Mohamed Sondo Prof Zhu Computer Vision(cs 471)

10/16/2017 Assingment 2

Question 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.

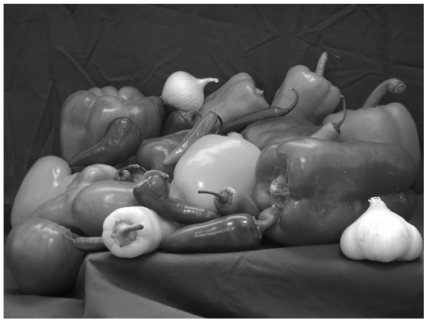
Orginal image Intensity image

After using Matlab Code in File imagesAsintensity :
RGB = imread('peppers.png');
imshow(RGB)
I = rgb2gray(RGB);
I was able to Generate Intensity image as pepperAsIntensity.png

The algorithm work as follows:

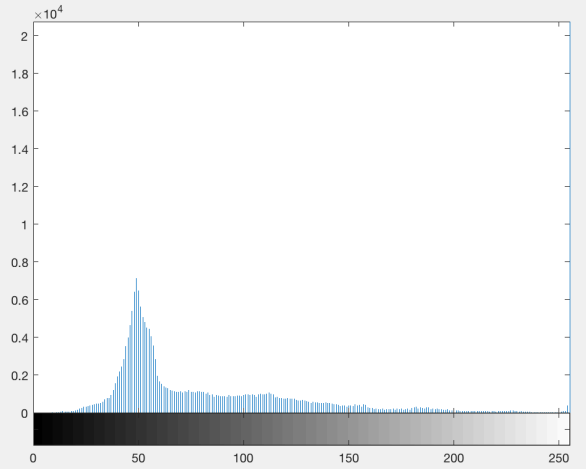
The intensity : intensity = 0.2989*red + 0.5870*green + 0.1140*blue

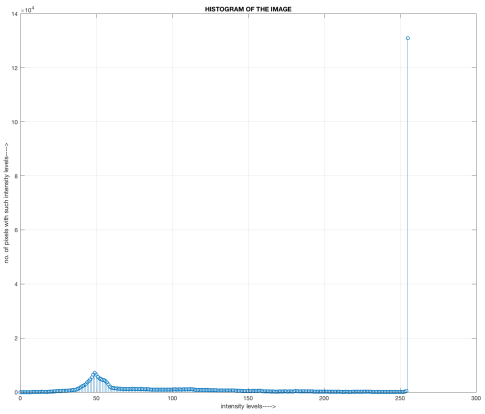
```
% Assume you have an RGB image of class double, or create a random one

rgb = rand(200,200,3);

% Convert it

gray = 0.2989 * rgb(:,:,1) + 0.5870 * rgb(:,:,2) + 0.1140 * rgb(:,:,3);
```



Histogram of Resulting intensity image. Using imhist(I) VS
Or my writing matlab code.

```
g=imread('pepperAsIntensity.png');
h=rgb2gray(g);
[M,N]=size(h);

t=1:256;
n=0:255;
count=0;

for z=1:256
   for i=1:M
      for j=1:N

         if h(i,j)==z-1
            count=count+1;
         end
      end
   end
         t(z)=count;
         count=0;
end
disp(t')

stem(n,t);
grid on;
ylabel('no. of pixels with such intensity levels---->');
xlabel('intensity levels---->'); title('HISTOGRAM OF THE IMAGE')
```





Using matlab imhist() using my histogram code.

Question: 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.

Applying Equalization:
Algorithm:

PMF

First we have to calculate the PMF (probability mass function) of all the pixels in this image.

CDF

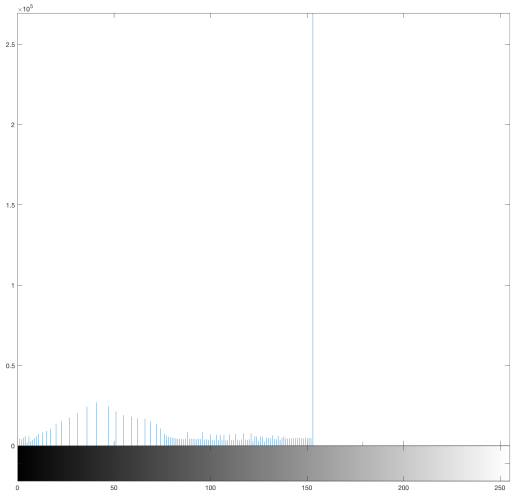
Our next step involves calculation of CDF (cumulative distributive function).

Calculate CDF according to gray levels

In Order to Equalized

```
HIm=uint8(zeros(size(GIm,1),size(GIm,2)));

freq=zeros(256,1);

probf=zeros(256,1);

probc=zeros(256,1);

cum=zeros(256,1);

output=zeros(256,1);

%freq counts the occurrence of each pixel value.
```

```
%The probability of each occurrence is calculated by probf.
for i=1:size(GIm,1)
   for j=1:size(GIm,2)
      value=GIm(i,j);
      freq(value+1)=freq(value+1)+1;
      probf(value+1)=freq(value+1)/numofpixels;
   end
end

%The cumulative distribution probability is calculated.
for i=1:size(probf)
  sum=sum+freq(i);
  cum(i)=sum;
  probc(i)=cum(i)/numofpixels;
  output(i)=round(probc(i)*no_bins);
end
for i=1:size(GIm,1)
   for j=1:size(GIm,2)
         HIm(i,j)=output(GIm(i,j)+1);
   end
end
figure,imshow(HIm);
title('Histogram equalization');
```



Result Images Equalized.

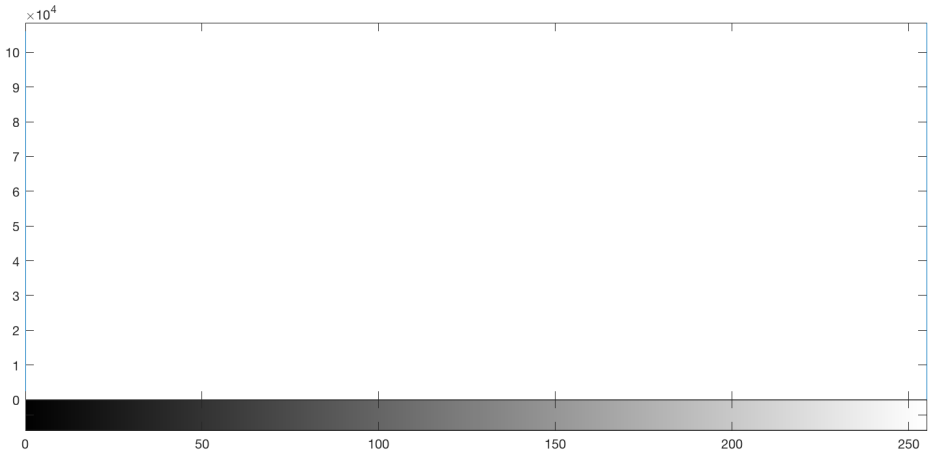
Result Images Equalized Histogram

Images Treshold Operation:
Convert Image to a binary Images.
Thresholding enables to achieve image segmentation in the easiest way. Image segmentation means dividing the complete image into a set of pixels in such a way that the pixels in each set have some common characteristics.
hresholding splits histogram, merges halves

pepperAsintensity.png

original image Vs Treshold images operation

Graph Of the Histogram Operation As Treshhold Operation



Treshold Image Historigram

Remark:

Some observations of the image based on its histogram

As you can clearly see above from the images that the new image contrast has been enhanced and its histogram has also been equalized. There is also one important thing to be note here that during histogram equalization the overall shape of the histogram changes, whereas in

histogram stretching the overall shape of histogram remains same. The Equalized image have more major peak spread out.
By looking at the histogram for a specific image a viewer will be able to judge the entire tonal distribution. Histogram quantifies the number of pixels for each intensity value

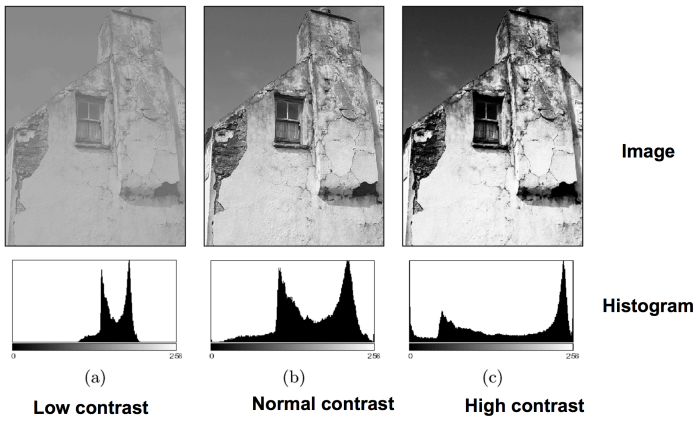
For contrast and Brightness, we see that all the pixel values has been shifted towards right and thus, it can be validated from the image that new image is darker and now the original image look brighter as compare to this new image.

Brightness of a grayscale image all pixels in image: using the Formula

$$B(I) = \frac{1}{wh}\sum_{v=1}^{h}\sum_{u=1}^{w} I(u,v)$$

We sump up all of pixel intensities divide by total number of pixels.

The contrast of a grayscale image indicates how easily objects in the image can be distinguished



Question 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 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**)

MatlabCode:

```
 i = imread('pepperAsIntensity.png');
I = rgb2gray(i);
BW1 = edge(I,'prewitt');
BW2= edge(I,'sobel');
BW3= edge(I,'roberts');
subplot (2,2,1);
imshow(I);
title('original');
subplot(2,2,2);
imshow(BW1);
title('Prewitt');
subplot(2,2,3);
imshow(BW2);
title('Sobel');
subplot(2,2,4);
imshow(BW3);
title('Roberts');
```

Orginal RGB Image.

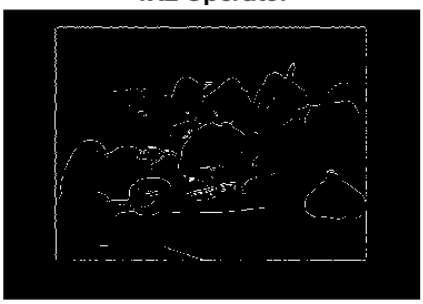
1x2 vs Sobel Operation:

The Sobel method finds edges using the Sobel approximation to the derivative. It returns edges at those points where the gradient of I is maximum

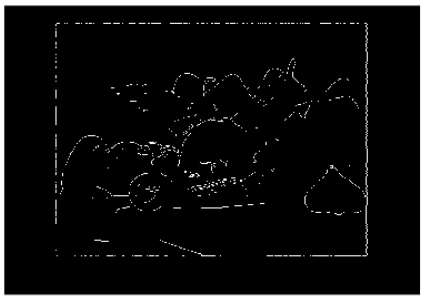
The 1x2 method finds edges using the Prewitt approximation to the derivative. It returns edges at those points where the gradient of I is maximum.

original

1X2 Operator



Sobel



Observation and Answer to question
Sobel vs 1X2 Operator Advantage.

Yes the Sobel operator have any clear visual advantages over the 1x2 operator
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.

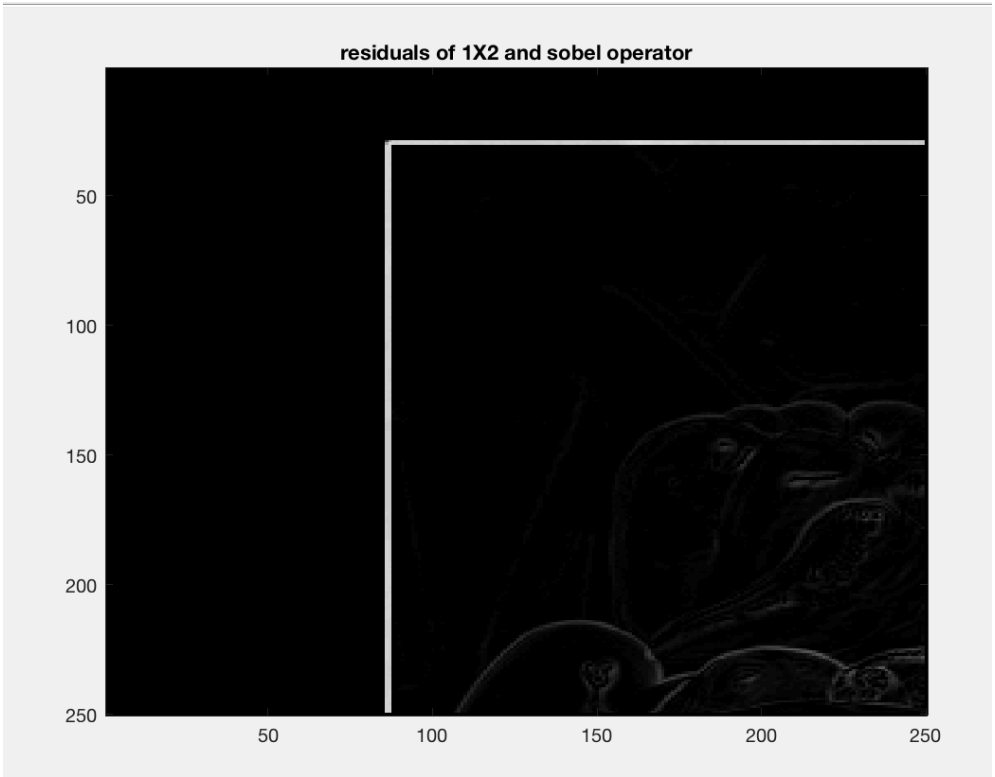
So, in this way you can see that we can detect both horizontal and vertical edges from an image. Also, if you compare the result of sobel operator with 1X 2 operator, we find that sobel operator finds more edges or make edges more visible as compared to 1X 2 operator. This is because in sobel operator we have allotted more weight to the pixel intensities around the edges.

Any disadvantages (5 points)

We Observe that the magnitude of the edges will degrade as the level of noise present in image increases. As a result, Sobel operator accuracy suffers as the magnitude of the edges decreases. Overall, the Sobel method cannot produce accurate edge detection with thin and smooth edge.

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)

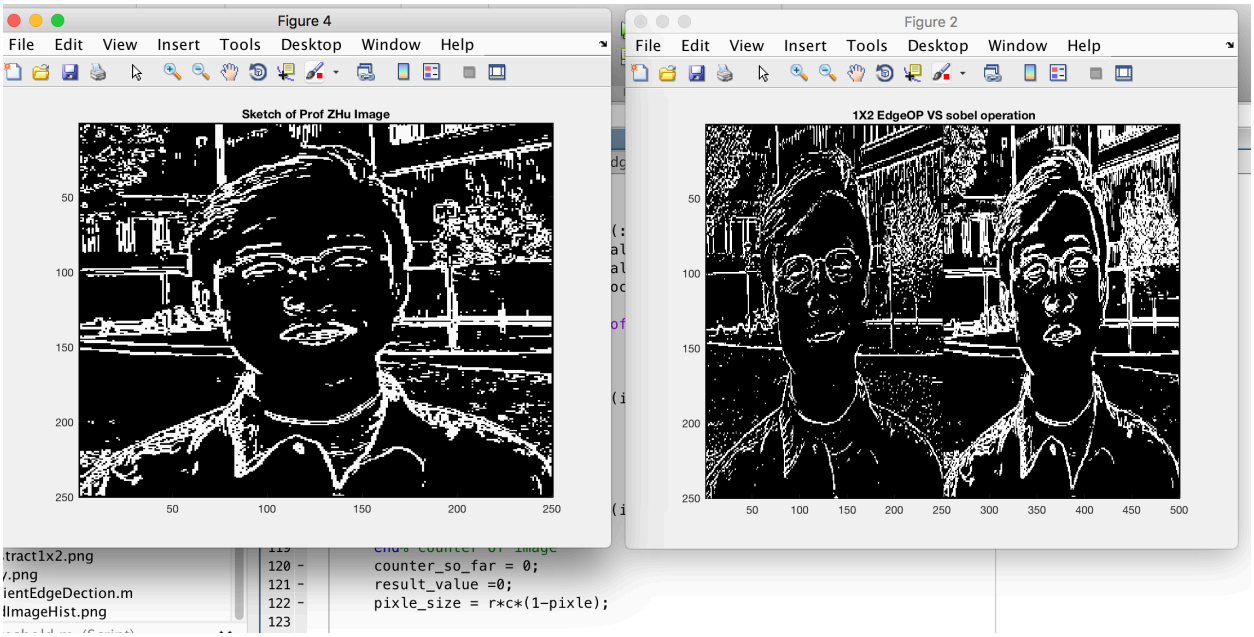
We do get some residual after the subtraction operation.



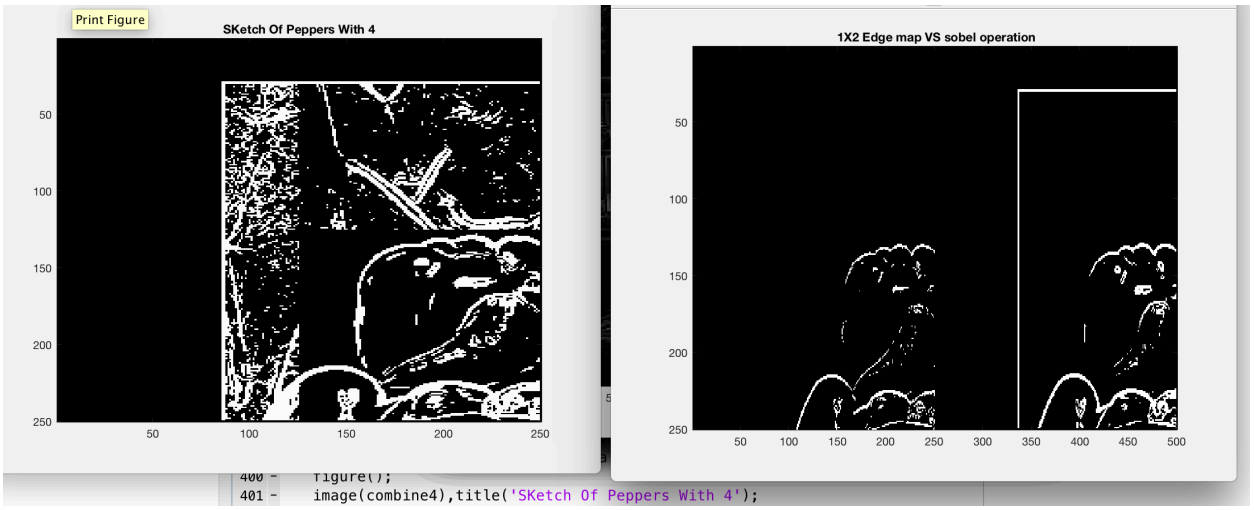
Result Of subtracting 1x2 from sobel

Question 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.



ProfZhu Steck



Steck Of Peppers

Question 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).

Well the result of increasing the kernel is going to depend on what type of image processing we are doing .
After Applying filter we observe that
With such a small filter matrix, this gives only a very soft blur for example: A 3X3 when increase to 5x5 will have bigger filter operation cause the image to be more blur. Increasing the 5x5 to 7x7 Will result in sharping the image.

When we apply a 1x2
Discuss computational cost is linear time. And we are doing n operation. The edge detection Gradient in apply on the x direction. Running time about O(n)

When we apply a 3x3 convolution kernel, we makes the operator less sensitive to noise. The operator also generally produces considerably higher output values for similar edges. These kernels are sensitive to the edges. This type of Kernel is for detecting in both directions, while *the edges* are sensitive for the horizontal and vertical edges respectively. The result of a 3x3 filter is too dark on the current image. And the Sum. that the sum of all the elements is 0 now, which will result in a very dark image where only the edges it detected are colored

When we apply a 5x5

the 2D convolution operation requires a 4-double loop. Meaning an increase in computation time. O(n^4) .The filter calculation itself is a 4-double loop that has to go through every pixel of the image, and then through every element of the filter matrix. The location imageX and imageY is calculated so that for the center element of the filter it'll be x, y, but for the other elements it'll be a pixel from the image to the left, right, top or bottom of x, y. Its modulo divided through the width (w) or height (h) of the image so that pixels outside the image will be wrapped around. Before modulo dividing it, w or h are also added to it, because this modulo division doesn't work correctly for negative values. Now, pixel (-1, -1) will correctly become pixel (w-1, h-1).

To sharpen the image is very similar to finding edges, add the original image, and the image after the edge detection to each other, and the result will be a new image where the edges are enhanced, making it look sharper. Adding those two images is done by taking the edge detection filter from the previous example, and incrementing the center value of it with 1. Now the sum of the filter elements is 1 and the result will be an image with the same brightness as the original, but sharper.

Algorithm For Filtering:

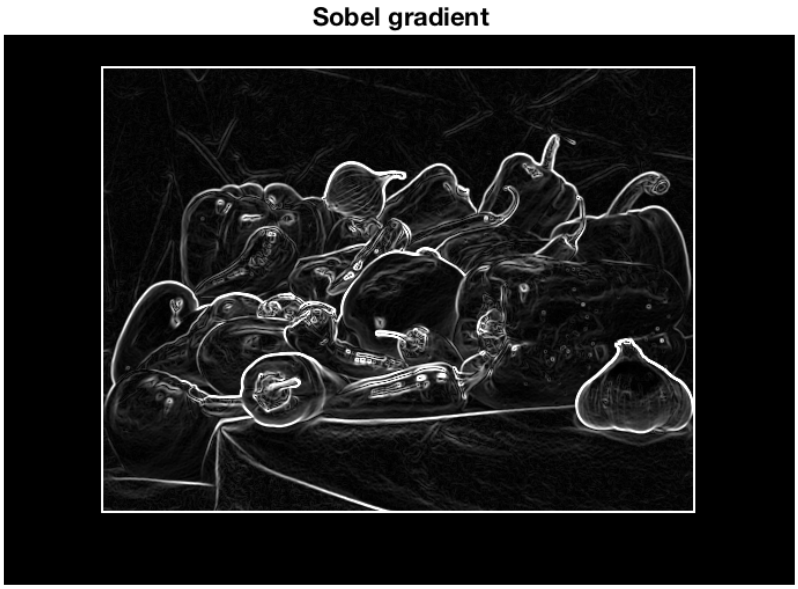
A convolution is done by multiplying a pixel's and its neighboring pixels color value by a matrix

The Kernel convolution is the core of Guassian Blur, mean Blur and and Edge Detection.
Algorithm:
We take a small grid of number and we pass them over the whole image, transforming it, based on what those number are. By Using different number in the kernel, we can perform blur, Edge Detection, sharpen, unshapen.
In term of Computation. Kernel Convolution work as follow:
Usually Our kernel is smaller than the image. For Every pixel on our image. We put our kernel over it, so that the pixel is on the center. What we now see a is nXm grid center around the pixel,
We then taken, whatever value in our kernel, multiply the value with that of the image, and sump the whole thing up. Finally, we normalized by dividing the Sum by the total value of our kernel to make sure that is don't get brighter or darker

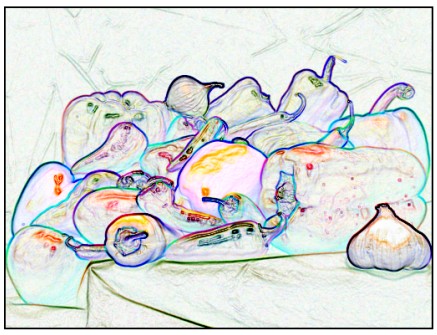
Question 5.
(20 points) Suppose you apply the Sobel operator to each of the RGB color bands of a color image. How might you combine these results into a color edge detector (5 points)? Do the resulting edge differ from the gray scale results? How and why (5 points)? You may compare the edge maps of the 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.

Sobel op Of RGB Can Gained using Eigen decomposition.
The edges in a color image can be done by decomposing the image into its channels, finding the gradients separately and fusing them In the end. the resulting edge differ from the gray scale results. Changes in color are detected even when the grayscale color of two pixels are the same. The edge strength is typically greater or equal to the magnitude obtained by simply filtering a grayscale image

This is a more precise description of the local gradients. Eigen-decomposition is then applied to the structure tensor to form the eigenvalues and eigenvectors. The larger eigenvalue shows the strength of the local image edges (gradient magnitude) and the corresponding eigenvector points across the edge (in
gradient direction). In other words, eigenvalues encode the gradient magnitude while the eigenvectors contain the gradient orientation information.



<u>Sobel Operation of Gray Scale Image: The Solbel Perform better noise suppression</u>



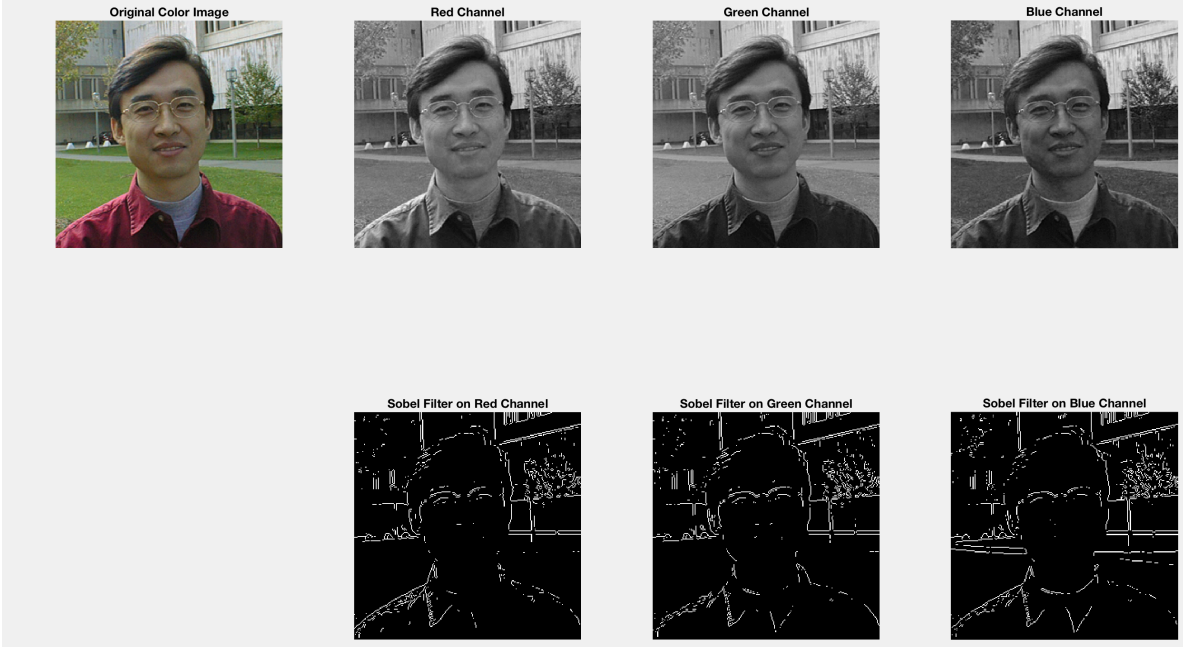
Sobel of RBG Image

In Term of comparison

We see that a significant amount of information is lost by the standard method(using Gray Scale image), but it is recovered with the gradient method

image of sobel operation on Color images.

```
rgbImage = cat(3, redEdgeImage, greenEdgeImage, blueEdgeImage);

imshow(rgbImage)
```



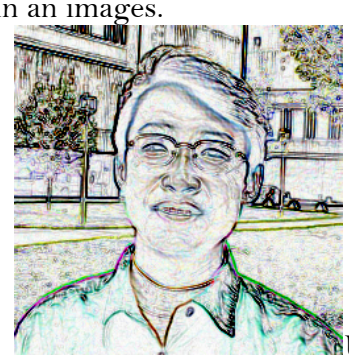
Prof Zhu Color image Edge Detection Using Channel separation methods.

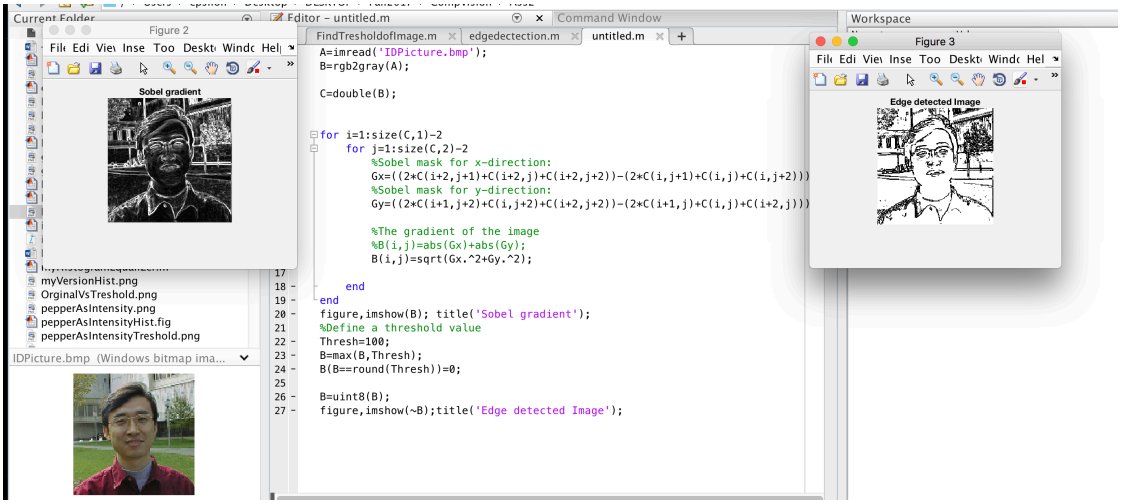
Similarities and differences, and how each of them can be used for image enhancement or feature extraction (5 points

Color edge operators are able to detect more edges than gray-level edge operators. In color image edge detection, additional features can be obtained that may not be detected in Gray-level images. However, the most commonly used edge detection techniques are Gradient-based .

Recall the sobel Op return a high response where there is sharp change in gradient of image and low response where theirs isn't the Sobel is not resolution independent. For High resolution image the gradient will be spread over many pixel, but work good on low resolution images.

The Sobel of Gray scale image produce Kind of Shallow edge and might not be for remove noise in an images.

 Using Color image edge Dectection



Sobel Operation Edge Detection on ProfZHu ID Image and threshold Level of 100.