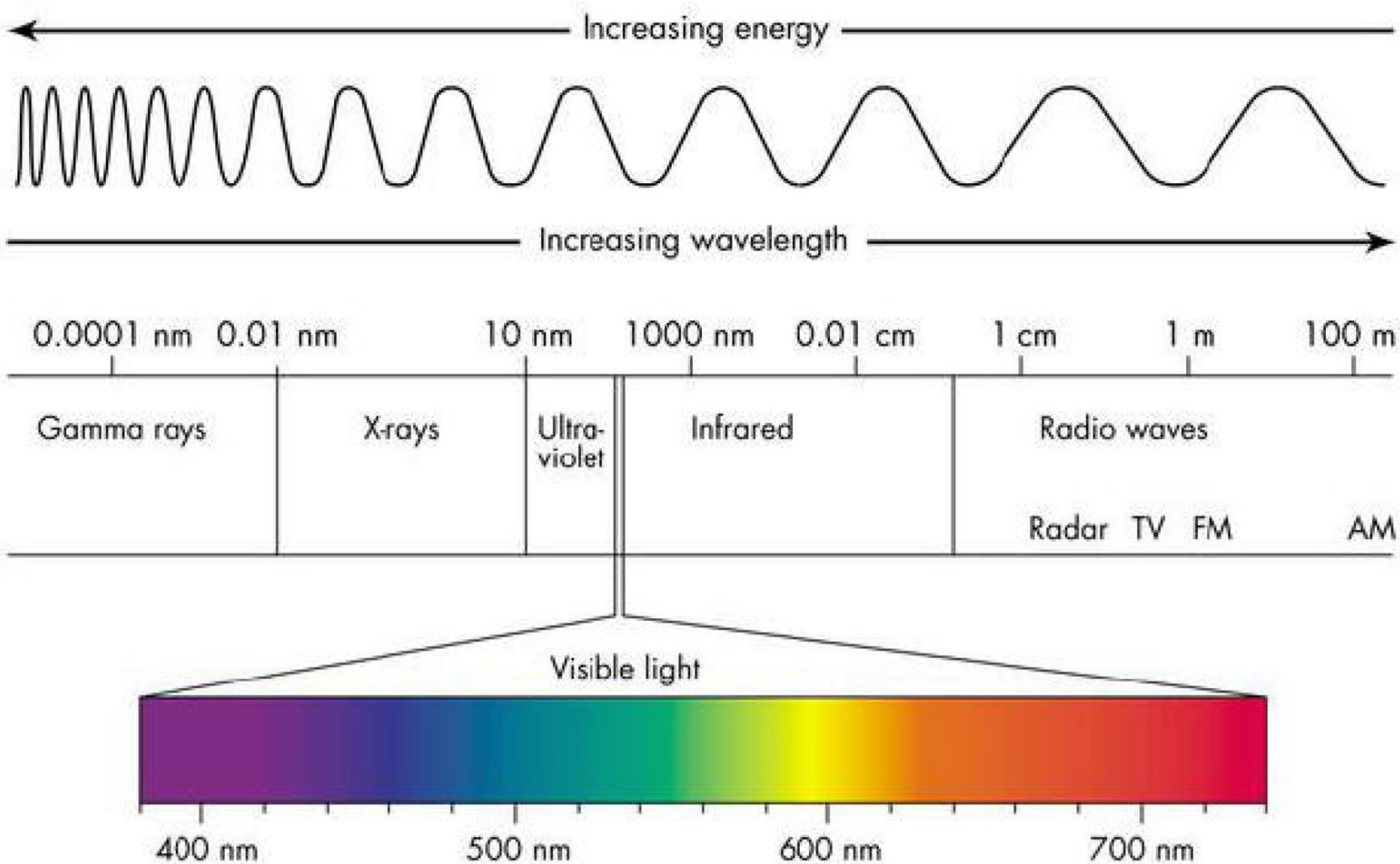
Computer Graphics

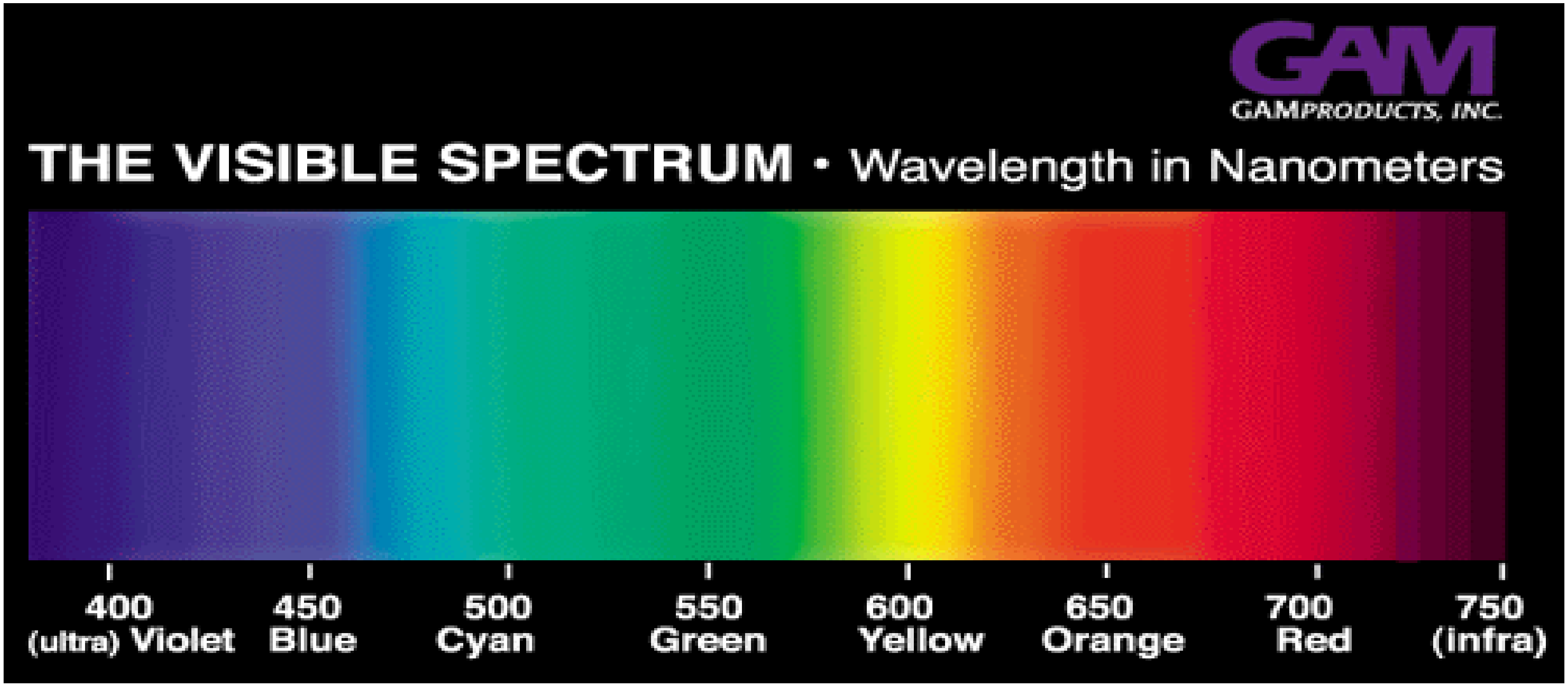
Lecture 12

Colors



Increasing energy
Increasing wavelength
0.0001 nm
0.01 nm
10 nm
1000 nm
0.01 cm
1 cm
1 m
100 m
Gamma rays
X-rays
Ultra-violet
Infrared
Radio waves
Radar
TV
FM
AM
Visible light
400 nm
500 nm
600 nm
700 nm

Visible Spectrum



The seven rainbow colors, each deffined by a single wavelength, are listed in the figure. Cyan may be replaced by Indigo (between Blue and Violet) in the list. White light is a mixture of wavelengths in equal radiances.

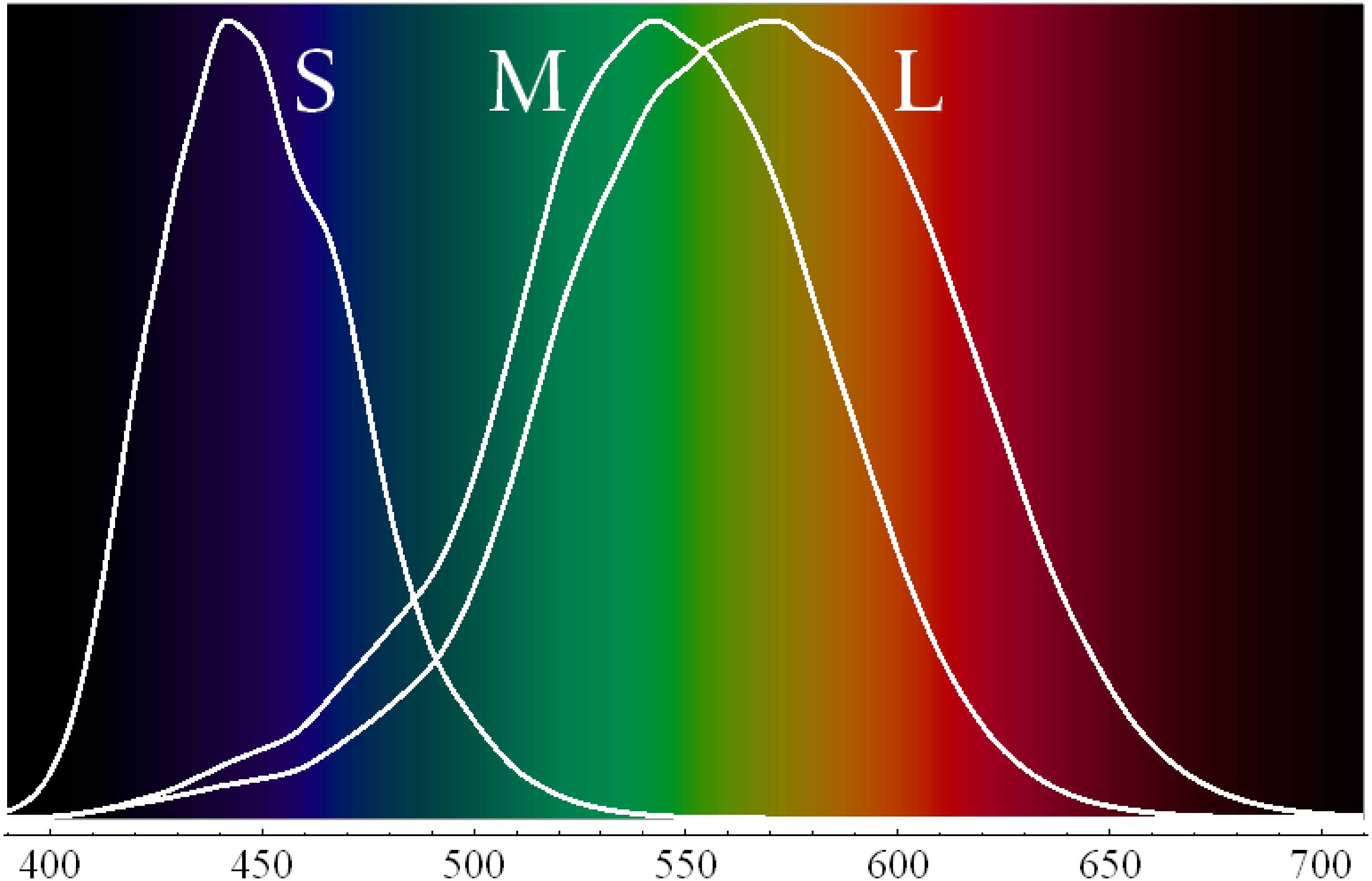
Light Intensity

The intensity of light may be described in three different ways.

1. Radiance (physical) is the rate of energy flow (power) measured in watts, for example.
2. Luminance (physical and physiological) is perceived energy from a light source, measured in lumens.
3. Brightness (psychological and perceptual) is a non-quantitative measure of intensity of a light-emitting source.

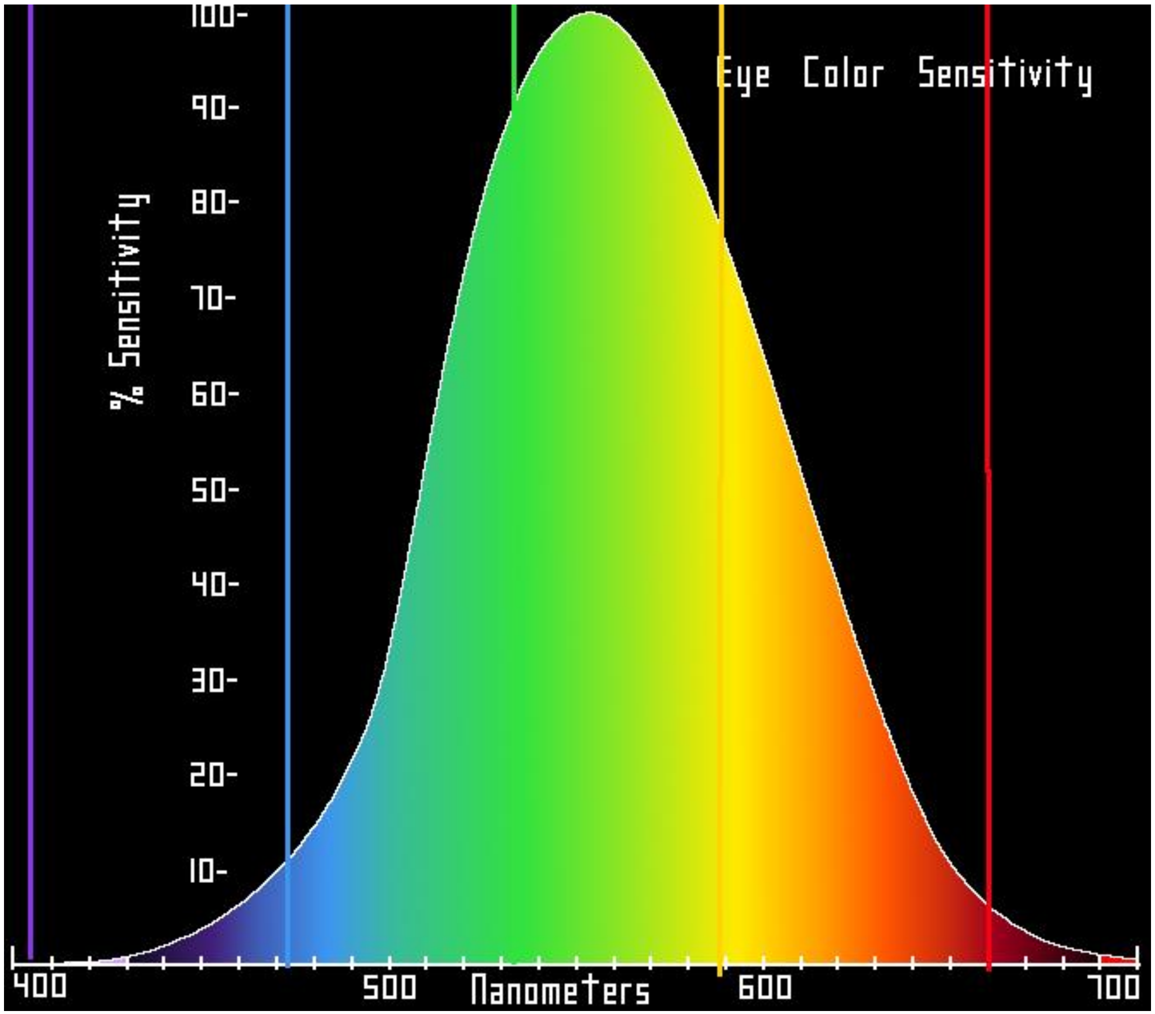
The term lightness, like brightness, is only meaningful in a relative sense, but refers to intensity of a light-reffecting source.

Normalized Cone Responses



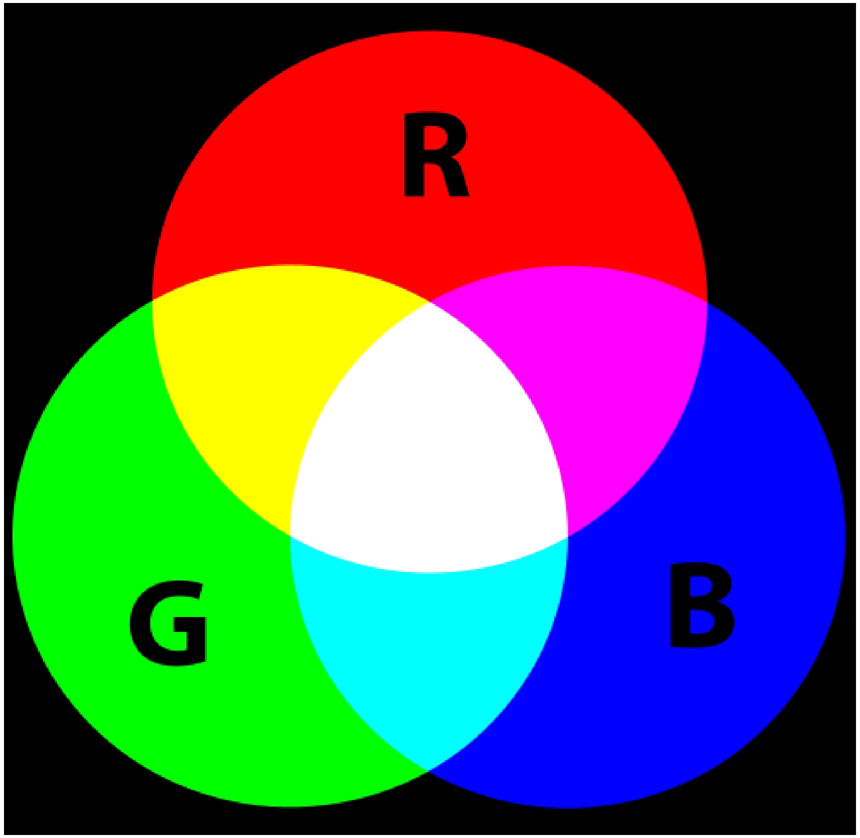
Cones (color sensors in the retina) are of three types defined by their peak response to Short (blue), Medium (green), or Long (red) wavelengths.

Color Sensitivity of the Eye



Relative brightness sensitivity of the human visual system as a function of wavelength

Additive Color Mixing

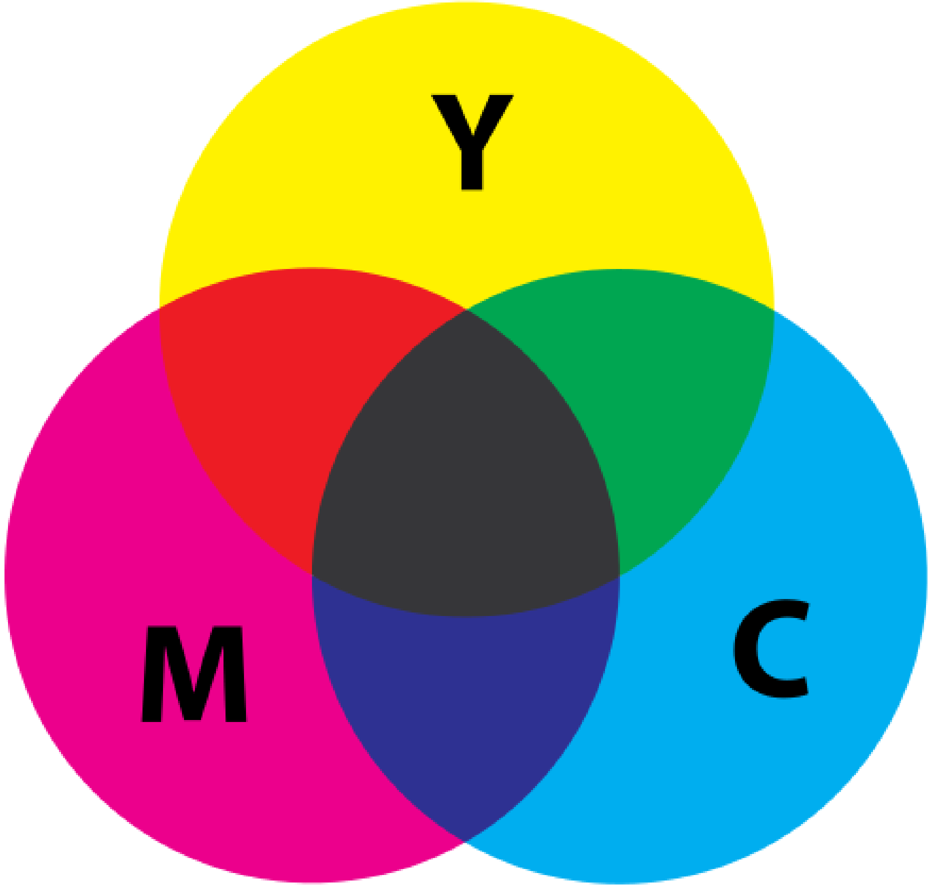


Red, Green, and Blue are primary colors for color mixing (light).

Mixtures of two primaries are secondary colors. Note that Cyan is a response to both a mixture of Green and Blue light and to a pure wavelength (which excites the Green and Blue cones).

Yellow is similar, but Magenta can only be obtained as a mixture.

Subtractive Color Mixing

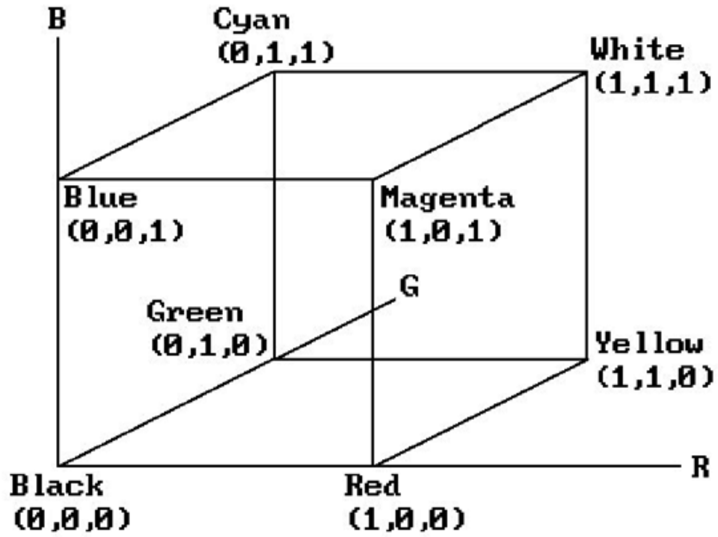


The primaries for mixing paints and dyes are the secondary colors for additive color mixing.

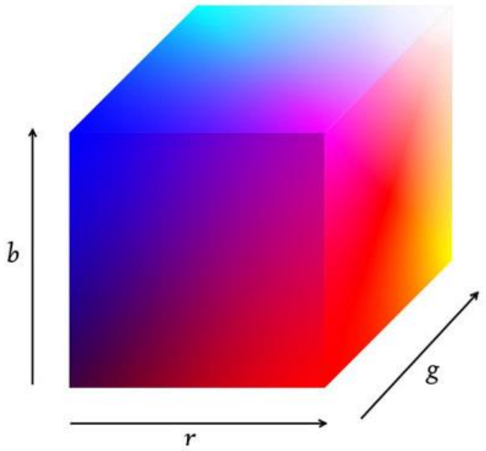
Cyan is obtained by absorbing Red (subtracting it from White), and Magenta subtracts Green.

The mixture is therefore Blue. The mixture of all three is black only in theory. Inkjet printers (CMYK) use black ink (K).

RGB Color Model

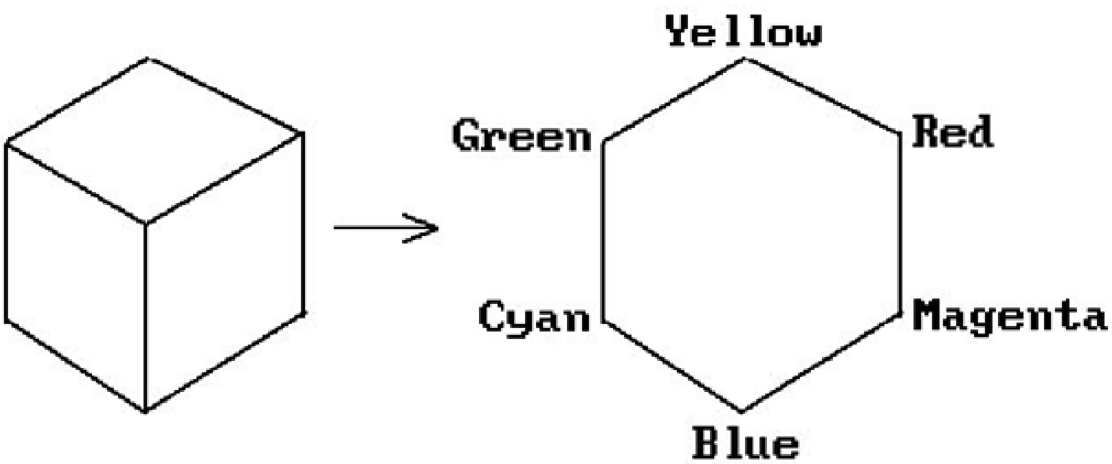


A color is defined by a triple of RGB intensities normalized to [0,1], and associated with a point in the unit cube. The diagonal connecting the origin to (1,1,1) is the grayscale line.

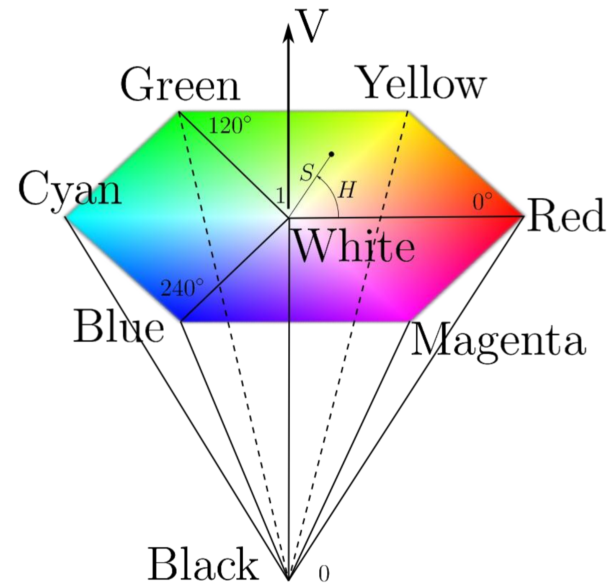


The Color Cube or RGB Model is the "natural" gamut model that represents the gamut as the unit cube [0,1] x [0,1] x [0,1].

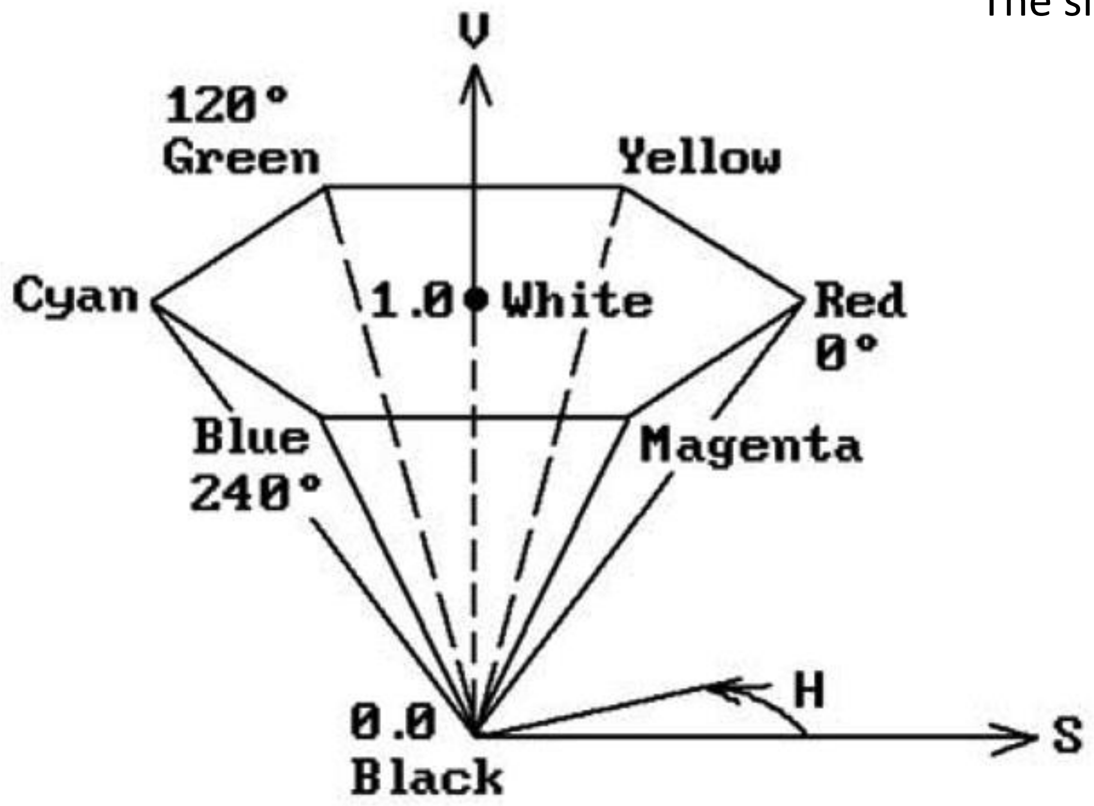
HSV Color Model



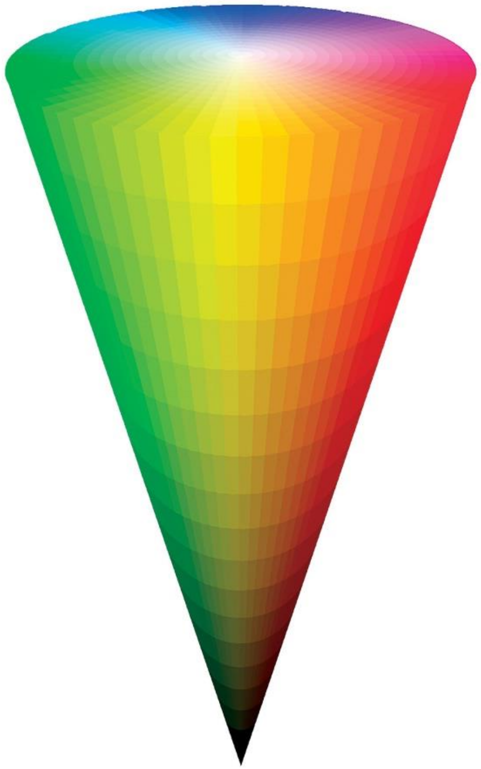
The RGB cube viewed along Black->White Diagonal



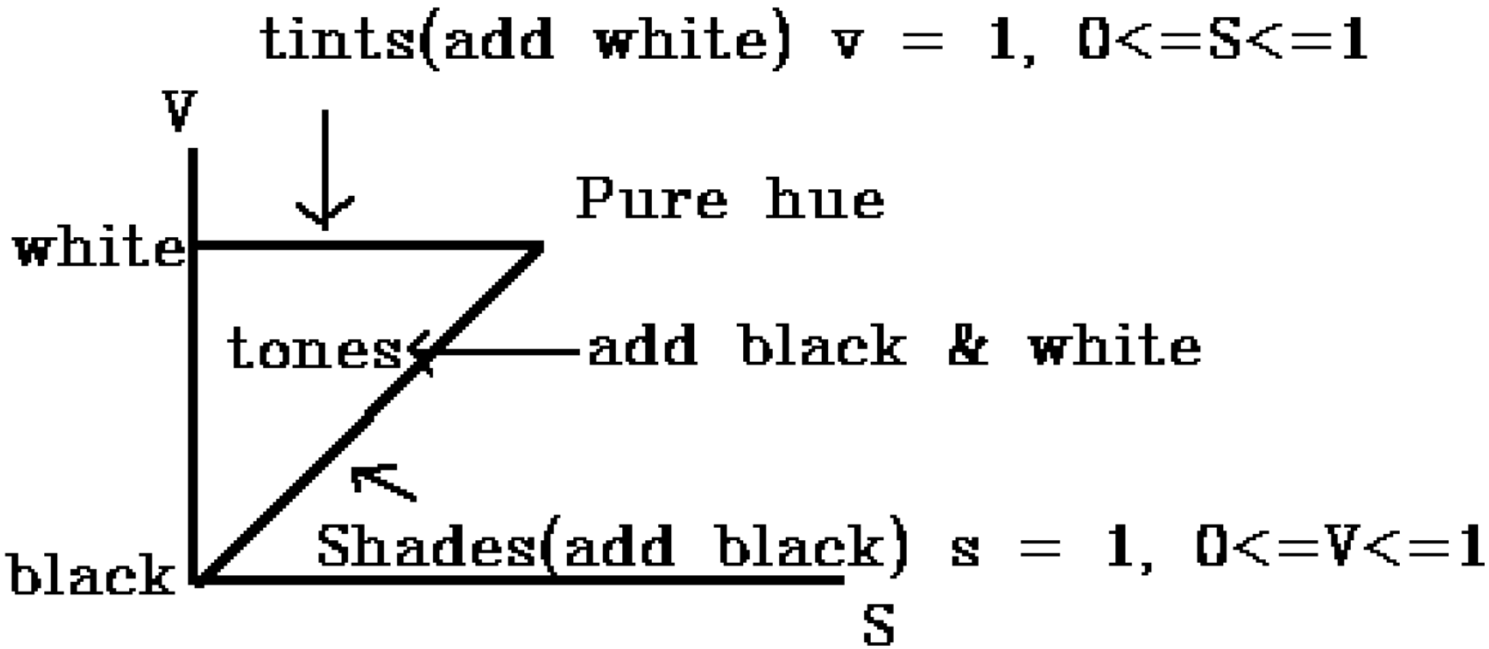
Hexcone



The single hexcone HSV color model



HSV Color Model continued



A cross section of the hexagonal pyramid associated with a particular hue is an artist's model: start with a pure hue of maximum intensity, and add black and/or white paint. Note that saturation S is the relative distance from the V axis (gray scale) to the boundary for fixed H and V.

Color Lookup Table

A color lookup table is a table that maps color indices into color values either in hardware or software. The VGA graphics adapter (the rst IBM PC standard for an analog display),

for example, included a lookup table consisting of 256 18-bit registers (six bits for each of R, G, and B). The limited quantity of video RAM allowed for storing only 8 bits per pixel which, rather than specifying a color directly, selected a color indirectly from a palette of size 218.

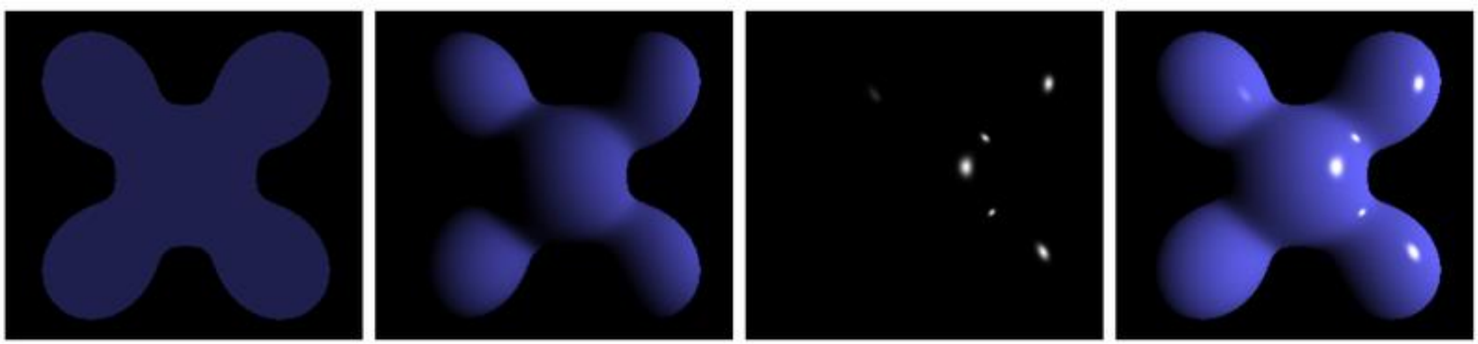
The number of simultaneously displayable colors, however, was limited to 256. The entire set of pixel colors can be altered simultaneously by changing the contents of the table. This is useful for special effects.

Video memory is now large enough to store true color (24 bits per pixel), but color lookup tables are still useful for storing pixmap image files efficiently.

Lights and Shading

```
const GLfloat light_ambient[]  = { 0.0f, 0.0f, 0.0f, 1.0f };
const GLfloat light_diffuse[]  = { 1.0f, 1.0f, 1.0f, 1.0f };
const GLfloat light_specular[] = { 1.0f, 1.0f, 1.0f, 1.0f };
const GLfloat light_position[] = { 2.0f, 5.0f, 5.0f, 0.0f };

const GLfloat mat_ambient[]    = { 0.7f, 0.7f, 0.7f, 1.0f };
const GLfloat mat_diffuse[]    = { 0.8f, 0.8f, 0.8f, 1.0f };
const GLfloat mat_specular[]   = { 1.0f, 1.0f, 1.0f, 1.0f };
const GLfloat high_shininess[] = { 100.0f };
```

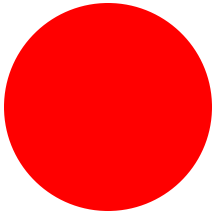


Ambient + Diffuse + Specular = Phong Reflection

Base Color

Base color is the color of the object

(C_R, C_G, C_B) – Base color



Consider the final light color of a pixel given by

$I_R = C_R \ldots$

$I_G = C_G \ldots$

$I_B = C_B \ldots$

Ambient Light

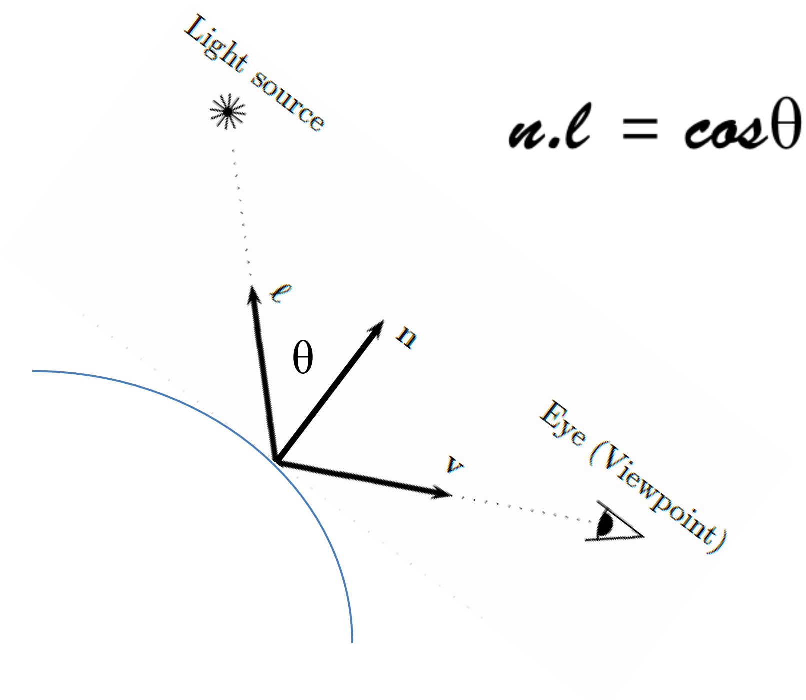
- Consider the Light source as a point
- Light has a color (amb_R, amb_G, amb_B)
- All background Light(Ambient Light) is define with a constant (***ambient coefficient***) defined by K_a *where* $K_a(amb_R, amb_G, amb_B)$

Increase K_a

$I_R = C_R(K_a\, amb_R \ldots$
$I_G = C_G(K_a\, amb_G \ldots$
$I_B = C_B(K_a\, amb_B \ldots$

Diffuse Light

Controls the relative intensity of diffusely reflected light



$$n.\ell = \cos\theta$$

Lambert's cosine law says that the radiant intensity or luminous intensity observed from an ideal diffusely reflecting surface or ideal diffuse radiator is directly proportional to the cosine of the angle θ between the direction of the incident light and the surface normal

Diffuse Light

Constant (***Diffuse coefficient***) defined by K_d

where $K_d(d_R, d_G, d_B) \rightarrow (diff_R, diff_G, diff_B)$

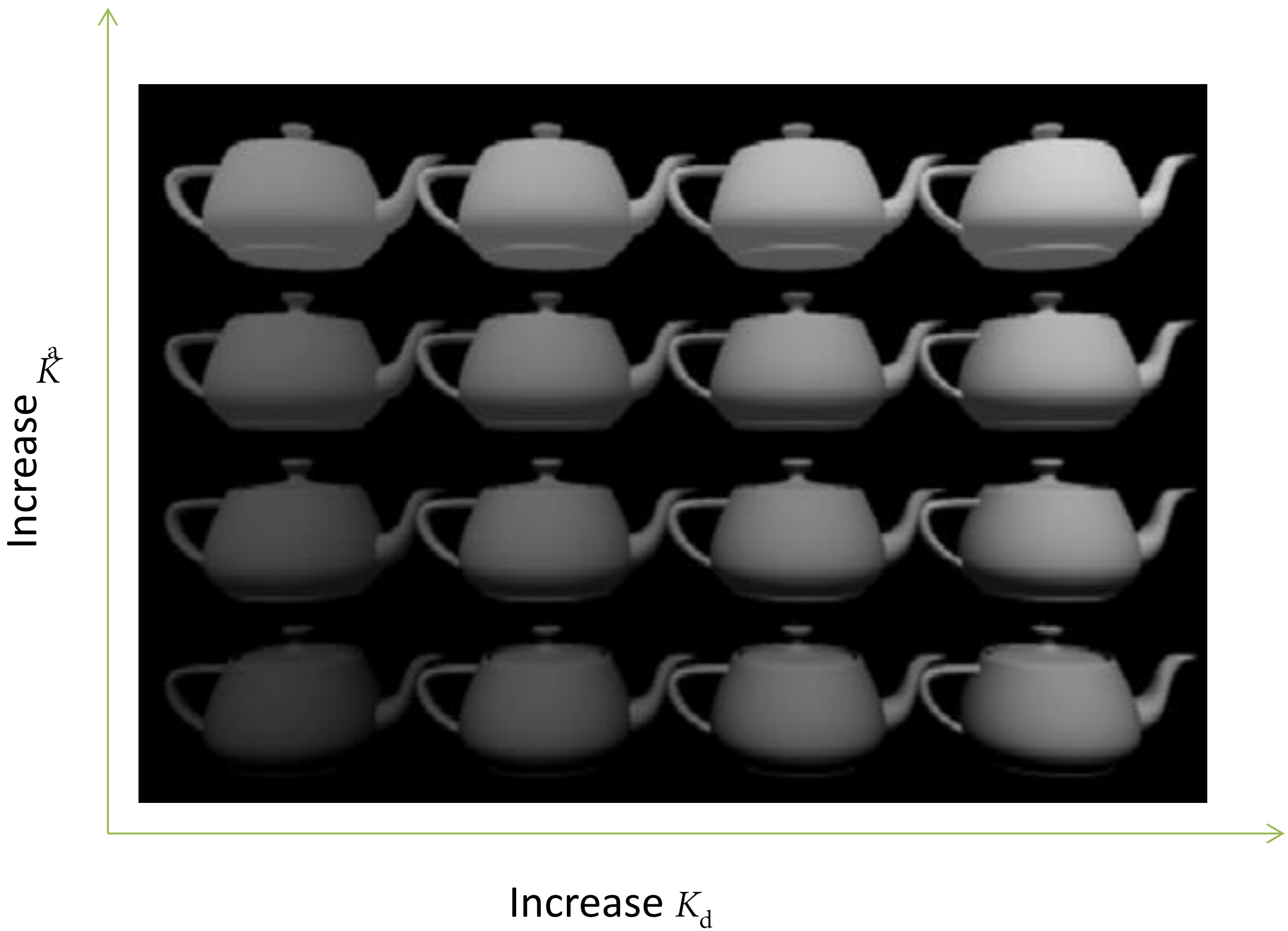
Increase K_d ($K_a = 0$)

$$I_R = C_R\,(K_a\, amb_R + diff_R\,(n.l)) \ldots$$

$$I_G = C_G\,(K_a\, amb_G + diff_R(n.l)) \ldots$$

$$I_B = C_B\,(K_a\, amb_B + diff_R\,(n.l)) \ldots$$

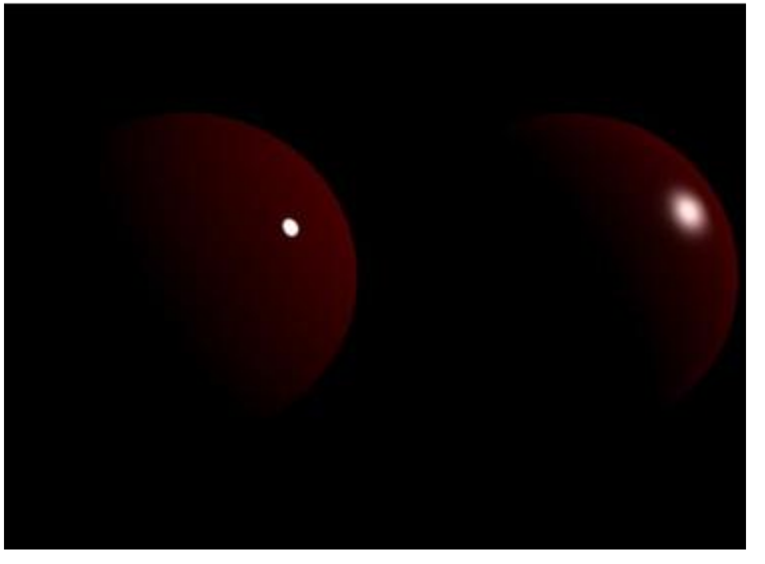
Ambient + Diffuse



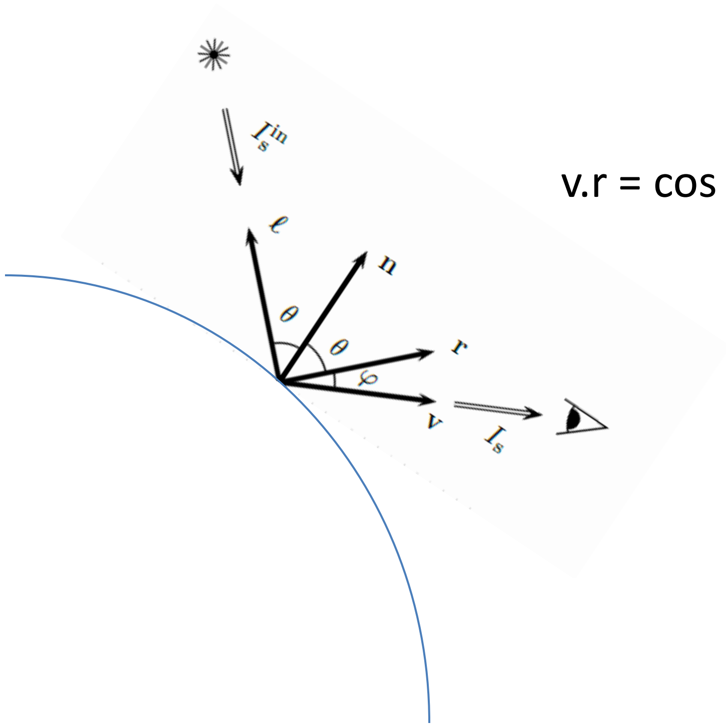
Specular Light

- Diffuse light not follow the eye position
- Specular handles the shininess of the object
- Controls the amount of specular reflection K_s

where $K_s(s_R, s_G, s_B)$



Specular Light



v.r = cos α

Phong Shading Model

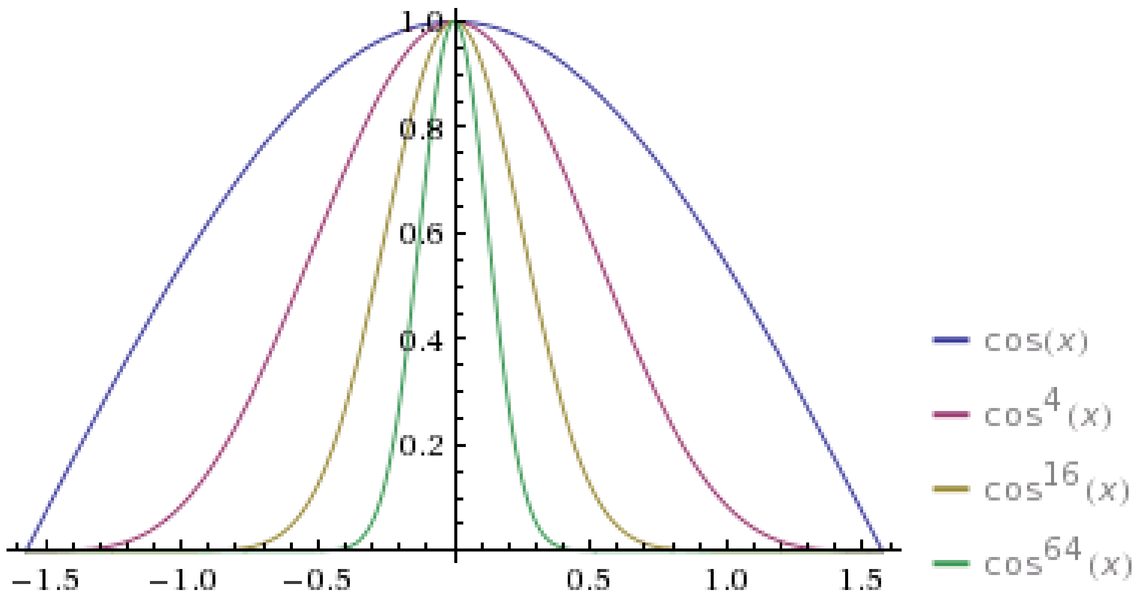
Since the v.r not giving enough intensity Specular light defined as

$(v.r)^n$ where *n* is the Glossiness factor

Phong's model

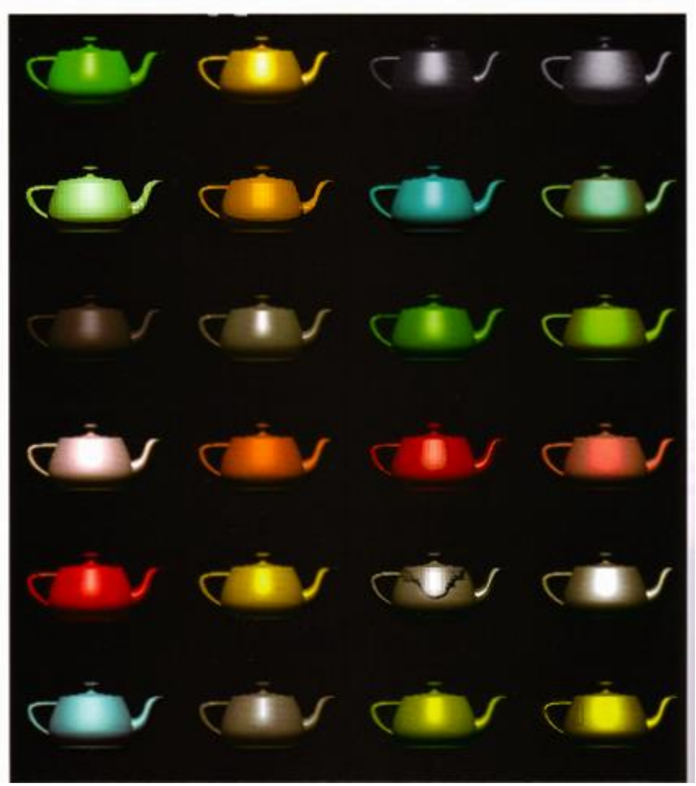
$(v.r)^n \rightarrow (\cos \alpha)^n$

Effect on Glossiness factor n



Specular Light

Light Model



Ambient + Diffuse + Specular

$$I_R = C_R\,(K_a\, amb_R + diff_R\,(n.l) + K_s\, Sp_R\,(v.r)^n\,)$$

$$I_G = C_G\,(K_a\, amb_G + diff_R(n.l) + K_s\, Sp_R\,(v.r)^n\,)$$

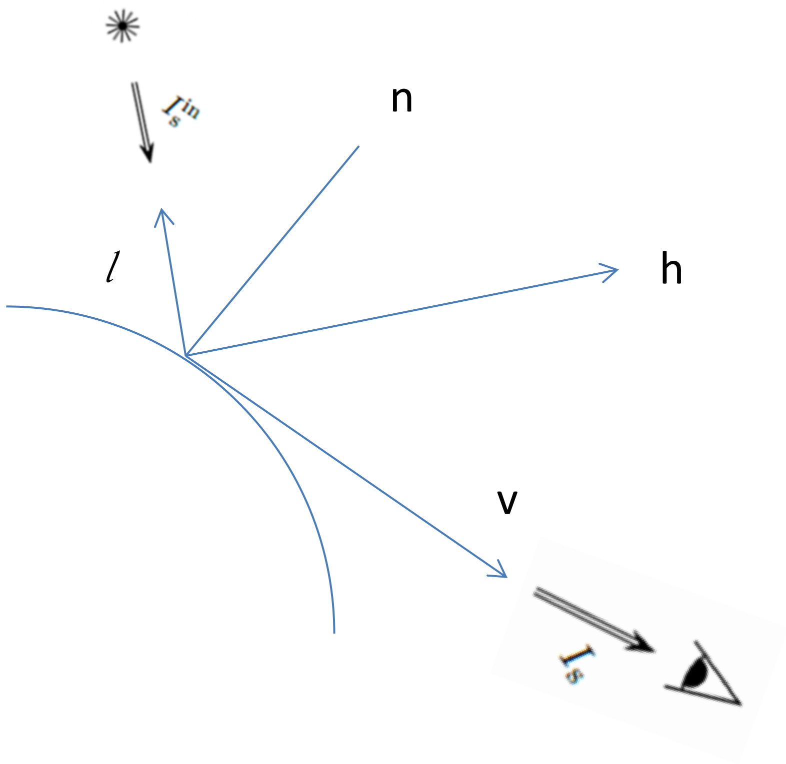
$$I_B = C_B\,(K_a\, amb_B + diff_R\,(n.l) + K_s\, Sp_R\,(v.r)^n\,)$$

Here other than light source and eye position we have to calculate the vector r for the specular light. In order to avoid this we usually use half angle vector(code hacks)

Blinn-Phong model

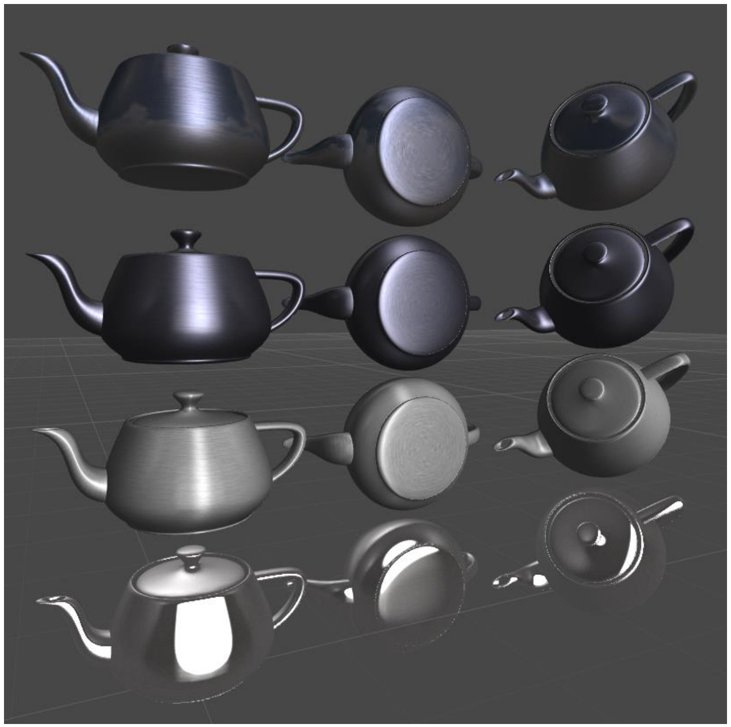
Let h be the half angle between l and v

where $h = l{+}v \,/\, \|\, l{+}v \,\|$



Replace vector of phong model by h since it is a faster calculation

Color model with Blinn-Phong



$$I_R = C_R\,(K_a\, amb_R + diff_R\,(n.l) + K_s\, Sp_R\,(v.h)^n\,)$$

$$I_G = C_G\,(K_a\, amb_G + diff_R(n.l) + K_s\, Sp_R\,(v.h)^n\,)$$

$$I_B = C_B\,(K_a\, amb_B + diff_R\,(n.l) + K_s\, Sp_R\,(v.h)^n\,)$$

Different combinations of cofactors of light in Shadermodels May result different effects