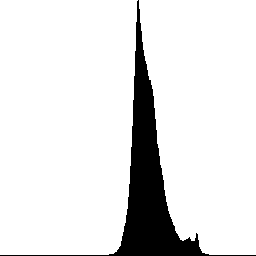
Project 2: Histogram Modification and Color Processing

# Introduction

The purpose of this assignment is to experiment with histogram analysis and modification in gray, RGB, and HSI color spaces. As such, five new filters were added to the image processor program: GrayHistogram, GrayHistogramEqualization, HistogramStretch, HistogramEqualization, and HsiHistogramEqualization. Each new filter supports ROI as well as a few other parameters specific to the filter process.

# Gray Histogram

In order to know if the experiments are performing as expected, a new filter had to be added that would create a histogram image based on some input image. The GrayHistogram filter will produce such an image. Below is an example of an input image and the resulting histogram image:



Because this filter only does a histogram on gray images, it will convert color images to grayscale to create the histogram.

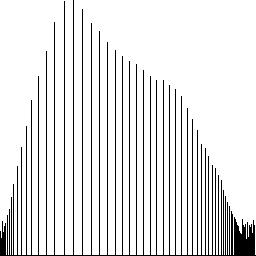
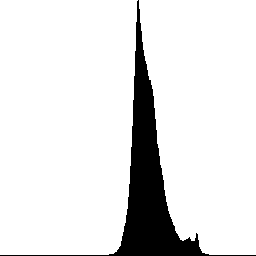
# Gray Histogram Equalization

Gray Histogram Equalization is a process for increasing the contrast of a gray scale image. The process starts by constructing a histogram of the image into an array of integers called . The index of the array represents the color value and the value of the array represents the number of times that color appears in the original image. After the histogram is constructed then a map of key and value pairs is constructed. The keys represent color values from the original image and the values represent the enhanced colors the image will be converted to. The values are created using this function ( is the original color value and 255 is the maximum color value):

The final step is to use the created map to map each pixel in the original image to an enhanced color in the new image. Below are before and after images after the gray histogram equalization:



Below is the histogram of the before image next to the histogram of the after image:



As you can see the histogram is spread across the whole spectrum and the after image is a lot clearer than the original image. The overall performance of the algorithm is .

# Histogram Stretching

Histogram stretching attempts to increase the contrast of an image by stretching the range of histogram values. The key to histogram stretching is to pick low and high values in the original histogram of the image. The implementation used in the Image Processor program allows the user to select a percentile or it will use a default value of 5%. The low and high values are labeled and respectively. Each pixel of the original image is converted using the following function on each color channel ( is the channel color):

Below is the before and after image using a 5% percentile. The overall performance of the algorithm is . Below is the before and after the histogram stretching. Visually the images do not look different:



# Histogram Equalization in RGB

Histogram equalization in RGB is similar to gray histogram equalization. The difference is that instead of doing an equalization on a single gray channel, and equalization process is done on each color channel. This means a histogram is created for each color channel and maps are created based on those histograms using the same method as in gray histogram equalization. Below is the before and after images of the filter:



The contrast of the image improved and is brighter. The overall performance of the algorithm is .

# Histogram Equalization in HSI

Histogram equalization in HSI is similar to histogram equalization in RGB. Due to the nature of how color information is stored HSI color space, the hue channel should not be equalized. However, the Image Processor program does give the user the option to choose any combination of the HSI channels. The default behavior is S and I channels only. It should be noted that the image processor works in the RGB color space. As such, the input image has to be converted into a HSI color format to perform the equalization process then back to RGB for display and file format reasons. The conversion process from RGB to HSI is:

Where r, g, and b are red, blue, and green respectively. The histogram equalization process is the same as in RGB histogram equalization except that the 255 maximum value used in the other algorithms is 1 for the S channel and for the H channel. After the histogram equalization process is complete, the image needs to be converted back to RGB. The conversion process is below in pseudocode (note that h is in degrees and not radians):

if h = 0 then

b = i \* (1 - s)

g = i \* (1 - s)

r = i \* (1 + 2 \* s)

else if h < 120 then

b = i \* (1 - s)

r = i \* (1 + s \* cos((h / 180) \* PI) / cos(PI \* (60 - h) / 180))

g = 3 \* i - r - b

else if h = 120 then

b = i \* (1 - s)

g = i \* (1 + 2 \* s)

r = i \* (1 - s)

else if h < 240 then

r = i \* (1 - s)

g = i \* (1 + s \* cos(PI \* (h - 120) / 180) / cos(PI \* (180 - h) / 180))

b = 3 \* i - r - g

else if h = 240 then

b = i \* (1 + 2 \* s)

g = i \* (1 - s)

r = i \* (1 - s)

else

g = i \* (1 - s)

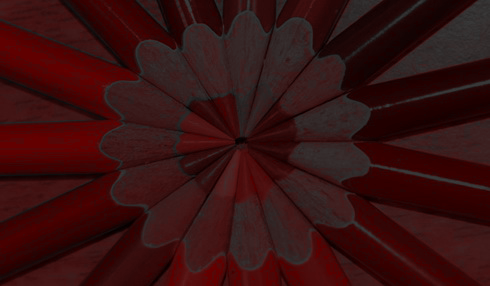
b = i \* (1 + s \* cos(PI \* (h - 240) / 180) / cos(PI \* (300 - h) / 180))

r = 3 \* i - g - b

end if

Below are some examples using various combinations of HSI channels for histogram equalization (labels next to the image indicate which channels were modified):

HS:



HSI:



I:



SI:



Looking at the photos the SI histogram equalization process improves the image the best. The overall performance of the algorithm is .

# Conclusion

Histogram equalization produces superior images to histogram stretching. However, the coefficient for the big O analysis between the two shows that histogram equalization is slower. In addition, HSI histogram equalization performs the slowest out of all the histogram equalization algorithms because of the additional calculations to convert the image from HSI to RGB then RGB to HSI. However, if the native format of the image was HSI and the device reading the image also supported a HSI format, then a histogram equalization on the I channel would be the fastest way to improve the image contrast (similar in performance to gray histogram equalization).