Avery Peiffer

Problem Set 0

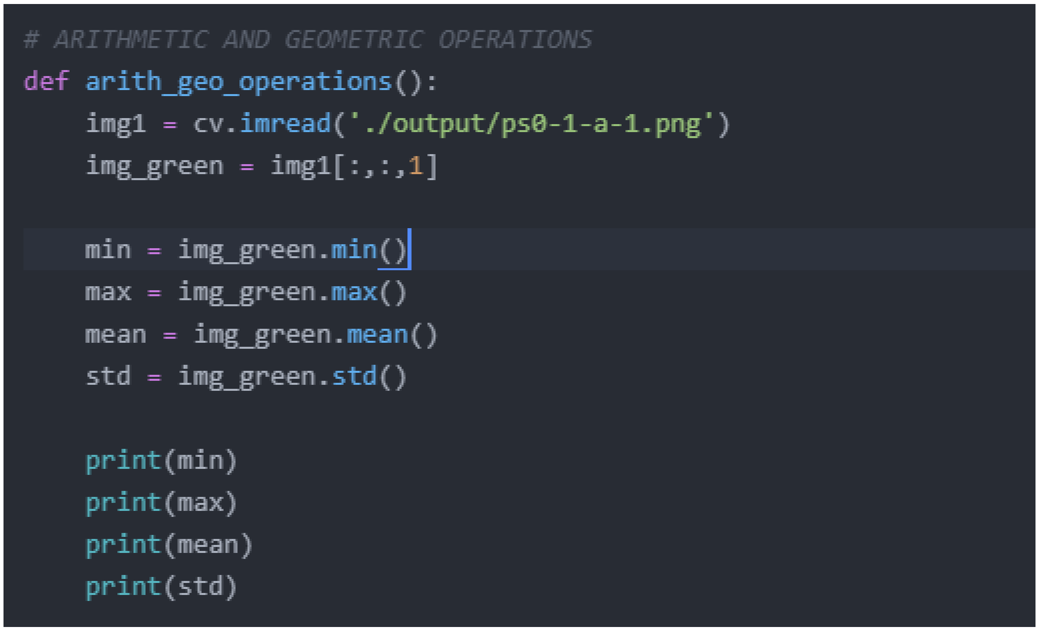
NOTES ABOUT RUNNING CODE

My code should run each part automatically (i.e. just by typing python ps0.py). If you'd like, you can comment out any of the individual sections on Lines 146-149.

TEXT RESPONSES AND CODE SNIPPETS

2d. The green channel image seems more aligned with my expectations of a monochrome image, since it has values that are spread more accurately from 0 to 255. I would expect computer vision algorithms having an easier time working with this channel, so this is the one that I have used for the next section.

4a. min = 1, max = 255, mean = 66.36, standard deviation = 51.07. I used the numpy functions on the green channel arrays to compute these values (code snippet is below).



```python
# ARITHMETIC AND GEOMETRIC OPERATIONS
def arith_geo_operations():
    img1 = cv.imread('./output/ps0-1-a-1.png')
    img_green = img1[:,:,1]

    min = img_green.min()
    max = img_green.max()
    mean = img_green.mean()
    std = img_green.std()

    print(min)
    print(max)
    print(mean)
    print(std)
```

4b. The image histogram is a representation of the frequencies of pixel intensity throughout the entire peppers image.

4d. Doing all of these operations has normalized the second image around the mean, so now every pixel's value is closer to the mean. There are very few outlier pixels now, to the point that they don't even register as significant enough for the histogram.

4f. The negative pixel values in the difference mean that the image is getting brighter as you move from left to right.

5a. I started to notice the noise at a sigma value of 0.5.

5b. The histograms are very similar because of the sheer amount of pixels in the image. The noise pixels are visible, but they don't detract from the overall image. I can see a slight uptick in the [0, 16] bin, for example, but it is a very small change.

5d. I would say the green channel image looks better, in that the Gaussian noise seems more normal. The white pixels in the blue channel image stick out much more clearly than in the green image.

5e. The median filter has smoothed everything out towards the median value, so it has mitigated the noise present in the noisy image.

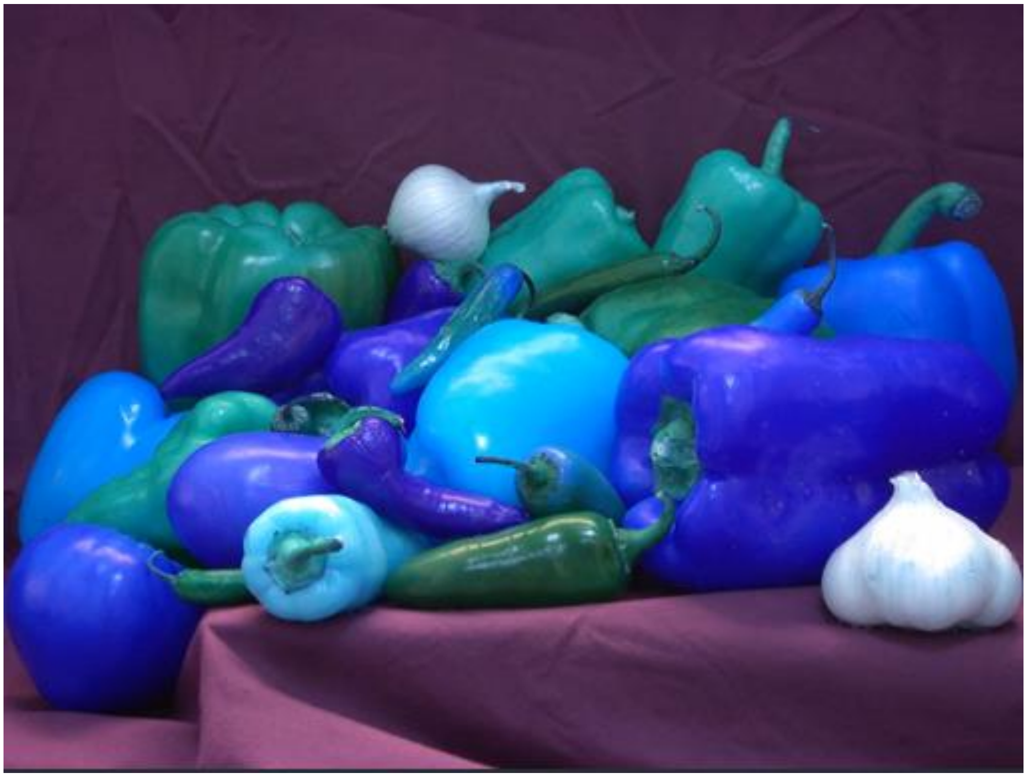
5f. The Gaussian filter has smoothed out the image, making it appear more blurry in comparison to the noisy image. However, the median filter has done a better job of preserving the edges of the objects in the image. For both the green and blue channel, the Gaussian filter seems to blur more than the median filter. For this specific image, it seems that the median filter is better since it does not blur (if that's what the goal is for this filter).

IMAGE FILES

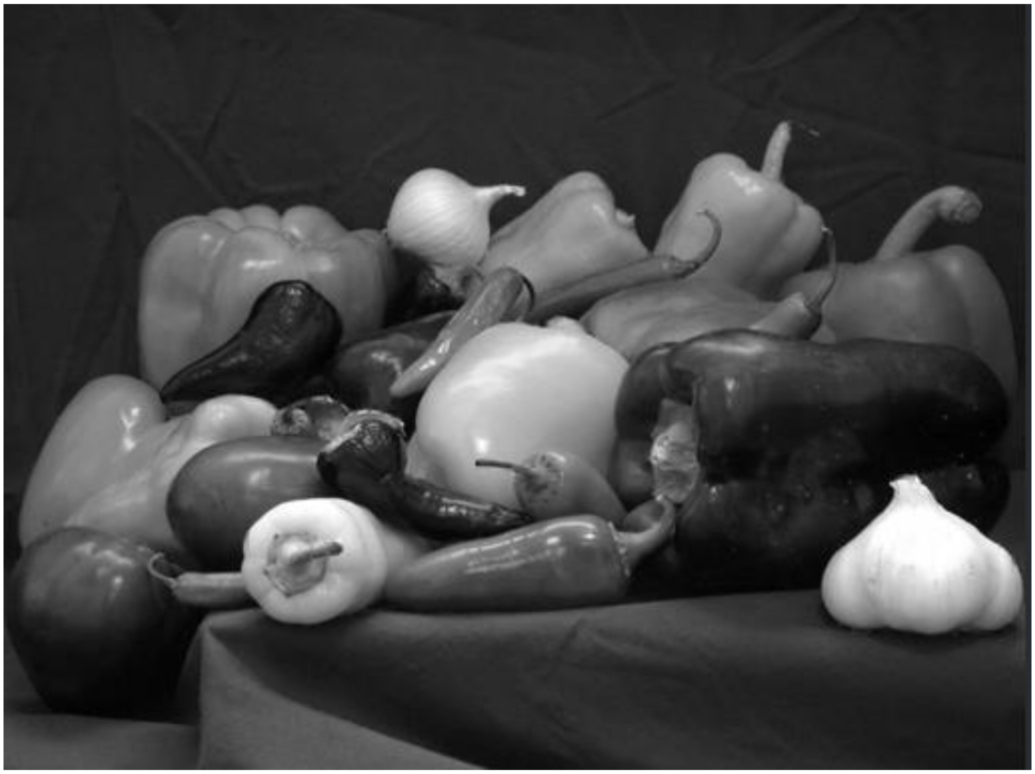
Ps0-1-a-1.png

Ps0-1-a-2.png

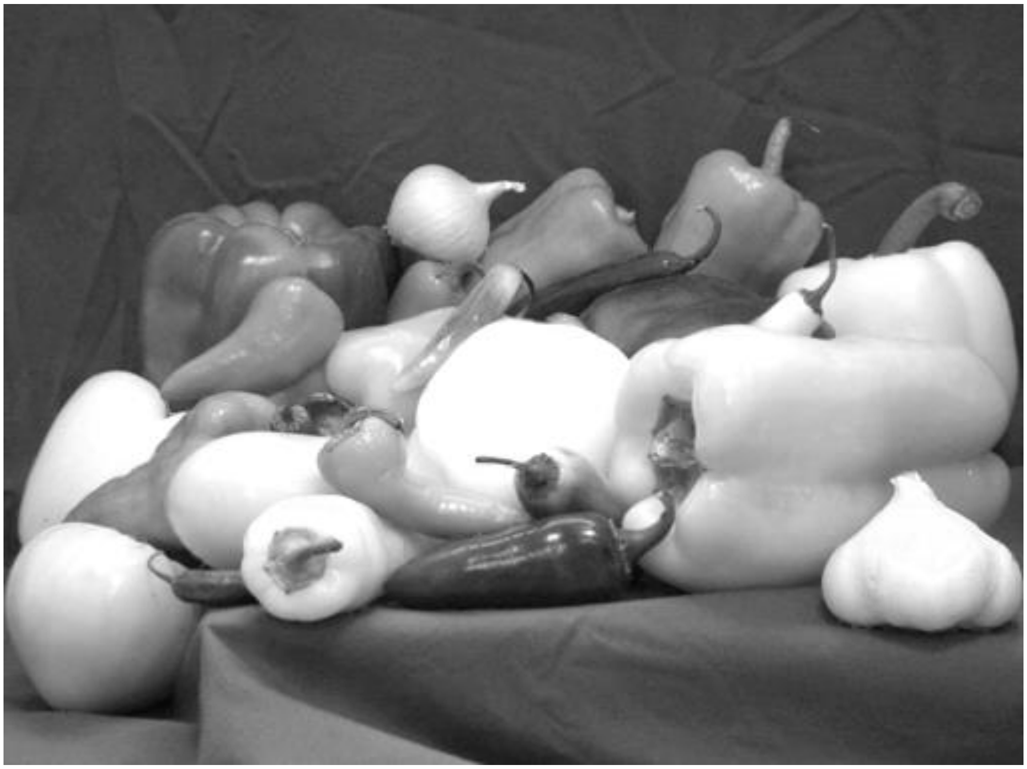
Ps0-2-a-1.png



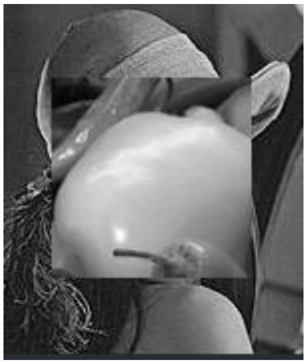
Ps0-2-b-1.png



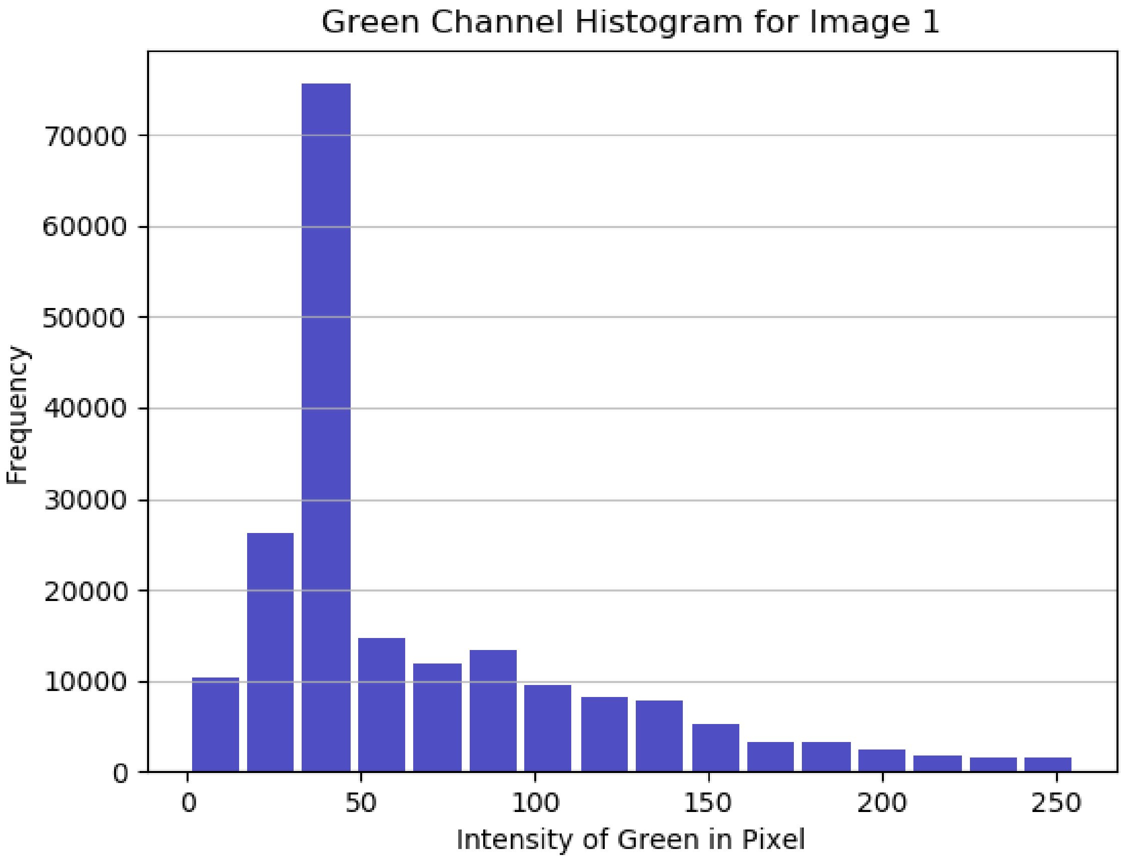
Ps0-2-c-1.png



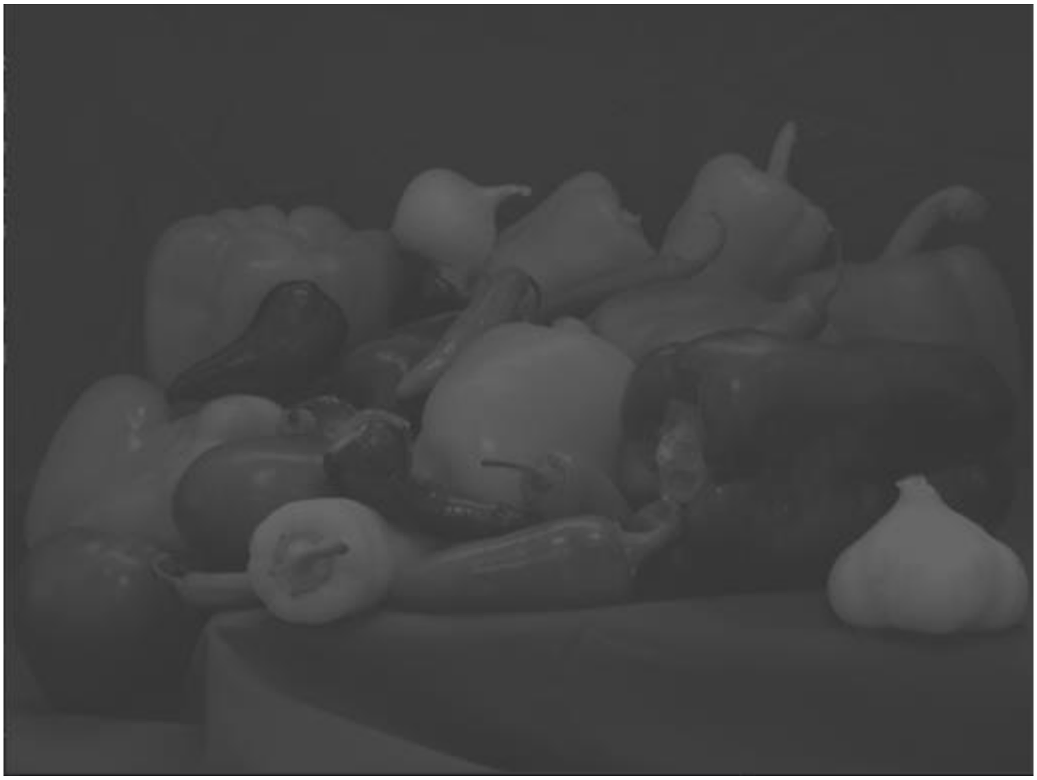
Ps0-3-a-1.png



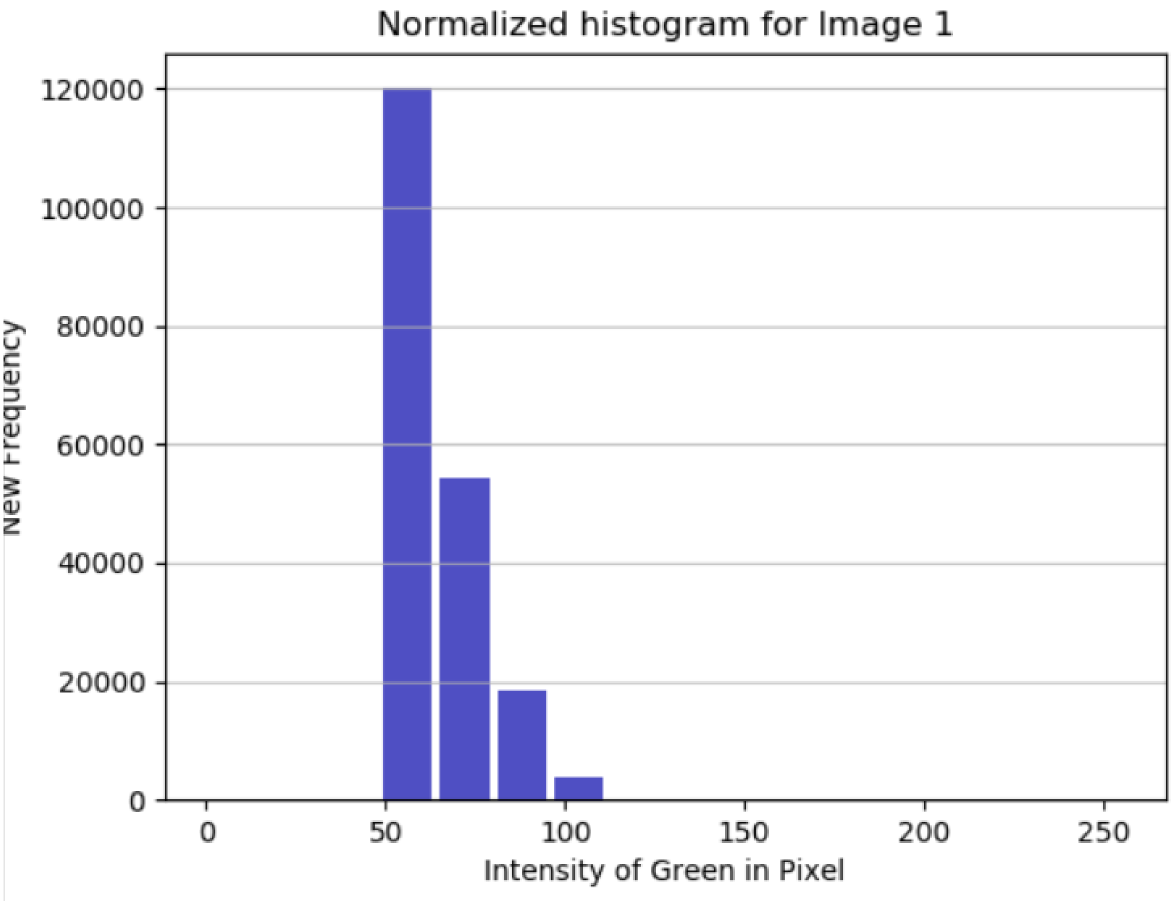
Ps0-4-b-1.png



Ps0-4-c-1.png

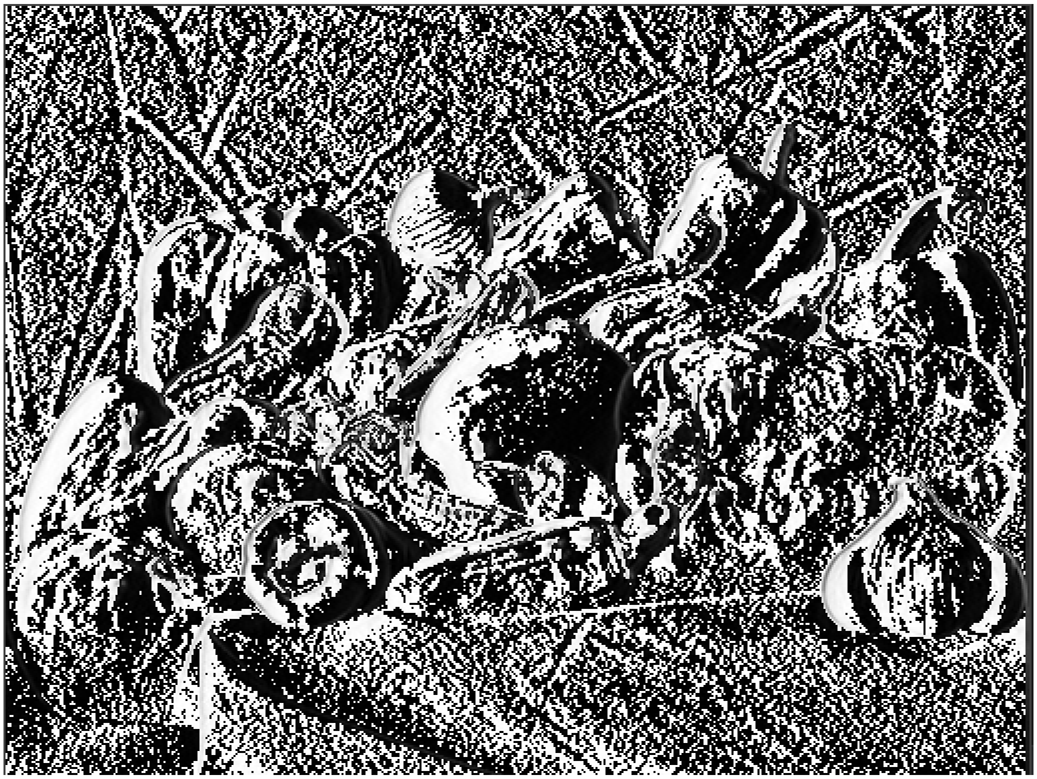


Ps0-4-d-1.png

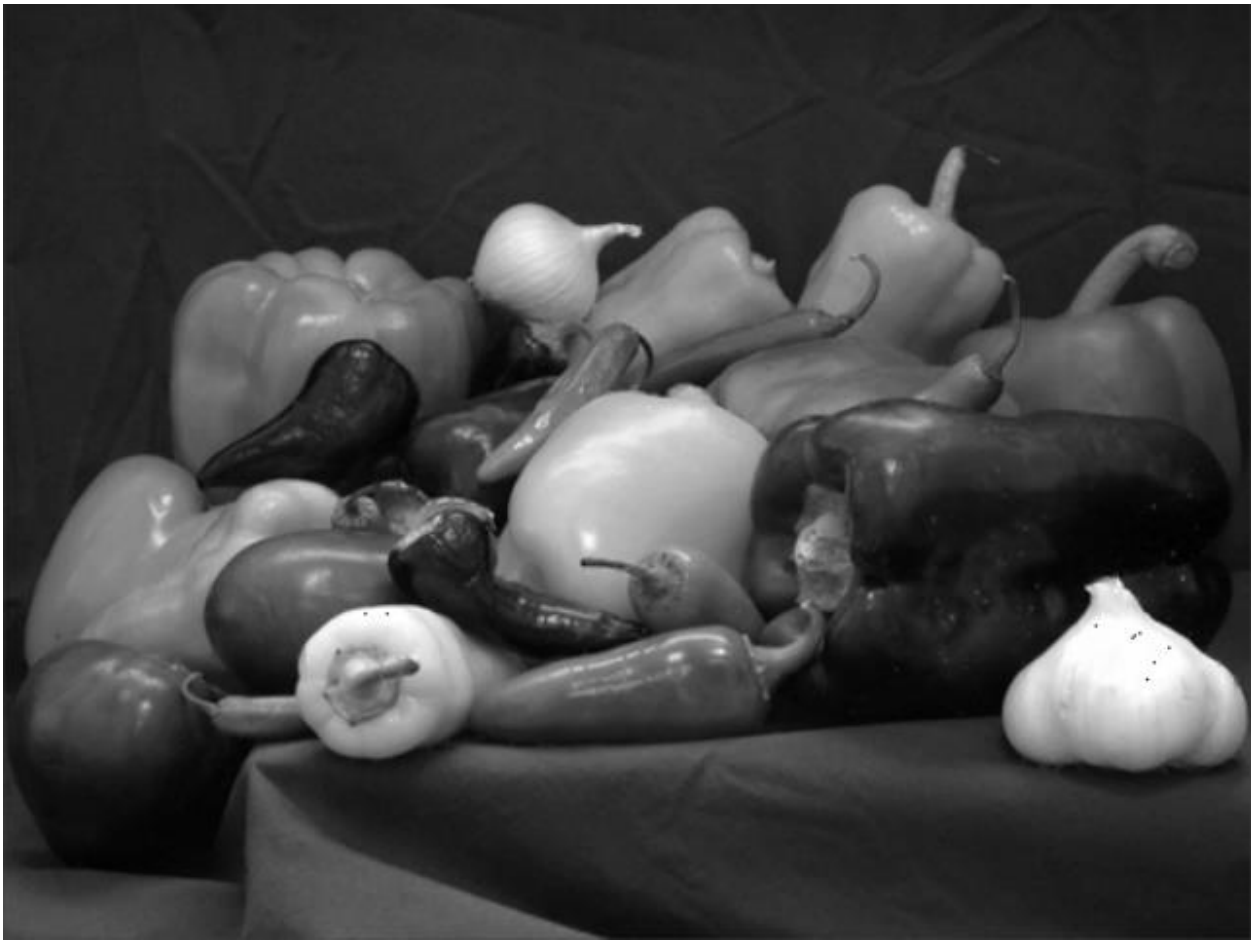


Ps0-4-e-1.png (notice small black sliver in two rightmost columns – I put it in front of a Notepad window to highlight the difference, the gray at the bottom and right are not part of the image)

Ps0-4-f-1.png



Ps0-5-a-1.png



Ps0-5-b-1.png

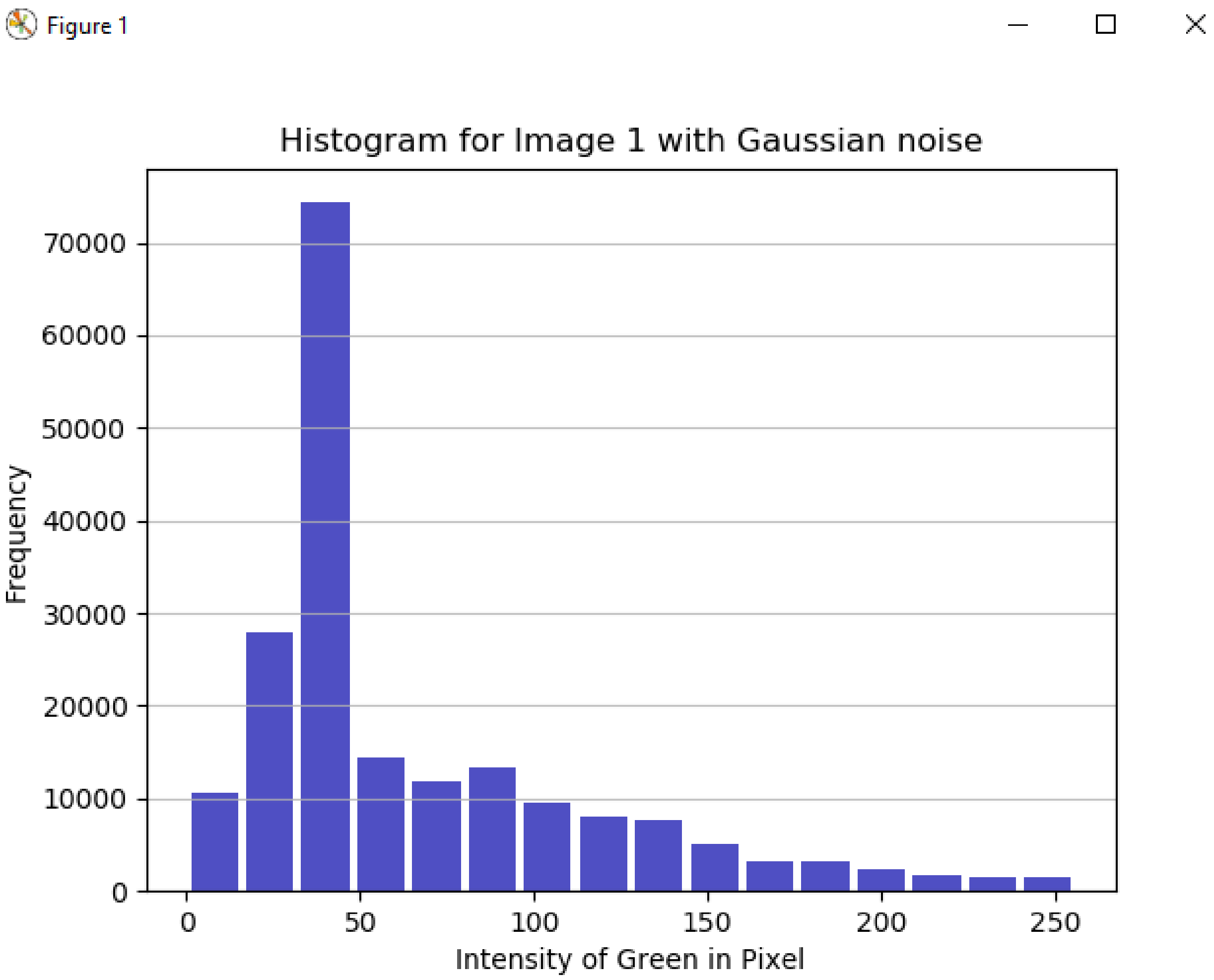
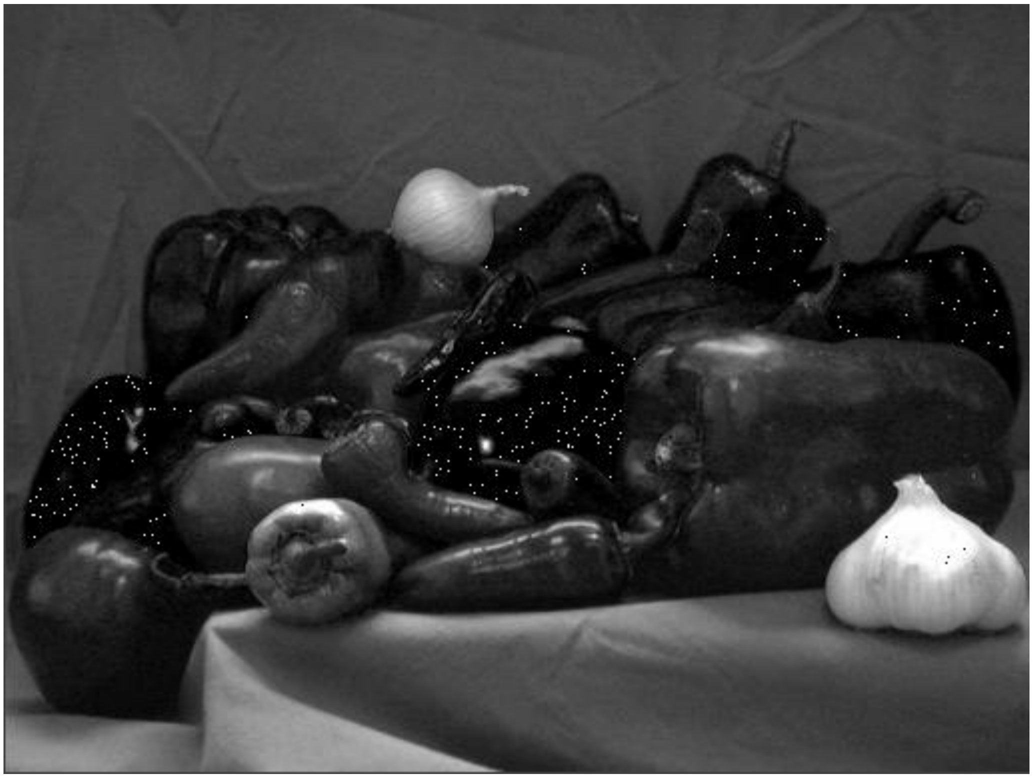
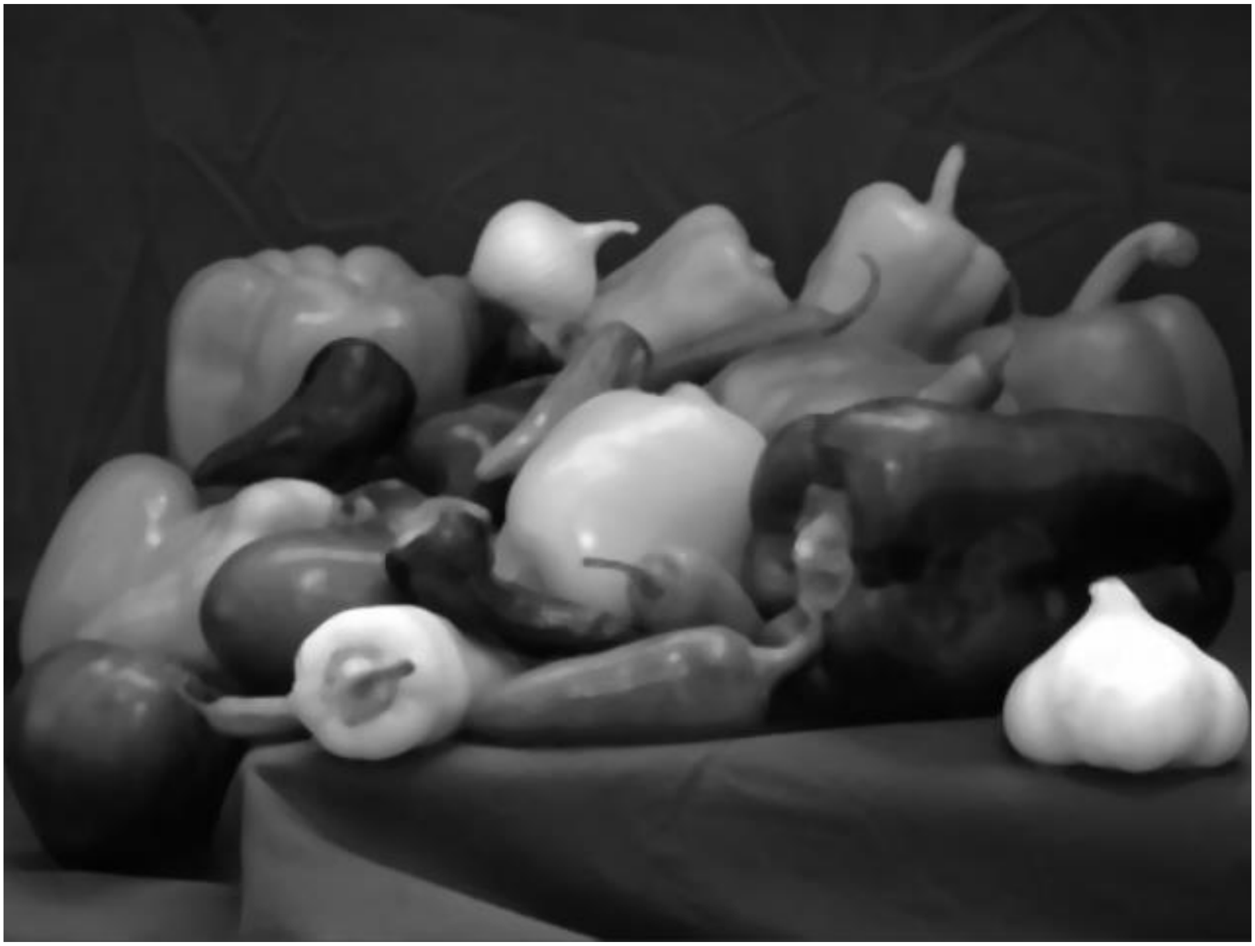


Figure 1
Histogram for Image 1 with Gaussian noise
Frequency
70000
60000
50000
40000
30000
20000
10000
0
0
50
100
150
200
250
Intensity of Green in Pixel

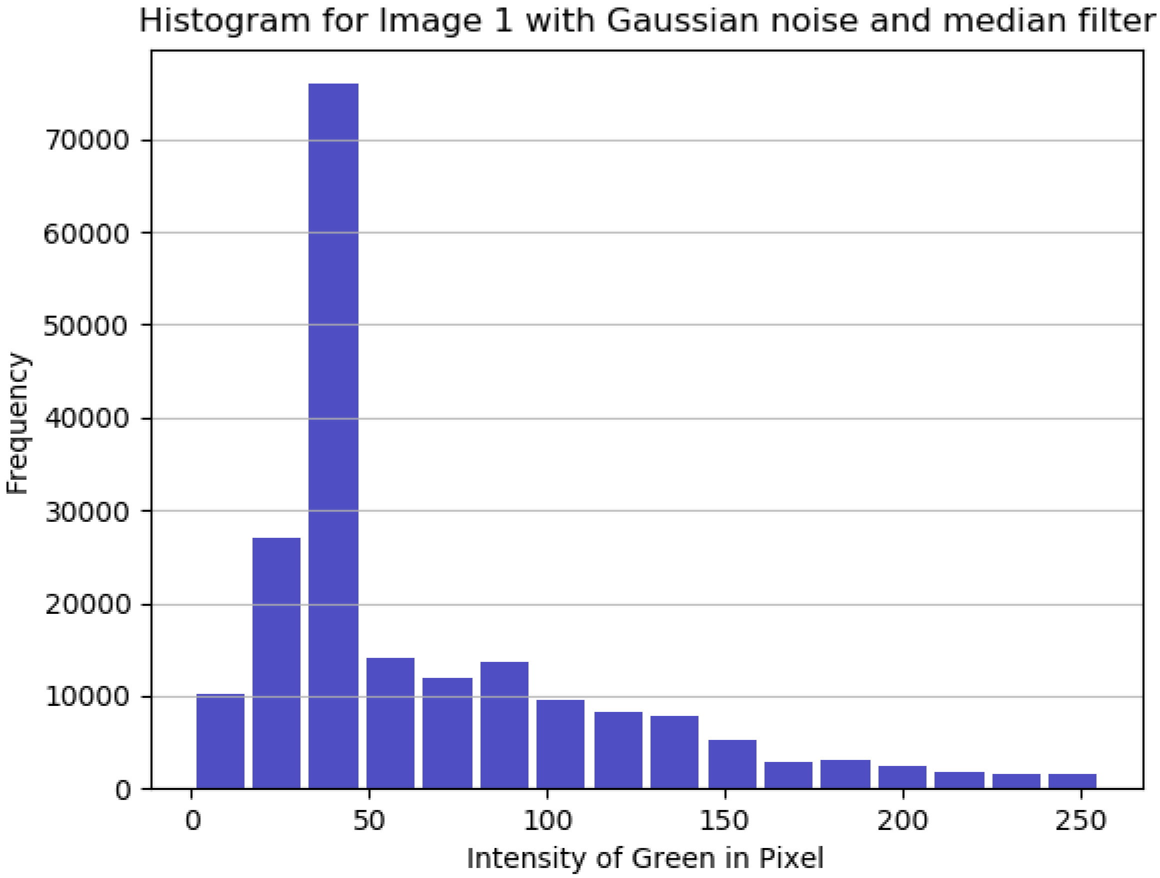
Ps0-5-c-1.png



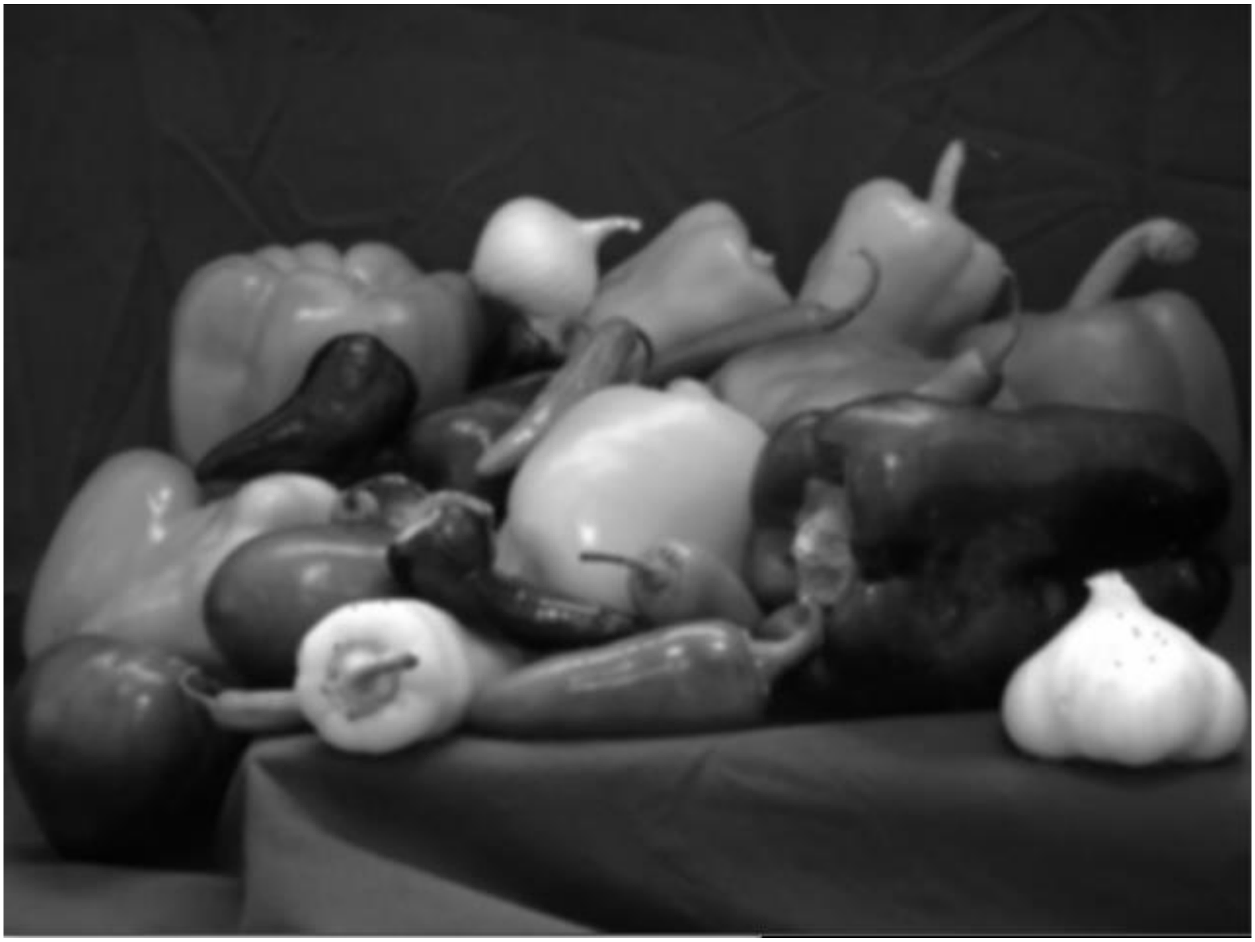
Ps0-5-e-1.png



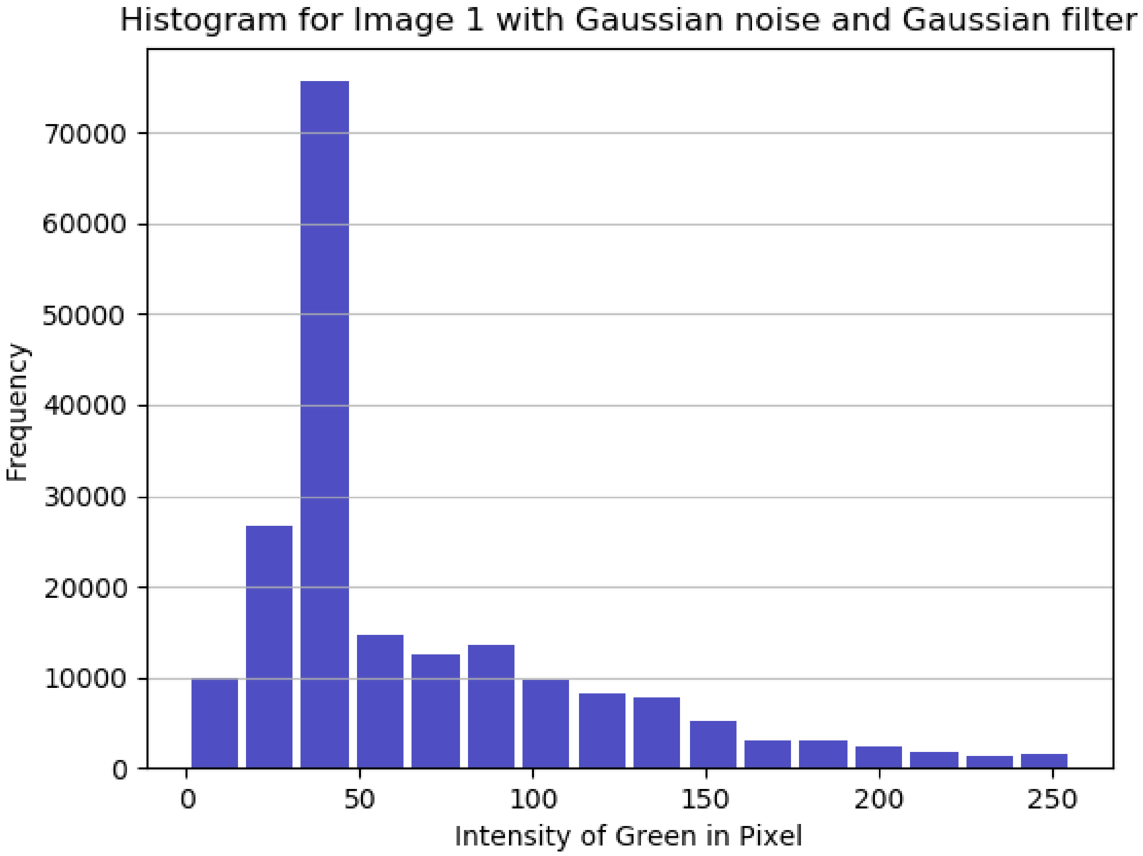
Ps0-5-e-2.png


Histogram for Image 1 with Gaussian noise and median filter
Frequency
0
10000
20000
30000
40000
50000
60000
70000
0
50
100
150
200
250
Intensity of Green in Pixel

Ps0-5-f-1.png



Ps0-5-f-2.png


Histogram for Image 1 with Gaussian noise and Gaussian filter
Frequency
70000
60000
50000
40000
30000
20000
10000
0
0
50
100
150
200
250
Intensity of Green in Pixel