

Illumination models: Phong analysis

Emilio González Montaña (egoeht69@hotmail.com)
at <http://emiliogonzalez.sytes.net>

2007/06/26

Abstract

The aim of this document is to study the Phong method as an illumination model. We will make a brief description of the method and we will also analyze the different parameters used by this method, and how they affect to the final result, illustrating these parameters with 3DS scenes.

1 Introduction

The Phong model combines two techniques: the perfect refraction and the empirically spread specular reflection [1].

The diffuse reflection is calculated by the cosine of the angle formed between the surface normal and the incoming light direction. This is called the Lambert's Cosine Law [2].

The other technique is the calculation of the specular reflection, this is done via an empirically spread specular term. The idea is don't reflect the light in surfaces as flat mirrors, and consider the surface as a lot of tiny mirrors oriented in different directions, this make the surface looks less perfect, so it will be look more realistic. The final appearance will be similar to plastic.

2 Phong components

These are the components that the Phong model takes into consideration in the illumination calculations [3]:

- Ambient light is a constant amount of light that gets added to the scene no matter what. It's goal is to fake the contribution of indirect reflections, which can normally only be accounted for using global illumination solutions. The ambient term is used simply to keep shadows from turning pitch black, which would be very unrealistic.
- Diffuse light is light that hits a surface and gets scattered equally into all directions. Its intensity depends on the angle at which the light strikes the surface (the angle between the light vector and the surface normal). The bigger the angle, the less bright the surface will appear.
- Specular light is light that gets reflected in a particular direction. Because of this, specular light is view-dependent, its intensity depends on

the angle between the light vector and the view vector. Specular lighting is responsible for creating the highlights that make an object look shiny.

The light is additive, so the final intensity (I) of the light reflected by a given surface will be the sum of the ambient (A), diffuse (D) and specular (S) components:

$$I = A + D + S$$

In this figure, we can see an example of how the mix of the three components influence into the final render (figure 1):

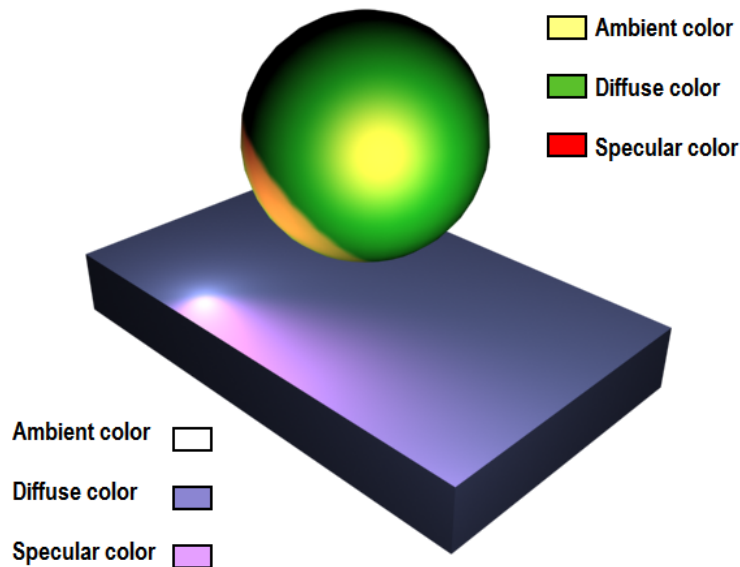


Figure 1: Sphere and box with Phong materials

2.1 Ambient

The ambient term (A), as mentioned above, is constant and this is the formula to represent it:

$$A = Al \times Am$$

The ambient term (A) is simply a combination of the ambient components of the light source (Al) and the surface material (Am). By multiplying the light color and the material color, we account for the fact that any material only reflects certain frequencies of light. A green surface (like the sphere into figure 1), for example, does not reflect red light. As we will see below, the diffuse and specular terms also combine the light and material colors in the same way.

2.2 Diffuse

Diffuse lighting is based on the angle between the light vector (L) and the surface normal (N). The surface normal is a given, and requires no further explanation.

The light vector L is the normalized vector from the point being shaded to the light source. Given these two vectors, the diffuse term is:

$$D = Dl \times Dm \times \max(L \cdot N, 0)$$

The dot product of L and N returns the cosine of the angle between those two vectors. When the two vectors are equal, the dot product is one. When they are perpendicular, the value is zero. If the two vectors point away from each other, the value becomes negative. When this happens, the value must be clamped to zero to prevent ending up with *negative light*.

The clamped dot product is also multiplied (or modulated) with the diffuse colors of the light (Dl) and the surface material (Dm).

2.3 Specular

The specular lighting term is the most difficult of the three terms. It is dependent on the angle between the reflected light vector (R) and the view vector (V). The view vector is the normalized vector from the point being shaded to the camera position. The reflection vector is calculated as follows:

$$R = 2(N \cdot L) \times N - L$$

The reflection vector represents the direction the incoming light would be reflected in if the surface were a perfect mirror. The angle between R and N is equal to that between L and N . Having calculated R and V , we can now compute the specular term:

$$S = Sl \times Sm \times \max(k \times (R \cdot V), 0)^n$$

The dot product is there because we want to favor reflections in the direction of the camera to those that point away from the camera. The larger the angle between R and V , the lower the specular term will be. Notice how we also raise the dot product to a power of n , where n is called the specular exponent of the surface and represents its shininess. Higher values of n lead to smaller, sharper highlights, whereas lower values result in large and soft highlights. The k value is a factor to influence the product.

To get a better understanding of what the specular exponent does, remember that $R \cdot V$ represents the cosine of the angle between R and V . The graph below (figure 2) plots a cosine function raised to a few different powers. As the graph shows, the higher the exponent, the faster the function falls off to zero.

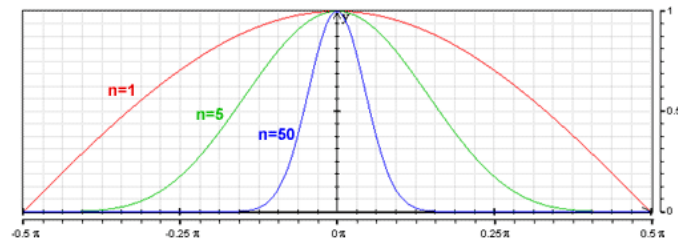


Figure 2: Several cosine functions raised to different powers

Regardless of the exponent used, the function is always zero when the angle between the two vectors is 90 degrees, and one when the angle is zero.

3 3DS parameters

In 3D Studio Max [4], we can define materials to use the Phong model, after this we can select every parameter (as we can see into figure 3).

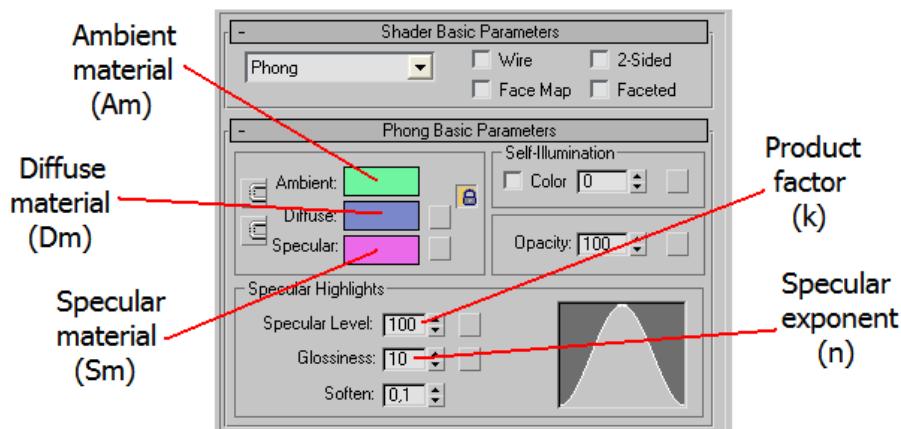


Figure 3: 3D Studio parameter for Phong model

For the three terms we can choose a color to be used as A_m , D_m or S_l (as we explained into section 2), in Diffuse (see figure 4) and Specular we also can choose a new material instead of a basic color (this will apply a different color to each pixel depending in which particular material we choose for each component).

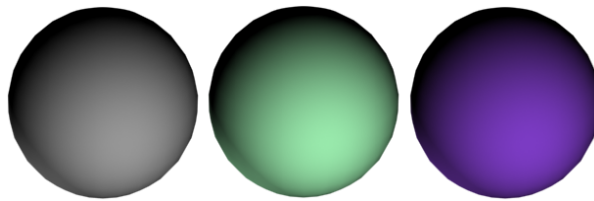


Figure 4: Several Diffuse components

With the specular term, we have all the needed parameters into the *Specular Highlights* group (we saw all this terms into section 2.3):

- Specular Level (figure 5): This is the first thing to do, because with value 0 we didn't have specular component (this will increase the cosine term).

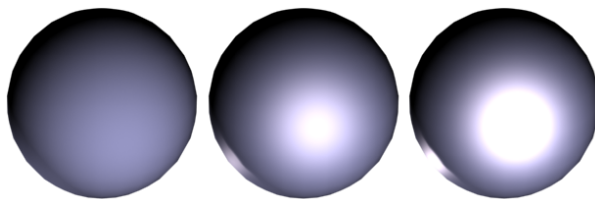


Figure 5: Several Specular Level components

- Glossiness (figure 6): This value makes the shine point smaller (with bigger values) or bigger.

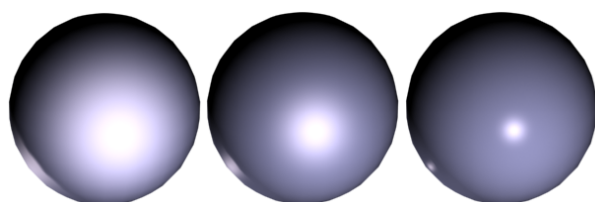


Figure 6: Several Specular Glossiness components

- Soften (figure 7): With this value we can make smoother gradients between the shiny points and the rest of the surface.

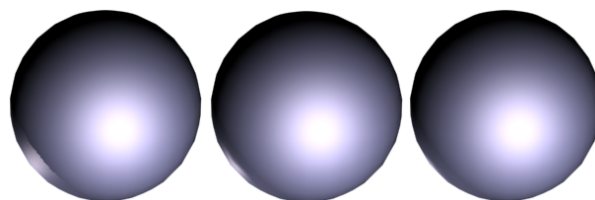


Figure 7: Several Specular Soften components

4 Conclusions

Phong's lighting model is ubiquitous in modern-day computer graphics, so it's important to understand it. Standard OpenGL uses Phong's equation to do vertex lighting, and modern hardware can also evaluate the equation per-pixel using programmable fragment processing.

It should be noted that Phong's lighting model does not have a very strong basis in physical reality. It looks convincing and is easy for 3D artists to control, though. If more physical realism is required, other lighting models exist which may do a better job, e.g. the Cook-Torrance model. This is still a local illumination model, however, which means that just like Phong's model, it does not capture indirect reflections. To account for the effects of indirect lighting and to obtain maximum realism, a global illumination solution such as radiosity is required.

References

- [1] Alan Watt: *3D Computer Graphics* (third edition), Addison-Wesley, 2000.
- [2] Johann Heinrich Lambert: *Photometria*, 1760.
- [3] David Escudero Mancebo: *Fundamentos de informática gráfica*, Ceysa, 2003.
- [4] 3D Studio Max Reference: *Autodesk*.