Tim Herrmann
EECS 332
MP3
Histogram Equalization

Programming Language:

- Python 2.7.6 with the following dependencies
  - matplotlib
  - numpy
  - cv2
  - pylab

Input:

An image, preferably gray scale, but the program will convert to gray scale if needed

Output:

An enhanced gray scale image with “flattened” histogram

Method:

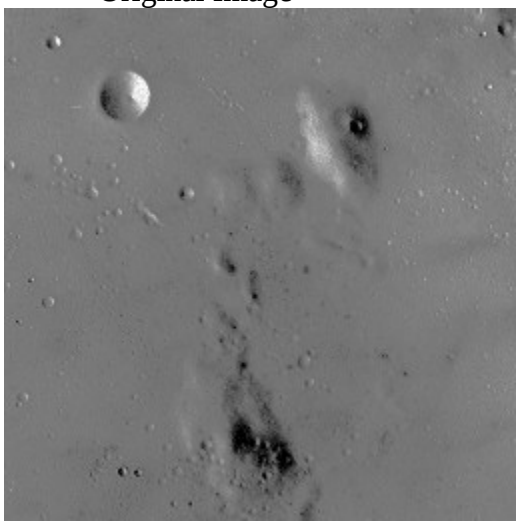
Step 1: read in image and convert to gray scale
Step 2: loop through all pixels and record the number of pixels at each value from 0-255
Step 3: calculate the cumulative distribution by summing the number of pixels at or below each value from 0-255
Step 4: loop through all the pixels calculate the new pixel grayscale value by dividing the cumulative distribution value by the total number of pixels, and multiplying by 255.

New Value = (CumulativeDistribution[Old Value]/TotalPixels) * 255

Step 5: save image to file and return the image array

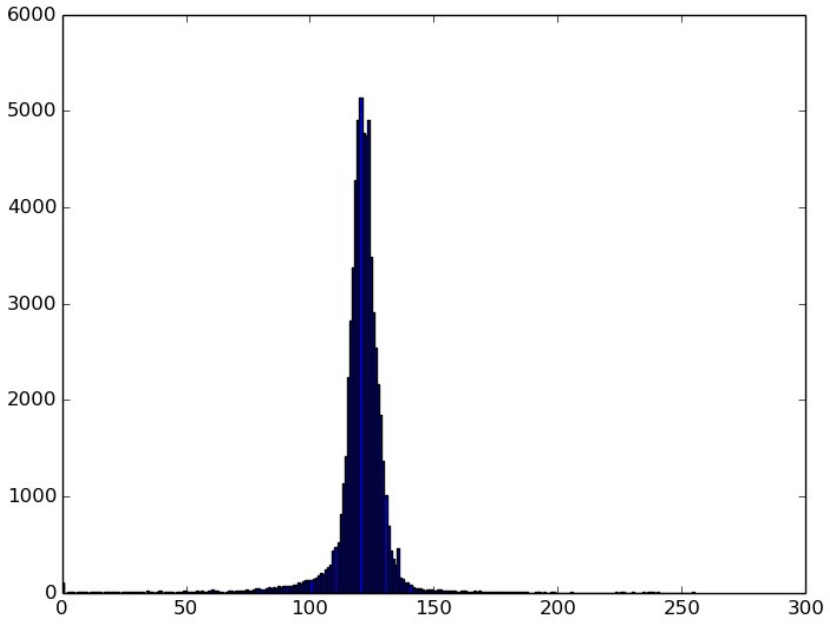
Example and Analysis:

Original Image

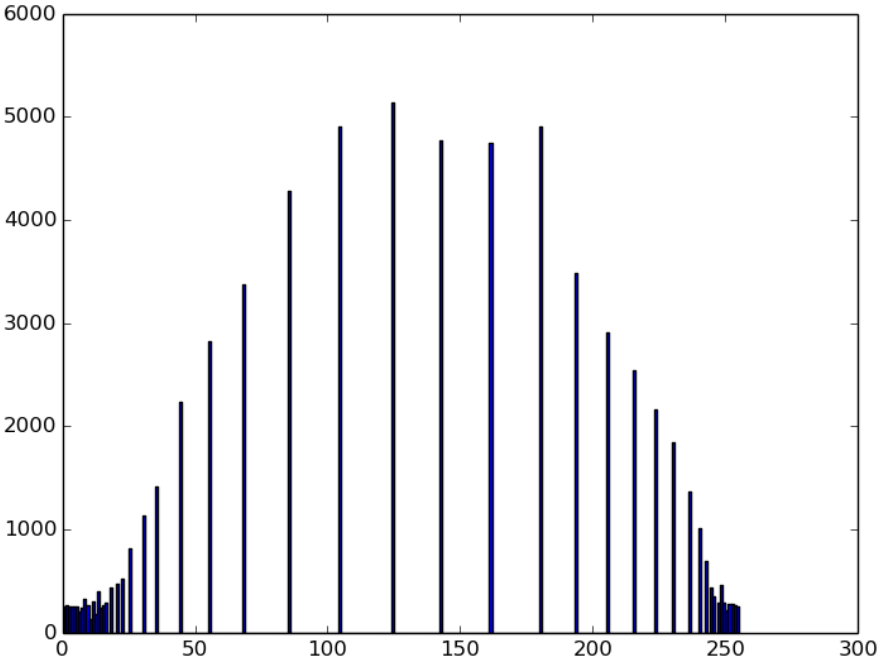


Enhanced Image

Original Histogram



Enhanced Histogram



The enhanced image takes all of the pixels centered around 115-140, and spreads them out over a much wider range. This allows us to see a lot of detail in areas that previously looked to be the exact same color. The enhanced histogram, when plotted with bins a single pixel wide, has a lot of gaps due to the extremely clumped nature of the original histogram. Pixels that were one value apart in the original

image, are now multiple values apart, with no pixels assigned to the values in between.

When the enhanced histogram is plotted with larger bins, the equalized nature becomes much more apparent.