BRDF models

Implementing the BRDF

The BRDF used for scan line rendering does not fulfill the energy conservation property in most of the cases.

Generally, the BRDF is expressed as the sum of two terms:

- *The diffuse reflection*
- *The specular reflection*

$$f_r(x, \overrightarrow{lx}, \omega_r) = f_{diffuse}(x, \overrightarrow{lx}, \omega_r) + f_{specular}(x, \overrightarrow{lx}, \omega_r)$$

Implementing the BRDF

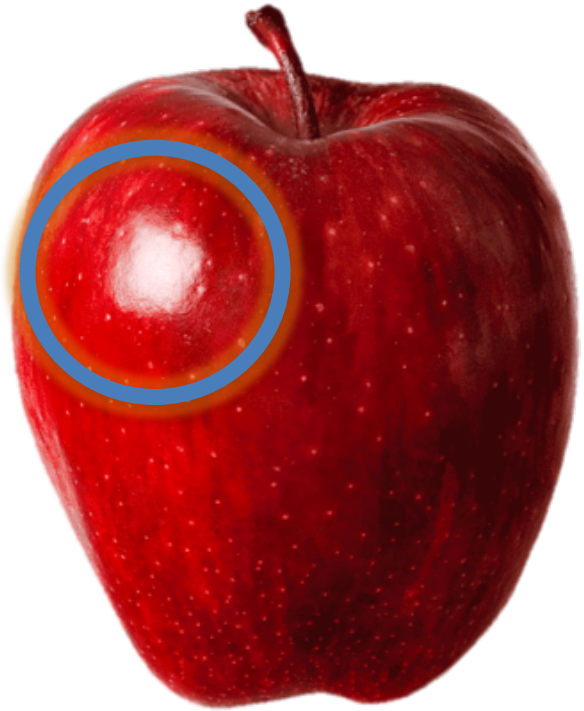
The *diffuse component* of the BRDF represents the main color of the object.

$$f_{diffuse}(x, \overrightarrow{lx}, \omega_r)$$

Implementing the BRDF

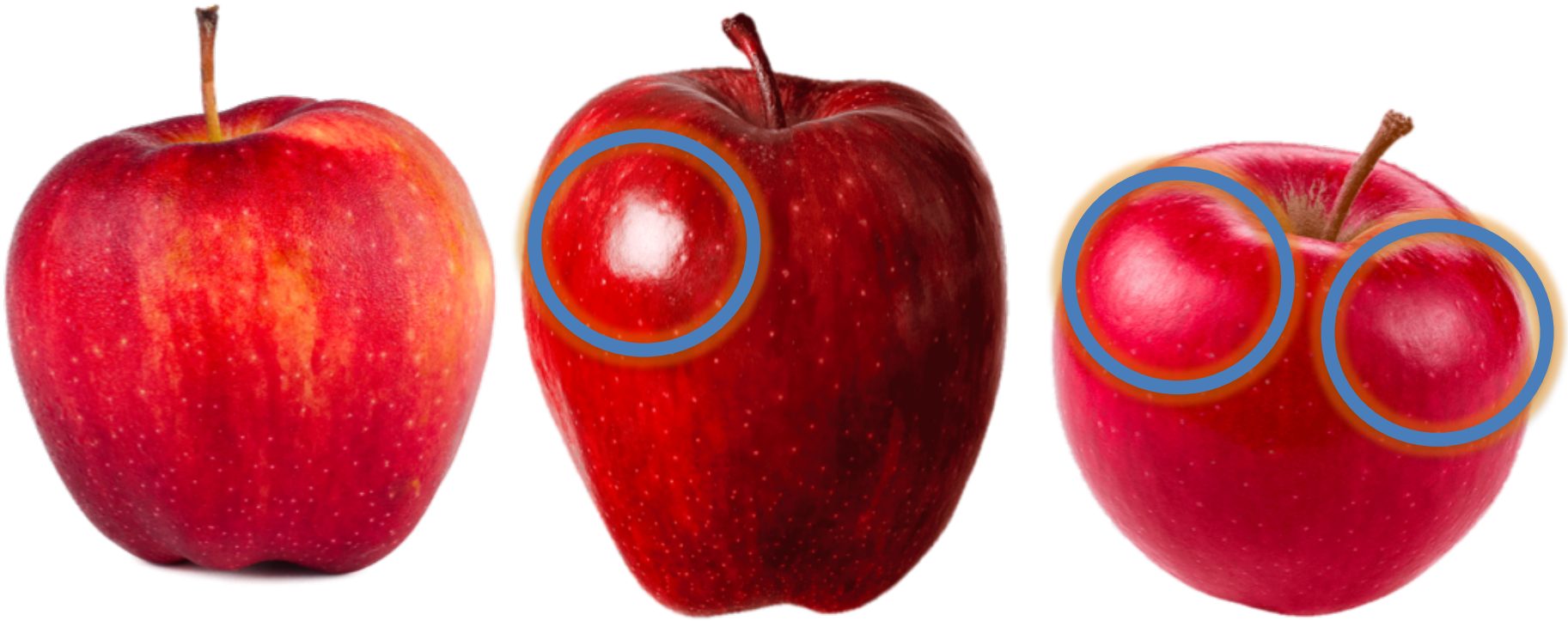
Shiny objects tend to reflect the incoming light in a particular angle, called *the specular direction*, which depends on the direction from which the object is seen ω_r. This effect is implemented in the *specular component* of the BRDF.

$$f_{specular}(x, \overrightarrow{lx}, \omega_r)$$



Implementing the BRDF

In general, the number and shape of highlights matches the one of the direct light sources in the scene.



Color range

In scan-line rendering, values of the BRDF for each color frequency, and for both the diffuse and specular components, are in the range [0,1].

$$f_{diffuse}(x, \overrightarrow{lx}, \omega_r) \in [0,1]$$
$$f_{specular}(x, \overrightarrow{lx}, \omega_r) \in [0,1]$$

Color range

However, due to the influence of lights the final color of the pixel can produce values that are larger than *1*.

$$\sum_{l} L(l, \overrightarrow{lx}) f_r(x, \overrightarrow{lx}, \omega_r) > 1$$

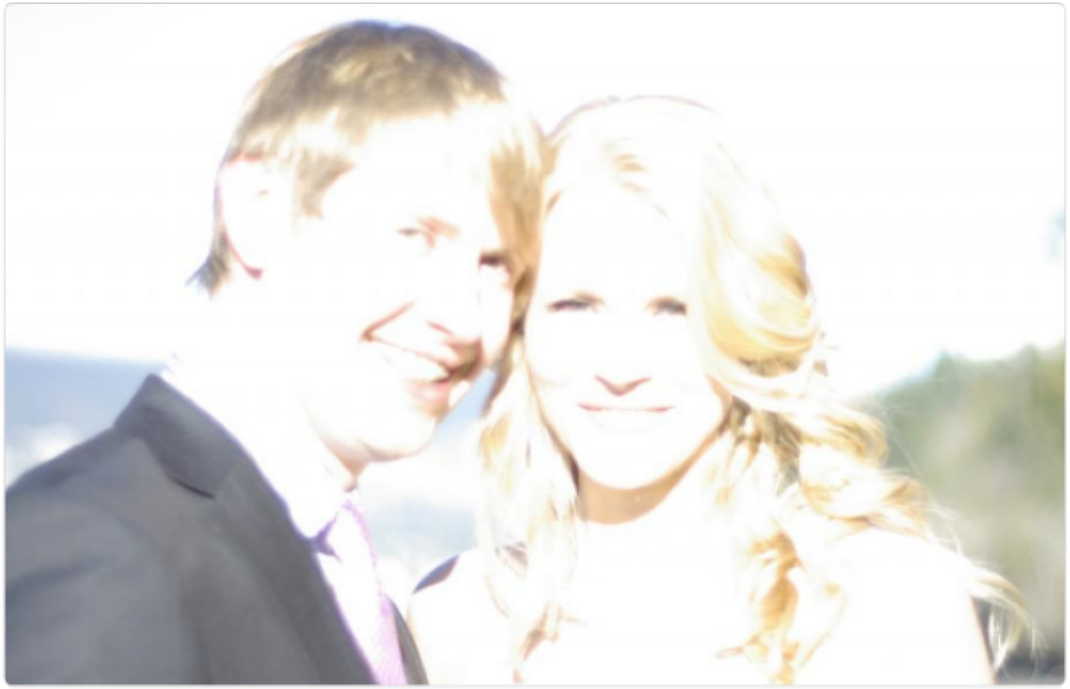
The solution applied is to clamp the values in the range *[0,1]* at the end of the computation:

$$L(x, \omega_r) = clamp\left(\sum_{l} L(l, \overrightarrow{lx}) f_r(x, \overrightarrow{lx}, \omega_r)\right)$$

$$clamp(y) = \begin{cases} 0 & y < 0 \\ y & y \in [0,1] \\ 1 & y > 1 \end{cases}$$

Color range

Although not physically correct, it creates effects that are similar to overexposure in photography, which represents area that receives too much light with white.



High Dynamic Range (HDR)

Newer rendering techniques, perform more advanced computation, allowing values outside the [0,1] range.

$$f_{diffuse}(x, \overrightarrow{lx}, \omega_r) \geq 0$$
$$f_{specular}(x, \overrightarrow{lx}, \omega_r) \geq 0$$

The final color is then mapped into the [0,1] range at the end of the process, using suitable non-linear functions.

$$L(x, \omega_r) = g(L'(x, \omega_r)) \qquad \text{where: } L'(\dots) \geq 0, and\ L(\dots) \epsilon [0,1]$$

High Dynamic Range (HDR)

This allows to consider lager color dynamics, leading to images that can have details in both dark and extremely lit areas.

HDR, however, requires much more memory (4x) and larger computation power to apply the final non-linear scaling function.

Diffuse and specular models

In this course we will present the following diffuse reflection models:

- Lambert
- Oren-Nayar
- Toon

And the following specular reflection models:

- Phong specular reflection
- Blinn specular reflection
- Ward
- Cook-Torrance specular reflection
- Toon

We will present them in order of implementation complexity.

The Lambert reflection

The simplest BRDF has only the diffuse part, which corresponds to a constant term.

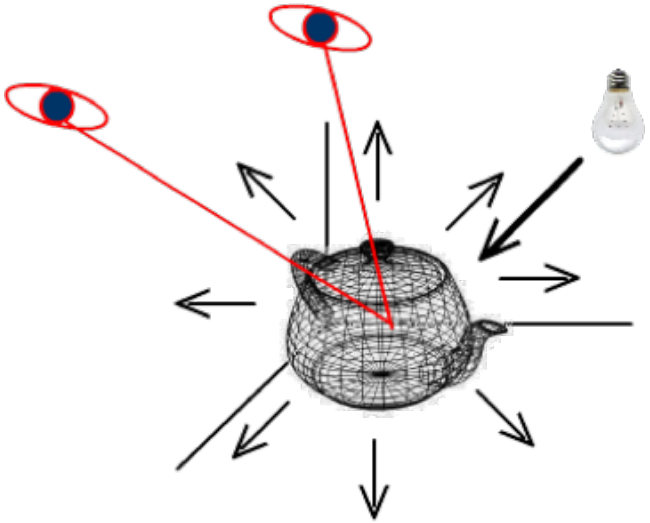
This is the BRDF used in the *Radiosity* technique previously introduced.

Constant BRDF causes a shading phenomenon know as *Lambert diffusion*.

The Lambert reflection

According to the reflection law by Lambert, each point of an object hit by a ray of light, reflects it with uniform probability in all the directions.

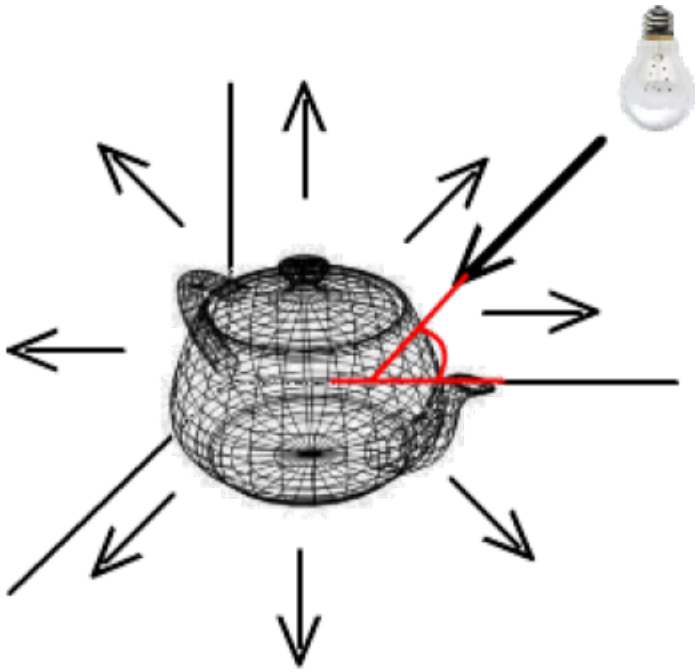
The reflection is thus independent of the viewing angle and it corresponds to a constant BRDF.



$$f_r(x, \omega_i, \omega_r) = \rho_x$$

The Lambert reflection

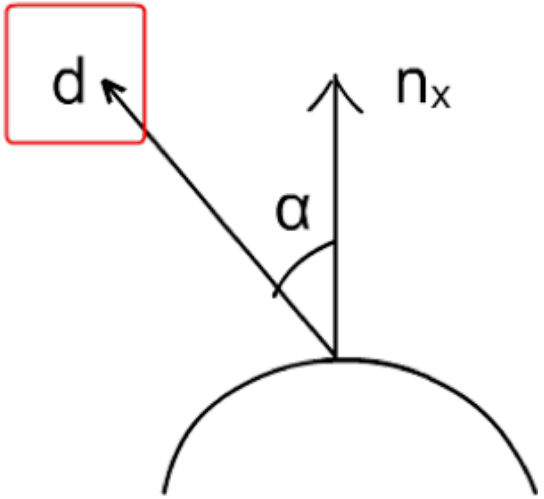
The quantity of light received by an object is however not constant: it depends on the angle between the ray of light and reflecting surface.



The Lambert reflection

Let us call n_x the unitary normal vector to the surface.

- Let us also call ***d*** the direction of the ray of light,
- and α the angle between the two vectors.

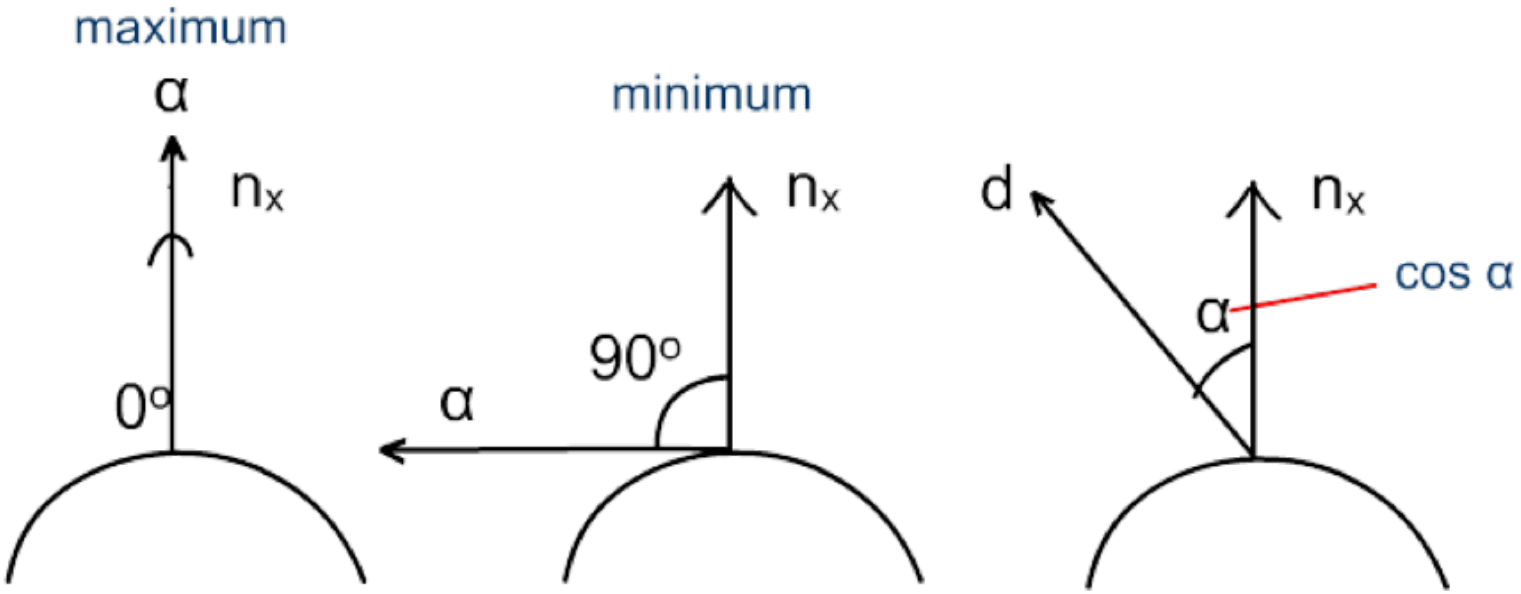


Let us also remember that the light model returns a unitary vector that is directed from the object to the light.

The Lambert reflection

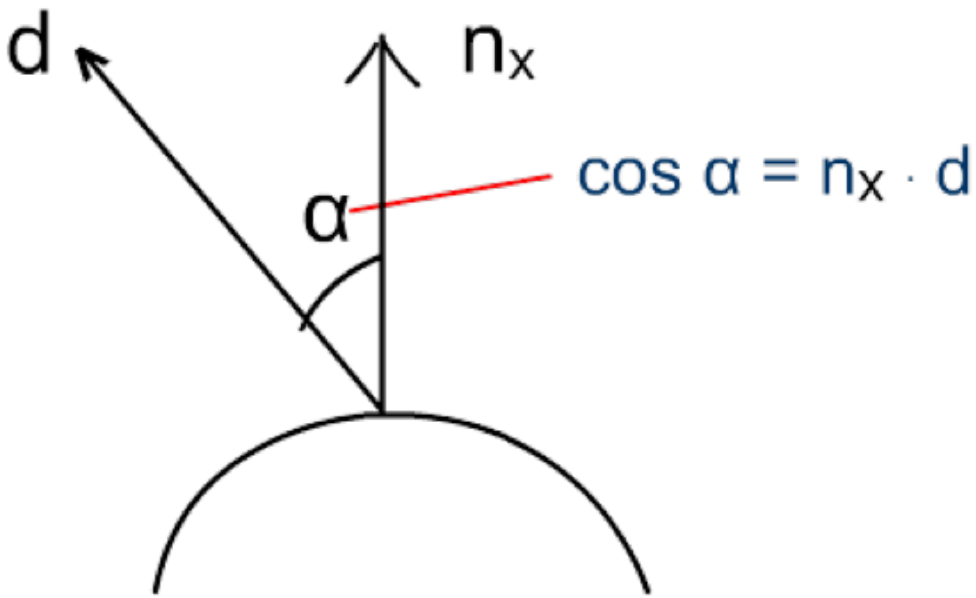
The incidence of the incoming light is maximized when angle α is 0^o, and null if it is greater or equal than 90^o.

In particular, Lambert has shown that the amount of light reflected is proportional to $\cos \alpha$.



The Lambert reflection

Thanks to the geometric properties of the scalar product, cos α can be computed as the dot product of the vectors corresponding to the normal vector n_x, and to the direction of light ***d***.



The Lambert reflection

Let us call m_D a vector that expresses the capability of a material to perform the Lambert reflection for each of the three primary color frequencies RGB (that is $\rho_x = m_D$, the constant assigned to the BRDF function).

$$m_D = (m_R, m_G, m_B)$$

We can express the BRDF of the Lambert reflection for scan-line rendering with the following expression:

$$f_r\left(x, \overrightarrow{lx}, \omega_r\right) = f_{diffuse}\left(x, \overrightarrow{lx}\right) = \boldsymbol{m}_D \cdot max\left(\overrightarrow{lx} \cdot \boldsymbol{n}_x, 0\right)$$

Notice the use of the *max()* function to limit the values of the Lambert diffuse term to be positive.

The Lambert reflection

When a face is on the opposite direction with respect to a light source, it cannot be illuminated despite of the angle.

For back faces the cosine is negative: clamping the value at zero simply avoids illuminating the faces (otherwise the formula would subtract light from the object).

Since Lambert reflection model only includes the diffuse component of lighting, vector m_D is usually called the *diffuse color* of the object, which represents its main color.

The Lambert reflection

Note also that the pixel color does not depend on ω_r: in other words when just Lambert diffuse reflection is used, the viewing angle has no effect, and the final image depends only on the position of the objects and on the direction the lights.

Below you can see the rendering equations for the cases in which only a single direct or point light (with no decay) are used.

$$L(x, \omega_r) = \boldsymbol{l} * \boldsymbol{m}_D \cdot clamp(\boldsymbol{d} \cdot \boldsymbol{n}_x)$$

Direct light

$$L(x, \omega_r) = \boldsymbol{l} * \boldsymbol{m}_D \cdot clamp\left(\frac{\boldsymbol{p} - \boldsymbol{x}}{|\boldsymbol{p} - \boldsymbol{x}|} \cdot \boldsymbol{n}_x\right)$$

Point light (with no decay)

Specular reflection

Specular reflection can be considered by adding the specular term to the diffuse one.

$$f_r\left(x, \overrightarrow{lx}, \omega_r\right) = f_{diffuse}\left(x, \overrightarrow{lx}\right) + f_{specular}\left(x, \overrightarrow{lx}, \omega_r\right)$$

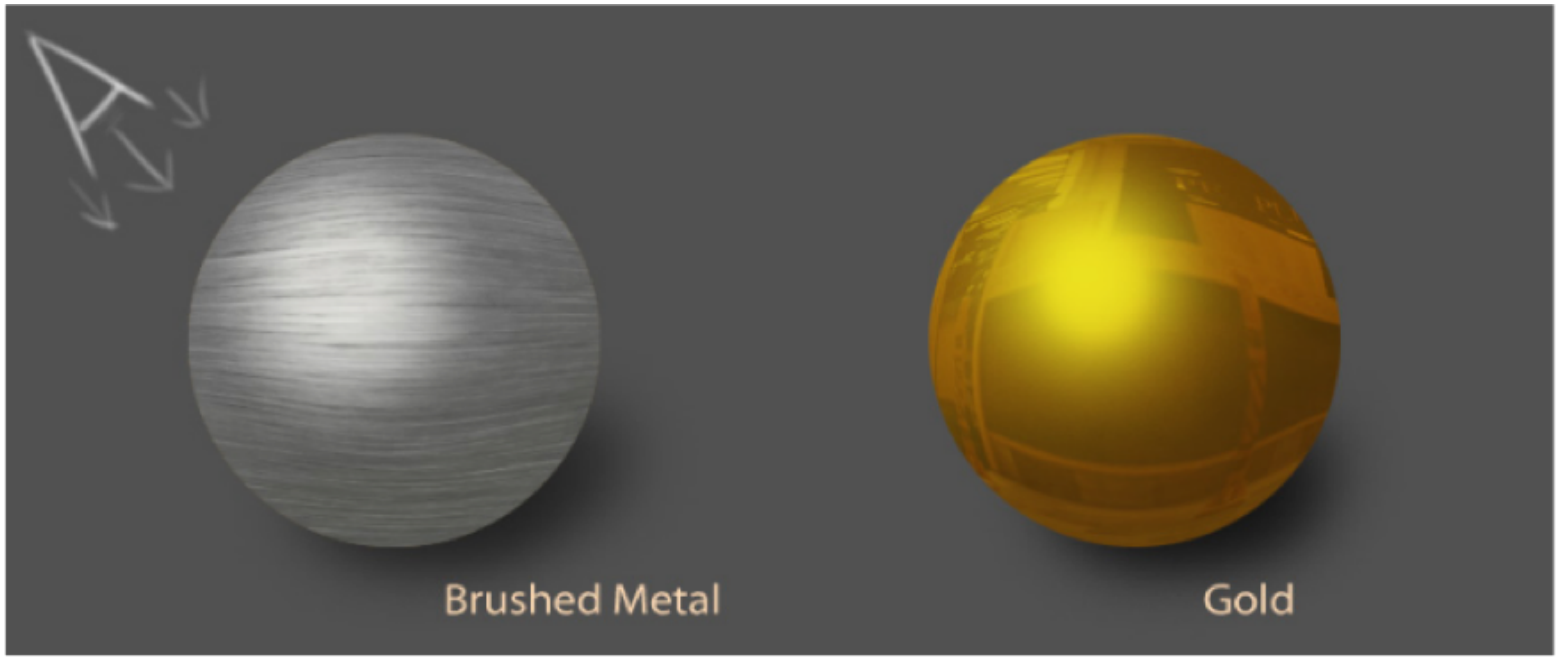
As for the diffuse case, the specular component is characterized by a color m_S that defines how the RGB components of the incoming light are reflected.

$$m_S = (m_{Rs}, m_{Gs}, m_{Bs})$$

Specular reflection

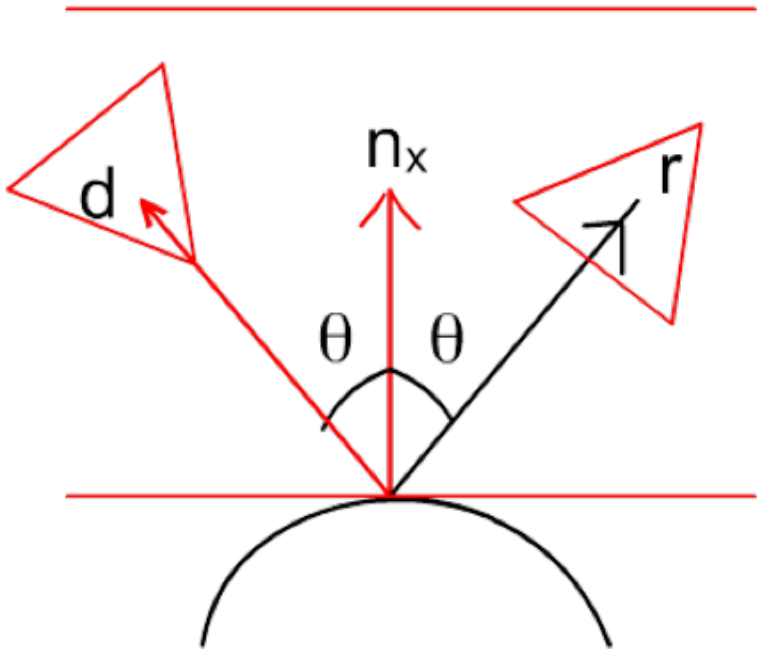
In most of the objects, the specular color is white, that is $m_S = (1,1,1)$.

However, metallic objects such as gold or copper have the specular color identical to their diffuse one.



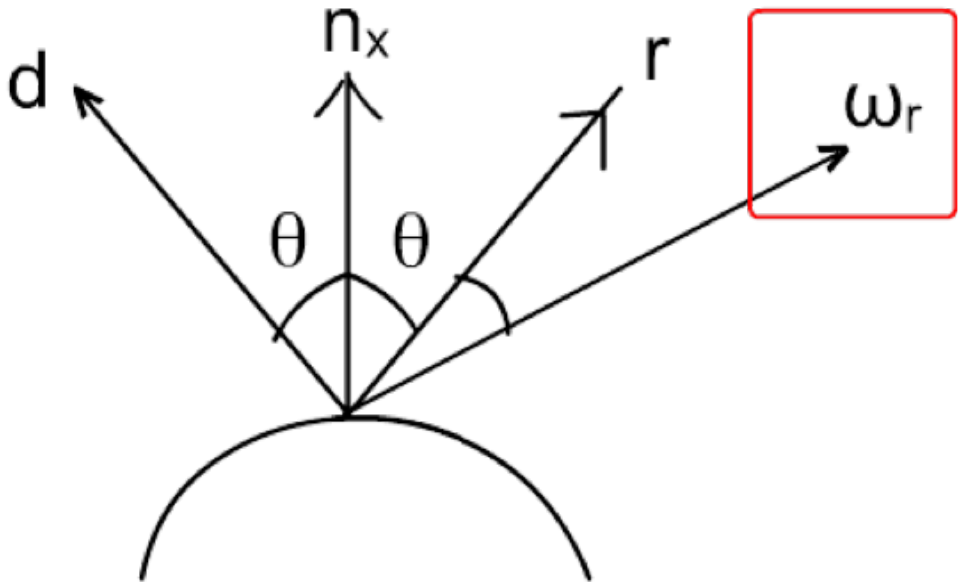
The Phong reflection model

In the Phong model, the specular reflection has the same angle as the incoming ray with respect to the normal vector, but it is oriented in the opposite direction, and it is positioned on the same plane as the light and the normal vectors.



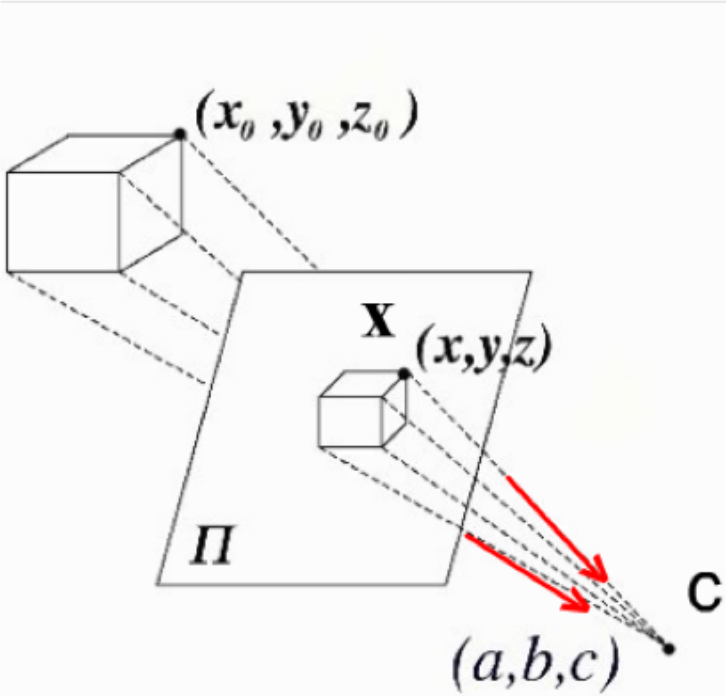
The Phong reflection model

Let's recall that we addressed with ω_r the vector that points in the direction from which the object is being observed in the BRDF:



The Phong reflection model

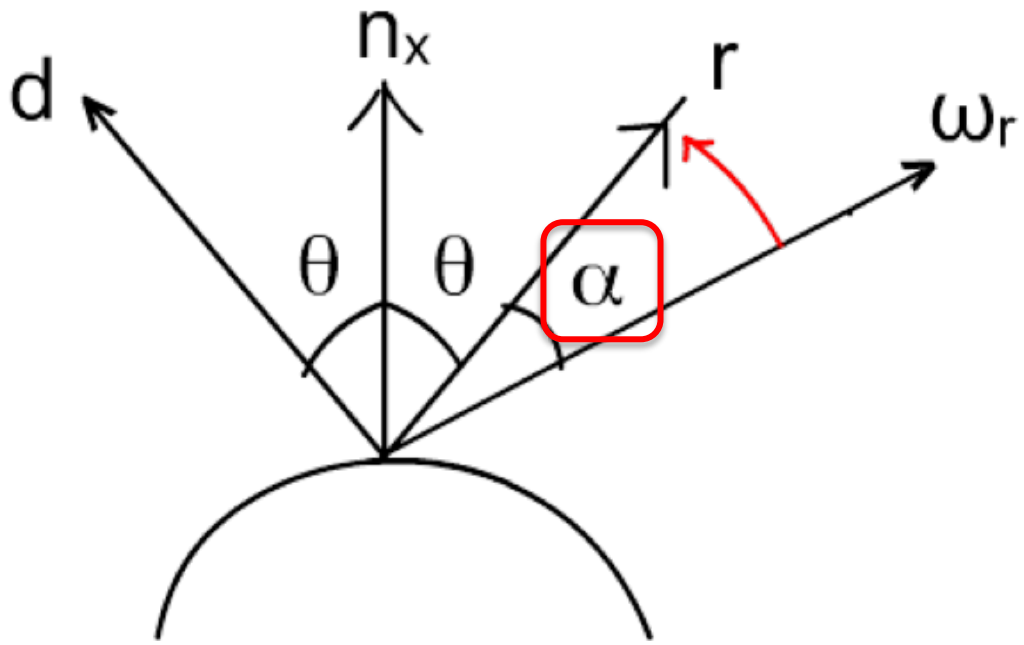
For parallel projection, it is constant. For perspective, ω_r can be computed as the normalized difference between the point on the surface *x* and the center of projection *c*.



$$\omega_r = \frac{\boldsymbol{c} - \boldsymbol{x}}{|\boldsymbol{c} - \boldsymbol{x}|}$$

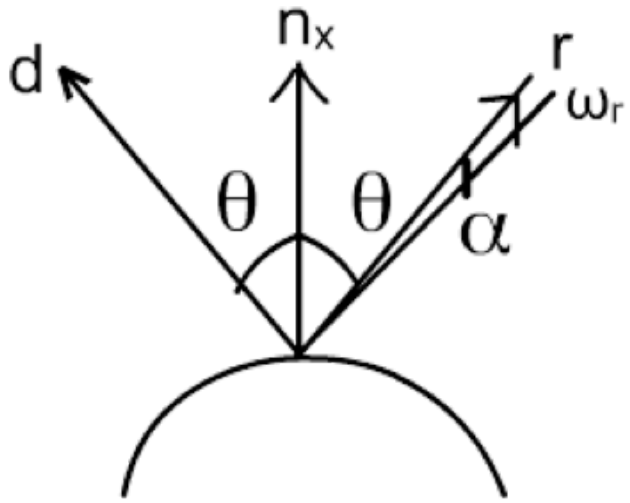
The Phong reflection model

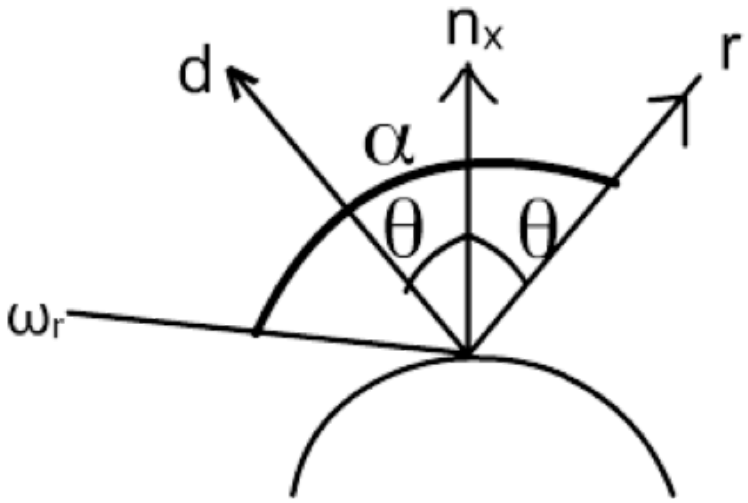
The Phong specular reflection model, accounts for the angular distance α between the specular direction and the observer.



The Phong reflection model

The Phong model computes the intensity of the specular reflection from *cos* α: in this way the term is maximum if the specular direction is aligned with the observer, and zero when the angle is greater than *90º*.





The Phong reflection model

To create more contained highlight regions, the term *cos* α is raised at a power γ. The greater is γ, the smaller is the highlight, and the *more shiny* the object appears to be.

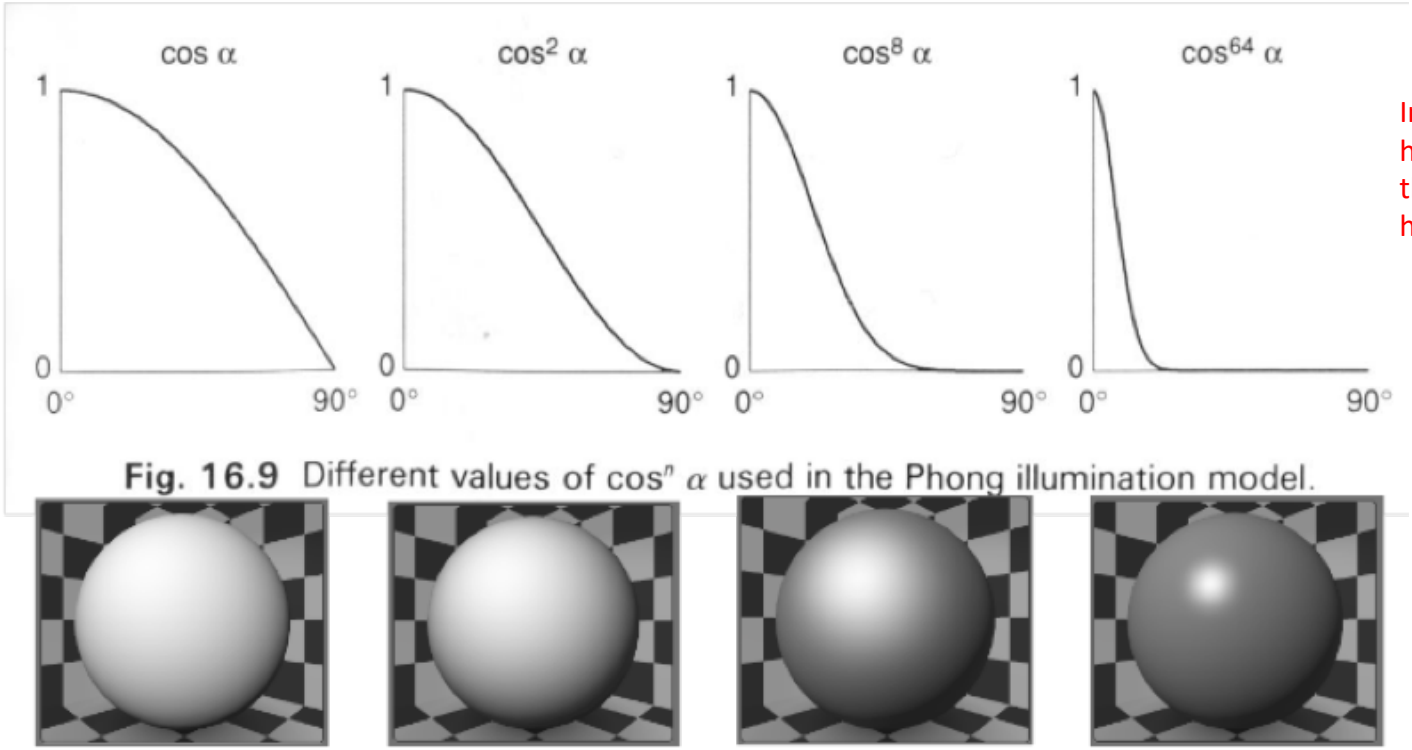


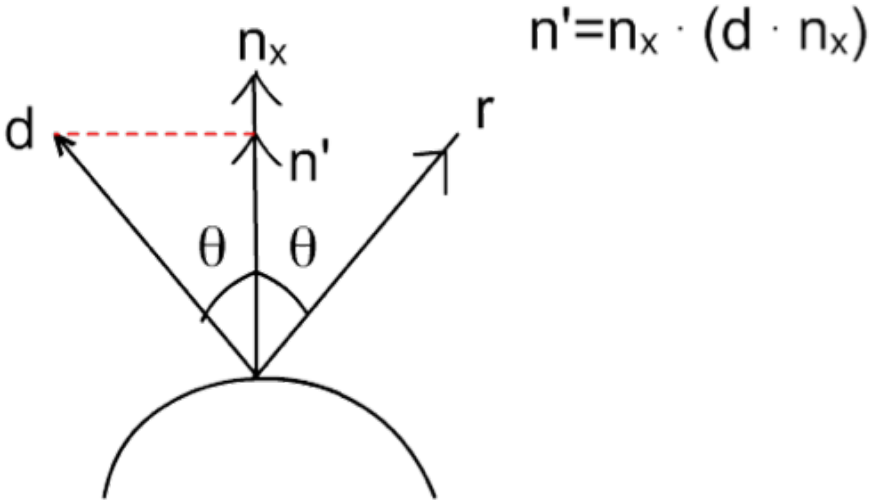
Fig. 16.9 Different values of $\cos^n \alpha$ used in the Phong illumination model.

In many practical cases, γ has very high values, in the order of one or two hundreds.

The Phong reflection model

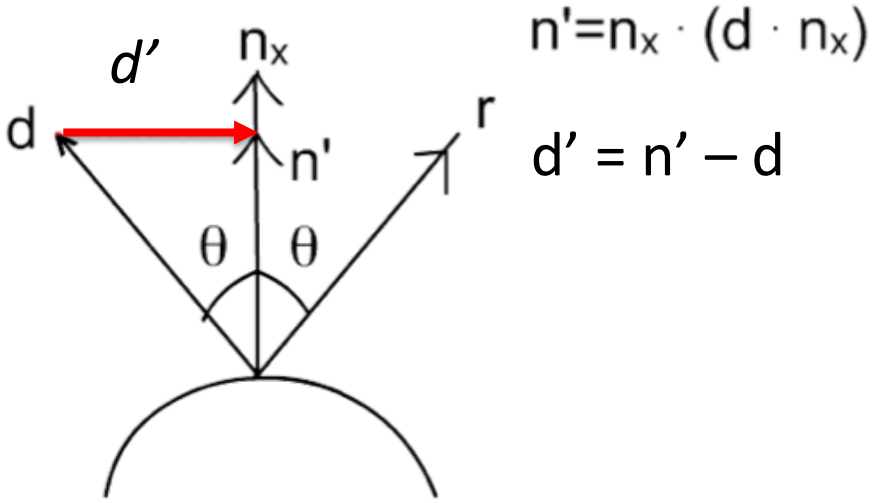
To compute the direction of the reflected ray, first we compute *n'*, the projection of the light vector over the normal vector.

This is performed by first computing its length with the dot product between the normal and the light direction $(d \cdot n_x)$, and then multiplying the result with the normal vector n_x to make it a vector in the perpendicular direction.



The Phong reflection model

Then if we subtract *n'* from the light vector, we obtain *d'* the perpendicular from *d* to *n*.

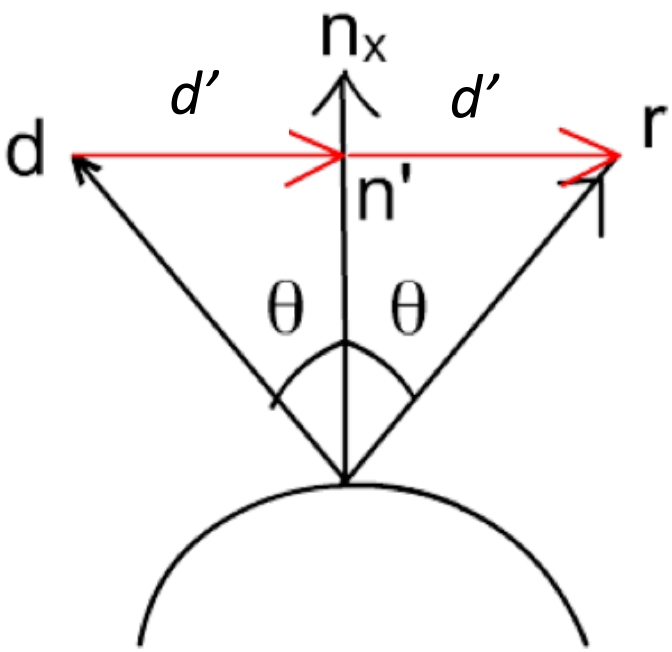


$$n' = n_x \cdot (d \cdot n_x)$$

$$d' = n' - d$$

The Phong reflection model

If we add *d'* two times to *d*, we obtain the reflected vector *r*.



$$n' = n_x \cdot (d \cdot n_x)$$

$$d' = n' - d$$

$$r = d + 2d'$$

The Phong reflection model

To summarize, we have:

$$r = d + 2(n_x \cdot (d \cdot n_x) - d) = 2\,(d \cdot n_x)\, n_x - d$$

If we use the notation of the rendering equation, we obtain:

$$\boldsymbol{r}_{l,x} = 2\left(\overrightarrow{xl} \cdot \boldsymbol{n}_x\right)\boldsymbol{n}_x - \overrightarrow{xl}$$

Note also that many shading languages have built-in functions to directly compute the reflected vector. For example, in GLSL:

$$\boldsymbol{r}_{l,x} = -reflect\left(\overrightarrow{xl}, \boldsymbol{n}_x\right)$$

Note that the reflection happens in the opposite direction. For this reason the minus sign is required.

The Phong reflection model

We can then compute the intensity of the specular reflection term as:

$$\cos^{\gamma} \alpha = clamp(\omega_r \cdot \boldsymbol{r})^{\gamma}$$

As for the Lambert diffuse term, it is necessary to exclude the cases in which the cosine is negative using the *clamp()* function.

To summarize we have:

$$\boldsymbol{r}_{l,x} = 2\boldsymbol{n}_x \cdot \left(\overrightarrow{lx} \cdot \boldsymbol{n}_x\right) - \overrightarrow{lx}$$

$$f_{specular}\left(x, \overrightarrow{lx}, \omega_r\right) = \boldsymbol{m}_S \cdot clamp\left(\omega_r \cdot \boldsymbol{r}_{l,x}\right)^{\gamma}$$

Simplified parameterization

Most of diffuse and specular light models depends on the direction of the normal vector, and are characterized by a main color.

For this reason, in most of the cases, the BRDF components can be expressed as:

$$f_D(\boldsymbol{l}, \boldsymbol{n}, \boldsymbol{v}, m_D) \qquad f_S(\boldsymbol{l}, \boldsymbol{n}, \boldsymbol{v}, m_S)$$

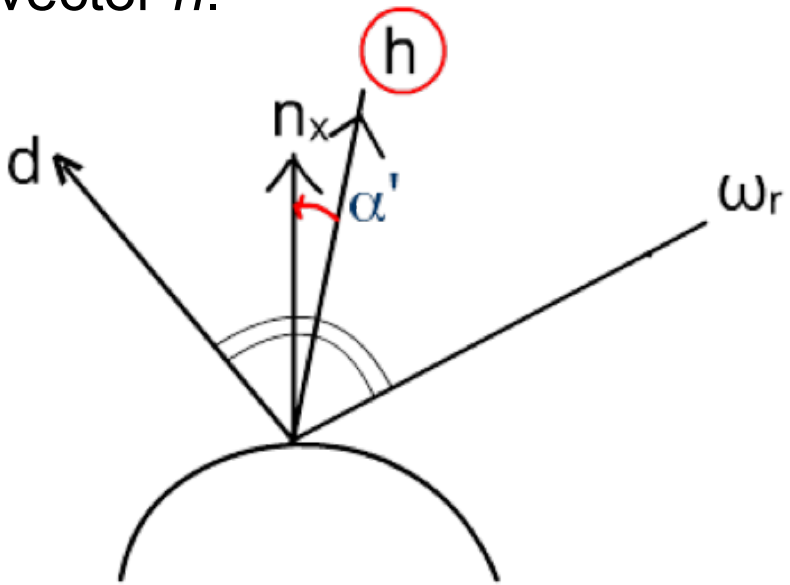
Where:

- $\boldsymbol{l}$ is the direction of the light
- $\boldsymbol{n}$ is the direction of the normal vector
- $\boldsymbol{v}$ is the view direction
- m_D and m_S are the other model specific parameter. For example, for m_D is the diffuse color for the Lambert model, and m_S is the collection of the specular color and the specular power for the Phong model.

The Blinn reflection model

The Blinn reflection model is an alternative to the Phong shading model that uses the *half vector h*: a vector that is in the middle between ω_r and *d*.

The angle α between the observer and the reflected ray, is then approximated by the angle α' between the normal vector n_x and the half vector *h*.



The Blinn reflection model

The half vector *h* can be computed as the normalized average of vectors *d* and ω_r. With the notation used for the rendering equations we have:

$$\boldsymbol{h}_{l,x} = \frac{\overrightarrow{lx} + \omega_r}{|\overrightarrow{lx} + \omega_r|} = normalize(\overrightarrow{lx} + \omega_r)$$

The formula when considering Blinn specular reflection becomes:

$$f_{specular}\left(x, \overrightarrow{lx}, \omega_r\right) = \boldsymbol{m}_S \cdot clamp\left(\boldsymbol{n}_x \cdot \boldsymbol{h}_{l,x}\right)^{\gamma}$$

Blinn and Phong reflection model comparison

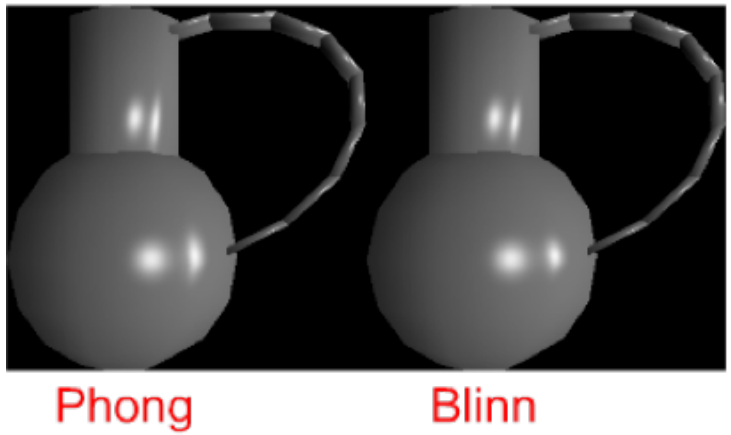
The Blinn specular model is usually slightly more expensive than the Phong one (since it requires normalization, which is more complex than the reflection), but still easily achievable in real-time by current hardware.

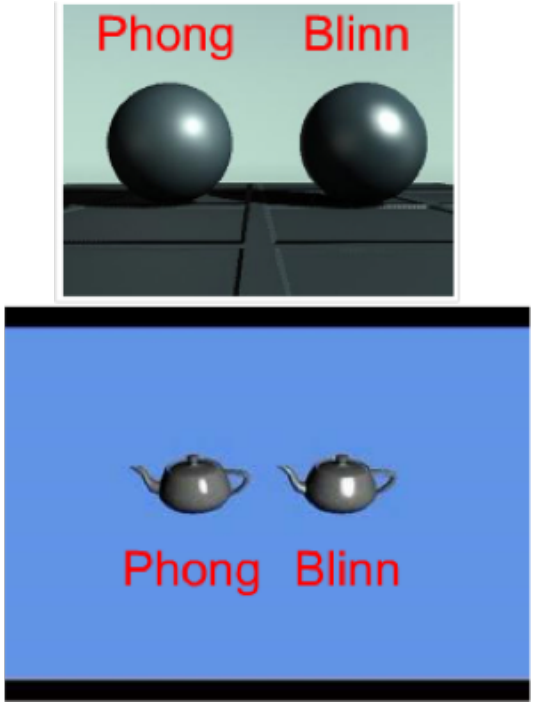
Since the two techniques provide slightly different results, they are usually both implemented and chosen by the artists to achieve different effects.

Blinn and Phong reflection model comparison

For example, these three pictures use on the left the Phong and on the right the Blinn specular computation.

As it can be seen, the Blinn has usually a larger decay area than the Phong with similar parameters.



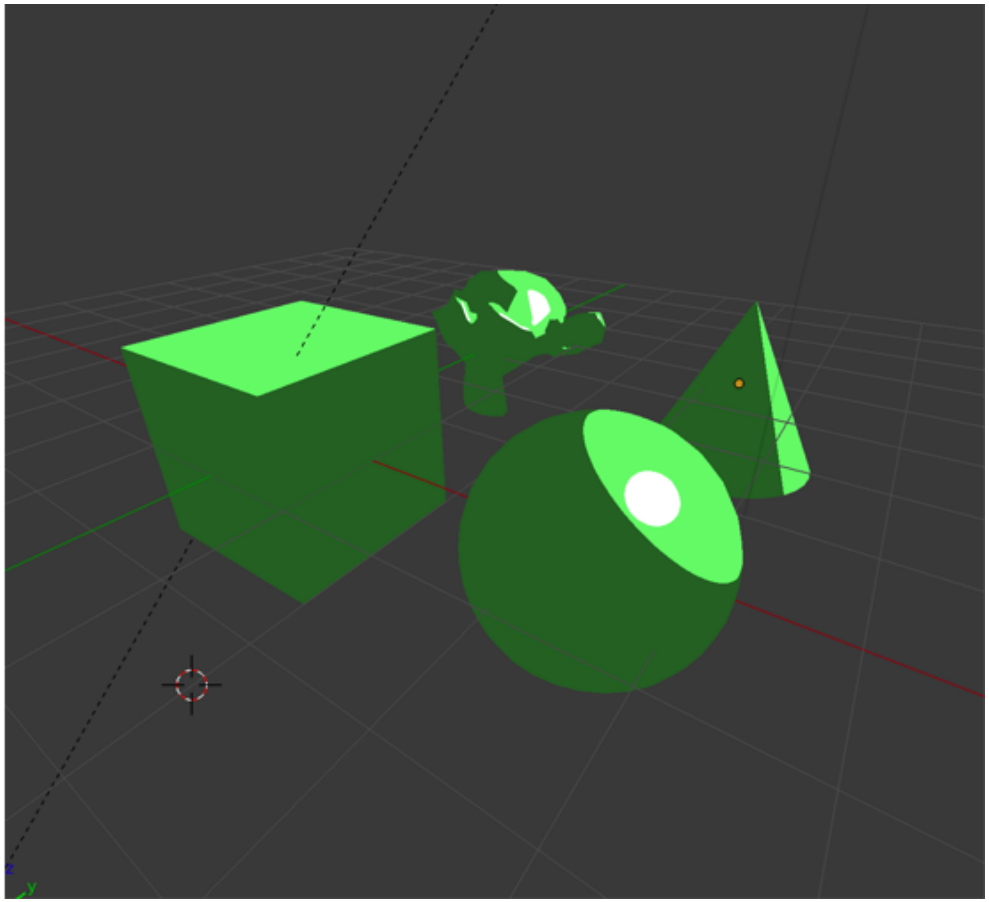


Toon shading

Toon shading simplifies the output color range, using only discrete values according to a set of thresholds.

In this way it achieves a cartoon-like rendering style.

It can be used both for the diffuse and specular components of the BRDF.



Toon shading

The technique starts from a standard Lambert BRDF for the diffuse, and from a Phong or Blinn BRDF with $\gamma=1$ for the specular components. Then it uses two colors ($\boldsymbol{m}_{D1}$, $\boldsymbol{m}_{D2}$) or ($\boldsymbol{m}_{S1}$, $\boldsymbol{m}_{S2}$) and a threshold (t_D or t_S) for determining which one to choose.

$$f_{diffuse}(x, \overrightarrow{lx}) = \begin{cases} \boldsymbol{m}_{D1} & \overrightarrow{lx} \cdot \boldsymbol{n}_x \geq t_D \\ \boldsymbol{m}_{D0} & \overrightarrow{lx} \cdot \boldsymbol{n}_x < t_D \end{cases}$$

$$\boldsymbol{r}_{l,x} = 2\boldsymbol{n}_x \cdot (\overrightarrow{lx} \cdot \boldsymbol{n}_x) - \overrightarrow{lx}$$

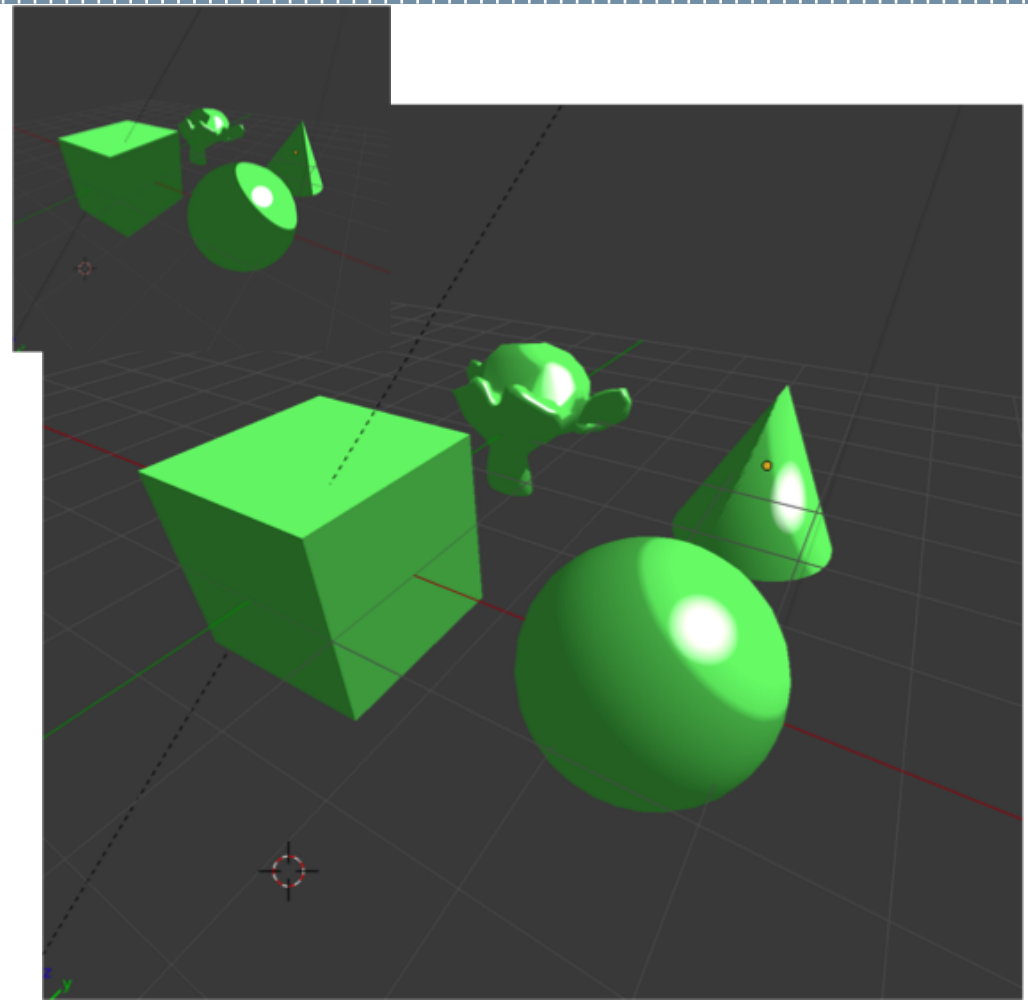
$$f_{specular}(x, \overrightarrow{lx}, \omega_r) = \begin{cases} \boldsymbol{m}_{S1} & \omega_r \cdot \boldsymbol{r}_{l,x} \geq t_S \\ \boldsymbol{m}_{S0} & \omega_r \cdot \boldsymbol{r}_{l.x} < t_S \end{cases}$$

(Phong version)

Toon shading

In general, to achieve better visual results, more than two colors are used for both the specular and diffuse parts.

Moreover, small gradients are added to smooth the transitions between different colors.



Toon shading

This is usually implemented by using a color that is function of the cosine of the angles between the considered rays.

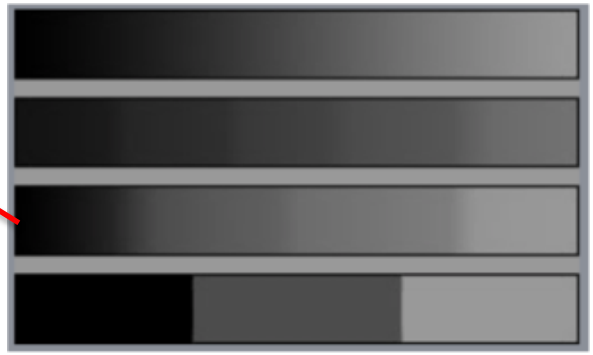
Functions are implemented as 1D textures (we will return on this later): in modern GPU hardware this adds an extra performance benefit since in general texture look-up is more efficient than program branching.

$$f_{diffuse}\left(x, \overrightarrow{lx}\right) = \boldsymbol{m}_D\left(\overrightarrow{lx} \cdot \boldsymbol{n}_x\right)$$

$$\boldsymbol{r}_{l,x} = 2\boldsymbol{n}_x \cdot \left(\overrightarrow{lx} \cdot \boldsymbol{n}_x\right) - \overrightarrow{lx}$$

$$f_{specular}\left(x, \overrightarrow{lx}, \omega_r\right) = \boldsymbol{m}_S\left(\omega_r \cdot \boldsymbol{r}_{l,x}\right)$$

(Phong version)



The Oren-Nayar diffuse model

Some materials are characterized by a phenomenon called *retroreflection*: they tend to reflect back in the direction of the light source.

They are characterized by very rough surfaces, and they cannot be accurately simulated with the Lambert diffuse reflection model.

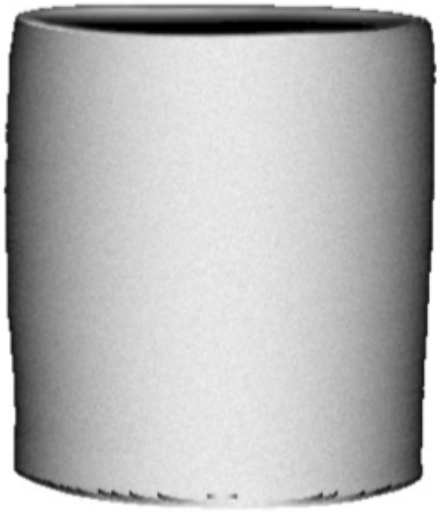
The Oren-Nayar diffuse reflection model has been devised to more appropriately model such materials.

In most of the cases, these materials do not show specular reflections.

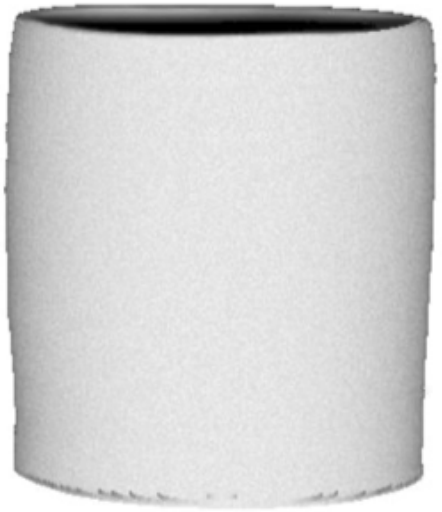
The Oren-Nayar diffuse model

Typical real life materials that require this special technique are clay, dirt and some types of cloths.

Real image
(clay vase)



Lambert



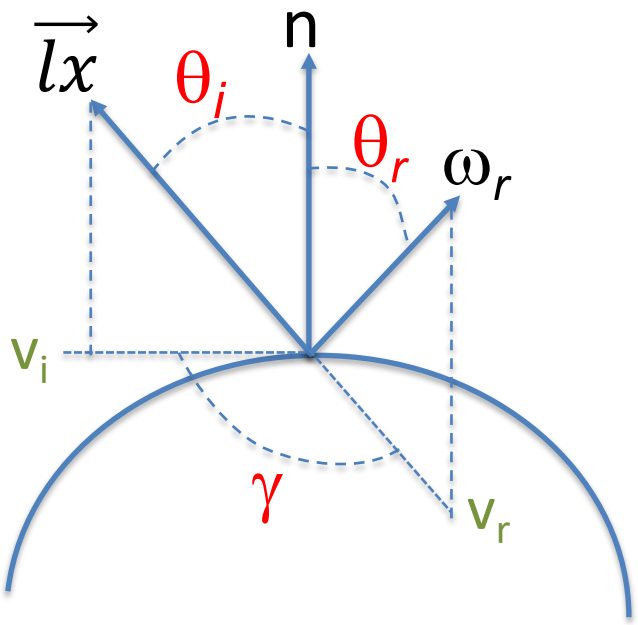
Oren-Nayar

The Oren-Nayar diffuse model

It requires three vectors: the direction of the light d, the normal vector n, and the direction of the viewer ω_r .

These vectors identify three angles:

- θ_i between d and n
- θ_r between ω_r and n
- γ between the projections of ω_r and d on the plane perpendicular to n. We call such projections respectively v_r and v_i.



The Oren-Nayar diffuse model

The model is characterized by two parameters:

- $m_D = (m_R, m_G, m_B)$ that is the main color of the considered material
- $\sigma \epsilon \left[0, \frac{\pi}{2}\right]$ which represents the roughness of the material.

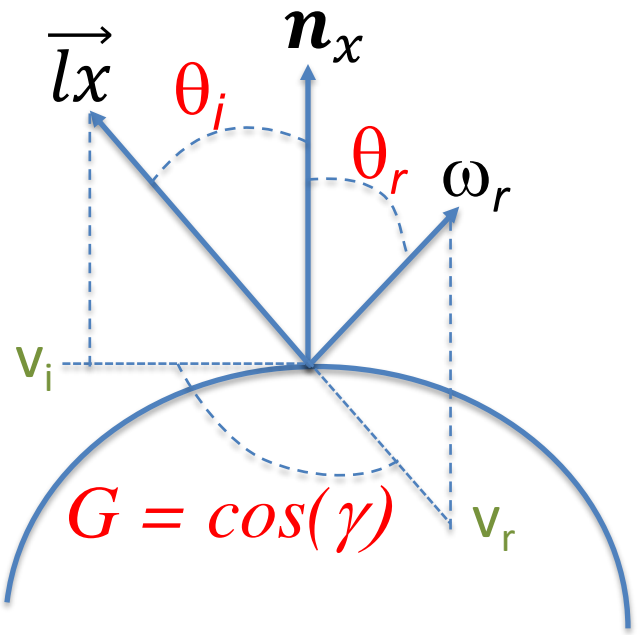
Higher values of σ produces rougher surfaces

The model converges to the Lambert diffusion for $\sigma = 0$.

The Oren-Nayar diffuse model

The model, can be computed with the following formulas:

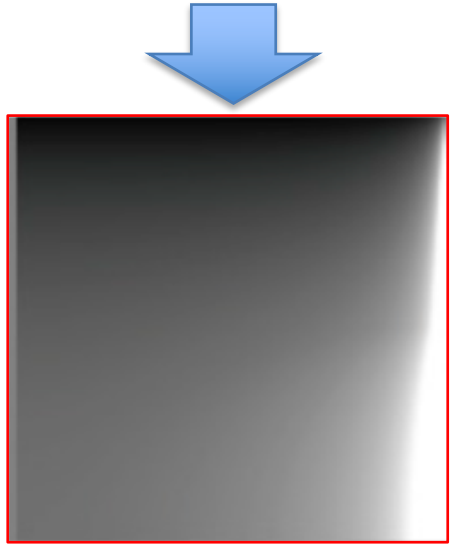
$$\theta_i = \cos^{-1}(\overrightarrow{lx} \cdot \boldsymbol{n}_x)$$
$$\theta_r = \cos^{-1}(\omega_r \cdot \boldsymbol{n}_x)$$
$$\alpha = \max(\theta_i, \theta_r)$$
$$\beta = min(\theta_i, \theta_r)$$
$$A = 1 - 0.5\frac{\sigma^2}{\sigma^2 + 0.33}$$
$$B = 0.45\frac{\sigma^2}{\sigma^2 + 0.09}$$
$$\boldsymbol{v_i} = normalize(\overrightarrow{lx} - (\overrightarrow{lx} \cdot \boldsymbol{n}_x)\boldsymbol{n}_x)$$
$$\boldsymbol{v_r} = normalize(\omega_r - (\omega_r \cdot \boldsymbol{n}_x)\boldsymbol{n}_x)$$
$$G = \max(0, \boldsymbol{v_i} \cdot \boldsymbol{v_r})$$
$$L = \boldsymbol{m}_D \cdot clamp(\overrightarrow{lx} \cdot \boldsymbol{n}_x)$$
$$f_{diffuse}(x, \overrightarrow{lx}, \omega_r) = L\,(A + B\,G \sin\alpha \tan\beta)$$



The Oren-Nayar diffuse model

The formula is quite complex: in most of the cases, the material specify directly parameter A instead of σ, and $B \sin\alpha \tan\beta$ is pre-computed and interpolated from a texture (a table), addressed below as f(), function of $\overrightarrow{lx} \cdot \boldsymbol{n}_x$ and $\omega_r \cdot \boldsymbol{n}_x$.

$$\theta_{\boldsymbol{i}} = \cos^{-1}(p)$$
$$\theta_{\boldsymbol{r}} = \cos^{-1}(q)$$
$$\alpha = \max(\theta_{\boldsymbol{i}}, \theta_{\boldsymbol{r}})$$
$$\beta = \min(\theta_{\boldsymbol{i}}, \theta_{\boldsymbol{r}})$$
$$B = 0.45\frac{\sigma^2}{\sigma^2 + 0.09}$$
$$f(p,q) = B \sin\alpha \tan\beta$$



$$\boldsymbol{v_i} = normalize\left(\overrightarrow{lx} - (\overrightarrow{lx} \cdot \boldsymbol{n}_x)\boldsymbol{n}_x\right)$$
$$\boldsymbol{v_r} = normalize(\omega_r - (\omega_r \cdot \boldsymbol{n}_x)\boldsymbol{n}_x)$$
$$G = \max(0, \boldsymbol{v_i} \cdot \boldsymbol{v_r})$$
$$L = \boldsymbol{m}_D \cdot clamp\left(\overrightarrow{lx} \cdot \boldsymbol{n}_x\right)$$
$$f_{diffuse}\left(x, \overrightarrow{lx}, \omega_r\right) = L\left(A + Gf\left(\overrightarrow{lx} \cdot \boldsymbol{n}_x, \omega_r \cdot \boldsymbol{n}_x\right)\right)$$

The Ward anisotropic specular model

Some objects are characterized by grooves on their surface: hairs, discs, brushed metals.

In this case, specular highlights are oriented along the grooves.

The Ward anisotropic specular model

This type of surfaces are called *anisotropic* materials.

The Ward specular model is important for two reasons:

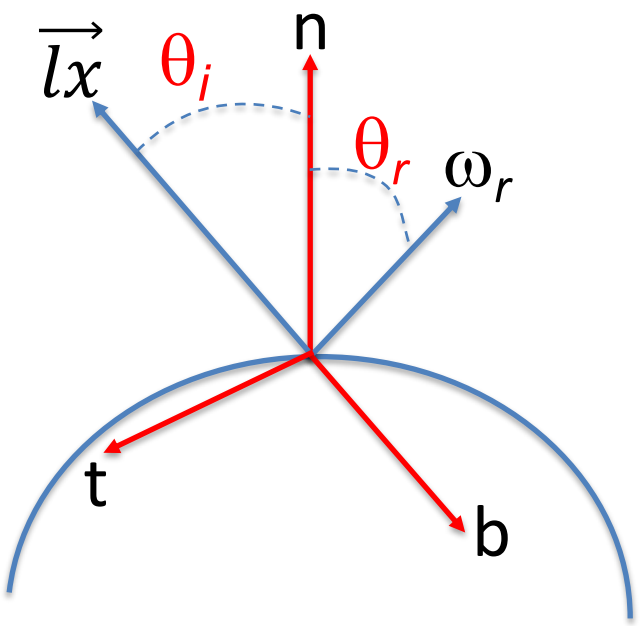
- It is derived from physically inspired principles
- It supports anisotropic reflections

The Ward anisotropic specular model

To support anisotropy, an orientation of the grooves on the surface must be specified: this is done assigning two extra vectors, beside the normal.

Such vectors are called tangent and bi-tangent, and are addressed with letters **b** and **t**.

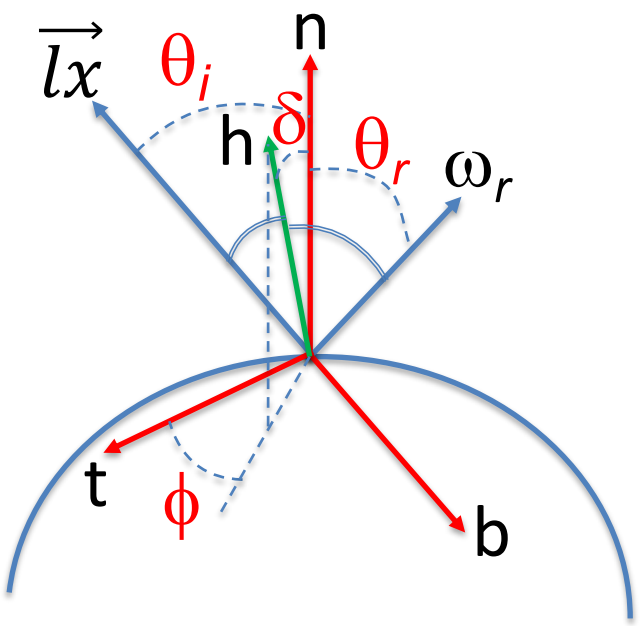
We will return on how to encode or derive such vectors in a future lesson.



The Ward anisotropic specular model

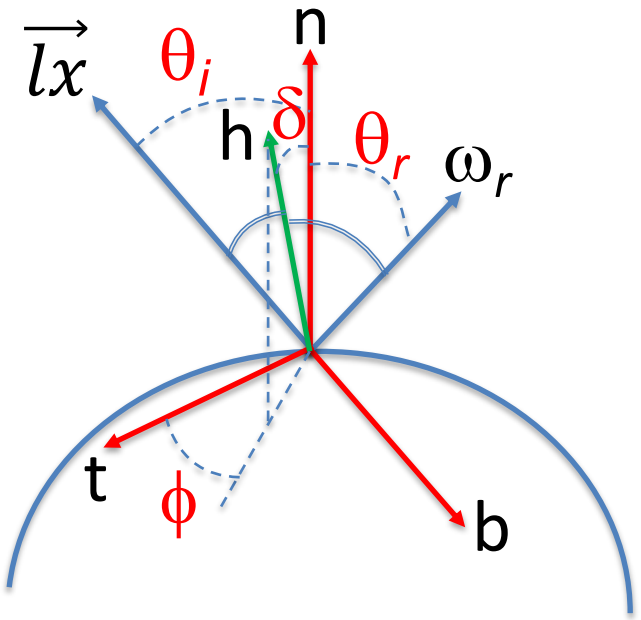
Similar to the Blinn specular model, the Ward technique depends on the *half-vector* **h** between the light and the viewer directions.

In particular, it depends on the angle of such vector with the normal (here denoted with δ), and on the on the angle between its projection on the **bt**-plane, with the groove direction (here denoted with ϕ).



The Ward anisotropic specular model

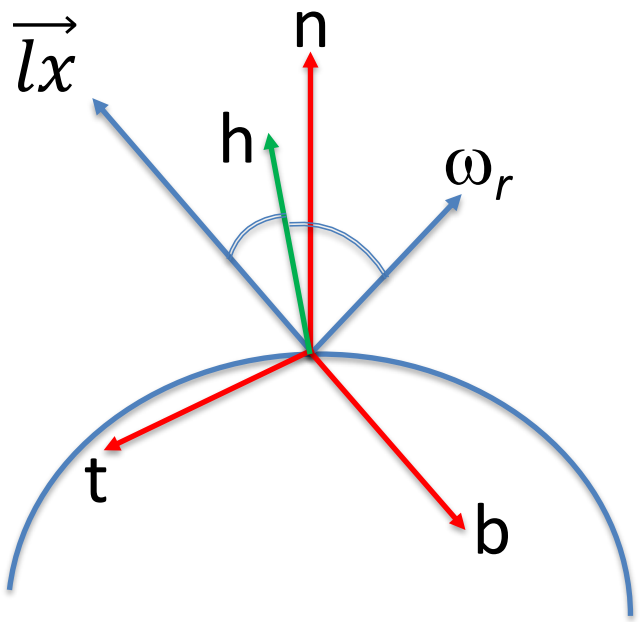
The formula for the specular component of the Ward model depends on two roughness parameters for the **t** and **b** directions, here denoted with α_t and α_b.



$$f_{specular}\left(x, \overrightarrow{lx}, \omega_r\right) = \boldsymbol{m}_S \frac{e^{-\tan^2 \delta \cdot \left(\frac{\cos^2 \phi}{{\alpha_t}^2} + \frac{\sin^2 \phi}{{\alpha_b}^2}\right)}}{4\pi \alpha_t \alpha_b \cdot \sqrt{\cos \theta_i \cos \theta_i}} \cos \theta_i$$

The Ward anisotropic specular model

Using the trigonometric definitions and the properties of the dot product, the formula can be expressed in a more computational efficient way.



$$f_{specular}\left(x, \overrightarrow{lx}, \omega_r\right) = \boldsymbol{m}_S \frac{e^{-\frac{\left(\frac{\boldsymbol{h}\cdot\boldsymbol{t}_x}{\alpha_t}\right)^2+\left(\frac{\boldsymbol{h}\cdot\boldsymbol{b}_x}{\alpha_b}\right)^2}{(\boldsymbol{h}\cdot\boldsymbol{n}_x)^2}}}{4\pi\alpha_t\alpha_b \cdot \sqrt{\frac{\omega_r \cdot \boldsymbol{n}_x}{\overrightarrow{lx} \cdot \boldsymbol{n}_x}}}$$

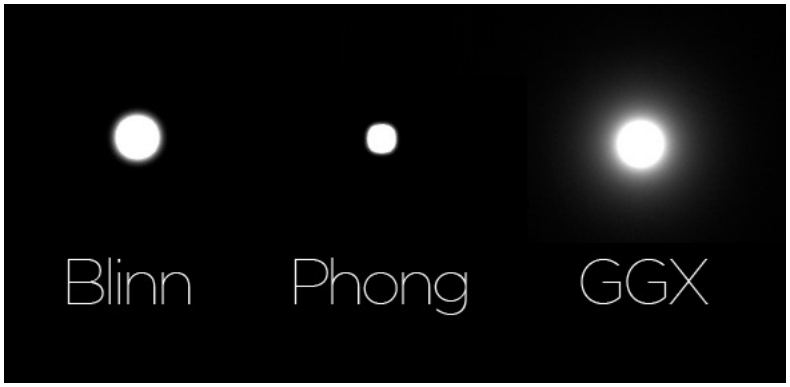
To simplify the presentation, clamping of angular values and checks to avoid denominator going to zero have been omitted. Also, many books uses the convention shown below (same equation, easier to insert in an engine, but less performant).

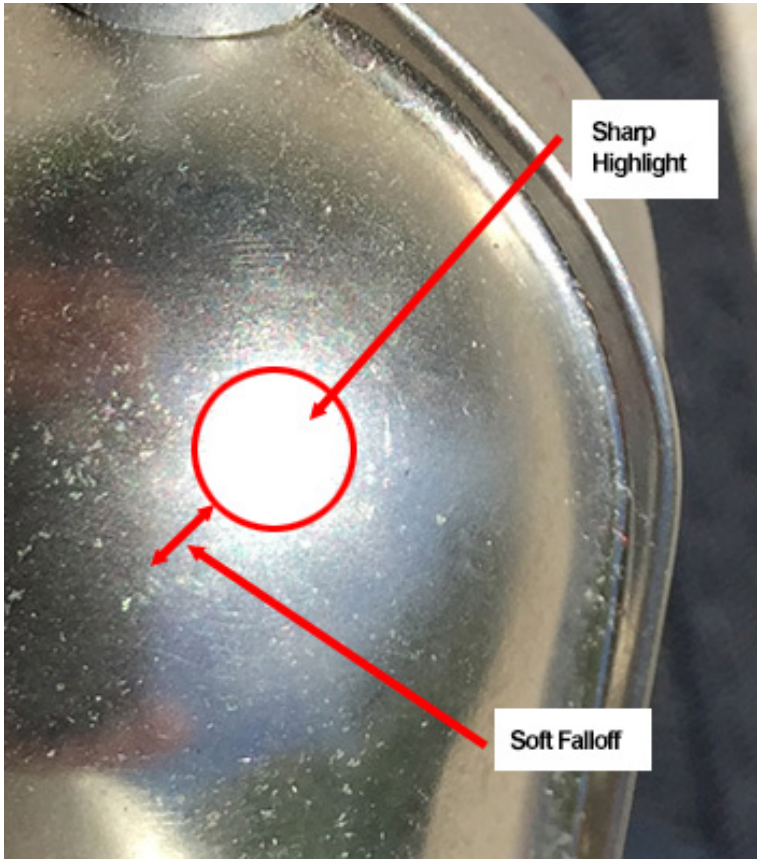
$$\boldsymbol{m}_S \frac{e^{-\frac{\left(\frac{\boldsymbol{h}\cdot\boldsymbol{t}_x}{\alpha_t}\right)^2+\left(\frac{\boldsymbol{h}\cdot\boldsymbol{b}_x}{\alpha_b}\right)^2}{(\boldsymbol{h}\cdot\boldsymbol{n}_x)^2}}}{4\pi\alpha_t\alpha_b \cdot \sqrt{(\omega_r\cdot\boldsymbol{n}_x)(\overrightarrow{lx}\cdot\boldsymbol{n}_x)}} (\overrightarrow{lx}\cdot\boldsymbol{n}_x)$$

The Cook-Torrance reflection model

Specular highlights on real objects, tend to exhibit a soft falloff area.

The Blinn and Phong specular models produce instead very sharp falloff area.





The Cook-Torrance reflection model

Moreover, due to the *Fresnel* principle, objects tend to have a larger specular reflection when the light is almost parallel to the surface.

These and other effects motivated the introduction of more complex specular illumination models that can better capture these features.

The Cook-Torrance reflection model

The *Cook-Torrance* reflection model, aims at computing both the specular and diffuse components in a physical accurate way.

The diffuse component follows the Lambert diffusion model. However, to achieve a physically accurate behavior, it is balanced with the specular part via linear interpolation, according to a coefficient *k*:

$$f_r(x, \overrightarrow{lx}, \omega_r) = k\ f_{diffuse}(x, \overrightarrow{lx}, \omega_r) + (1 - k) f_{specular}(x, \overrightarrow{lx}, \omega_r)$$

$$f_{diffuse}(x, \overrightarrow{lx}, \omega_r) = \boldsymbol{m}_D \cdot clamp(\overrightarrow{lx} \cdot \boldsymbol{n}_x)$$

The Cook-Torrance reflection model

The specular term is computed as the product of three terms:

- *F* – the Fresnel term
- *G* – the Geometric term
- *D* – the Distribution term

$$f_{specular}(x, \overrightarrow{lx}, \omega_r) = \boldsymbol{m}_S \frac{D \cdot F \cdot G}{4 \cdot clamp(\omega_r \cdot \boldsymbol{n}_x)}$$

$f_{specular}(x, \overrightarrow{lx}, \omega_r) = \boldsymbol{m}_S \frac{D \cdot F \cdot G}{4 \cdot clamp(\overrightarrow{lx} \cdot \boldsymbol{n}_x) \cdot clamp(\omega_r \cdot \boldsymbol{n}_x)}$

Note: many books and web-sites use the formula on the left. In those cases, however, the specular contribution is later multiplied by $clamp(\overrightarrow{lx} \cdot \boldsymbol{n}_x)$ since it is added to the diffuse contribution which in such context is assumed to be constant. The formulation here is taken from the article: "Model of light reflection for computer synthesized pictures" from J.F. Blinn, 1977.

The Cook-Torrance reflection model

Similarly to the other specular models, it is characterized by a specular color $\boldsymbol{m_S}$.

The $clamp(\omega_r \cdot \boldsymbol{n}_x)$ is a geometric term that normalizes the values computed by components *D*, *F* and *G*.

$$f_{specular}(x, \overrightarrow{lx}, \omega_r) = \boldsymbol{m_S} \frac{D \cdot F \cdot G}{4 \cdot clamp(\omega_r \cdot \boldsymbol{n}_x)}$$

The Cook-Torrance reflection model

For each of the three terms *D*, *F* and *G*, several definitions exists, each one with its features and complexities.

In this course we will present only the main ones.

The Cook-Torrance reflection model

The *microfacet* version of the geometric term *G*, is not characterized by any parameters, and depends only on the angles.

In particular, it depends on the half-vector *h*, defined in the same way as for the Blinn specular model.

$$\boldsymbol{h}_{l,x} = \frac{\overrightarrow{lx} + \omega_r}{|\overrightarrow{lx} + \omega_r|} = normalize(\overrightarrow{lx} + \omega_r)$$

$$G = \min\left(1, \frac{2\left(\boldsymbol{h}_{l,x} \cdot \boldsymbol{n}_x\right)\left(\omega_r \cdot \boldsymbol{n}_x\right)}{\left(\omega_r \cdot \boldsymbol{h}_{l,x}\right)}, \frac{2\left(\boldsymbol{h}_{l,x} \cdot \boldsymbol{n}_x\right)\left(\overrightarrow{lx} \cdot \boldsymbol{n}_x\right)}{\left(\omega_r \cdot \boldsymbol{h}_{l,x}\right)}\right)$$

The Cook-Torrance reflection model

The *GGX* version for the geometric term G, uses an helper function that is called first with the light, and then with the view direction. It depends on a constant k that defines the roughness of the surface.

$$g_{\text{GGX}}(\boldsymbol{n}, \boldsymbol{a}) = \frac{clamp(\boldsymbol{n}\cdot\boldsymbol{a})}{(1-k)\cdot clamp(\boldsymbol{n}\cdot\boldsymbol{a})+k}$$

$$G = g_{\text{GGX}}(\boldsymbol{n}_x, \omega_r) \cdot g_{\text{GGX}}(\boldsymbol{n}_x, \overrightarrow{lx})$$

The Cook-Torrance reflection model

The Fresnel term *F*, depends on a parameter $F_0 \in [0,1]$. It defines how the light response changes with the angle of incidence with respect to the viewer, and it can be approximated with the following definition:

$$F = F_0 + (1 - F_0)\left(1 - clamp\left(\omega_r \cdot \boldsymbol{h}_{l,x}\right)\right)^5$$

The Cook-Torrance reflection model

The distribution term D accounts for the roughness of the surface. The *Beckmann* version depends from one parameter $m > 0$ that defines the average slope of the surface at a microscopic level.

$$\alpha = \cos^{-1}\left(\boldsymbol{h}_{l,x} \cdot \boldsymbol{n}_x\right)$$

$$D = \frac{e^{-\left(\frac{\tan \alpha}{m}\right)^2}}{\pi \cdot m^2 \cos^4 \alpha}$$

Using trigonometric relations and the properties of the dot product, the formula can be simplified as shown below.

$$D = \frac{\frac{\left(\boldsymbol{h}_{l,x} \cdot \boldsymbol{n}_x\right)^2 - 1}{e^{\left(\boldsymbol{h}_{l,x} \cdot \boldsymbol{n}_x\right)^2 m^2}}}{\pi \cdot m^2 \left(\boldsymbol{h}_{l,x} \cdot \boldsymbol{n}_x\right)^4}$$

The Cook-Torrance reflection model

The *GGX* version for the distribution D, depends on a constant α that defines the roughness of the surface. This constant is related to parameter *k* used in the *GGX* version for *G*.

$$D = \frac{\alpha^2}{\pi(clamp(\boldsymbol{n}_x \cdot \boldsymbol{h}_{l,x})^2(\alpha^2 - 1) + 1)^2}$$

With *k* used in the GGX version of G equal to: $k = \frac{(\alpha+1)^2}{8}$

The Cook-Torrance reflection model

To summarize, the model depends on two colors $\boldsymbol{m}_D$ and $\boldsymbol{m}_S$ and three parameters, F_0, m and k. Using the *Beckmann* and *microfacet* models, we have:

$$\boldsymbol{h}_{l,x} = normalize(\overrightarrow{lx} + \omega_r)$$

$$G = \min\left(1, \frac{2(\boldsymbol{h}_{l,x} \cdot \boldsymbol{n}_x)(\omega_r \cdot \boldsymbol{n}_x)}{(\omega_r \cdot \boldsymbol{h}_{l,x})}, \frac{2(\boldsymbol{h}_{l,x} \cdot \boldsymbol{n}_x)(\overrightarrow{lx} \cdot \boldsymbol{n}_x)}{(\omega_r \cdot \boldsymbol{h}_{l,x})}\right)$$

$$D = \frac{e^{-\frac{1-(\boldsymbol{h}_{l,x} \cdot \boldsymbol{n}_x)^2}{(\boldsymbol{h}_{l,x} \cdot \boldsymbol{n}_x)^2 m^2}}}{\pi \cdot m^2 (\boldsymbol{h}_{l,x} \cdot \boldsymbol{n}_x)^4}$$

$$F = F_0 + (1 - F_0)\left(1 - clamp(\omega_r \cdot \boldsymbol{h}_{l,x})\right)^5$$

$$f_{specular}(x, \overrightarrow{lx}, \omega_r) = \boldsymbol{m}_S \frac{D \cdot F \cdot G}{4 \cdot clamp(\omega_r \cdot \boldsymbol{n}_x)}$$

The expression of the GGX version is left for exercise.

$$f_{diffuse}(x, \overrightarrow{lx}, \omega_r) = \boldsymbol{m}_D \cdot clamp(\overrightarrow{lx} \cdot \boldsymbol{n}_x)$$

$$f_r(x, \overrightarrow{lx}, \omega_r) = k\, f_{diffuse}(x, \overrightarrow{lx}, \omega_r) + (1 - k) f_{specular}(x, \overrightarrow{lx}, \omega_r)$$

The Cook-Torrance reflection model

Although its complexity, the model can provide realistic reflections that take into account many physical behaviors.

In general, the Cook-Torrance reflection model is used as a basic building block for many advanced rendering techniques.

Such techniques produce special textures that are used as look-up tables to be able to render this reflection model in real-time.