Introduction to Lighting

Background

Several terms for this topic:

- Lighting Model
- Illumination Model
- Shading Model

To achieve photo-realism requires multiple components. Some of these include:

- Accurate model (objects) descriptions; also known as materials.
  - Geometry, color, reflection, transparency, surface texture, emissive properties, etc.
- Accurate physical description of lighting
  - Point source, area light, color, position
- Accurate simulation of light transport, followed by rendering of the scene

For this class, we are going to focus on a lighting model that is loosely based in real-world physics. It isn't at the level needed for photo-realism, but it is pretty good.

Types of Light/Material in a Scene

For the lighting model in this class, we will consider three sources of light and materials in a scene.

1. **Ambient** : Non-directional, uniform distribution illumination throughout a scene.
2. **Diffuse** (from a point source) : Light that is scattered equally in all directions on the surface of an object.
3. **Specular** (from a point source) : Light that is scattered non-uniformly about the normal to the surface of a model.

We are using a three channel, Red Green Blue, model for illumination and materials (reflection).

Lighting Model

Light sources emit light intensity in each of the three channels.

Ambient $L_a = \begin{bmatrix} L_{ar} \\ L_{ag} \\ L_{ab} \end{bmatrix}$ Diffuse $L_d = \begin{bmatrix} L_{dr} \\ L_{dg} \\ L_{db} \end{bmatrix}$ Specular $L_s = \begin{bmatrix} L_{sr} \\ L_{sg} \\ L_{sb} \end{bmatrix}$

All values are in the range [0, 1]

Material Model

Ambient $k_a = \begin{bmatrix} k_{ar} \\ k_{ag} \\ k_{ab} \end{bmatrix}$ Diffuse $k_d = \begin{bmatrix} k_{dr} \\ k_{dg} \\ k_{db} \end{bmatrix}$ Specular $k_s = \begin{bmatrix} k_{sr} \\ k_{sg} \\ k_{sb} \end{bmatrix}$ (plus a shininess factor p)

All values are in the range [0, 1]

Reflection Model

Ambient

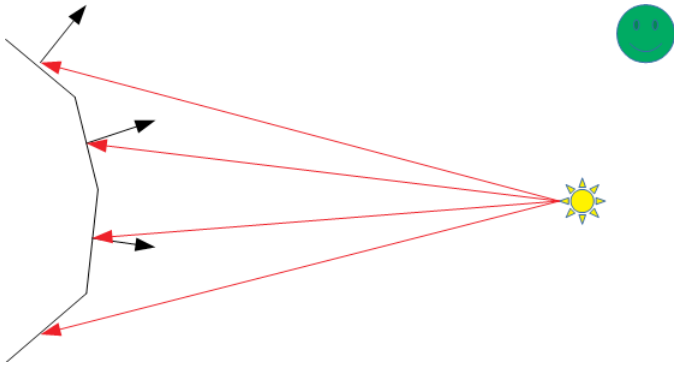
$$I_a = k_a L_a$$

Diffuse

A perfectly diffuse reflector scatters light uniformly in all directions. As a result, the reflection from the surface appears the same to all viewers; the reflection is view-independent. The amount of reflected light, however, is dependent on the material, some of the light is absorbed.

Lambert's Law of Consines

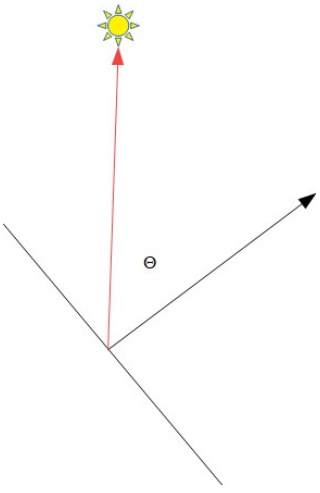
> Radiant energy from any small surface in any direction (Θ_N) relative to the surface normal is proportional to the $\cos(\Theta_N)$; where Θ_N is the angle between the normal of the surface and the direction of the light source. In other words, the light reflected from the surface is the same for any viewing direction.



We want the cos of the angle formed between the surface normal (N) and the vector from the point of interest on the surface to the light (L). We already have the surface normal provided by the geometry. The vector from the surface to the light is given by:

$L = P_L - P_g$ Where P_L is the position of the light and P_g is the surface point.

Both of these vectors need to be unit vectors.



The cos of the angle formed between these two vectors is the dot product:

$$\cos(\Theta) = N \cdot L$$

Therefore the reflected diffuse intensity is:

$$I_d = k_d L_d (N \cdot L)$$

Gouraud Shading – Per Vertex Lighting

In Gouraud shading the intensities are computed at the triangle vertices and then interpolated over the surface.

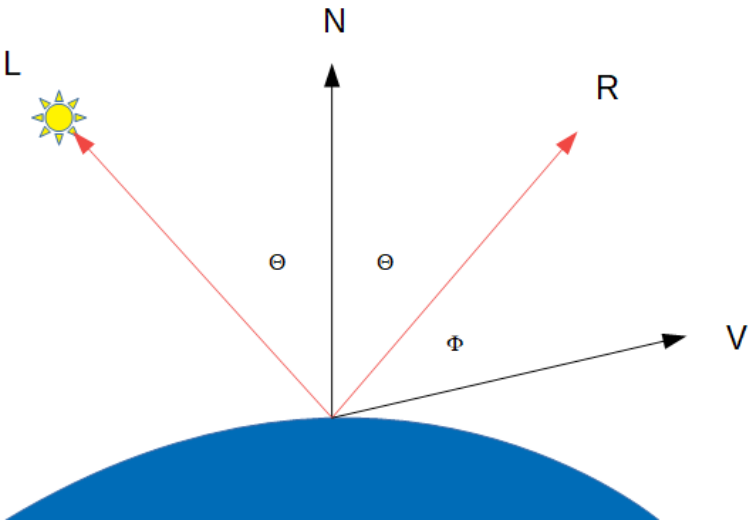
Phong Shading – Per Pixel Lighting

In Phong shading the vertex normals are interpolated over the surface and the lighting is computed at each pixel (fragment).

Specular

Diffuse shading results in surfaces that have a dull appearance to them, because the lighting is uniform across the surface. Shiny surfaces, on the other hand, such as an apple, metal, or my head, have a highlight (or bright spot) that is relative to the viewer. This is known as a ***specular*** reflection. This comes from the total or near-total reflection of the incident light around a particular region on the object's surface.

The specular reflection angle is the reflected angle of the light vector about the surface normal. Additionally, there is a shininess factor (n_s) that describes how shiny or dull the surface is. This factor is a power function of the cos of the angle formed between the reflection vector and the vector to the viewer (eye/camera).



The reflection vector is computed as:

$$R = 2(N \cdot L)N - L \quad \text{(reflection vector of the light)}$$

The viewing vector is computed as:

$$V = P_{eye} - P_v$$

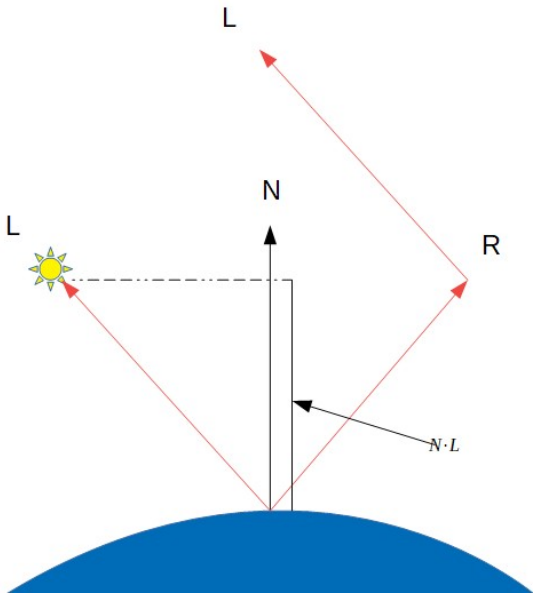
Combining everything we know, the specular reflection is:

$$I_s = k_s L_s \cos(\Phi)^{n_s}$$

$$I_s = k_s L_s (V \cdot R)^{n_s}$$

Reflection Vector

The reflection vector can be computed in terms of L and N. The projection of L onto N is obtained with $N \cdot L$. Using that and considering the diagram, we have:



$$R + L = (2N \cdot L)N$$

After rearranging we get:

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

Total Reflection

$$I = k_a L_a + k_d L_d (N \cdot L) + k_s L_s (V \cdot R)^{n_s}$$

- Computed for each color channel
- This is for a single light source. If more than one light source, accumulate the contribution from each light source.

What About Normals?

So, what about normals, what's the deal? Here is the deal…

When geometry is transformed by scaling or translation, normals aren't affected, they still point in the same direction as they did with the untransformed geometry. But when geometry is transformed by rotation, the normals aren't valid anymore and can't apply the rotation matrix to the normal vector.

Think of normals as the coefficients of plane equations. Specifically, have to do this using homogeneous coordinates for it to work correctly.

Remember that $\vec{n}\cdot\vec{v}$ (dot product of the normal with a vector in the plane is 0).

$$(n_x, n_y, n_z, n_w)\cdot(v_x, v_y, v_z, v_w)=0$$

$$n_x v_x + n_y v_y + n_z v_z + n_w v_w = 0$$

In matrix form

$$(n_x, n_y, n_z, n_w)\begin{bmatrix} v_x \\ v_y \\ v_z \\ v_w \end{bmatrix}=0$$

The plane equation is multiplying the transposed normal and the vertex together. Let's do a trick and insert $M^{-1}M$ in-between; where M is the Model-View matrix. Note that $M^{-1}M$ is the identity matrix.

$$(n_x, n_y, n_z, n_w)\, M^{-1} M \begin{bmatrix} v_x \\ v_y \\ v_z \\ v_w \end{bmatrix}=0$$

The right part $M\begin{bmatrix} v_x \\ v_y \\ v_z \\ v_w \end{bmatrix}$ is transforming the vertex to "eye space". The left part $(n_x, n_y, n_z, n_w)\, M^{-1}$ is the normal in "eye space" because the plane equation is also transformed. Given this, transforming the normal from "object space" to "eye space" given a Model-View matrix, is…

$$\begin{bmatrix} n_{x-eye} \\ n_{y-eye} \\ n_{z-eye} \\ n_{w-eye} \end{bmatrix}=(n_{x-obj}, n_{y-obj}, n_{z-obj}, n_{w-obj},)\, M^{-1}$$ Convert to pre-multiply form $$\begin{bmatrix} n_{x-eye} \\ n_{y-eye} \\ n_{z-eye} \\ n_{w-eye} \end{bmatrix}=(M^{-1})^T \begin{bmatrix} n_{x-obj} \\ n_{y-obj} \\ n_{z-obj} \\ n_{w-obj} \end{bmatrix}$$