Shirong Zheng

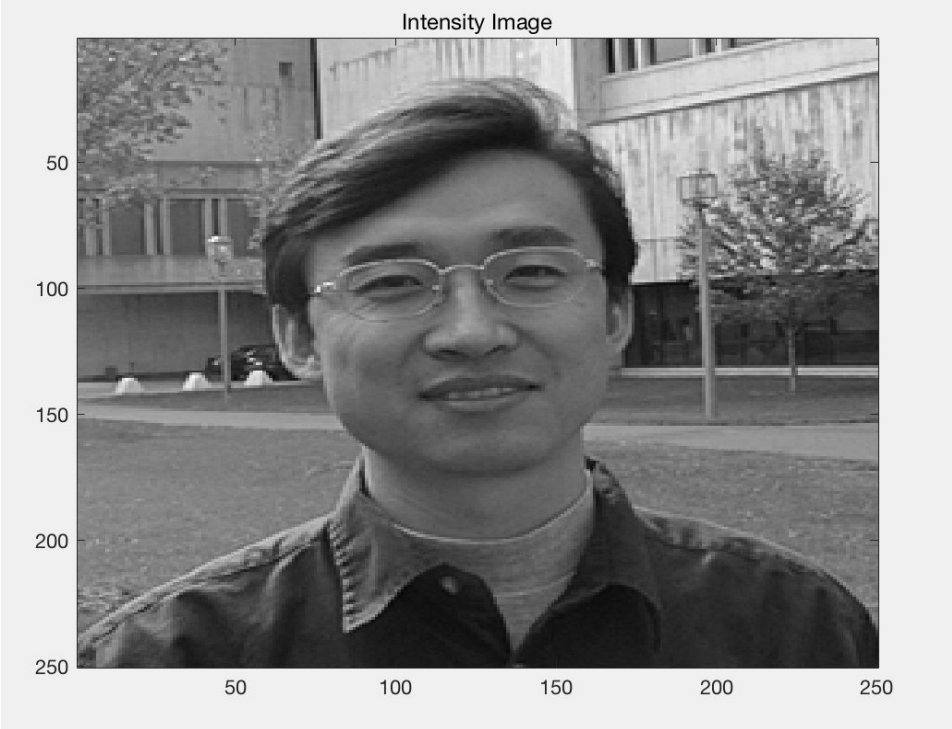
CSC47100-E

Prof. Zhu

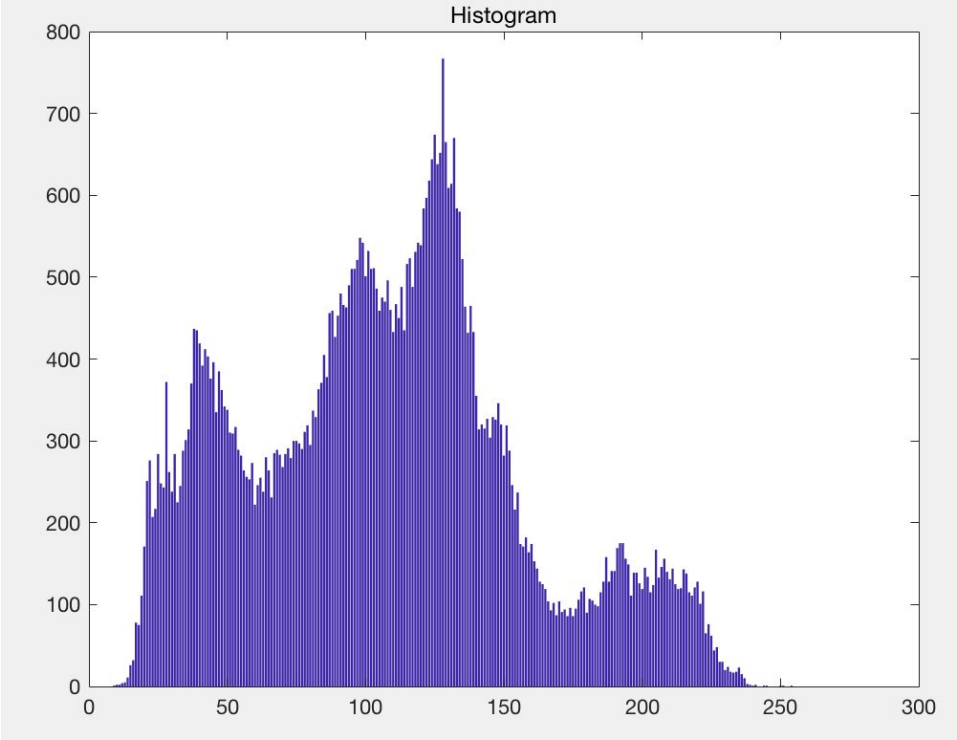
10/13/2017

Assignment 2

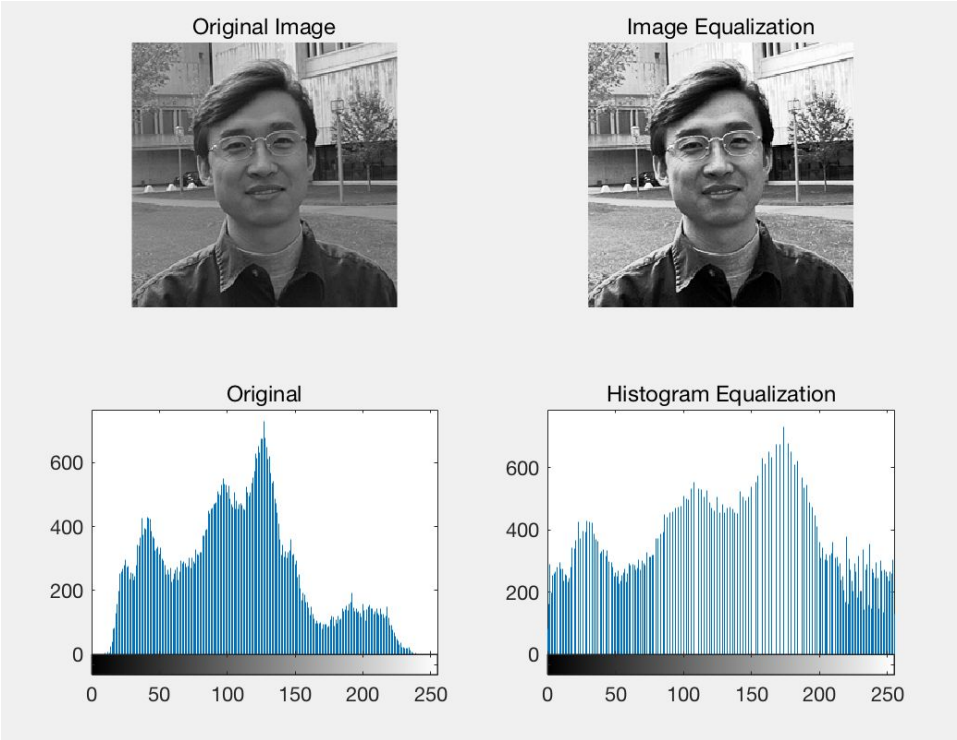
1. (20 points) Generate the histogram of the image you are using, and then perform a number of histogram operations (such as contrast enhancement, thresholding and equalization) to make the image visually better for either viewing or processing (10 points). If it is a color image, please first turn it into an intensity image and then generate its histogram. Try to display your histogram (5 points), and make some observations of the image based on its histogram (5 points). What are the general distributions of the intensity values? How many major peaks and valleys does your histogram have? How could you use the histogram to understand, analyze or segment the image? Please also display the histograms of the processed images and provide a few important observations.



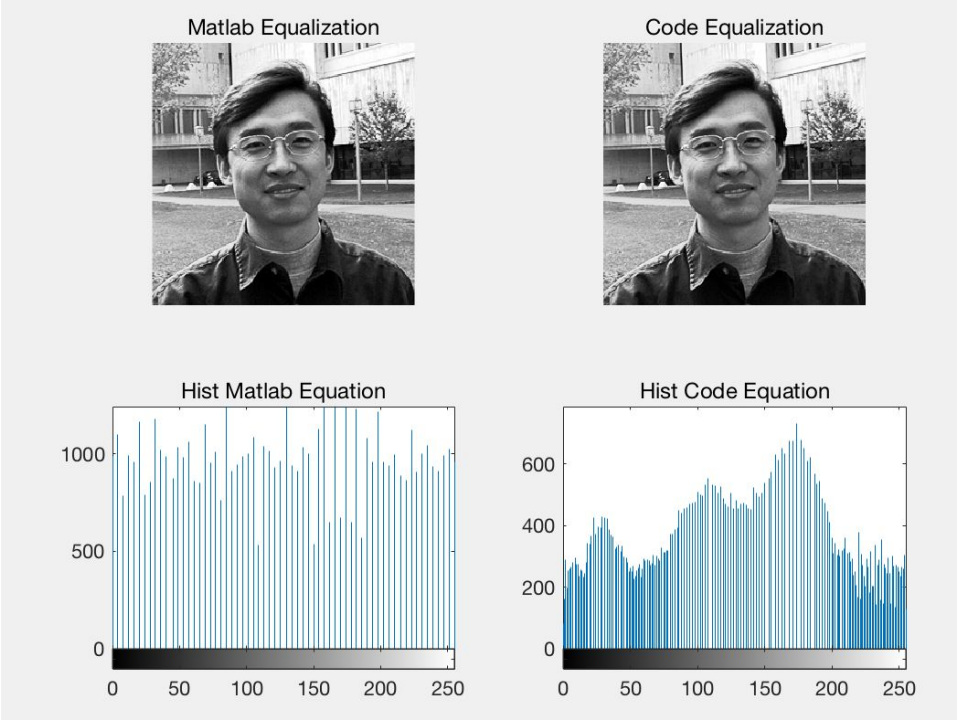
In this Intensity Image, I just find out RGB then use function
Intensity=0.2999*CR1+0.587*CG1+0.114*CB1



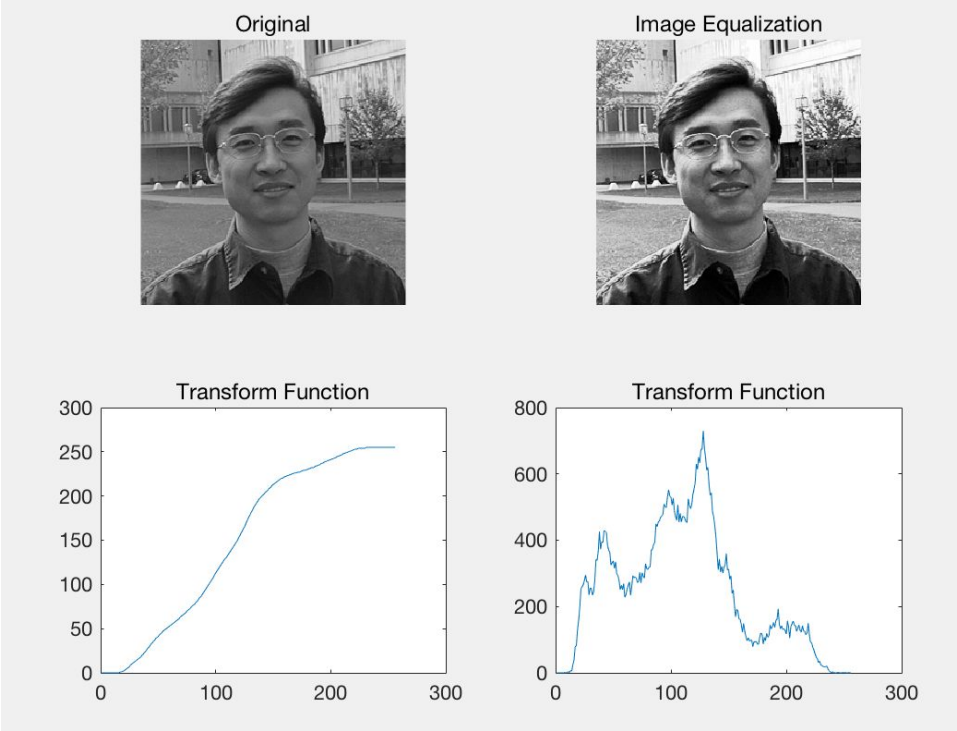
The histogram bar of the intensity image. This diagram also shows the general distributions of the intensity values. The major peak is between 800 ~ 700, the major valley is almost down to 0. As you can see, most shadow is appear on left side of the middle.



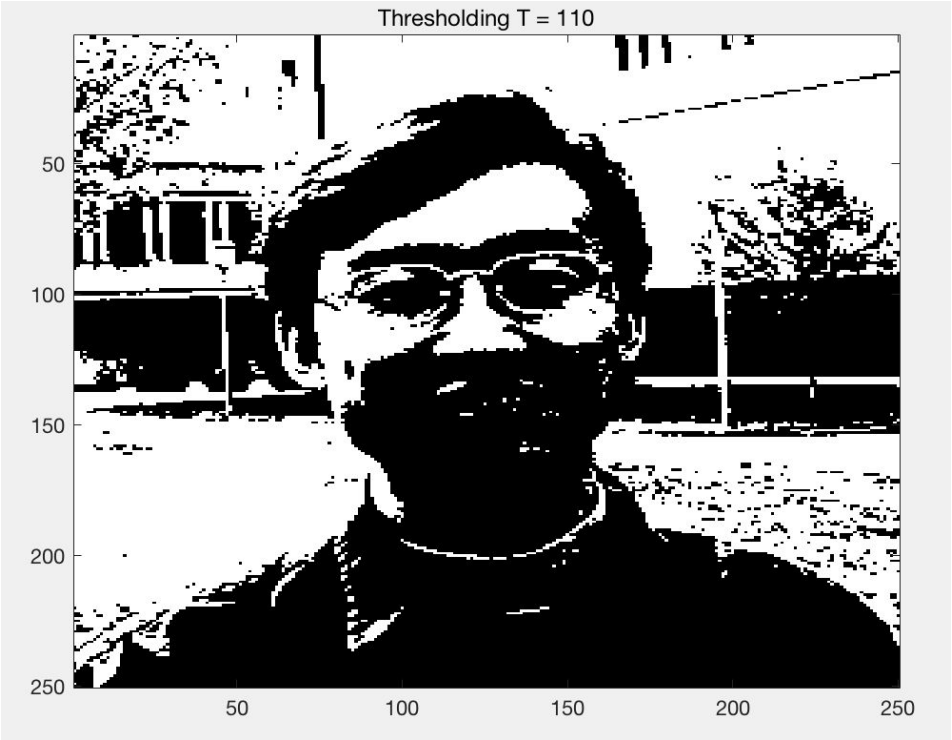
Then it perform a number of histogram operations. In this case, it is histogram equalization. The central idea of histogram equalization is to change the gray histogram of the original image from a relatively concentrated gray scale to a uniform distribution over the entire gray scale. The image will become more bright.



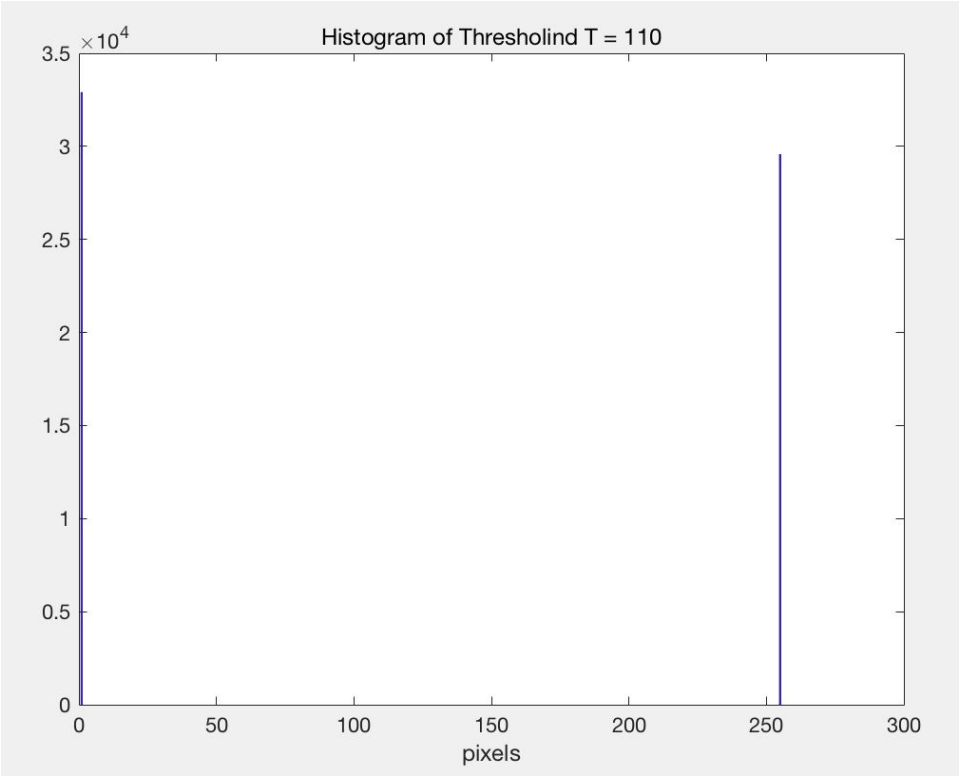
Compare the equalization build in function in matlab and my code equation, it demonstrate my code equation is most close to the original intensity histogram.



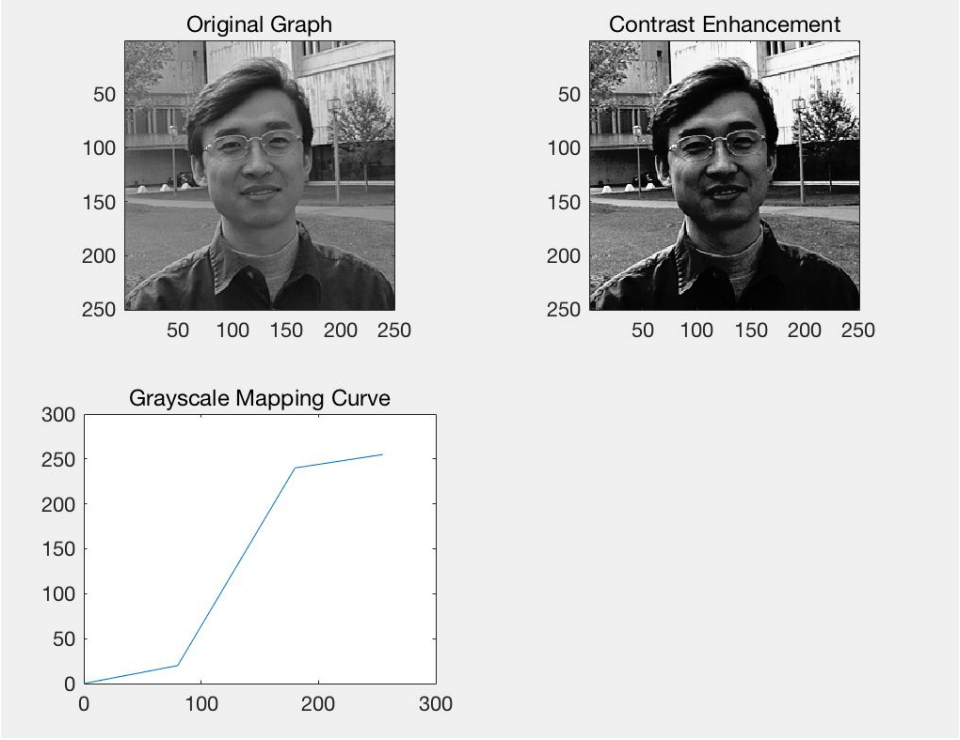
Rather than transform function of original, the transform function of equalization image is similar to histogram of intensity image.



Threshold conversion method to grayscale image into a binary image. Generally refers only to pure black (0), pure white (255) of the image. In this case, we set the value of threshold value(T) to 110.

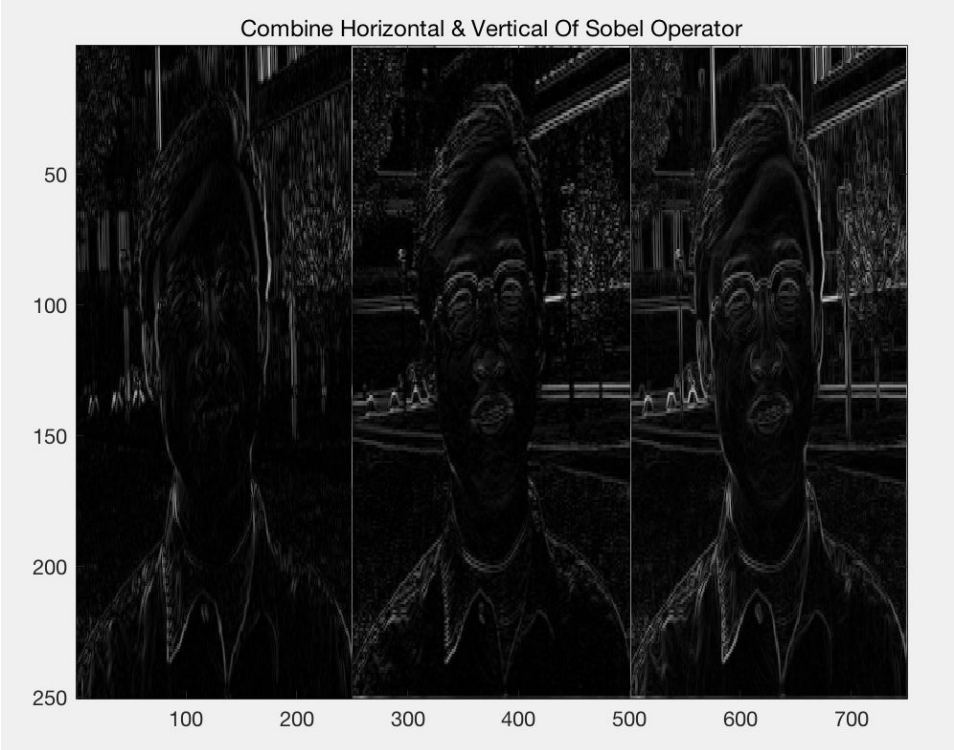


The pixels of threshold is 250.



For the gray point, it do mapping in the image, making the overall image of the gray scale roughly consistent with the uniform distribution. That enhance the contrast of the image.

2. (20 points) Apply the 1x2 operator and Sobel operator to your image and analyze the results of the gradient magnitude images (including vertical gradients, horizontal gradients, and the combined) (10 points). Please don't forget to normalize your gradient images, noting that the original vertical and horizontal gradients have both positive and negative values. I would recommend you to display the absolute values of the horizontal and vertical gradient images. Does the Sobel operator have any clear visual advantages over the 1x2 operator? Any disadvantages (5 points)? If you subtract the 1x2 edge image from the Sobel are there any residuals? You might use two different types of images: one ideal man-made image, and one image of a real scene with more details (5 points). (Note: don't forget to normalize your results as shown in slide # 29 of feature extraction lecture: part 2)



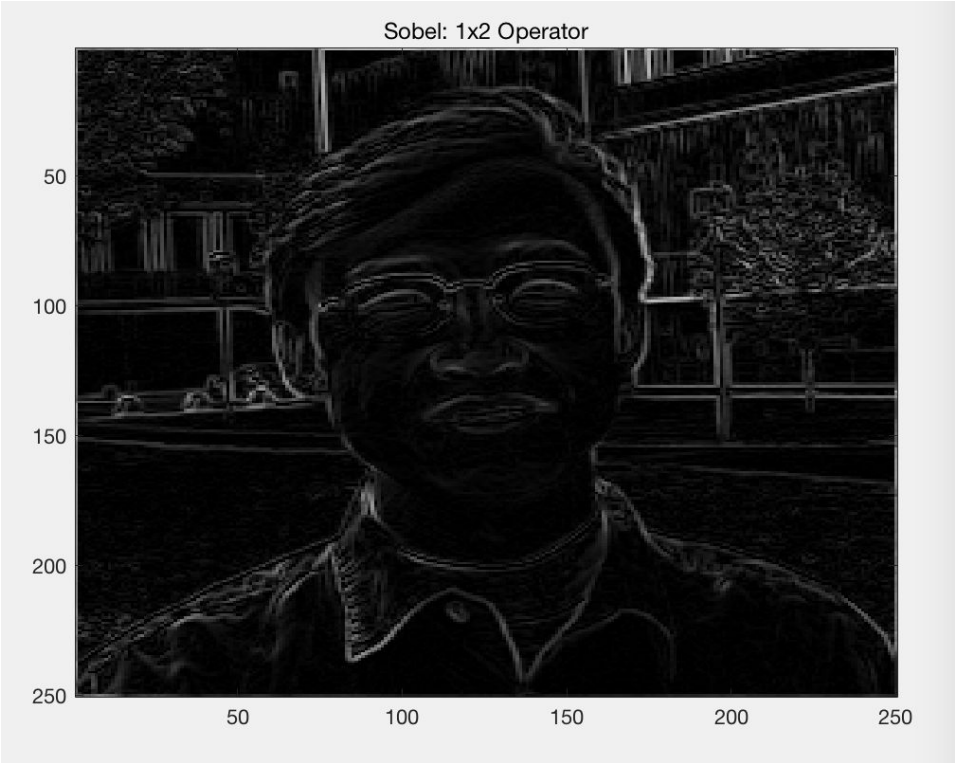
Compare these two types of images, sobel operator is more darker than 1x2 operator. In other side, it could said the sobel operator have clear visual advantages over the 1x2 operator. For 1x2 it is 1, and for sobel it is 4=1+2+1. Usually we use the basic two 3x3 kernels to find out Gx and Gy

$$\mathbf{G}_x = \begin{bmatrix} +1 & 0 & -1 \\ +2 & 0 & -2 \\ +1 & 0 & -1 \end{bmatrix} * \mathbf{A} \quad \text{and} \quad \mathbf{G}_y = \begin{bmatrix} +1 & +2 & +1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix} * \mathbf{A}$$

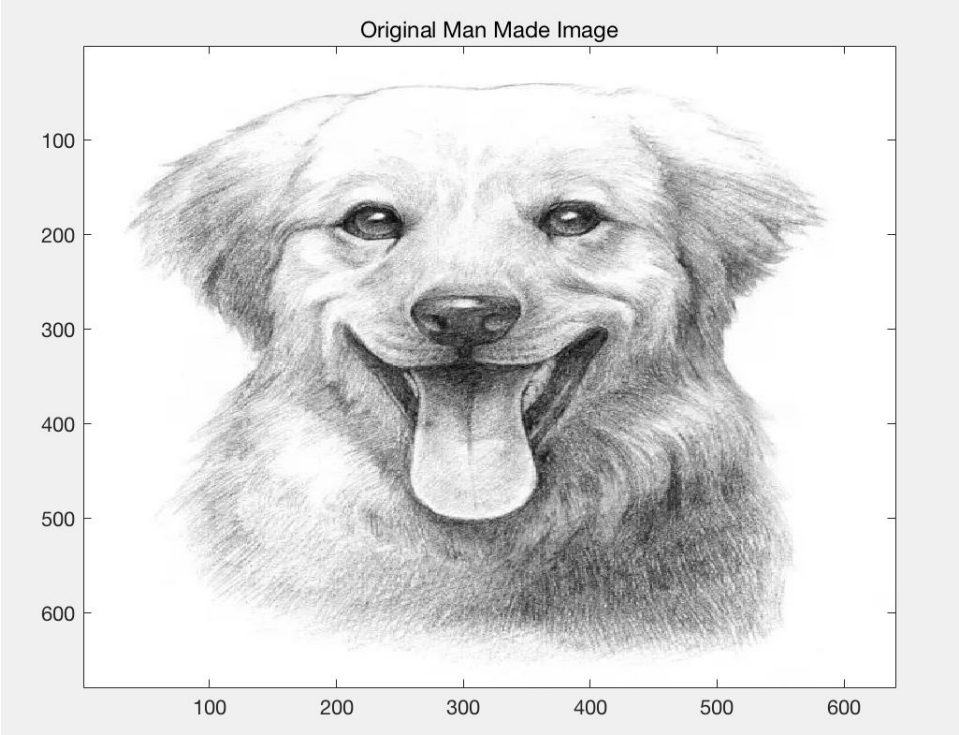
To combined to give the gradient magnitude, using:

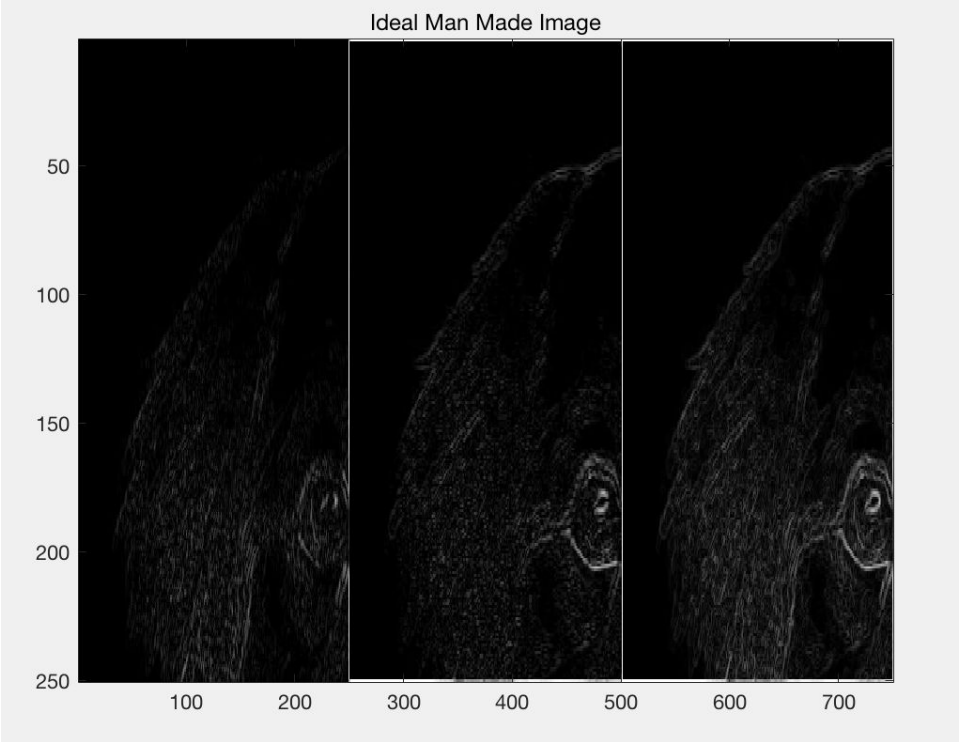
$$\mathbf{G} = \sqrt{\mathbf{G}_x{}^2 + \mathbf{G}_y{}^2}$$

For 1x2 operator the normalizing factor is 1 so it is already normalized. The normalizing factor is the number of pixels which do summation. In this process, I do normaliza gradient of sobel operator.



When I do subtract the 1x2 edge image from the sobel operator, there is no residual and it is look likes exactly to sobel operator.



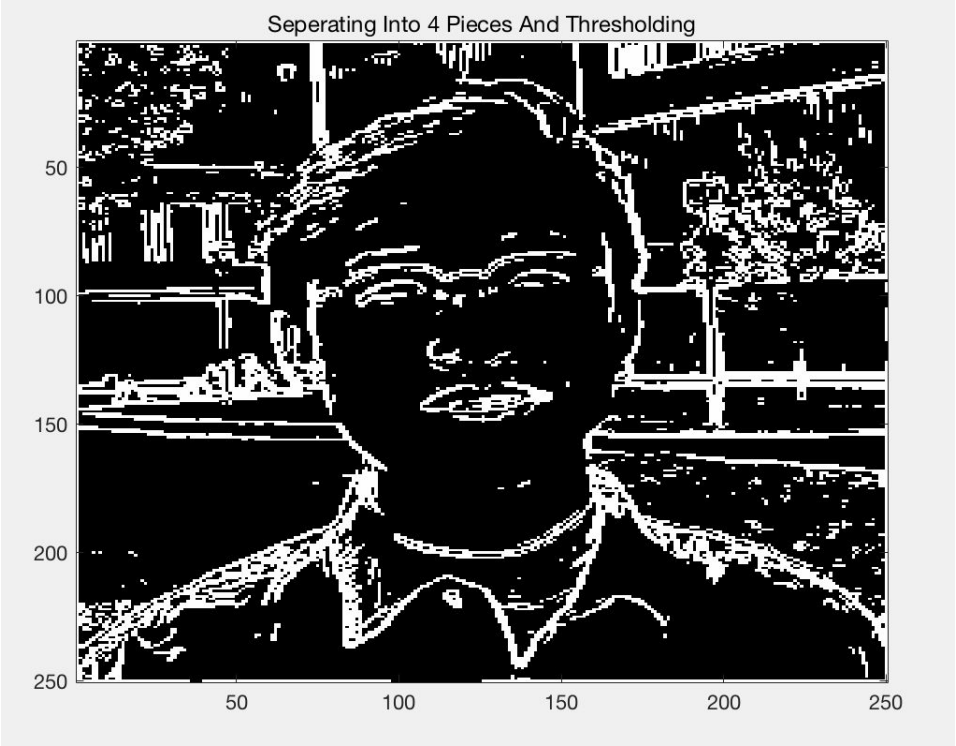


In this part, I did apply sobel operator to a ream scene image. Then sobel will exhaustive change the shape of original image.

3. (20 points) Generate edge maps of the above two combined gradient maps (10 points). An edge image should be a binary image with 1s as edge points and 0s as non-edge points. You may first generate a histogram of each gradient map, and only keep certain percentage of pixels (e.g. 5% of the pixels with the highest gradient values) as edge pixels (edgels) . Use the percentage to automatically find a threshold for the gradient magnitudes. In your report, please write up the description and probably equations for finding the threshold, and discuss if 5% is a good value. If not what is (5 points) ? You may also consider to use local, adaptive thresholds to different portions of the image so that all major edges will be shown up nicely (5 points). In the end, please try to generate a sketch of an image, such as the ID image of Prof. Zhu.

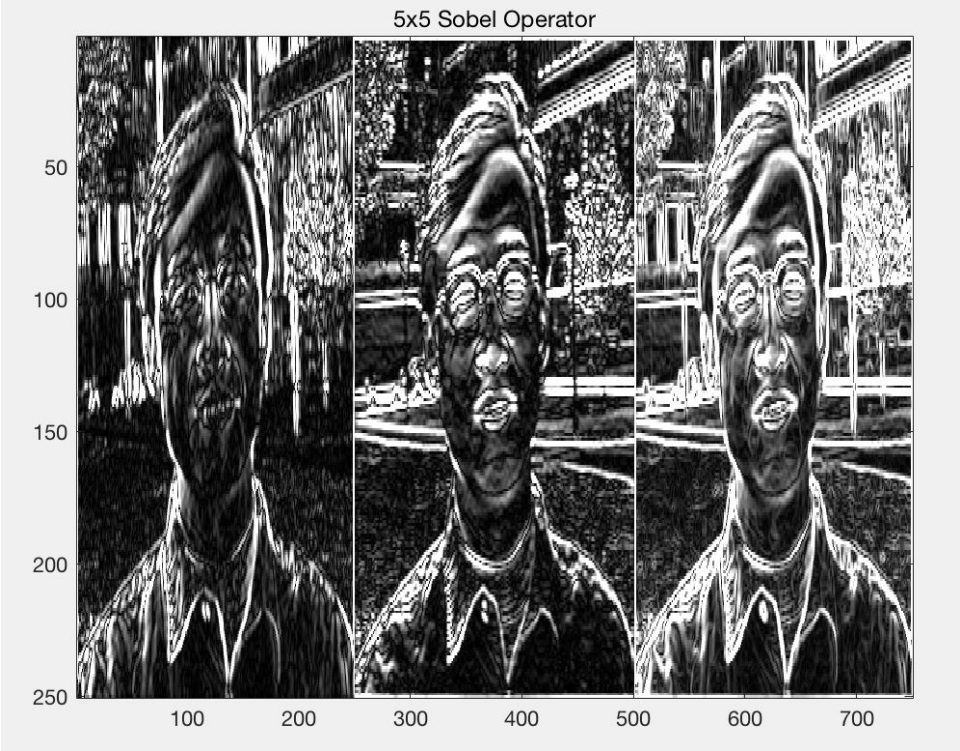
It would be used this function to find out the threshold :
histo(image(i,j)+1)= histo(image(i,j)+1) + 1

5% is a good value of the pixels with the highest gradient values as edge pixels. Because it is the average pixels of gradient values.

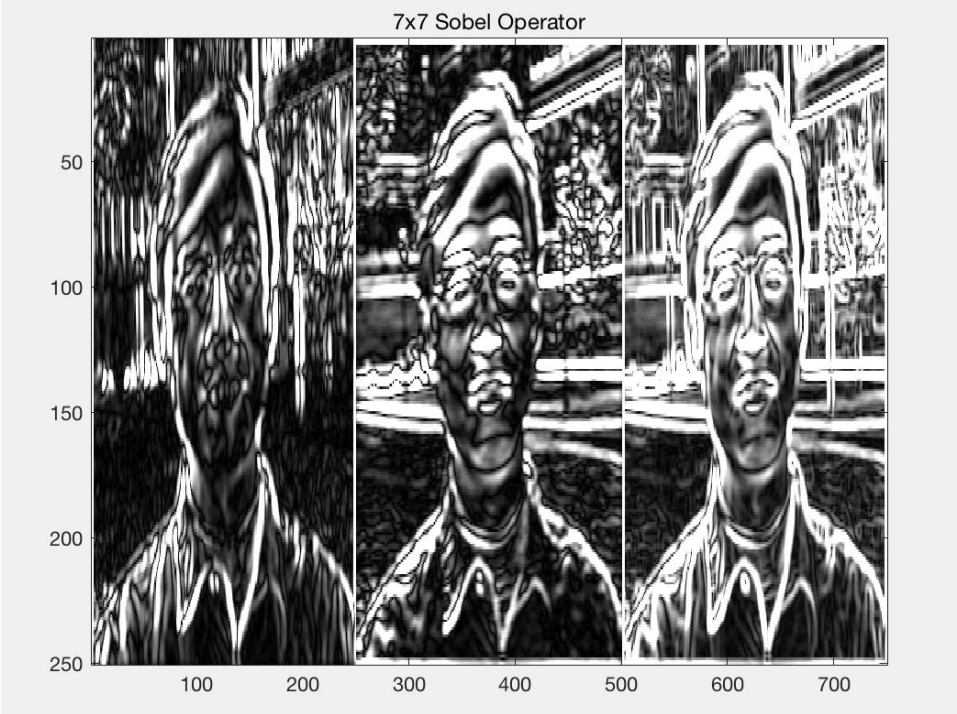


The combine sobela divide into 4 pieces. The value of piece 1 is (1:125, 1:125); the value of piece 2 is (126:250, 1:125); the value of piece 3 is (1:125, 126:250); the value of piece 4 is (126:250, 126:250).

4. (20 points) What happens when you increase the size of the edge detection kernel from 1x2 to 3x3 and then to 5x5 , or 7x7? Discuss computational cost (in terms of members of operations, and the real machine running times - 5 points), edge detection results (5 points) and sensitivity to noise, etc. (5 points). Note that your larger kernel should still be an edge detector. Please list your kernels as matrices in your report, and tell us what they are good for (5 points).



Gx5 = [1,2,0,-2,-1;4,8,0,-8,-4;6,12,0,-12,-6;4,8,0,-8,-4;1,2,0,-2,-1];
Gy5 = [-1,-4,-6,-4,-1; -2, -8, -12, -8, -2;0,0,0,0,0;2,8,12,8,2;1,4,6,4,1];



Gx7 =
[3,2,1,0,-1,-2,-3;4,3,2,0,-2,-3,-4;5,4,3,0,-3,-4,-5;6,5,4,0,-4,-5,-6;5,4,3,0,-3,-4,-5;4,3,2,0,-2,-3,-4;3,2,1,0,-1,-2,-3];
Gy7 =
[-3,-4,-5,-6,-5,-4,-3;-2,-3,-4,-5,-4,-3,-2;-1,-2,-3,-4,-3,-2,-1;0,0,0,0,0,0,0;1,2,3,4,3,2,1;2,3,4,5,4,3,2;,3,4,5,6,5,4,3];

Where I increase normal size 3x3 kernel to 5x5 and 7x7 kernel, that the image will be more bright and sharpening. Then you could clearly see the shape of the object. The larger size kernet will produce more bright image as the above shown.
The running time for 5x5 and 7x7 kernel will be O(n^2).

5. (20 points) Suppose you apply the Sobel operator to each of the RGB color bands of a color image. How might you combine these results into a color edge detector (5 points)? Do the resulting edge differ from the gray scale results? How and why (5 points)? You may compare the edge maps of the intensity image (of the color image), the gray-scale edge map that are the combination of the three edge maps from three color bands, or a real color edge map that edge points have colors (5 points). Please discuss their similarities and differences, and how each of them can be used for image enhancement or feature extraction (5 points). Note that you want to first generate gradient maps and then using thresholding to generate edge maps. In the end, please try to generate a color sketch of an image, such as the ID image of Prof. Zhu. You may also consider local, adaptive thresholding in generating a color edge map.

The sobel operator is very similar to Prewitt operator (detect edges using the Prewitt method). It is also a derivative mask and is used for edge detection. Like Prewitt operator sobel operator is also used to detect two kinds of edges in an image:

- Vertical direction
- Horizontal direction

Difference with Prewitt Operator

The major difference is that in sobel operator the coefficients of masks are not fixed and they can be adjusted according to our requirement unless they do not violate any property of derivative masks.

Following is the vertical Mask of Sobel Operator:

-1	0	1
-2	0	2
-1	0	1

This mask works exactly same as the Prewitt operator vertical mask. There is only one difference that is it has “2” and “-2” values in center of first and third column. When applied on an image this mask will highlight the vertical edges.

How it works

When we apply this mask on the image it prominent vertical edges. It simply works like as first order derivative and calculates the difference of pixel intensities in a edge region.

As the center column is of zero so it does not include the original values of an image but rather it calculates the difference of right and left pixel values around that edge. Also the center values of both the first and third column is 2 and -2 respectively.

This give more weightage to the pixel values around the edge region. This increase the edge intensity and it become enhanced comparatively to the original image.

Following is the horizontal Mask of Sobel Operator

-1	-2	-1
0	0	0
1	2	1

Above mask will find edges in horizontal direction and it is because that zeros column is in horizontal direction. When you will convolve this mask onto an image it would prominent horizontal edges in the image. The only difference between it is that it have 2 and -2 as a center element of first and third row.

How it works

This mask will prominent the horizontal edges in an image. It also works on the principle of above mask and calculates difference among the pixel intensities of a particular edge. As the center row of mask is consist of zeros so it does not include the original values of edge in the image but rather it calculate the difference of above and below pixel intensities of the particular edge. Thus increasing the sudden change of intensities and making the edge more visible.

Now it’s time to see these masks in action:

After applying Vertical Mask

After applying vertical mask on the above sample image, following image will be obtained.

After applying Horizontal Mask

After applying horizontal mask on the above sample image, following image will be obtained

Comparison

As you can see that in the first picture on which we apply vertical mask, all the vertical edges are more visible than the original image. Similarly in the second picture we have applied the horizontal mask and in result all the horizontal edges are visible.

So in this way you can see that we can detect both horizontal and vertical edges from an image. Also if you compare the result of sobel operator with Prewitt operator, you will find that sobel operator finds more edges or make edges more visible as compared to Prewitt Operator.

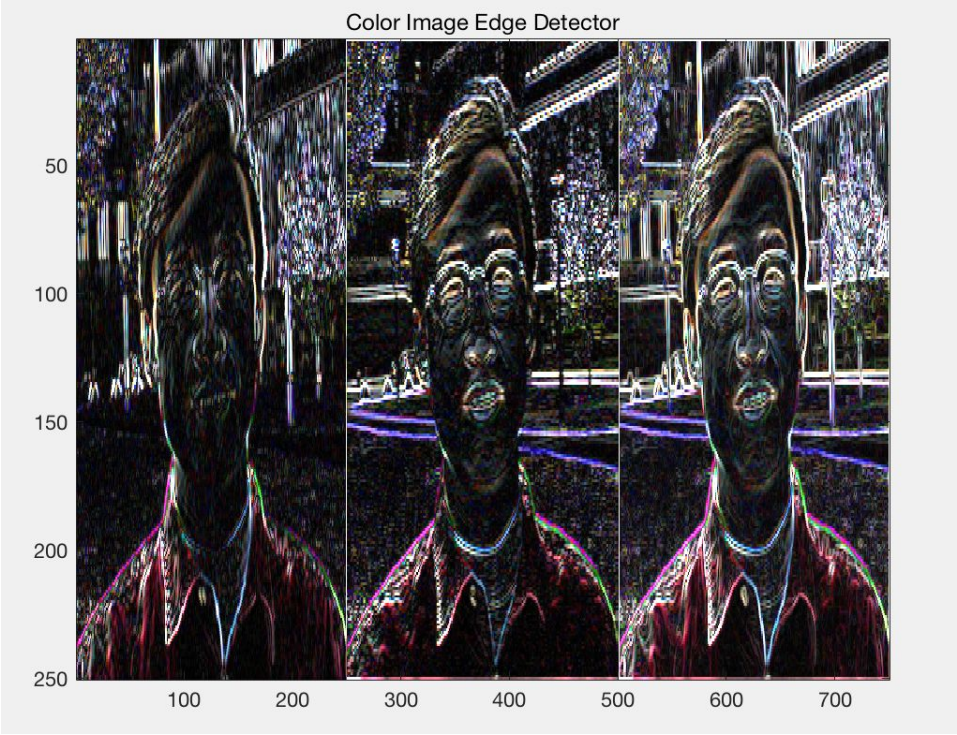
This is because in sobel operator we have allotted more weight to the pixel intensities around the edges.

Applying more weight to mask

Now we can also see that if we apply more weight to the mask, the more edges it will get for us. Also as mentioned in the start of the example that there is no fixed coefficients in sobel operator, so here is another weighted operator

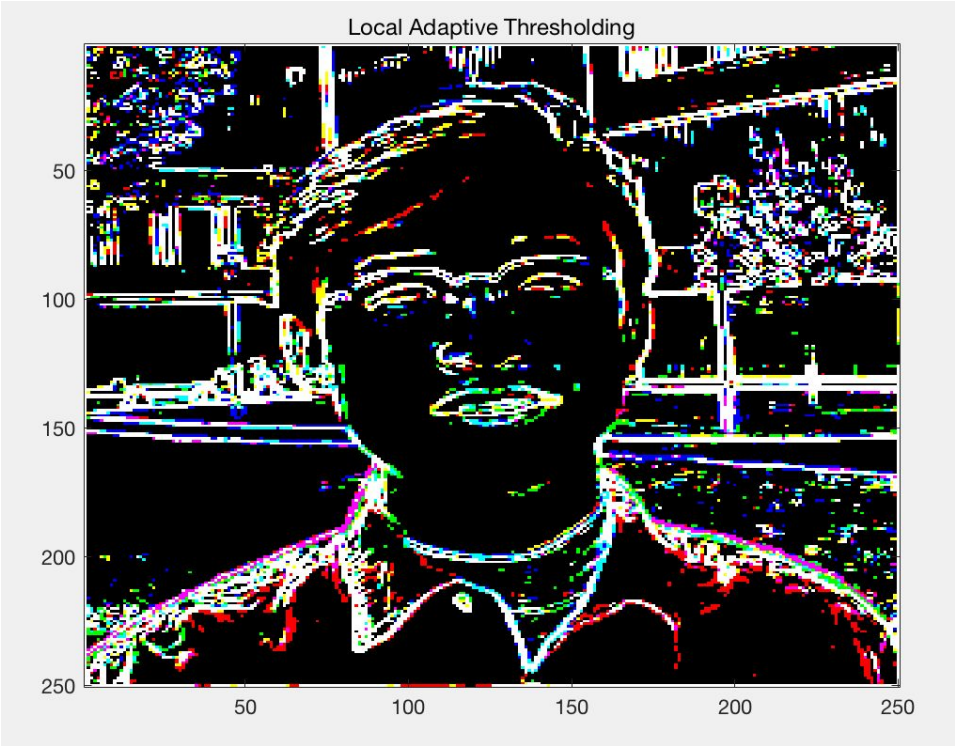
-1	0	1
-5	0	5
-1	0	1

If you can compare the result of this mask with of the Prewitt vertical mask, it is clear that this mask will give out more edges as compared to Prewitt one just because we have allotted more weight in the mask.


Color Image Edge Detector
50
100
150
200
250
100
200
300
400
500
600
700


Thresholding Of Color Image
50
100
150
200
250
50
100
150
200
250

In first step, I just create x-axis and y-axis for each RGB band. Then I would like to convert those to sober operator in (i,j). I used this equation: C_Combine(i,j,k) = sqrt((double(ccx(i, j, k))).^2+ (double(ccy(i, j, k))).^2);



The local adaptive thresholding shows some RGB colors in the image. It is similar to color image edge detector and thresholding of color image.