BBL588 E

ADVANCED TOPICS IN COMPUTER VISION

HOMEWORK 1

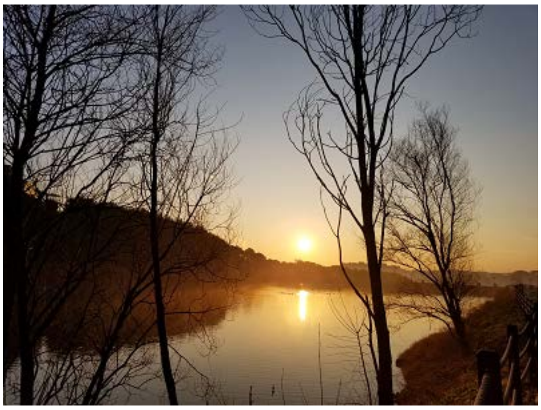
Associate Professor Uğur Behçet TÖREYİN

Seher MAVİ

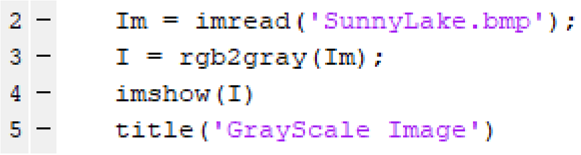
518181050

Project GitHub Link: https://github.com/SeherMavi/BBL588E-HomeWork1.git

1.

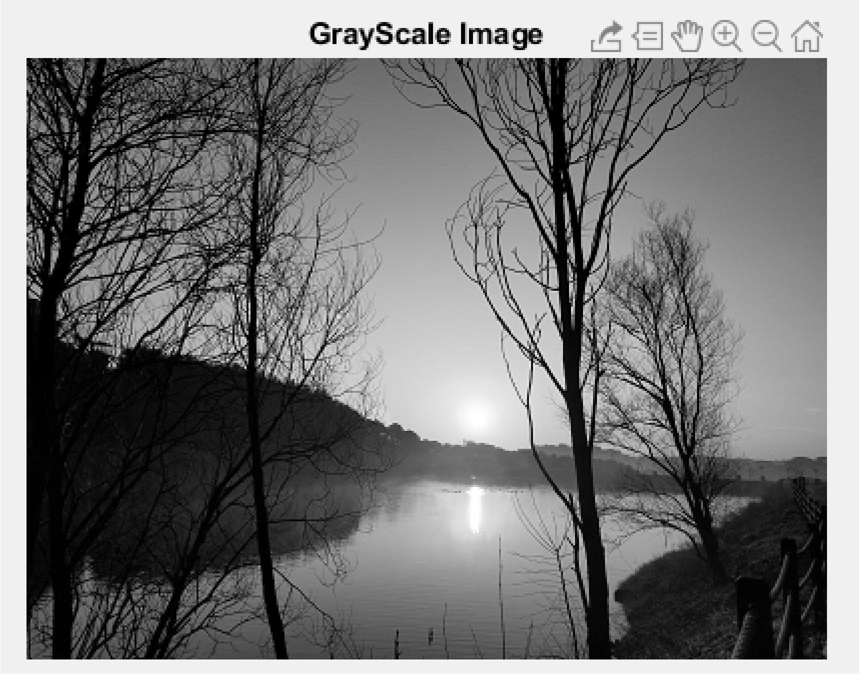


2.

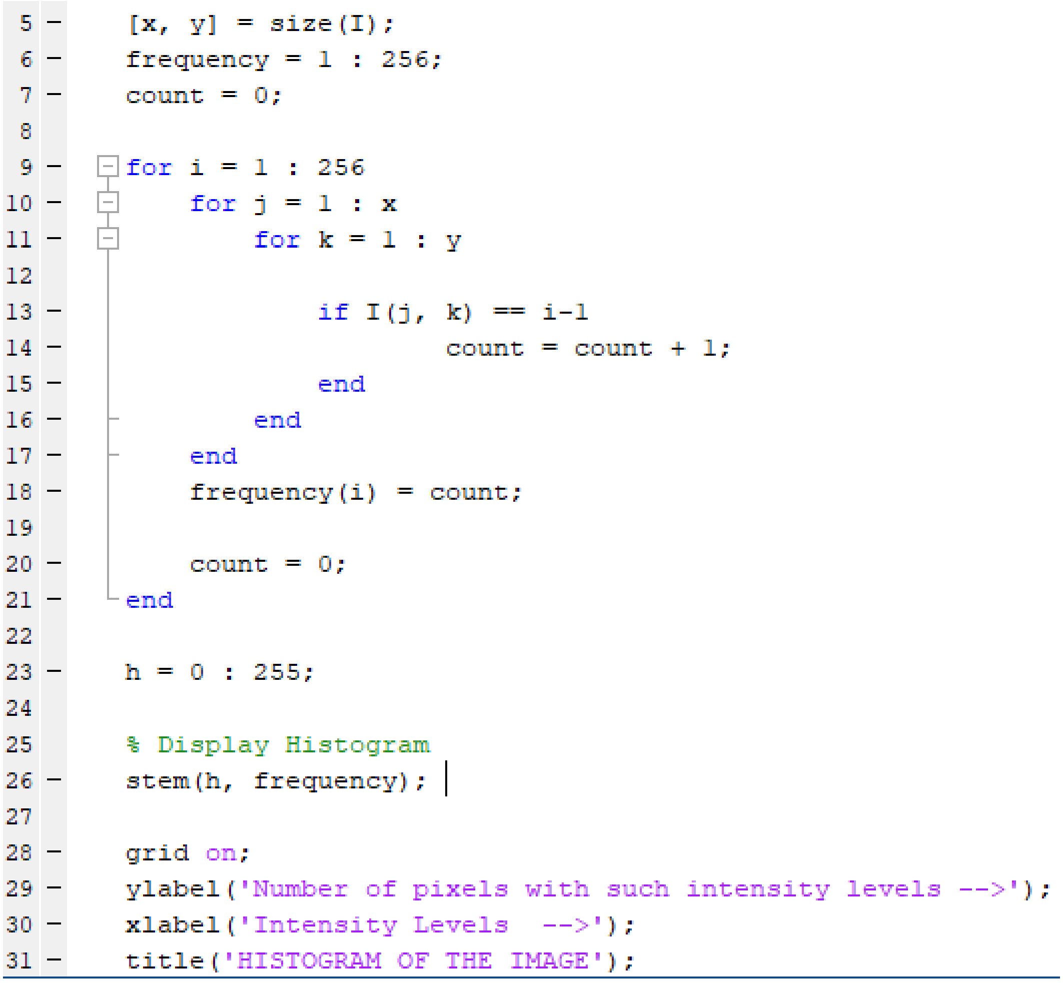


```
Im = imread('SunnyLake.bmp');
I = rgb2gray(Im);
imshow(I)
title('GrayScale Image')
```

rgb2gray converts RGB values to grayscale values by forming a weighted sum of the R, G and B components.

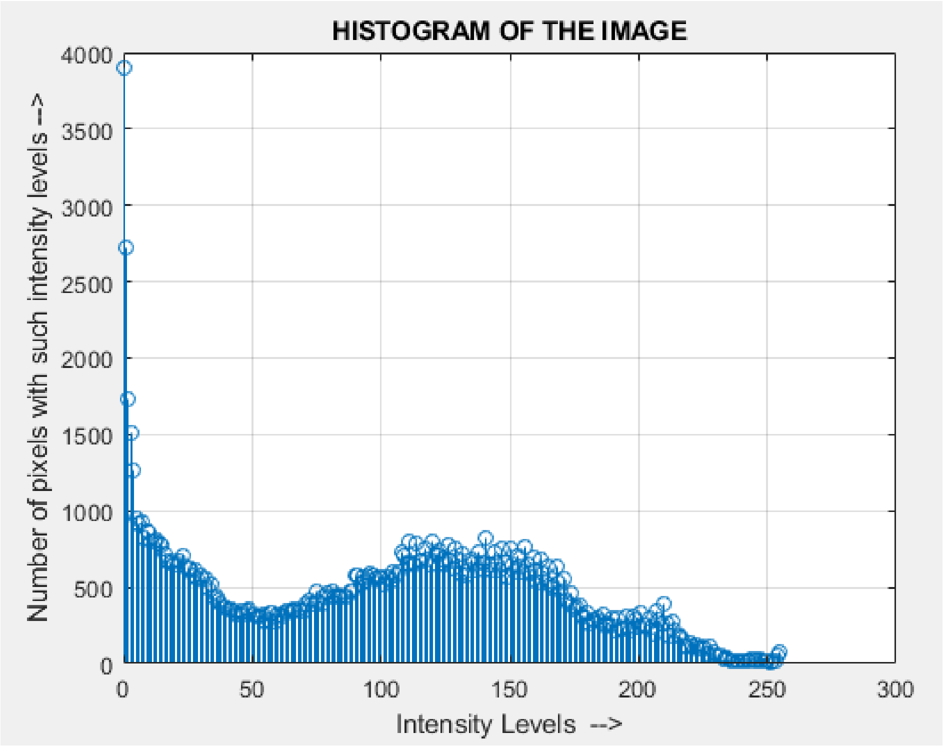


3.

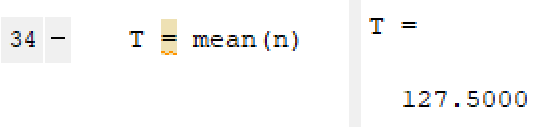


```
[x, y] = size(I);
frequency = 1 : 256;
count = 0;

for i = 1 : 256
    for j = 1 : x
        for k = 1 : y

            if I(j, k) == i-1
                    count = count + 1;
            end
        end
    end
    frequency(i) = count;

    count = 0;
end

h = 0 : 255;

% Display Histogram
stem(h, frequency);

grid on;
ylabel('Number of pixels with such intensity levels -->');
xlabel('Intensity Levels  -->');
title('HISTOGRAM OF THE IMAGE');
```

Get the dimension of the image, create a frequency array of size 256, iterate over grayscale image matrix , for every possible intensity value and count them, update its position of frequency array with count and display the histogram;



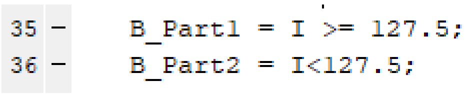
4.



```
T = mean(n)
```

```
T =

   127.5000
```

T is average number of histogram values. I choose it as a threshold.



```
B_Part1 = I >= 127.5;
B_Part2 = I<127.5;
```

5.

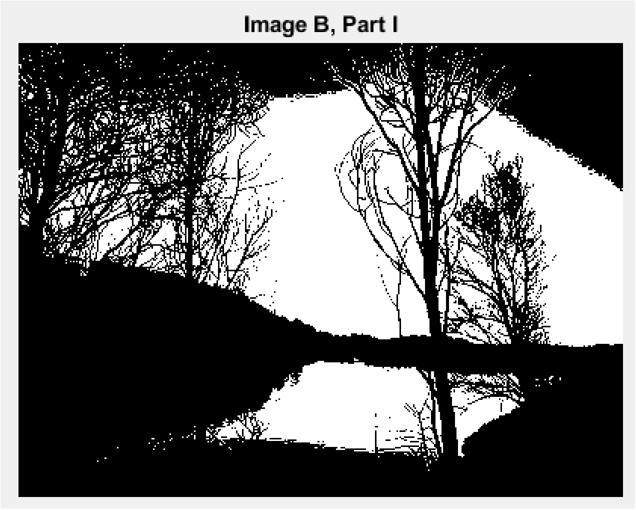


Image B, Part I

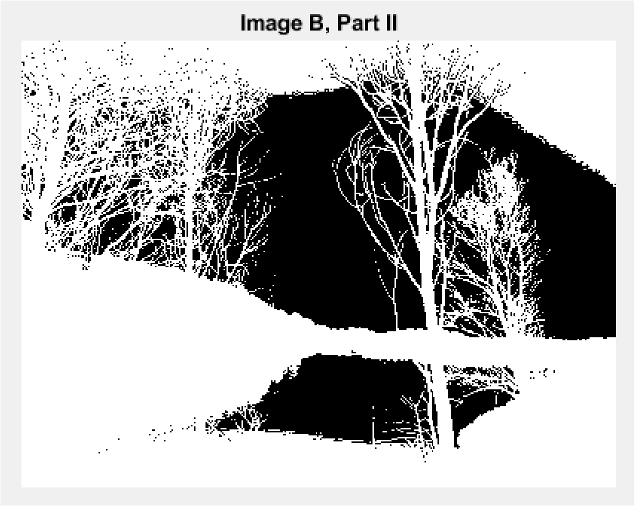
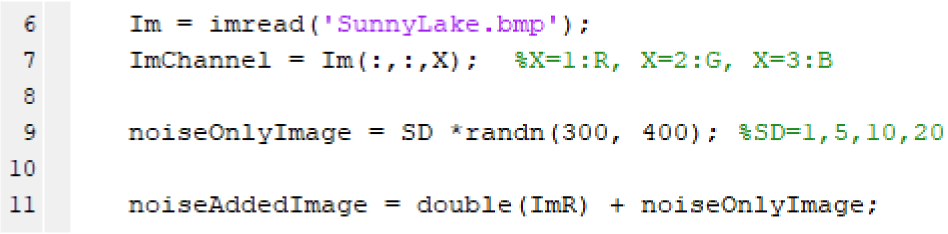
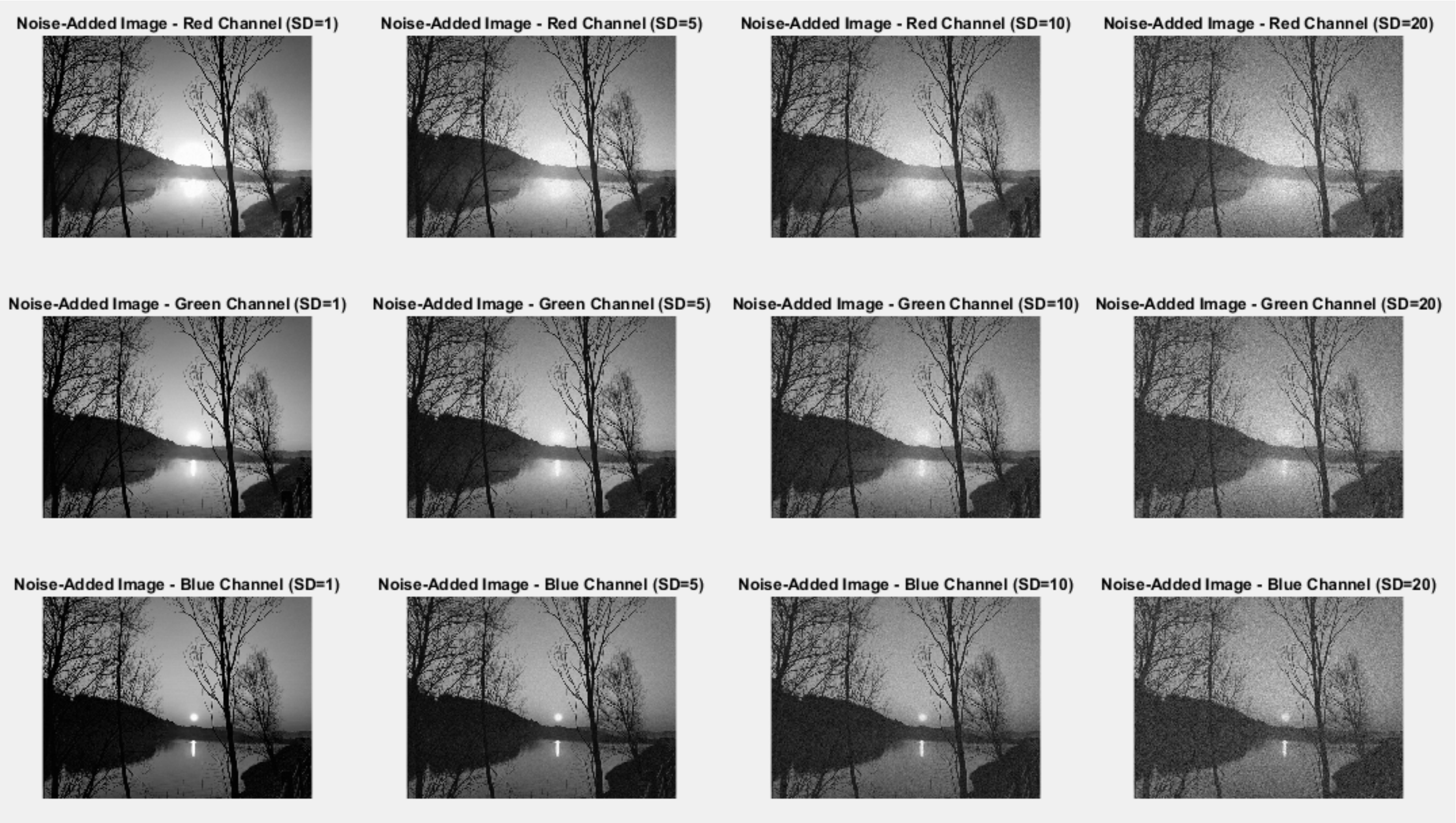


Image B, Part II

6.



```
Im = imread('SunnyLake.bmp');
ImChannel = Im(:,:,X);  %X=1:R, X=2:G, X=3:B

noiseOnlyImage = SD *randn(300, 400); %SD=1,5,10,20

noiseAddedImage = double(ImR) + noiseOnlyImage;
```

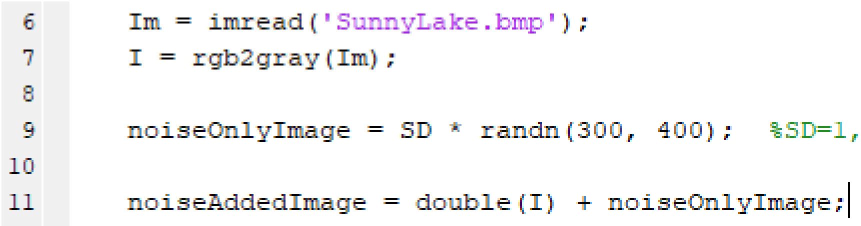


Noise-Added Image - Red Channel (SD=1)
Noise-Added Image - Red Channel (SD=5)
Noise-Added Image - Red Channel (SD=10)
Noise-Added Image - Red Channel (SD=20)

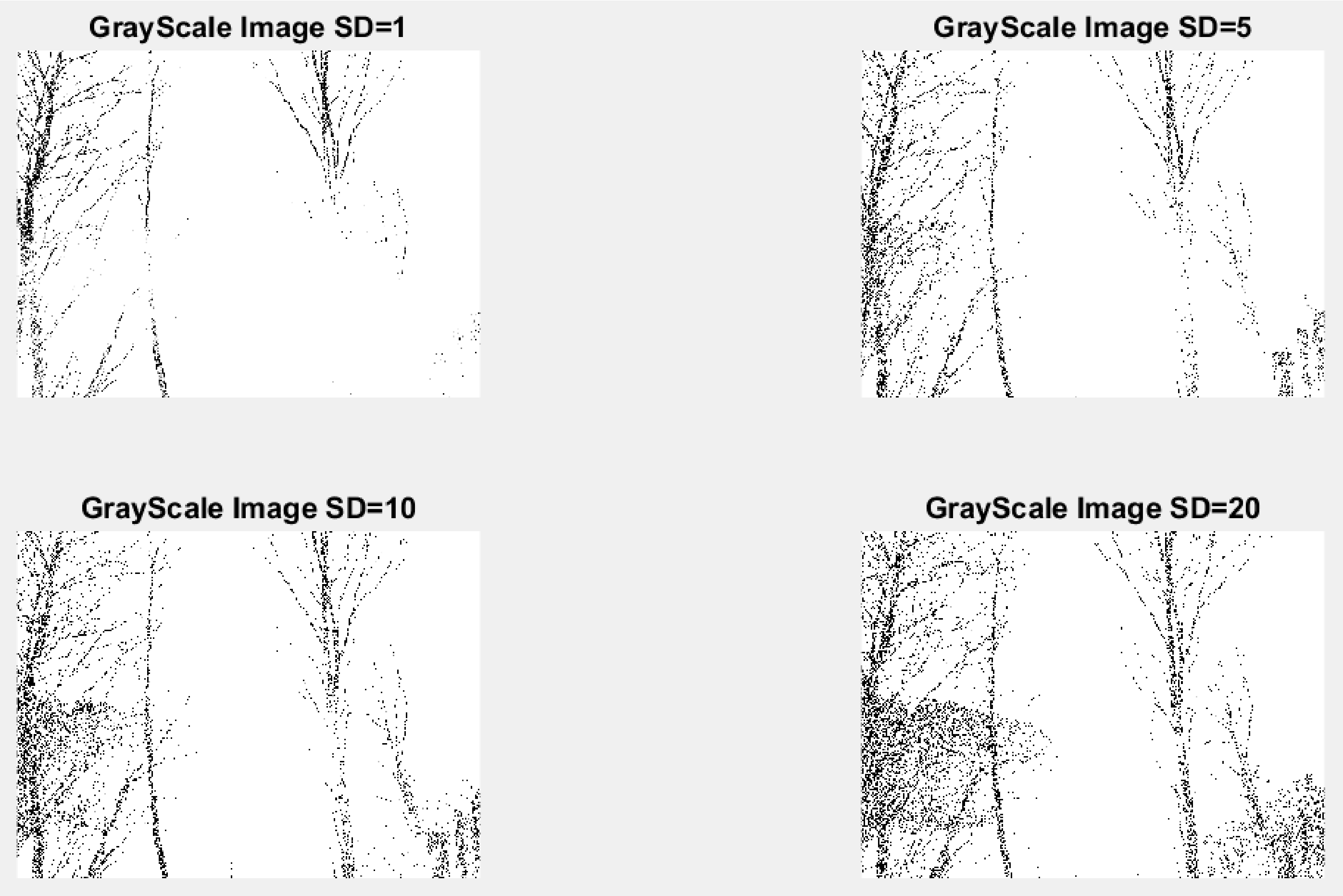
Noise-Added Image - Green Channel (SD=1)
Noise-Added Image - Green Channel (SD=5)
Noise-Added Image - Green Channel (SD=10)
Noise-Added Image - Green Channel (SD=20)

Noise-Added Image - Blue Channel (SD=1)
Noise-Added Image - Blue Channel (SD=5)
Noise-Added Image - Blue Channel (SD=10)
Noise-Added Image - Blue Channel (SD=20)

7.



```
Im = imread('SunnyLake.bmp');
I = rgb2gray(Im);

noiseOnlyImage = SD * randn(300, 400);   %SD=1,5,10,20

noiseAddedImage = double(I) + noiseOnlyImage;
```



GrayScale Image SD=1

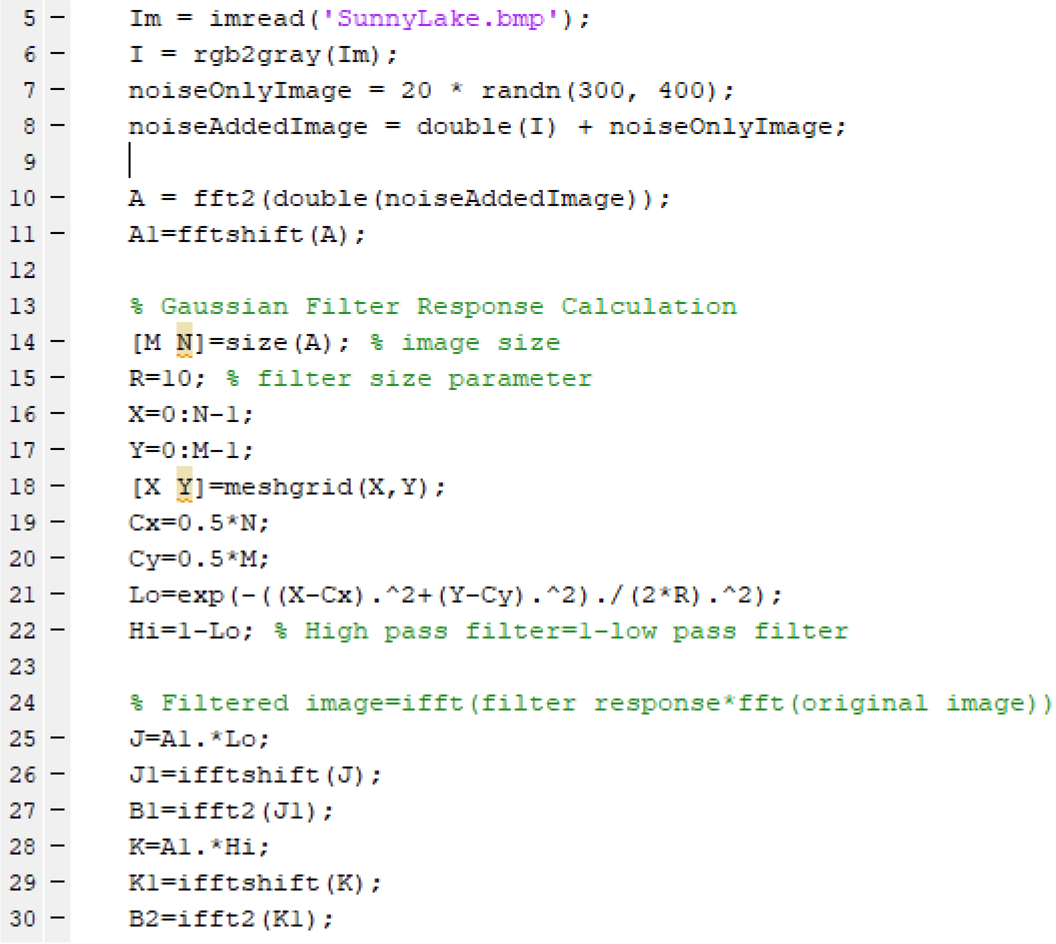
GrayScale Image SD=5

GrayScale Image SD=10

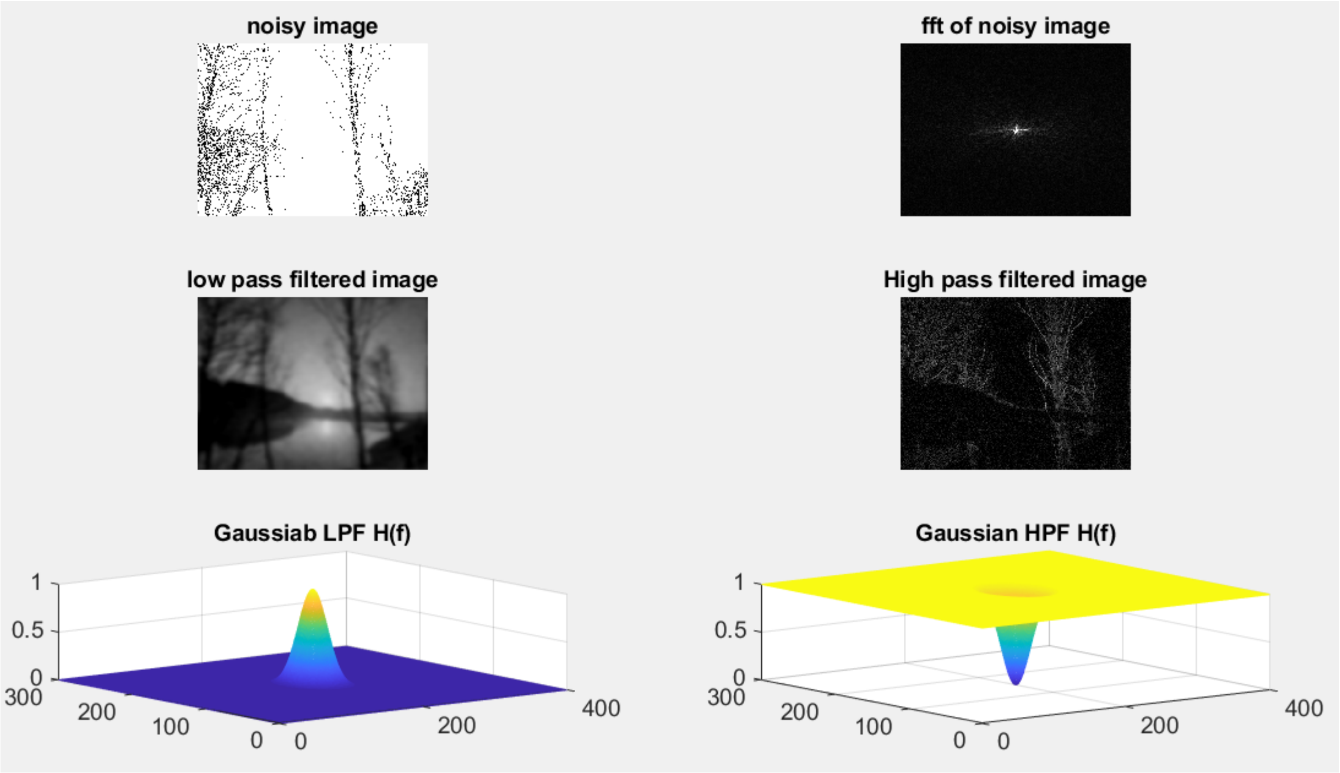
GrayScale Image SD=20

8. Combined with 9.

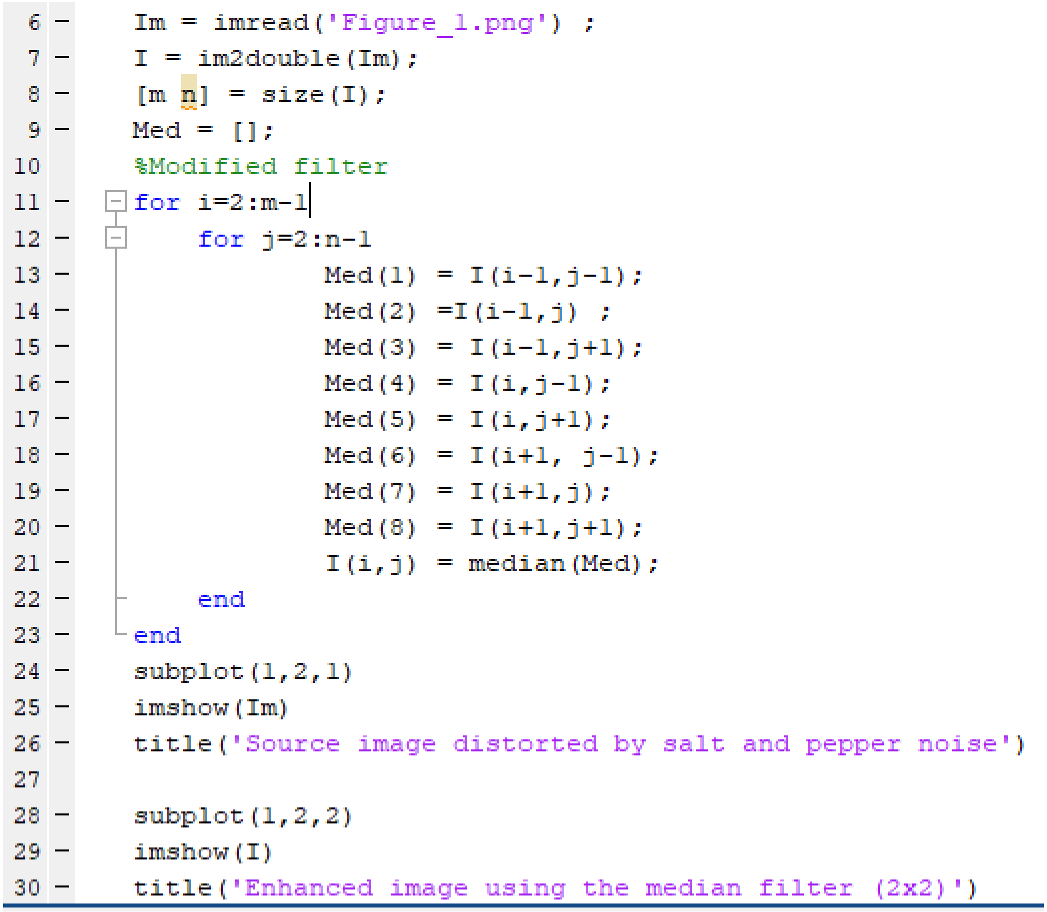
9.



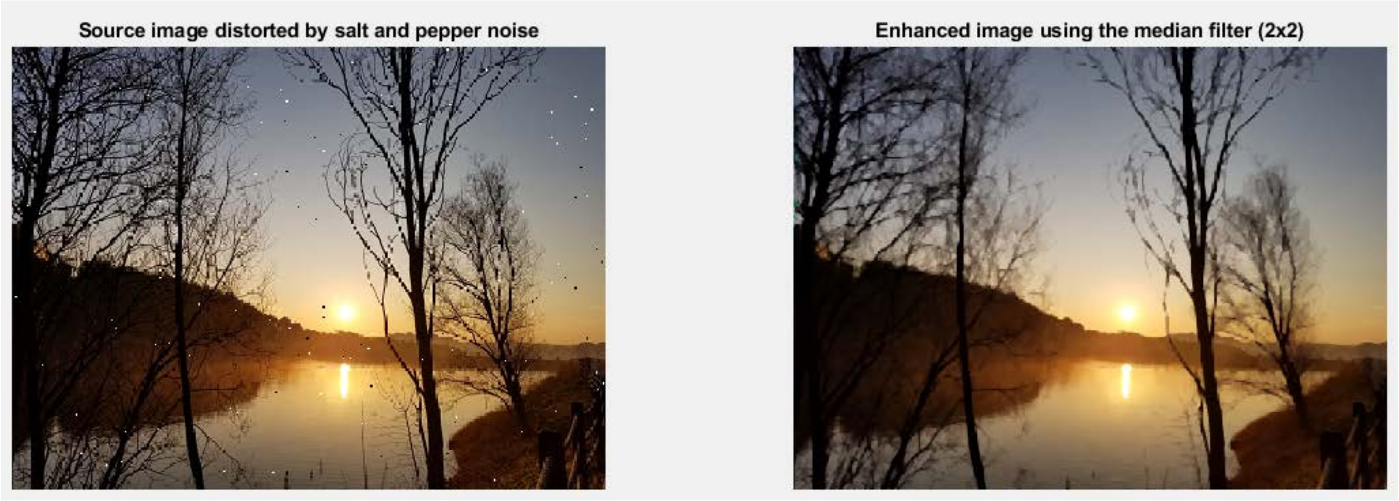
```
Im = imread('SunnyLake.bmp');
I = rgb2gray(Im);
noiseOnlyImage = 20 * randn(300, 400);
noiseAddedImage = double(I) + noiseOnlyImage;

A = fft2(double(noiseAddedImage));
A1=fftshift(A);

% Gaussian Filter Response Calculation
[M N]=size(A); % image size
R=10; % filter size parameter
X=0:N-1;
Y=0:M-1;
[X Y]=meshgrid(X,Y);
Cx=0.5*N;
Cy=0.5*M;
Lo=exp(-((X-Cx).^2+(Y-Cy).^2)./(2*R).^2);
Hi=1-Lo; % High pass filter=1-low pass filter

% Filtered image=ifft(filter response*fft(original image))
J=A1.*Lo;
J1=ifftshift(J);
B1=ifft2(J1);
K=A1.*Hi;
K1=ifftshift(K);
B2=ifft2(K1);
```



10. Type of noise is ‘Salt and Paper’. The problem can be fixed by applying a median filter.



```
Im = imread('Figure_1.png') ;
I = im2double(Im);
[m n] = size(I);
Med = [];
%Modified filter
for i=2:m-1
    for j=2:n-1
            Med(1) = I(i-1,j-1);
            Med(2) =I(i-1,j) ;
            Med(3) = I(i-1,j+1);
            Med(4) = I(i,j-1);
            Med(5) = I(i,j+1);
            Med(6) = I(i+1, j-1);
            Med(7) = I(i+1,j);
            Med(8) = I(i+1,j+1);
            I(i,j) = median(Med);
    end
end
subplot(1,2,1)
imshow(Im)
title('Source image distorted by salt and pepper noise')

subplot(1,2,2)
imshow(I)
title('Enhanced image using the median filter (2x2)')
```


Source image distorted by salt and pepper noise
Enhanced image using the median filter (2x2)