CMP_SC 7650: Digital Image Processing
Homework 1B: Point Processes
By: Mikey Joyce
Due: 9/14/2023

Abstract:

This assignment required for the computation of image histograms for RGB and grayscale images. It also required for the computation of image histograms given a mask, so that only a specific region of the image would be the resulting histogram. Another goal of the assignment was to compute a min-max linear stretch and also discard certain percentages of pixels from an image. All of the goals in this assignment were achieved shown in the results section.

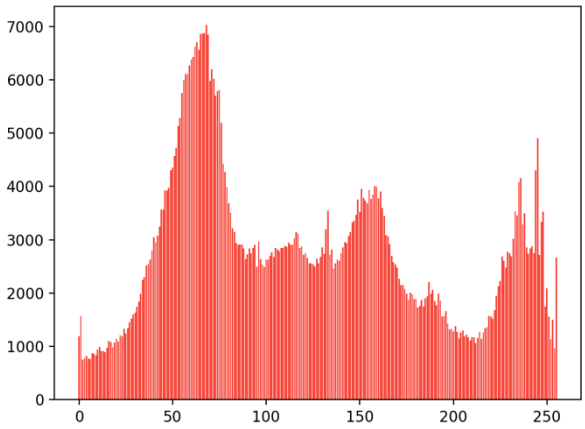
Introduction:

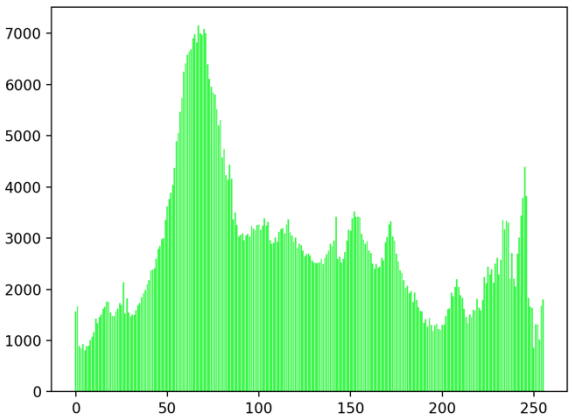
This assignment will be completed in two parts. Part 1 is dealing with histogram computation where the input test image is *rgb_stop.jpg* with a given input mask *mask_stop2.png*. A histogram will be computed for each channel in the RGB image for bins=256 and bins=16. The same histograms will be computed a second time except utilizing the mask. The function myimghist(file_path, nbins, mask) was created to compute these histograms for any number of bins 1-256. Lastly, to ensure this function is efficient the time taken to complete the histogram computation will be reported as well. Part 2 of this assignment is dealing with the min-max linear contrast stretch and the pixel discards. In this part a min-max linear contrast stretch function will be created to perform that operation. Another function called perform_discard(image, amount) was also created to perform the pixel discard based on percentages. The intensity ranges for each operation will be reported at the end as well.

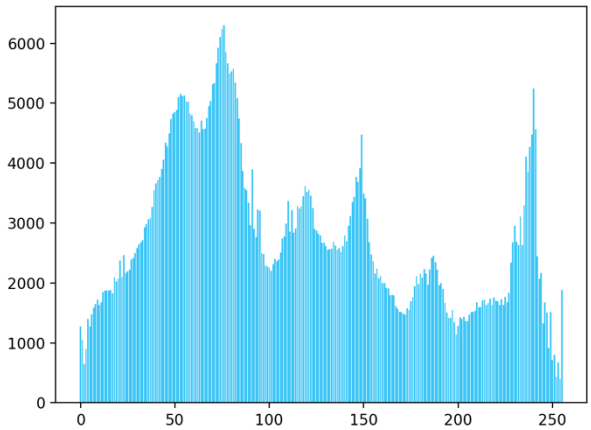
Histogram Computation Test Results:

Figure 1 (left) and 2 (right) show the input test images for this assignment. Figure 1 = rgb_stop.jpg; Figure 2 = mask_stop2.png

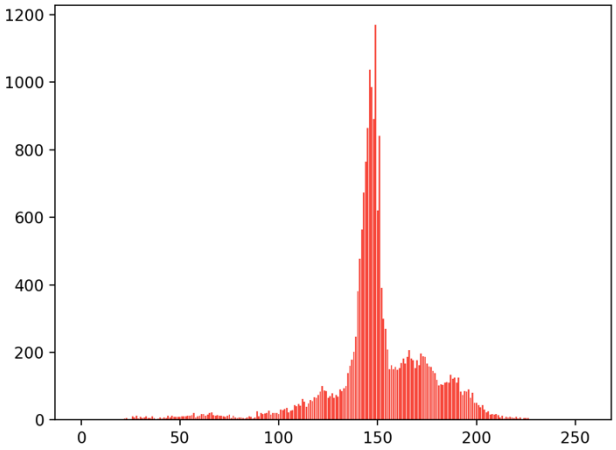
Entire Image Histograms (nbins = 256):

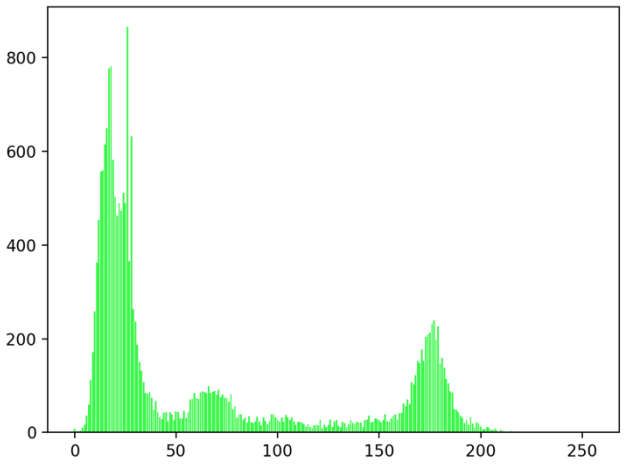


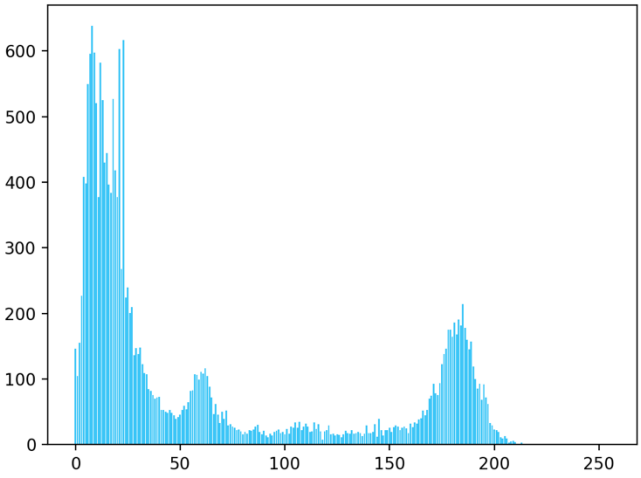


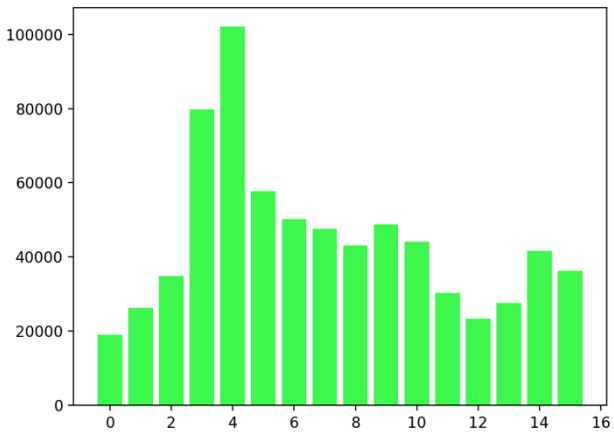


Regional Histograms (nbins=256):

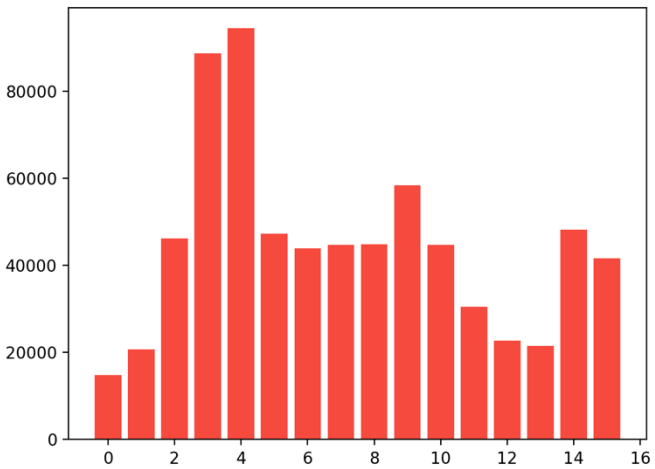


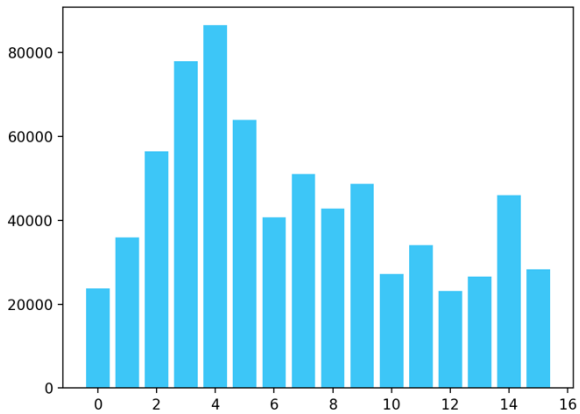




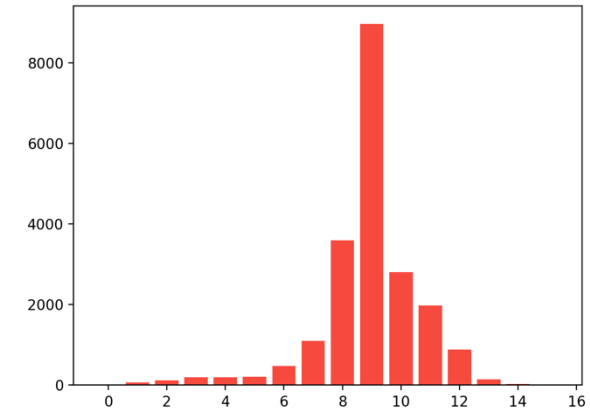


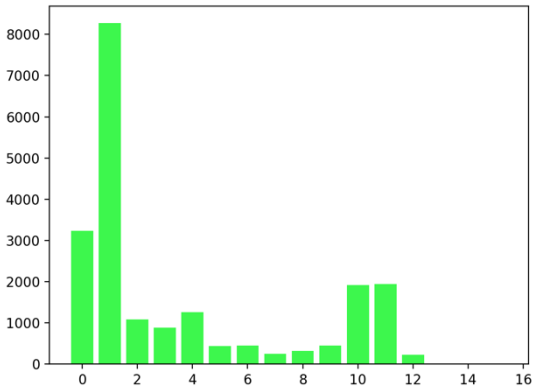
Entire Image Histograms (nbins=16):





Regional Histograms (nbins=16):





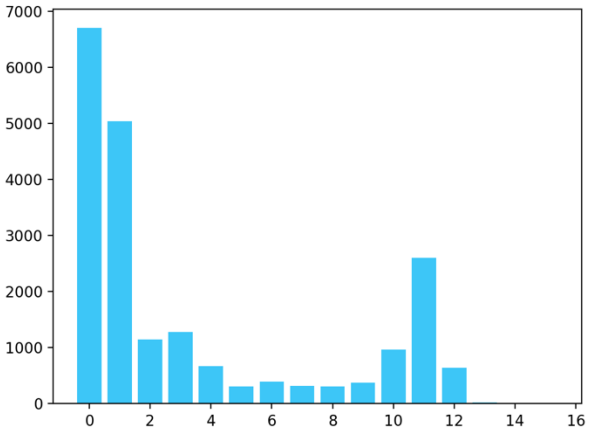


Table 1: Processing times

Histogram Type	nbins	Processing time
entire image	256 bins	0.7235 s
entire image	16 bins	0.6951 s
regional	256 bins	1.1056 s
regional	16 bins	1.1298 s

Contrast Enhancement Test Results:

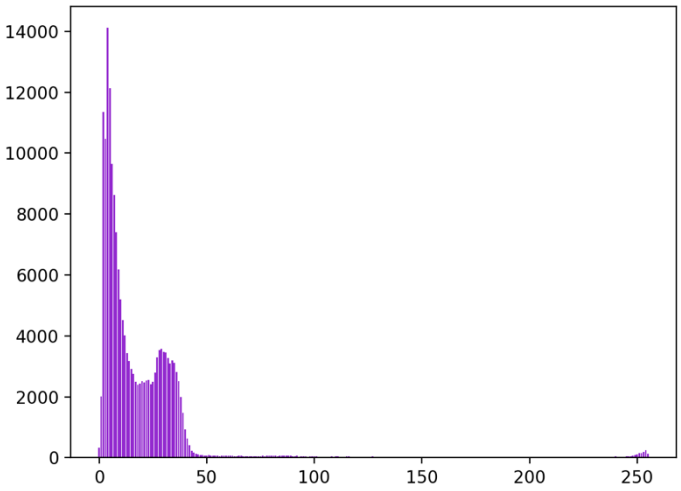
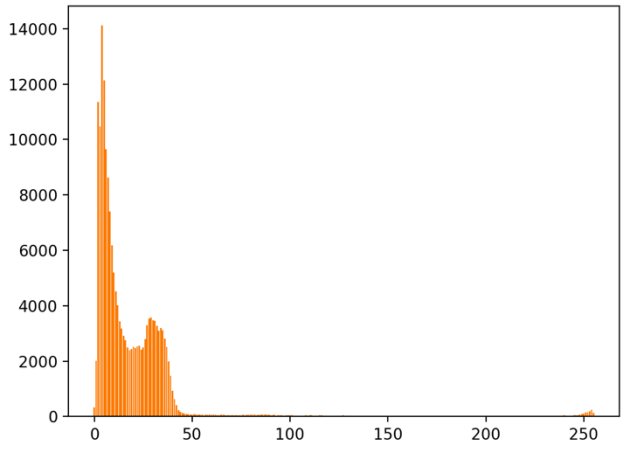


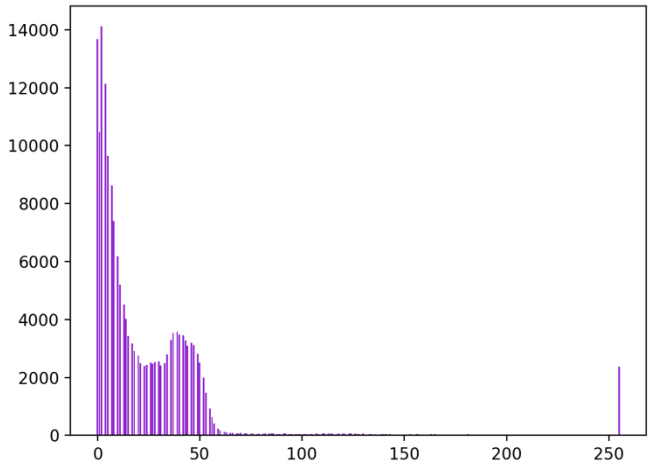
Figure 3 (left) and 4 (right). Figure 3 = night_gray.png; Figure 4 = original histogram

Enhanced Images:

Linear Stretch: Top left
1% Discard: Top right
5% Discard: Bottom

Enhanced Histograms:





Linear Stretch: Top left
1% Discard: Top right
5% Discard: Bottom

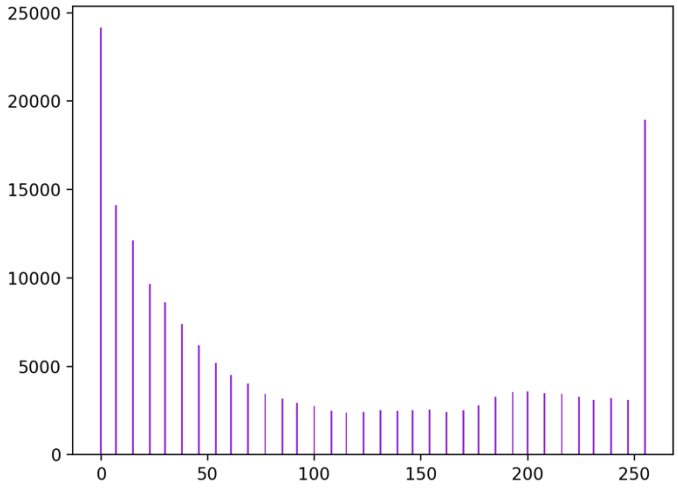


Table 2: Intensity Ranges

Image	Minimum Intensity	Maximum Intensity
Original	0	255
Min-Max Linear Stretch	0	255
1% Discard	2	178
5% Discard	3	36

Conclusion:

First, the results from the histogram computation will be reviewed. Initially, when developing this function, the computation time for this was taking a significant amount of time (roughly 10 minutes). This would be unacceptable in production as the compute needed to process 1 image would be way too much. Dr. Bunyak gave good advice in class and mentioned instead of using 2 separate loops (one for the pixels and one for the bins) that it could be accomplished with just a loop for the pixels. This is because it is possible to map the histogram indices utilizing this equation:

round(pixel_value / bin_size)

This allowed for the computations of the histograms to be much faster, with each computation being roughly in the ballpark of 1 second to complete. Another note on this part is that it was interesting to see how the histograms changed with the mask. Since the mask was hovering over the stop sign, the red pixels were more towards the left. However, I expected it to be closer to 255 for the red channel with the mask, but since there are a lot of white pixels as well, the peak of the red histogram is closer to the center.

Now, the results from the min-max stretch and percentage discards will be reviewed. Two different functions were created to complete these operations. The first function is min_max_stretch(img) that performs the min-max linear stretch. The second function is perform_discard(img, amount) which takes an image and percentage value and discards pixels from the percentage provided. The first operation we expect will do nothing to the image because the minimum and maximum intensities to the given image are 0 and 255 respectively. For the min-max stretch to do something there needs to be intensities ranges not used either on the low end, high end, or both. As we see in the first output image, it looks exactly the same as the input image, which is what was expected. The same goes for the histograms. The second output image is utilizing a 1% pixel discard before performing the min-max linear stretch. Because of this it can be observed that the output image is slightly different than the input image. If we look at the sign and the curb, we can see we lost some details because of the increased brightness. The histogram is also slightly different, although it still preserves a similar shape as the initial histogram. With the last output image, we perform a 5% discard before doing the min-max stretch. We expect that this image will be much brighter due to all of the initial bright pixels being discarded. As we can see we lost many details in the image due to increased brightness. The resulting histogram from this image is highly distributed throughout the entire range of intensities.

References:

1. Python, PyCharm, OpenCV, NumPy, time, sys, matplotlib, Preview
2. Lec03_IntensityTransformations.pdf
3. Lec03_IntensityTransformations_v2.pdf
4. OpenCV documentation: https://docs.opencv.org/4.x/