

Chapter 1

Color

In this chapter we will investigate more carefully what is meant by color. Thusfar, we have represented color images by using three values, red green and blue. Why have we done this? Can I obtain all possible colors using just red green and blue? What is the relationship between the wavelenght of light and its color?

In short, we will see that many distributions of light can appear the same to a human observer, and we say that all of these lights have the same color. It turns out that these colors can be treated as vectors in a three dimensional space. There are various color spaces coordinate systems we can use to describe colors including ones called x, y, z and r, g, b .

Finally we we will look at gamma correction which is a non-linear color mapping which is commonly used in image representation.

1.1 Color As a linear space

Light is a physical phenoniminon called elocrtromagnetic radiation. Monochromatic electric radiation comes in different flavors called wavelenghts λ . Humans are sensitive to light with $380 > \lambda > 770$ measured in nanometers. Most real world light sources are not monochromatic, but are a combination of lights of various freqencies. We can describe each light source by $l(\lambda)$, the distribution of light of various frequencies that make up this light.

Humans have various kinds of light sensitive cells as part of our eyes. Rods are used for black and white vision under dark lighting conditions, while cones are used for color vision under adaqately illuminated scenes.

The fact that color can be treated as a three dimensional linear space is intrinsicially tied up with the the fact that humans have three kinds of cones. Each of these types of cells has particular sensitivities to various wavelenghts. For example, one of the cone

types is more sensitive to low frequency wavelength light. When encountering some distribution of light, each of the three kinds of cones fire with some intensity. The amount of the three kinds of firing gives us the experience of color. If two different light distributions give rise to the same amount of firing, then they will create the same color experience. In this case, we call these two distributions metamers.

Some people only have two kinds of cones. Their experience of color comes only from the two kinds of firing. As a result, they cannot distinguish some distributions, that would be distinguished by the third cone firing amount. These people are called colorblind.

Let us consider all light distributions that give create the same color experience as some particular distribution $l_1(\lambda)$. Let us call this entire group of light distributions $\vec{c}(l_1(\lambda))$. If l_1 and l_2 are metamers, then $\vec{c}(l_1) = \vec{c}(l_2)$. We will call these equivalence classes of light distributions “colors”.

What happens when we try to treat these colors as vectors. First we must define addition of colors and scalar multiplication. Let us define the operations as follows

$$\vec{c}(l_1(\lambda)) + \vec{c}(l_2(\lambda)) = \vec{c}(l_1(\lambda) + l_2(\lambda))$$

and

$$\alpha \vec{c}(l(\lambda)) = \vec{c}(\alpha l(\lambda))$$

These operations will only make sense if their application gives the same result for all light distribution in the color equivalence class. Otherwise, we can't really say that the operations act on the colors themselves. For example, if

$$\vec{c}(l_1(\lambda)) = \vec{c}(l_2(\lambda))$$

then we want

$$\vec{c}(l_1(\lambda)) + \vec{c}(l_3(\lambda)) = \vec{c}(l_2(\lambda)) + \vec{c}(l_3(\lambda))$$

and

$$\vec{c}(l_1(\lambda)) = \vec{c}(l_2(\lambda))$$

In fact this is true, as can be verified by experiment. A similar experiment shows that our definition of scalar multiplication is valid.

A true linear space must be closed under addition and scalar multiplication. So for example $-\vec{c}l$ must be a valid color. But using our rules $-\vec{c}l = \vec{c}(-l)$. This says that we may have a color whose equivalence class is made up of light distributions with negative light. Of course negative light does not exist. This is not a real problem. We can “extend” our linear space to be closed under the vector operations. The closed space will include these “unreal” colors in our space, even though they are not equivalence classes of metameric real light distributions. This approach will work as long as

these unreal colors behave consistently with the real ones. For example if $\vec{c}_1, \vec{c}_2, \vec{c}_3$ are real colors then

$$\begin{aligned}\vec{c}_1 + -\vec{c}_2 &= \vec{c}_3 \text{ iff} \\ \vec{c}_1 &\equiv \vec{c}_3 + \vec{c}_2\end{aligned}$$

To summarize, we can define colors as equivalence classes of light distributions that appear the same to human observers. All of the light distributions on a color not only look the same, but they act the same when added with other light distributions or scaled by α . Therefore we can define vector addition and scalar multiplications over these colors. Finally, we do have to include unreal colors to give us a linear space that is closed under the vector operations.

1.2 Color matching experiments

The color matching experiment is an experiment which demonstrates that the dimension of color space is three. It also gives us a basis for the space. Finally it gives us a formula for transforming a light distribution into a color coordinate vector.

Here is how the experiment works. A person watches two screens. On the right screen, he is shown a monochromatic “target” light of some fixed intensity and some wavelength λ . On the left screen, the user is shown a superposition of three controllable monochromatic light sources. The wavelengths of the light sources happen to be 444, 526 and 645. The intensities of the three lights is set by the subject using three dials. The subject tries to set the three dials so that the observed colors on both screens look identical.

If the user succeeds then we record the three numbers on the three dials as $k_{444}(\lambda), k_{526}(\lambda), k_{645}(\lambda)$. These will be three positive numbers. If the user cannot succeed, then they are allowed to use negative amounts of the control lights as well. This is accomplished by moving one of the three controllable lights over to the right side of screen. Then they can try to match the superposition of the target and the moved control light against the remaining two control lights. If they succeed, then the moved light is considered as if it was unmoved negative light. This is consistent with our ideas of negative colors described above. In this case we will record a negative number for one of the knob values $k_{444}(\lambda), k_{526}(\lambda), k_{645}(\lambda)$. The resulting three k functions are called matching functions.

In this experiment, the doggedly determined user can succeed in matching all of the visible wavelengths λ using just the three control lights, providing they are allowed to use negative dial settings as well. This demonstrates that color is a three dimensional linear space of vectors.

Since color is a three dimensional linear space, all colors can be expressed as a linear combination of basis colors. Our first choice for a basis will be the colors of the three control lights $[\vec{c}_{444}, \vec{c}_{526}, \vec{c}_{645}]$. The knob settings from the experiment tell us the

coordinates of the color of the monochromatic light $\vec{c}(\lambda)$ with respect to this basis.

$$\vec{c}(\lambda) = \begin{bmatrix} \vec{c}_{444} & \vec{c}_{526} & \vec{c}_{645} \end{bmatrix} \begin{bmatrix} k_{444}(\lambda) \\ k_{526}(\lambda) \\ k_{645}(\lambda) \end{bmatrix}$$

Because color is linear, the coordinates of the color obtained by adding two lights together is simply the addition of the coordinates of the two lights. Real light distributions $l(\lambda)$ are combinations of not just two lights, but light of all wavelengths. Thus we can obtain the color coordinates of a general light distribution as

$$\vec{c}(l(\lambda)) = \begin{bmatrix} \vec{c}_{444} & \vec{c}_{526} & \vec{c}_{645} \end{bmatrix} \begin{bmatrix} \int d\lambda l(\lambda)k_{444}(\lambda) \\ \int d\lambda l(\lambda)k_{526}(\lambda) \\ \int d\lambda l(\lambda)k_{645}(\lambda) \end{bmatrix}$$

1.3 Other color bases

We have seen already that vectors in a linear space can be described by many different bases, or coordinate systems. One way to represent a different basis is with a (non degenerate) 3 by 3 matrix. This matrix tells us how to represent the new basis vectors as linear combinations of the original basis vectors. With color, another way to describe a new basis is to describe an example light distribution for each of the three basis colors. Finally, a third way to describe a basis is by its three matching functions. Here we will explore the relationship between these three aspects.

One way to describe a new basis, is to give example light distributions for each of the three new basis colors. Let us call these three light distribution functions $e_a(\lambda)$, $e_b(\lambda)$, $e_c(\lambda)$.

Each of the new colors can be written as a linear combination of the original basis colors, which can be written as the matrix equation

$$\begin{bmatrix} \vec{c}_a & \vec{c}_b & \vec{c}_c \end{bmatrix} = \begin{bmatrix} \vec{c}_{444} & \vec{c}_{526} & \vec{c}_{645} \end{bmatrix} \begin{bmatrix} M_{1,1} & M_{1,2} & M_{1,3} \\ M_{2,1} & M_{2,2} & M_{2,3} \\ M_{3,1} & M_{3,2} & M_{3,3} \end{bmatrix}$$

The matrix elements in a column are simply the coordinates of one of the basis colors. These coordinates can be determined by integrating the example light distribution against the matching functions. So, for example

$$M_{1,1} = \int d\lambda e_a(\lambda)k_{444}(\lambda)$$

We can also derive the new matching functions to use with the new basis. The three new matching functions will be linear combinations of the three original matching functions.

$$\begin{aligned}
\vec{c}(\lambda) &= \begin{bmatrix} \vec{c}_{444} & \vec{c}_{526} & \vec{c}_{645} \end{bmatrix} \begin{bmatrix} k_{444}(\lambda) \\ k_{526}(\lambda) \\ k_{645}(\lambda) \end{bmatrix} \\
&= \begin{bmatrix} \vec{c}_{444} & \vec{c}_{526} & \vec{c}_{645} \end{bmatrix} \mathbf{M} \mathbf{M}^{-1} \begin{bmatrix} k_{444}(\lambda) \\ k_{526}(\lambda) \\ k_{645}(\lambda) \end{bmatrix} \\
&= \begin{bmatrix} \vec{c}_a & \vec{c}_b & \vec{c}_c \end{bmatrix} \begin{bmatrix} k_a(\lambda) \\ k_b(\lambda) \\ k_c(\lambda) \end{bmatrix}
\end{aligned}$$

And so

$$\mathbf{M}^{-1} \begin{bmatrix} k_{444}(\lambda) \\ k_{526}(\lambda) \\ k_{645}(\lambda) \end{bmatrix} = \begin{bmatrix} k_a(\lambda) \\ k_b(\lambda) \\ k_c(\lambda) \end{bmatrix}$$

1.4 XYZ color space

The most standard color basis is called the XYZ basis. It is made up of three basis colors $[\vec{c}_x, \vec{c}_y, \vec{c}_z]$. All colors are described in this basis with three coordinates $[X, Y, Z]^t$. Given a light distribution, its color coordinates can be computed by integrating it against the matching functions $k_x(\lambda), k_y(\lambda), k_z(\lambda)$.

The XYZ color space was in fact officially described by the set of matching functions. These functions are of course some linear combination of the matching functions from the color matching experiment. These functions were chosen to have other useful properties as well. k_y was chosen so that

$$Y = \int d\lambda l(\lambda) k_y(\lambda)$$

described the overall perceived brightness of the light $l(\lambda)$. In this sense the Y coordinate of a color describes its black and white brightness. Also the functions were chosen so that they are positive everywhere. As a result, all monochromatic lights must have positive $[X, Y, Z]^t$ coordinates. Therefore, all real colors, which are non negative combination of monochromatic light, must also have positive $[X, Y, Z]^t$ coordinates.

All of monochromatic colors are in the first octant, and they are all on the surface of the horseshoe shaped region shown in figure !!. All non-monochromatic lights have their color in the interior of this horseshoe shaped region.

What about the basis colors \vec{c}_x, \vec{c}_y and \vec{c}_z . These colors are outside of the horseshoe shaped region. This means that they cannot be obtained using non negative linear combinations of monochromatic light. They can only be obtained by using some amount of negative light. So surprisingly enough, in the XYZ color coordinate system, which is the most agreed upon color standard, the basis colors themselves are unreal colors.

This is not an accident. The horseshoe shaped region is a convex 3D region with a curved boundary. Meanwhile, the color region obtainable from non negative combinations of three basis colors must be the three sided open ended pyramid defined by the basis vectors. Colors outside this pyramid will have negative coordinates. Thus, if one chooses three basis colors on (or within) the horseshoe, there will be some real colors that can only be obtained using negative amounts of those basis colors. If we want a coordinate system where all real colors have only positive coordinates, we must choose basis colors that are outside the horseshoe, and are unreal colors.

1.4.1 x y color diagram

In a the 3d linear color space, the color $[0, 0, 0]^t$ is what we call black. If we multiply some non zero color by a scalar value greater than one, we get a brighter version of the same color. To name colors, it is often useful to factor out the brightness. One common way of doing this is with the x, y color diagram.

A color $[X, Y, Z]^t$ is normalied to x, y as follows

$$\begin{aligned} x &= \frac{X}{X + Y + Z} \\ y &= \frac{Y}{X + Y + Z} \end{aligned}$$

A picture of the diagram is shown in figure !!! . In this diagram, monochromatic colors are along the outer horseshoe. We describe these colors as saturated. Colors obtained by non monochromatic light are in the interior of the shoe. As we move towards the center, the colors get desaturated, and white/grey is in the center.

Two colors that add to white/grey are called complementary. In the diagram, complementary colors are equidistant from white along a single line. If non white color can be obtained by mixing white with a monochromatic color $\vec{c}(\lambda)$, then we say that λ is its dominant frequency. There are non white colors that cannot be represented as such a mixture. These are called non spectral colors. These colors have no saturated version in a rainbow. Purple ? is such an example.

1.5 RGB space

We produce colors on a monitor by combining three phosphors we call red green and blue. Each of these phosphors produces some fixed distribution of light which is observed as some fixed color. These three colors form yet another linear basis for the three dimensional color space. We call the coordinates of a color with respect to this basis $[R, G, B]^t$. The relationship between these coordinates and XYZ coordinates is expressed by the matrix equation

$$\begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = \begin{bmatrix} .4124 & .3576 & .1804 \\ .2126 & .7152 & .0722 \\ .0193 & .1191 & .9502 \end{bmatrix} \cdot \begin{bmatrix} R \\ G \\ B \end{bmatrix}$$

$$\begin{bmatrix} R \\ G \\ B \end{bmatrix} = \begin{bmatrix} 3.2405 & -1.5372 & -.4985 \\ -.9693 & 1.8760 & .0416 \\ .0193 & .1192 & .9502 \end{bmatrix} \cdot \begin{bmatrix} X \\ Y \\ Z \end{bmatrix}$$

The values achievable on a monitor are those where the RGB coordinates are each between zero and one. Coordinates with $R = G = B$ are grey colors. $[0, 0, 0]^t$ is black while $[1, 1, 1]^t$ is white. Because the three basis colors are real colors, we cannot actually obtain all possible real colors by non-negative combinations of these colors. That is, some real colors have a negative coordinate in this basis. The space of colors obtained by non negative combinations is called the “gamut” of the monitor.

1.5.1 Subtractive colors

Sometimes it is useful to think of colors as acting subtractively. For example as one mixes paints together, less light is reflected by the mixture. To do this we use CMY (cyan magenta yellow) color coordinates. The relationship to RGB is defined as

$$\begin{bmatrix} C \\ M \\ Y \end{bmatrix} = \begin{bmatrix} 1 - R \\ 1 - G \\ 1 - B \end{bmatrix}$$

This is an affine, and not linear transformation, CMY coordinates are actually affine coordinates in an appropriate frame.

1.6 Gamma correction

In a computer monitor, the voltage levels fed to the pixels determines their output colors. Typically the relationship between voltage and color is not linear. The coordinate of the observed color $[R, G, B]^t$ is approximately

$$\begin{aligned} R &= R'^{\frac{1}{.45}} \\ G &= G'^{\frac{1}{.45}} \\ B &= B'^{\frac{1}{.45}} \end{aligned}$$

where R' is the voltage sent to the red phosphor. The exponent $\frac{1}{.45}$ is known as gamma.

In order to compensate for this, video cameras purposely correct for this behavior, and record $[R', G', B']^t$, “gamma corrected” values

$$\begin{aligned}R' &= R^{.45} \\G' &= G^{.45} \\B' &= B^{.45}\end{aligned}$$

When this color is fed to a monitor, the input color is produced.

Many computers have a lookup table to apply gamma correction before values are sent to the pixels. By setting this appropriately, one can directly send the desired $[R, G, B]^t$ coordinates to the frame buffer.

1.6.1 Non linear spaces and Perceptual distance

Two colors that are equidistant in the linear XYZ or RGB color space do not necessarily appear equally distant to human observers. In particular, we are very sensitive to small differences between dark colors, but less so between bright colors.

There exists a non linear transformation to a color space known as (L^*, a^*, b^*) . In this color space, coordinate distance is similar to perceptual distance. In this transformation, L^* is defined as

$$L^* = 116Y^{.33} - 16;$$

Notice that L^* is obtained by raising the linear color space coordinate Y to a fractional exponent value. This is somewhat similar to gamma correction. It turns out therefor that gamma corrected $R'G'B'$ is a pretty good color space when it comes to perceptual distance.

This becomes very important when quantizing each color channel to 8 bits. Eight bits gives me 256 different red color values. When the reds colors are broken up into 256 evenly spaced bins in R' coordinates. one cannot typically perceive differences between neighboring bins. But if one breaks up the colors in to 256 evenly spaced R coordinate bins, then one can distinguish between neighboring dark bins. As a result, $R'G'B'$ is a more efficient way of storing image data.

On the other hand for performing image manipulation, such as alpha compositing, one is dealing with computations that are derived with respect to the linear light and color spaces. And so for these kinds of image manipulation, RGB should be used.

1.6.2 Non linear spaces and image compression

For image compression we often want to use a color space that has one value representing black and white brightness (like the Y coordinate), and two values representing the color information. This is useful for a few reasons. First humans are more sensitive

to mistakes in the overall brightness than they are to mistakes in color. In fact, many image compression techniques immediately blur and downsample the color components by a factor of two in the vertical and horizontal direction. Secondly, most images have more variance in their brightness channel than in their color channels, and so it is better to allocate more bits to that channel.

Meanwhile, better perceptual bit usage is obtained in the gamma corrected domain than in the linear color space domain. As a result compression is usually done in a coordinate system called $Y'P'_BP'_R$ space. This space is both gamma corrected, and decomposed into brightness/color components.

To obtain these new coordinates, one starts with the gamma corrected $R'G'B'$ color coordinates. This is transformed into a “gamma corrected brightness” Y' coordinate along with the residual color information.

$$\begin{bmatrix} Y' \\ B' - Y' \\ R' - Y' \end{bmatrix} = \begin{bmatrix} .299 & .587 & .114 \\ -.299 & -.587 & .886 \\ .701 & -.587 & -.114 \end{bmatrix} \cdot \begin{bmatrix} R' \\ G' \\ B' \end{bmatrix}$$

Note that due to the nonlinearity of the gamma correction, $Y' \neq Y^{.45}$. In this transformation, if the second two coordinates are zero, then we are describing a greyscale color.

Next we scale the second two coordinates so they fit in the range $[-5\dots5]$.

$$\begin{bmatrix} Y' \\ P'_B \\ P'_R \end{bmatrix} = \begin{bmatrix} .299 & .587 & .114 \\ -.169 & -.331 & .5 \\ .5 & -.419 & -.081 \end{bmatrix} \cdot \begin{bmatrix} R' \\ G' \\ B' \end{bmatrix}$$

A variant color space, which we will not go into is $Y'C'_BC'_R$. There are also some older color spaces which were used in the same way called $Y'U'V'$ and $Y'I'Q'$. These spaces are getting phased out.

1.7 Perception in Computer Graphics

Human visual perception is a very rich topic, and one that is not yet fully understood. Understanding how visual perception works is very important for computer graphics. If we only have a finite amount of resources to use for the image generation and display process, we must know how to use these resources most usefully. For example if I only have 8 bits/channel of color values, how should I allocate these colors to display a scene that has a large distribution of very dark and very bright colors. A full exploration of this topic is beyond the scope of this book.

1.8 Colors and reflection

Perceived color is a 3d linear space. As such, we can represent images with three coordinates without losing any information. It is important to realize though that to properly model reflections and shading, one cannot use three coordinates without running the risk of losing information.

In particular suppose there are two lights $l_1(\lambda)$ and $l_2(\lambda)$ that are metamers, $\vec{c}(l_1) = \vec{c}(l_2)$, then of course they will look the same to a human observer. Suppose each of these lights is reflected off of the same material. Let us express the surface reflectivity per wavelength as $m(\lambda)$. In this case the reflected light r_1 and r_2 can be computed as

$$\begin{aligned}r_1(\lambda) &= l_1(\lambda)m(\lambda) \\ r_2(\lambda) &= l_2(\lambda)m(\lambda)\end{aligned}$$

There is generally no reason to believe that the colors of the reflected light will be the same, ie $\vec{c}(r_1) \neq \vec{c}(r_2)$. Thus the only way to predict the color of the reflected light is to keep around an explicit representation of $l_1(\lambda)$. Only after the reflected light distribution $r_1(\lambda)$ is computed can we determine the color of the light.

For high quality rendering people keep around a discrete representation of l_1 by storing its value at a large number of sample wavelengths. In practice though, for most applications people just use RGB values for all computation and do not care about the errors created.