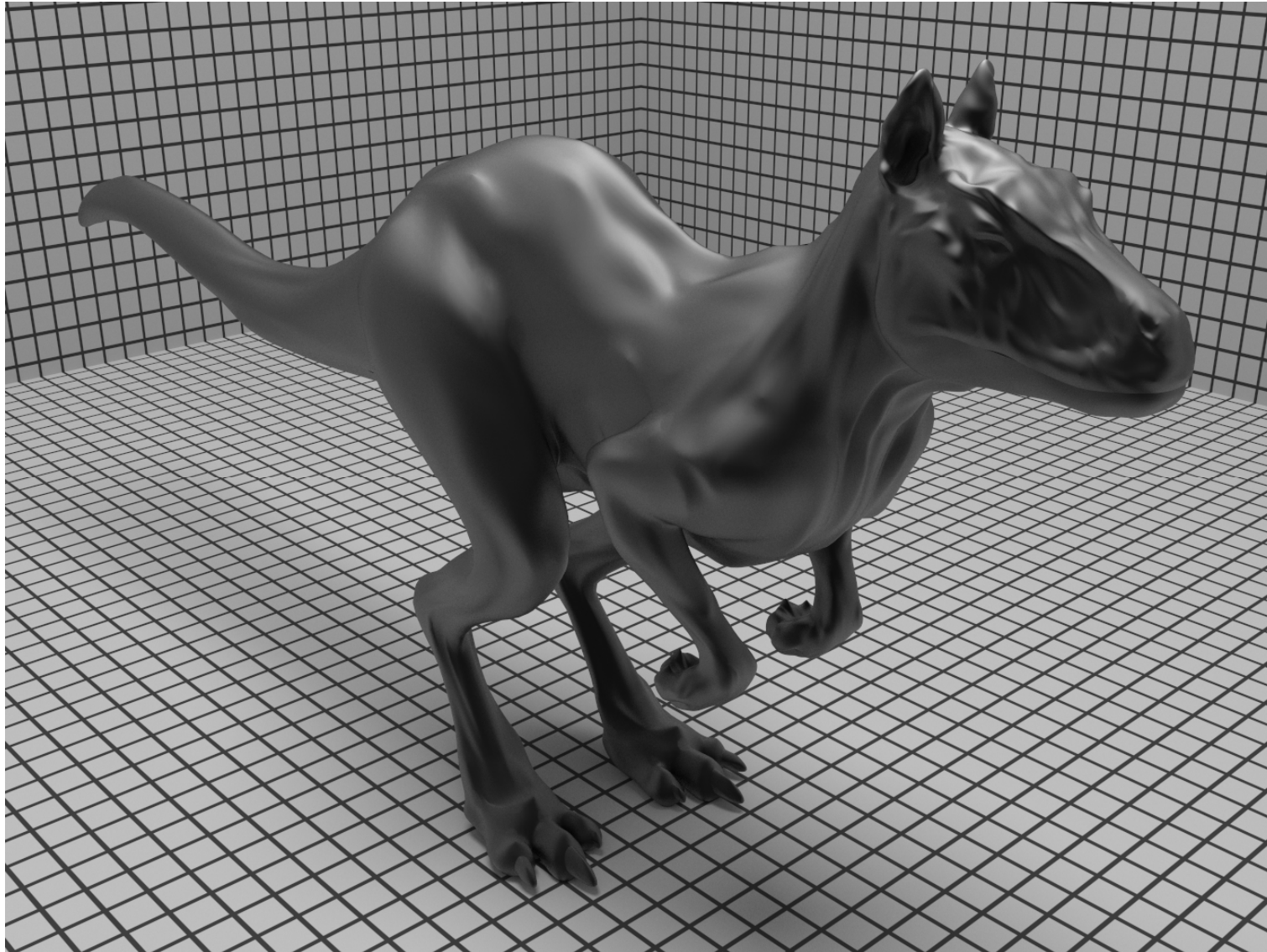
# Blinn microfacet distribution



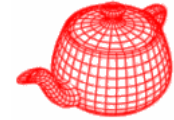
```
class Blinn : public MicrofacetDistribution
{
    ...
    float Blinn::D(const Vector &wh) const {
        float costhetah = fabsf(CosTheta(wh));
        return (exponent+2) * INV_TWOPI *
            powf(max(0.f, costhetah), exponent);
    }
}
```

# Torrance-Sparrow with Blinn distribution

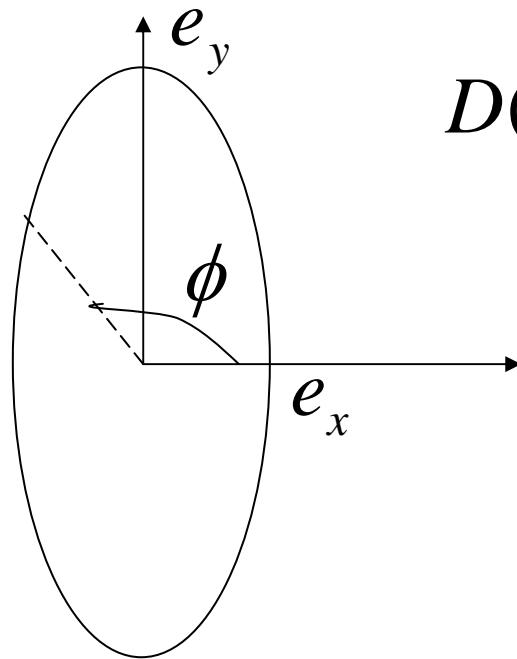
---



# Anisotropic microfacet model

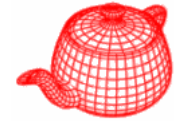


- Blinn microfacet model is radially symmetric (only depending on  $\theta_h$ ); hence, it is isotropic.
- Ashikmin and Shirley have developed a microfacet model for anisotropic surfaces



$$D(\omega_h) \propto (\omega_h \cdot n)^{e_x \cos^2 \phi + e_y \sin^2 \phi}$$

# Ashikmin-Shirley model



$$\int_{\Omega} c(\omega_h \cdot \mathbf{n})^{e_x \cos^2 \phi + e_y \sin^2 \phi} \cos \theta_h d\omega_h = 1$$

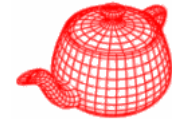
$$\int_0^{2\pi} \int_0^{\pi/2} c(\cos \theta_h)^{e_x \cos^2 \phi_h + e_y \sin^2 \phi_h + 1} \sin \theta_h d\theta_h d\phi_h = 1$$

$$c \int_0^{2\pi} \int_0^{\pi/2} (\cos \theta_h)^{e_x \cos^2 \phi_h + e_y \sin^2 \phi_h + 1} d \cos \theta_h d\phi_h = -1$$

$$c \int_0^{2\pi} \frac{(\cos \theta_h)^{e_x \cos^2 \phi_h + e_y \sin^2 \phi_h + 2}}{e_x \cos^2 \phi_h + e_y \sin^2 \phi_h + 2} \Big|_1^0 d\phi_h = -1$$

$$c \int_0^{2\pi} \frac{1}{e_x \cos^2 \phi_h + e_y \sin^2 \phi_h + 2} d\phi_h = 1$$

# Ashikmin-Shirley model



$$c \int_0^{2\pi} \frac{1}{e_x \cos^2 \phi_h + e_y \sin^2 \phi_h + 2} d\phi_h = 1$$

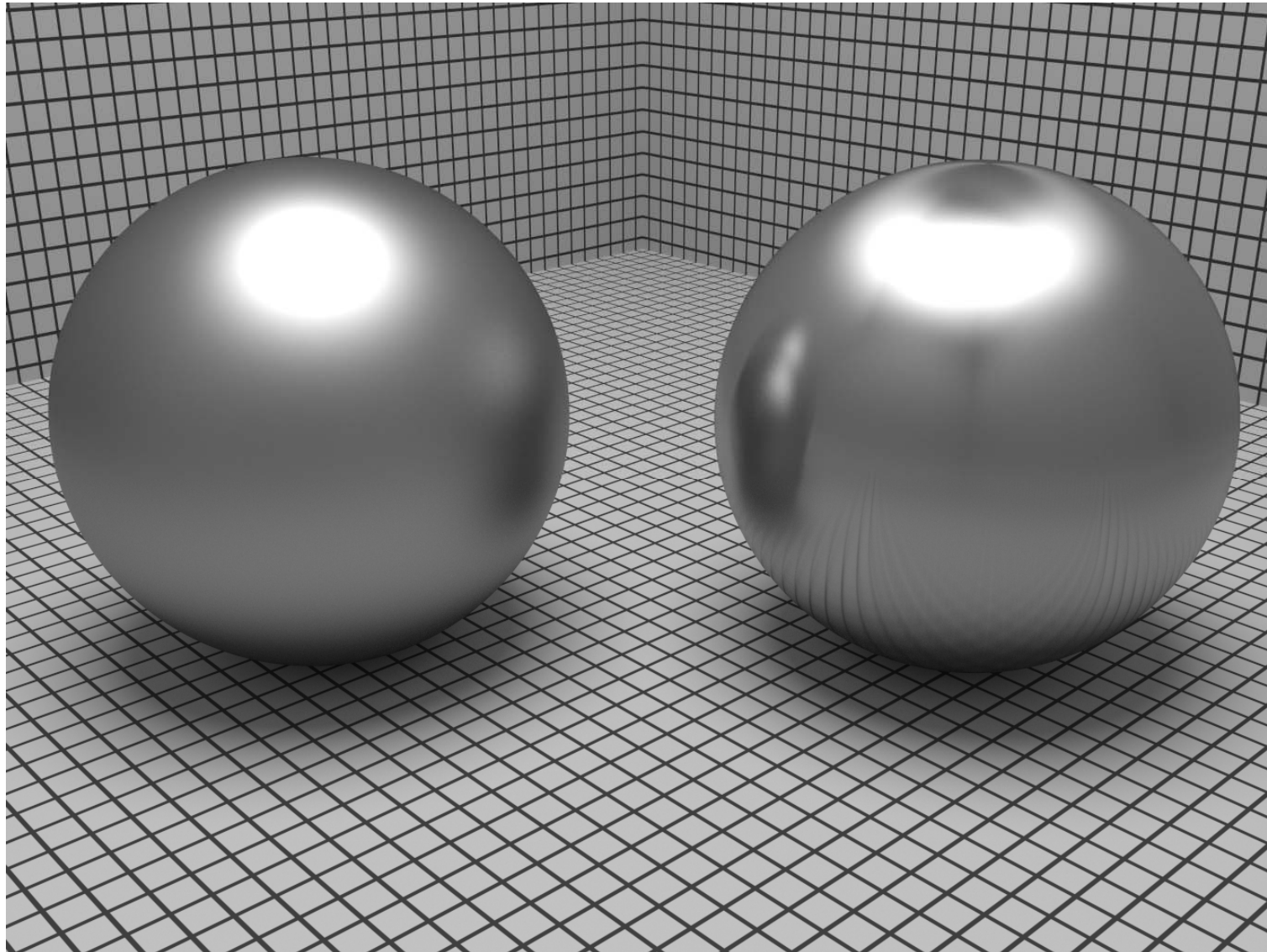
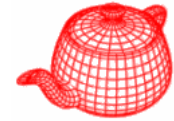
$$\int \frac{1}{a \cos^2(x) + b \sin^2(x) + 2} dx = \frac{\tan^{-1}\left(\frac{\sqrt{b+2} \tan(x)}{\sqrt{a+2}}\right)}{\sqrt{a+2} \sqrt{b+2}}$$

$$c \frac{2\pi}{\sqrt{e_x + 2} \sqrt{e_y + 2}} = 1$$

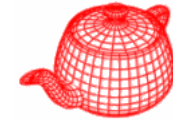
$$D(\omega_h) = \frac{\sqrt{(e_x + 2)(e_y + 2)}}{2\pi} (\omega_h \cdot n)^{e_x \cos^2 \phi + e_y \sin^2 \phi}$$

# Anisotropic microfacet model

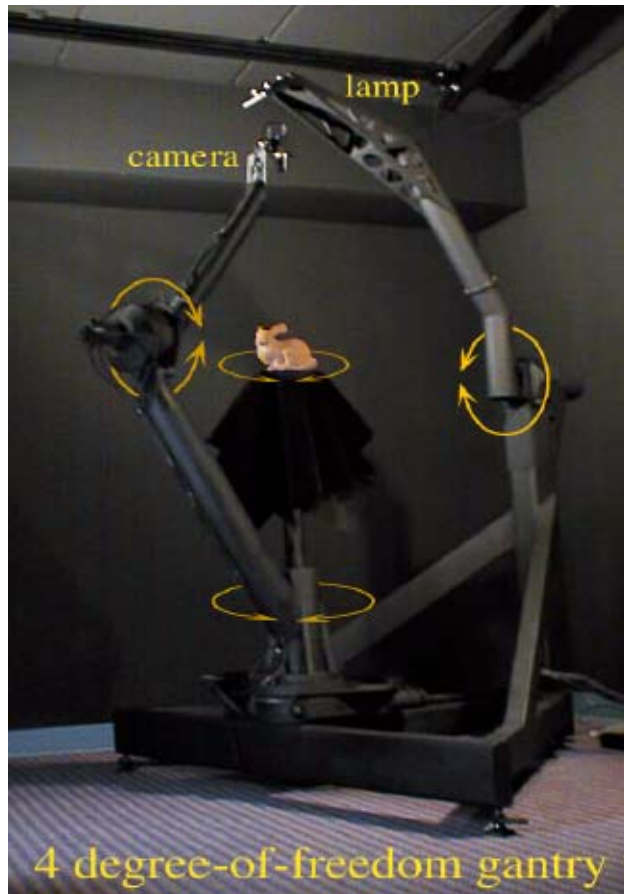
---



# Lafortune model



- An efficient BRDF model to fit measured data to a parameterized model with a relatively small number of parameters



modified Phong model

$$f_r(p, \omega_o, \omega_i) = (\omega_o \cdot R(\omega_i, \mathbf{n}))^e$$

Lafortune model

$$f_r(p, \omega_o, \omega_i) = \frac{\rho_d}{\pi} + \sum_{i=1}^n (\omega_o \cdot (\omega_{ix} o_{i,x}, \omega_{iy} o_{i,y}, \omega_{iz} o_{i,z}))^{e_i}$$

# Lafortune model (for a measured clay)

---

