CENG 421 Assignment 1

Cameron Long V00748439

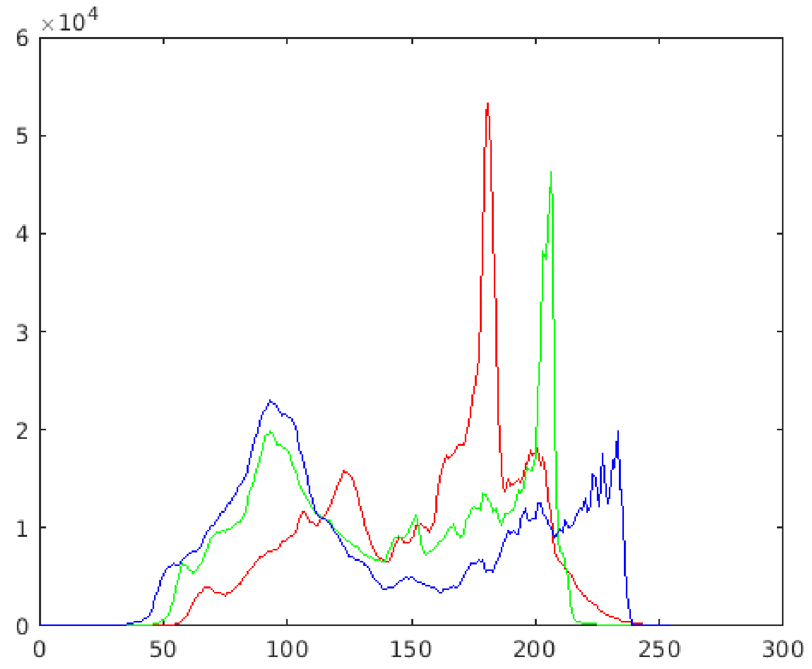
Tuesday January 30th, 2018

Part 0:

The rationale for this image is relatively simplistic, I felt it had a good amount of color, and clearly a higher amount of red. It was also a high-resolution image just in case the assignment questions needed it.

Part 1:

The generated histogram of the image is:

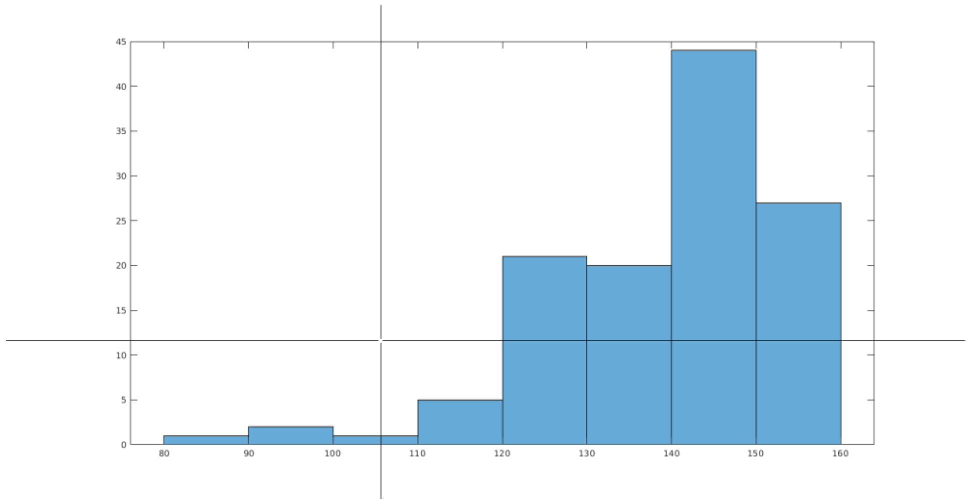


The three selected images have values as below:

POINT	RED	GREEN	BLUE	INTENSITY	MEAN	STANDARD DEVIATION
1	177	140	122	149	138.9537	4.2477
2	198	136	91	148	124.3554	0.5816
3	194	197	199	196	195.8676	2.4925

Homogenous/ Non-Homogenous distribution of intensity values?

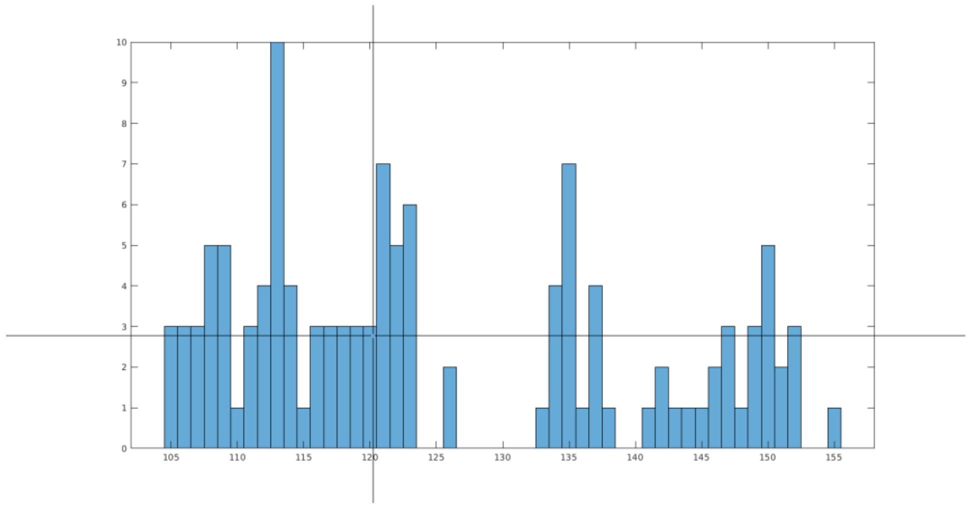
Point 1:



The point was taken in the sky at the top of the image, and is a reasonably homogenous area of intensity. Although there are outliers at the tail end that have skewed the standard deviation somewhat, the overwhelming majority of pixel intensities far within a 10-point range.

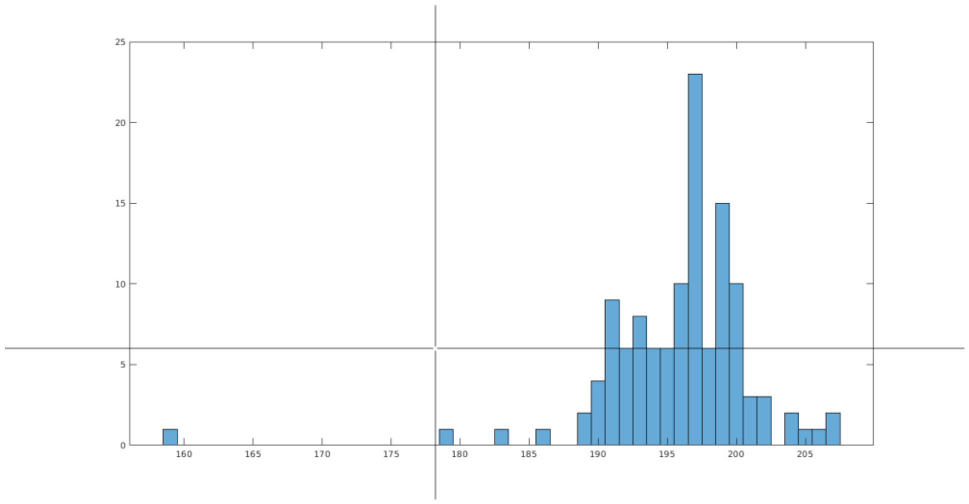
Point 2:

White stairs



This pixel was taken from the white stairs of the building, and has produced some interesting results. The range of intensity is certainly a lot lower than the previous point, but there are never too many pixels in any bucket. The standard deviation is extremely low, which is to be expected as this point exhibits very heterogenous behavior.

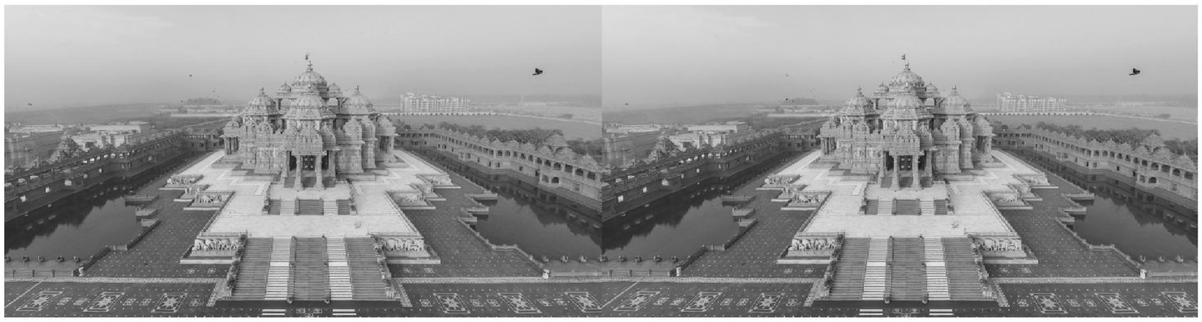
Point 3:



This point, taken at the edge of two extreme color spaces, is a bit of a mix of the two previous points. There is a bit more of a cluster than point 2, but less so than point 1. There are also visible outliers, suggesting a contrast in the two areas of the window, also seen in the standard deviation metric. I would classify this as heterogenous behavior.

Part 2:

The Output of the myrbg2gray.m file looks like this:



With the mat2grey() picture on the left, and the applied function on the right. As you can see, The equation Y=0.299R+0.59G+0.11B is a very good approximation of a greyscale image, or the function used to apply images to greyscale in the first place. As I understand it, what we are doing is calculating the intensity of the color values in the given pixel, and interpreting it as a gray level in an 8-bit space.

Part 3:

The final output of mtrbg2ind() looks something like this:

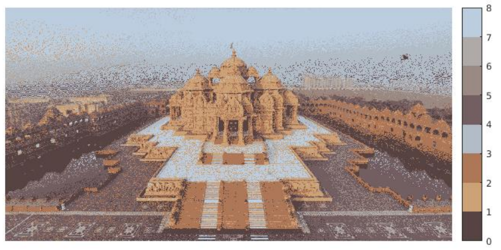


Image indexing is useful for managing colors in a digital image to save memory/image sizes. We are removing the color data from the image and storing it as a separate piece of data called a "palette". Every piece of the array represents a color, indexed by its position within the array.

Indexed color saves a lot of memory, storage space, and transmission time. For example, using the TrueColor Method, a 640 X 480-pixel image requires 300KiB, instead of 900KiB uncompressed. These sorts of techniques were widely used in the early days of personal computing, to reduce costs for RAM chips and the like. Over time, the overhead needed to create such images was reduced, so we could create more color-accurate images using a higher bit color space. In the above example, we have only used an 8-bit color space, so the image is not re-created with as much color accuracy.