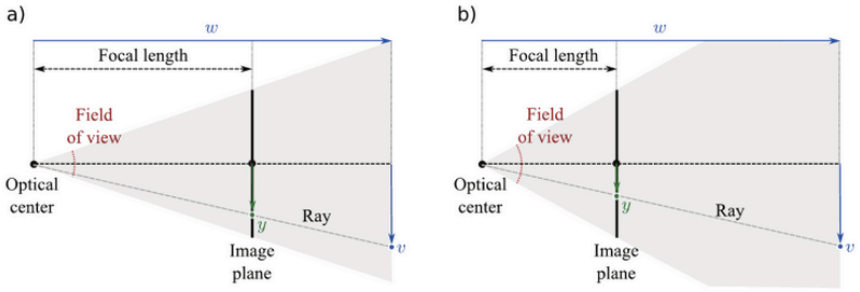
Student: Gurpreet Singh
CSc 59866/59867
Capstone I/II Fall 2015-Spring 2016

Professor Zhigang Zhu
September 30, 2015
Assignment #1

1. **Writing Assignments** (10x3 = 30 points)

a. How does an image change (e.g., objects' sizes in the image, field of view, etc.) if the focal length of a pinhole camera is varied?

Focal length, F, defines the field of view, FOV. The relationship between the F and FOV is that when the F decreases, the FOV increases, and vise versa. The image size changes as we change focal length. The smaller the focal length, the closer we are to objects in the scene and the wider we see, and the objects will seem bigger.



The image above shows the relationship between F and FOV. Image a shows the original F and Image b halve the F. As you can see the FOV increases if we go from a to b.

b. Give an intuitive explanation of the reason why a pinhole camera has an infinite depth of field.

Depth of field, DOF, refers to the region in front of and behind the sharply focused part of a scene that appears to be in focus as well. A regular camera is only able to focus at one distance at a time, so it has limited DOF. Therefore, any part of a scene, even slightly in front of the subject in focus or the slightest distance behind it, is out of focus.

On a pinhole camera, the size of the pinhole controls the amount of light rays that come into the camera. The larger the pinhole, the higher number of light rays gets inside the camera and vice versa. However, to make sharp images of farther points, the pinhole needs to be small. If the pinhole is small enough, all the far points in the scene will appear to be in focus and we reach infinite depth of field.

c. Prove that, in the pinhole camera model, three collinear points (i.e., they lie on a line) in 3D space are imaged into three collinear points on the image plane. You may either use geometric reasoning (with line drawings) or algebra proof (using equations).

Three points are collinear if and only if the determinant found by placing the x-coordinates in the first column, the y-coordinates in the second column, and z-coordinates in the third column is equal to zero.

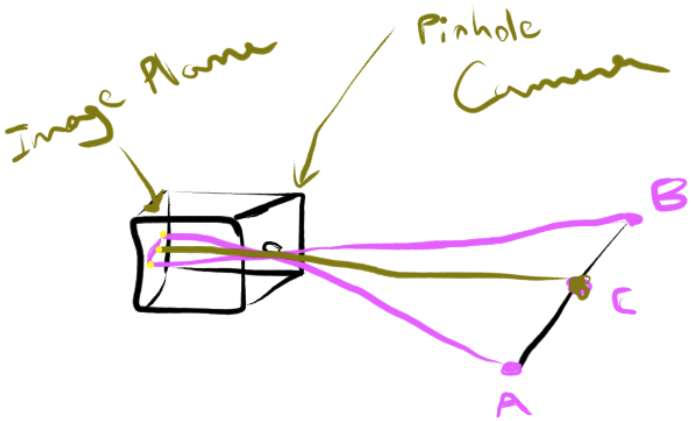
$$Det \begin{bmatrix} X1 & Y1 & Z1 \\ X2 & Y2 & Z3 \\ X2 & Y3 & Z3 \end{bmatrix} = 0$$

We can apply the perspective projection to each point to get the three points X1, X2, X3 and the

$$Det \begin{bmatrix} \frac{X1}{Z1} & \frac{Y1}{Z1} & 1 \\ \frac{X2}{Z2} & \frac{Y2}{Z2} & 1 \\ \frac{X3}{Z3} & \frac{Y3}{Z3} & 1 \end{bmatrix} = 0 \quad \longrightarrow \quad Det \begin{bmatrix} x1 & y1 & 1 \\ x2 & y2 & 1 \\ x3 & y3 & 1 \end{bmatrix} = 0$$

The determinant of this matrix where the points are projected on the image plane is also zero, which means that they are collinear.

We can see it intuitively if we have 2 points in 3D space and then define a line, and choose a point from that line and project it on the image plane. This approach is shown below:

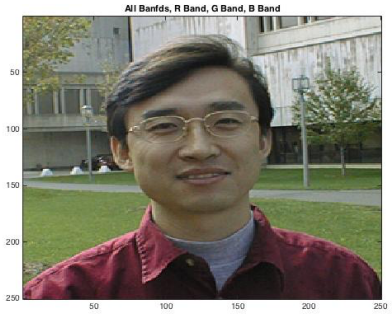


2. **Programming Assignments** (70 points in total)

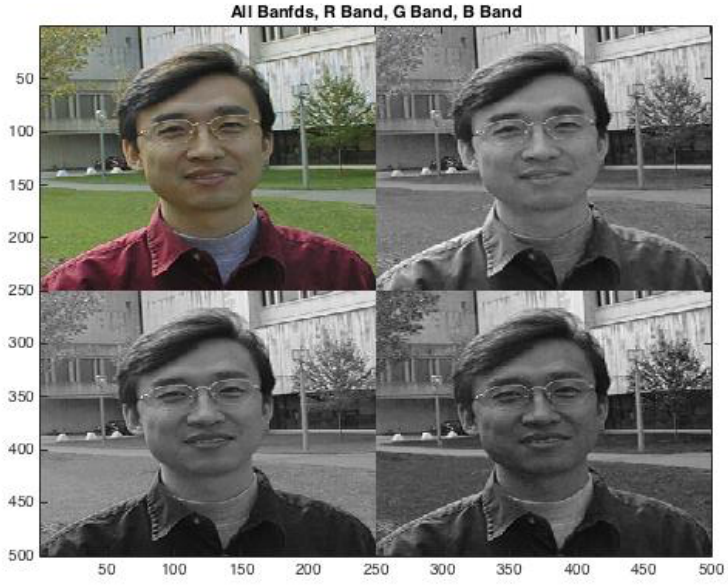
PART I

(30 points) Image formation. In this small project, you are going to use Matlab to read, manipulate and write image data. The purpose of the project is to make you familiar with the basic digital image formations. Your program should do the following things:

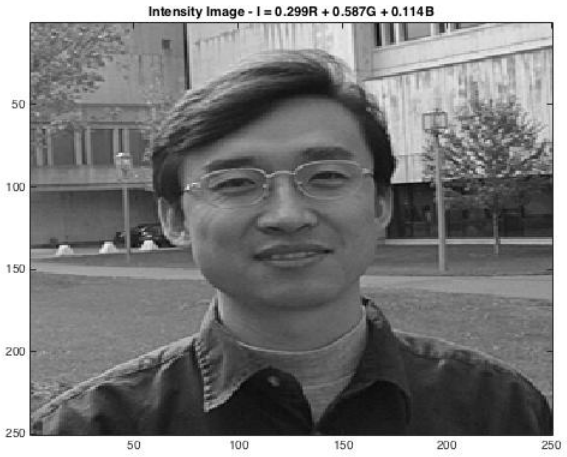
a. Read in a color image C1(x,y) = (R(x,y), G(x,y), B(x,y)) in Windows BMP format, and display it.

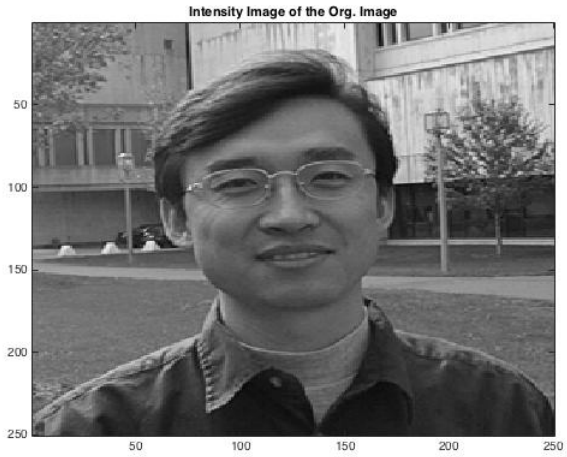


b. Display the images of the three color components, R(x,y), G(x,y) and B(x,y), separately. You should display three black-white-like images.



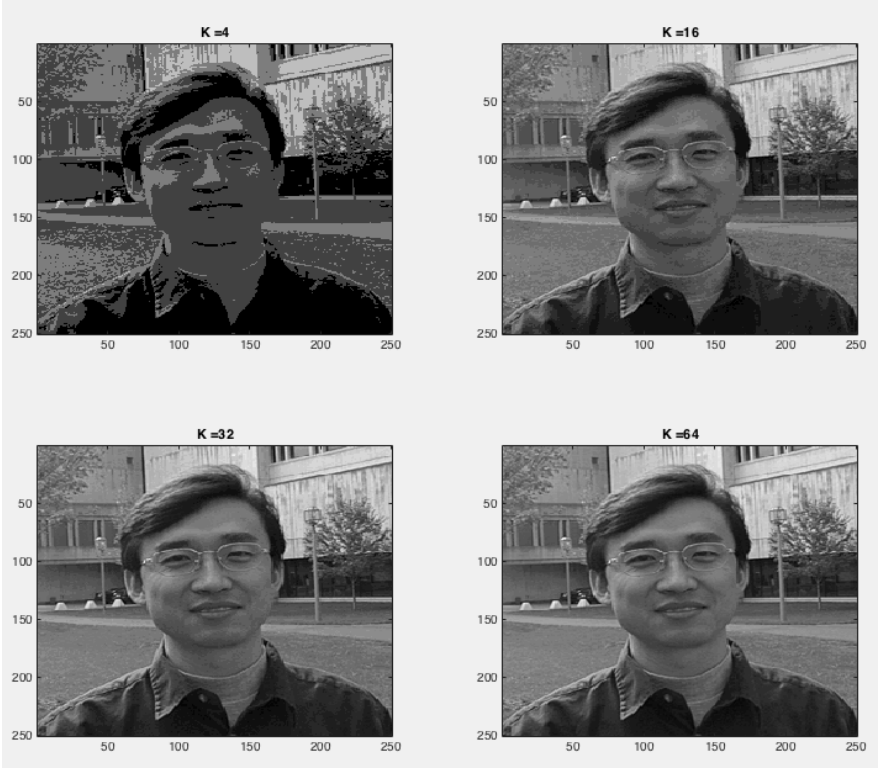
c. Generate an intensity image I(x,y) and display it. You should use the equation I = 0.299R + 0.587G + 0.114B (the NTSC standard for luminance).





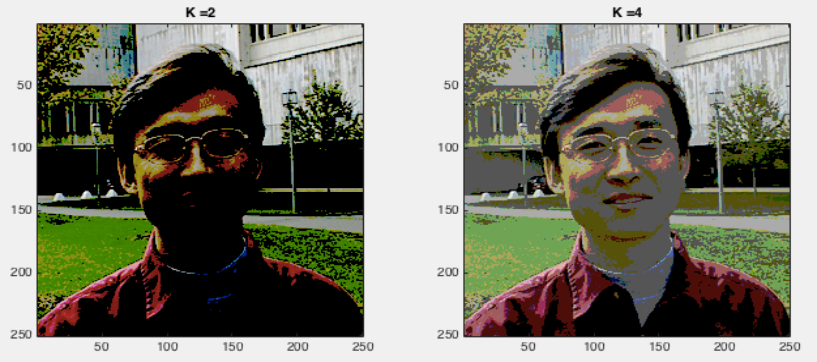
The image of the left was created using the formula. The image on the right was created using the MATLAB's built in function. The human eye cannot tell the difference between the two images.

d. The original intensity image should have 256 gray levels. Please uniformly quantize this image into K levels (with K=4, 16, 32, 64). As an example, when K=2, pixels whose values are below 128 are turned to 0, otherwise to 255. Display the four quantized images with four different K levels and tell us how the images still look like the original ones.



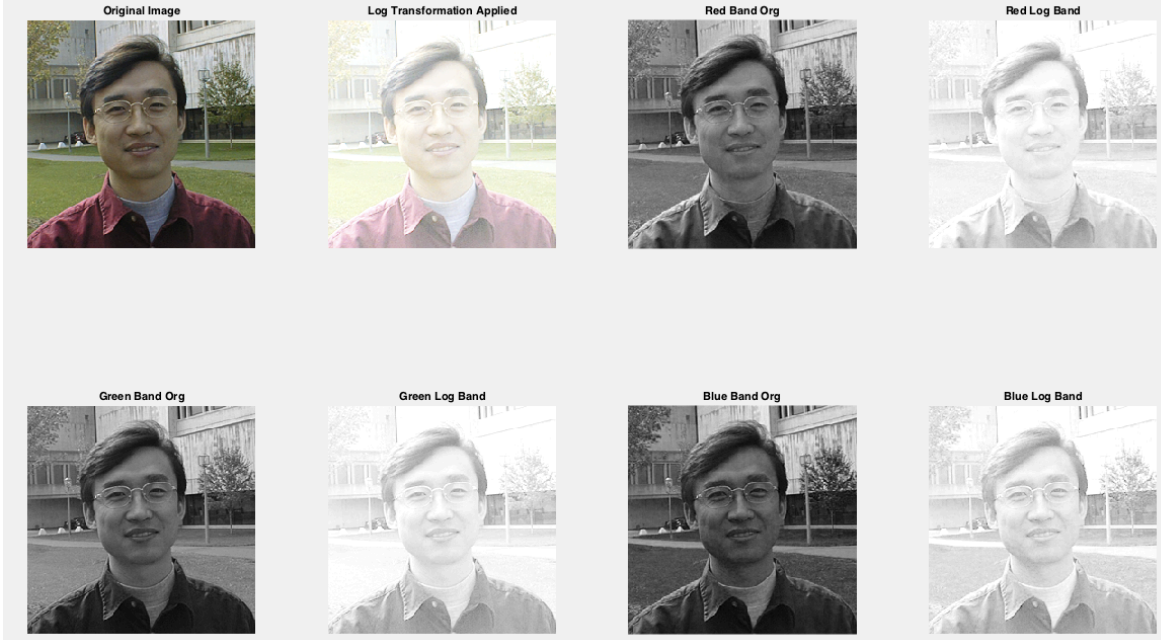
You can see the intensity images above for different value of K. I observed that the more colors you have in the image the better image quality you get and the details become clear. For each K, I divided 256 into K equal intervals, and then if a pixel value falls in a range, P < v < P', I took the lower value, P, from the interval and made that the pixel value. As the number of intervals increase the image looks more smoother,

e. Quantize the original three-band color image C1(x, y) into K level color images CK(x, y)= (R$\hat{}$(x, y), G$\hat{}$(x, y), B$\hat{}$(x, y)) (with uniform intervals) , and display them. You may choose K=2 and 4 (for each band). Do they have any advantages in viewing and/or in computer processing?



For the images above, I took the R, G, B bands and for each band, I created intervals and if a pixel falls in the interval, P < v < P', I make the pixel value P. By doing this we decrease the quality of the image because we reduce the number of colors in the image, but an advantage to do this in computer processing would be that the image can we easily compressed. Since we reduced the colors in the image, we can see the edges more clearly now as well. For higher quality images, we will need a higher K value.

f. Quantize the original three-band color image C1(x,y) into a color image CL(x,y)= (R$\hat{}$(x,y), G$\hat{}$(x,y), B$\hat{}$(x,y)) (with a logarithmic function) , and display it. You may choose a function I' =C ln (I+1) (for each band), where I is the original value (0~255) , I' is the quantized value, and C is a constant to scale I' into (0~255), and ln is the natural logarithm. Please find the best C value so for an input in the range of 0-255, the output range is still 0 - 255. Note that when I = 0, I' = 0 too.



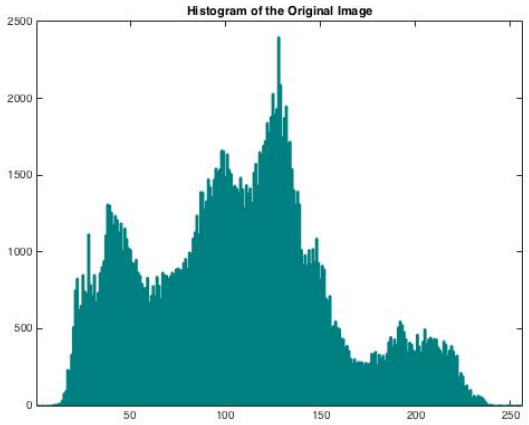
The logarithmic function is used to reduce the contrast of brighter regions. The result can be seen in the images above. In the above images, it can be observed that all the dark regions in the original image become lighter. You can see the log transformation of all the bands. The final image after combining all the bands looks enhanced.

PART II

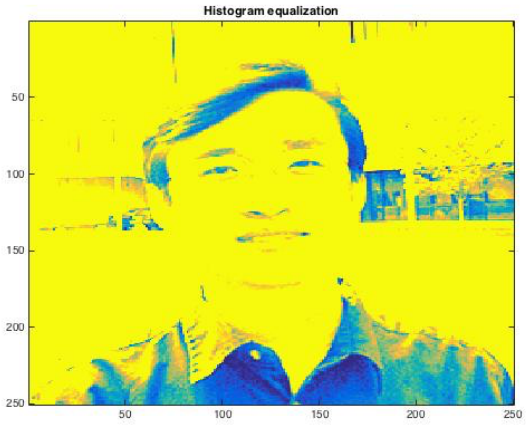
Generate the histogram of the image you are using, and then perform a number of histogram operations (such as contrast enhancement, thresholding and equalization) to make the image visually better for either viewing or processing (10 points). If it is a color image, please first turn it into an intensity 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.

Histogram Equalization

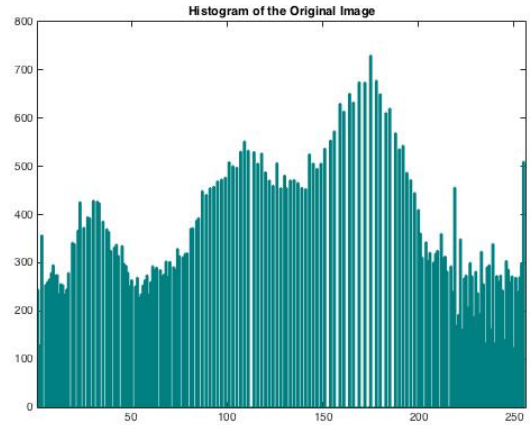
A



B

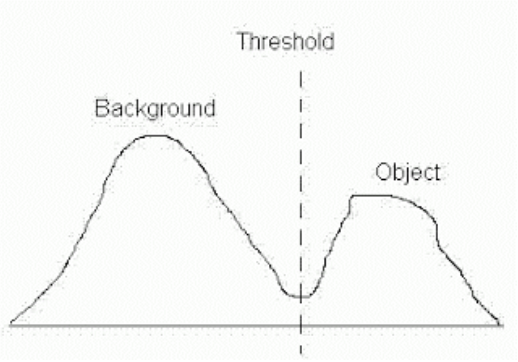


C



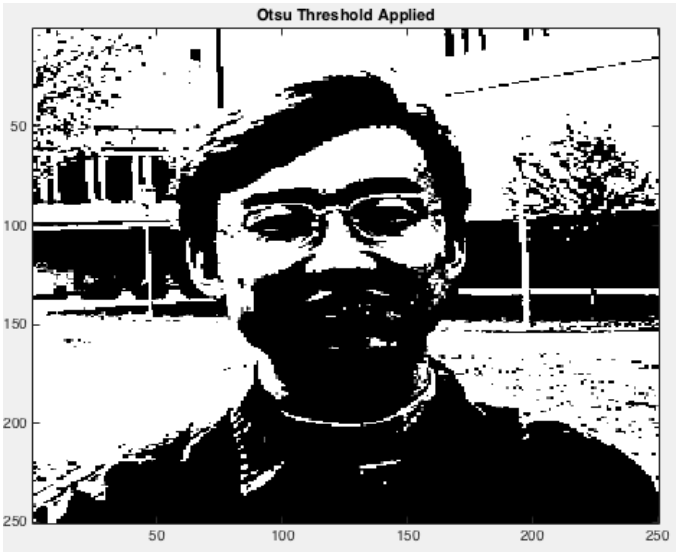
D

The histogram of the original image can be observed in Image B. The histogram has several peaks. Mostly, the pixels are between 20-170, and then there is a small peak between 170-240. The histogram is multi-modal. The original image is low in contrast. By looking at the histogram in image B, a threshold can found, where all pixels above certain brightness level is maximized to the maximum value and others are minimized to the minimum value. This is very useful in object and background separation if there is significant difference between an object and its background on an image.



The result of applying the Histogram Equalization Algorithm, which stretches the histogram, is shown in image C and the resulting histogram can be seen in Image D. It can be observed that the image is more contrast and the histogram is more stretched

Threshold



By looking at the histogram of the original, it can be observed that the threshold is between the middle of the large peaks between 80-130, close to 105. And by finding that threshold, I made all the pixels that are below that threshold 0, and all the pixels that are more than that 255.

PART III

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). 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 ? In the end, please try to generate a sketch of an image, such as the ID image of Prof. Zhu.

The Sobel operator uses a 3 x 3 mark that considers the neighborhood pixels for the gradient computation. The magnitude of the gradient is at the center pixel (i, j) in the matrix below and is calculated by Gx and Gy, also shown below:

$G_{mx} =$

−1	0	1
−2	0	2
−1	0	1

$G_{my} =$

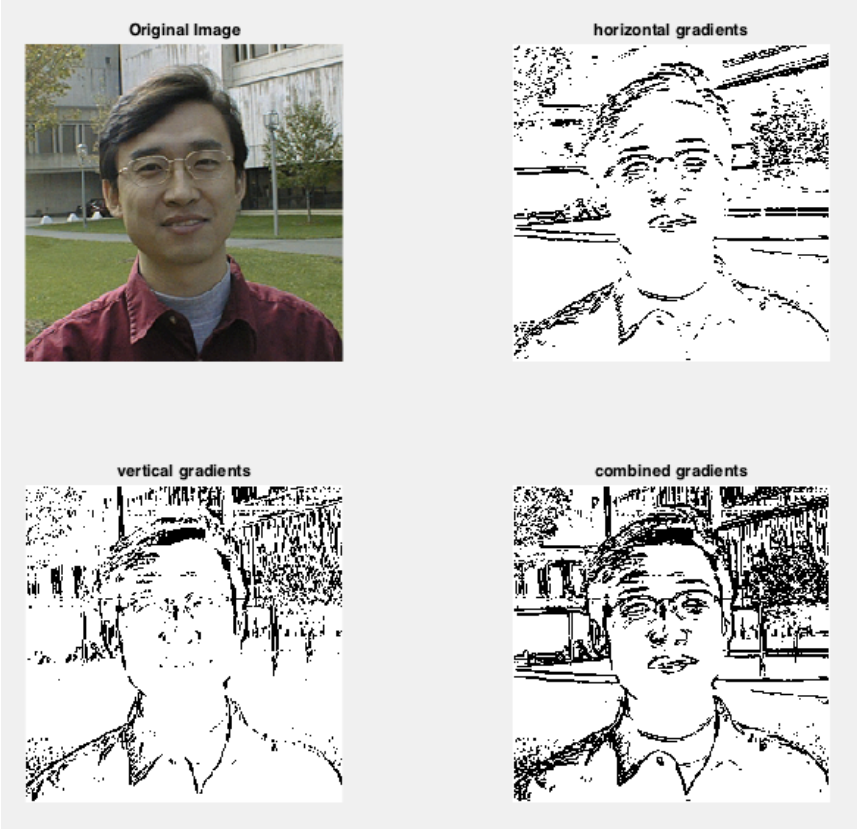
1	2	1
0	0	0
−1	−2	−1

$$G_x = (f_{13} + 2f_{23} + f_{33}) - (f_{11} + 2f_{21} + f_{31})$$

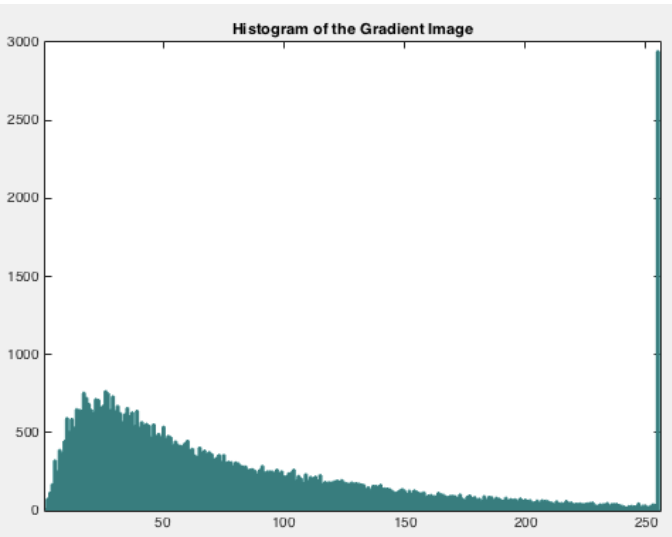
$$G_y = (f_{11} + 2f_{12} + f_{13}) - (f_{31} + 2f_{32} + f_{33})$$

The sobel operator places emphasis on pixels located coser to the center pixel. It provides both differencing and smoothing effects.

You can see the results on the sobel operator below. Gx is the Horizontal gradient and Gy is the vertical gradient. For each pixel, I calculate the magnitude of Gx and Gy and then I compare to the threshold to determine if the pixels belong there or not. I combine the horizontal and vertical gradients and the results are shown below.

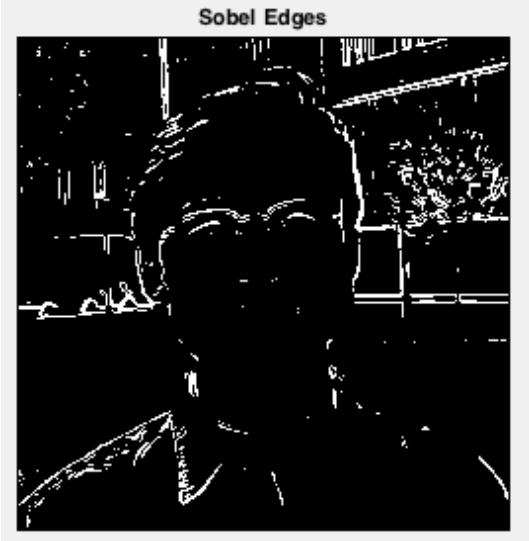


This image below shows the gradient without any threshold. The histogram of this image can be seen below:

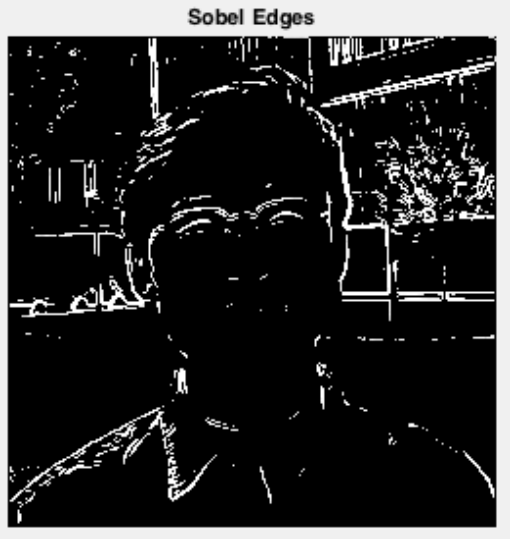


Below images keep certain percentage of pixels (e.g. 5% of the pixels with the highest gradient values) as edge pixels (edges).

5%



15%

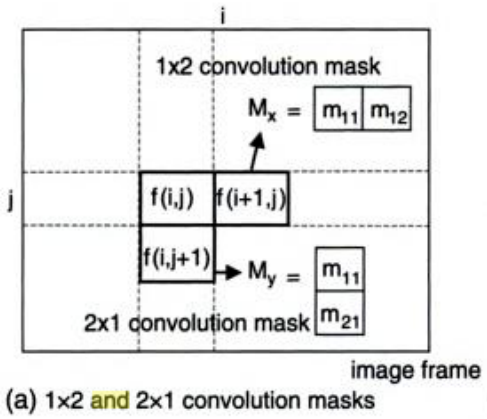


25%

35%

From the results above, I conclude that the if we keep between 15% - 30% of the pixels, then we get better results. This might not be the general case but for this image it seems that there isn't much difference between 5% and 15%. We can also analyze the histogram of the gradient image and see that most of the pixels are between 0-70 and then there is a Hugh bump at 255, which are probably the edge pixels.

The 1x2 operator uses the following masks.

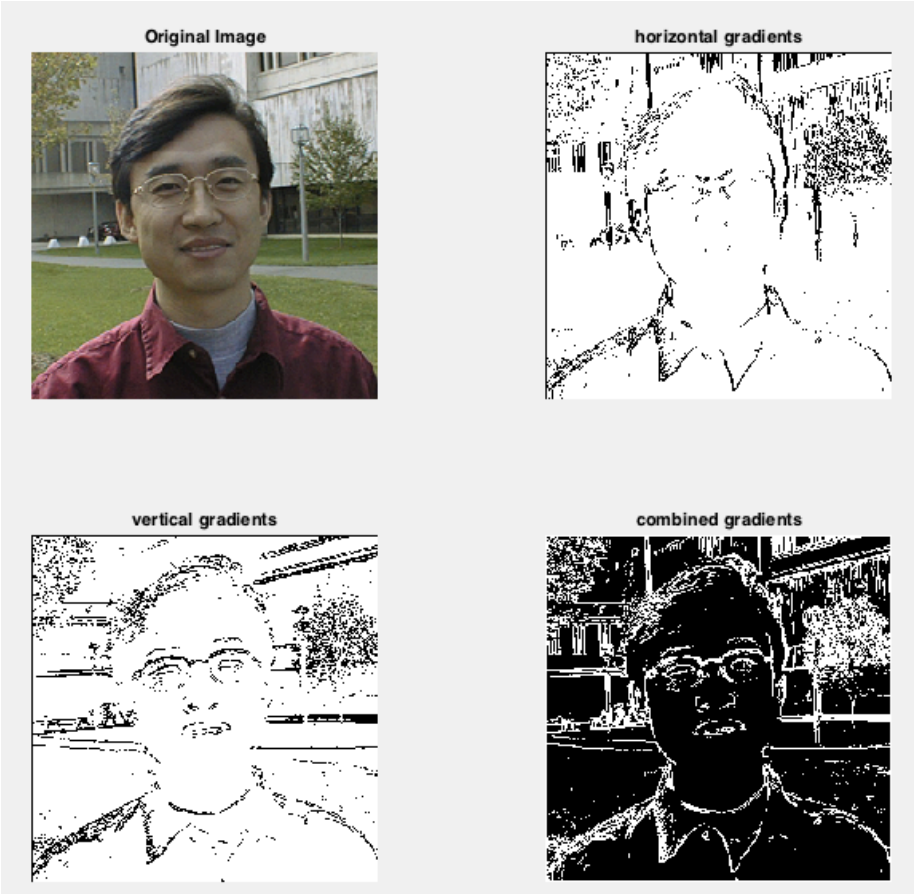


(a) 1×2 and 2×1 convolution masks

The values are computed using the following equations:

$$M_x = m_{11} f(i,j) + m_{12} f(i+1,j), \qquad M_y = m_{11} f(i,j) + m_{21} f(i,j+1)$$

The results of the 1x2 operator are shown below:



For the 1x2 operator, I used a small threshold to get the edges of the images.

In the case of Sobel, I used 107 as the threshold and in the case of 1x2 I used 23. The conclusion of using a smaller threshold for the 1x2 operator tends to produce an image that contains noise and other misc info. The use of a higher threshold for the 1x2 results in a single color image with few minute pixels. The use of a higher threshold for Sobel produced a much cleaner image as well as an image that is much easier to work with.