**Assignment – 1**

**Task 1:**

**Write a MATLAB script which reads, converts a color image into grayscale image, and performs the histogram equalization on the grayscale one. Note that you are not allowed to use histeq, instead you have to implement a function to do histogram equalization. Do find some examples where histogram equalization achieves the worse results. Please discuss the pros and cons of histogram equalization method.**

**Source code:**

clear all;

close all;

clc;

%% reading the color image

rgb\_img=imread('bear.jpg');

figure;

imshow(rgb\_img);

title('rgb color image')

%% converting the color image to gray image

gray\_img=rgb2gray(rgb\_img);

figure

imshow(gray\_img);

title('gray image')

%% calculating the histogram

gray\_img=double(gray\_img);

maxv\_gray=max((max(gray\_img)));

[x y]=size(gray\_img);

l=x\*y;

m=zeros(1,300);

n=zeros(1,300);

for i=1:x

for j=1:y

if gray\_img(i,j) == 0

gray\_img(i,j)=1;

end

end

end

for i=1:x

for j=1:y

t = gray\_img(i,j);

m(t) = m(t) + 1;

end

end

% The height of each bar is, (number of observations in the bin) / (total number of observations \* width of bin)

probf = m/l;

% The height of each bar is equal to the cumulative relative number of observations in the bin and all previous bins.

cumf(1) = probf(1);

for x=2:maxv\_gray

cumf(x) = probf(x) + cumf(x-1);

end

v = round(cumf \* maxv\_gray);

v = v + 1;

for r=1:x

for s=1:y

temp=gray\_img(r,s);

val(r,s)=v(temp);

t=val(r,s);

n(t)=n(t)+1;

end

end

val=val-1;

figure

subplot(2,1,1)

bar(m)

title('Histogram of the Original Image');

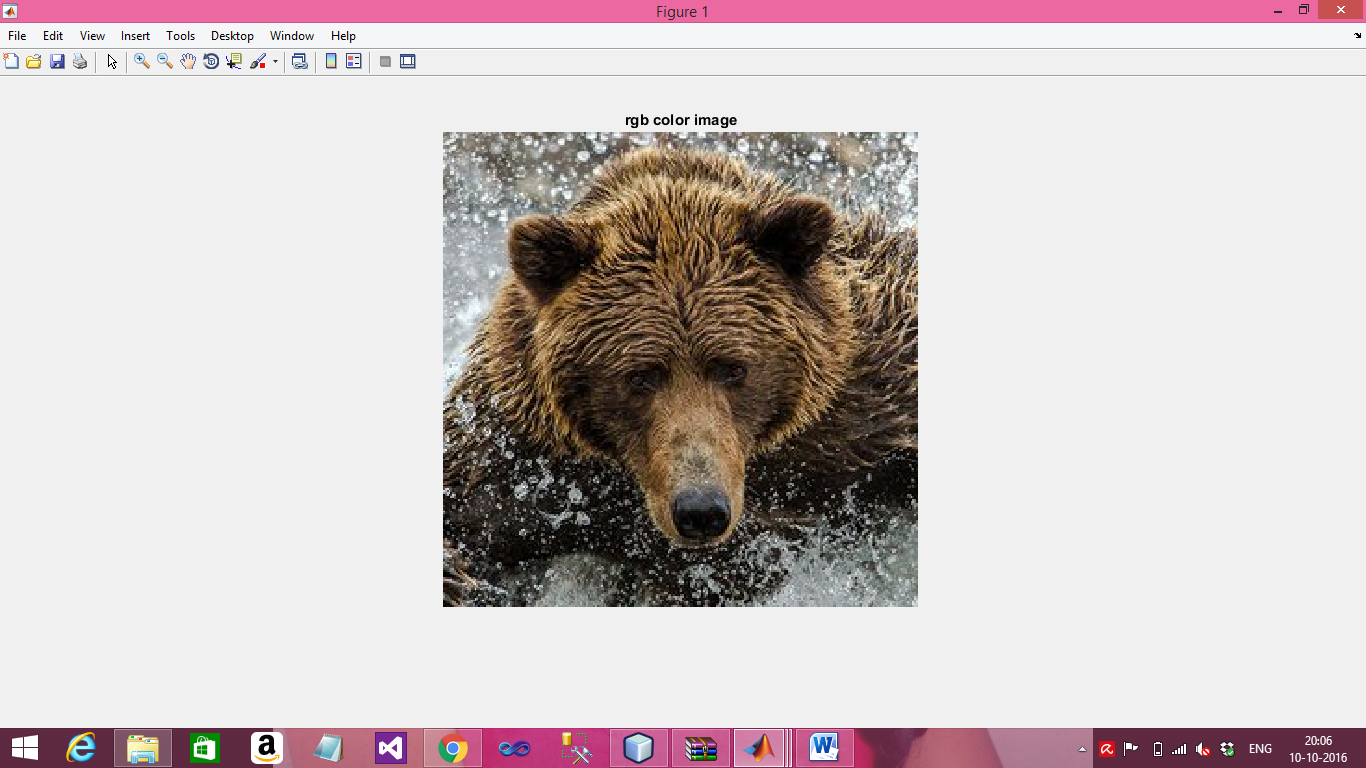
subplot(2,1,2)

bar(n)

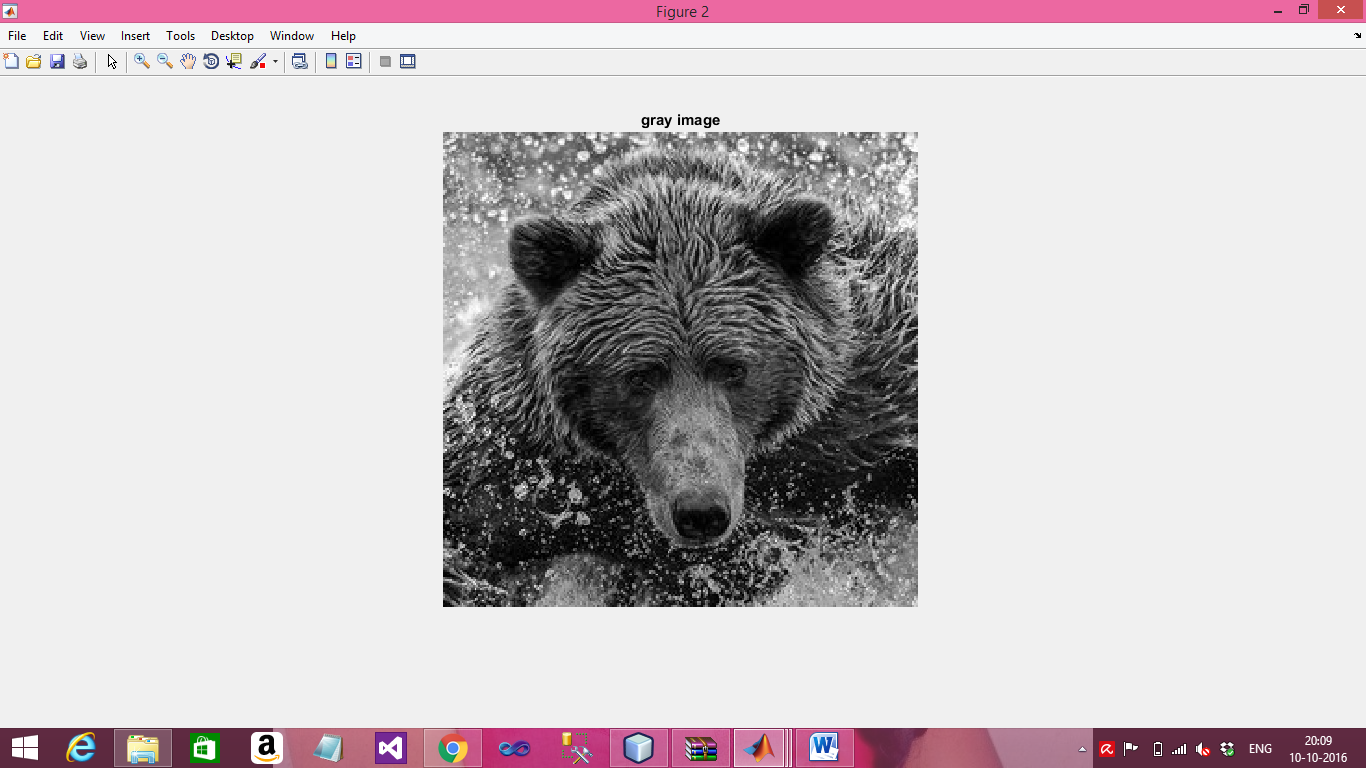
title('Histogram equalisation of Image');

**Output:**

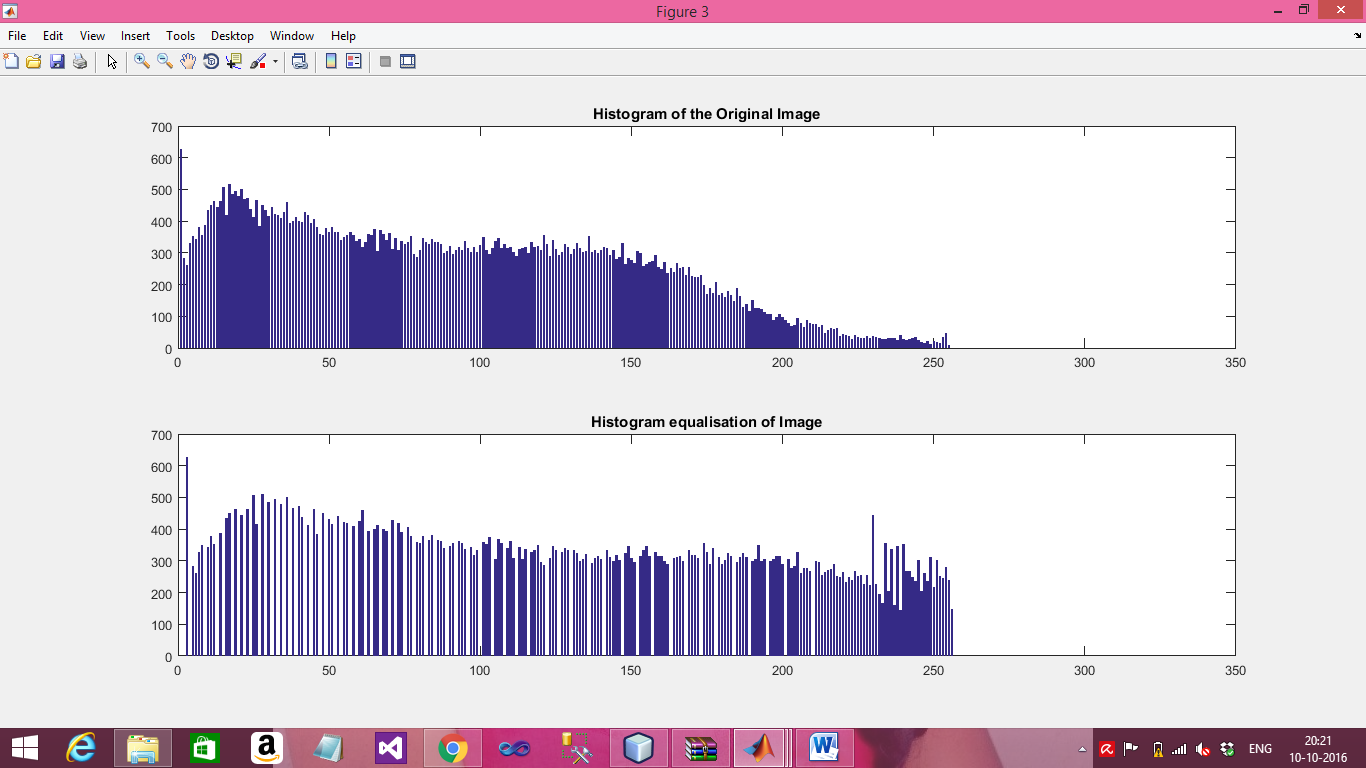
**Figure 1**



**Figure 2**



**Figure 3**



**Task 2 :**

**Write a MATLAB script which reads, converts a color image into grayscale image, and performs min, max and median filtering on the grayscale image. Please discuss/elaborate on each filtering method (min, max, and median).**

**Source code:**

Clc;

clear all;

close all;

image=imread('spin.jpg'); %read the image

figure

imshow(image)

title('rgb image')

image=rgb2gray(image); %convert the rgb image to grayscale image

figure

imshow(image); %show the original image

title('gray image')

image=imnoise(image,'salt & pepper',0.05);

figure,imshow(image) %show the noise image

title('noise image')

p=2\*2;

z=uint8(zeros(size(image)+2\*(p-1))); %padding the zeros

%loop for padding the zeros

for x=1:size(image,1)

for y=1:size(image,2)

z(x+p-1,y+p-1)=image(x,y);

end

end

%loop for finding the median & replacing the central pixel

for image= 1:size(z,1)-(p-1)

for j=1:size(z,2)-(p-1)

kernel=uint8(ones((p-1)^2,1));

t=1;

for x=1:p-1

for y=1:p-1

kernel(t)=z(image+x-1,j+y-1);

t=t+1;

end

end

filt=sort(kernel);

out(image,j)=filt(5);

end

end

figure,imshow(out);

title(' output image after median filtering')

%% min filter

A = imread('spin.jpg');

A = rgb2gray(A(1:250,1:250,:));

%preallocating the output matrix

L=zeros(size(A));

%zero padding

z=padarray(A,[1 1]);

x=[1:3]';

y=[1:3]';

for image= 1:size(z,1)-2

for j=1:size(z,2)-2

window=reshape(z(image+x-1,j+y-1),[],1);

L(image,j)=max(window);

L=uint8(L);

end

end

figure,imshow(L),title('output image after min filtering');

for image= 1:size(z,1)-2

for j=1:size(z,2)-2

window=reshape(z(image+x-1,j+y-1),[],1);

L(image,j)=min(window);

end

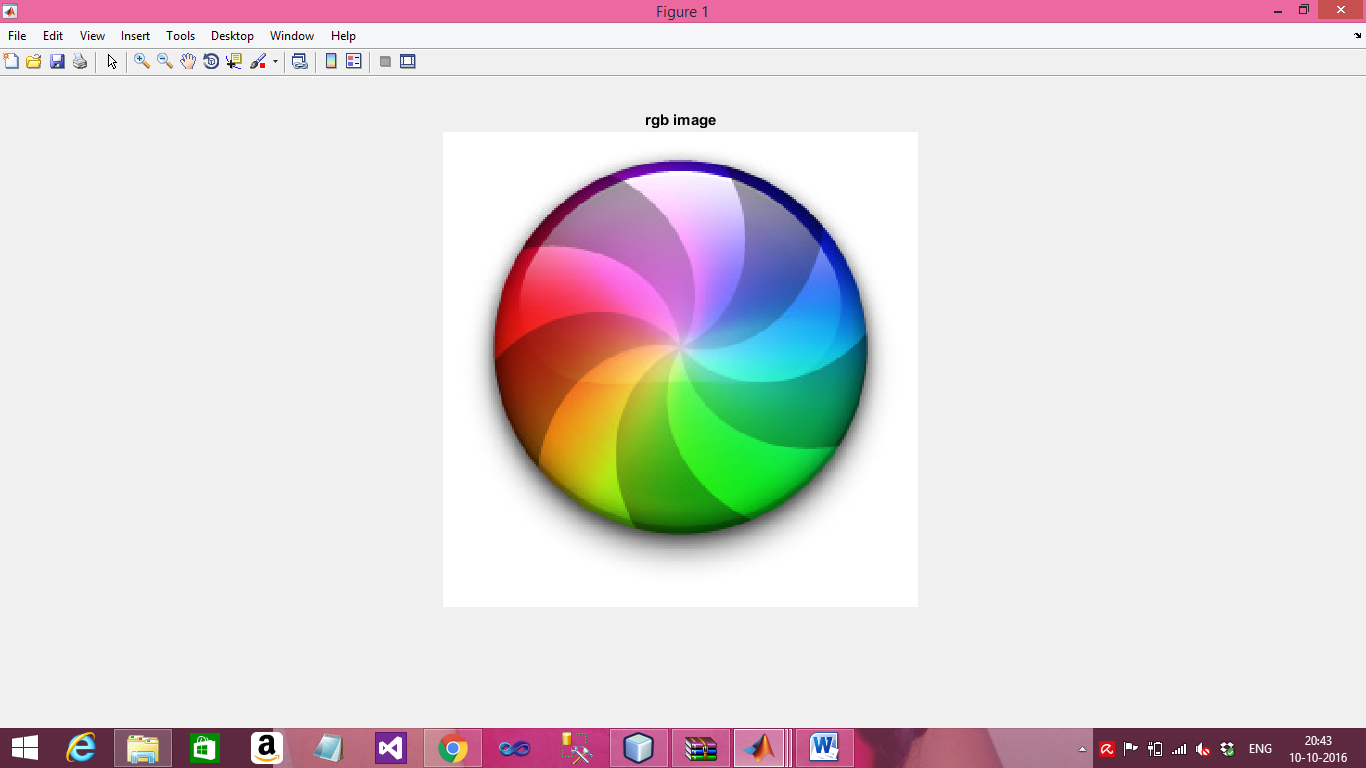
end

L=uint8(L);

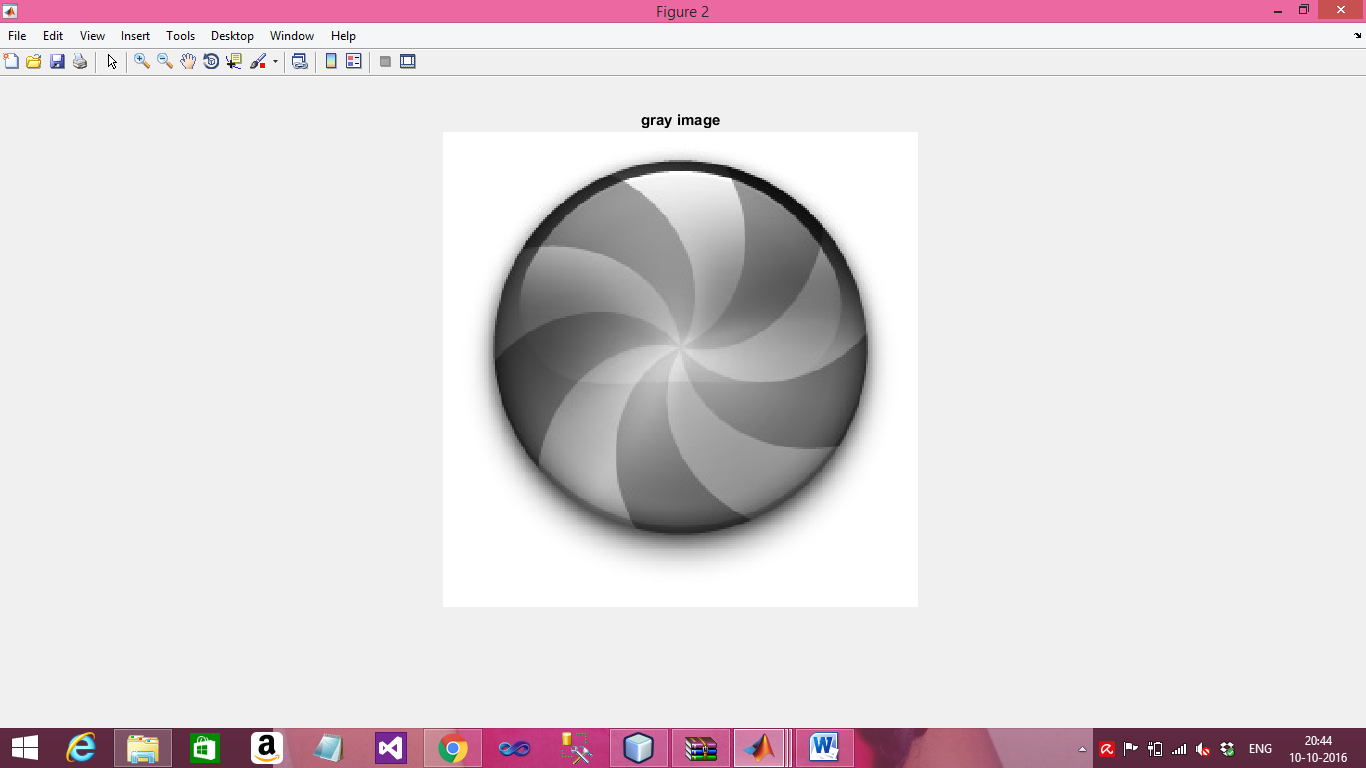
figure,imshow(L),title('output image after max filtering');

**Output:**

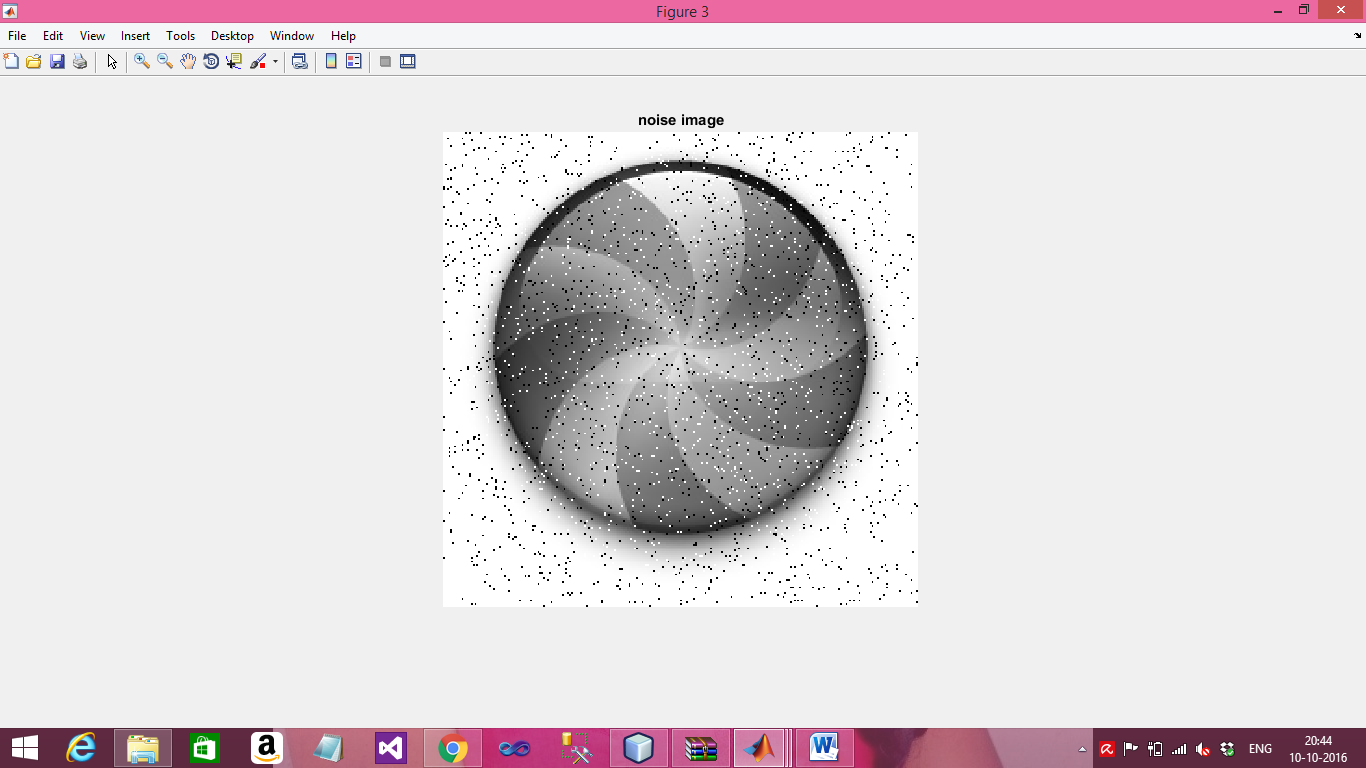
**Figure 1**



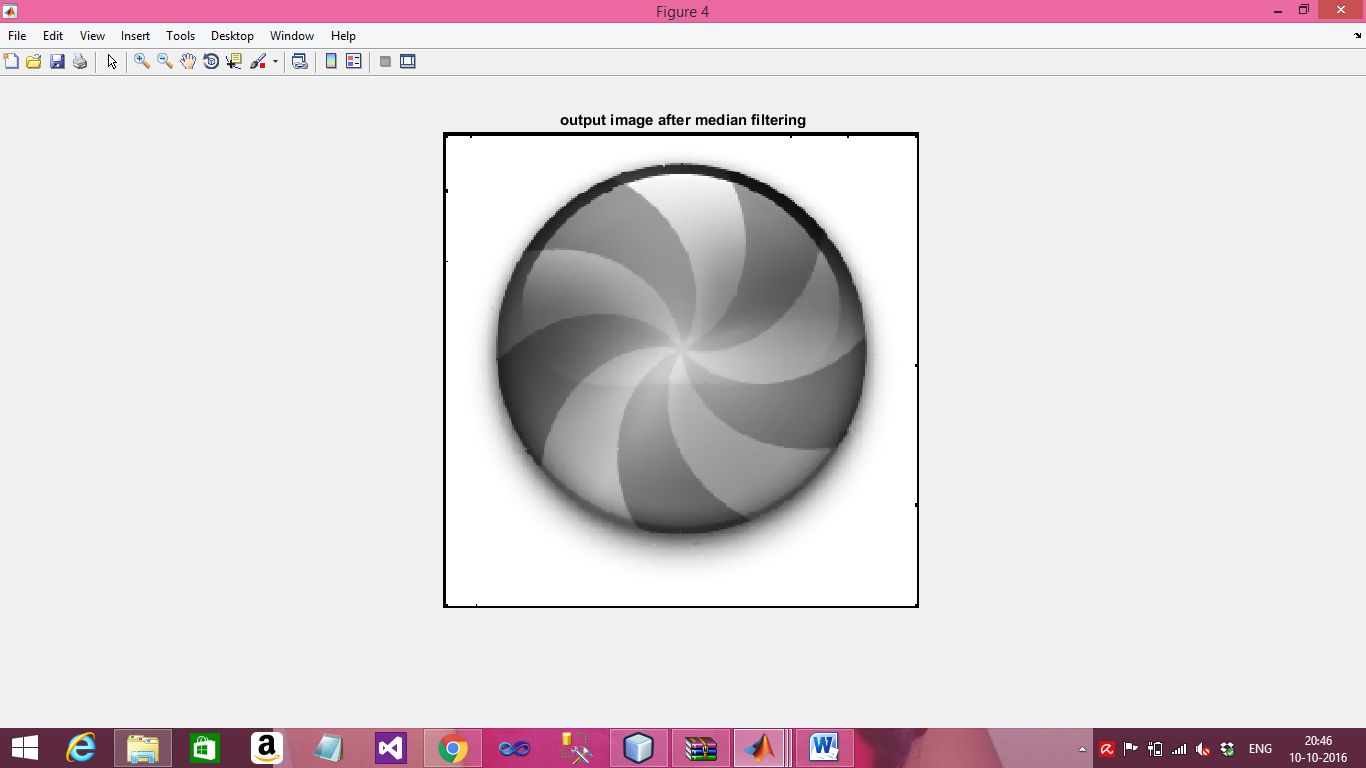
**Figure 2**



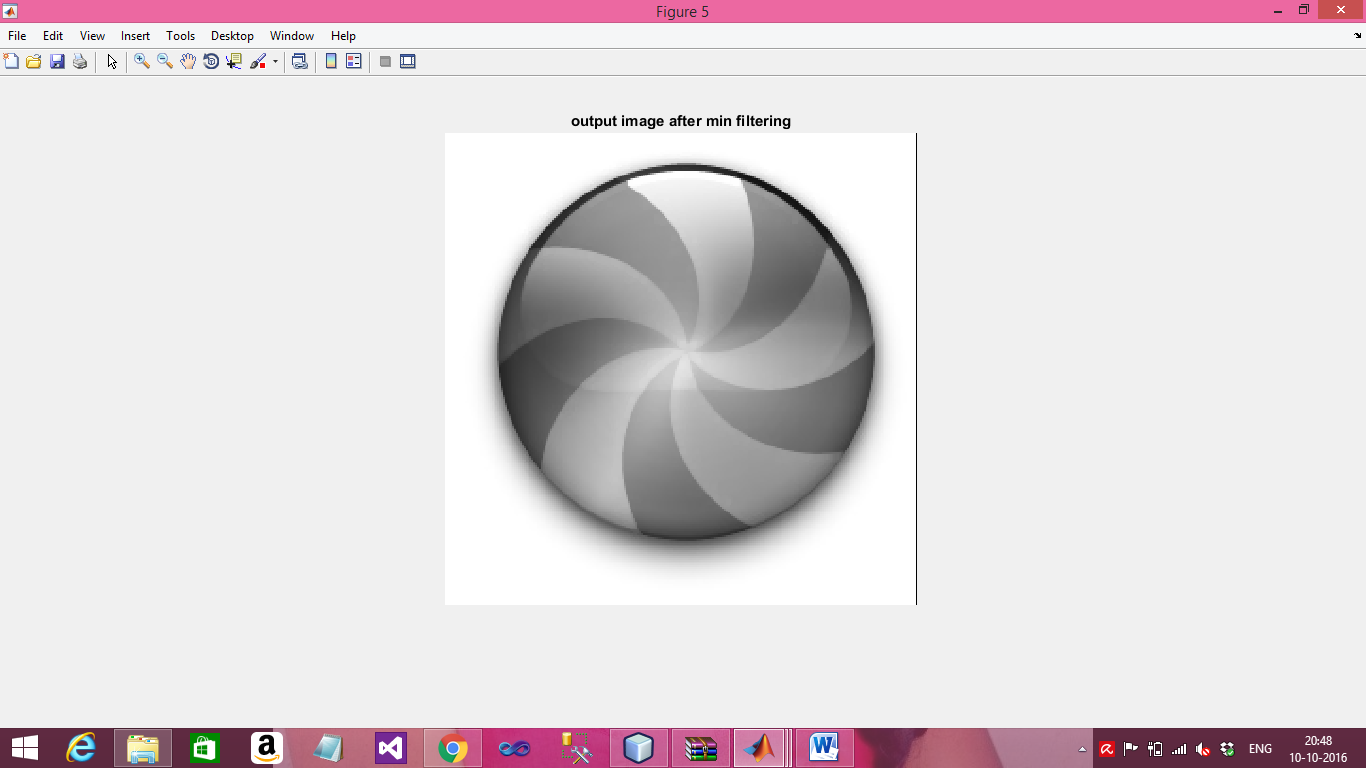
**Figure 3**



**Figure 4**



**Figure 5**



**Figure 6**

