Unit 8. Processing Color Images

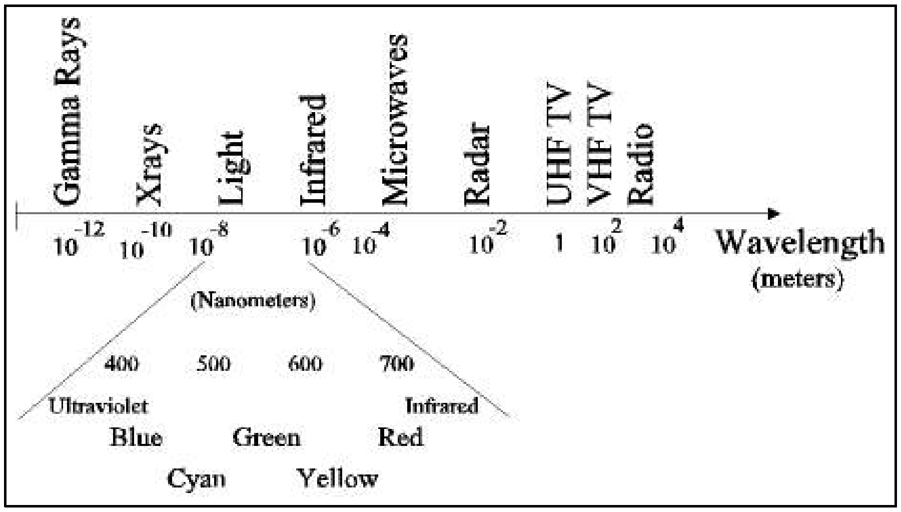
8.1 The Light Spectrum and Human Perception

Light as Electromagnetic Energy. Light can be considered as waveforms or rays of particles (photons) because of the extremely small wavelength *w* (as the wavelength becomes smaller, the wave behavior becomes more like that of a particle). This means that the frequency f is extremely high because

$$f = c/w, \quad w = c/f \tag{8.1}$$

where c is the speed of light. In vacuous space, c = $3x10^8$ m/sec. Thus f becomes (m/sec)/m = 1/sec. (the number of wavelengths per second). Figure 8.1 presents the electromagnetic spectrum and the visible light portion. A waveform can be represented as a function r(x,y,w,t) as it moves across an area in x and y through time t at a wavelength of w. We call r the radiant flux per area wavelength, or irradiance per wavelength.

Figure 8.1. The Electromagnetic Spectrum and Visible Spectrum



Perception of Light by the Human Eye. The wavelengths of light energy affect the receptors in the back of the human eye. There is not enough strength of this effect to pass the threshold for perception except for wavelengths of a limited range. However, from about 700 nm (red) to 400 nm (violet), there is perception, witht the most efficient detection being in the middle at about 546 nm (green). A nanometer (nm) is a billionth of a meter.

The cones (color sensors) in the center region of the back of the eye require higher energy for detection, while the rods (grayscale sensors) in the region outside of the cones are extremely sensitive to light (they give us peripheral vision and vision in faint light). Thus the human eye is most sensitive to small changes in grayscale and is less sensitive to small changes in color, which affects the way we need to process color images. Figure 8.2 shows the perceptive sensitivity of the human eye to wavelengths of electromagnetic energy.

R is lower in the visible frequency spectrum (longer wavelength), G is in the middle and B is higher. Below R is infrared, while above B is ultraviolet. Neither infrared nor ultraviolet can be detected by the human eye. The wavelengths in nanometers (a nm is 10^{-9} meter) are

Red: w = 700 nm   Green: w = 546 nm   Blue: w = 435.8 nm

Figure 8.2. Human perception of light energy.

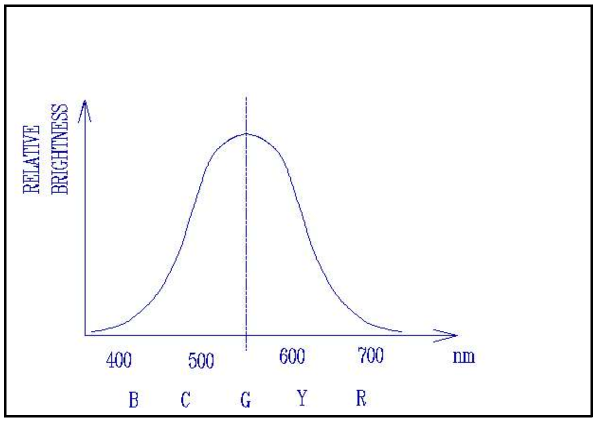
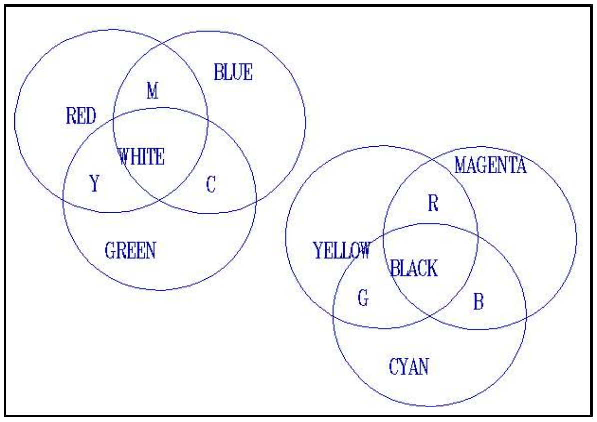


Figure 8.3. Additive and subtractive colors.



8.2 The RGB and CMY Color Models

The RGB Additive Color Model. The three primary (independent) colors red, green and blue (RGB) can be added in various combinations to compose any color in the spectrum. Figure 8.2 demonstrates the additive property. The additive relationships shown in Figure 8.3 are

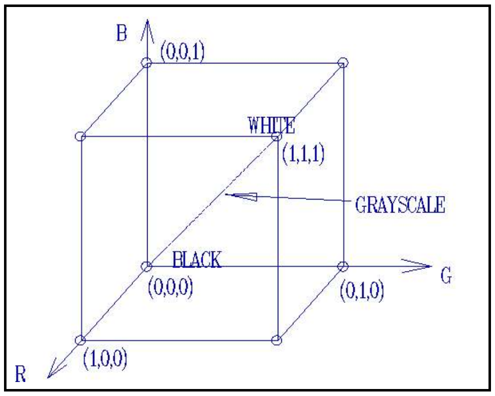
$$G + B = \text{Cyan (C)}, \quad B + R = \text{Magenta (M)}, \quad R + G = \text{Yellow (Y)}, \quad R + G + B = \text{White (W)}$$

In the RGB model, CMY are secondary colors. Figure 8.3 shows the RBG color cube. Each of R, G and B is standardized to take values between 0 and 1. The origin (0,0,0) is Black because there is no energy (0 intensity), while the point (1,1,1) is pure white because it has equal parts of R, G and B at maximum value. We can normalize the colors via

$$r = R/(R+G+B), \;\; g = G/(R+G+B), \;\; b = B/(R+G+B) \tag{8.2}$$

$$r + g + b = 1 \tag{8.3}$$

Figure 8.4. The RGB color cube.



The CYM Subtractive Color Model. We can also take CYM (Cyan, Yellow and Magenta) as the primary colors, in which case RGB become the secondary colors. Molecules in a material of a subtractive color absorbs energy at the frequency of the subtractive colors and thus subtracts that color. Figure 8.3 also shows the subtractive model. Paints of colors C, Y and M can be mixed to form

$$C + M = \text{Blue}, \quad C + Y = \text{Green}$$

$$M + Y = \text{Red}, \quad C + M + Y = \text{Black}$$

Conversion Between RGB and CMY. Given the RGB or CMY coordinates of a pixel color, it can be respectively converted into CMY or RGB via

$$\begin{vmatrix} C \\ M \\ Y \end{vmatrix} = \begin{vmatrix} 1 \\ 1 \\ 1 \end{vmatrix} - \begin{vmatrix} R \\ G \\ B \end{vmatrix} \qquad (8.4)$$

$$\begin{vmatrix} R \\ G \\ B \end{vmatrix} = \begin{vmatrix} 1 \\ 1 \\ 1 \end{vmatrix} - \begin{vmatrix} C \\ M \\ Y \end{vmatrix} \qquad (8.5)$$

CMY is useful in special situations. For example, some color printers use the subtractive model and have built-in logic for converting RGB to CMY.

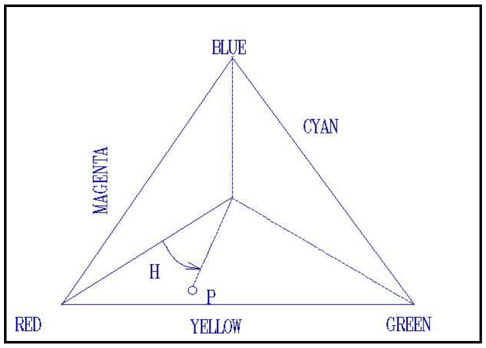
8.3 HSI and RGB-HSI Transformations

The HSI Model. The *hue*, *saturation* and *intensity* model of color takes advantage of the way humans perceive color. *Hue* (H) is the color perceived due to the wavelength. *Saturation* (S) is the degree to which the color is pure, or free, from white light (grayscale) dilution, or pollution. S is therefore the color saturation. High saturation means that the color is highly pure, while low saturation means that there is a lot of dilution with white light that is made by the presence of the R, G and B primary components. H and S taken together provide the *chromaticity*, or color content of a pixel.

Intensity is the brightness, or energy level of the light and is devoid of any color content. I is independent of the hue and saturation. Together, HSI (hue, saturation and intensity) make any color and intensity of light. Figure 8.5 displays the chromaticity triangle. Any point P on the triangle, at a fixed intensity slice I, represents a color.

As an example of saturation, let a pixel have the color (R, G, B) = (200, 100, 50). The amount of gray, or white light pollution is found by taking the minimum value 50 and subtracting to obtain the pure color (R, G, B) = (150, 50, 0), which is a reddish yellow and is an additive hue on top of the white light (pollution) (R, G, B) = (50, 50, 50), which is a gray.

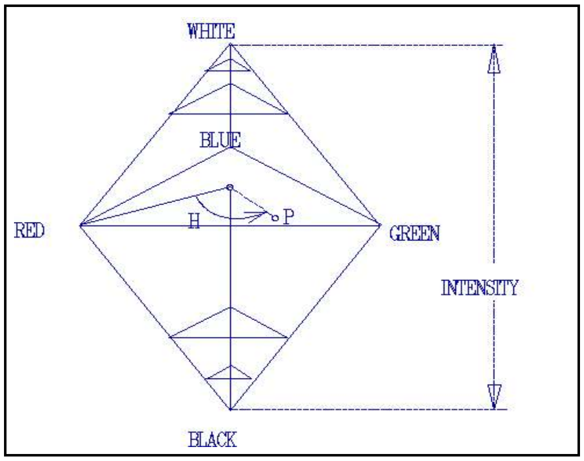
Figure 8.5. The chromaticity triangle.



The boundary of the chromaticity triangle represents pure color with no pollution by white light. Thus there is only a single wavelength of light present along the boundary (a color that can be made by two primary colors). Points closer to the middle have proportionately more pollution by white light (the minimum amount of a third primary color increases) so that at the middle point, there is nothing but white light (equal amounts of R, G and B). The proportion of the distance from the center point W to P of the distance from the center point to the boundary through P is the saturation.

The hue H of a color at point P in Figure 8.5 is the angle that the line from W to P makes with the line from W to P_R (at R). Starting at P_R and traversing the boundary in the counterclockwise direction, we go to Yellow (60°), then Green (120°), Cyan (180°), Blue (240°), Magenta (300°) and then back to Red (360°, which we take as 0° again).

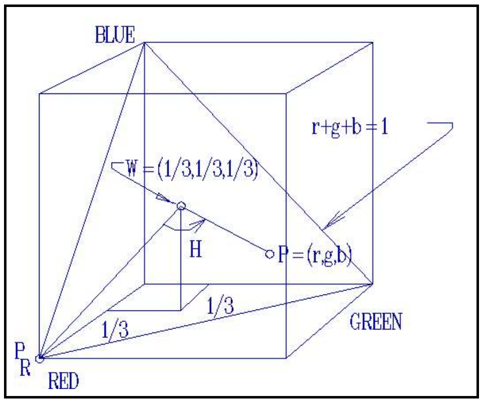
Figure 8.6. The HSI color solid.



***A Main Principle of Processing Color Images*.** To sharpen, smooth, edge enhance or otherwise process an RGB image as we have done so far for grayscale images (anything other than modifying the colors), the secret is to convert the RGB image to HSI first and then process the intensity image I. Then we convert the resulting intensity image, along with the H and S, back to RGB. The human eye is sensitive to minor changes in the intensity but not so much with color. If we process the color values in the usual manner of image processing, the result is unreal and the disproportionate colors can be eerie and unpleasing to the eye.

The chromaticity triangle of Figure 8.5 does not model the intensity, which is a fixed value for each triangle. We can see from Figure 8.6 that the HSI *color solid* provides the intensity, which is independent of H and S, as well as the chromaticity. For any intensity level, a cross-section perpendicular to the line from (0, 0, 0) to (1, 1, 1) is a chromaticity triangle. Figure 8.7 shows that the major slice of the color cube gives the color triangle. The color solid is obtained by rotating the color cube so that the corner point (1, 1, 1) is at the top center.

Figure 8.7. A slice of the color cube.



***Conversion between HSI and RGB*.** We desire to convert the colors of the image pixels from RGB to HSI is so that we can process the intensity part (I) to obtain the new intensity part ($I^{\sim}$), then convert the H, S and $I^{\sim}$ back to R, G and B for display. That way we do not change the colors disproportionately but can sharpen, smooth, or otherwise process the intensity of the image. For any color pixel (R, G, B) we desire to convert it to HSI directly.

RGB → Intensity. Let (R, G, B) be a color pixel. Each of R, G and B is standardized as in Figure 8.7 to take values between 0 and 1, with a maximum total intensity of 1. Thus we obtain a standardized I ($0 \le I \le 1$) via

$$I = (R + G + B) / [3(255)] \tag{8.6}$$

RGB → Saturation. White light is caused by equal amounts of each of R, G and B. If we take the minimum one of these and subtract that amount from each of R, G and B, then we eliminate the grayscale (intensity) and what is left has only two color components instead of three (the minimum one is now zero). Thus this new color has no white light pollution and so it is fully saturated and represents a single wavelength (see Figure 8.2) or a combination of only two primary colors. It is the saturation. Because I has a maximum value of 1 here, we can take

$$S = 1 - \min\{r, g, b\} = 1 - [\min\{R,G,B\}/I] = [I - \min\{R,G,B\}] / I \tag{8.7}$$

RGB → Hue. Any color pixel (R, G, B) is standardized according to r = R/(R+G+B), g = G/(R+G+B) and b = B/(R+G+B) so that r + g + b = 1. The HSI chromaticity triangle, for a given intensity, corresponds to a plane through the RGB color cube perpendicular to the line from black to white. Figure 8.7 shows the *major triangle* of all points (r,g,b) that satisfy the linear equation r + g + b = 1. The center point **W** has coordinates (1/3,1/3,1/3). We can find the angle H for any point P = (a, b, c) on the triangle by the geometry of the triangle. Let **P** be the point as shown in Figure 8.5 and **W** be the center. Hue is an angle of 0° to 360°.

<u>Case 1</u>. We consider first the situation where **P** is in the lower half of the HSI chromaticity triangle where $b \leq g$, which is below the line from **R** through **W** to **C** (Cyan). Upon projecting the vector **P** - **W** onto the vector (line) **R** - **W** via the *cosine formula* for the dot product (see any trigonometry or calculus book), we have (where $\|X - Y\|$ is the Euclidean distance between points X and Y).

$$(\mathbf{P} - \mathbf{W})\circ(\mathbf{P}_R - \mathbf{W}) = \|\mathbf{P} - \mathbf{W}\|\ \|\mathbf{R} - \mathbf{W}\|\cos(H) \tag{8.8a}$$

$$\cos(H) = (\mathbf{P} - \mathbf{W})\circ(\mathbf{P}_R - \mathbf{W}) / \|\mathbf{P} - \mathbf{W}\|\ \|\mathbf{R} - \mathbf{W}\| \tag{8.8b}$$

We first obtain the Euclidean distance between **P** and **W**.

$$\|\mathbf{P} - \mathbf{W}\| = \|(r, g, b) - (1/3, 1/3, 1/3)\| = \{(r - 1/3)^2 + (g - 1/3)^2 + (b - 1/3)^2\}^{1/2} = \tag{8.9}$$

$$\{[R/(R+G+B) - 1/3]^2 + [G/(R+G+B) - 1/3]^2 + [B/(R+G+B) - 1/3]^2\}^{1/2} =$$

$$\{[3R - R - G - B]^2/9 + [3G - R - G - B]^2/9 + [3B - R - G - B]^2/[3(R+G+B)]^2\}^{1/2} =$$

$$(1/3)[1/(R+G+B)]\{[2R - G - B]^2/9 + [2G - R - B]^2/9 + [2B - R - G]^2/9\}^{1/2} =$$

$$(1/3)[1/(R+G+B)]\{(4R^2 - 4RG - 4RB + G^2 + 2GB + B^2) +$$

$$(4G^2 - 4RG - 4GB + R^2 + 2RB + B^2) + (4B^2 - 4GB - 4RB + G^2 + 2RG + R^2)\}^{½} =$$

$$(1/3)[1/(R+G+B)]\{ 6R^2 + 6G^2 + 6B^2 - 6RG - 6RB - 6GB\}^{1/2} =$$

$$\{6[R^2 + G^2 + B^2 - RG - RB - GB]\}^{1/2} / [3(R+G+B)] =$$

$$\{6[R^2 + G^2 + B^2 - 3RG - 3RB - 3GB + 2RG + 2RB + 2GB]\}^{1/2} / [3(R+G+B)] =$$

$$\{6[(R^2 + G^2 + B^2 + 2RG + 2RB + 2GB) - 3RG - 3RB - 3GB)]\}^{1/2} / [3(R+G+B)] =$$

$$\{6[(R + G + B)^2 - 3(RG + RB + GB)]\}^{1/2} / [3(R+G+B)] =$$

Next, we obtain the distance between $\mathbf{P}_R$ and **W**.

$$\|\mathbf{P}_R - \mathbf{W}\| = \|(1, 0, 0) - (1/3, 1/3, 1/3)\| = \|(2/3, -1/3, -1/3)\| = (2/3)^{1/2} \tag{8.10}$$

To take the dot product $(\mathbf{P} - \mathbf{W})\circ(\mathbf{P}_R - \mathbf{W})$, we review its definition, which is

$$(a, b, c)\circ(x, y, z) = ax + by + cz \tag{8.11a}$$

Thus

$$(\mathbf{P} - \mathbf{W})\circ(\mathbf{P}_R - \mathbf{W}) = (r-1/3, g-1/3, b-1/3)\circ(2/3, -1/3, -1/3) = \tag{8.11b}$$

$$(R/[R+G+B] - 1/3, G/[R+G+B] - 1/3, B/[R+G+B] - 1/3)\circ(2/3, -1/3, -1/3) =$$

$$[R+G+B]^{-1}((3R - R - G - B)/3, (3G - R - G - B)/3, (3B - R - G - B)/3)\circ(2/3, -1/3, -1/3) =$$

$$[9(R+G+B)]^{-1}(2R-G-B, 2G-R-B, 2B-R-G)\circ(2, -1, -1) =$$

$$[9(R+G+B)]^{-1}[(4R-2G-2B) - (2G-R-B) - (2B-R-G)] =$$

$$[9(R+G+B)]^{-1}[2R - G - B] = \{2R - G - B\} / \{9(R+G+B)\}$$

From Equation (8.8b) we solve for H (with $b \leq g$) via

$$H = \arccos\left\{[(\mathbf{P} - \mathbf{W})\circ(\mathbf{P}_R - \mathbf{W})] \,/\, [\|\mathbf{P} - \mathbf{W}\|\,\|\mathbf{P}_R - \mathbf{W}\|]\right\} \tag{8.12}$$

Thus H is the angle whose cosine is:

$$\frac{[2R - G - B] / [9(R+G+B)]}{\{[6(R + G + B)^2 - 3(RG + RB + GB)]\}^{1/2} / [3(R+G+B)]} = \tag{8.13a}$$

$$\frac{[2R - G - B] / 3}{\{[6(R + G + B)^2 - 3(RG + RB + GB)]\}^{1/2}} \tag{8.13a}$$

Case 2. When the color (r, g, b) is in the upper half of the chromaticity triangle, we must subtract the angle H from from 360° to obtain the hue.

$$H^{\sim} = 360 - H, \quad B > G \tag{8.13b}$$

Conversion from HSI to RGB. To convert a pixel (H, S, I) in the opposite direction to (R, G, B), we consider the cases where the point **P** is in one of the following regions of the chromaticity triangle: *H < 120°* (red-green), *120° ≤ H < 240°* (green-blue), or else *240° ≤ H < 360°* (blue-red). The following describe the inverse conversions.

H < 120° :
$b = (1/3)(1 - S)$
$r = (1/3)\{1+ S\cos(H)/\cos(60°$
$°-H)\}$
$g = 1 - (r + b)$ (8.14a,b,c)

120° ≤ H < 240° :
$r = (1/3)(1 - S)$
$g = (1/3)\{1 + S\cos(H)/\cos(60°-H)\}$
$b = 1 - (r + g)$ (8.15a,b,c)

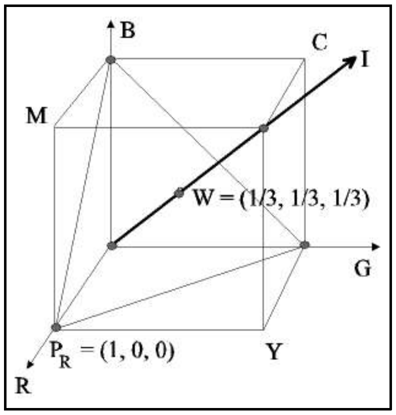
240° ≤ H < 360° :
$g = (1/3)(1 - S)$
$b = (1/3)\{1 + S\cos(H)/\cos(60°-H)\}$
$r = 1 - (g + b)$ (8.16a,b,c)

We discuss the first case here (the others are similar). The saturation S is a proportion between 0 (all white dilution) and 1 (all color). Here H < 120°, which means that blue is the minimum value and so determines the amount of white light. What is left after subtracting the minimum is the saturation, which in this case leaves red and green that make a pure color and thus has a saturation of 1. The amount of white light dilution then is 1 - S, where S is the color saturation value for the pixel. This white light dilution (pollution) has equal parts of r, g and b, so we can take

$$b = (1/3)[1 - S] \tag{8.17}$$

Given the value for r, which is shown above in Equation (8.14a) and which we derive carefully below, the last value g can be found from the intensity because the colors satisfy r + g + b = l. This can be solved for g from knowledge of r and b: g = 1 - r - b = 1 - (r + b). We now show the complete derivation for $r = (1/3)\{I + S\cos(H)/\cos(60-H)$, as in Equation (8.14b).

Figure 8.8 The color cube revisited **Figure 8.9. The projection triangle.**



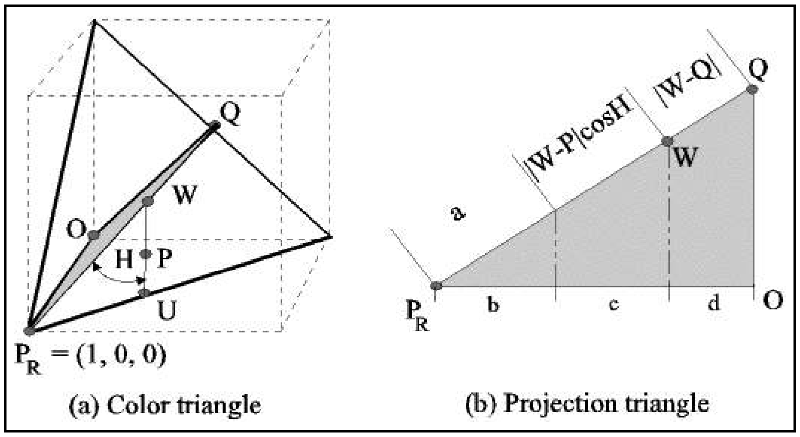


Figure 8.8 displays the color triangle again with the points of interest without a color point. Figure 8.9(a) shows the color point P at an angle of H (the hue), the origin O and the point U where the line through W and P meets the triangle boundary. Here we form a triangle that we call the *projection triangle* inside of the color cube, which is formed from the lines $||P_R\text{-}Q||$, $||W\text{-}O||$ and $||O\text{-}P_R||$. Because of the similarity of the triangles, we have that

$$a/b = ||P_R-Q|| / ||P_R-O|| = ||P_R-Q|| / 1 = ||P_R-Q|| \tag{8.18}$$

Solving for b, we obtain

$$b = a / ||P_R-Q|| \tag{8.19}$$

To get an expression for a, we use the hypotenuse length in Figure 8.9(b) as follows.

$$||P_R-Q|| = a + ||W-P||\cos H + ||W-Q|| \tag{8.20}$$

so that

$$a = ||P_R-Q|| - ||W-P||\cos H - ||W-Q|| \tag{8.21}$$

Upon substitution of this a into Equation (8.19) we see that

$$b = a / ||P_R-Q|| = (||P_R-Q|| - ||W-P||\cos H - ||W-Q||) / ||P_R-Q|| = \tag{8.22}$$

$$1 - (||W-P||\cos H / ||P_R-Q||) - (||W-Q|| / ||P_R-Q||) \tag{8.23}$$

The line segment from P_R to O has length 1 and coincides with the (red) R-axis. The value $1 - b = c + d$ is the red coordinate *r* of the point P (projected onto the line through P_R and Q and then projected onto the R axis). Thus

$$r = 1 - b = (||W-P||\cos H / ||P_R-Q||) + (||W-Q|| / ||P_R-Q||) \tag{8.24}$$

A key ingredient that permits us to solve for r in terms of H and S here is that (see Figure 8.10)

$$(||W-Q|| / ||P_R-Q||) = 1/3 \tag{8.25}$$

so that

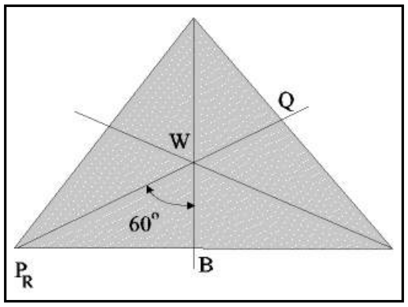
$$r = 1 - b = (\|W-P\|\cos H / \|P_R-Q\|) + 1/3 \quad (8.26)$$

From Figure 8.9 (a) we see that the saturation is the ratio

$$S = \|W-P\| / \|W-U\|, \quad \|W-P\| = S(\|W-U\|) \quad (8.27)$$

Observing from the equilateral color triangle of Figure 8.10 that the angle between the line P_R to Q and the line W to B is 60^o, we can solve for the unknown $\|W-P\|$ by

Figure 8.10. The 60^o angles.



$$\|W-B\| = \|W-U\|\cos(60^o - H) \quad (8.28)$$

Thus we get

$$\|W-U\| = \|W-B\| / \cos(60^o - H) \quad (8.29)$$

One more observation from the triangle of Figure 8.10 assures us that

$$\|W-B\| = \|W-Q\| \quad (8.30)$$

Upon substituting these last parts into Equation (8.26) above, we obtain the red coordinate

$$r = 1 - b = (\|W-P\|\cos H / \|P_R-Q\|) + 1/3 = \quad (8.31$$

$$S(\|W-U\|)\cos H / \|P_R-Q\| + 1/3 = \quad \text{[Eqn. (8.27)]}$$

$$(S\cos H)(\|W-U\|) / (\|P_R-Q\|) + 1/3 =$$

$$(S\cos H)(\|W-P\| / \cos(60^o - H) / (\|P_R-Q\|) + 1/3 = \quad \text{[Eqn. (8.29)]}$$

$$(S\cos H)(\|W-P\| / (\|P_R-Q\|) / \cos(60^o - H) + 1/3 = \quad \text{[Eqn. (8.30)]}$$

$$(S\cos H)(1/3) / \cos(60^o - H) + 1/3 = \quad \text{[Eqn. (8.32)]}$$

$$(1/3)\{1 + [S\cos H) / \cos(60^o - H)]\} \quad \text{(q.e.d.)}$$

8.4 Processing Color Images

Smoothing Hue and Saturation. Because the human has higher sensitivity to the intensity component and relatively less sensitivity to color, we convert to HSI and perform different types of processing on I and the color components. The I component image $f_I(m,n)$ can be processed as any other grayscale image, and so we can sharpen it, equalize its histogram, stretch the contrast, remove noise from it via standard grayscale methods, or process it by other grayscale methods.

The color is not disturbed by the intensity processing. A common type of processing for the color components is to pass the hue and saturation through lowpass filters to remove the color noise. This is done effectively with a median filter. Slight blurring of hue and saturation is not noticeable when the processed image is displayed.

Generally, we have the 3 images $f_R(m,n)$, $f_G(m,n)$ and $f_B(m,n)$ of the R, G and B values, respectively. These are converted into $f_H(m,n)$, $f_S(m,n)$ and $f_I(m,n)$ (or $f_Y(m,n)$, $f_I(m,n)$ and $f_Q(m,n)$: see the section below).

Then the appropriate component images are processed and converted back to the RGB component images for display.

Color Balance. Color may be out of balance due to digitizing during capture or scanning, or due to the photoprocessing of photographic film. The imbalance is caused by the shifting of color from what it should be. The most important test is to examine image objects that are known to be gray to see if they have color, and if so, then the color is out of balance. A second test is to examine objects that have highly saturated colors to see if the colors are incorrect.

The usual situation is that the balance can be restored by linear grayscale transformations on two of the R, G, and B colors, as only two of these need to be changed to match the third. By taking the average values for each of R, G and B over an object of dark gray, the average values μ_R, μ_G, and μ_B should be equal. The same holds true for the average values of the R, G and B values over a region of light gray. Linear transformations can be designed for two of the colors to change the average values so that they are all equal both on a dark gray object and on a light gray object. The transformation for adjusting the means is given in Section 2.7 of Chapter 2.

Color Processing. The most common type of color processing of images is the grayscale processing of the intensity values when the image is converted from RGB to HSI, and then converting back to RGB with the new processed $I^{\sim}$ values. This type of processing can also be done with the YIQ model, described below, where Y is the intensity (I and Q provide the color). However, the colors may be changed to enhance the image, and indeed, they sometimes need to be changed. Some of this can be done using RGB, while other types of processing require HSI or YIQ.

The colors can be made stronger (purer) by increasing the saturation, S, by multiplying it by a number greater than 1. Multiplying by a number less than 1 causes the color to become weaker. Only those pixels with significant saturation should be changed because the increased saturation of colors that are too close to the point **W** can cause color imbalance.

Colors that contain larger proportions of R than of G and B are called *warm*, while those that contain more B than of R or G are called *cool*. Hues can be shifted by adding or subtracting ΔH to the H values to make the colors "warmer" or "cooler." The increment should be small so as to not yield an undesirable appearance.

8.5 The YIQ Color Model

The YIQ Model for Commercial TV Broadcasting. The Electronic Industries Association (EIA) defined the RS-170 specifications for monochrome television, using techniques that were defined in the 1930s. The image was painted on a cathode ray tube (CRT) with a raster scan of lines to compose a picture of 525 lines per frame, interlaced so that the even numbered lines were written 30 times per second and similarly for the odd numbered lines, so that there was the illusion of 60 frames per second (60 half-frames, or *fields*) to avoid flicker. The *aspect ratio* was 4-to-3, with 646 square pixels per line. A few of the lines (11) were used for synchronization and were not actual lines of pixels. The signal was analog with a horizontal synchronizing pulse at the end of each line, and a vertical synchronizing pulse at the end of 525 lines, which indicates a return to the beginning of the frame. The 30 fields were written in 33.3 milliseconds (that is, a field every 1/30th of a second). The RS-170 signal ranges from -0.286 to 0.714 volts for a total difference of 1 volt, but the actual voltages represent the brightness (intensity) from black to white and varies from 0.143 to 0.714. Synchronizing pulses go down to -0.286 volts.

When commercial television broadcasting systems changed to color, there were still millions of monochrome TV sets in use. The color TV signal that was adopted retained the monochrome signal Y that would work with these monochrome receiver sets, but contained components I and Q for the color that could be used by the color TV receivers. Thus the *YIQ* model was initiated. The color TV system in the United States is known as the NTSC for the *National Television Systems Committee*, was accepted in 1953 and was adopted as the EIA RS-170A specification.

The brightness is the Y signal from the RS-170, but a color subcarrier waveform carries the chrominance. The hue is modulated in the waveform as a phase, while the saturation is carried as an amplitude. A color reference signal (the *color burst*) is added at the beginning of each video line. The composite signal is an analog waveform in a 4.2 MHz bandwidth centered on the channel frequency. The color subcarrier is at 3.58 MHz and is quadrature modulated onto the carrier wave. The bandwidth of the I component is 1.3 MHz and the Q bandwith is 0.6 MHz.

Y contains the luminance, or intensity, while I and Q contain the color information that is decoupled from luminance. Thus the Y component can be processed by grayscale methods without affecting the color content. For example, Y can be processed with histogram equalization. The situation is the same as processing with HSI.

Recall that the human eye contains sensors in the retina (inside surface of the back of the eyeball) that are of two types: i) color cones that receive the light ray more directly and require stronger signals (daytime vision); ii) and grayscale rods that are very sensitive and are used mostly in peripheral and nighttime vision. Because the eye is more sensitive to changes in luminance (intensity of light) than in colors, the Y signal contains a wider bandwidth (in Hz) than the I and Q for greater resolution in Y. Thus we can smooth I and Q to remove color noise and process the Y image $f_Y(m.n)$ as we would any grayscale image.

Gamma Correction. The brightness (intensity) of the pixels is modeled as a constant times a power of the voltage plus a constant that is the *blankout* (black) level at which the screen goes black. The model is

$$\text{brightness} = cv^g + b \qquad (8.32)$$

where c is the gain, v is the voltage of the signal and b is a level at which the color is still black in the CRT. *Gamma* is the exponential g. Doubling of a grayscale pixel value, for example, does not double the brightness (18% higher is seen as 50% brighter). The response curve for a video monitor is set by the gamma value g.

The g of a CCD (charged-coupled device) is *1.0*, while that of a videocon type picture tube is *1.7*. CRT tubes, including TV sets and video monitors, vary from *2.2* to *2.5* (usually *2.2*). To avoid the necessity of having gamma correcting inside each TV set, the NTSC includes the gamma correction to be made before the signal is broadcast. A gamma correction of *1/2.2 = 0.45* is made in the signal and then it is broadcast.

Conversion between RGB and YIQ. The conversion from RGB to YIQ (similar to a digital color model called YCbCr, but YIQ is in the analog NTSC specifications) is done by the simple linear transformation

$$\begin{vmatrix} Y \\ I \\ Q \end{vmatrix} = \begin{vmatrix} 0.299 & 0.587 & 0.114 \\ 0.596 & -0.275 & -0.321 \\ 0.212 & -0.523 & 0.311 \end{vmatrix} \begin{vmatrix} R \\ G \\ B \end{vmatrix} \qquad (8.33$$

The conversion from the gamma corrected Y'I'Q' to R'G'B' is done via

$$\begin{vmatrix} R' \\ G' \\ B' \end{vmatrix} = \begin{vmatrix} 1.0 & 0.956 & 0.621 \\ 1.0 & -0.272 & -0.649 \\ 1.0 & -1.106 & 1.703 \end{vmatrix} \begin{vmatrix} Y' \\ I' \\ Q' \end{vmatrix} \qquad (8.34)$$

To convert from YIQ to RGB without using gamma correction requires the inverse of the transformation matrix in Equation (8.17). This is necessary for conversion to YIQ for image processing and then conversion back to YIQ for display. However, an image in Y'I'Q' format originates from television broadcast or video tape and so already has the gamma correction built in. Many people use Equations (8.33) and (8.34) to convert from RGB to YIQ, process the Y, I and Q, and then convert back to RGB for display.

High Density Television. The standards for *high definition television* (HDTV), also called *high density television*, were considered in the early 1980s and have been evolving since then. When they are used in commercial TV they will have 1125 lines per frame and an aspect ratio of 16-to-9, so there will be 2000 pixels across a line. They will be interlaced using two fields of half as many lines (even and odd lines).

8.6 Pseudo Color

Intensity Slicing. Many photographs, such as those taken from the air, from microscopes, x-rays, or satellites, are grayscale (intensity). There is often a desire to add colors according to the gray level to enhance the information available to the human viewer of the image.

The simplest color recoding scheme is the use of a single threshold gray level T. Suppose that we choose (r_T,g_T,b_T) as the color to be used for grays above the threshold, and (r_L,g_L,b_L) as the color to be used for grays below the threshold. A simple algorithm is show below.

Algorithm 8.1: Thresholded Pseudo-Color

```
for m = 1 to M do
   for n = 1 to N do
    if (f(m,n) < T) then
       $g_R(m,n) = r_L$;
       $g_G(m,n) = g_L$;
       $g_B(m,n) = b_L$;
    else
       $g_R(m,n) = r_T$;
       $g_G(m,n) = g_T$;
       $g_B(m,n) = b_T$;
```

In general, we use r thresholds $T_1,...,T_r$ and use a different color combination for each range 0 to T_1, T_1 to T_2, ... , T_{r-1} to T_r. The idea is to choose warmer colors for lighter grays, or some other scheme that is consistent with the goal.

General Gray to Color Transformations. In the general case, we have three functions that each map the graylevel image f(m,n) into respective colors of R, G and B. These transformations may be nonlinear and continuous, although they operate on discrete data. Figure 8.11 shows the scheme. Figure 8.12 shows three such transformations that map the graylevels into R, G and B, respectively. The transformations shown in Figure 8.12 may also use Fourier transforms for filtering, or convolution mask processing.

Figure 8.11. Pseudo-color transformations.

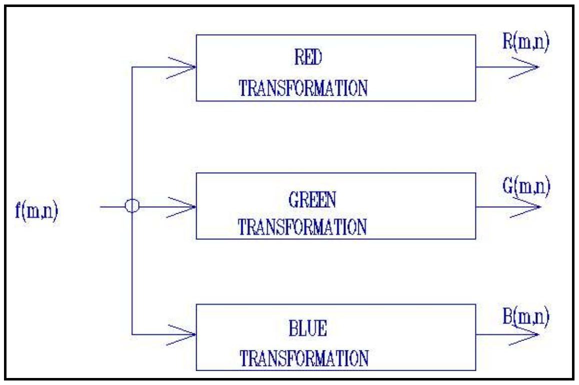
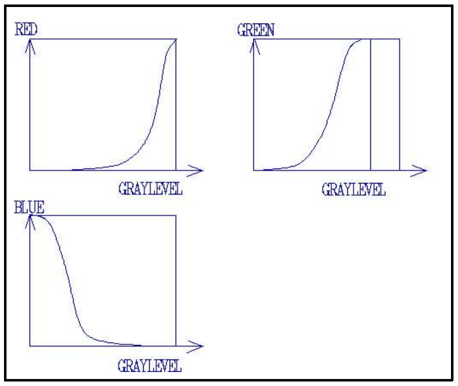


Figure 8.12 Example pseudo-color maps.



8.7 Color Experiments with *Matlab*

Sharpen Color Image. First we must point out that *Matlab* will not display TIFF color images that have the older (1987a) formatting, but will display color if the formatting is the newer type (1989). We use the RGB image *sanfran.tif* that we have converted to the newer TIF format using a newer graphics program from *Micrografix*.

To sharpen an image, we must first convert it to HSI, which is often called *HSV* as is it by *Matlab*. After that we sharpen the *V* (I) image then we can apply a sharpening filter to it. Alternatively we can convert it to YIQ, called *YCbCr* by Matlab and other sources. The code that we type at the command line is given below.

```
>> RGB = imread('san_fran.tif');          %read in color image
>> imshow(RGB);                           %show original image RGB
>> YIQ = rgb2ntsc(RGB);                   %convert image to YIQ format
>> Y = YIQ(:,:,1);                        %get luminance (intensity) image as Y
>>  I = YIQ(:,:,2);                       %get first color component as image I
>> Q = YIQ(:,:,3);                        %get second color component as image Q
```

Now we could smooth I and Q lightly to remove noise or we could despeckle with a median filter. In either case, we don't want to change the color. We know how to do this by running a smoothing filter with the function *imfilter()*. An example is provided here. Figures 8.13 through 8.18 show the results.

```
>> hsmooth = [ 1 1 1; 1 2 1; 1 1 1] / 10;      %define a smoothing mask
>> hsharp = [-1 -1 -1; -1 17 -1; -1 -1 -1] /18;%define unsharp masking filter for sharpening
>> Y2 = imfilter(Y,hsharp);                    %sharpen image Y with hsharp and put result in Y2
>> I2 = imfilter(I, hsmooth);                  %filter image Y with mask hsmooth and put in I2
>> Q2 = imfilter(Q,hsmooth);                   %filter image Q with mask hsmooth and put in Q2
>> figure, imshow(Y2);                         %display grayscale image Y2
>> figure, imshow(I2);                         %display grayscale color component 1
>> figure, imshow(Q2);                         %dispaly grayscale color component 2
>> YIQ2(:,:,1) = Y2;                           %build 3-byte image: first component is Y2 image
>> YIQ2(:,:,2) = I2;                           %second component is I2 image
>> YIQ2(:,:,3) = Q2;                           %third component is Q2 image
>> RGB2 = ntsc2RGB(YIQ2);                      %convert YIQ2 to a new RGB image, RGB2
>> figure, imshow(RGB2);                       %show new RGB image, RGB2
```

Figure 8.13. Original old San Francisco.

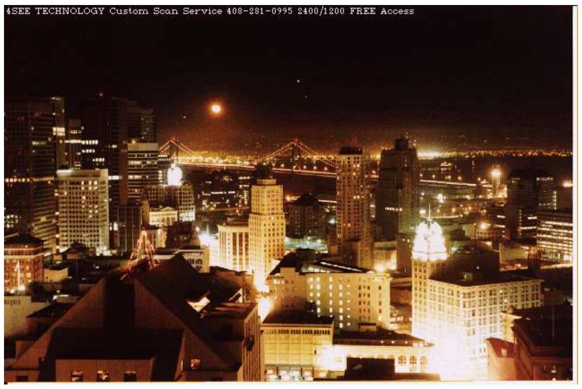

4SEE TECHNOLOGY Custom Scan Service 408-281-0995 2400/1200 FREE Access

Figure 8.14. Sharpened Y component.

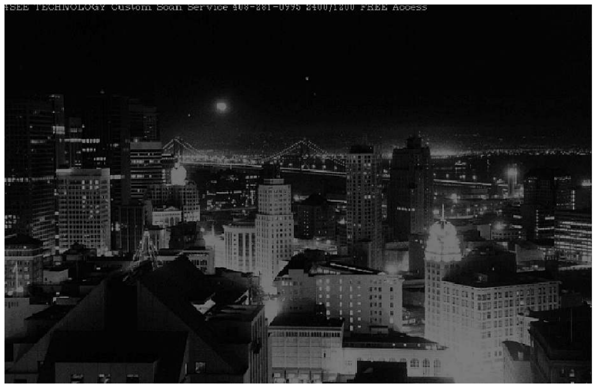

4SEE TECHNOLOGY Custom Scan Service 408-281-0995 2400/1200 FREE Access

Figure 8.15. Smoothed I component.

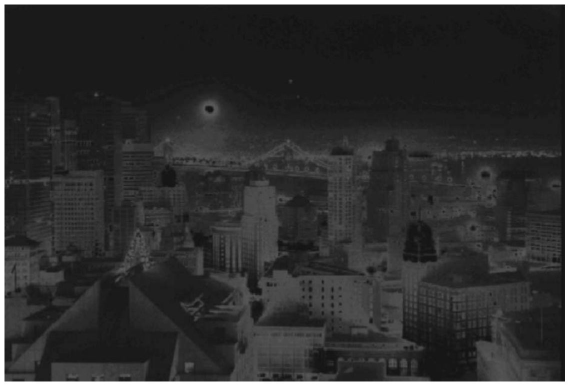


Figure 8.16. Smoothed Q.

Figure 8.17. Combined YIQ to RGB image.

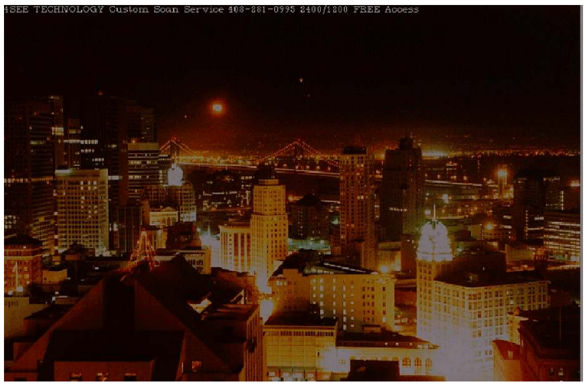
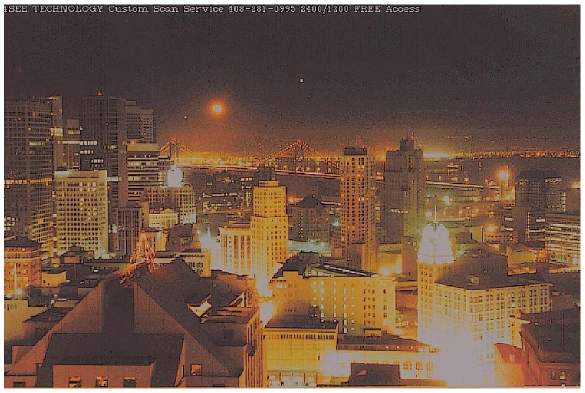

SEE TECHNOLOGY Custom Scan Service 408-281-0995 2400/1200 FREE Access

Figure 8.18. Adjusted Y, gama = 0.5, combined.


SEE TECHNOLOGY Custom Scan Service 408-281-0995 2400/1200 FREE Access

We can see that the conversion from YIQ to RGB in Figure 8.17 did not set the gamma factor correctly. A consequent is that the brightness is not sufficiently high. Figure 8.18 shows the result of twice sharpening the Y image, adjusting the result with *gamma = 0.5*, combining them and then convertnig to RGB for display.The Y part should each be scaled using the *imadjust()* function of *Matlab*. For example, the following uses the gamma adjustment parameter of 0.5 (g < 1.0 brightens and g > 1.0 darkens).

```
>> I3 = imadjust(I2, [], [], 0.5);            %g = 0.5 is the gamma adjustment
```

However, when we scale I and Q the colors shift terribly and the image becomes a purplish-reddish-pink.

Similar experiments can be done with RGB to HSI, processing and then converting back. An example is provided below. Figures 8.19 through 8.22 display the resulting images.

```
>> RGB = imread('san_fran.tif');              %read in RGB image
>> imshow(RGB);                               %show the RGB image on screen
>> HSV = rgb2hsv(RGB);                        %convert RGB image to HSV image
>> H = HSV(:,:,1);                            %strip out the H (hue) pixel components
>> S = HSV(:,:,2);                            %strip out the S (saturation) pixel components
>> V = HSV(:,:,3);                            %strip out the I (or V, the intensity value components
>> hsmooth = [ 1 1 1; 1 2 1; 1 1 1] / 10;     %construct a smoothing mask
>> hsharp = [-1 -1 -1; -1 17 -1; -1 -1 -1] / 18;%construct a sharpening mask (unsharp masking)
>> H2 = imfilter(H,hsmooth);                  %filter H image
>> S2 = imfilter(S,hsmooth);           %filter S image
>> V2 = imfilter(V,hsharp);            %filter I image
>> figure, imshow(H2);                 %show sharpened H image H2
>> figure, imshow(S2);                 %show smoothed S image S2
>> figure, imshow(V2);                 %show smoothed I image I2
>> V2 = imadjust(V2, [], [], 0.6);     %adjust contrast automatically, put gamma = 0.6
>> HSV2(:,:,1) = H2;                   %build 3-byte image: first component is Y2 image
>> HSV2(:,:,2) = S2;                   %second component is I2 image
>> HSV2(:,:,3) = V2;                   %third component is Q2 image
>> RGB2 = HSV2RGB(HSV2);               %convert YIQ2 to a new RGB image, RGB2
>> figure, imshow(RGB2);               %show new RGB image, RGB2
```

Figure 8.19. Smoothed H component.

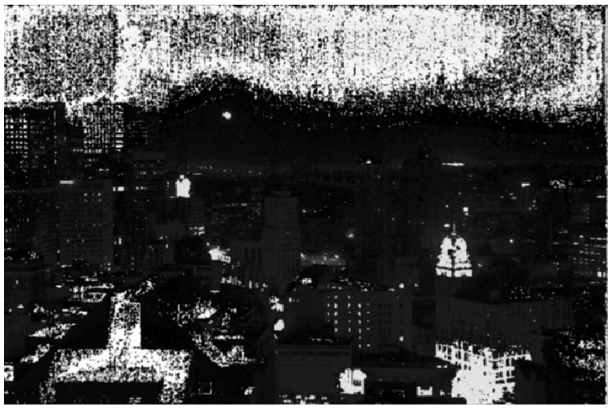


Figure 8.20. Smoothed S component.

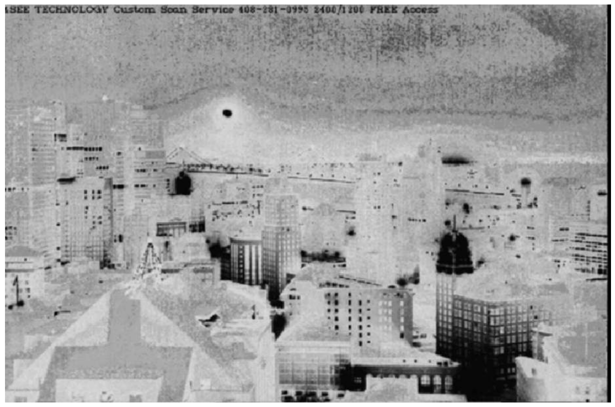

ISEE TECHNOLOGY Custom Scan Service 408-281-0995 2400/1200 FREE Access

Figure 8.21. I sharpened, V-gamma = 0.6.

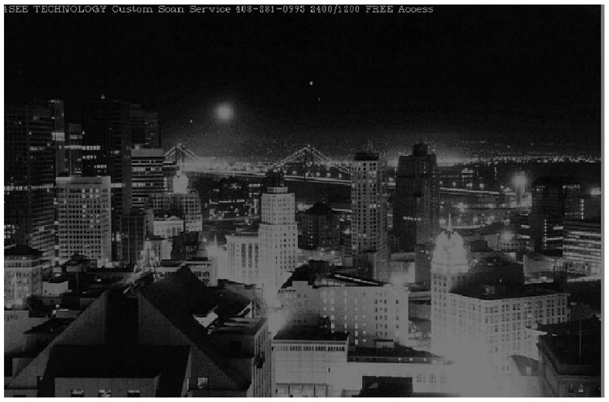
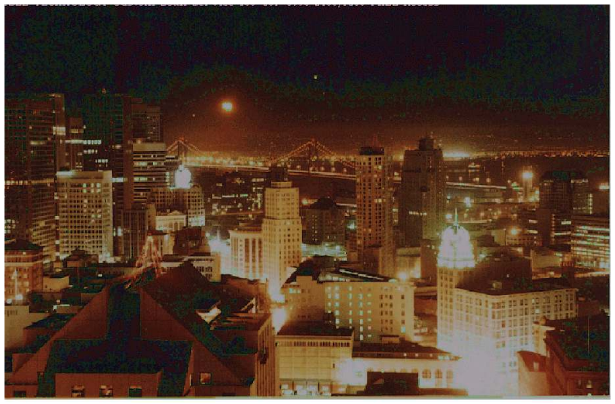

ISEE TECHNOLOGY Custom Scan Service 408-281-0995 2400/1200 FREE Access

Figure 8.22. Combined HSV, HSV to RGB.



8.8 Computer Experiments with XV

Color Images and Control Windows. XV automatically displays a color image as color when it is one of the color formats that it knows (GIF, TIF, JPEG, BMP, TARGA, and raw data PPM files are included). We use the *sf.gif* file from the same source as the other files (*lena256.tif* and *shuttle.tif*). Copy *sf.gif* to the personal images directory as before. Then type

/images$ **startx<ENTER>**

Next, run XV with the filename *san_fran.tif*.

/images$ **xv sf.gif<ENTER>**

The image displayed is a color photograph of part of old San Francisco with a view toward the Bay Bridge to Oakland/Berkeley. It is 640x422 and is in the 8-bit color mode (256 colors), of which 231 unique colors are present. Click the right mouse button with the pointer inside of the image area to obtain the *xv controls* window. It will display the information given above in the bottom half of this window.

Depending on the size of one's color screen, one may want to reduce the size of the displayed image by left-clicking on the *Image Size* bar at the top left of the *xv controls* window, holding the left mouse button down and dragging the pointer down to the line *10% Smaller* and releasing. This can be repeated until the right size is obtained (and use *10% Larger* if necessary to undo a resizing or make larger if desired).

Move the *xv controls* window and the image to the top of the screen beside each other. Put the pointer on the *Windows* bar type button in the *xv controls* window, hold the left mouse button and move the pointer down to the line *Color Editor* and release. The lrge *xv color editor* window comes up at the bottom of the screen. It is a good idea to move it up so that it overlaps slightly the image and *xv controls* windows. On the upper left of the *xv color editor* window is a 16x16 block of color squares that is a color map used in this image.

Recall that the 8-bit color mode uses 256 values (0,...,255) that are numbers (addresses) of the color registers. Three n-bit values are stored in each color register (n = 6 for SVGA graphics interfaces) to represent the R, G and B values. For example, 000000 111111 111111 would hold R = 0, G = 63 and B = 63, which is Cyan.

At the bottom of the *xv color editor* window is the *intensity* block. On the right are the R, G and B level squares. All of these squares contain the graphs of the form y = x with nodes for changing these identity transformations as desired (and a *RESET* button for resetting them back to the identity maps).

Experiments with HSI. We assume here that the user is in XV with the color image *sf.gif* and the *xv controls* window side by side at the top of the screen and the *xv color editor* just below them. To the top left of the *xv color editor* window is the color map. Just below that is a section with three "clock" counters with captions *Red*, *Green* and *Blue*. Above the *Blue* counter is the *RGB/HSV* bar button. HSV is the HSI color model that we are familiar with, where I is called V for *value*.

We left-click the mouse on the *RBG/HSV* bar and the *Red*, *Green* and *Blue* counters immediately become the *Hue*, *Saturation* and *Value* counters. Click again and then return to the original setting. Observe that *Red* is the highest, *Green* is the next, and *Blue* is the lowest. This image is composed of warm colors.

In the middle vertical strip of the *xv color editor* window, center position, is a counter designated as *Saturation*. By clicking on the arrow buttons (or on the clock itself), we can move the counter hand counter clockwise or clockwise to reduce or increase the saturation, respectively. Move it counter clockwise to the -60° position (or even the -100% position). Now examine the displayed image. It is devoid of color content because it is thoroughly diluted with white light (intensity). Now move counter clockwise to -40%. A slight tinge of color is detectable. At -36% the slight color content is more obvious and at -30% it appears in the lights on the bridge and in the lights on top of the dark building on the left in the foreground. At -20% to -

10% the image appears quite natural, but a 0% it is beginning to appear vivid. At 10% the image is strongly colored to the extent that the green lights appear unnatural. Continuing in this manner, the color grows more vivid at 20% and completely unreal at 30%.

RGB Experiments. Click on the *RESET* button at the bottom left of the window to return to the original image. Now we experiment with the intensity I = (R + G + B)/3. At the bottom center of the *xv color editor* window is the *Intensity* window. Left-click and drag the nodes on the graph to change the intensity levels. Click on the *RESET* button in the *Intensity* block to obtain the original image back. We can make the contrast greater by moving the lower node downward and the upper node upward.

After resetting to get the original image back, let us change the *Red* transformation at the top right of the *xv color editor*. To strenghten the R and make the colors warmer, we use the *GAM* (*gamma*) button in the *Red* box (just below the reset button). We then click on *GAM* and type in 1.7 from the keyboard and hit <ENTER>. The graph changes to be concave and the image colors become warmer immediately. Similarly, change the *GAM* to 1.4 for *Green* and to 1.2 for *Blue*. This increases the intensity level overall because each of R, G and B has been increased. We account for this approximately by changing the gamma for *Intensity* to 0.72.

With the current color setting, we sharpen the image intensity by left-clicking on *Algorithms* in the upper left corner of the *xv controls* window, hold the mouse button down and drag the pointer down to the line *Sharpen* and release. Left-click on the *OK* button (with enhancement factor of 75). This process takes a moment, but then the sharpened image appears. To save this image, click on the *Save* bar of the *xv controls* window, then click and drag to select *PS* (postscript). Be sure that *Color* is selected and then click on *OK*. The sizing of the image to be saved as *sf.ps* is requested. We click on *OK* after that and the image is saved in Postscript color file format for printing to a color printer.

Figure 8.23 shows the original image *sf.gif*, while Figure 8.24 presents the image that was processed by the gamma adjustments and sharpening of the intensity, which we saved as *sf.ps*.

Figure 8.23. San Francisco (w/Bay Bridge).

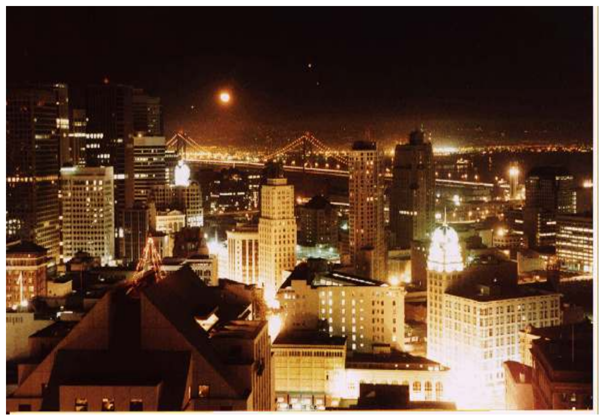
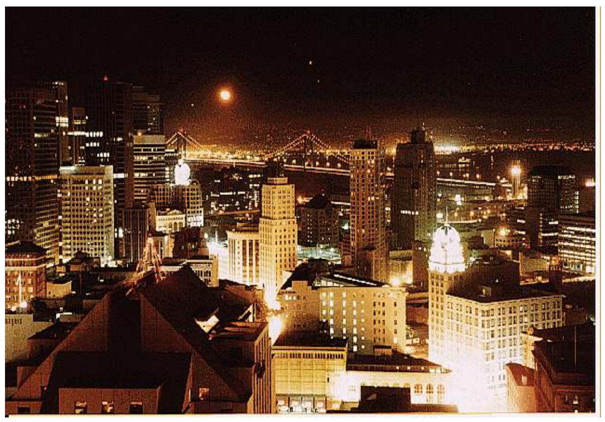


Fig. 8.24. Processed San Fran. (w/Bay Bridge).



8.9 Color Experiments with *LView Pro*

Here we bring up *LView Pro* on a PC, select *File* at the upper left of the menu bar and then select the image file *san_fran.tif*. From the menu bar we now select *Color*, which brings up a pop-up menu with items from which we select *Filters*. When the *Pre-defined Image Filters* pop-up menu comes up, then we select *sharpen* or *sharpen* (we select the latter here). We click the Apply button and the image is sharpened, after which we select *File* and *Save As* and save in the desired format. Figure 8.25 shows the *LView Pro* window and the Color pop-up menu that appear on the screen and Figure 8.26 displays the result.

Figure 8.25. The selection of Filters in the Lview Pro window.

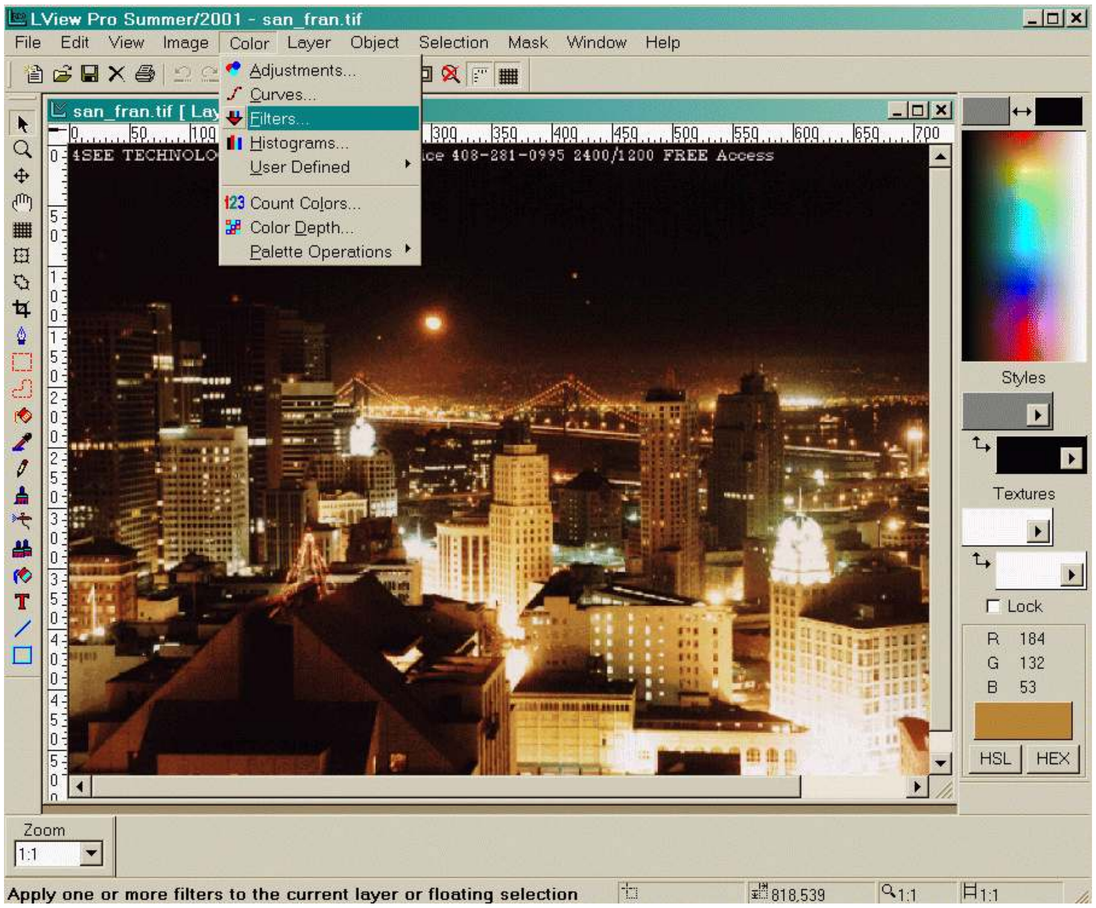
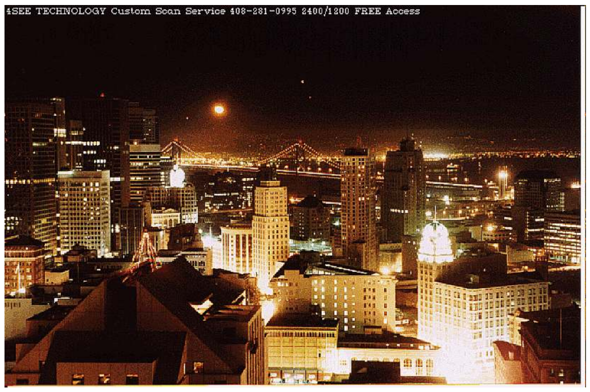


Figure 8.26. The sharpened result.


4SEE TECHNOLOGY Custom Scan Service 408-281-0995 2400/1200 FREE Access

8.10 Exercises

8.1 An golden orange color is to be used to color a graphic drawing of a moon. Using two primary colors and combining them linearly, design a color (use two of R, G and B in the proper proportions). Draw a circle on a paper, color it yellow with a felt-tipped pen and then scan it as a TIF, PNG or JPEG file. Convert to HSV and change the hue to be a golden orange.

8.2 In Exercise 8.1 above, lower the saturation of color to add white light to the moon color. Give a proportional combination of the colors so that the white light dilution is about 25%. Note that *saturation* can be used in XV to add the white light to get the desirable effect. With Matlab the S image can be lowered.

8.3 Derive Equation (8.11b).

8.4 Derive Equation (8.12).

8.5 Derive Equation (8.32).

8.6 Convert H = 30°, S = 0.80, I = 0.70 to (r,g,b) and then to (R,G,B), where each of r, g and b is between 0 and 1.

8.7 Convert the image *san_fran.tif* (RGB) to HSV, separate the components into H, S and V, make H warmer and sharpen V. Then (after scaling, if necessary), put it back together as an HSV, convert to RGB and display. Show the before and after images.

8.8 Repeat Exercise 8.7 above but use YIQ in place of RGB. In this case, increase Y by 10% and then sharpen. Convert back to RGB and display the before and after images.

8.9 For pseudo-color, some people put the darkest grays at black and grays near white at white, but add color to shades in between, with lighter shades of gray being converted to more reddish (warmer) shades of color of greater intensity and more white light dilution near the upper end (white). The darker shades are converted to more bluish colors of lesser intensity near black. What are the advantages and disadvantages of this approach?

8.10. Design a pseudo color transformation for each of R, G and B that fits with the strategy of Exercise 8.9 above. Make the transformations as continuous as possible (see Figures 8.11 and 8.12).

8.11 Use XV to process the image *san_fran.tif* so that the colors are warmer and slightly stronger. You may use a combination of processing with parts of HSI and RGB.

8.12 Use Lview Pro to process san_fran.tif in the manner described in Exercise 8.11 above.

8.13 Which, if any, of R, G, B; H, S, I; and Y, I, Q components can be processed in the frequency domain (or its equivalent filter in the spatial domain) to some useful purpose?