Lighting

Physics vs. graphics

Computer graphics terms are somewhat confusing and disagree with physics:

Graphics: color of an object; physics: reflection spectrum (i.e. fraction of light of each frequency that gets reflected).

Graphics: intensity or color of a light ray; physics: radiance distribution (measured in watts/steradian/meter^2/meter)

Graphics: intensity or color of a point light source; physics: intensity spectrum of a point light source (almost in agreement!) measured in watts/steradian/meter.

Local lighting model

Describes interaction of the light with the surface.

Almost never truly based on physics: perception plays a greater role.

Visible light: electromagnetic waves, with wavelengths 400nm (violet) - 700nm (red); intensity can vary over many orders of magnitude.

Computer model: only three “frequencies”: RGB,

intensity varies over a small range, typically only 255 discrete values/ color.

Local illumination model

Describes interaction of the light and the surface

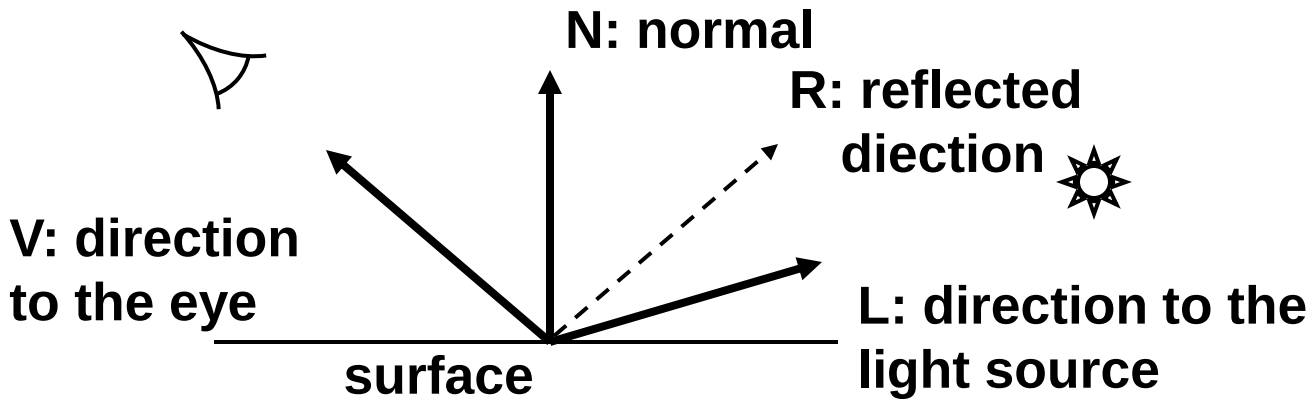
- **light can be reflected, absorbed and transmitted**
- **most important: reflection**
- **reflection can be ideal specular, diffuse and and anything between**
- **reflection equation relates the outgoing radiance in some direction to the incoming radiance**

Illumination model

Two main components:

- **light source characteristics**
  - **position**
  - **intensity for each freq. (color)**
    often, different intensity can be specified for different colors
  - **directional distribution**
- **surface properties**
  - **reflectance for each freq. (color)**
  - **different reflectance can be specified for diffuse and specular light**

Reflection geometry



$$R = 2(N,V)N - V$$

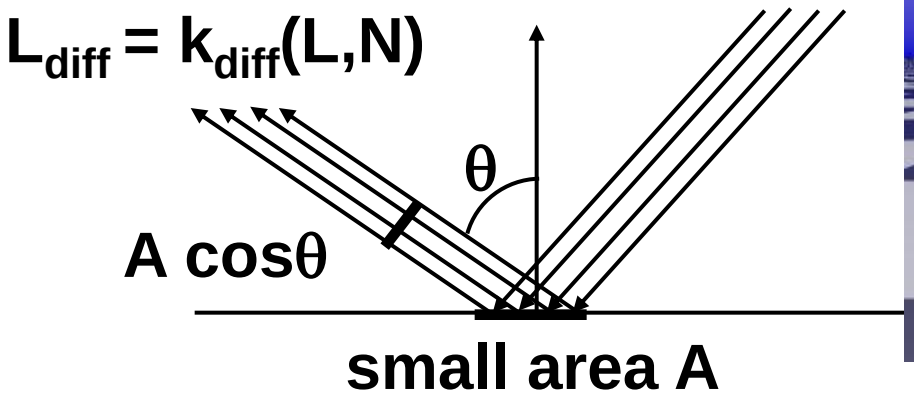
A simple model

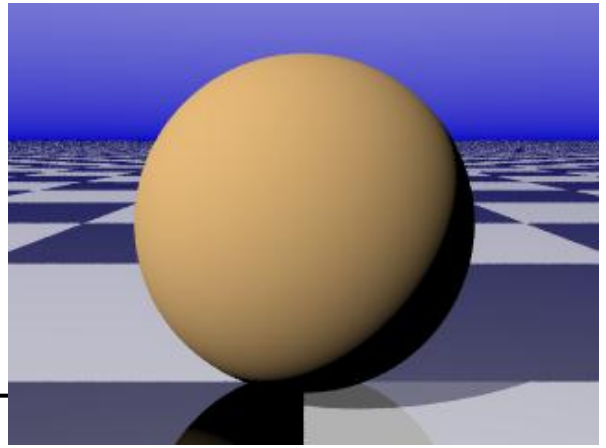
In the model commonly used in graphics applications, there are several components

- diffuse relection: intensity does not depend on the direction to the viewer
- specular: simulates relective surfaces and specular highlights depending on the direction to the viewer
- ambient: a crude approximation to the illumination created by the light diffusely reflected from surfaces

Diffuse component

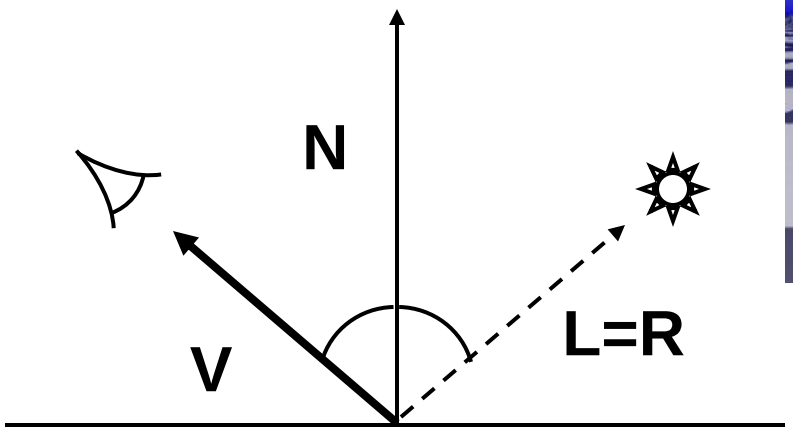
Diffuse surfaces are surfaces following the Lambert's law: the energy of the light reflected from a surface in a direction D is proportional to the cosine of the angle between the normal and D. As intensity (radiance) is proportional to the energy times cross section of the ray, it does not depend on the view direction, but is proportional to the cosine of the angle between the normal and the direction to the light.

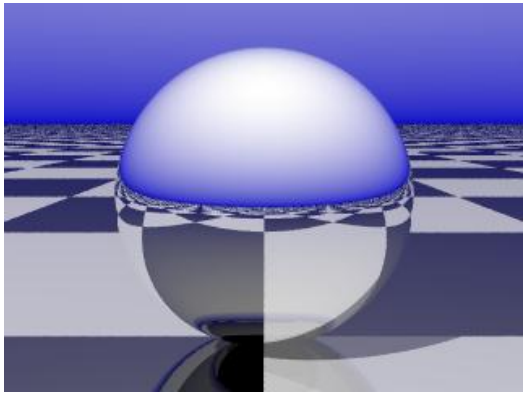




Specular component

Specular component approximates behavior of shiny surfaces. If a surface is an ideal mirror, the light from a source reaches the eye bouncing of a fixed point of the surface, only if the direction to the light coincides with the reflected direction to the eye:





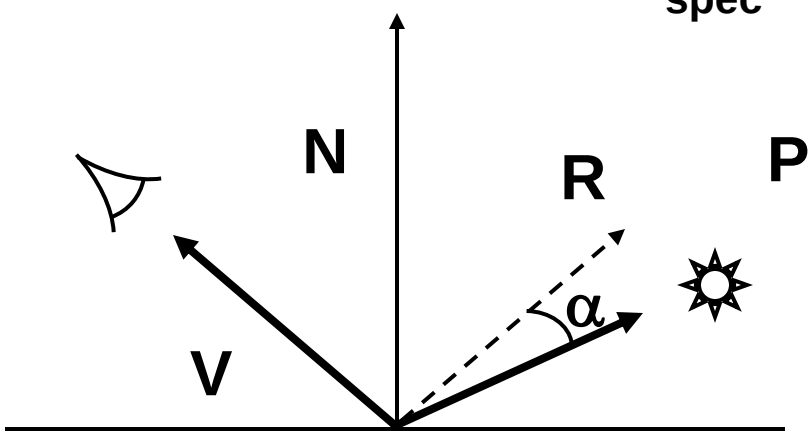
ideal mirror sphere

Specular reflection

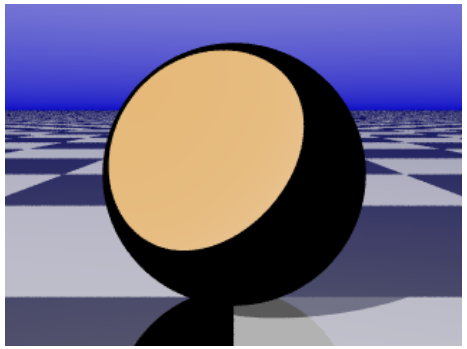
For non-ideal reflectors, the reflection of the light is still the brightest when L=R but decays, rather than disappears, as the angle between L and R increases. One way to achieve this effect is to use cosine of the angle to scale the reflected intensity:

$$L_{spec}=k_{spec}(R,L)^{p}$$

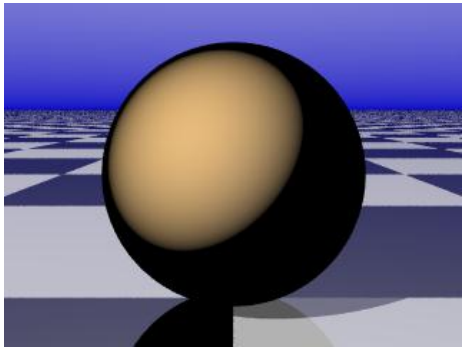
Phong exponent



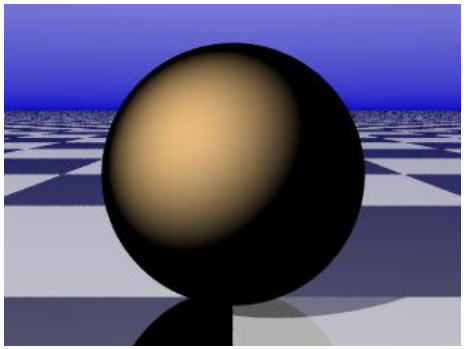
Specular reflection



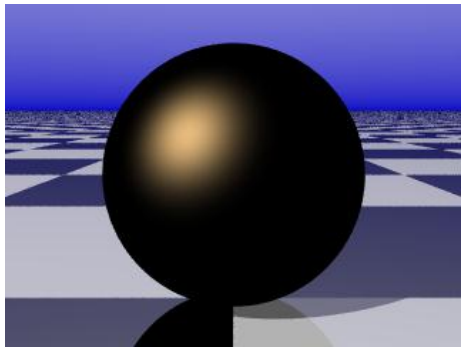
p = 0



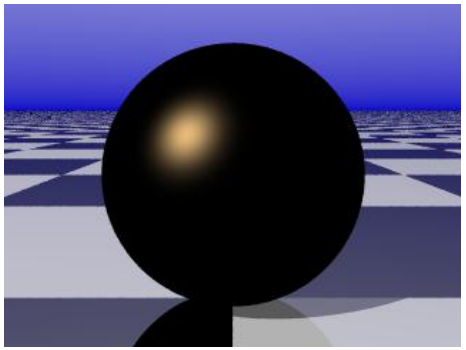
p = 1



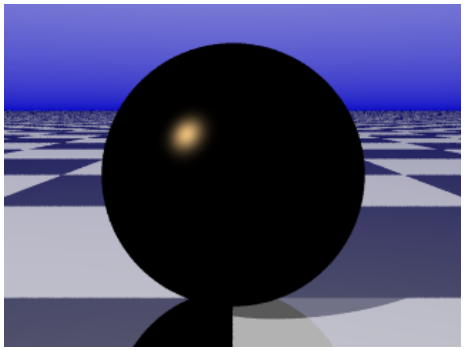
p = 2



p = 10



p = 25

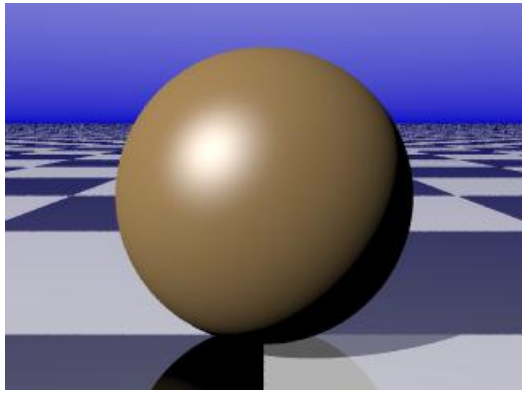


p = 100

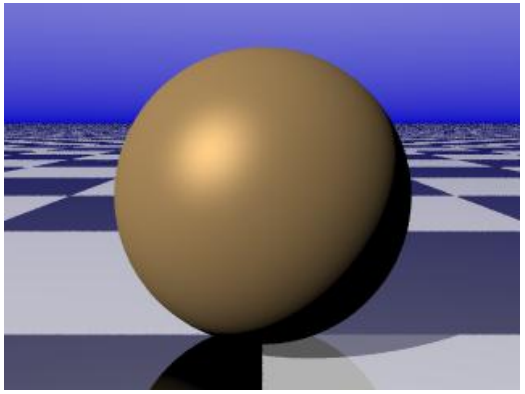
Metal vs. plasitic

Natural look of metallic surfaces is difficult to simulate, but the first approximation is obtained using proper highlight color.

For plastic objects, highlights are close in color to the color of the light. For metals, to the color of the surface. Assuming white lights, for plastic set k_{spec}=[c,c,c], where c is a constant, for more metallic look set k_{spec}= k_{diff}



plastic look

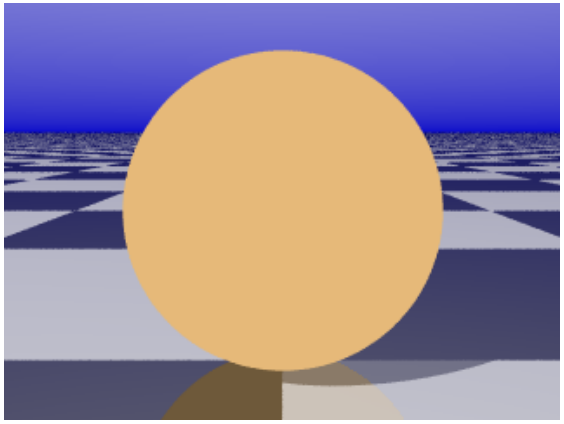


metallic look

Ambient component

Not all light illuminating a surface comes from light sources, or reflections of light sources in ideal mirrors; however, the light diffusely reflected from other surfaces is difficult to take into account, especially for real-time rendering. It is approximated by the ambient component: a constant is added to all objects. To have more control over ambient contribution, surfaces can be assigned ambient reflectivity.

$L_{amb}=k_{amb}I_{amb}$



Complete equation

$$I_{total} = k_{amb} I_{amb} + \sum_{\text{all lights}} I_i \left(k_{diff} (L_i, N) + k_{spec} (L_i, R)^p \right)$$

intensity of i-th light

direction to i-th light

If we are ray tracing for rendering, in summation only visible lights are present, and there are two additional terms: contribution from the reflected ray and transmitted ray. If we are using Z-buffering, then all active light sources are regarded as visible (OpenGL model)

Attenuation

In real life, radiance reaching us from a light source decreases with distance as $1/r^2$ (the stars are much less bright than the sun). However, due to the nature of approximations used in graphics, the inverse-square law typically results in pictures that are too dark; the fix is to allow the programmer to control how fast the decay is. I_i in the formulas is replaced not by I_i/r^2, as physics suggest, but by

$$\frac{I_i}{a + br + br^2}$$

and the most common choice of constants is

a = 1, b = c = 0, that is, no attenuation!

OpenGL model

Phong exponent called shininess. Several additions:

emmision;
ambient, diffuse, specular light "intensities"

Setting material parameters (K_{diff},K_{spec},K_{amb},p)

```
Vec3f  mat_diffuse, mat_spec, mat_amb;
   GLfloat shininess;

...
glMaterialfv(GL_FRONT,GL_DIFFUSE,mat_diffuse);
glMaterialfv(GL_FRONT,GL_SPECULAR,mat_spec);
glMaterialfv(GL_FRONT, GL_AMBIENT,mat_amb);
glMaterialf(GL_FRONT, GL_SHININESS,shininess);
```