Project 2: Point Operations

Chris Ruby

September 25, 2018

1 Introduction

This report is an investigation into image histograms, point operations, and background image subtraction. Image histograms are an important tool in image processing allowing a user to extract pixel data from a picture and be able to see the distribution of the pixels. Upon collecting and analyzing the data, point operations allow for a shift in colors to be determined. Four of the point operations that are looked at in this report are shifting the colors to make an image brighter, shifting the colors to make an image darker, stretch the colors linearly so there is more contrast, and the use of histogram equalization.

2 Histogram Explorations

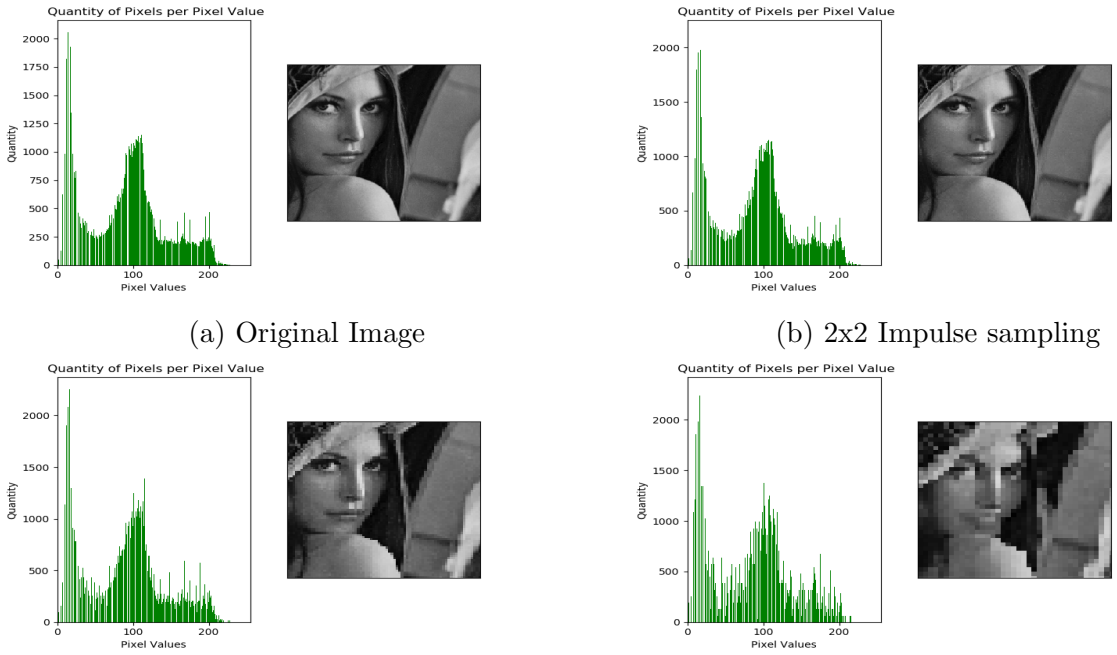
The original pictures to be used for Histogram Explorations were Lena.png and Crowd.png images that were created in Homework 1.

2.1 Lena

Figure 1: Lena.png

2.1.1 Impulse Sampling

Impulse Sampling in the pictures shown in Figure 2 consists of having a sampling region, NxN, square passes through an image and changes the selected area to the same value as the top left pixel value. As demonstrated in the histograms of the pictures, the frequency of pixel values tend to stay the same. As the NxN region gets larger the image shows more evidence of pixelation but does not distort the frequency in which pixel values are present in the image.



(a) Original Image

(b) 2x2 Impulse sampling

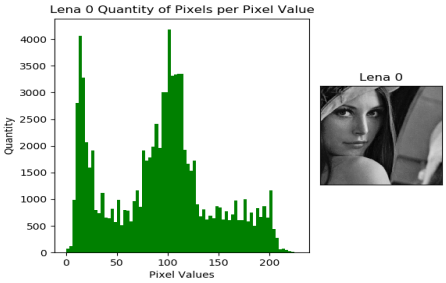
(c) 4x4 Impulse sampling

(d) 8x8 Impulse sampling

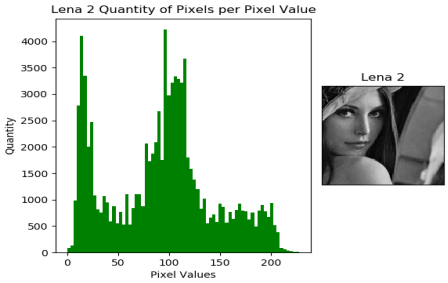
Figure 2: Impulse Sampling of Lena.png

2.1.2 Averaging Sampling

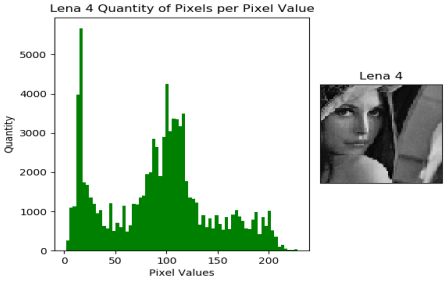
Average Sampling consists of sampling an NxN square and taking the average value of the sampled section and replacing all values with that value. As demonstrated in the images seen in Figure 3, the picture appears to show the affects of pixelation as the NxN square grows in size however, the pixels tend to stay roughly in the same bin of the histogram.



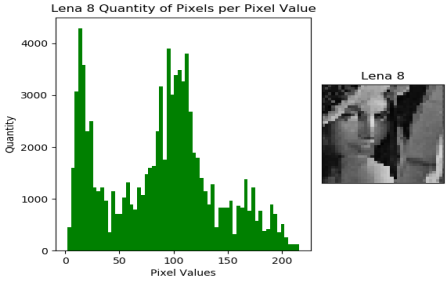
(a) Original Image



(b) 2x2 average sampling



(c) 4x4 average sampling

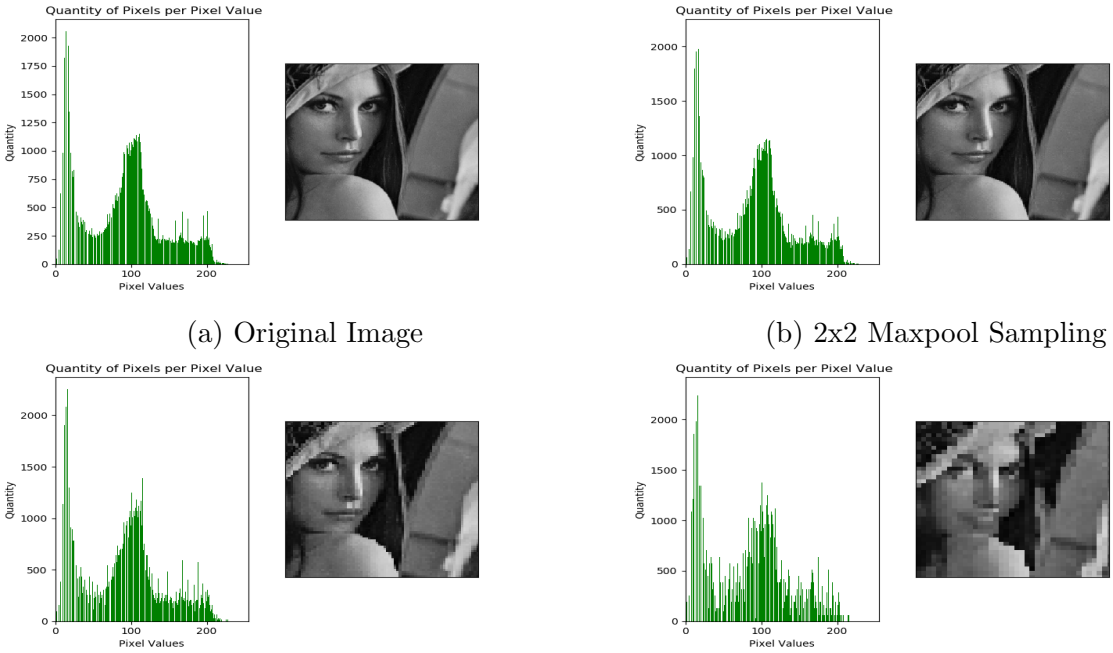


(d) 8x8 average sampling

Figure 3: Average Sampling of Lena.png

2.1.3 Maxpool Sampling

Maxpool sampling consists of sampling an NxN square and taking the highest value of the sampled section and replacing all values with that value. As the NxN square gets larger the picture tends to get a little brighter as expected. Seen in Figure 4 (d) there looks to be a slight change in pixel frequency cause the histogram to skew slightly to the right hand side or the 255 pixel value.



(a) Original Image

(b) 2x2 Maxpool Sampling

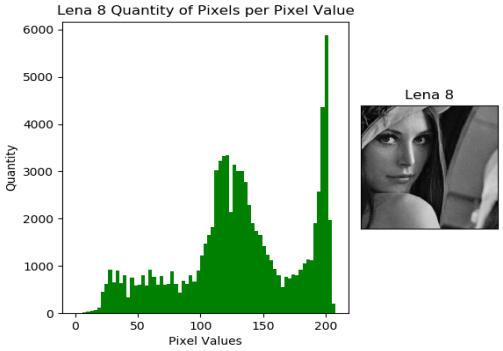
(c) 4x4 Maxpool Sampling

(d) 8x8 Maxpool Sampling

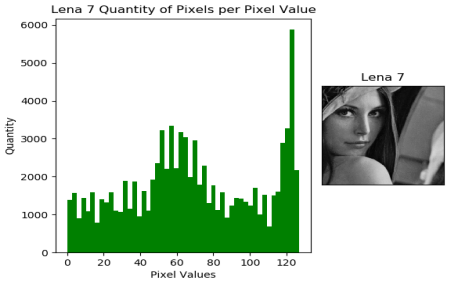
Figure 4: Intensity Quantization of Lena.png

2.1.4 Intensity Quantization

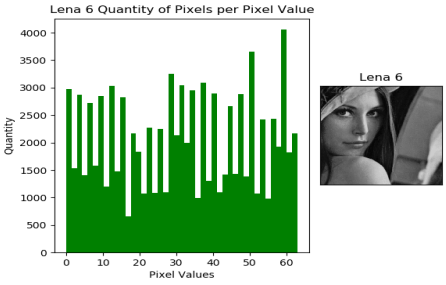
Intensity Quantization in this case consists of going from an 8-bit gray value number to a 7-bit gray value and so on down to a 5-bit gray level. The histograms of these images saw the most change out of the other operations mentioned previously. As seen in Figure 5, from (a) to (b) the histogram starts to spread out and equalize the pixel values. As the bit level continues to go down the histogram becomes more and more equalized.



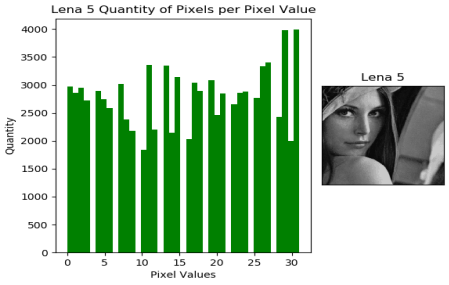
(a) 256 Gray Level (8-bits)



(b) 128 Gray Level (7-bits)



(c) 64 Gray Level (6-bits)



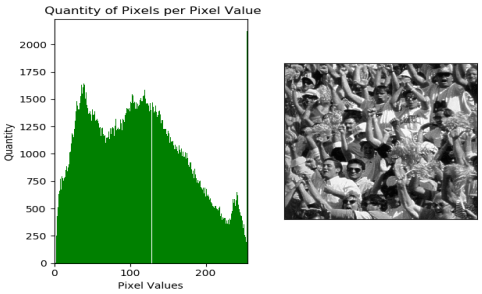
(d) 32 Gray Level (5-bits)

Figure 5: Intensity Quantization of Lena.png

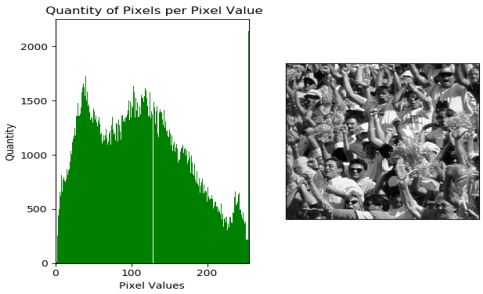
2.2 Crowd

Figure 6: Crowd.png

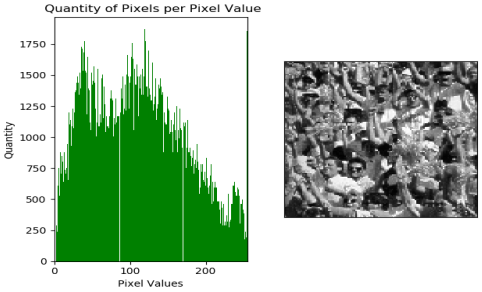
2.2.1 Impulse Sampling



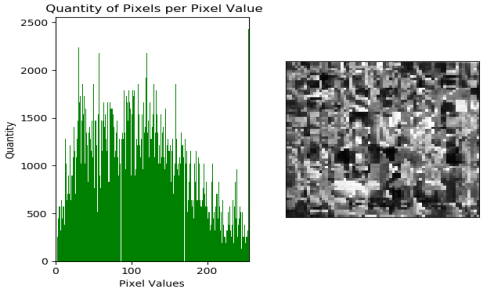
(a) Original Image



(b) 2x2 Impulse sampling



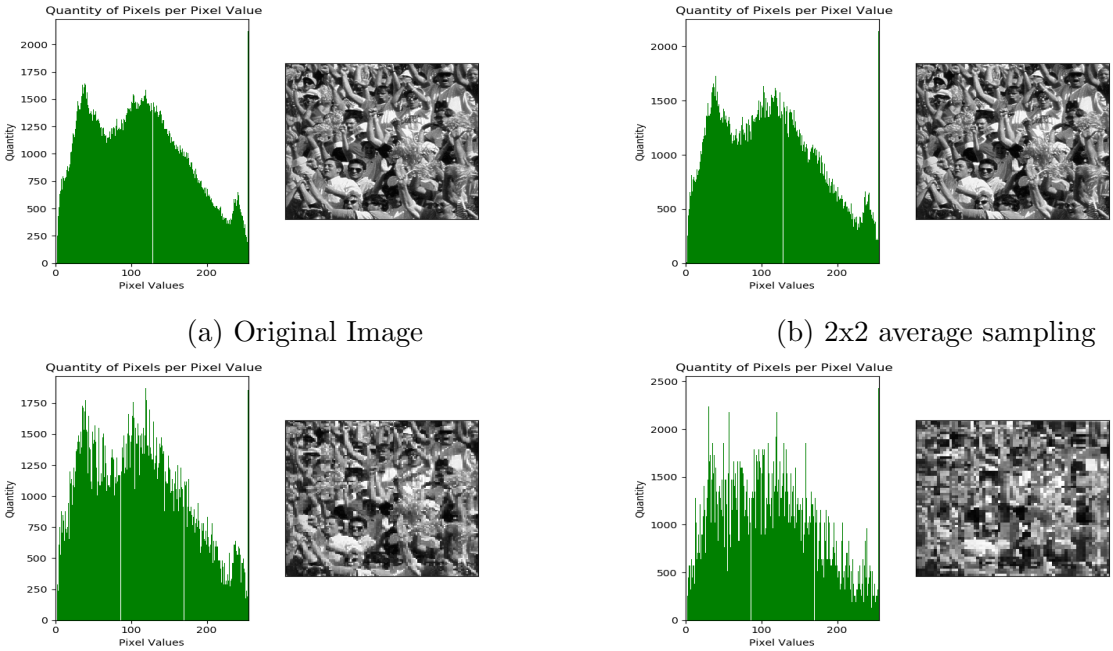
(c) 4x4 Impulse sampling



(d) 8x8 Impulse sampling

Figure 7: Impulse Sampling of Crowd.png

2.2.2 Averaging Sampling



(a) Original Image

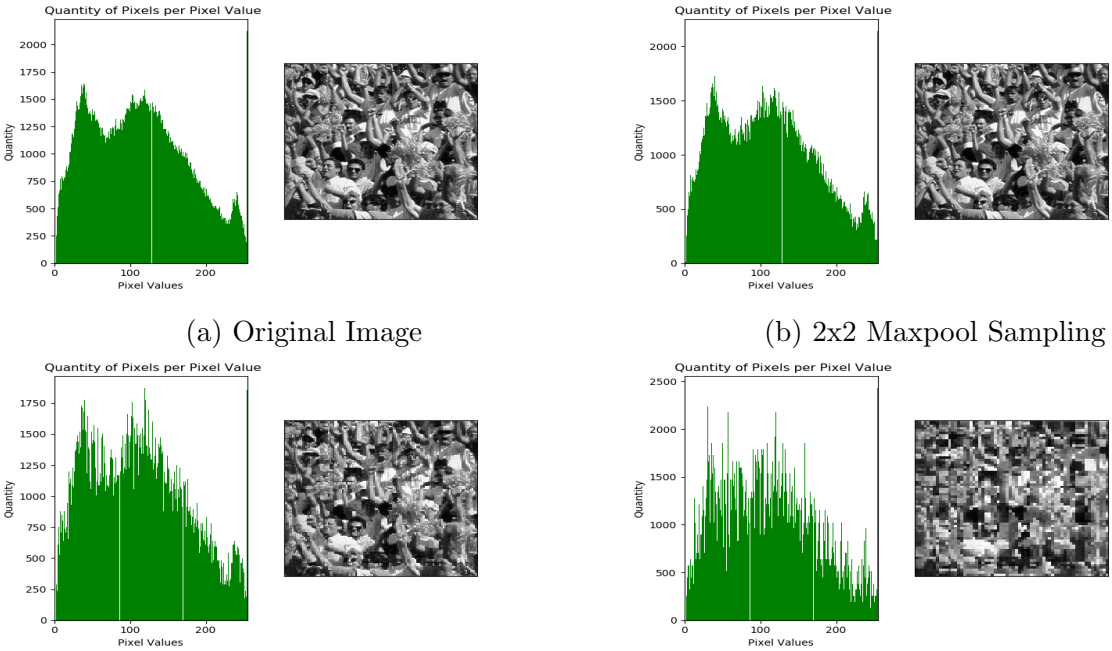
(b) 2x2 average sampling

(c) 4x4 average sampling

(d) 8x8 average sampling

Figure 8: Average Sampling of Crowd.png

2.2.3 Maxpool Sampling



(a) Original Image

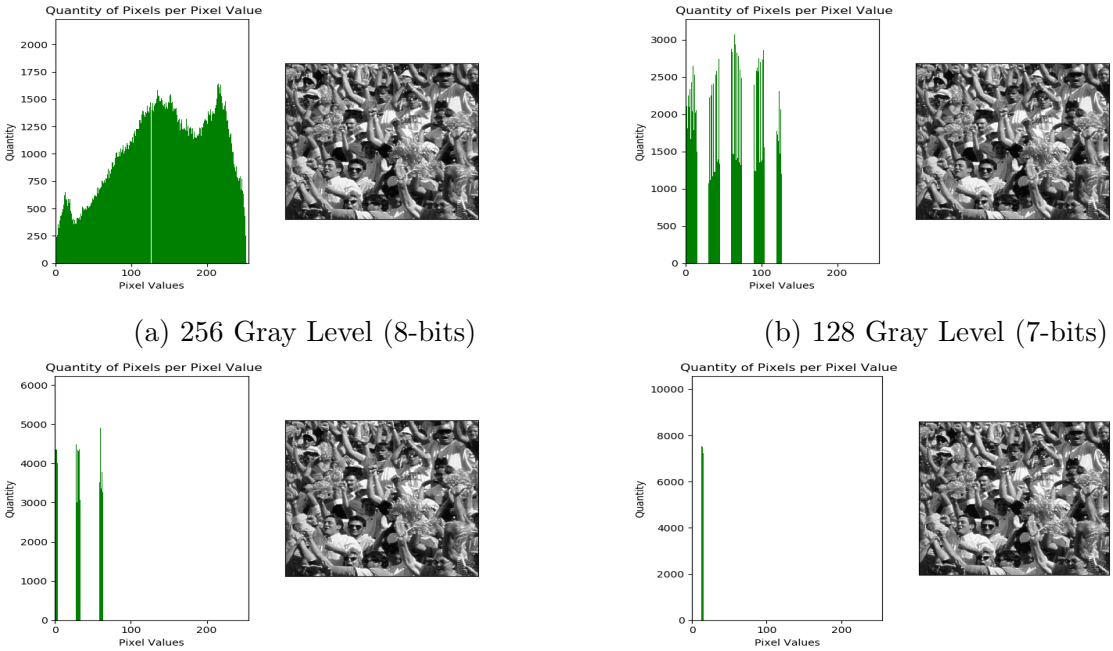
(b) 2x2 Maxpool Sampling

(c) 4x4 Maxpool Sampling

(d) 8x8 Maxpool Sampling

Figure 9: Intensity Quantization of Crowd.png

2.2.4 Intensity Quantization



(a) 256 Gray Level (8-bits)

(b) 128 Gray Level (7-bits)

(c) 64 Gray Level (6-bits)

(d) 32 Gray Level (5-bits)

Figure 10: Intensity Quantization of Crowd.png

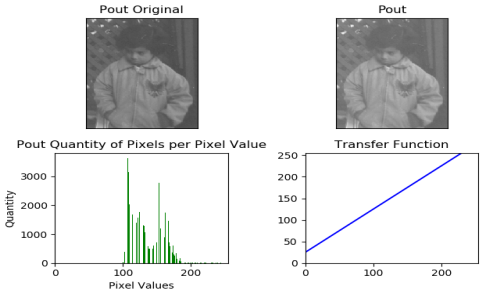
3 Point Operation Explorations

3.1 Pout

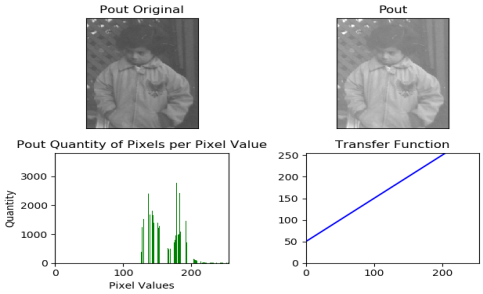
Figure 11: Pout.png

3.1.1 Shift The Colors So The Image Is Brighter(Linear Op)

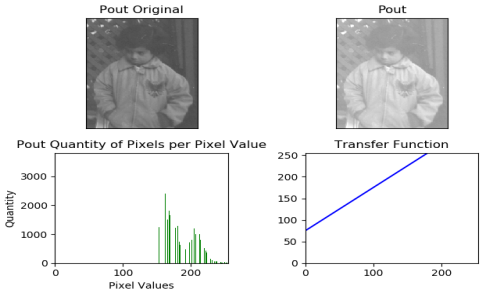
To accomplish a shift to make the image brighter, the image needed to be read into an array and evaluated at each x and y coordinate. Then an array of all zeros was created which was added to the desired value to make it brighter. Positive numbers will make the picture lighter and negative numbers will make the picture darker. In order for the values to stay within the 255 pixel value limit, the code searches the values and replaces the pixels greater than 255 with a pixel value of 255.



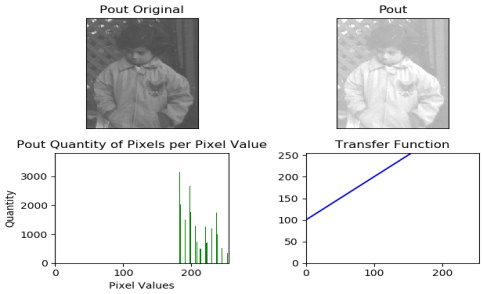
(a) Pout 25 gray levels brighter



(b) Pout 50 gray levels brighter



(c) Pout 75 gray levels brighter

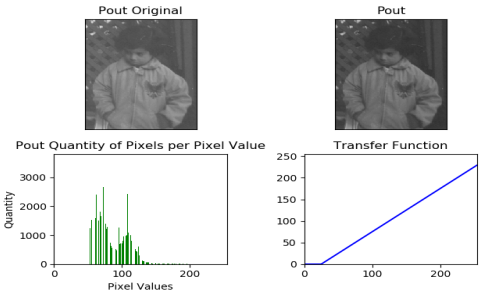


(d) Pout 100 gray levels lighter

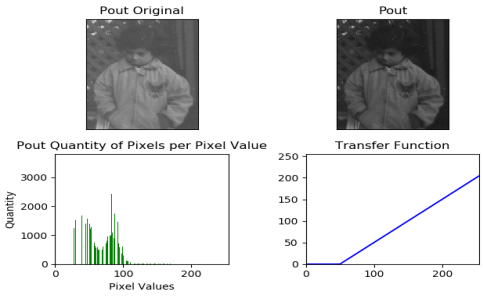
Figure 12: Shifting gray levels to make Pout brighter

3.1.2 Shift The Colors So The Image Is Darker(Linear Op)

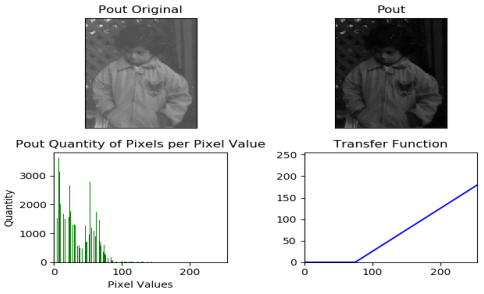
To accomplish a shift to make the image darker, the image needed to be read into an array and evaluated at each x and y coordinate. Then an array of all zeros was created which was subtracted the desired value to make it darker. Negative numbers will make the picture darker and positive numbers will make the picture lighter. In order for the values to be within the 0 pixel value minimum limit, the code searches the values and replaces the pixels with a pixel value negative with a 0.



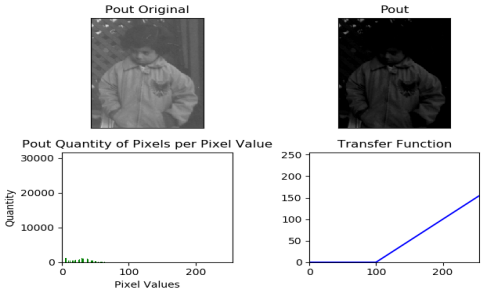
(a) Pout -25 gray levels brighter



(b) Pout -50 gray levels brighter



(c) Pout -75 gray levels brighter

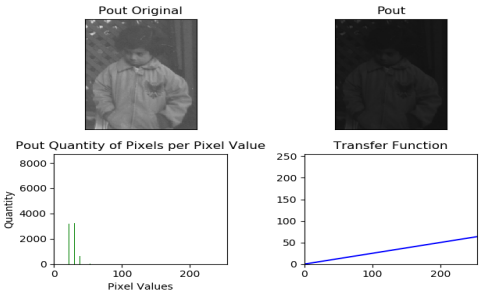


(d) Pout -100 gray levels lighter

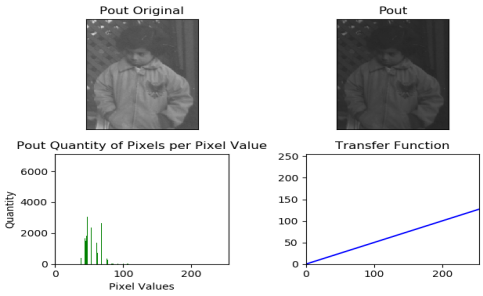
Figure 13: Shifting gray levels to make Pout darker

3.1.3 Stretch The Colors Linearly So There Is More Contrast (Linear Op)

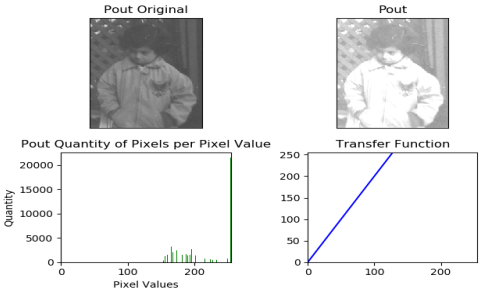
To accomplish a stretched image, the image needed to be read into an array and evaluated at each x and y coordinate. Then, an array of all zeros was created which was multiplied to the value to make the image have more contrast. Numbers less than 1 will make the image compress and numbers greater than 1 will make the picture stretch. In order for the values to be within the 0 pixel value minimum limit, the code searches the values and replaces the pixels with a pixel value negative with a 0. The values that are greater than 255 will be scanned through and replaced with a 255 pixel value.



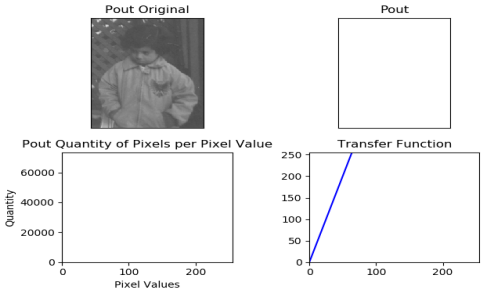
(a) Pout compressed by 1/4



(b) Pout compressed by 1/2



(c) Pout stretched by 2



(d) Pout stretched by 4

Figure 14: stretching image so there is more contrast

3.1.4 Equalize The Colors(Histogram Equalization)

Color equalization was by far the most complex of the point operations asked of us in this project. This process consisted of reading in the picture and putting the pixel values in their corresponding locations in an array. The bins of the histogram where divided into 256 even bins starting at 0 going to 255 then the cumulative sum was taken from the bin values. The cumulative sum was then passed through a CDF function to get each value from a range from 0 to 1. Upon finding the CDF that number was then multiplied by 255 to get each new pixel value. This was then distributed back throughout the pixel array and formed back into a picture.

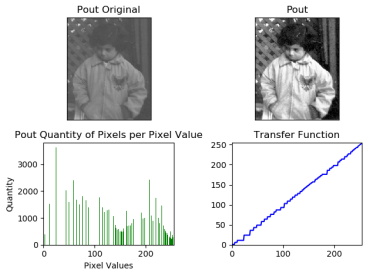


Figure 15: Pout Histogram Equalization

4 Image Subtraction

4.1 Original Images

(a) BFvideo1

(b) BFvideo2

(c) BFvideo3

(d) BFvideo4

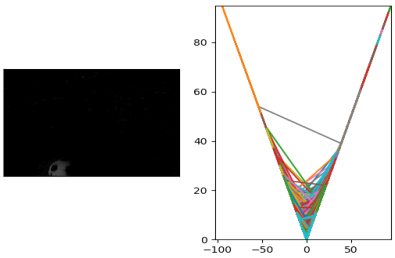
(e) BFvideo5

Figure 16: Original Still Frame Shots Of Ball Rolling Across Table

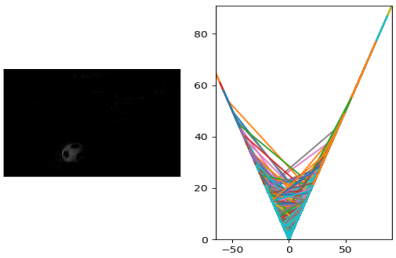
4.1.1 Absolute Value

The absolute value of the pictures was used when subtracting the two images it allows for negative values to be come positive pixel values allowing the values fall in the 0 to 255 pixel value bounds of an image. Absolute Value of image 1 Subtracted From Remaining Images

- BFvideo1 abs.png - BFvideo1 abs.png

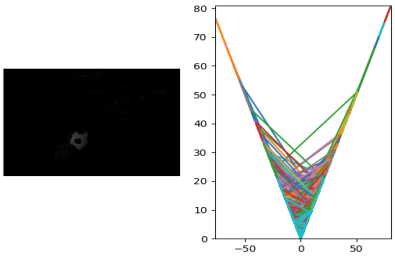


(a) 2-1

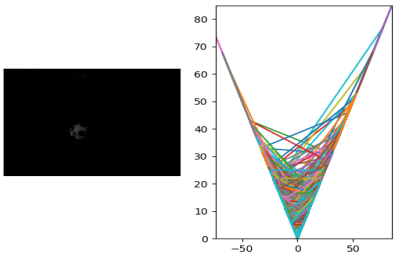


(b) 3-1

- BFvideo1 abs.png - BFvideo1 abs.png



(c) 4-1

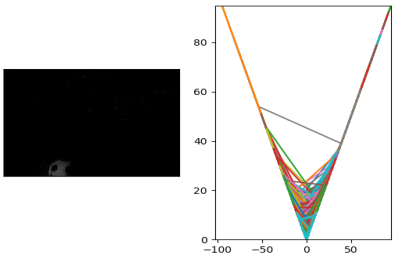


(d) 5-1

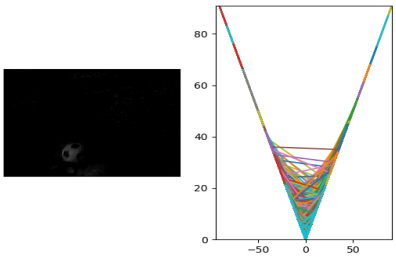
Figure 17: Absolute Value of image 1 Subtracted From Remaining Images

Absolute Value of Adjacent Images Subtracted From One Another

- BFvideo1 abs.png - BFvideo2 abs.png

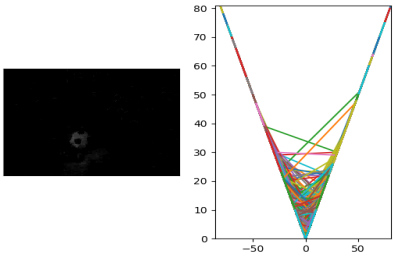


(a) 2-1

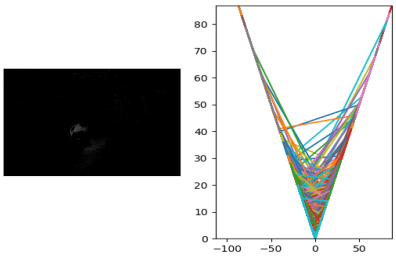


(b) 3-2

- BFvideo3 abs.png - BFvideo4 abs.png



(c) 4-3



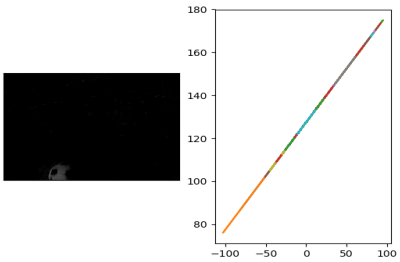
(d) 5-4

Figure 18: Absolute Value of Adjacent Images Subtracted From One Another

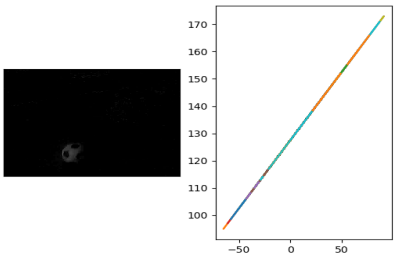
4.1.2 Linear Transform

The linear transformation I used after subtracting the pictures i took the array and added 255 and divided the whole sum by 2 this guarantees all values will fall in the range of 0 to 255. Linear Transform of image 1 Subtracted From Remaining Images

- BFvideo1 squish.png - BFvideo1 squish.png

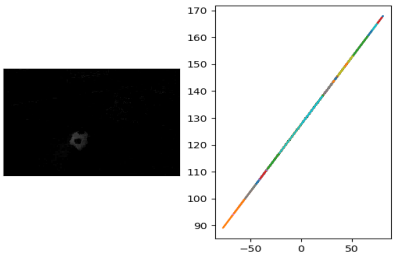


(a) 2-1

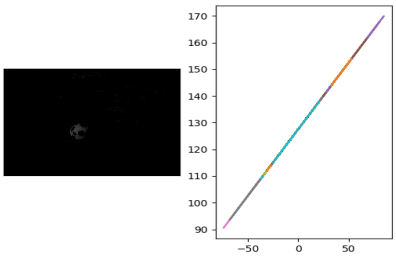


(b) 3-1

- BFvideo1 squish.png - BFvideo1 squish.png



(c) 4-1

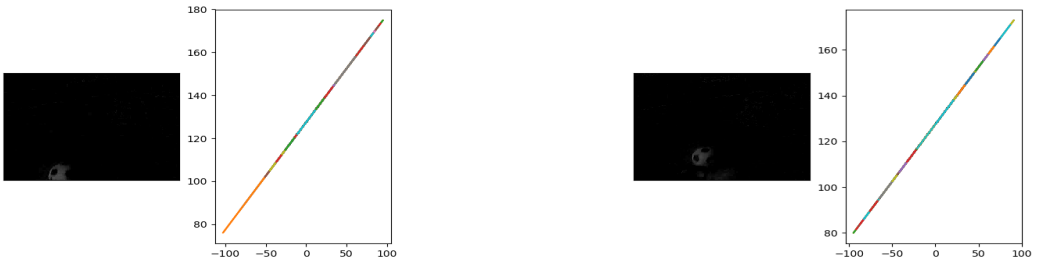


(d) 5-1

Figure 19: Linear Transformation of image 1 Subtracted From Remaining Images

Linear Transformation of Adjacent Images Subtracted From One Another

- BFvideo1 squish.png - BFvideo2 squish.png



(a) 2-1 (b) 3-2

- BFvideo3 squish.png - BFvideo4 squish.png

(c) 4-3 (d) 5-4

Figure 20: Linear Transformation of Adjacent Images Subtracted From One Another

5 Conclusions

In conclusion the most complex portion of this project was the histogram equalization but honestly the most rewarding. I found it interesting that you can take an image with the values skewed one way or the other and get a pretty decent image back just from equalization. I understand this will not work in all cases, i would assume if a corner was bright white it would have really thrown of the image. Seeing the effects of basic shifting of point values was interesting to see a minor shift in pixel values can really change the picture for both good or bad. I assume these techniques in the end could be used to put a picture through a function that will overall make the image more usable for the intended use.

I found the histograms to be interesting because even when it looked like we dramatically changed the image due to pixelation when you look at the histogram we stayed pretty true for the most part to the original distribution.