Vanderbilt University
School of Engineering

EECE 253 IMAGE PROCESSING

LABORATORY ASSIGNMENT 3

Jack Minardi
9 September, 2011

Abstract

This paper reports the results of experiments done to explore color correction of images using Matlab. Color correction can be done in many ways, and some of those methods are explored in the report. First, color correction by color band manipulation is explored in experiment 1. It can be seen that this can have a fairly significant effect on the overall color of an image. Second, color manipulation in HSV space (hue, saturation, and value) was explored. This allows you fine grain control on more abtract color attributes such as hue, saturation and value. And finally in the last experiment color correction through color matching was explored. This allows you to set target colors for certain colors, and then to apply this transformation to the image as a whole. This is great if you know the actual values of some of the pixels in the image.

Table of Contents

Introduction

Color correction is one of the most common tasks done with digital images. It involves manipulating the overall color of an image in some manner. This can be accomplished in many ways, but three of these are explored in the following experiments. The first experiment explores modifying the individual color bands. This can increase or decrease the overall amount of red, green, or blue depending on which band you change. The second experiment explores manipulation in HSV space. The allows you to change things such as the hue, saturation, and value. The last experiment shows how to do color correction through color matching. This is where you match pixel values to other values, and then use a transformation matrix to modify the whole image.

Description of Experiments

There were four experiments performed in this lab. Matlab was used for all the computation. All the scripts that were written can be found in the appendix and in the zip file this report came in. The results were somewhat subjective and discussions can be found in the report below.

Experiment 1.

This problem explores the changes in color that occur when the individual bands of a true-color image are transformed independently of each other. Find a truecolor image that contains a wide range of colors.

(a) Use the gamma correction function you wrote for Lab Assignment 2 to increase and decrease the overall brightness of each band separately. Display the original image and the 6 altered versions. Describe the differences between the images. (b) Find the mean intensity of each band. Threshold each band. That is let J = uint8(zeros(size(I))); J(:,:,1) = uint8(255*(I(:,:,1) >mr)); J(:,:,2) = uint8(255*(I(:,:,2)>mg)); J(:,:,3) = uint8(255*(I(:,:,3)>mb)); where I is the original image and mr, mg, and mb are the mean red, green, and blue intensities. 1 Display J. How many colors do you see? What are they? Explain why this particular set of colors is present. What are the colors that could possibly be present in an image created this way?

Experiment 2. This problem asks you to write color conversion programs and to perform color manipulation in the HSV color space.

Write a program, RGBtoHSV to convert an image from RGB to HSV format. The syntax should be: [H,S,V] = RGBtoHSV(R,G,B); where R, G, and B are the red, green, blue bands from an image, I. H, S, and V are of class double. Write the program so that H ranges from 0 to 2π with 0 = red, 2π 43 = green, 3π = blue, and 2π = red. S ranges from 0 to 1 with 1 = full saturation. V ranges from 0 to 255 with non-integer values likely. After operating on the H, S, and V bands as described below, you will need to convert the results back into R, G, and B bands of type uint8. For this you will need to write another program, HSVtoRGB, [R,G,B] = HSVtoRGB(H,S,V);. H and S, must be of class double and lie within the ranges described above. V may be either class double or uint8. In either case, the values of V must lie in the range [0, 255]. R, G, and B should be returned as type uint8. For conversion of RGB to HSV, use the definition given in EECE253_05_ColorCorrection.ppt, pp. 78-82. For conversion of HSV to RGB, use the definition given in EECE253_05_ColorCorrection.ppt, pp. 83-91. Those are the exact definitions; use them in your program. There are a number of similar, faster somewhat different approximations listed in the literature, e.g. , the RGBtoHSI algorithm listed on p. 299 of Gonzalez and Woods and pp. 211-214 of Gon- zalez, Woods, and Eddins. But don't use those. Start with the definitions in the presentation and write your own algorithms to implement them. Comment your code liberally and include it in the appendix of your report. Use your programs in problem 2 of the lab assignment. The three bands must be recombined into a single image. If R, G, and B were origi- nally extracted from a three-band color image I, then the following Matlab code will recombine the bands: J = reshape([R G B],size(I));

(a) Use the same image you used in problem 1. Apply your gamma function to the V band, both to increase it and to decrease it. Reconstruct two new images using the altered V bands via your HSVtoRGB program. Describe the results.

(b) Linearly increase and linearly decrease the saturation. Reconstruct two new im- ages using *the altered S bands via your program HSVtoRGB. Describe the results. Note that a linear increase may increase the value of the saturation of some pixels to > 1. What can you do to correct for that?*

(c) Apply a constant circular shift to the H band to shift the colors in the green direction from hue = 0. Note that a circular shift of amount δ away from red toward green could cause the hues of some pixels to exceed 2π. These must be "wrapped around". That is, if H(r, c) > 2π then replace H(r, c) with H(r, c)−2π, Repeat the same experiment but shift the colors in the blue direction. Note that you must perform an analogous wrapping to deal with the pixels whose hues are shifted to below zero in value. Reconstruct 2 new images using the altered H bands via your program HSVtoRGB. Describe the results. How do these results differ from those of problem 1?

Experiment 3. Correction of discolored photos through the linear transformation of color vectors.

(a) Download the images, lance_armstrong_orig.jpg and lance_armstrong_alt.jpg, from the lab assignment 3 folder on Blackboard. The first image is the original and the second is a color-manipulated version of the original. Select 15 (or more) different colors from the original image. That is, record the rgb values from at least 15 different pixel locations in the image. Select a wide range of colors and intensities. Ideally, you would select dark, medium, and bright pixels for each of the 6 primary colors (red, yellow, green, cyan, blue, magenta) and grayscale for a total of 21 colors from 21 pixel locations. But, of course, most images (including this one) do not contain all colors. I suggest that you use the Matlab function, ginput, to select locations from the image. Select the same pixels from the altered image. Make a table that lists the pixel locations and the corresponding rgb triplets from both images.

(b) Construct a linear transformation matrix from the selected colors1 List the trans- form matrix.

(c) Transform the altered image using the transformation matrix. Display the original and the result, plot their pdfs, and comment on the differences between all three images.

Results of Experiments

Experiment 1

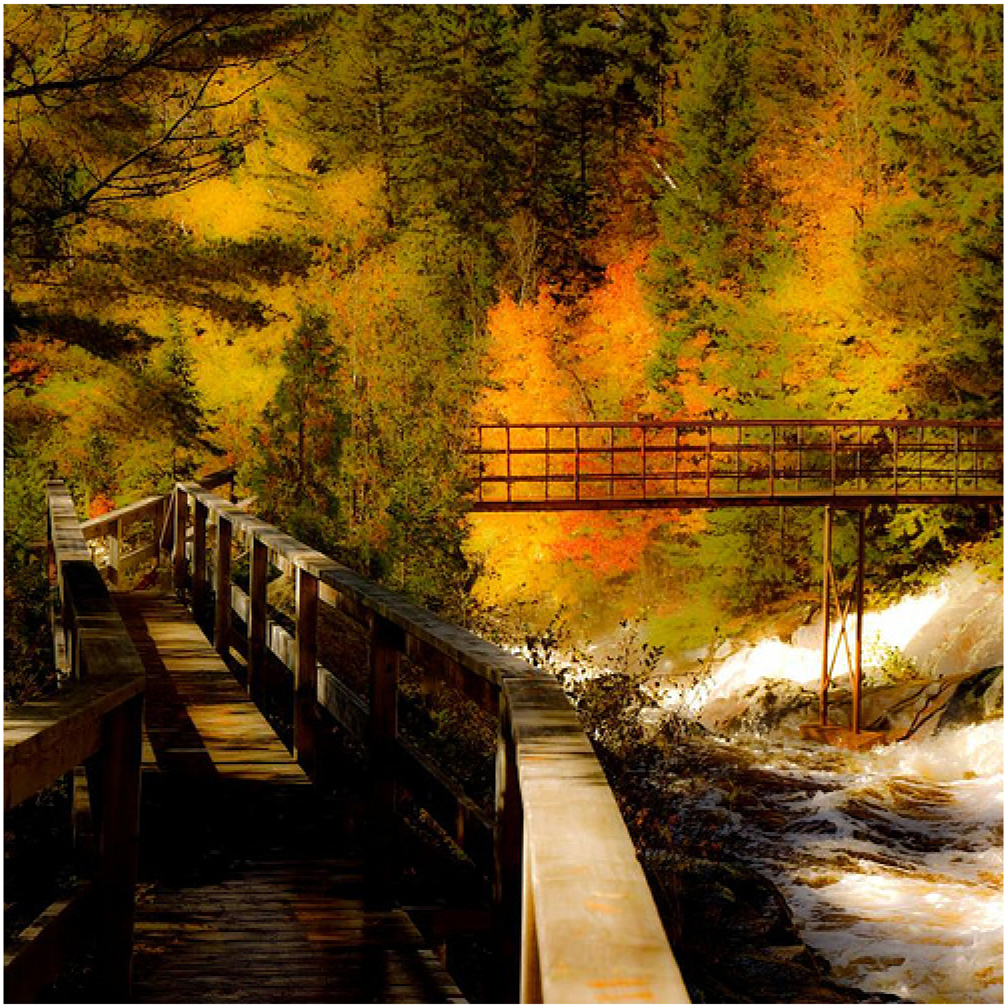

Figure 1. Image used in following experiments. Source: http://wildernessnaturalist.com/nature-photography/nice-nature-photography-photos-4/

a)

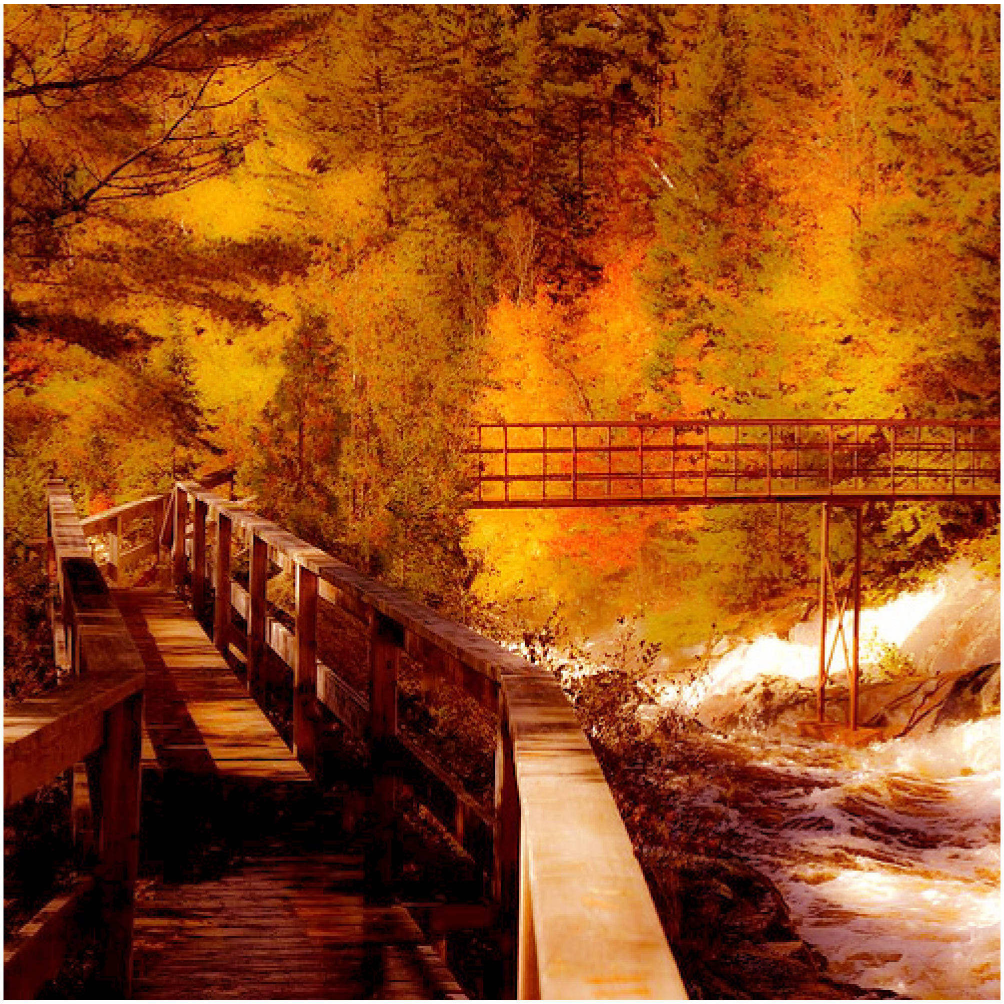

Figure 2. Red band brightened

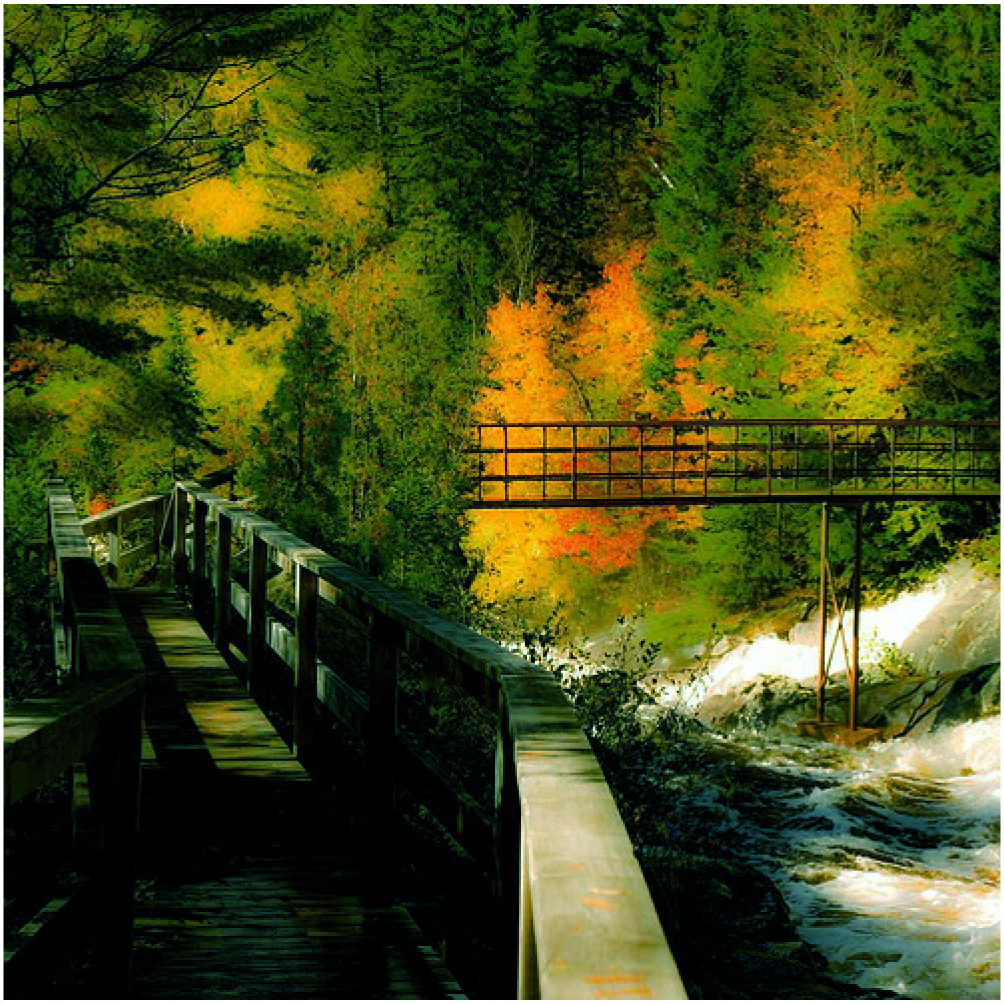

Figure 3. Red band darkened

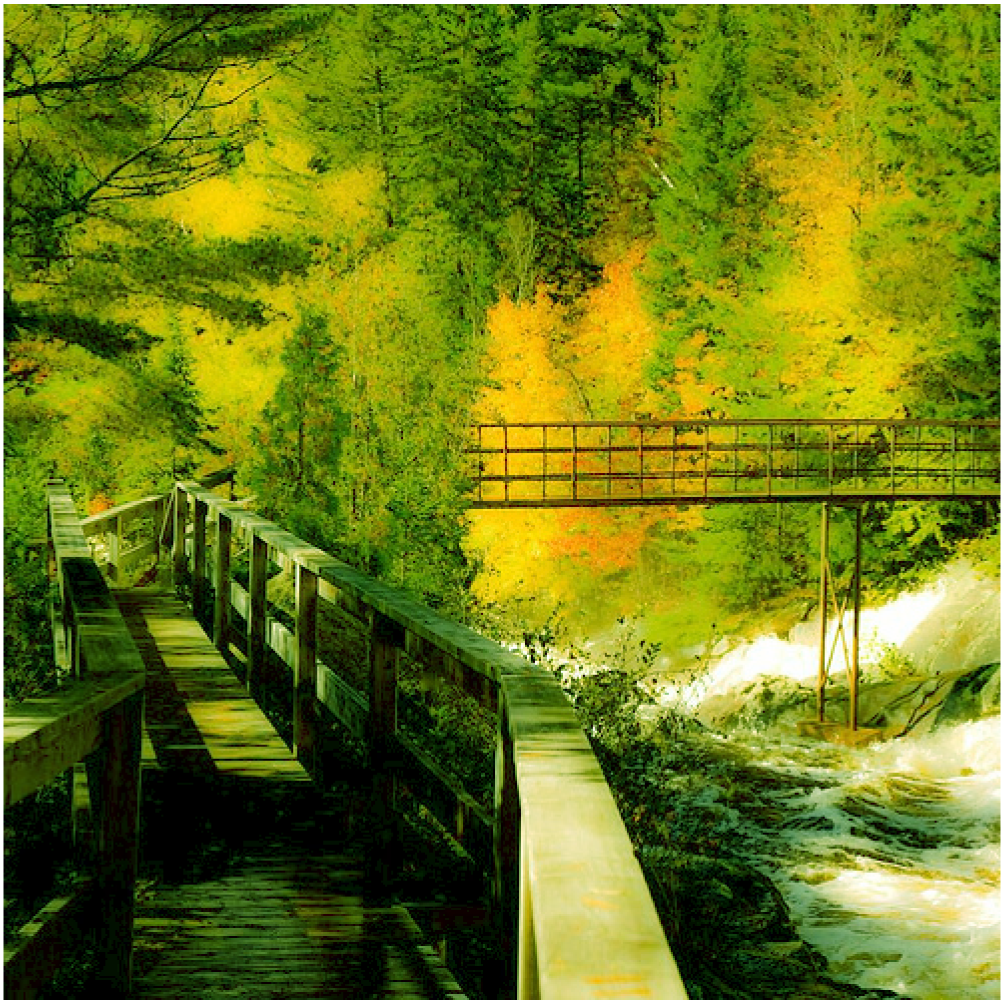

Figure 4. Green band brightened

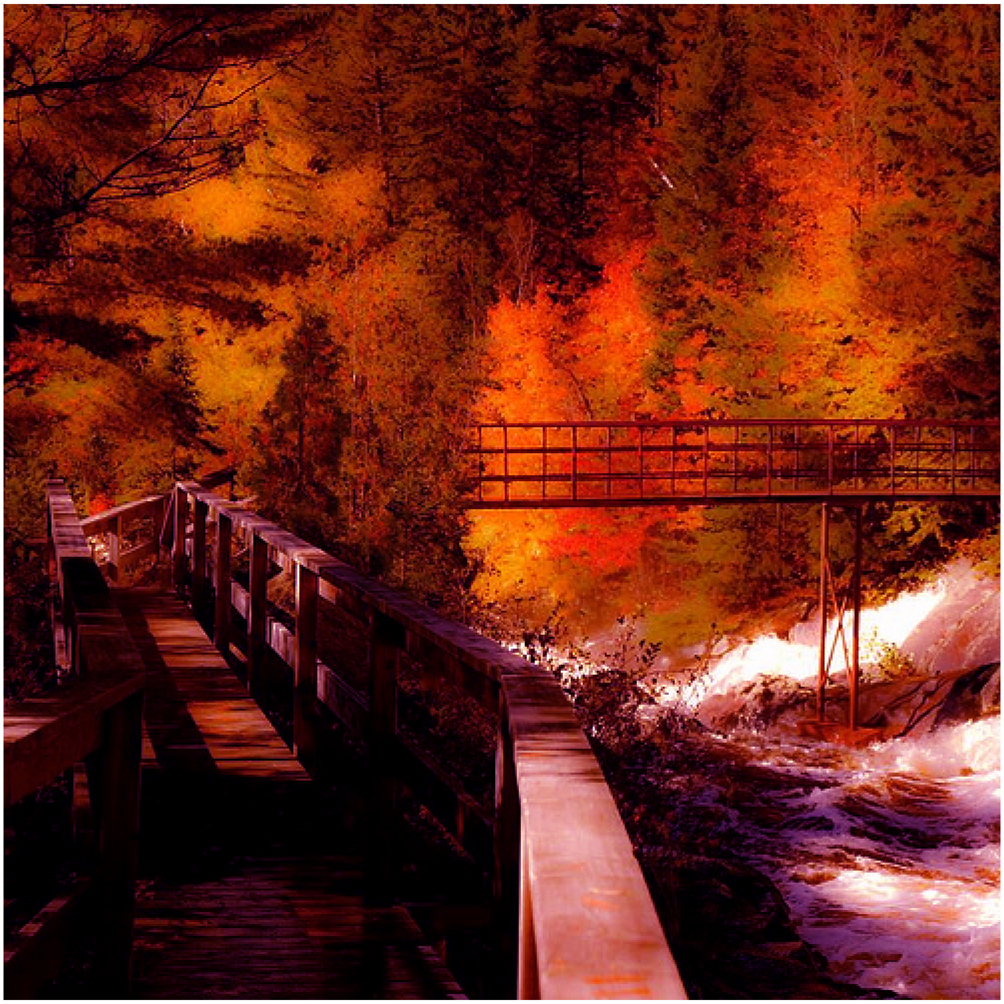

Figure 5. Green band darkened

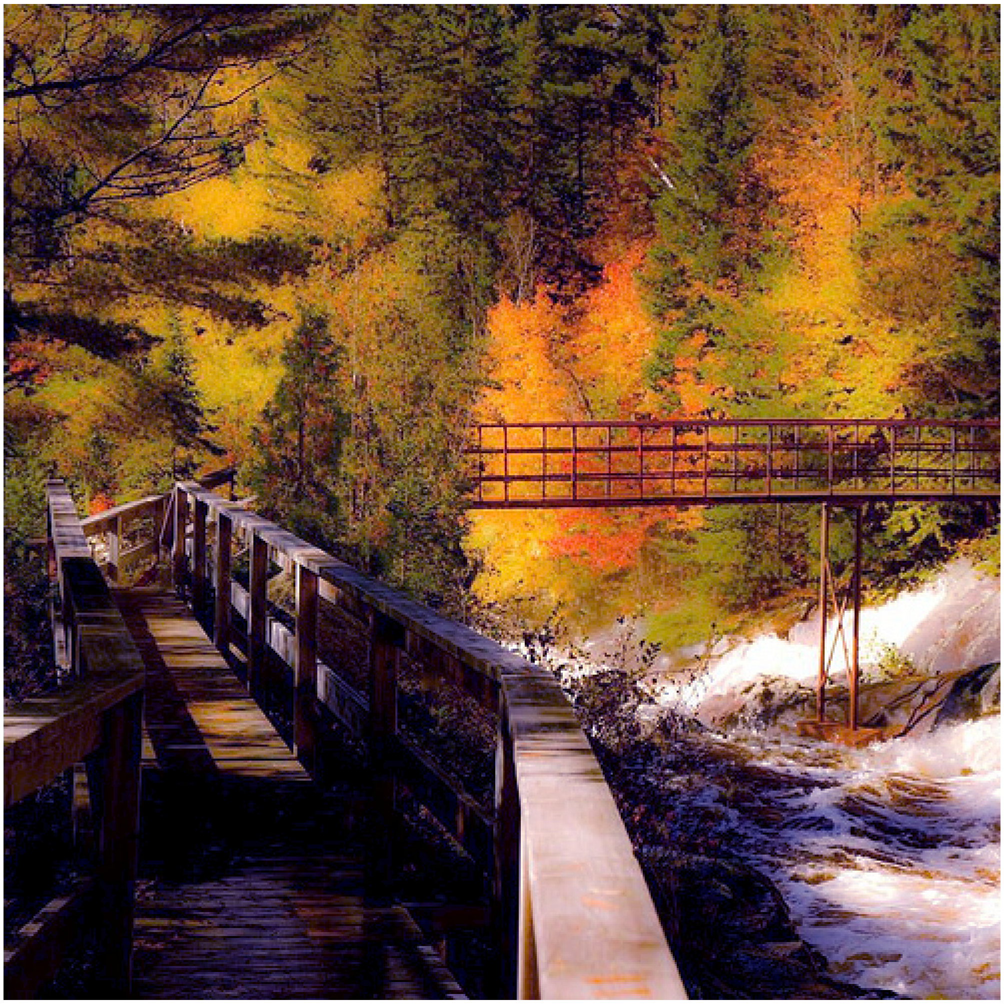

Figure 6. Blue band brightened

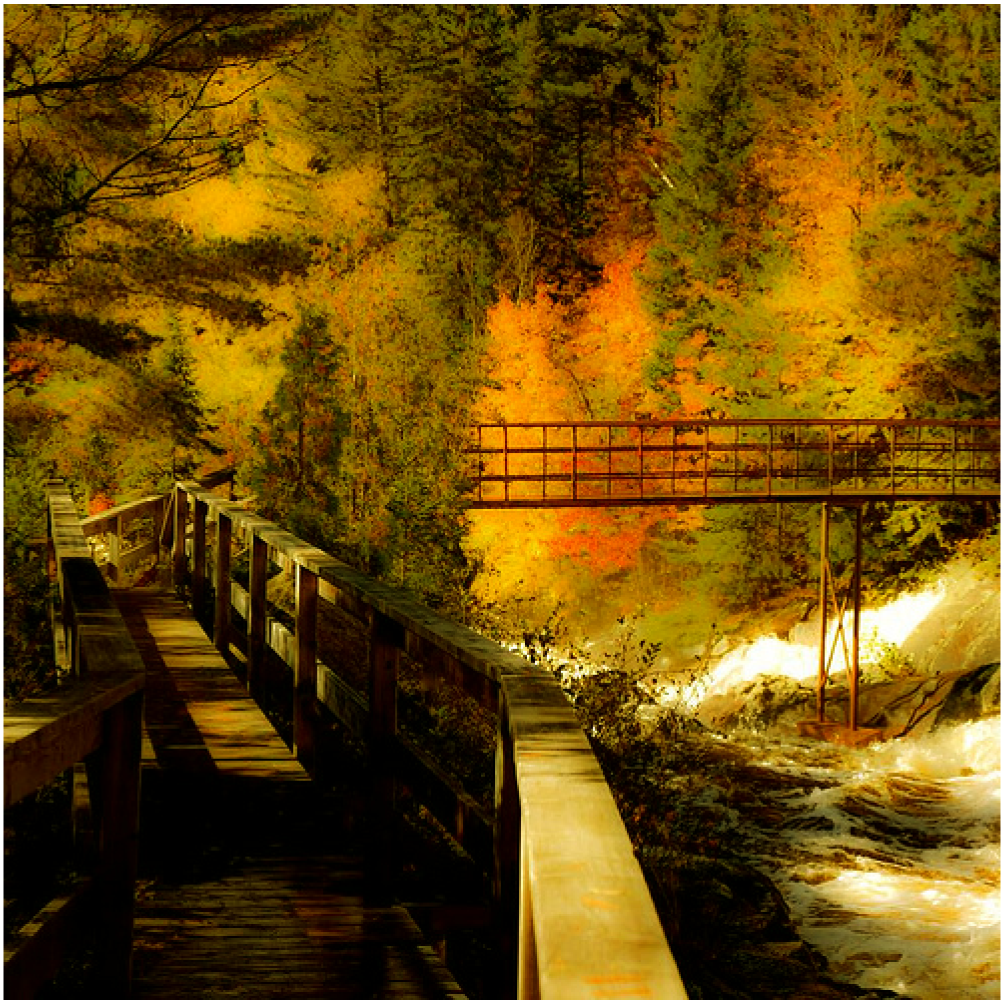

Figure 7. Blue band darkened

As you can see adjusting individual bands has a pretty drastic effect on the overall image. When an individual band is brightened (linearly increased) that bands color dominates the whole image. You can see in each of the examples where the band was increased the color is apparent. However if you look at the blue band, it is less apparent then the others. This is because our brain is less sensitive to blue, and an overall increase is harder to notice. When the bands are decreases, the other two colors begin to show more prominently.

Figure 8. Threshold image.

b) Figure 8 is the image threshold above the mean values for each band. This essentially sets a pixels value to 1 if it is greater than the average value, and 0 is it is less. Because each band can only have either a 1 or a 0, there can only be 8 colors in this image. They are as follows:

r g b	color
000	black
001	blue
010	green
011	teal
100	red
101	purple
110	yellow

111 black

Experiment 2

a)

Figure 10. Value band gamma increased

Figure 11. Value band gamma decreased.

This experiment increased or decreased the value band only, and did not touch either the hue band or the saturation band. As you can see, this had the overall effect of lightening the image when the value band was increased, and darkening the image when the value band was decreased. The image does not get more or less saturated though, and this is evident in the figures above. Adjusting the value band has the effect of sliding up or down the grey line.

b)

Figure 12. Saturation band linearly increased.

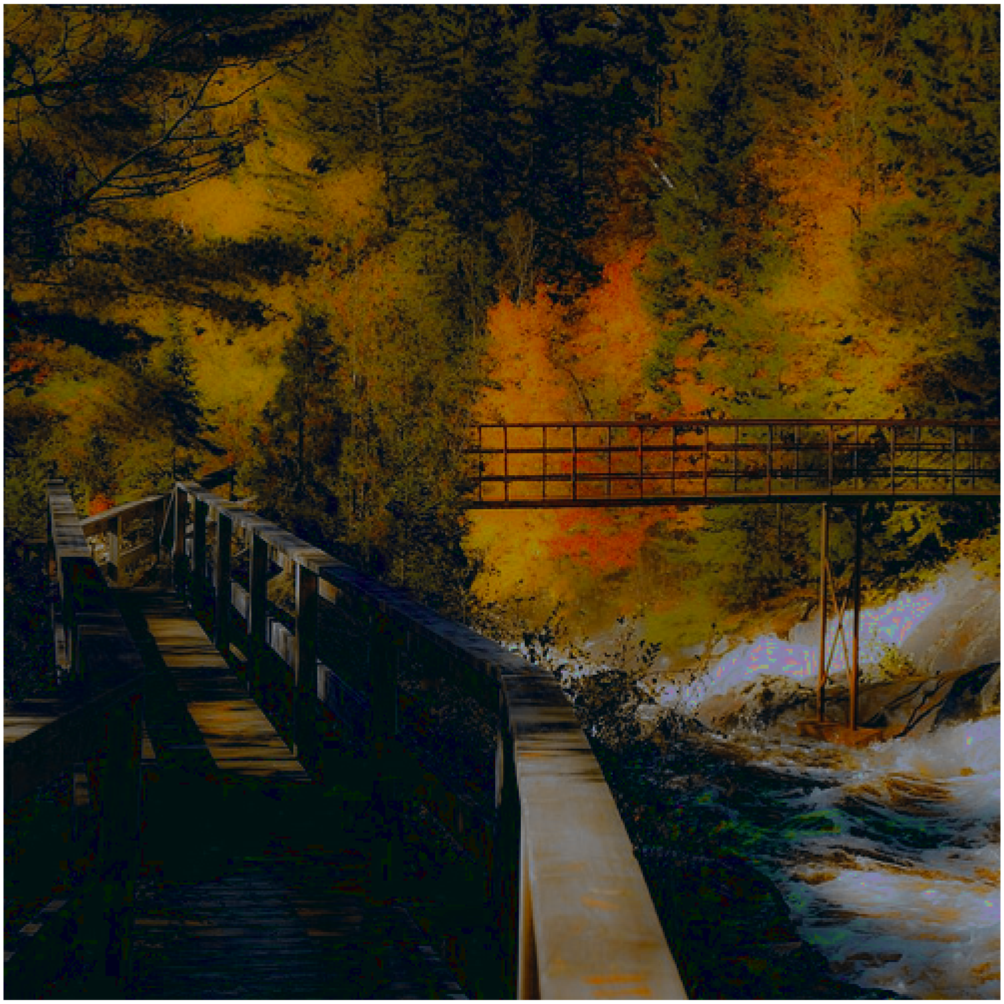

Figure 13. Saturation band linearly decreased.

Increasing or decreasing the saturation band adjusts all the colors at once. As you can see some artifacts do arise in the process. This can be attributed to saturation values greater than 1 or less than 0. To fix these issues, a matlab script can be written that will go through and set values greater than 1 back down to 1 and values less than 0 up to 0.

c)

Figure 14. green shifted hue.

Figure 15. Blue shifted hue.

The two images above are both linear hue shifts. Figure 14 is hue+1.5 and figure 15 is hue-1.5. As you can see this tints the overall image. Note how the blue effect is less noticeable, which is again attributable to how our brain reads blue values. In the code for these shifts, if the values was above 2pi or below 0, it was wrapped around to stay in the range of [0 2pi].

Experiment 3

a)

Figure 16. Original image

Figure 17. Altered image.

This experiment dealt with restoring color altered image. This was accomplished with a transformation matrix applied to each pixel value. To do this you first need a table mapping colors in the altered image to colors in the unaltered image. Below is the table used in this example.

pixel (x y)	12 22	32 47	332 82	330 16	114 186	163 159	170 148	265 339	316 293	376 530	32 552	52 396	80 458	91 395	265 81
orig (rgb)	171 172	220 101	144 164	169 174	242 207	179 142	54 61	252 241	156 119	3 3	252 254	127 138	94 91	155 119	211 148

	164	81	191	177	53	113	53	100	93	3	243	140	82	85	131
alt (rgb)	174 171 166	218 142 84	148 145 190	175 169 179	215 239 55	181 160 115	58 57 55	211 249 102	159 136 95	7 2 6	246 249 242	128 131 138	100 93 85	158 138 88	211 171 135

Table 1. Training set used for color correction

b) The transformation matrix that these pixel values produce is:
B =

$$\begin{matrix} 0.8044 & 0.3518 & -0.1569 \\ -0.4445 & 1.2178 & 0.2587 \\ -0.0417 & 0.0229 & 1.0157 \end{matrix}$$

c)

Figure 18. Color corrected image.

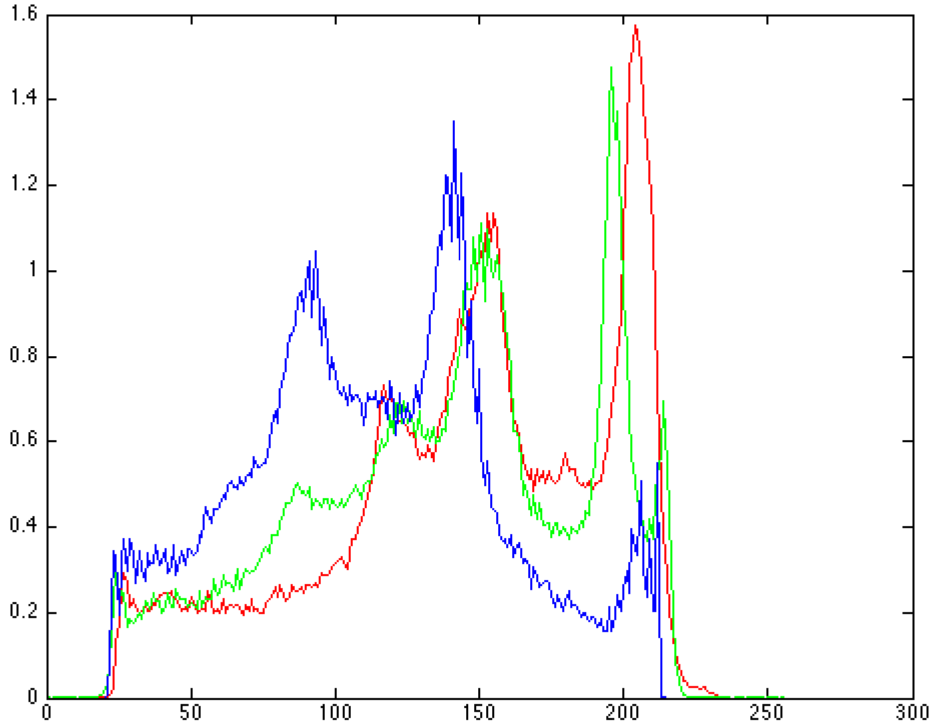

Figure 19. PDF of color corrected image

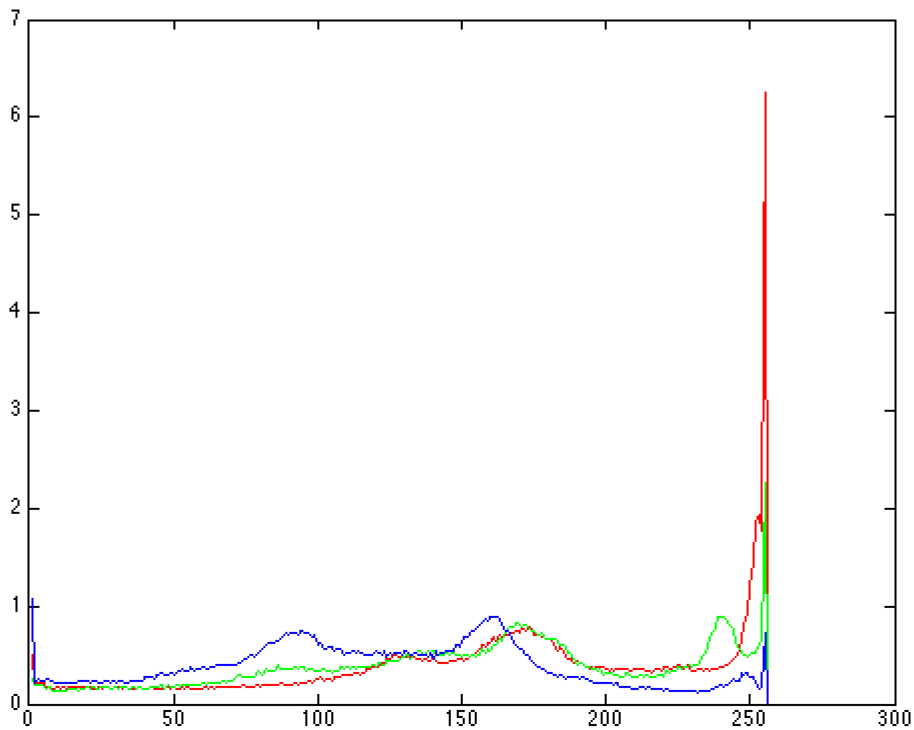

Figure 20. PDF of original image.

As you can see the color corrected image does not look exactly like the original image, but it is close. The probability density functions are quite different, as you can see in figures 19 and 20. This shows up somewhat in the actual images themselves.

Conclusion

Each of these experiments showed different methods to modify the overall color content of an image. An image can be thought of as a combination of red, green and blue values, and this is the most common way to represent images. Computers display images in this manner. But you can also represent an image as a combination of hue, saturation, and value components. This is essentially the Cartesian coordinates of RGB rotated and represented in cylindrical coordinates, with the center axis of the cylinder the diagonal of the RGB cube. While HSV values are a little more difficult to work with, they provide a much richer interface for interacting and manipulating colors.