EE 475/575
DIGITAL IMAGE PROCESSING
PROJECT 2

Instructor: Dr. Dennis Helder

Student: Harsh Dubey

Date: 09/30/2019

Worked with: Lin Zeng and Gagan

1. Histogram Specification

Write a MATLAB script that will perform histogram specification on cgray.img. The specified histogram is $p_z(z) = (\pi/2)\sin(\pi z)$ for $0 \le z \le 1$. Note that this is a normalized histogram (the pixel values are normalized from zero to one, and the area under the histogram is unity). Again, show before and after images and histograms, and the transformation function. Compare results to histogram equalization and normalization.

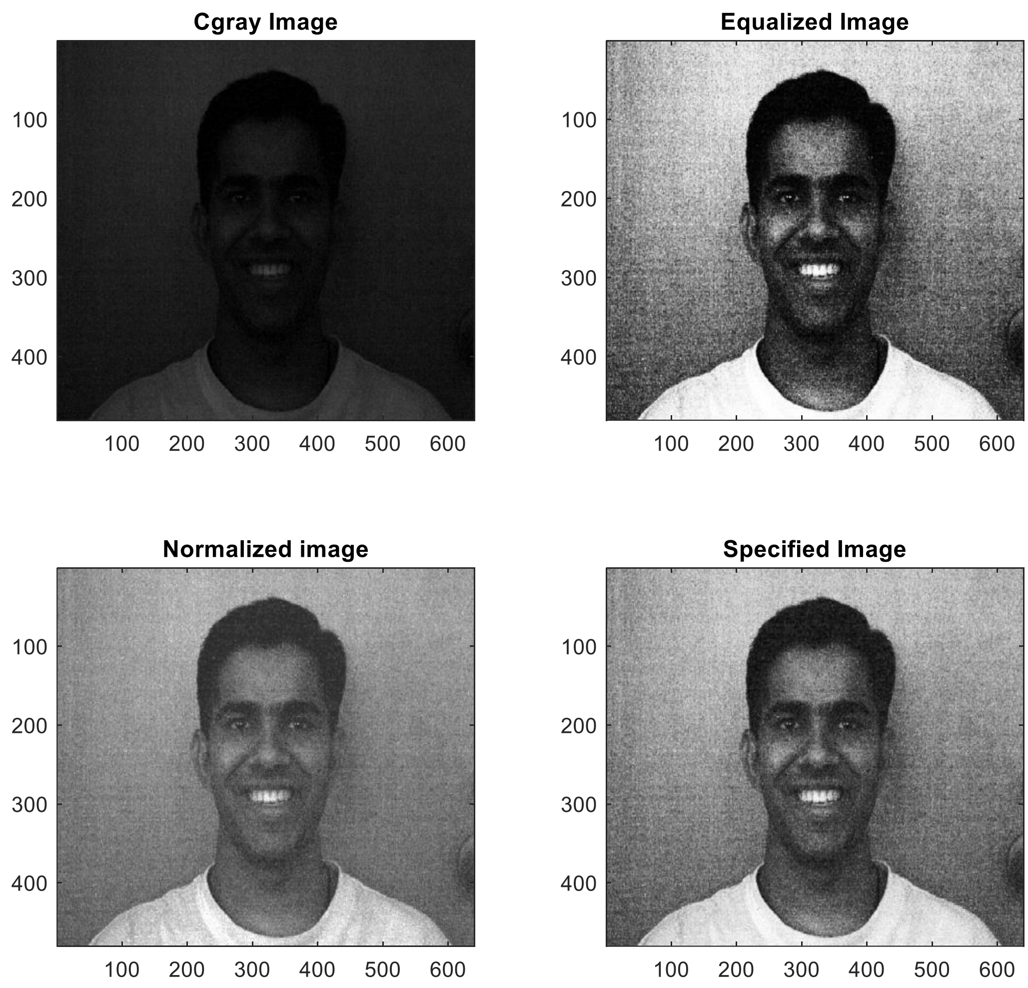

Figure 1: Comparison of Original, Equalized, Normalized, Specified Image

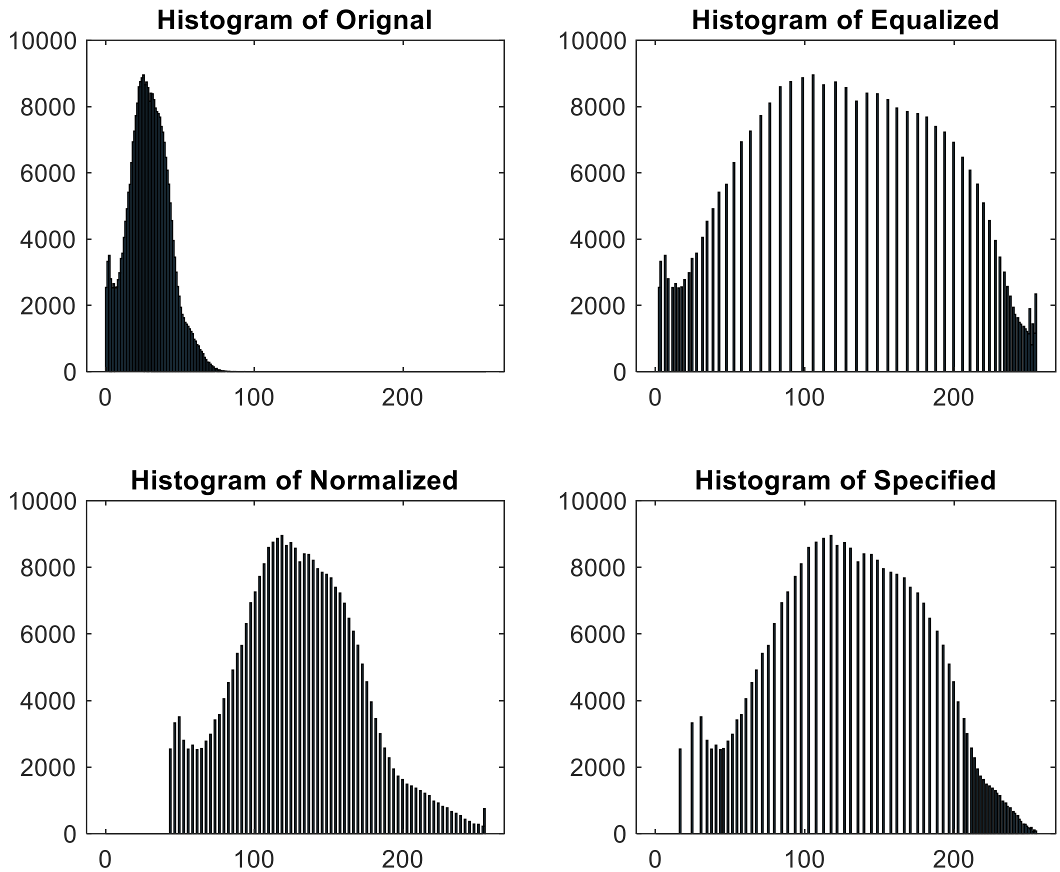

Figure 2: Comparison of Original, Equalized, Normalized, Specified Histogram

On comparing the images and their respective histograms, it can be noticed that the specified image enhances details better than any other image. The original image is dark, and its histograms also suggests the same. Even though the pixels in the normalized image are evenly distributed, however, there are none in the low intensity side, hence, resulting in poor contrast. Equalized image and specified image both have pixel even distributed across most intensity values. However, in case of equalized image the pixel density is much higher in low [0 to 30] and high [230 to 255] intensity values, hence, the contrast is too much in equalized image. In case of specified image, it is missing pixels in low [0 to 15] intensity range and have dense pixels in high [205 to 255] intensity range, this results in better contrast as compared to other images.

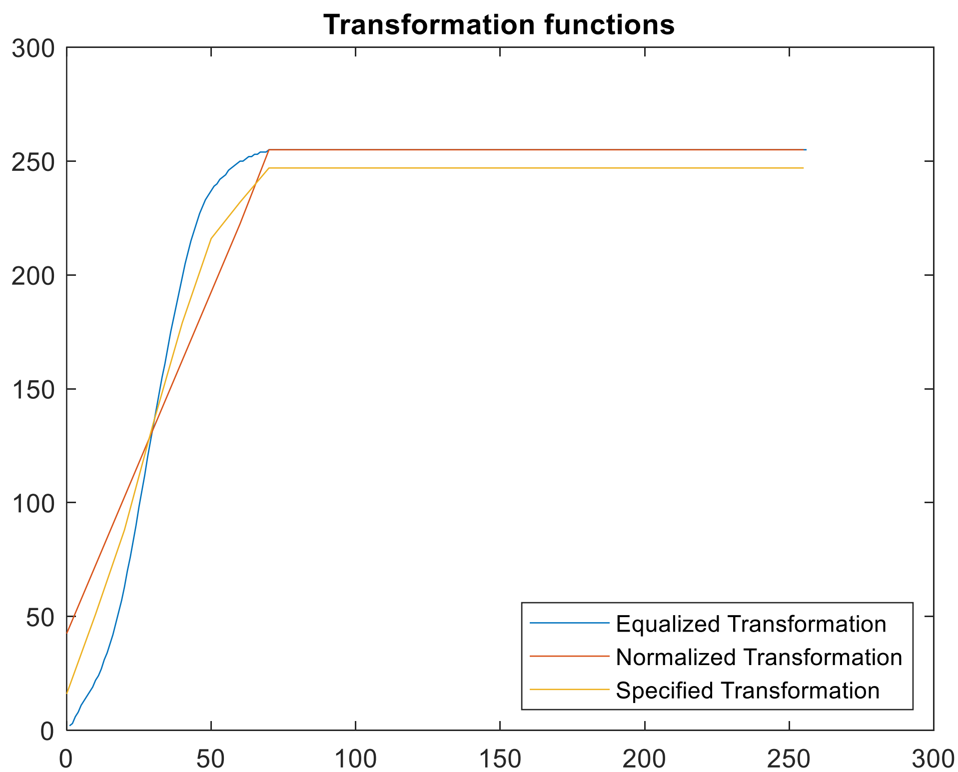

Figure 3: Transformation function comparison

Transformation function of equalized image has a curve at high intensity and low intensity values. That means it is increasing the high intensity value and decreasing the low intensity value and that gives a sharp contrast. Which is consistent with the image we received after performing histogram equalization, Fig.1.

Transformation function of normalized image starts at 42 and has a linear shape after that. Which is also consistent with the image we received after performing linearization.

Transformation function of specified image starts 16 and has linear shape. Which is consistent with our image.

Part2: Spatial Smoothing Filters
Apply a 3x3 and 11x11 neighborhood averaging filter to NoisyLena.img. Note there is random white noise as well as impulse noise present! In a bright, smooth region of the image, make an estimate of the noise standard deviation both before and after filtering. Are your results consistent with theory discussed in class?
Next, apply a 3x3 median filter to the image. Again, estimate noise standard deviation after filtering.

Procedure: A mask was created by hand drawing a random shape around 300x400 pixel location using drawfreehand function in MATLAB. Then the image was masked, and the hand drawn part was separated from the original image, Fig.4, also the background was set to NaN. Later the hand drawn part was filtered [entire image], and noise standard deviation was calculated.

Then noisy lena was filtered by using a 3by3 and 11by11 mean filter and 3by3 median filter, Fig.5, this was accomplished by using MATLAB conv2(a,b) for mean and MATLAB medfilt2(I) function for median.

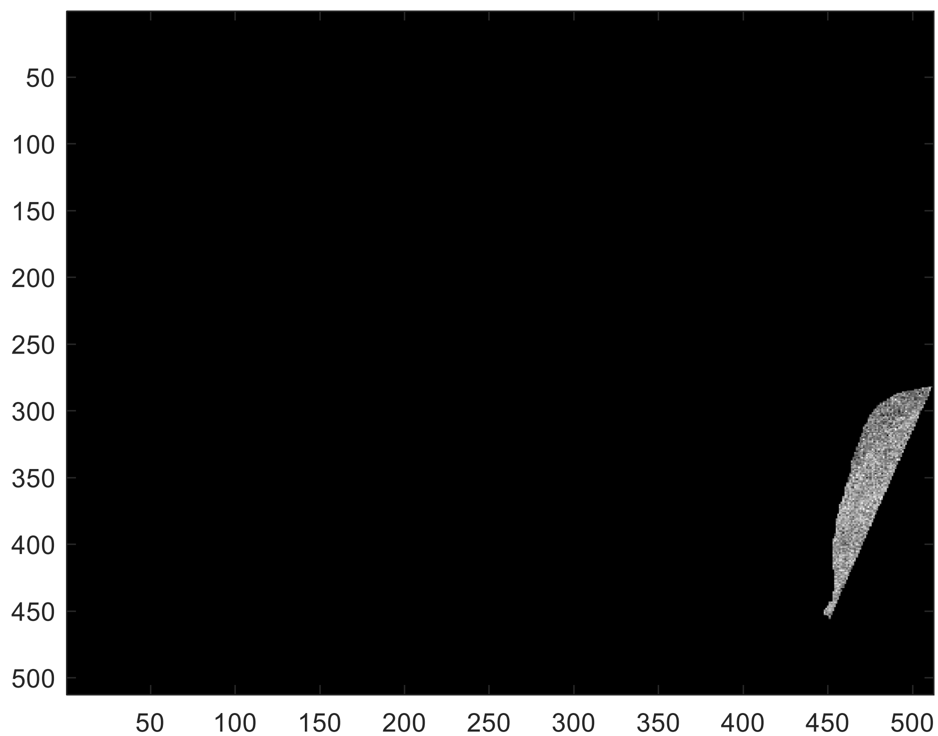

Figure 4: Bright region masked for std analysis

Table 1: Noise standard deviation

	Noise standard deviation
Before filtering	21.53
After 3by3 mean filter	10.26
After 11by11 mean filter	8.37
After 3by3 median filter	11.53

Form the table it is evident that noise has significantly decreased after filtering. The standard deviation of noise is less in 11by11 filtering as compared to 3by3 mean filtering. Which is consistent with what we studied in class. Since the filtering window was bigger, therefore more noise was filtered out and more image smoothing happened. This phenomenon can also be seen in the noisy Lena image, where 11by11 blurs image more as compared to 3by3 mean filter, Fig.5.

As seen in the table 3by3 median filter has the highest standard deviation, which is also evident in noisy Lena image which was filtered by 3by3 median filter. The 3by3 filtered Lena is grainier, however, the image is not blurred. Which is consistent to what we studied in class, since we studied that median filtered is good at preserving edges.

However, 3by3 mean filter gave the best result overall, which suggests the image had more white noise than impulse noise.

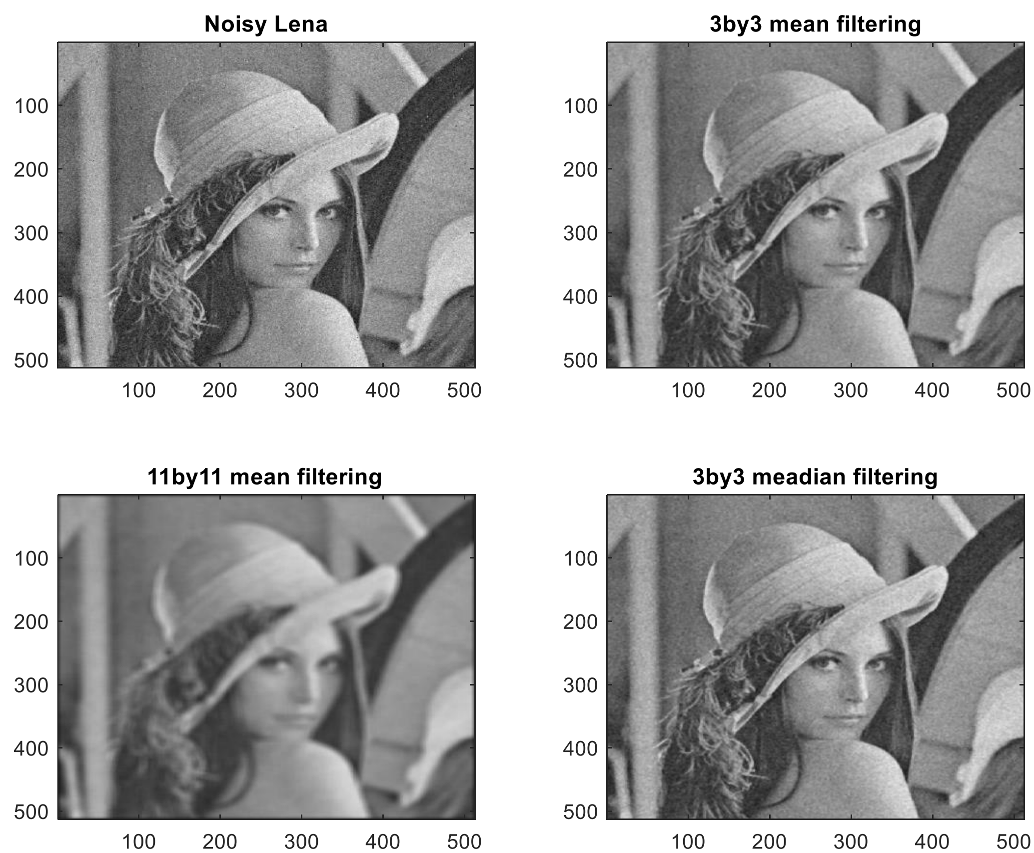

Figure 5: Noise Filtering of Lena

Note there is a bright pixel at [x,y] = [454,118] in the original image. Plot the horizontal line of data on either side of that pixel from 354 to 512 for the original, 3x3 smoothing filtered image, and 3x3 median filtered image on the same axis (different colors though and large enough to see the differences!). Explain the amount of noise reduction obtained by both methods.

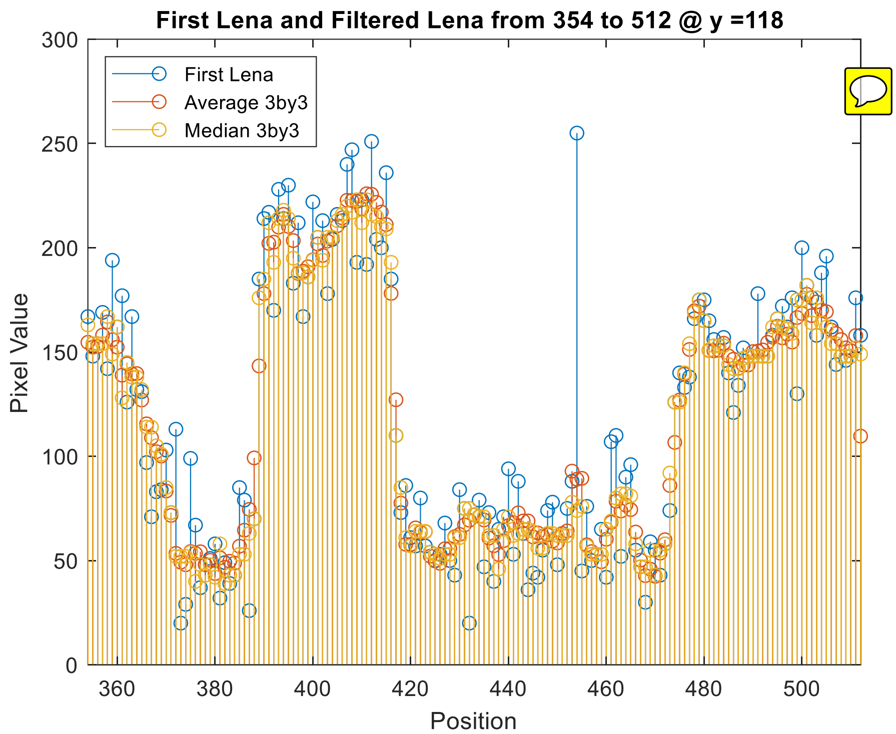

Figure 6: Comparison of Lena and Filtered Lena

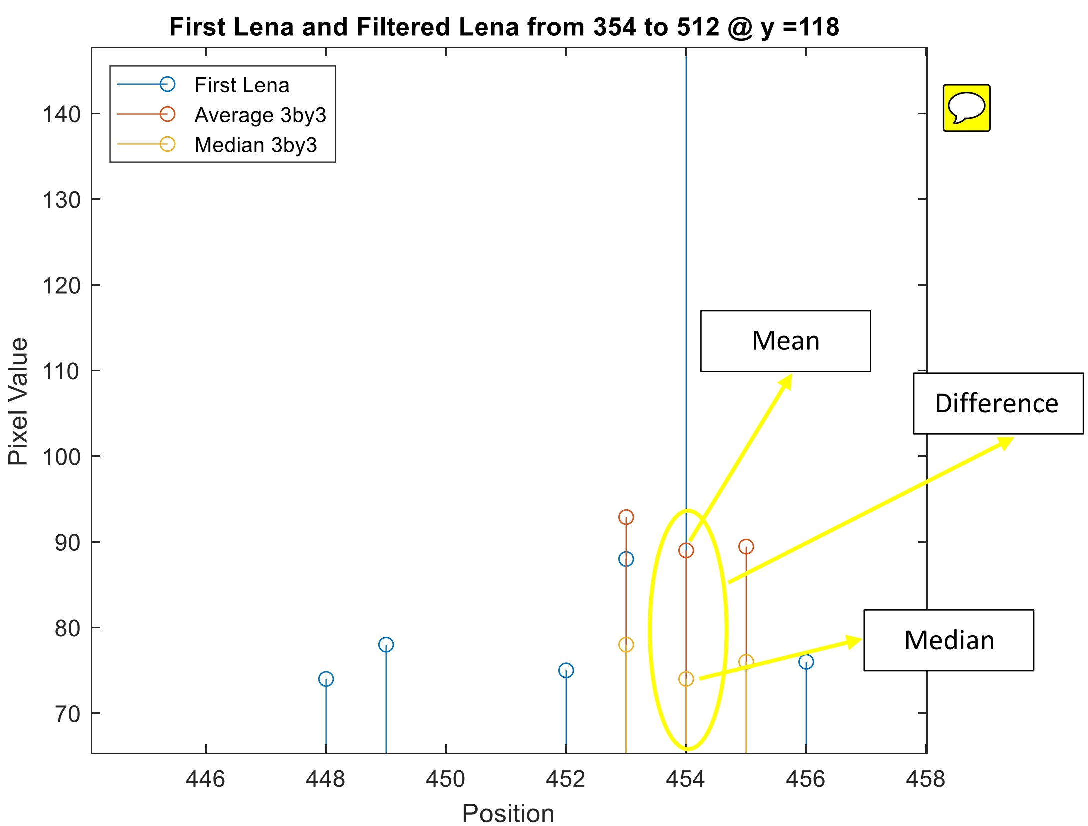

Figure 7: Difference highlighted

It is evident from the highlighted difference, fig.7, that the median filter has reduced more impulse noise as compared to the mean filter, which is consistent to what we studied in class.

Which method provided better removal of white random noise—3x3 averaging or 3x3 median filtering?

3X3 averaging filter provided better removal of white noise.

Which method provided better removal of impulse noise—3x3 averaging or 3x3 median filtering?

3X3 median filter provided better removal of impulse noise.

Be sure to explain your procedure to me and describe the results of each filter in detail. Lastly, since this image has two types of noise in it, what do you suggest would be an optimal approach for noise removal? Please describe your process and show the resulting image.

The best way

For the noise removal three different techniques were used, first was filtering the image using 3by3 mean and then filtering it with 3by3 median, second was filtering the image with 3by3 median and then 3by3 mean and third was filtering the image with 3by3 median filter and then filtering it with a gaussian filter with standard deviation of 0.5.

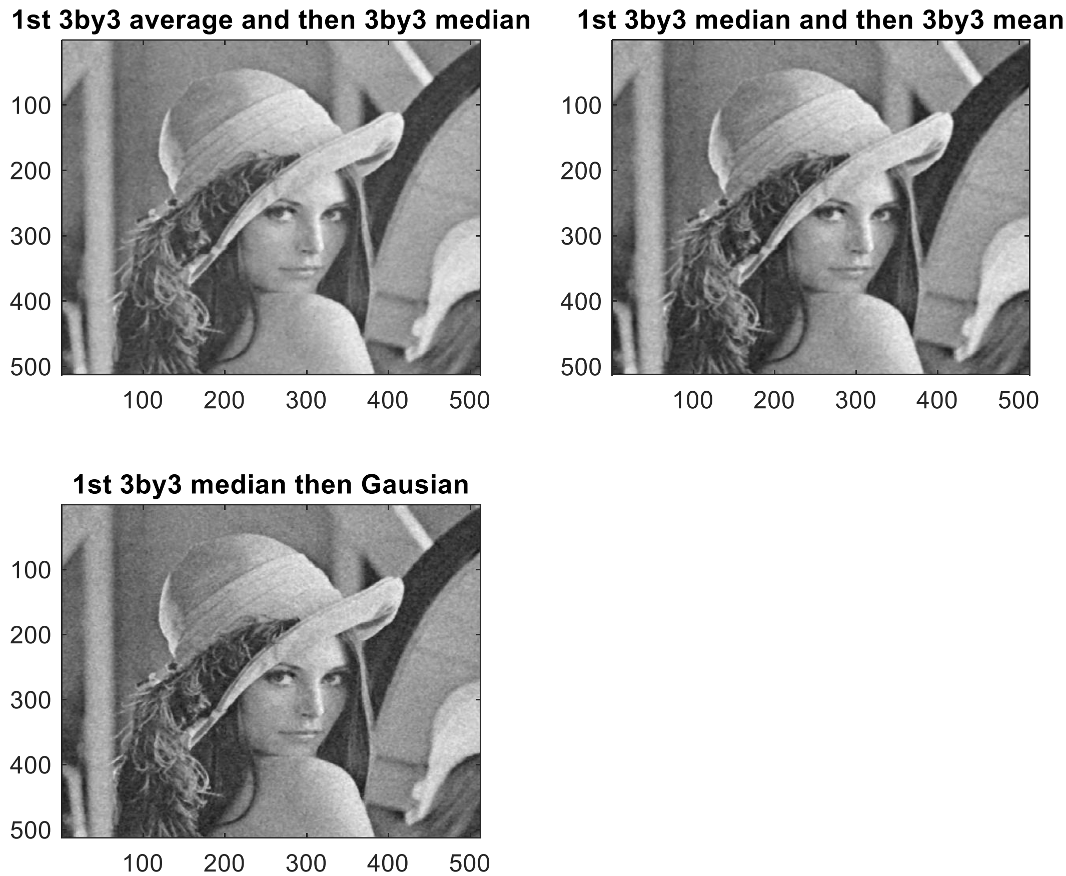

Figure 8: Result of three kinds of filtering

After analyzing the result, the first technique of filtering with 3by3 mean and then with 3by3 median produced the best result. The image looked less grainy and smooth.

However, for further analysis Lena’s eye (a complex human organ which is the result of years of “evolution”) was chosen. Image was zoomed in on the eye, fig.9, and it was noticed that median filter is better at preserving edges than gaussian and averaging filter. The gaussian filter had the worst performance when it came to preserving the edges but maintained better contrast than averaging filter.

Hence, the best technique would depend on the application.

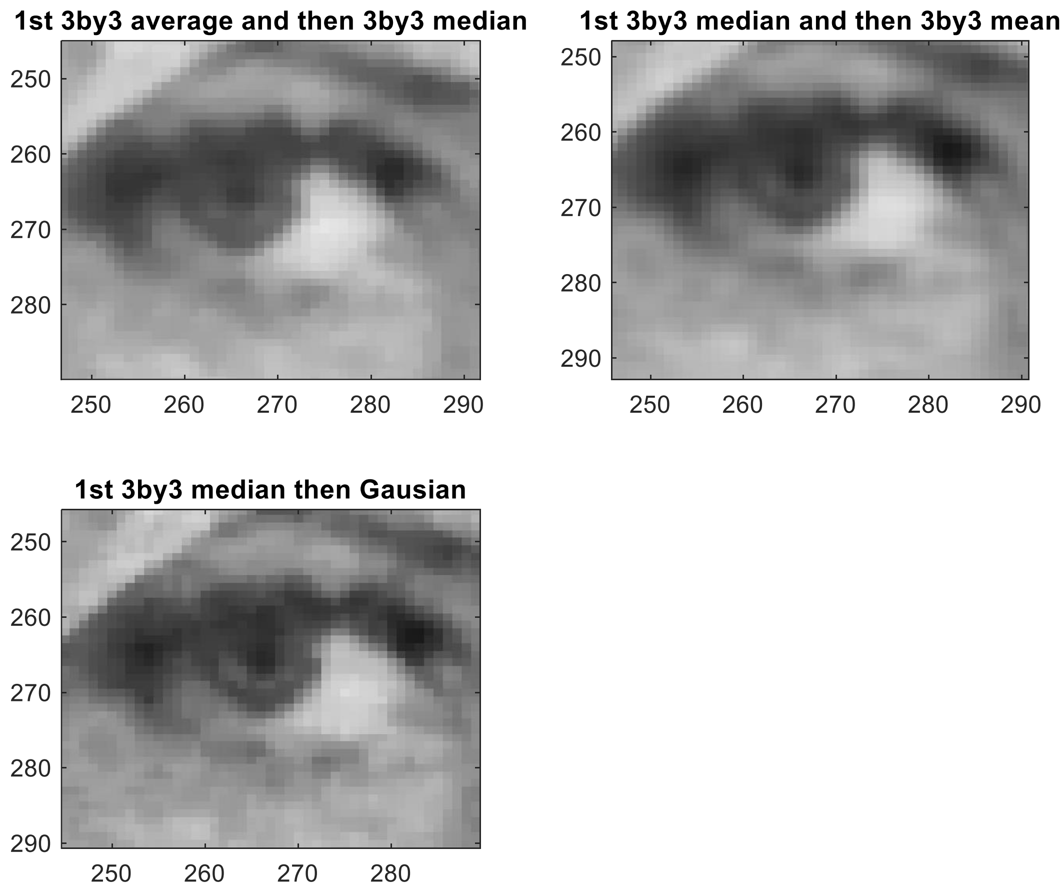

Figure 9:Comparision of Lena's eye

Appendix

```
%Author: Harsh Dubey
%Date: 09/29/19
%Citation:
http://www.osmanoglu.org/supplement/unpublishedWork/Digital
ImageProcessing_HistogramEqualizationMatching.pdf?fbclid=Iw
AR3hTthcuI5U4srTkwOGKcQp6HCpTCigNkFqHY-i_Opd3sEenk0zxCwxbhk
%Refrenced this lab and code for histogram matching part
%Worked Together: Me, Lin and Gangan. We shared ideas and
also discussed
%the matlab code. Since, matlab was challenging for
%this project

clear all
close all

%get Histogram
firstImage = loadraster('cgray.img',480,640);
colormap(gray(256));
figure;imshow(firstImage);title('Cgray Image');
figure, histogram(firstImage);title('Histogram of Cgray
Image');

%Normalized Image
ImageNormal = 3*firstImage+42.5;
figure;imshow(ImageNormal);colormap(gray(256));title('Nomal
ized Image');
figure;histogram(ImageNormal,0:255);title('Histogram of
Normal Image');

%Histgogram Equalization
[ImageEq,TransFunc] = histeq(firstImage,256);
figure;histogram(ImageEq);title('Histogram of equalized
Image');
colormap(gray(256));
figure;imshow(ImageEq);title('Equalized Image');


%CDF of first Image
colormap(gray(256));
Levels=256;
firstImage=double(firstImage);
[m,n]=size(firstImage);
```

```
length=m*n;
x=reshape(firstImage,length,1);
ImagePDF=hist(x,[0:Levels-1]);

ImagePDF=ImagePDF/length;
ImageCDF=cumsum(ImagePDF);%CDF of Image
figure; plot(ImageCDF);title('Transformation of First
Image');

%Histogram Specification
z = 0:1/255:1;
%figure; plot(z);
SinePDF = pi/2*sin(pi*z);
%figure; stem(SinePDF);
SinePDF=SinePDF/sum(SinePDF);
%figure; stem(SinePDF);
SineCDF=cumsum(SinePDF)/sum(SinePDF);%CDF of Sine
figure; plot(SineCDF);title('Sine CDF');

mapping=zeros(256);
%figure; plot(mapping);

z0=zeros(m,n);%256 zeros
%figure; plot(z0);
for q=1:Levels%going from 0 to 255
 for p=(mapping(q)+1):Levels%p = 0 + [1 to 255]
 if ((SineCDF(p)-ImageCDF(q)) >= 0)
 mapping(q) = p;
 list=find(firstImage == q-1);
 z0(list)=p;
 break;
 end
 end
end

z=reshape(z0,length,1);
figure;histMatched=histogram(z,0:255);
ShowHistM = hist(z,0:255)/307200;
title('Histogram of Speficied Image');
figure;image(z0);colormap(gray(256));title('Specified
Image');

figure;
```

```
subplot(511),stem([0:1/255:1],ImagePDF,'.'),title('Histogra
m of Orignal');
subplot(512),stem([0:1/255:1],SinePDF,'.'),title('Histogram
of Sine');
subplot(513),stem([0:1/255:1],ImageCDF,'.'),title('CDF of
the Image');
subplot(514),stem([0:1/255:1],SineCDF,'.'),title('CDF of
sine function');
subplot(514),stem([0:1/255:1],ShowHistM,'.'),title('Specifi
ed Histogram');

array_mapping =
[0,10,20,30,40,50,60,70,94,100,150,200,255];
array_specification =
[16,51,88,135,179,216,232,247,247,247,247,247,247];
array_normal =
[42.5,72.5,102.5,132.5,162.5,192.5,222.5,255,255,255,255,25
5,255];
figure;plot(255*TransFunc);
hold on;
plot(array_mapping,array_normal);
hold on;
plot(array_mapping,array_specification);
hold off;
legend1 = sprintf('Equalized Transformation');
legend2 = sprintf('Normalized Transformation');
legend3 = sprintf('Specified Transformation');
legend({legend1,legend2,legend3},'Location','southeast');
title('Transformation functions');

figure;
subplot(221),histogram(firstImage,0:255),title('Histogram
of Orignal');
subplot(222),histogram(ImageEq,0:255),title('Histogram of
Equalized');
subplot(223),histogram(ImageNormal,0:255),title('Histogram
of Normalized');
subplot(224),histogram(z,0:255),title('Histogram of
Specified');

figure;
subplot(221),image(firstImage),title('Cgray
Image');colormap(gray(256));
```

```
subplot(222),image(ImageEq),title('Equalized
Image');colormap(gray(256));
subplot(223),image(ImageNormal),title('Normalized
image');colormap(gray(256));
subplot(224),image(z0),title('Specified
Image');colormap(gray(256));

%Part2: Noisy lena and filtering
Lena=loadraster('NoisyLena.img',512,512);%load Lena
figure;imagesc(Lena);colormap(gray(256));title('First
Lena');%show Lena
cut=drawfreehand;copy=createMask(cut);%Draw lena

Avg3by3=conv2(Lena,ones(3)/9,'same');colormap(gray(256));fi
gure;image(Avg3by3);
title('Smoothing Filter of 3by3');

Avg11by11=conv2(Lena,ones(11)/121,'same');colormap(gray(256
));figure;image(Avg11by11);
title('Smoothing Filter of 11by11');

Meadian3by3=medfilt2(Lena);colormap(gray(256));figure;image
(Meadian3by3);colormap(gray(256))
title('Meadin filter of 3by3');

figure;
subplot(221),image(Lena),title('Noisy
Lena');colormap(gray(256));
subplot(222),image(Avg3by3),title('3by3 mean
filtering');colormap(gray(256));
subplot(223),image(Avg11by11),title('11by11 mean
filtering');colormap(gray(256));
subplot(224),image(Meadian3by3),title('3by3 meadian
filtering');colormap(gray(256));

deviation_b4Filter=double(Lena).*copy;
deviation_b4Filter(deviation_b4Filter==0)=nan;
figure;imagesc(deviation_b4Filter);colormap(gray(256));
deviation_b4Filter=std(deviation_b4Filter,0,'all','omitnan'
);

deviation_aftr3by3ave=double(Avg3by3).*copy;
deviation_aftr3by3ave(deviation_aftr3by3ave==0)=nan;
figure;imagesc(deviation_aftr3by3ave);colormap(gray(256));
```

```
after3by3=std(deviation_aftr3by3ave,0,'all','omitnan');

deviation_aftr11by11=double(Avg11by11).*copy;
deviation_aftr11by11(deviation_aftr11by11==0)=nan;
figure;imagesc(deviation_aftr11by11);colormap(gray(256));
after11=std(deviation_aftr11by11,0,'all','omitnan');

deviation_aftr3by3med=double(Meadian3by3).*copy;
deviation_aftr3by3med(deviation_aftr3by3med==0)=nan;
figure;imagesc(deviation_aftr3by3med);colormap(gray(256));
after3by3med=std(deviation_aftr3by3med,0,'all','omitnan');

figure;
stem(Lena(118,:));
xlim([354 512]);
xlabel('Pixel-Position');
ylabel('Pixel-Value');
title('Pixel 354-512 @y = 118 at first Lena');
hold on;
stem(Avg3by3(118,:));
xlim([354 512]);
xlabel('Pixel-Position');
ylabel('Pixel-Value');
title('Pixel 354 to 512 @ y=118 @ average 3by3');
hold on;
stem(Meadian3by3(118,:));
xlim([354 512]);
xlabel('Position');
ylabel('Pixel Value');
title('Pixel 354 to 512 @ y=118 @ median 3by3');
hold off

legend1 = sprintf('First Lena');
legend2 = sprintf('Average 3by3');
legend3 = sprintf('Median 3by3');
legend({legend1,legend2,legend3},'Location','northwest');
title('First Lena and Filtered Lena from 354 to 512 @ y
=118 ');

MeanFilteredLena=medfilt2(Avg3by3);
figure;
subplot(221);
imagesc(MeanFilteredLena);colormap(gray(256));
title('1st 3by3 average and then 3by3 median');
```

```
MedianFilteredLena=conv2(Meadian3by3,ones(3)/9,'same');
subplot(222);
imagesc(MedianFilteredLena);colormap(gray(256));
title('1st 3by3 median and then 3by3 mean');

B = imgaussfilt(Meadian3by3,0.5);
subplot(223);imagesc(B);colormap(gray(256));
title('1st 3by3 median then Gausian');
```