LAB ASSIGNMENT RECORD

ON

ECS756

Digital Image Processing using SciLab

COLLEGE OF COMPUTING SCIENCES AND INFORMATION TECHNOLOGY

TMU, MORADABAD

Session: 2022 – 23 (Odd Semester)

Submitted To:

Dr. Saurabh Pathak

Submitted By:

Saiyam Jain

VII Sem / IV Year

B. Tech CSE (AI+ML+DL)

TCA1959024

Lab Assignment - 1

S. No	Program Name	Page No.	Date	Sign.	Remarks
1.	WAP to demonstrate some basic commands and functions of SciLab.				
2.	WAP in SciLab to import any image and display it.				
3.	WAP to convert color image (RGB) into Grayscale.				
4.	WAP to print Negative Image from Color Image using SciLab.				
5.	WAP to display Histogram and Histogram Equalization on any image.				
6.	WAP to Affine Transformation – To learn basic image transformation: (a) Translation (b) Rotation (c) Scaling				

Lab Assignment - 2

S. No	Program Name	Page No.	Date	Sign.	Remarks
1.	WAP to understand how frequency distribution can be used to represent an image.				
2.	WAP to enhance image using with median filtering.				
3.	WAP to implement Low Pass Filter				
4.	WAP to implement High Pass Filter				
5.	WAP to study the effect of the size of neighborhood on the result of processing.				

Lab Assignment - 1

1. WAP to demonstrate some basic commands and functions of SciLab.

```
⇨ a=21; b=10;                //assigning values to variables

  --> a==b                   //checking equality
   ans = F

  --> a>=b                   //checking greater or equal
   ans = T

  --> a+b                    //addition
   ans = 70.

  --> a-b                    //subtraction
   ans = 30.

  --> a*b                    //multiplication
   ans = 1000.

  --> a/b                    //division
   ans = 2.5


  --> A = [1,2,3;4,4,4]      //matrix declaration
   A = 1.   2.   3.
       4.   4.   4.

  --> A'                     //transpose of a matrix
   ans = 1.   4.
         2.   4.
         3.   4.

  --> max(A)                 //maximum of matrix
   ans = 4.

  --> min(A)                 //minimum of matrix
   ans = 1.

  --> length(A)              //length of matrix
   ans = 6.

  --> sum(A)                 //gives the sum of all elements of the matrix
   ans = 18.
```

2. WAP in SciLab to import any image and display it.

⇨ // import an image and display it

```
--> image = imread('brain.jpg');
--> imshow(image);
```

⇨ **Output:**

----

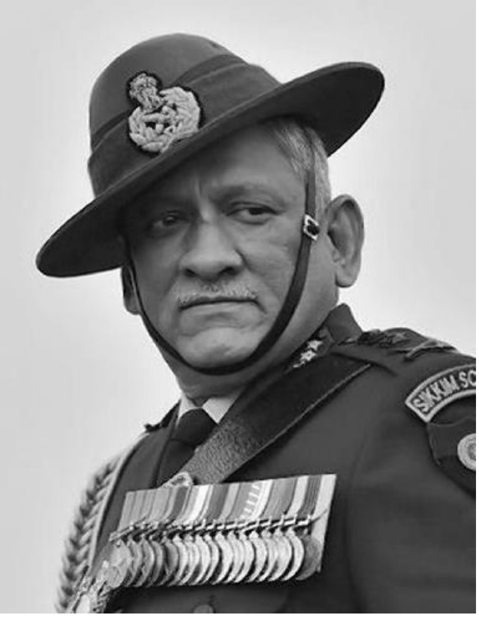
3. WAP to convert color image (RGB) into Grayscale.

⇨ --> //convert color image(RGB) to grayscale

```
--> colorImage = imread('general.jpg');
--> gray = rgb2gray(colorImage);
--> imshow(gray);
```

⇨ **Output:**

color image(RGB)



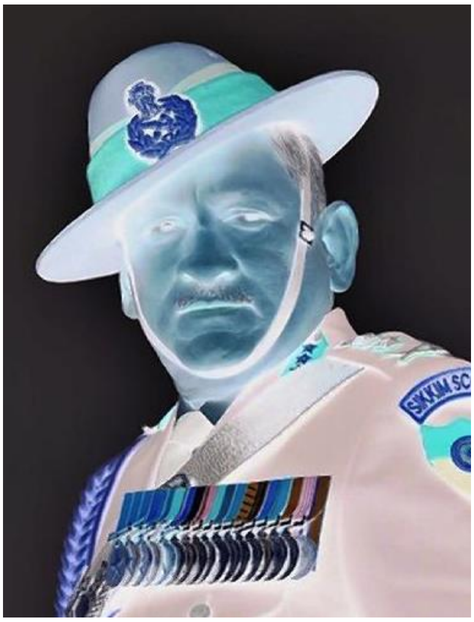
grayscale image

4. WAP to print Negative Image from Color Image using SciLab.

⇨
```
--> //convert color image (RGB) to its negative

--> colorImage = imread('general.jpg');
--> negative = imcomplement(colorImage);
--> imshow(negative);
```

⇨ **Output:**



Negative Image

- Negative Image:

A negative picture is the inverse of a normal or positive image. In a negative image, areas that are white or light appear dark and darker ones appear to be light. Usually negative photos are turned into positive images in a darkroom.

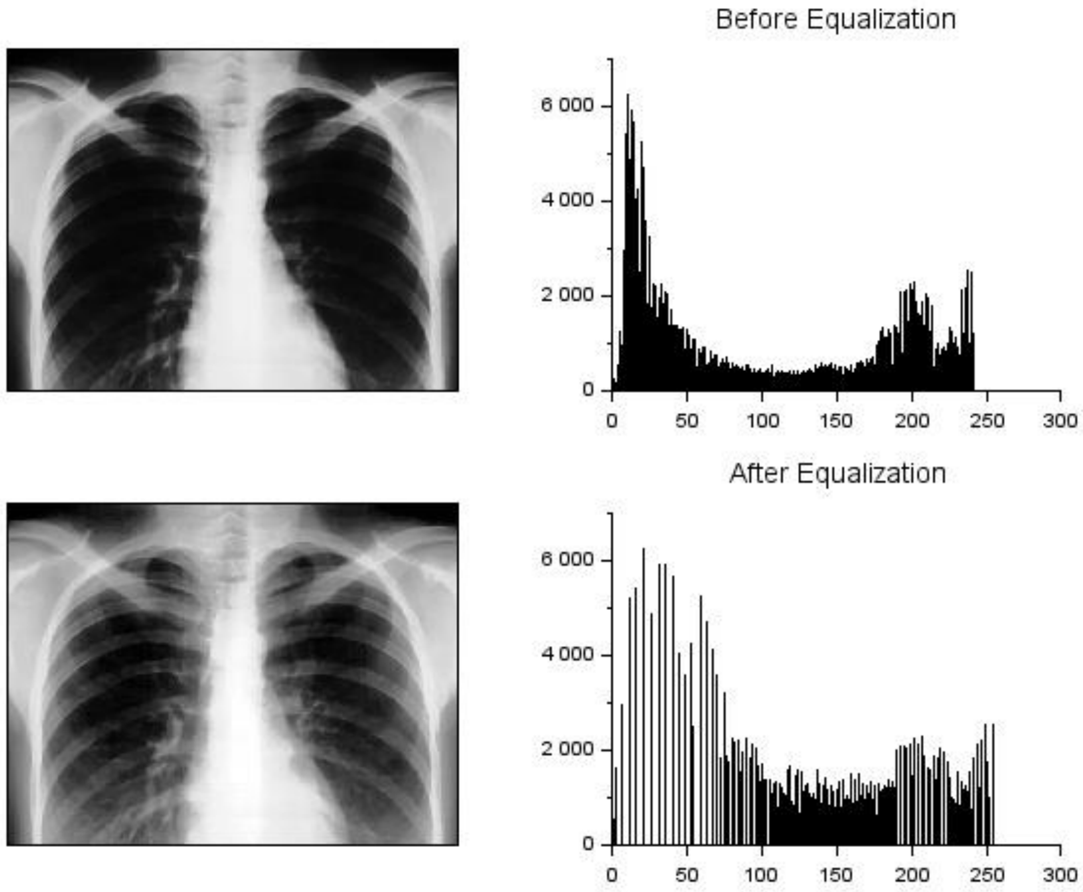
5. WAP to display Histogram and Histogram Equalization on any image.

⇨

```
--> //histogram and histogram equalization of any image

--> image = imread('img.jpg');
--> gray = rgb2gray(image);
--> [y,x] = imhist(gray);
--> grayeq = imhistequal(gray);
--> [y1,x1] = imhist(grayeq);
--> subplot(2,2,1); imshow(gray); subplot(2,2,2); plot2d3(x,y);
    subplot(2,2,3); imshow(grayeq); subplot(2,2,4); plot2d3(x1,y1);
```

⇨ **Output:**



- <u>Histogram Equalization:</u>

Histogram Equalization is a computer image processing technique used to improve contrast in the images. It accmpishes this by affectively spreading out the most frequent intensity values, i.e. streching out the intensity range of the image.

6. WAP to Affine Transformation - learn basic image transformation: Translation, Rotation & Scaling

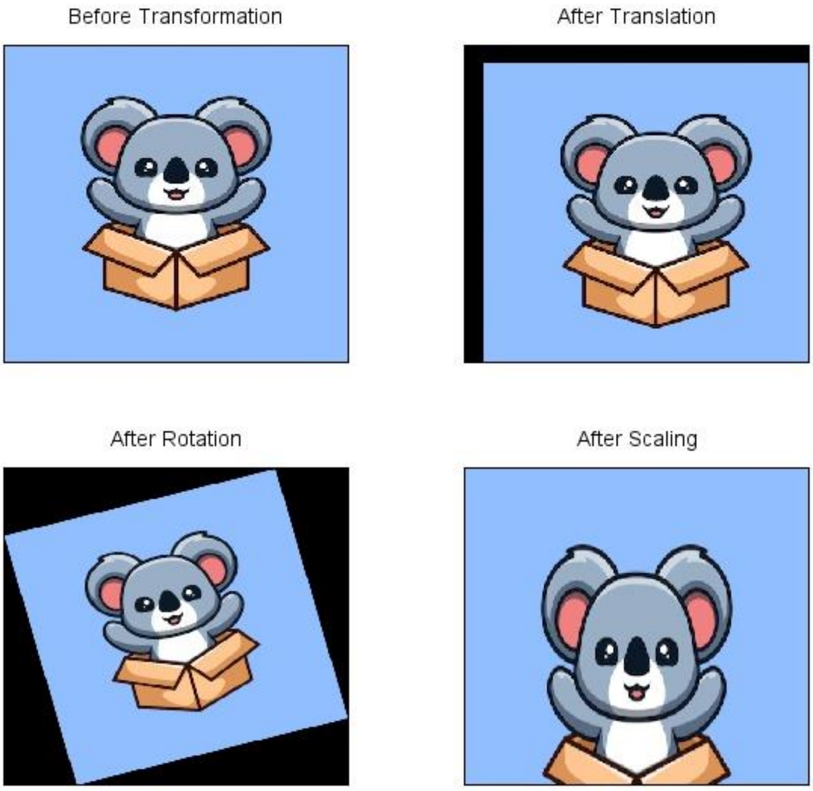
⇨
```
--> image = imread('cartoon.jpg');

--> // translation - displacement along x & y directions
--> x=20; y=20;
--> mat = [1, 0, 0;
-->         0, 1, 0;
-->         x, y, 1];
--> newImg = imtransform(image,mat,'affine');
--> subplot(2,2,1); imshow(image); title('Before Transformation');
--> subplot(2,2,2); imshow(newImg); title('After Translation');

--> // rotation - rotate the image by some angle q
--> q=15;
--> newImg = imrotate(image,q);
--> subplot(2,2,3); imshow(newImg); title('After Rotation');

--> // scaling - to change the scale of the image
--> x=1; y=1.5;
--> mat = [x, 0, 0;
-->        0, y, 0;
-->        0, 0, 1];
--> newImg = imtransform(image,mat,'affine');
--> subplot(2,2,4); imshow(newImg); title('After Scaling');
```

⇨ **Output:**



Lab Assignment - 2

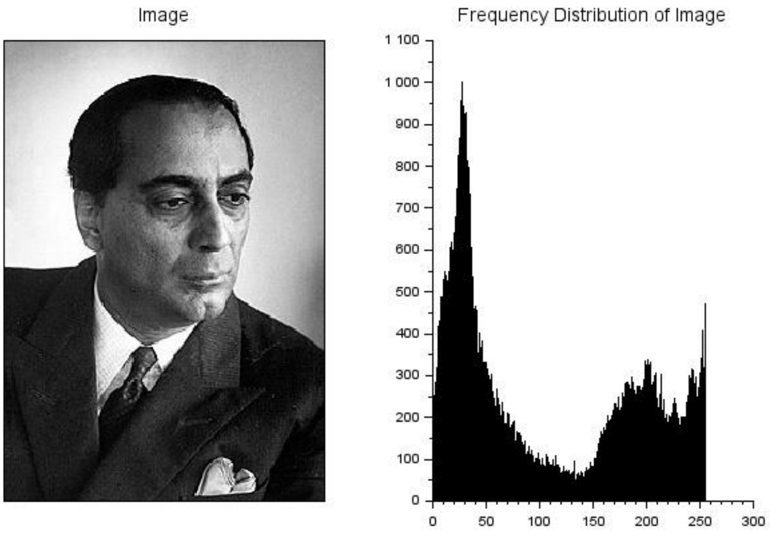
1. WAP to understand how frequency distribution can be used to represent an image.

⇨
```
// frequency distribution of an image

--> image = imread('bhabha.jpg');

--> subplot(1,2,1); imshow(image); title('Image');
--> subplot(1,2,2); imhist(image,[],); title('Frequency Distribution of Image');
```

⇨ **Output:**



The frequency distribution of a digital image is a distribution of its discrete intensity levels in the range [0, L-1]. The distribution is a discrete function h associating to each intensity level: r_k the number of pixels with this intensity: n_k.

2. WAP to enhance image using with median filtering.

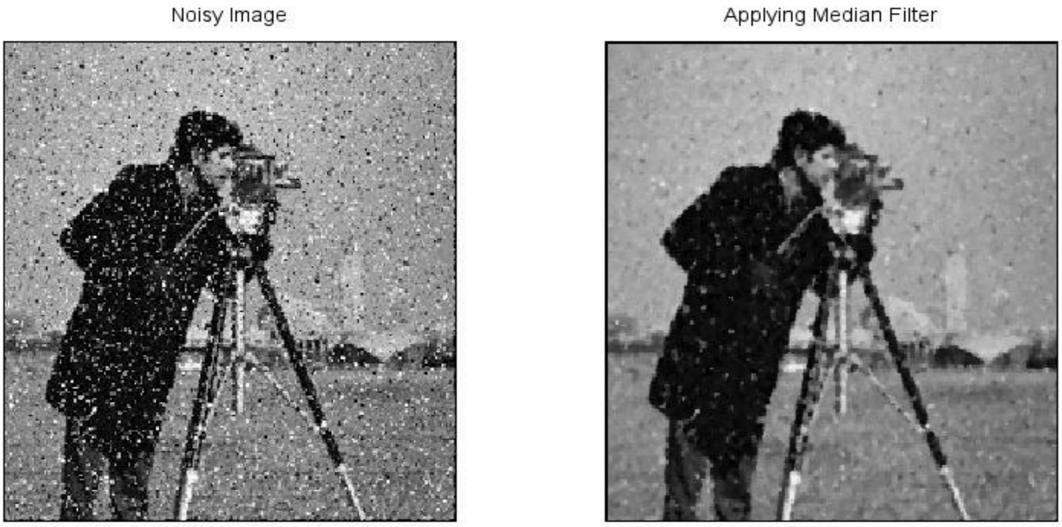
⇨ // removal of noise using median filter - applying 3*3 filter

```
--> a = imread('man.jpg');
--> [m,n]=size(a);

--> for i=2:m-1
-->     for j=2:n-1
-->         p(i,j) = median([ a(i-1,j-1), a(i,j-1), a(i+1,j-1), a(i-1,j), a(i,j), a(i+1,j), a(i-1,j+1), a(i,j+1),
a(i+1,j+1)]);
-->     end
--> end

--> subplot(1,2,1); imshow(a); title('Noisy Image');
--> subplot(1,2,2); imshow(p); title('Applying Median Filter');
```

⇨ **Output:**



<u>Median Filter:</u>

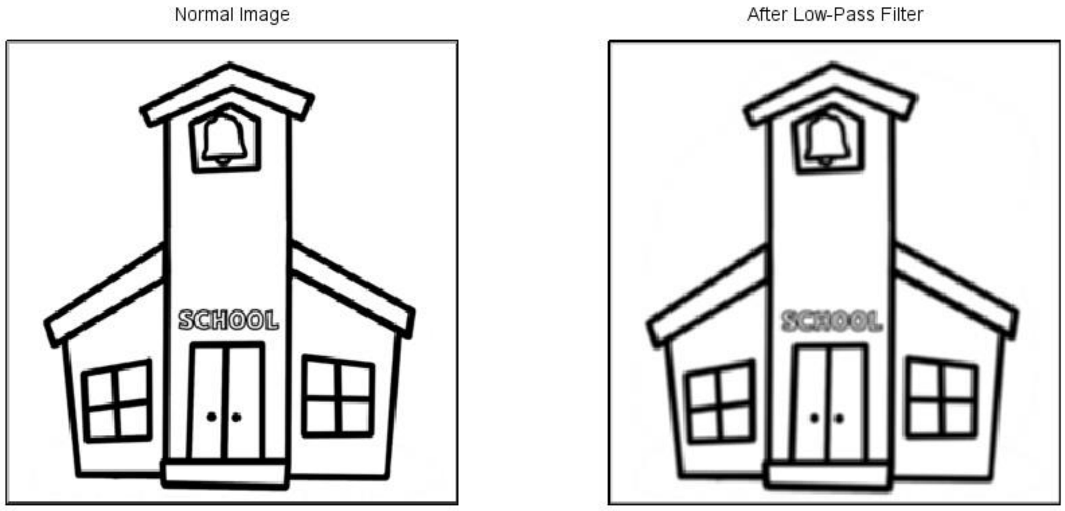
The median filter is the filtering technique used for noise removal from the images and signals. Median filter is very crucial in the image processing field as it is well known for preserving the edges during noise removal. It is also known as non-linear filtering. It is used to eliminate the salt and pepper noise. Here the pixel value is replaced by the median value of the neighboring pixel.

3. WAP to implement Low Pass Filter.

⇨ // low-pass filter

```
--> image = imread('school.jpg');
--> a = double(image);
--> [m,n]=size(a);
--> w = [1,1,1;1,1,1;1,1,1];
--> for i=2:m-1
-->    for j=2:n-1
-->        L(i,j)=[w(1) * a(i-1,j+1) + w(2) * a(i,j+1) + w(3) * a(i+1,j+1) +
                  w(4) * a(i-1,j) + w(5) * a(i,j) + w(6) * a(i+1,j) +
                  w(7) * a(i-1,j-1) + w(8) * a(i,j-1) + w(9) * a(i+1,j-1)] / 9;
-->    end
--> end
--> L = uint8(L);
--> subplot(1,2,1); imshow(image); title('Normal Image');
--> subplot(1,2,2); imshow(L); title('After Low-Pass Filter');
```

⇨ **Output:**



<u>**Low Pass Filter:**</u>

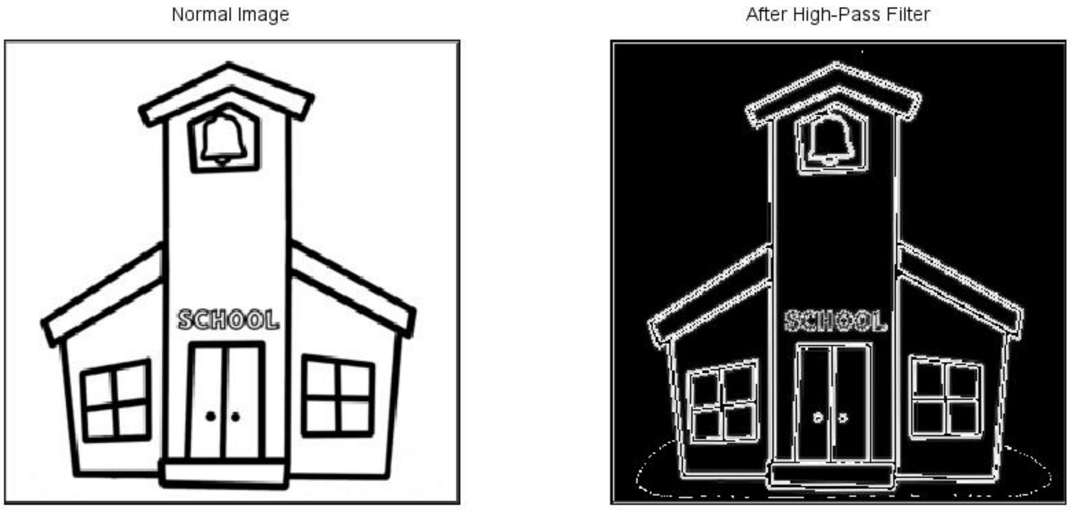
Low pass filter is the type of frequency domain filter that is used for smoothing the image. It attenuates the high frequency components and preserve the low frequency components.

4. WAP to implement High Pass Filter.

⇨ // high-pass filter

```
--> image = imread('school.jpg');
--> a = double(image);
--> [m,n]=size(a);
--> w = [-1,-1,-1;-1,8,-1;-1,-1,-1];
--> for i=2:m-1
-->    for j=2:n-1
-->        H(i,j)=[w(1) * a(i-1,j+1) + w(2) * a(i,j+1) + w(3) * a(i+1,j+1) +
                   w(4) * a(i-1,j) + w(5) * a(i,j) + w(6) * a(i+1,j) +
                   w(7) * a(i-1,j-1) + w(8) * a(i,j-1) + w(9) * a(i+1,j-1)] / 9;
-->    end
--> end
--> H = uint8(H);
--> subplot(1,2,1); imshow(image); title('Normal Image');
--> subplot(1,2,2); imshow(H); title('After High-Pass Filter');
```

⇨ **Output:**



Normal Image

After High-Pass Filter

High Pass Filter:

High pass filter is the type of frequency domain filter that is used for sharpening the image. It attenuates the low frequency components and preserve the high frequency components.

5. WAP to study the effect of the size of neighborhood on the result of processing.

⇨ // effect of the size of neighborhood on the result of processing

```
--> I = imread('man.jpg');
--> [m,n]=size(I);
```

// Applying 3*3 Mask

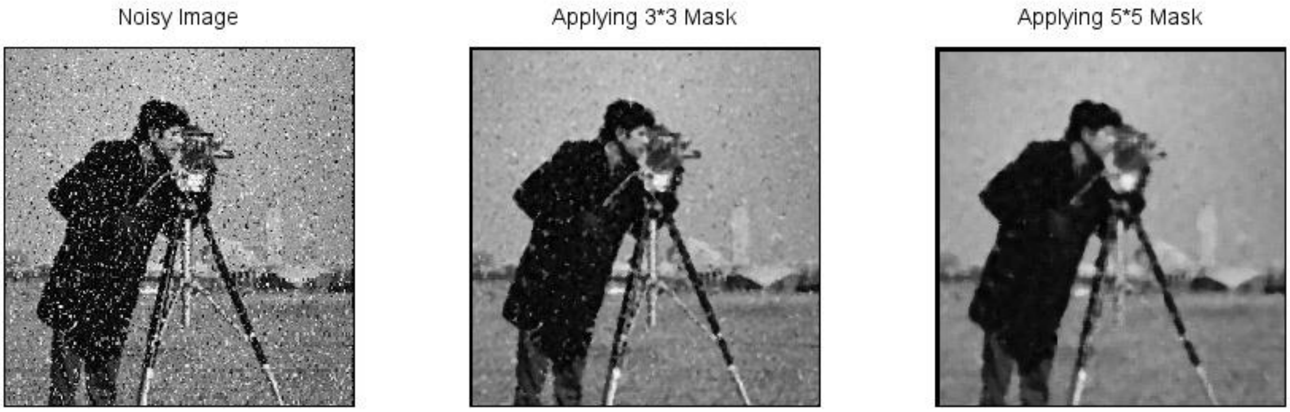
```
--> for i=2:m-1
-->     for j=2:n-1
-->         I2(i,j) = median([ a(i-1,j-1), a(i,j-1), a(i+1,j-1), a(i-1,j), a(i,j), a(i+1,j), a(i-1,j+1), a(i,j+1), a(i+1,j+1)]);
-->     end
--> end
```

// Applying 5*5 Mask

```
--> for i=3:m-2
-->     for j=3:n-2
-->         I3(i,j) = median([ a(i-2,j-2), a(i-1,j-2), a(i,j-2), a(i+1,j-2), a(i+2,j-2), a(i-2,j-1), a(i-1,j-1), a(i,j-1), a(i+1,j-1), a(i+2,j-1), a(i-2,j),a(i-1,j), a(i,j), a(i+1,j), a(i+2,j), a(i-2,j+1), a(i-1,j+1), a(i,j+1), a(i+1,j+1), a(i+2,j+1), a(i-2,j+2), a(i-1,j+2), a(i,j+2), a(i+1,j+2), a(i+2,j+2)]);
-->     end
--> end

--> subplot(1,3,1); imshow(a); title('Noisy Image');
--> subplot(1,3,2); imshow(I2); title('Applying 3*3 Mask');
--> subplot(1,3,3); imshow(I3); title('Applying 5*5 Mask');
```

⇨ **Output:**



Noisy Image

Applying 3*3 Mask

Applying 5*5 Mask

⇨ More salt & pepper noise is removed in case of 5*5 mask with a small degree of blur is added to the image. High degree of median filter can promise with an acceptable result with high density value of noise.