Light, Color and Materials

Light

Light is *electromagnetic radiation* in the *visible spectrum*.

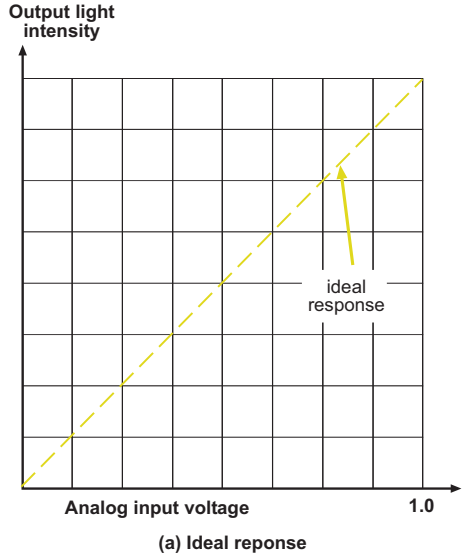
- The range of wavelength of visible light is from 400nm to 700nm (1 nm (nanometer) = 10^{-9} meters).
- Color is determined by the dominant wavelength of light that we perceive.
- The brightness is determined by the intensity of the radiated light.

Note: *Intensity* describes the physical amount of energy. *Brightness* describes our perception of this energy.

In developing a COLOR virtual camera we need to understand :

- the *physics* of light scattering by different materials (objects and the surrounding medium) is complicated.

- our *perception* of light is a function of our eyes, which perform numerous unconscious corrections and modifications. For example, equal intensity of colored light are perceived as being of different brightness depending on the color.
- *hardware devices* can only approximate colors to a limited precision.

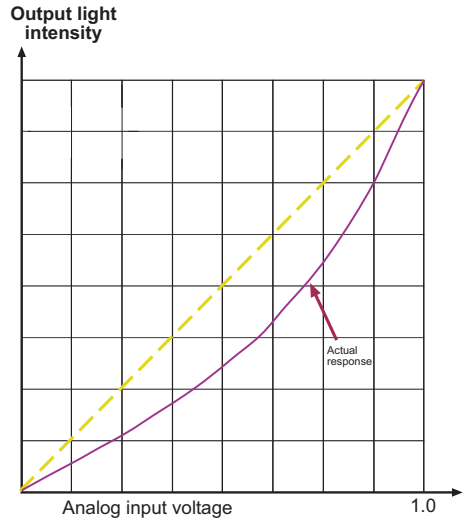


Gamma Correction

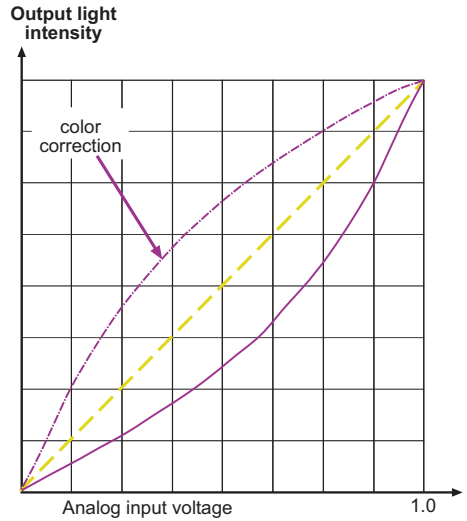
Achromatic light is only characterized by *intensity*. We assume that intensity values are specified as a number from 0 to 1.

Problem: Human perception system for light and color is nonlinear. Our eye cannot perceive the linear variation of brightness for linear variation of the intensities. The reason is that the eye is sensitive to the ratios of intensities, rather than absolute differences. Therefore, to achieve *perceptual* uniformity, intensities should be chosen on a *logarithmic*, rather than a linear scale.

Gamma correction: Most graphics monitors provide a form of automatic correction so that a linear variation in the supplied intensity values (e.g. RGB) generates a perceived linear variation in brightness.



(b) Actual response



(c) Color correction

Color Models

What is Color?

Color is perceived and reproduced depending on a complex interplay between:

- the physics of light radiated or reflected from materials
- the physiology of the human perception system
- the nature of the hardware device (CRT monitors, photographic and video cameras, offset printers, etc).

Our comprehensive visual system processes, the combination of sensors - rods (intensity) and 3 types of cones (red, blue, green wavelengths) data, and what we collectively, perceive is called *color*.

Because our perception of color is dependent on a combination of these values, most color systems describe color as a point in a 3-dimensional space.

Lightness (or *brightness*) are not physical quantities, but perceptual quantities.

- We are more sensitive to light in some spectral ranges than others. For example, given equal amounts of red, green and blue light, we will perceive the green as being the brightest, red second, and blue a distant third.
- Our perception system is *nonlinear*. It responds logarithmically to increases in intensity. That is, we perceive differences in terms of *ratios* of intensities rather than absolute differences.

See also for example http;//www.siggraph.org/education/materials/HyperGraph/color/coloreff.htm for some principles for the use of color

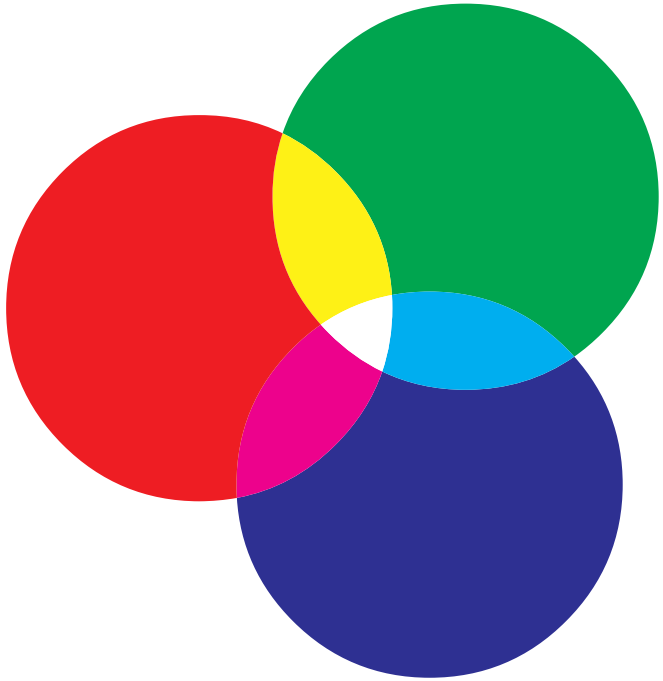
Luminosity

- For some interval of the electromagnetic spectrum, we can describe the physical quantity of *intensity* of light as the physical amount of energy that is incident to some material. This is sometimes called *linear light*.
- *Spectral Power Distribution* (SPD): Describes the amount of power of each wavelength of light in the visible spectrum.
- *Luminance* accounts for the sensitivity of our eye to different spectral ranges. For equal intensities of blue and green light, the blue light will have much lower luminance than the green light.
- Luminance is just intensity multiplied by a *luminance-efficiency function* for the human eye. This function is 0 outside the visible spectrum (400 to 700 nm) and peaks near green at around 555 nm.
- The luminance (linear brightness) of an SPD is a scalar quantity. To compute this you integrate the SPD using the luminous-efficiency function as a weighting function. This quantity is denoted by Y, by the CIE (Commission Internationale de l'Éclairage).
- Note that luminance is still a linear quantity. It does not take into account the logarithmic

nature of perception. Thus, gamma correction would have to be applied to scale the light to a perceptually smooth range of values.

RGB Color Model

The RGB model uses the additive primaries red, green, and blue. So (assuming our normalization from 0 to 1), red is $(1, 0, 0)$, green is $(0, 1, 0)$, and blue is $(0, 0, 1)$. These colors can be added to make other colors, black is $(0, 0, 0)$, white is $(1, 1, 1)$.



Three primaries: red, green, and blue

CMY Color Model

RGB is an additive color system. However, most printing devices use a subtractive system. This means that the ink absorbs a particular color, and hence what you see is what is reflected (not absorbed). When inks are combined, they absorb a combination of colors, and hence the reflected colors are reduced, or subtracted.

Subtractive primaries are cyan, magenta, and yellow. It is very easy to convert from RGB to CMY (and vice versa), by simply subtracting each component from 1

$$(C, M, Y) = (1 - R, 1 - G, 1 - B).$$

Thus in the CMY model, black is $(1, 1, 1)$ and white is $(0, 0, 0)$.

A variation of CMY is CMYK, where an extra redundant component K has been added to represent black.

YIQ Color Model

The YIQ model is used in the US for color television broadcasting. It is a recording of RGB with some tricks for broadcasting efficiency and compatibility with black-and-white. The Y component is equal to the Y component of the CIE standard. This is basically a measure of luminance, so black-and-white sets simply display this one component. The I and Q components encode hue and saturation. Since our visual system is more sensitive to variations in luminance, the I and Q components can be transmitted with lower bandwidth that what is used for Y.

User oriented systems

The models RGB, CMY, and YIQ are tailored for particular devices (CRT's, printers, and video). User's find these systems difficult to work with. For this reason there are systems that describe colors in terms that are more natural. For example, the HSV system defines color in terms of hue (color), and uses saturation (purity), and value (lightness) to add white or black. Changing the saturation parameter corresponds to adding/subtracting white and changing the value parameter corresponds to adding or subtracting black.

The HSV is derived from the RGB cube. Looking down the gray long diagonal one sees a hexagon that is the HSV hexcone. The hue is given by the angle about the vertical axis with red at 0 degress yellow at 60 degrees, green at 120 degrees, cyan at 180 degrees, blue at 240 degrees, and magenta at 300 degrees. Primary colors are 120 degrees apart. Saturation S varies between 0 and 1, is ratio of purity of a related hue to its maximum purity at saturation equal to 1 (i.e. concentric hexagons). Also S = 0 yields the grey-scale corresponding to Value V of HSV i.e. the diagonal of the RGB cube, with V =0 being black and V = 1 corresponding to white.

To choose a color we first select a pure hue, and set S = V = 1. Next to add black we

decrease V and/or to add white we decrease S. E.g. Pure Blue : H=240deg S = V = 1; Dark Blue: H = 240deg, S =1 , V =0.4 and Light Blue: H=240deg, S = 0.3, V = 1.0

Light Sources in OpenGL

Luminance function:

$$I = (I_r, I_g, I_b)$$

Components of OpenGL local illumination model:

glLightfv(source,parameter,pointer-to-array)

glLightf(source,parameter,value)

- ambient emission

$$I_a = \begin{pmatrix} I_{ar} \\ I_{ag} \\ I_{ab} \end{pmatrix}$$

- point emission

$$I(p_0) = \begin{pmatrix} I_r(p_0) \\ I_g(p_0) \\ I_b(p_0) \end{pmatrix}$$

GLfloat ambient0[]=1.0,0.0,0.0,1.0

GLfloat diffuse0[]=1.0,0.0,0.0,1.0

GLfloat specular0[]=1.0,1.0,1.0,1.0

GLfloat light0-pos[]=1.0,2.0,1.0,1.0

Note that the light position (1,2,1) above is a homogeneous 4 vector. So to create a distant light source (i.e. at infinity) we set the light direction vector (1,2,1) with the homogeneous component set to 0.

GLfloat light1-pos[]=1.0,2.0,1.0,0

glLightfv(GL-LIGHT0,GL-AMBIENT,ambient0)

glLightfv(GL-LIGHT0,GL-POSITION,light0-pos)

Lights can be set with an attenuation that is a function of distance from the point light position. The attenuation can be constant, linear, or quadratic

Attenuation function:

$$I(p, p_0) = \frac{1}{a + bd + cd^2} I(p_0)$$

where $d = |p - p_0|^2$.

Example usage,

glLightf(GL-LIGHT0,GL-CONSTANT-ATTENUATION,a)

Lighting calculations must be enabled, and each light source must be enabled individually. For a single source we could use

glEnable(GL-LIGHTING); glEnable(GL-LIGHT0);

Color in OpenGL

Color can be specified in RGB, RGBA and color-index mode. Unless your using lighting or texture mapping each object is drawn using the current color.

Specifying Color in OpenGL is via the glColor*() command

glColor4b s i f d ub us ui (TYPE r, TYPE g, TYPE b, TYPE a) glColor3b s i f d ub us uiv (const TYPE *v)

suffix 3 or 4 indicates whether you supply an alpha value or not (default is 1.0). The second suffix indicates type for parameters byte, short, integer, float, double, unsigned byte, unsigned short, unsigned integer. Optional v, indicates that the argument is a pointer to an array of values of the given data type.

Example to specify red color for an object (in this case a point):

glColor3f(1.0, 0.0, 0.0) glBegin (GL-POINTS); glVertex3fv(point-array); glEnd ();

With color-index mode, OpenGL uses a color map (a lookup palette). In OpenGL use the glindex() command to select a single-valued color index as the current color index.

For most systems more colors can be simultaneously represented with RGBA mode than with color-index mode. Also for shading lighting, texture mapping, fog etc, RGBA provides more flexibility.

Color-index mode can be useful for color-map animation and drawing in layers.

Setting Material Properties in OpenGL

By setting material properties, we can model objects that radiate or relect specific wavelengths of Light. In OpenGL, material properties match up directly with the supported light sources, and a certain Illumination Model based on reflection. More on this Illumination Model next time. The commands to set the material properties for diverse reflection behaviors, are

glMaterialfv(face,type,pointer-to-array)

glMaterialf(face,value)

Materials can be modeled to have different ambient, diffuse and specular reflectivity coefficients

GLfloat ambient[]=1.0,0.0,0.0,1.0

GLfloat diffuse[]=1.0,0.0,0.0,1.0

GLfloat specular[]=1.0,1.0,1.0,1.0

Additionally, GL-SHININESS is used to specify a shininess of a material surface. As we shall see later, it shall be used in the Phong illumination model in OpenGL. GLfloat shininess = 100.0

and then specified for example as,

glMaterialfv(GL-FRONT-AND-BACK,GL-AMBIENT,ambient)

glMaterialfv(GL-FRONT-AND-BACK,GL-SPECULAR,specular)

OpenGL also allows materials to have an emissive component that characterizes self-luminous sources.

GLfloat emission[]=0.0,1.0,0.0,1.0

Example usage,

glMaterialfv(GL-FRONT,GL-EMISSION,emission)

For examples see color plates in OpenGL red book http://glprogramming.com/red/appendixi.html

Reading Assignment and News

Please review the appropriate sections related to this weeks' lectures in chapter 5, and associated exercises of the recommended text.

(Recommended Text: Interactive Computer Graphics, by Edward Angel, Dave Shreiner, 6th edition, Addison-Wesley)

Please track Blackboard for the most recent Announcements and Project postings related to this course.

(http://www.cs.utexas.edu/users/bajaj/graphics2012/cs354/)