Lab Report

Clab-1 Report

ENGN4528

Hongming Zhang

U6693651

Bachelor

29/03/2019

Task1: Matlab Warm-up. (2 marks, 0.2 per each)

Describe (in words where appropriate) the result/function of each of the following Mat-lab commands in your report. Use the Matlab help command if necessary but try to generate the output without entering these commands into Matlab. DO NOT submit a screenshot of the result of using Matlab.

(1) a = [1 : 2 : 5 0; 51 :2: 100] ;

This code is used to build a 2*25 matrix. The first raw of matrix is 25 odd numbers from 1 to 49. The second raw is another 25 odd numbers from 51 to 99.

(2) b = a(2,:);

This code means build a new matrix b by using the data from the second row of matrix a(the first question's matrix). Matrix b is a 1*25 matrix and it has 25 odd numbers from 51 to 99.

(3) f = randn(500, 1);

This code means build a 500*1matrix f with 500 different random numbers.

(4) g = f(find(f>0)) ;

This code means find the data in matrix f which is more than 0 and make these numbers as a new matrix g.

(5) x = zeros (1,100) + 0.25 ;

This code means build a 1*100 matrix with 0 and then plus 0.25 on the 1*100 zero matrix to be the new matrix x.

(6) y = 0.5 .* ones(1,length(x));

This code means build a all one matrix which one row and its row length is as long as the row length of matrix x. Then let this matrix multiply by 0.5 and get the new matrix y.

(7) z = x + y;

This code means build a new matrix z which is got by matrix x plus matrix y.

(8) a = [1: 300];

This data means build a 1*300 matrix a with 300 numbers from 1 to 300.

(9) b = a([end: -1:1]);

This code means build a new matrix b. The matrix b is the order of a matrix rom tail to head.

(10) b(b≤50)=0;

This code means find all the numbers in matrix b which is not more than 50 and change these data be 0.

Task2: Basic Coding Practice (1 marks, 0.2 per each)

Write functions to process an input grayscale image with following requirements, where you need to

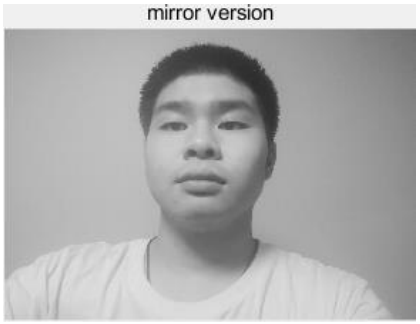
write a script to load an image, apply each transformation to the input, and display the results in a figure using the MATLAB subplot()function. Label each subplot with an appropriate title.

(1) Load a grayscale image, and map the image to its negative image, in which the lightest values appear dark and vice versa. Display it side by side with its original version.

In this question, the main task is to get the negative image of a grayscale image. The central idea is to interchange the light and the dark parts of the image. It is obvious that the uint8(from 0~255) style grayscale image can be changed to negative image by using 255 minus the original image's data. In MATLAB there is function called imcomplement() can get the minus image. The way to show the original and negative together is use subplot() function in MATLAB, which can let images in the same figure window. The results are as follow:

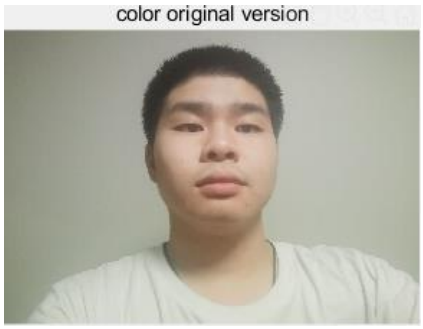
(2) Map the image to its "mirror image", i.e., flipping it left to right.

In this question, the main task is to flip the photo from left to right. The central idea is using the original image's matrix from right to left to get the new mirror image matrix. In MATLAB the function flip() not only can do this well so by using flip() function the code can get the result what the question want. The results are as follow:



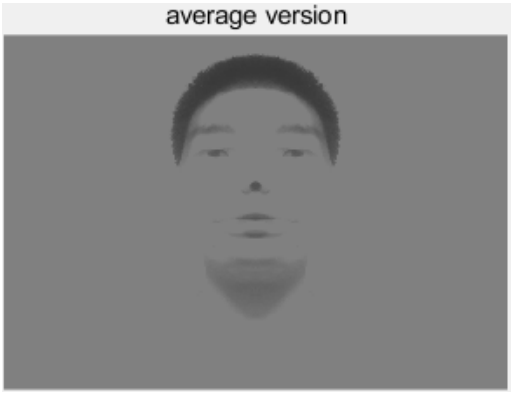
(3) Load a color image, swap the red and green color channels of the input.

Loading the color image by using function imread() in MATLAB. In a RGB image three color channels RED, GREEN and BLUE is the three different floors of the image matrix. Exchange the red and green channels is means exchange the image matrix's the first and the second floor. In MATLAB is exchange the matrix's (:,:,1) and (:,:,2). The results are as follows:



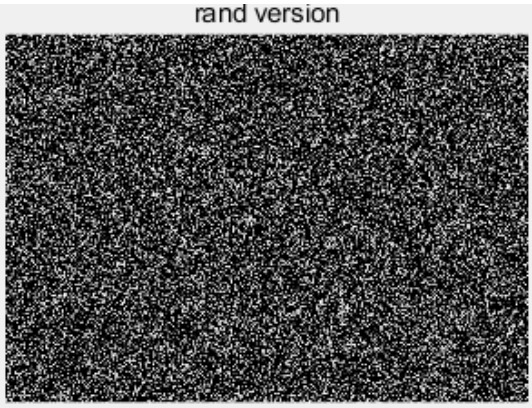
(4) Average the input image with its mirror image (use typecasting).

The main task of this question is summing the two image and getting a mirror image. The way to sum the original one with the mirror one is as same as the matrix sum. Images are matrixes in computer so just plus the matrix of the original one and the mirror one is enough. The average of the plus result image is gotten by the plus result image divided by 2. The result is as follows:



(5) Add or subtract a random value between [0,255] to every pixel in the grayscale image, then clip the new image to have a minimum value of 0 and a maximum value of 255.

The main task of this question is sum a random on each pixel of the image and then let the number in [0,255]. The way to add the random number between
[-255, 255] is make a bouble-layer cycle with the size of the image. Every single step of the cycle let one matrix value of one pixel add a random number by using the function randi([-255,255]) in MATLAB. To let all the data in the result matrix of the image is between 0 to 255, using the function mod() to mod the result matrix with 255 can get the result what the question want. The result is as follow:



Task3: Basic Image I/O

Choose one face image, for example, face01, and then program a short computer code that does the following tasks:

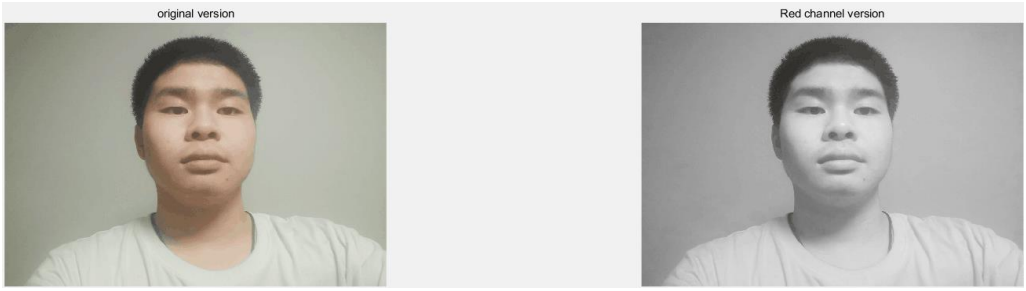
(a)Read this face image from its JPG file, and resize the image to 768×512in columns x rows (0.2′).

This question's main task is resized the original image. In MATLAB there is imresize() function can do it. By using the function as imresize(face,[512,768]) can get the resized image which the question wants. The results are as follows:

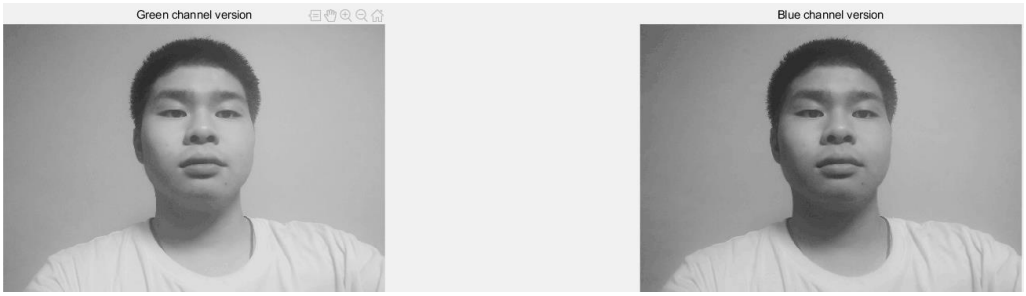
original image resized image

(b)Convert the colour image into three grayscale channels, i.e., R,G,B images, and display each of the three channel grayscale images separately (0.2′foreach channel, 0.6′in total).

The main task of this question is separating R,G,B three different channels from the original image. The RGB image is a three-dimensional matrix and each color channel is one of the dimensional. Because of this, building three new uint8 matrix by using the data from the three different channels can successfully separate R,G,B channels and got their own channel's grayscale image. The results are as follows:



Original image Red channel image

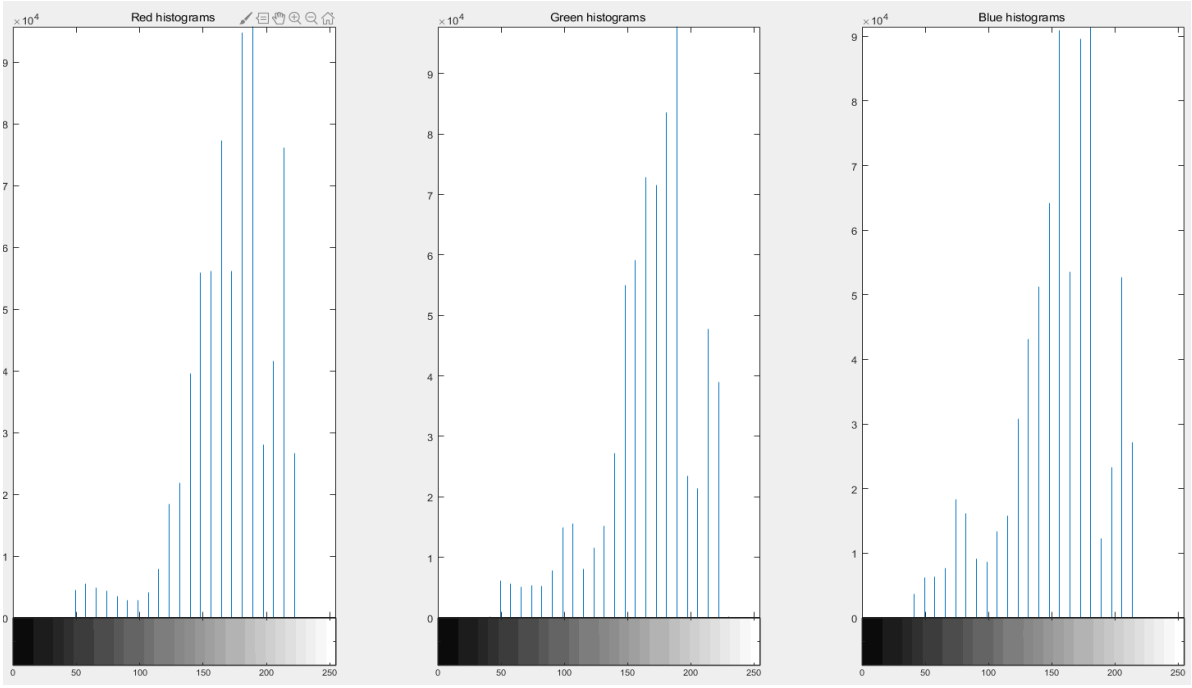


Green channel image Blue channel image

(c) Compute the 3 histograms for each of the grayscale images, and display the 3 histograms (0.2′for each histogram, 0.6′in total).

The main task of this question is getting the three color channels images' histograms. In MATLAB there is a function imhist() working for the image histogram making. By using the data from the last (b) question, the three-color channels' histograms can be drawing as follows:

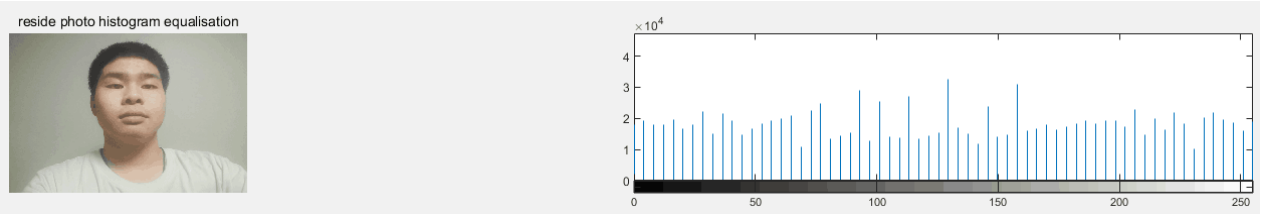
X value is the gray level; Y value is pixel number



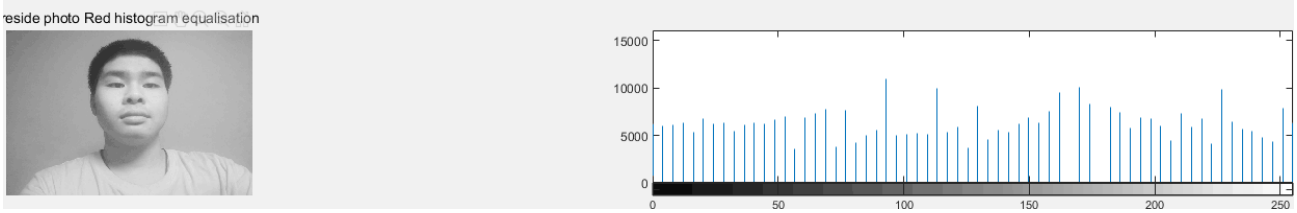
(d) Apply histogram equalization to the resized image and its three grayscale channels, and then display the 4 histograms equalization image (0.15′foreach histogram, 0.6′in total). (Hint:you can use Matlab's inbuilt histeq()function)

The main task of this question is building the histogram equalization. In fact, it is the nonlinear stretching of the image, and the pixel values of the image are reallocated, so that the number of pixel values in a certain gray range is roughly equal. In this way, the contrast of the peak part in the middle of the original histogram is enhanced, while the contrast of the bottom part of the valley on both sides is reduced, and the histogram of the output image is a flatter segmented histogram. In MATLAB, the function histeq() can do this work well. By using the data from question (b) and (c), the results of histogram equalization are as follows:

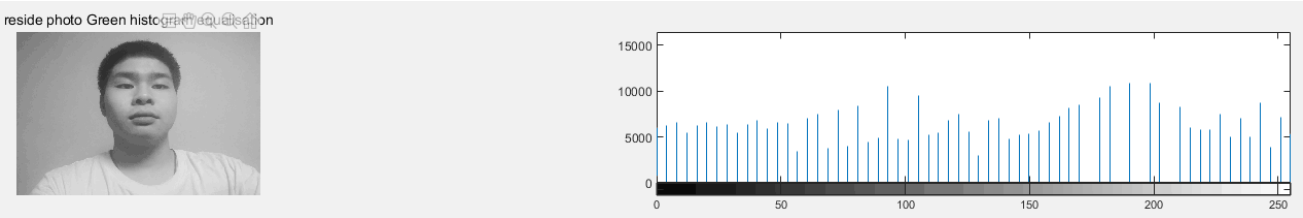
Original color image



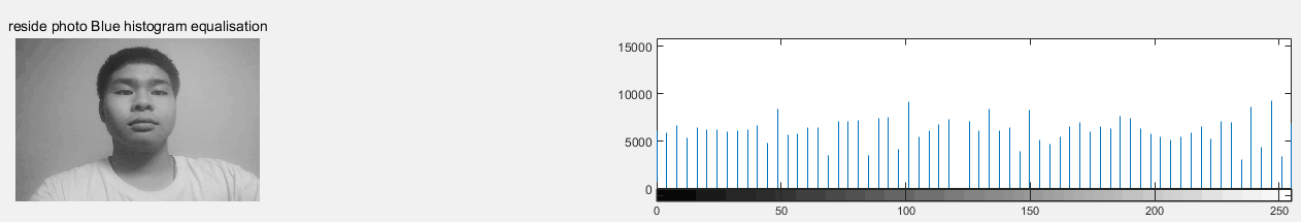
Red color channel image



Green color channel image



Blue channel image



X value is the gray level; Y value is pixel number

Task4: Color Space Conversion (5 marks)

Use the two images in Fig.1 to study color space conversion from RGB to HSV (youcan download them from the Wattle site):

1. Based on the formulation of RGB-to-HSV conversion, write your own function cvRGB2HSV()that converts the RGB image to HSV color space (1.4′). Read inFig.1(a) and convert it with your function, and then display the H, S, V channels in your report (0.2′ for each channel, 0.6′ in total).

RGB to HSV conversion formula

The R,G,B values are divided by 255 to change the range from 0..255 to 0..1:

$R' = R/255$

$G' = G/255$

$B' = B/255$

$Cmax = \max(R', G', B')$

$Cmin = \min(R', G', B')$

$\Delta = Cmax - Cmin$

Hue calculation:

$$H = \begin{cases} 0^\circ & ,\Delta = 0 \\ 60^\circ \times \left(\frac{G'-B'}{\Delta} mod 6\right) & ,C_{max} = R' \\ 60^\circ \times \left(\frac{B'-R'}{\Delta} + 2\right) & ,C_{max} = G' \\ 60^\circ \times \left(\frac{R'-G'}{\Delta} + 4\right) & ,C_{max} = B' \end{cases}$$

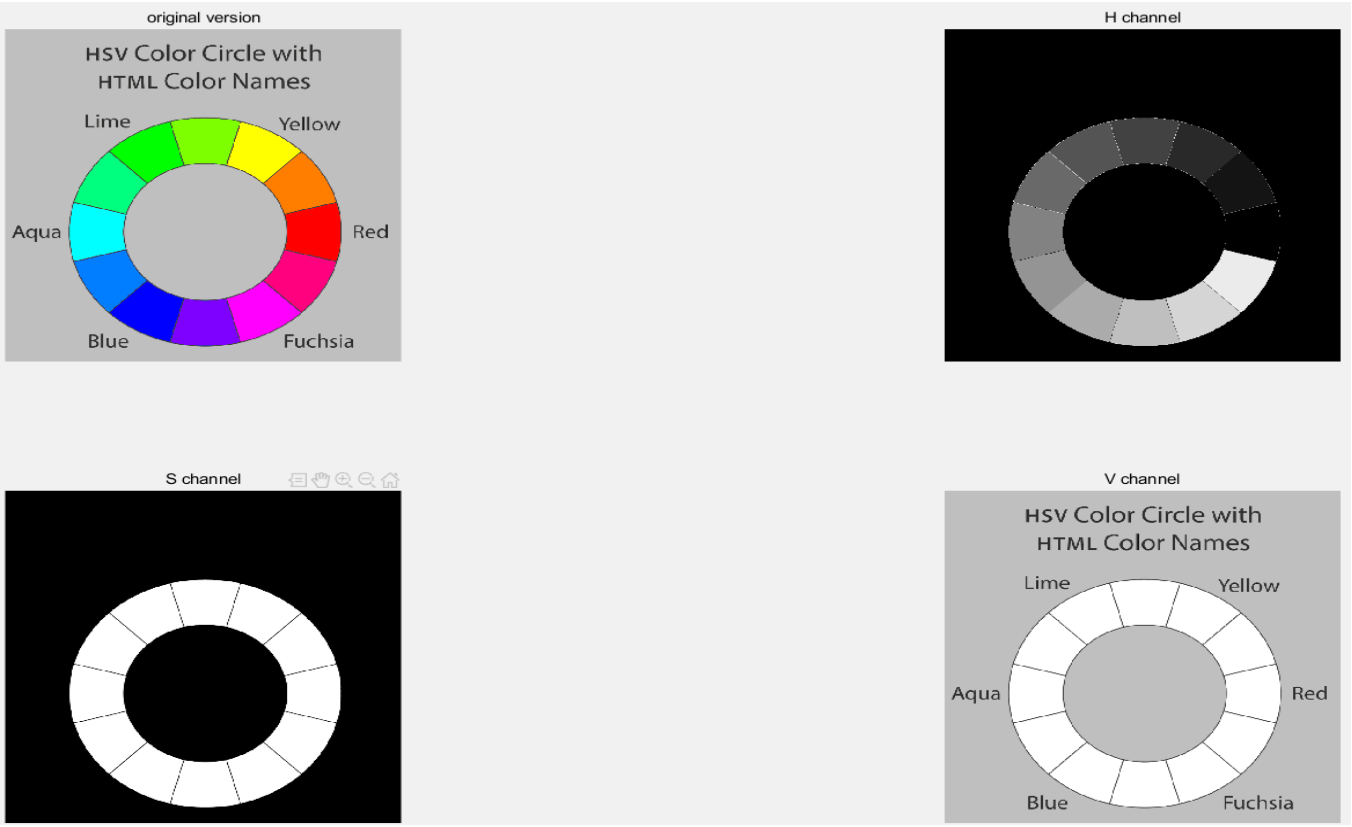
Saturation calculation:

$$S = \begin{cases} 0 & ,C_{max} = 0 \\ \frac{\Delta}{C_{max}} & ,C_{max} \neq 0 \end{cases}$$

Value calculation:

From: https://blog.csdn.net/hanshanbuleng/article/details/80383813

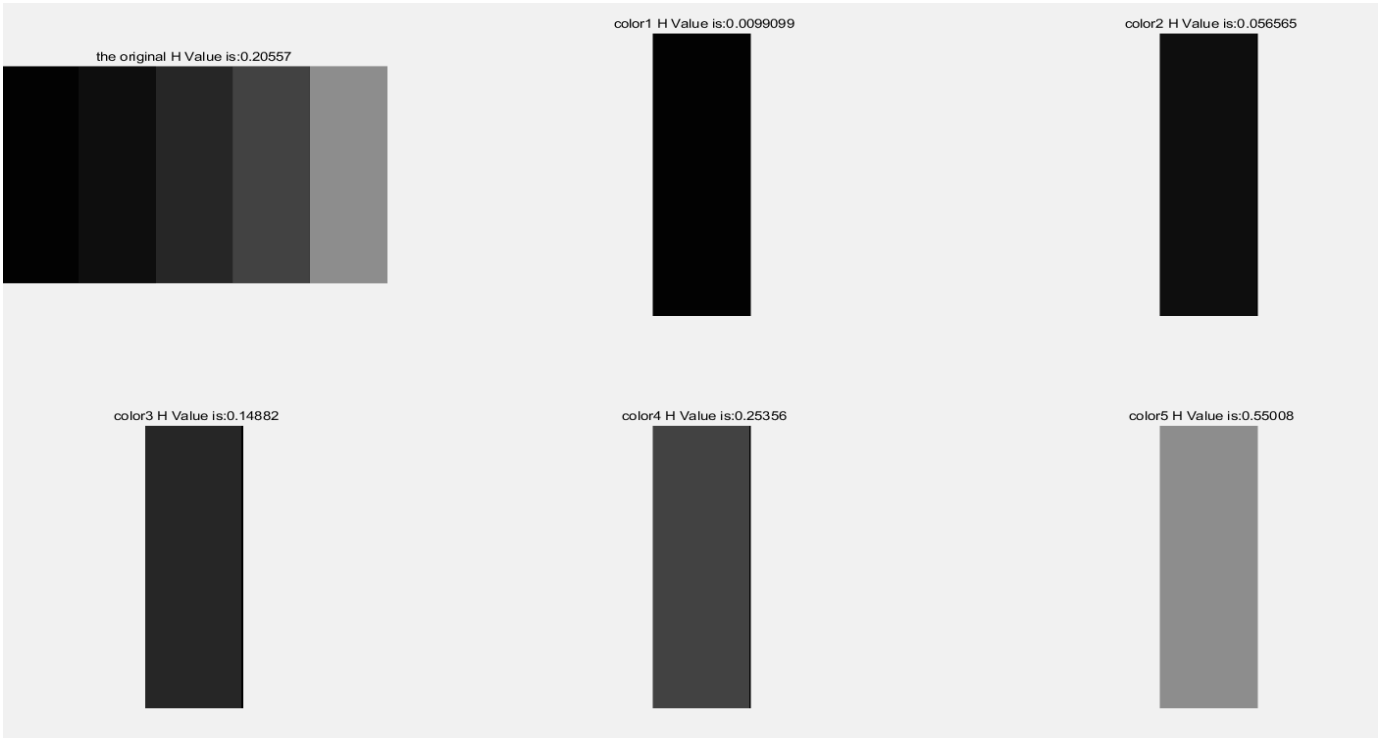
The main task of this question is translate RGB to HSV.
In class, there is a formulation shown how to translate RGB to HSV. However, the formulation in class is not complete.
From the resources on the Internet, there is a more detailed formulate as the image shown on the left. Based on the completed formulation the own function cvRGB2HSV() can be built. Then, use the function to translate the RGB image to the HSV image. The each channels' results are as follows:



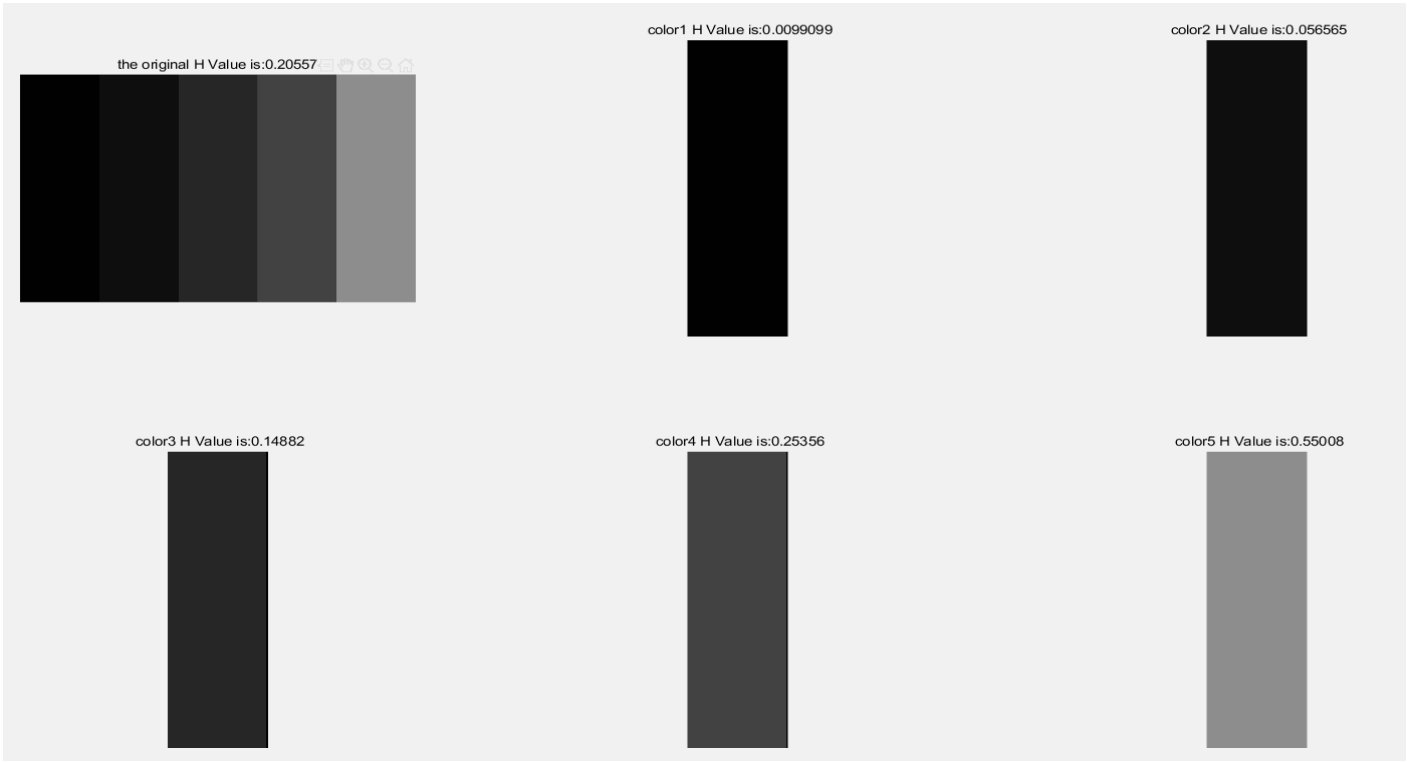
2. Compute the average H values of five color regions in Fig.1(b) with your function and the MATLAB's inbuilt functionrgb2hsv(). Print both of them under the corresponding regions (0.2′ for each value, 1′ in total). You also need to explain how to distinguish and divide the five regions, and how to calculate the average Hue value

The main task of this question is how to distinguish and divide the five regions, and how to calculate the average H value. The first step is changing the RGB image to HSV image and choose the H value part to complete this question. Secondly, because the color of the image is evenly distributed in a color block, the edge of each color block is easily be found which means in a row of the H value matrix if one data is not same as the number before it and not same as the number after it this data's location is the edge. Firstly, only choosing one row(such as the row tenth) in the H matrix. Then, building a cycle to read the data from head to tail of this row, if the number is the head, pop the column number in an empty matrix(such as k); if the number is equal with the number before it but not equal with the number after it, pop the column number in k as the tail of one color block; if the number is not equal with the number before it but equal with the number after it, pop the column number in k as the head of one color block; if the number is the last one, pop the column number in k as the tail of the last color block. By doing the algorithm like this, each color blocks' head and tail range can be found. By using these data each color block can be cut from the original matrix into five different new matrixes. The way to get the average of each color block is using the mean() function in MATLAB twice on matrixes. By using the function rgb2hsv() in MATLAB can get the MATLAB result to make a compare. Based on comparison, own function is work as same as the MATLAB function on this situation. The results are as follows:

The own cvRGB2HSV() results



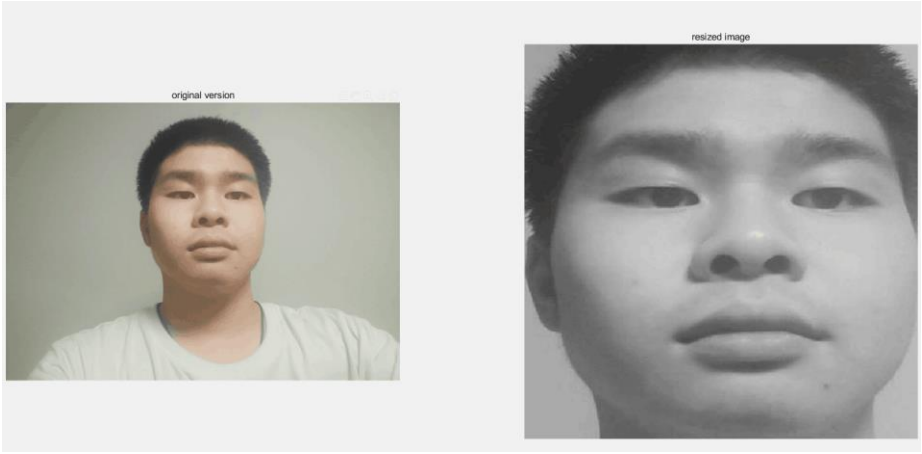
The MATLAB function results



Task5: Image Denoising via Gaussian Filter (4 marks)

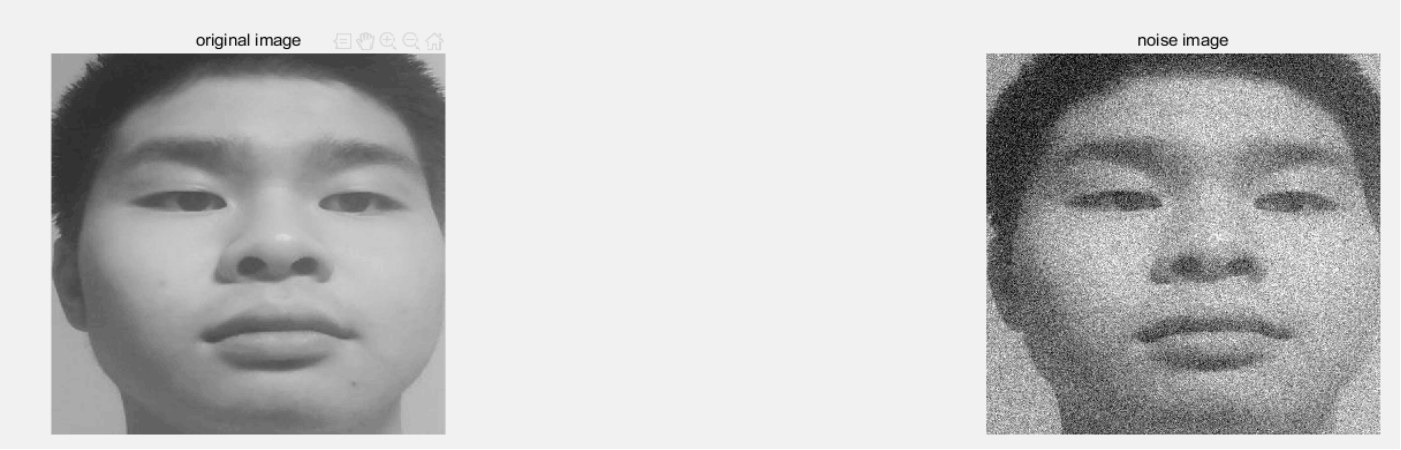
1.Read in one of your colour face images. Crop a square image region corresponding to the central facial part of the image, resize it to 512×512, and save thissquare region to a new grayscale image. Please display the two images. Makesure the pixel value range of this new image is within [0, 255] (0.5′).

By using the function of imcrop() in MATLAB the central face can be cropped as a new rgb image. Then, by using the function rgb2gray() in MATLAB the face image can be changed to grayscale image. Lastly, by using the function imresize() in MATLAB can make the image be a 512×512 square image. The result is as follow:



2. Add Gaussian noise to this new 512×512 image (if you are using Matlab, youcan use functionimnoise()). Use the following Gaussian noise with zero mean,and standard deviation of 30 (0.5′).Hint:Make sure your input image range is within [0, 255]. Kindly, note thatMatlab function imnoise() normalizes the input in the range [0,1] by default. So,normalize your variance accordingly like variance = (standard deviation2)/(2552)if you are usingimnoise()in default setting.

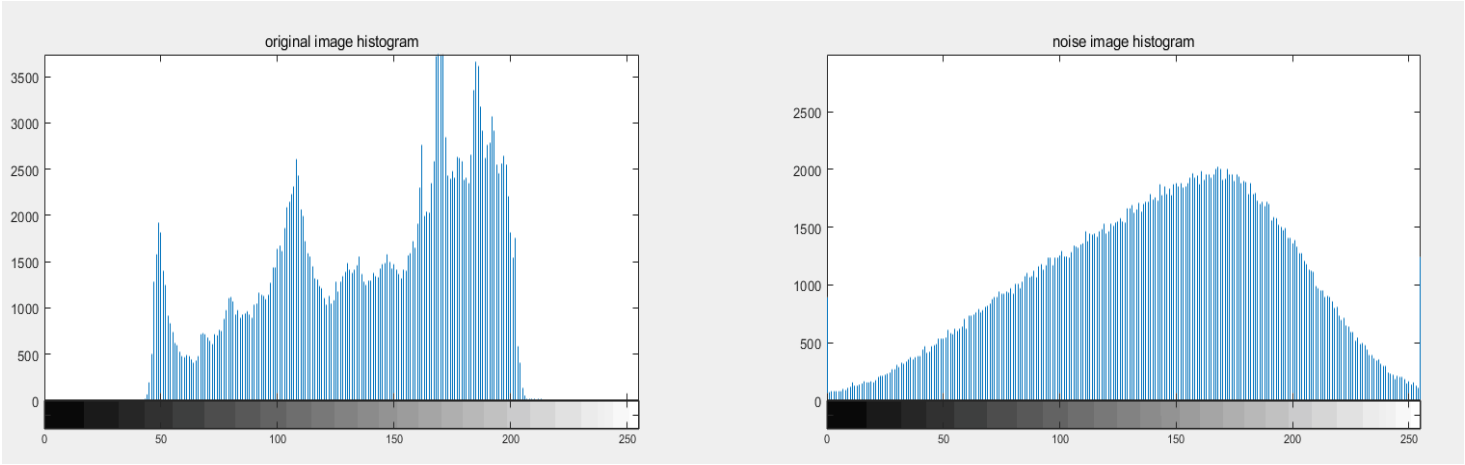
In MATLAB the function imnoise() can be used to solve this question. Because the function imnoise() not need the standard deviation but need the variance, translating standard deviation of to 30^2/255^2 is useful. Finally, by using the function like imnoise(imagename,'gaussian', 0, 30^2/255^2) can get the result as follows:



3. Display the two histograms side by side, one before adding the noise and oneafter adding the noise (0.25′for each histogram, 0.5′in total)

By using the function subplot() in MATLAB can let two histograms printed in one figure side by side. Using the original image data and the noise add image data from the second question and the function imhist() in MATLAB is the way to get the two histograms plot. The results are as follows:

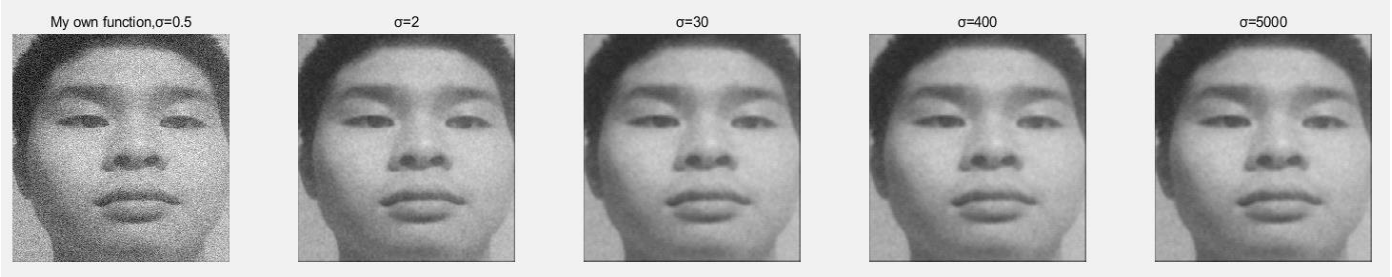
X value is the gray level; Y value is pixel number



4. Implement your own Matlab function that performs a 9×9 Gaussian filtering (1′).Your funct ion interface is:outputimage=myGaussfilter (noisyimage, my9×9gausskernel)Apply your Gaus sian filter to the above noisy image, and display the smoothedimages and visually check their n oise-removal effects (0.5′in total).

4One of the key parameters to choose for the task of image filtering is the standarddeviation of your Gaussian filter. You may need to test and compare differentGaussian kernels with differe nt standard deviations (0.5′).Note:In doing this task you MUST NOT use any Matlab's inbuilt i mage filteringfunctions (e.g.imfilter(),filter2(),conv2(), orfilter(),conv()). In other words,you ar e required to code your own 2D filtering matlab code, based on the originalmathematical defini tion for 2D convolution. However, you are allowed to useMatlab's functionfspecial()to generat e a 9×9 sized Gaussian kernel.

In matlab the function fspecial() can be used as fspecial('gaussian',[9,9],0.5) to get the 9×9 sized Gaussian kernel. The idea of building the own outputimage=myGaussfilter (noisyimage, my9×9gausskernel): the first step is to make the noise image as a double matrix.
In the second step, the center of the convolution kernel(such as abcd) is aligned with the first element of x, and then the corresponding elements are multiplied and if there are no elements because of the edge, building a limitation let the edge pixel not in the cycle.
In the third step, each element is calculated like this, get an output matrix, which is the convolution result. By using fspecial() several times the different standard deviations can be compared. Based on comparison, the Gaussfilter works better with the increase in variance. The results are as follows:



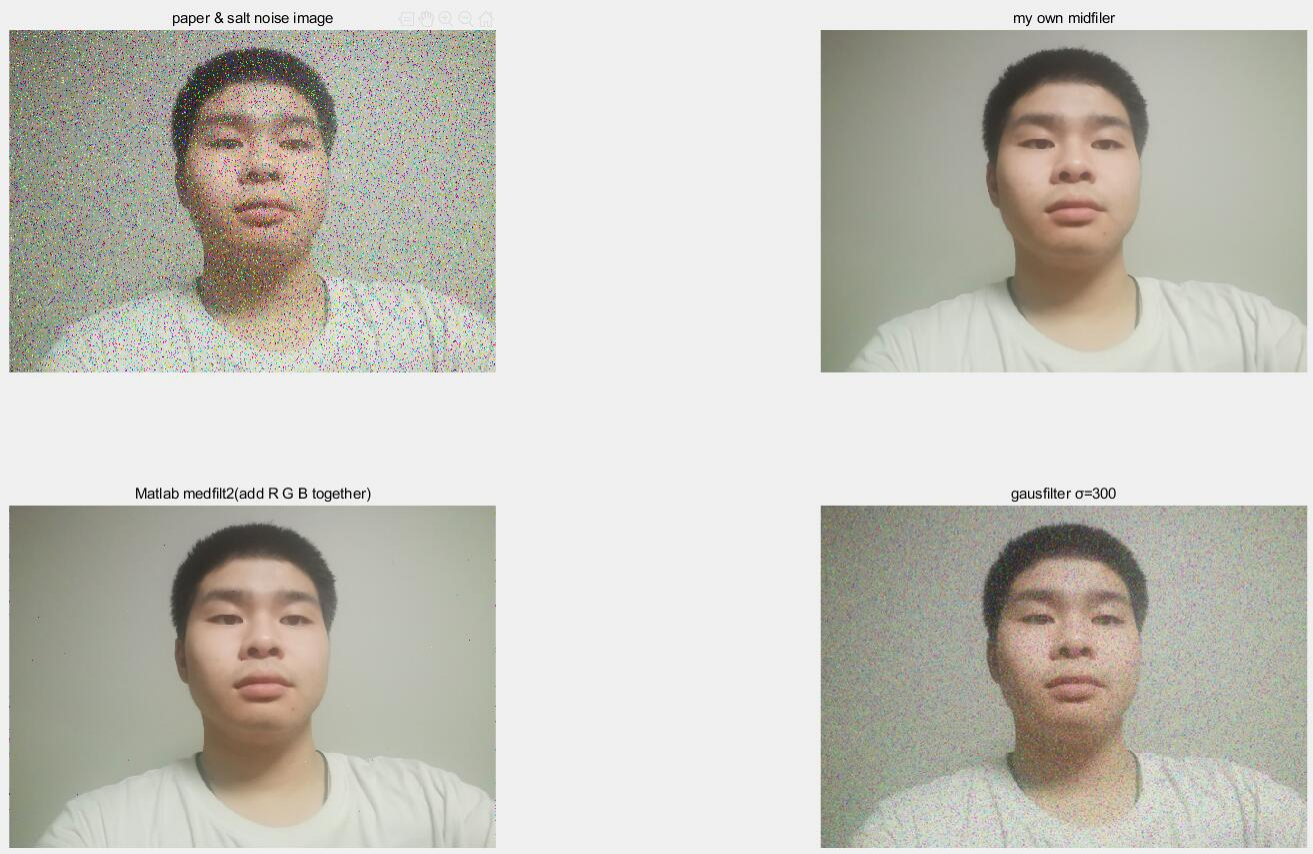
5. Compare your result with that by Matlab's inbuilt 9×9 Gaussian filter, filter2(), orimfilter(). Please show that the two results are nearly identical (0.5′)

By using function imfliter() in MATLAB and my own function, the results can be compared. Using the MALTLAB function can process the image all well but my own function cannot work very well on the edge of image. The comparable results are as follows:

Task6: Implement Your3×3Median and Sobel Filter (4marks)

1.Median filter. Add 10% salt and pepper noise to your original colour face image,e.g., to randomly pick 10% of the image pixels, and change their values to eitherpure white or pure black at 50-50 chance. Implement your own 3×3 MedianFilter, and use it to denoise your noisy face image. Display the results (1′), andthen compare your result with Matlab′ s inbuilt 3×3 median filter (0.5′). Whichfilter (Gaussian or Median) is more suitable for removing salt-and-pepper noise(0.2′)? Why (0.8′)?

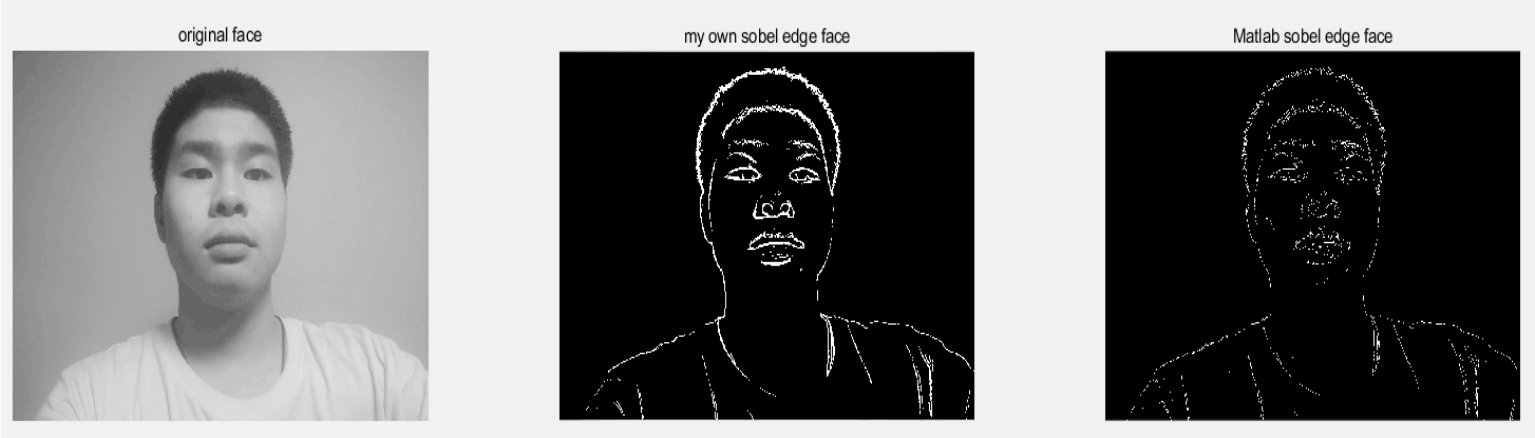
In MATLAB, the function imnoise() can do good work on adding salt-and-paper noise on 10% pixel of images with 50-50 chance of black and white by using it as imnoise(face,'salt & pepper',0.1).
To make all the results can be shown as RGB image, every R, G, B channels are processed one by one and then add together to get the color image.
My own filter function is working in these steps: Firstly, change the image matrix into double style. Secondly, building the cycle from 1 to the size of image minus the size of kernel plus 1. Thirdly, in the cycle, find the media pixel be the new pixel data of the convolution part's central data. Finally, change it into uint8 style matrix image. By using suplot() function in MATLAB, the results can be compared. Based on the comparison, my own function's result is as same as the MATLAB function's result while my own function's result is a little bit lighter than the MATLAB function's result.
By using the function imfilter() in MATLAB, the σ =300 Gaussian filter result is gotten. Obviously, the Median filter is working better than the Gaussian filter on solving salt-and-paper noise. The reason why this situation happens is that salt-and-pepper noise is approximately equal in amplitude but randomly distributed in different positions, there are both contaminated and clean points in the graph.



Median filter is one of the most commonly used nonlinear smoothing filters, which can eliminate isolated noise points in the image and produce less blur. However, Gaussian filter is the process of weighted average of the whole image. the value of each pixel is obtained by weighted average of itself and other pixel values in the neighborhood so that Gaussian filter not good at solving salt-and-paper noise. The results are as follows:

2.Sobel edge detector. You need to implement your own 3×3 Sobel filter. Again, you must not use MATLAB's inbuilt edge detection filter. Test it on your face images(suggest to convert it to gray-scale) (1′), and compare your result with MATLAB's inbuilt Sobel edge detection function (0.5′).

To build own Sobel edge detector function, building a cycle to processing of the image gray value is necessary. There are formulas such as:
Gx = [f(x+1,y-1)+2*f(x+1,y)+f(x+1,y+1)]-[f(x-1,y-1)+2*f(x-1,y)+f(x-1,y+1)]
Gy = [f(x-1,y-1) + 2f(x,y-1) + f(x+1,y-1)]-[f(x-1, y+1) + 2*f(x,y+1)+f(x+1,y+1)]
Gx and Gy represent the gray value of the image detected by transverse and longitudinal edge detection, respectively. The horizontal and longitudinal gray values of each pixel of the image are calculated by formula G=sqrt(Gx^2+Gy^2) to calculate the gray level of the point. After doing that for each pixel in the image matrix, build a logic judgement that if the gradient G is greater than a certain threshold, which is 45 in my own function, the point (x, y) is considered to be the edge point. In MATLAB, the edge() function can do the sobel edge detector like edge(face,'sobel'). Base on the comparison, my own function totally find more edges than the MATLAB function did but the boundary sometimes lager than the MATLAB function. The results are as follows:

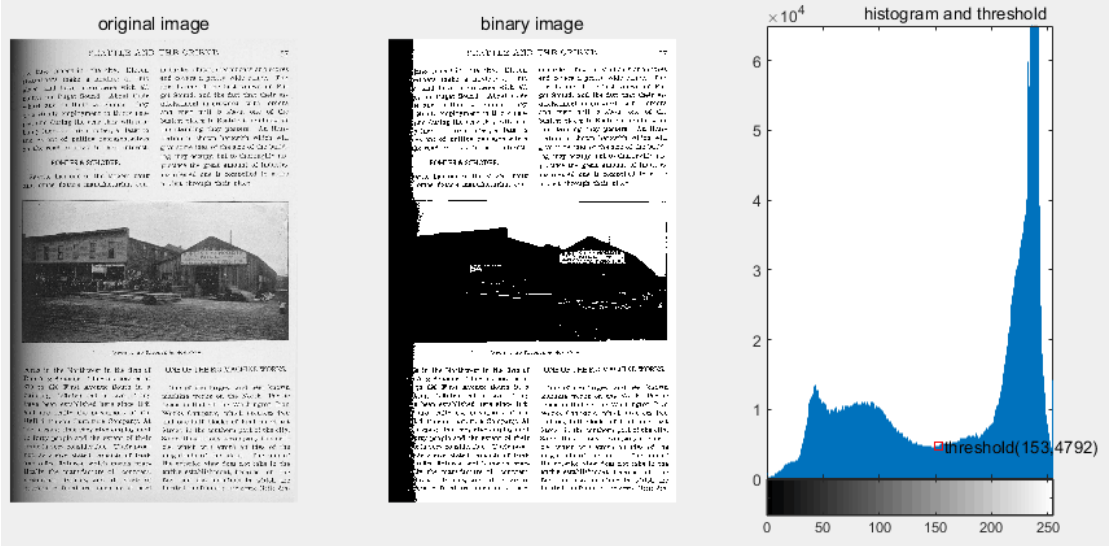


Task7: Image Morphology (4 marks)

The digital library contains large collections of digitized books. Some book pages needto be scanned. However, due to the curvature of the pages, there always exist unevenillumination in the scanned images, especially near the spine (seeing Fig.2(a)). Foroptical character recognition (OCR), the scanned images are first binarized and thenfurther processed by the OCR engine. This question aims to study how to eliminate theeffect of uneven illumination in binary images.

1. Read in the image file ”bookpage.jpg”(the image is from the Stanford Univer-sity).
2. Generate a binary image by performing global thresholding. You can use theMatlab’s inbuilt function graythresh()to automatically choose the threshold. Dis-play the binary image (0.5′), and show the histogram of the original image’ s grayvalues and mark the threshold of the chosen threshold on the histogram (0.5′).

Using function imread() in MATLAB can read the image. In MATLAB, function graythresh() can find an image’s threshold such as using like graythresh(image). In MATALB the function imbinarize() can make an image’s binary image by using the threshold found before. By using the function imhist() can get the images’s histogram and using plot() and text() function can point out the point. Results are as follows:



3. Design and implement your own algorithm that can eliminate the invisible textregion effect caused by global thresholding (2′, the final mark is mainly depen-
dent on the quality of the binary image. One desired output is shown in Fig.2(b)for reference o nly.)Hint:One possible solution is to perform local thresholding. More specifically,(1) use a hor izontally sliding window; (2) within each window, compute the localvariance of grayscale value s; (3) manually set a thresholdTv; (4) if the localvariance exceedsTv, binarize the local region withgraythresh(). Feel free todesign your own algorithm; if so, please specify the main steps.

SEATTLE AND THE ORIENT. 57

the busy places in the city. Eleven steamboats make a landing at this place, and have intercourse with all points on Puget Sound. About their wharf and in their warehouse they give steady employment to thirty people, and during the year they will probably increase this number, at least to the extent of putting representatives on the road to travel in their interest.

ROHLFS & SCHODER.

Seattle has one of the largest bank and office fixture manufacturing concerns in the Northwest in the firm of Rohlfs & Schoder. They are located at 610 to 620 First Avenue South in a building 150x150 feet in size. They have been established here since 1889 and are really the successors of the Hall & Polson Furniture Company. At the present time they give employment to forty people and the extent of their trade is very considerable. Their product, as above stated, consists of bank and office fixtures, which means practically the manufacture of counters, steamboat fixtures, and all kinds of stationary furniture, such as is used in banks, offices, steamboats and stores and covers a pretty wide range. The firm is one of the best known on Puget Sound, and the fact that their establishment is crowded with orders and their mill is about one of the busiest places in Seattle is evidence of the standing they possess. An illustration is shown herewith which will give some idea of the size of the building they occupy, but to thoroughly appreciate the great amount of industry manifested one is compelled to make a visit through their place.

FACTORY OF ROHLFS & SCHODER.

ONE OF THE BIG MACHINE WORKS.

One of the largest and best known machine works on the North Pacific Coast is that of the Washington Iron Works Company, which occupies two and one-half blocks of land on Grant Street, in the southern part of the city. Some illustrations accompany this article, which will afford an idea of the magnitude of the plant. The one of the exterior view does not take in the entire establishment, because of the fact that the buildings in which are located the foundry are some little dis-

To solve the invisible text region problem on the binary image, the most important thing is to remove the shadow on the original image. One of the possible solutions is Homomorphic filtering. Here are steps to make a Homomorphic filtering. Firstly, logarithm the image matrix (such as I2) as log(I2+1) to avoid the situation such as log(0). Secondly, using function fft2() to get a Fourier transform. Thirdly, building a row × column cycle with the formulate:

$$H(u,v)=(\gamma_H-\gamma_L)\left[1-e^{-c\left[D^2(u,v)/D_0^2\right]}\right]+\gamma_L$$

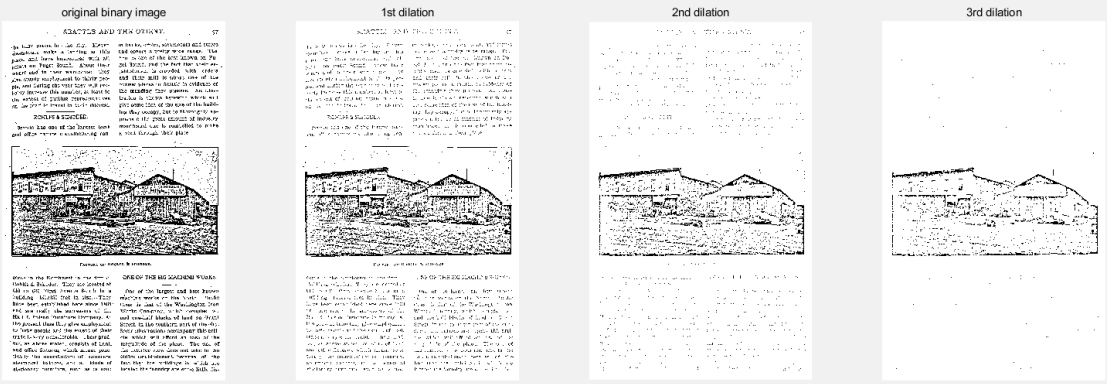
From: https://blog.csdn.net/cjsh_123456/article/details/79351654

can get a Gaussian high-pass filter to process the image matrix. Lastly, make the Gaussian high-pass filter result has an inverse Fourier transform to get a image matrix with less shadow. To get the image the question want, should find a new
threshold for the minus of the processed image.
Finally, using medfilt2() function remove the noise and using the imcomplement() to get image the question request. The result is shown on the left.

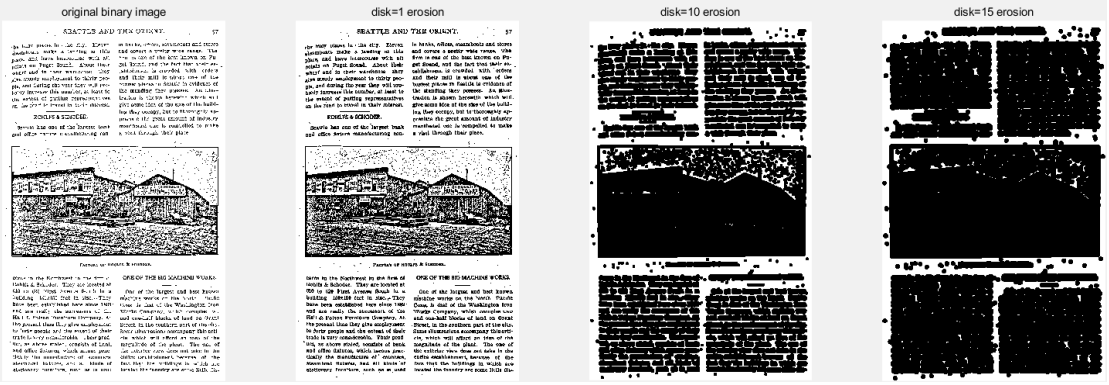
4. Test the effects of applying Matlab's inbuilt morphological operators of 'erosion','dilation', 'opening', and 'closing' to the binary image, and visually comparethem in your report (0.25′fo r each image, 1′in total)

In MATLAB, by using the function imdilate() can do dilation on the image; by using the function imerod() can do erosion on the image; by using imopen() and imclose() can do open and close on the image. The results are as follows:

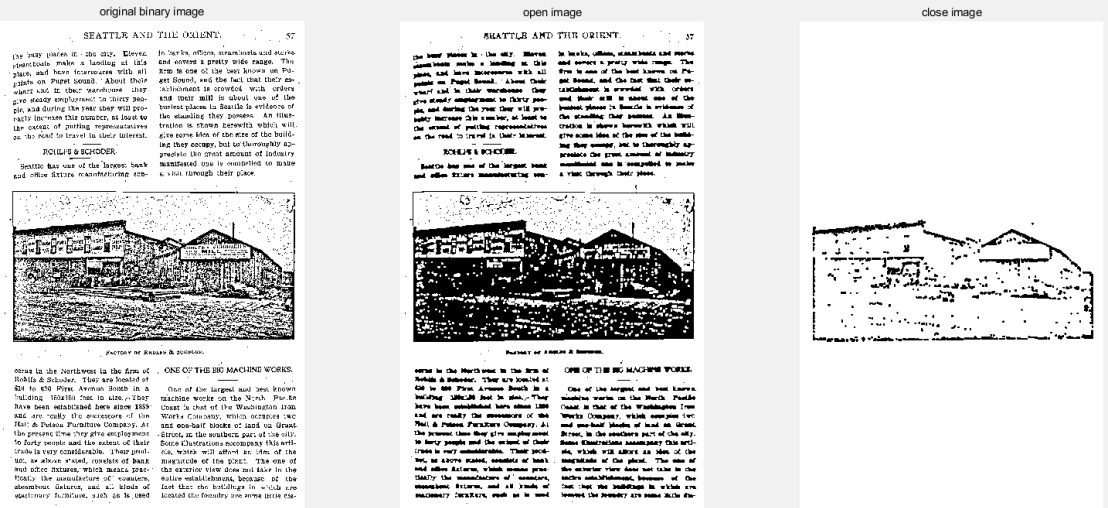
Dilation 3 times results: 1 2 3



Erosion with disk=1, 10 & 15 results:



Opening and closing results: open close

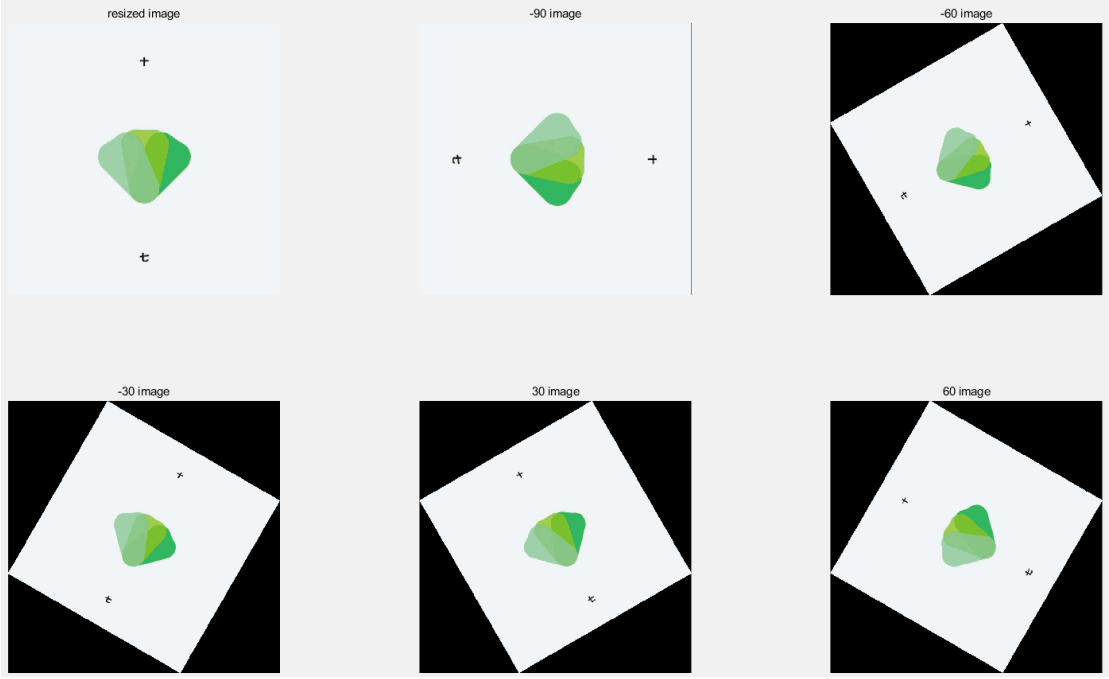


Task8: Image Rotation (3 marks)

Choose one of your favorite images (such as Fig. 3) and resize it to512×512,

1. **Implement your own functionmyrotation()for image rotation by any given anglebetween [−90◦,90◦]. Display images rotated by −90◦, −60◦, −30◦,30◦, and60◦(0.2′ for each image, 1′ in total).**

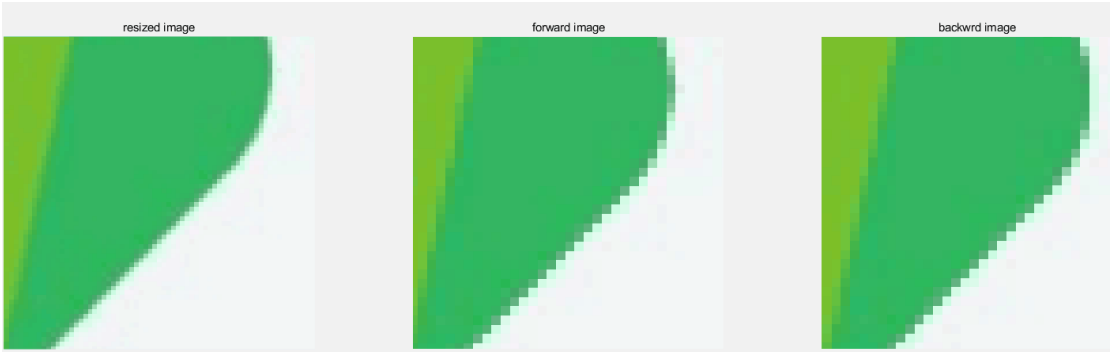
   The first step to do own my_rotataion() function is calculate the transformed coordinate system and transform it. Then translate the angle should do to the matrix. After that, use the cycle to translate every pixel in the image and use function round() to avoid sometimes some holes appear. The results are as follows:



2. **Compare forward and backward mapping, and analyze their difference (1′).**

Base on the comparison, comparing the same part of the image's pixels can find some differences. Firstly, the original image has much more detail than the forward and backward. Secondly, the backward image keeps more information than the forward one such as a line of pixel and backward image is sharper. The original image, forward image and backward image are as follows:

original image forward image backward image



3. Compare different interpolation methods, and analyze their difference (1′).Hint:When analyzing the difference, you can focus on: (1) visual results; (2)the principles in terms of formulation or others relevant; (3) advantages and draw-backs; (4) the computational complexity. You can also think about it from otheraspects.

In this task, two interpolation methods, nearest neighbor method and bilinear interpolation method will be compared. Comparing the vision results, the two different methods both make some information change but the nearest neighbor method will lose more information. In the view results, the original image uses five line of pixel to show a cross while bilinear interpolation method uses three line of pixels and the nearest neighbor method only use two line of pixel to show the cross. These two methods' computational complexity is totally same, such as the depth of cycle they use is same so that the running time is even no difference. However, the bilinear interpolation method need more logical judgements than nearest neighbor method which is a part that more complex. The nearest neighbor method is a simple image scaling algorithm, the effect is also the worst, the enlarged image has a very serious mosaic, the reduced image has a very serious distortion because some float numbers are ignored. The bilinear interpolation method also has some negative, for instance if some pixels are rounded by the noise pixels this method will enlarge the noise sizes. In conclusion, nearest neighbor method is usually worse than bilinear interpolation method but still have advantages on running time and others. The image results are as follow:

original image nearest neighbor bilinear interpolation