AE2IIPCoursework Report

Task1

To implement a face detector, it is necessary to obtain a cluster of face pixels first. Then, compared the value of image pixels with the threshold in the clusters. If the value within the threshold, set to 255 and if not, set to 0 (for HSV, the value from 0 to 1). After getting the detected image, some morphological operations need to be done to make results better.

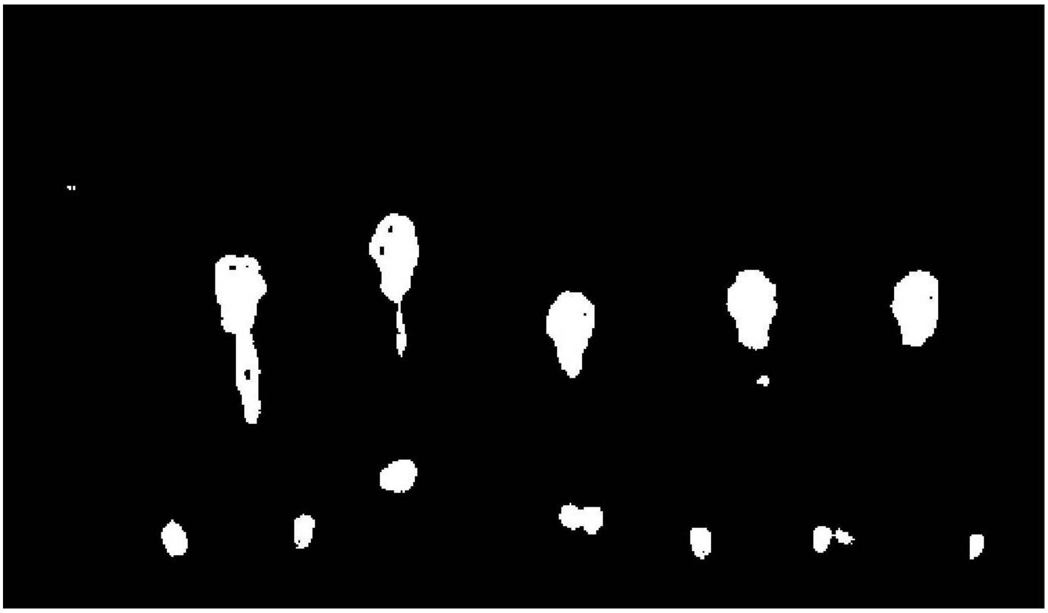
For task1, I choose YUV color space to get the clusters of face pixel for face1.jpg and HSV color space for face2.jpg. After reading the input image, use rgb2ycbcr (or rbg2hsv) function to convert RGB to YUV (or HSV) and obtain Y (or H) and U (or S) intervals respectively which is easy to implement. Then calculate the thresholds for the Y and U intervals separately and save it in the faceModel array. To make the detect result better, there are two sample images which is used to get threshold. The specific method is obtaining the threshold of the two pictures and take the union of these two thresholds. Having got the cluster of face pixels, the algorithm will read the image needs to be tested, get its gray scale image and covert RGB to YUV (or HSV). Then, the values of each pixel value in U (or H) space and V (or S) space are compared with the threshold obtained previously. If both U (or H) and V (or S) space are within the thresholds, set this pixel value to 255, otherwise set to 0. After comparison, morphological opening operation is applied and then do morphological closing operation to make the result looks better.

For the cluster of pixels, the algorithm uses a 1 X 4 array to save it. cluster1 = [U_Max, U_min, V_Max, V_Min], cluster2 = [H_Max, H_Min, S_Max, S_Min].

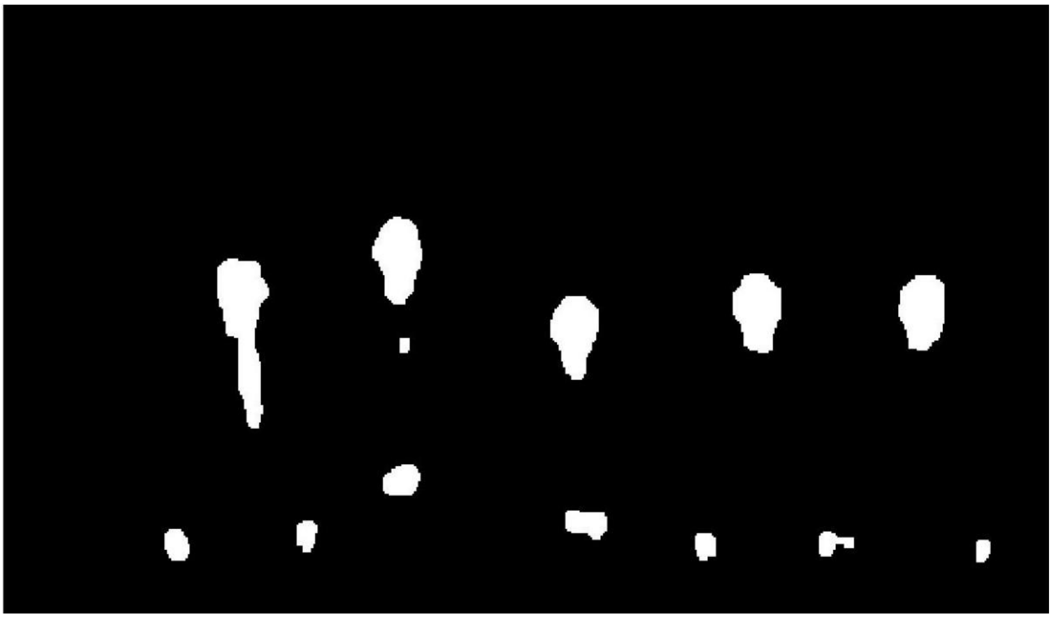
For face1.jpg:

The cluster1 is [120,88,162,139]

The image before morphological operations:



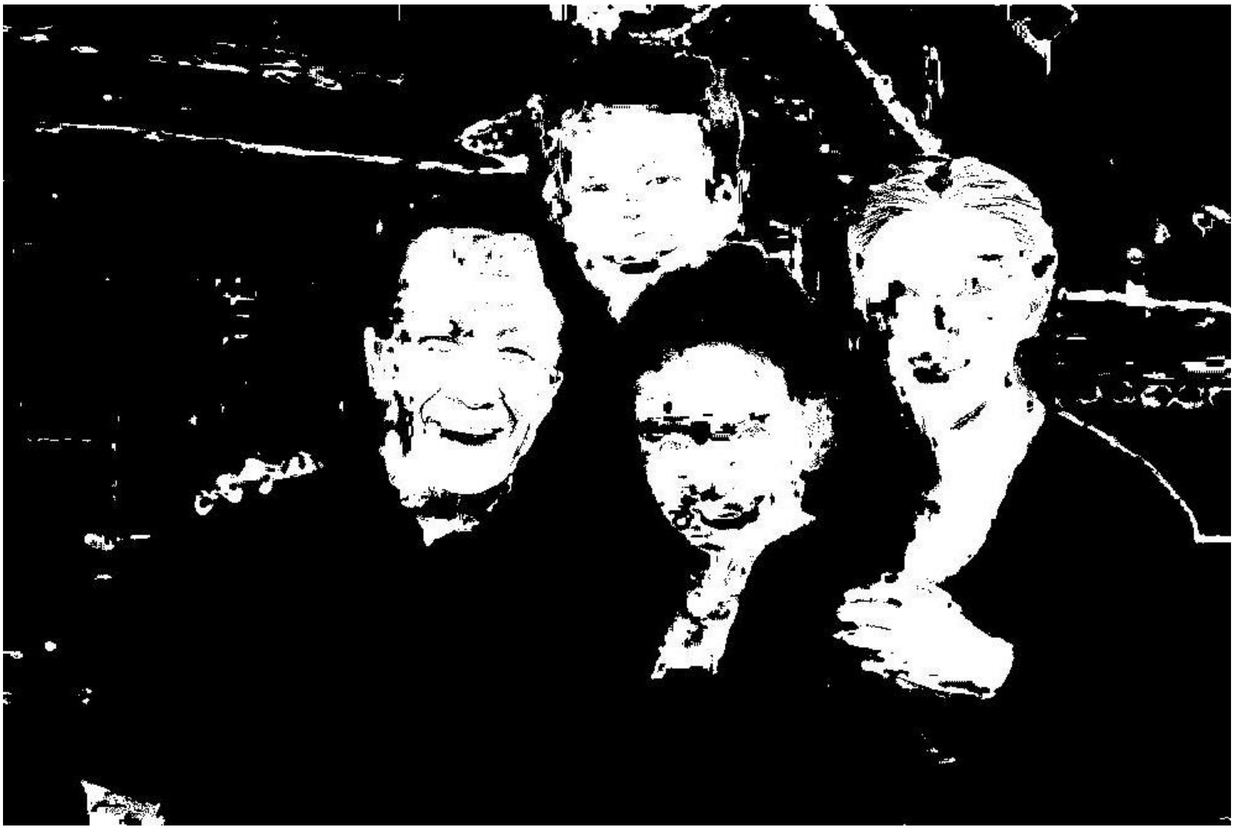
The final binarized image:



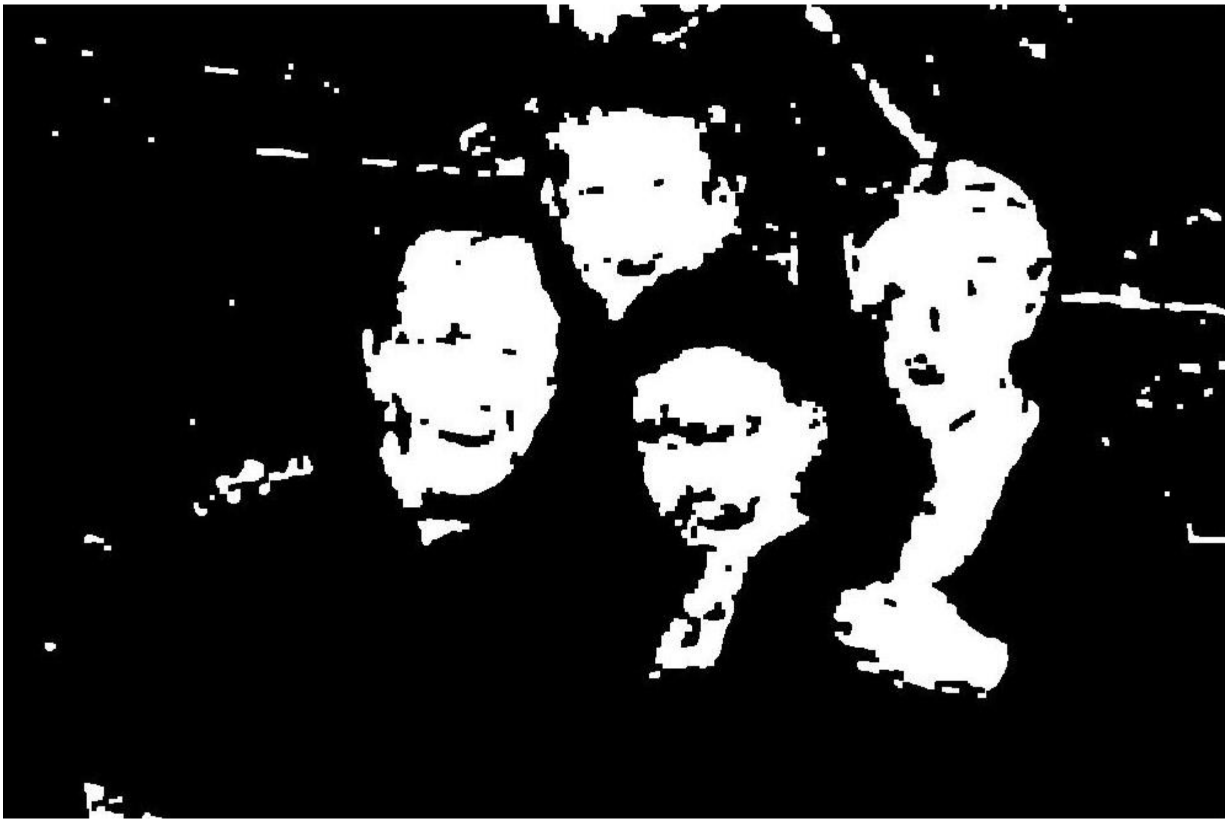
For face2.jpg:

The cluster2 is [0.0920,0.0267,0.4330,0.0988]

The image before morphological operations:



The final binarized image:



As you can see, after morphological manipulation, the final image looks better.

Task2

Part1 Filter

Mean Filter

Mean filter is a kind of linear filter whose output is a sample average value of the pixels in the neighborhood template. In this algorithm, 3 X 3 mean filter is implemented. When the filter working, it will calculate the sum of the value of all the pixels in the 3 X 3 template. Then divide the sum by 9 and this value is the final value of pixels in the center of the template.

Gaussian Filter

Gaussian filter is linear smoothing filter and mainly used to eliminate gaussian noise. To implement gaussian noise, the size of the template and σ are needed to built the filter template. For the gaussian filter used in this coursework, the gaussian filter is set with 5 X 5 template and σ = 1. Then calculate the template by formula $p' = \frac{1}{\sigma^2 2\pi} e^{\frac{-x^2-y^2}{2\sigma^2}}$. Then use conv2(image, filter_template) function to process the image.

Median filter

Median filter is a kind of nonlinear filter and very useful to eliminate salt & pepper noise. In the algorithm, 3 X 3 median filter is implemented. It works by getting the median value of the template and set it as the center pixels value. To implement it, the algorithm traverses all the pixel on the image as the center of a 3x3 matrix and reset the value as the median value in the matrix.

Anisotropic filter

Anisotropic filter is also a nonlinear filter. To implement it, the similarity function (D-d) / (D+ 1e-10) is used. As the filter template 3 X 3, so the algorithm will obtain a 3 X 3 matrix from the image each loop until all the pixels in the image have been traversed. After obtaining the matrix, D can be got by calculating the maximum value in the matrix minus the minimum. Then calculate the similarity for each pixel in the matrix. To calculate the final value, the formula $p' = \frac{\sum q * s(p,q)}{\sum s(p,q)}$ is used while $s(p,q)$ is the similarity with the center of matrix p and q is the corresponding point. After traversing all the pixels, the algorithm will convert the matrix from double to uint8 and show it.

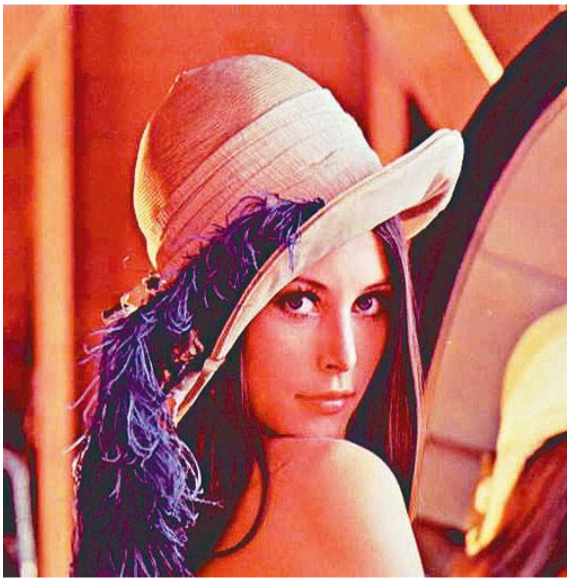
Bilateral filter

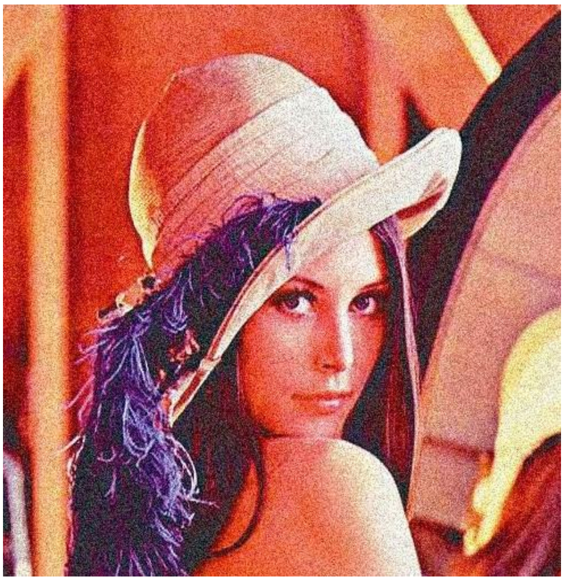
Bilateral filter is a kind of filter which can protect the edge while eliminate the noise. This filter is built by two function. The first function is used to determine the space rate by geometric space distance, another function is used to get the range rate by calculating the difference of pixels in the template. To implement it, the space function parameter is set to 1, the range function parameter is set to 10 and the size template is set to 5 X 5. Same with anisotropic filter, a 5 X 5 matrix is got from image each time to do bilateral filter. As the space rate is determined by geometric space distance, so the algorithm only need to calculate one time. However, the rage rate needs to be calculated each loop as the difference will changed. After getting the matrix, the algorithm will do calculation to get two filter coefficients and then multiply two coefficients to get the coefficient which will be used to calculate result. Finally, use formula to get the result and save it in corresponding position of the image matrix.

Part2 Distorted Images

Original Image and Distorted Images

Original Image, add Gaussian noise, add salt & pepper noise, Gaussian with 7 X 7 and σ=2



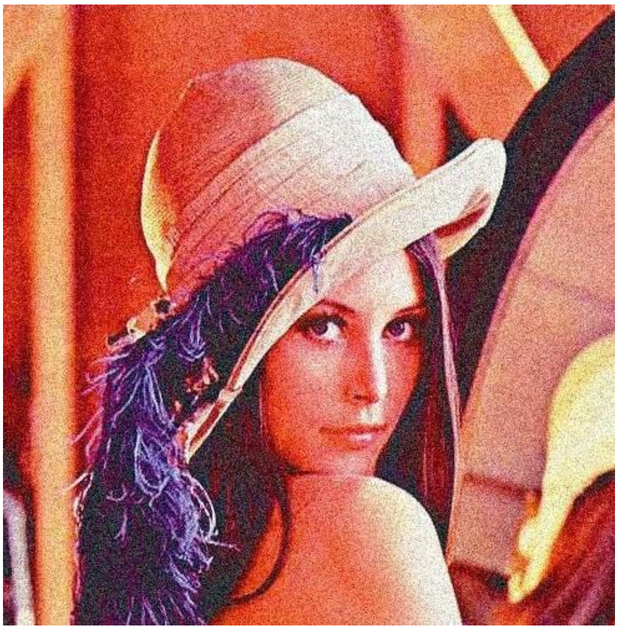


Part3 Distorted Images with five filters

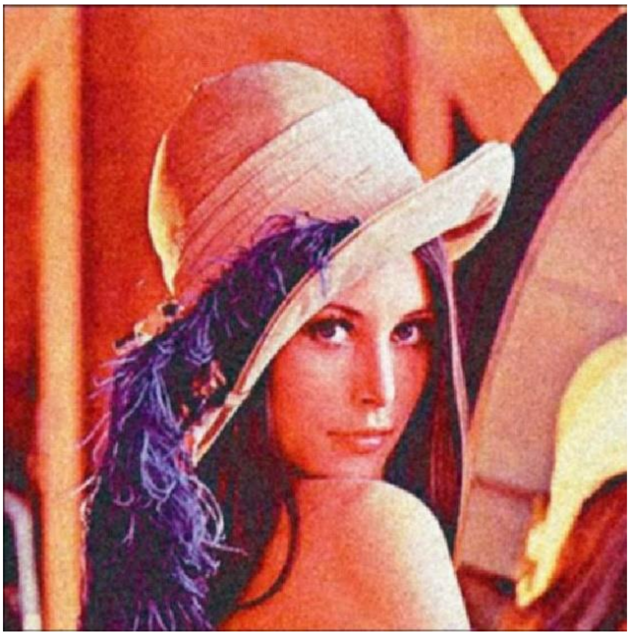
The filters is used in the following order : 3 X 3 mean filter, 5 X 5 Gaussian filter with σ=1, 3 X 3 median filter, 3 X 3 anisotropic filter, 5 X 5 bilateral filter with space sigma = 1 and range sigma = 10. The processed image is 1 to 5 and image 6 is the input image.

For Gaussian noise with 5 filter the result as follows:

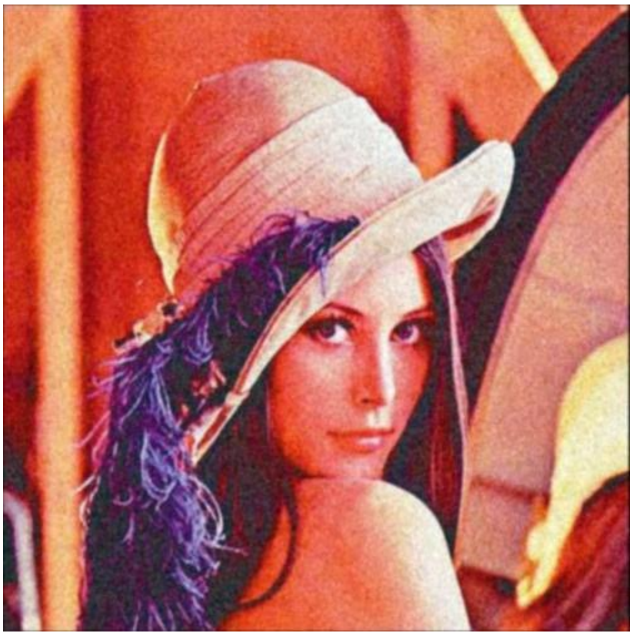
1 Unprocessed image 2 Mean filter



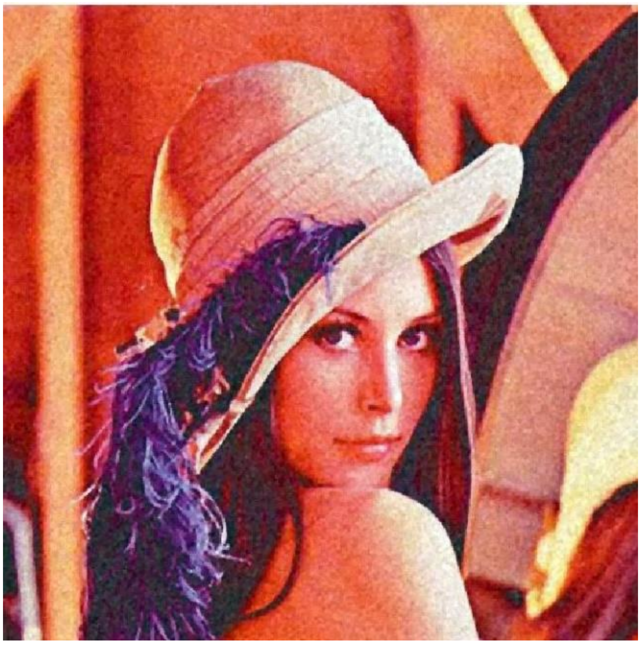
3 Gaussian filter



4 Median filter



5. Anisotropic filter

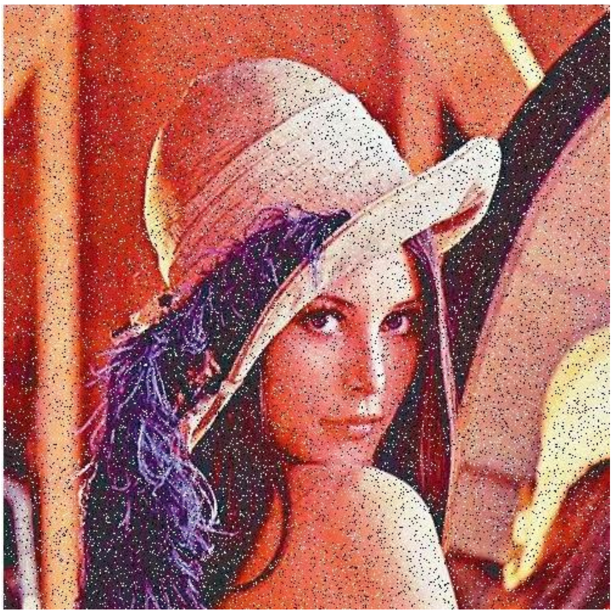


6. bilateral filter

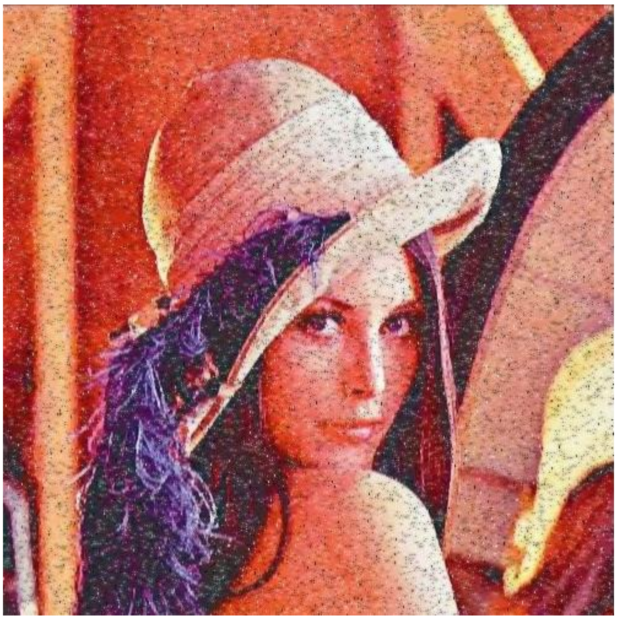
As the processed image shows, gaussian filter is better than other four filters when the image has a lot of gaussian noise. This may because gaussian filter is designed based on the same function as Gaussian noise which is helpful for estimating the Gaussian noise. In general, Gaussian filtering is the process of weighted averaging of the entire image.

For 10% salt & pepper noise image, the result with five filter as follows:

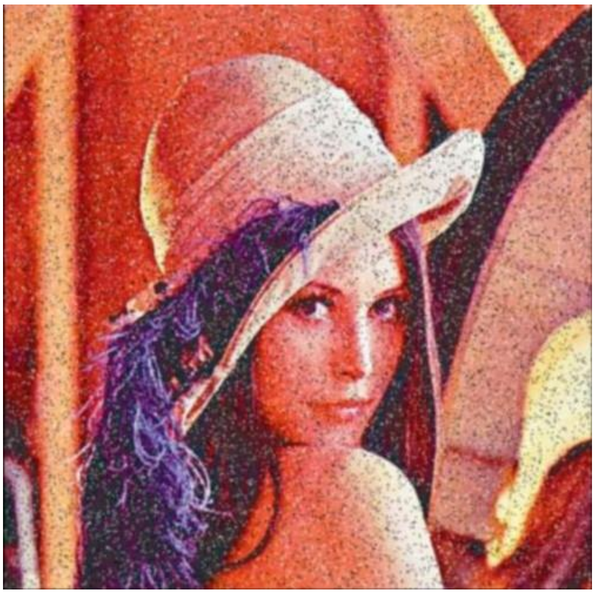
1 Unprocessed image



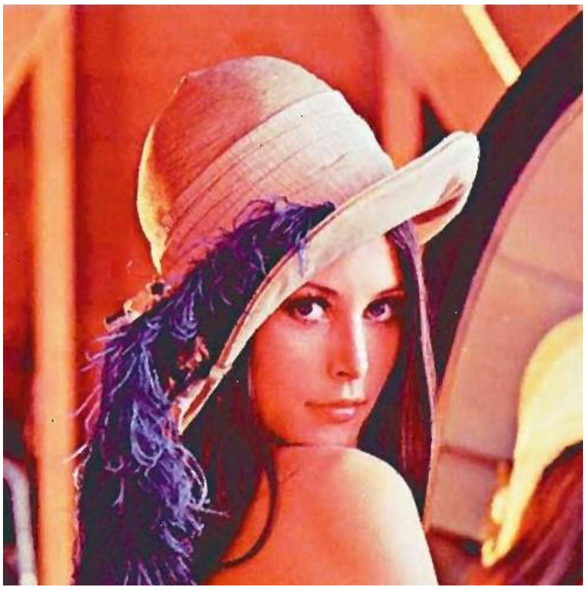
2 Mean filter



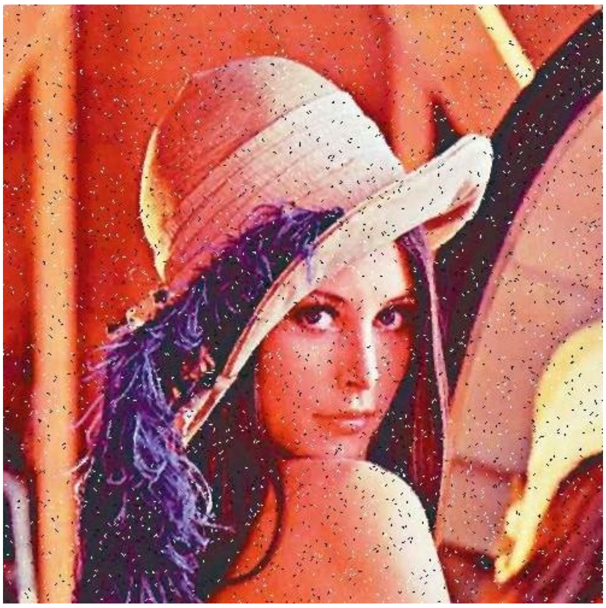
3 Gaussian filter



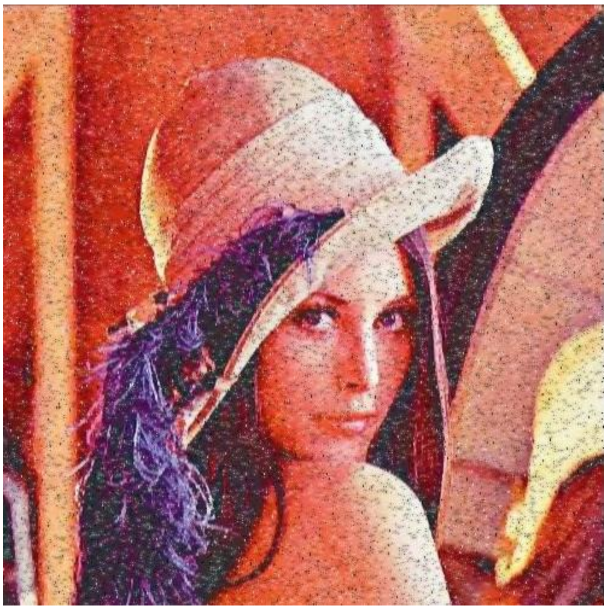
4 Median filter



5. Anisotropic filter



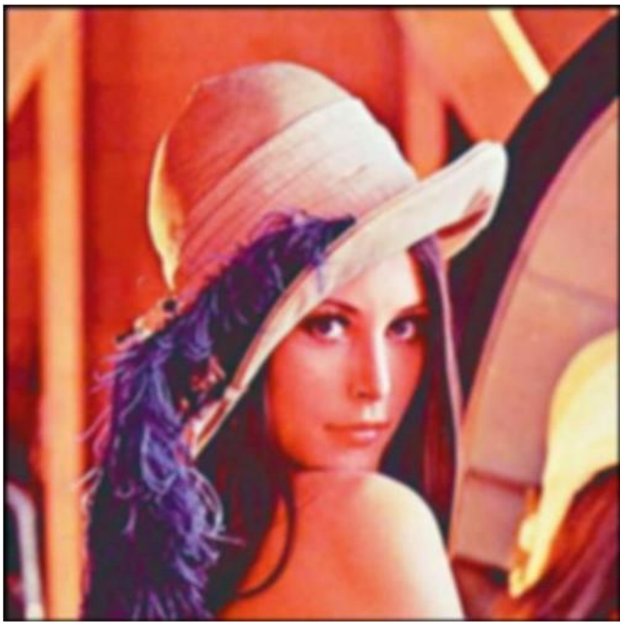
6. bilateral filter



As the result images show, median filter significantly is much better than other filters. The anisotropic filter performs can also reduce the salt & pepper noise but not good at median filter. Median filter mainly selects median value in its template as the pixel value, and for salt & pepper noise, the value is always 255 or 0 which is not easy to be selected. This may be the reason why median filter can eliminate noise very well

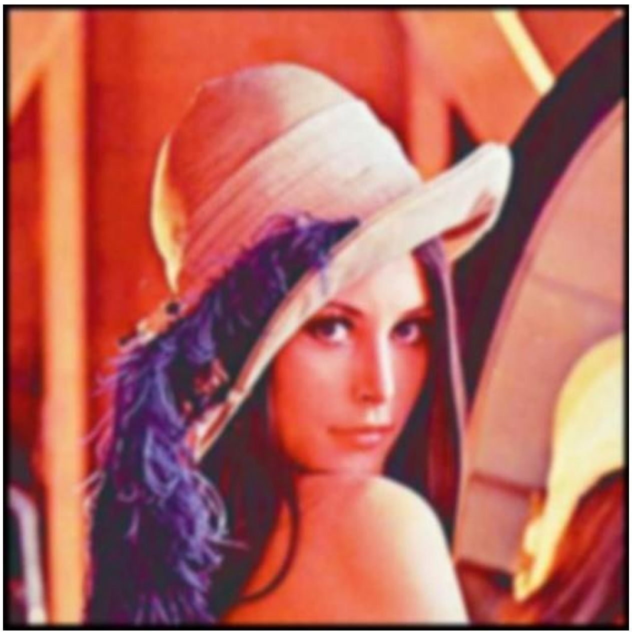
For imaged processed with Gaussian filter (7 X 7 and σ=2), the result with five filter as follows

1 Unprocessed image

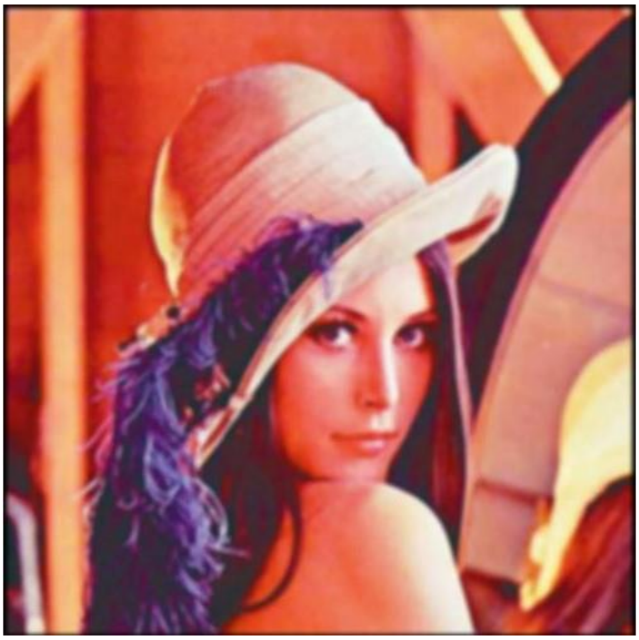


2 Mean filter

3 Gaussian filter

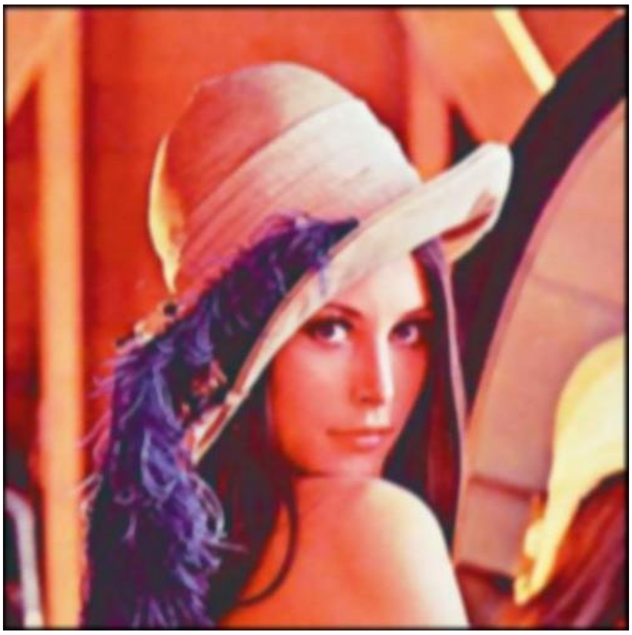


4 Median filter



5. Anisotropic filter

6. bilateral filter



For the image processed by Gaussian filter, it will be more blurred. If it processed by linear filter, the situation will get worse. As a result, for mean filter and Gaussian filter, the image will become more blurred and as the other three filters is nonlinear filter, the situation did not get worse. For bilateral filter, it has a good performance when retaining edges.

Task3

The algorithm explanation:

For task3, Frangi2D filter is used to process the image. Before using Frangi2D, serval operations are needed to make result better. After reading the image, convert the image to the grayscale image. Then use Gaussian filter to filter image. I use two Gaussian filter to improve the result. The first Gaussian filter is 5 X 5 size with delta = 0.4, the second Gaussian filter is 7 X 7 size with delta = 0.66. After these operations, the Frangi2D is used to get intermediate result. As the intermediate result is grayscale image and the result image is binarized image, the imbinarize function is used to. There is problem when use imbinarize, a threshold value is asked to provide and it not easy to find a perfect threshold. Finally, using morphological closed operation to get the final image.

The main Filter: Frangi2D

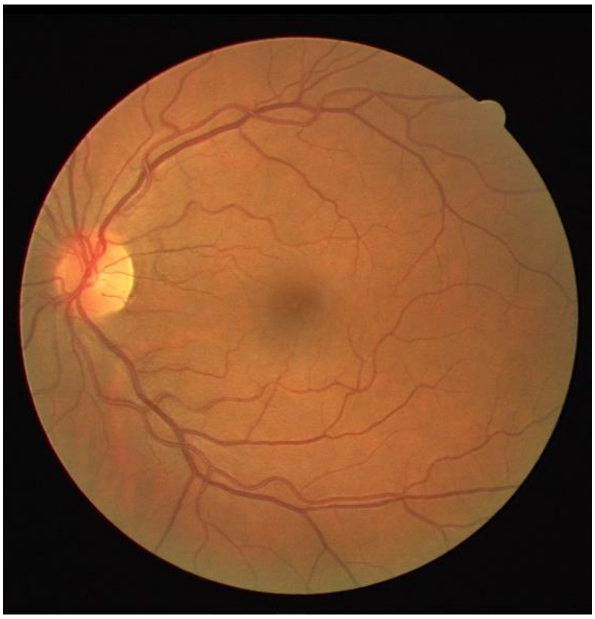
Frangi2D Filter is used to process the image and obtain the intermediate result. Basically, the codes for implementing Frangi2D Filter are from the Internet. The Frangi2D Filter operating principle as follows:

To implement Frangi2D filter, six parameters need to be set: FraniScaleRange, FrangiScaleRatio, FrangiBetaOne, FrangiBetaTwo, BlackWhite and verbose.

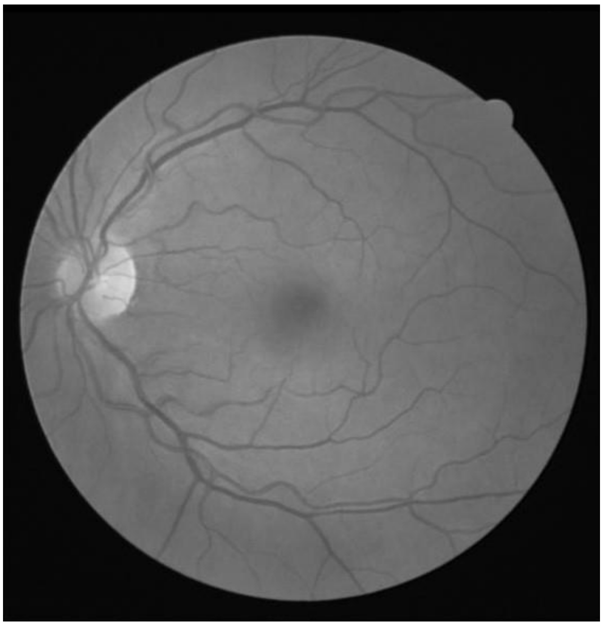
The FraniScaleRange represent the range of sigmas this filter used, the FrangiScaleRatio is the step size between sigmas, which means for the first two parameters, the larger the first parameter, the smaller the second parameter, the more the number of loops. To get a more accurate results, we can appropriately increase the interval of the first parameter and decrease the value of the second parameter. The FrangiBetaOne is used to adjust the sensitivity of distinguishing between block and strip, The FrangiBetaTwo is used to affect the overall smoothness of the filtered image. The larger the parameter, the smoother the result image. BlackWhite and verbose are used to change the result and detail display mode and will not affect the result image.

To implement Frangi2D filter, the Hessian matrix is needed to build first. The function Hessian2D(I,Sigma) is called to built Hessian matrix. By using Hessian2D function, the algorithm will get three the 2nd derivatives and then use the corresponding sigma, then the algorithm will use this three derivatives as input to call eig2image(Dxx,Dxy,Dyy) and calculate the eigenvalues and vectors. The eig2image function will return four values: Lambda1, Lambda2, Ix and Iy. For the retina blood vessel image, basically, the pixels in the image are roughly divided into three categories according to the image shape: background, isolated point and blood vessel. And with Lambda1, Lambda2 and formula $A=\exp(-\frac{R_b^2}{\beta^2})$, $B=1-\exp(-\frac{S_{\square}^2}{2c^2})$, the filter can recognize different types of points. Then the algorithm can compute the output image. Finally, the algorithm will return the maximum scale (sigma) value for each pixel.

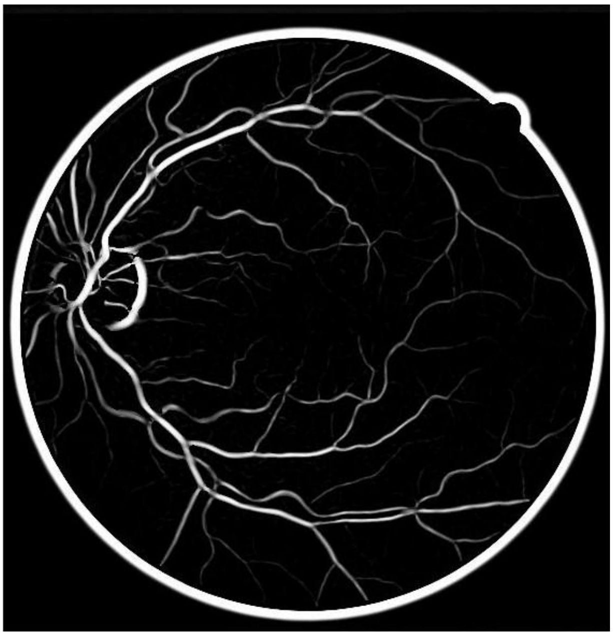
The original image:



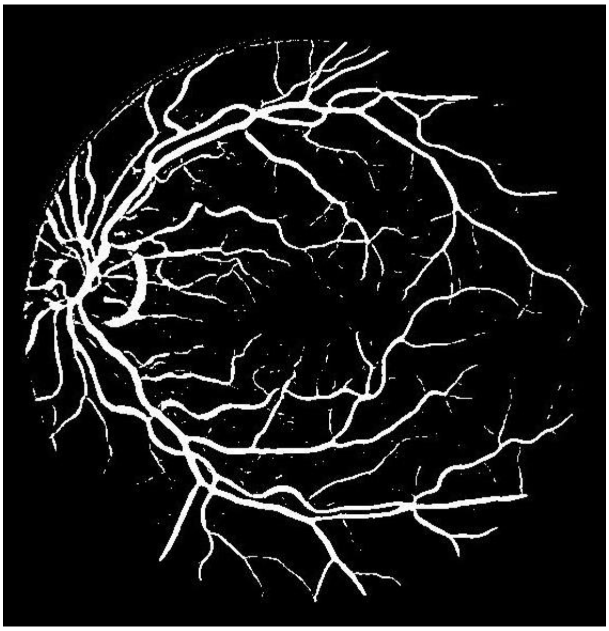
The grayscale image processed by Gaussian filter



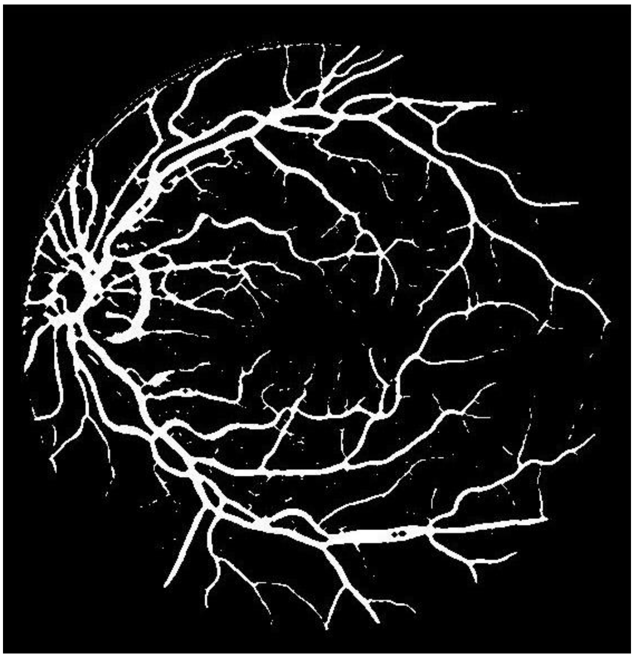
The intermediate result (After frangi2D)



The intermediate result (binarized)



The final result



The per-pixel level segmentation results

P: 0.8208

N: 0.9589

T: 0.9408

The advantage and disadvantage of the algorithm

This algorithm uses frangi2D filter as main filter to process the image. This filter has a high performance in detecting retinal blood vessels if the parameter is well set. This filter use Iterative method to calculate result, so the accuracy of the result is high.

The disadvantage: As Hessian matrix is very sensitive to noise, before use this filter, serval filters may be used to remove noise. Also, there are too parameters that need to set manually and it is very difficult to find a perfect combination of parameters, as a result, it not easy to get the best result.