**CODING:**

Ioriginal = imread('cameraman.tif');

imshow(Ioriginal);

title('Original Image');

Ioriginal = imread('cameraman.tif');

Idouble = im2double(Ioriginal);

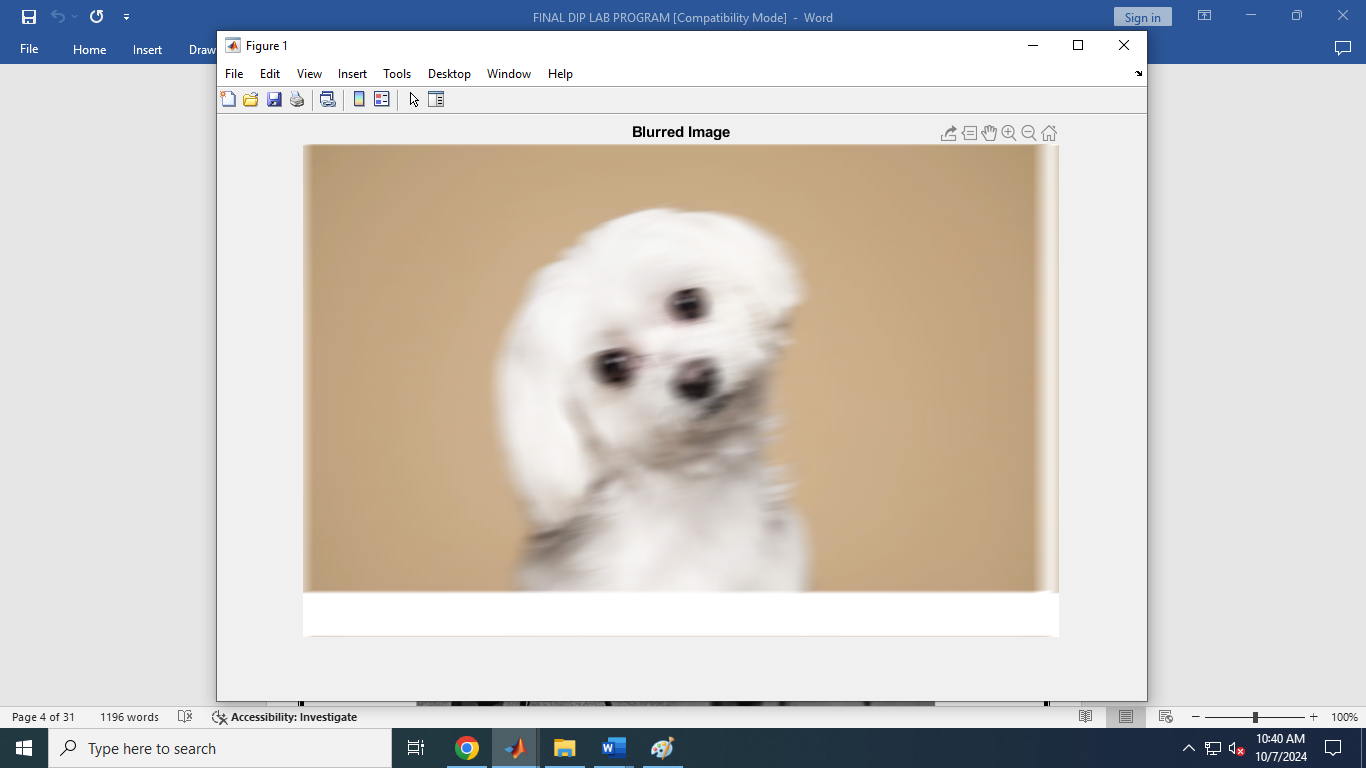
PSF = fspecial('motion', 21, 11);

blurred = imfilter(Idouble, PSF, 'conv', 'circular');

imshow(blurred);

title('Blurred Image')

# OUTPUT:



**CODING:**

a=imread('flower3.jpg');

b=rgb2gray(a);

subplot(2,2,1);

imshow(b);

title('original grayscale image');

subplot(2,2,3);

imhist(b);

title('histrogram of original grayscale image');

j=histeq(b);

subplot(2,2,2);

imshow(j);

title('image after histogram equalization');

subplot(2,2,4);

imhist(j);

title('histogram of image after histogram equalization');

**Output:**



**CODING:**

originalImage = imread('rahul.png');

originalImage = rgb2gray(originalImage);

h = fspecial('motion', 15, 45);

blurredImage = imfilter(originalImage, h, 'replicate');

figure;

subplot(1, 2, 1);

imshow(originalImage);

title('Original Image');

subplot(1, 2, 2);

imshow(blurredImage);

title('Blurred Image');

F = fft2(blurredImage);

H = fft2(h, size(blurredImage, 1), size(blurredImage, 2));

epsilon = 0.01;

restoredImageFreq = F ./ (H + epsilon);

restoredImage = real(ifft2(restoredImageFreq));

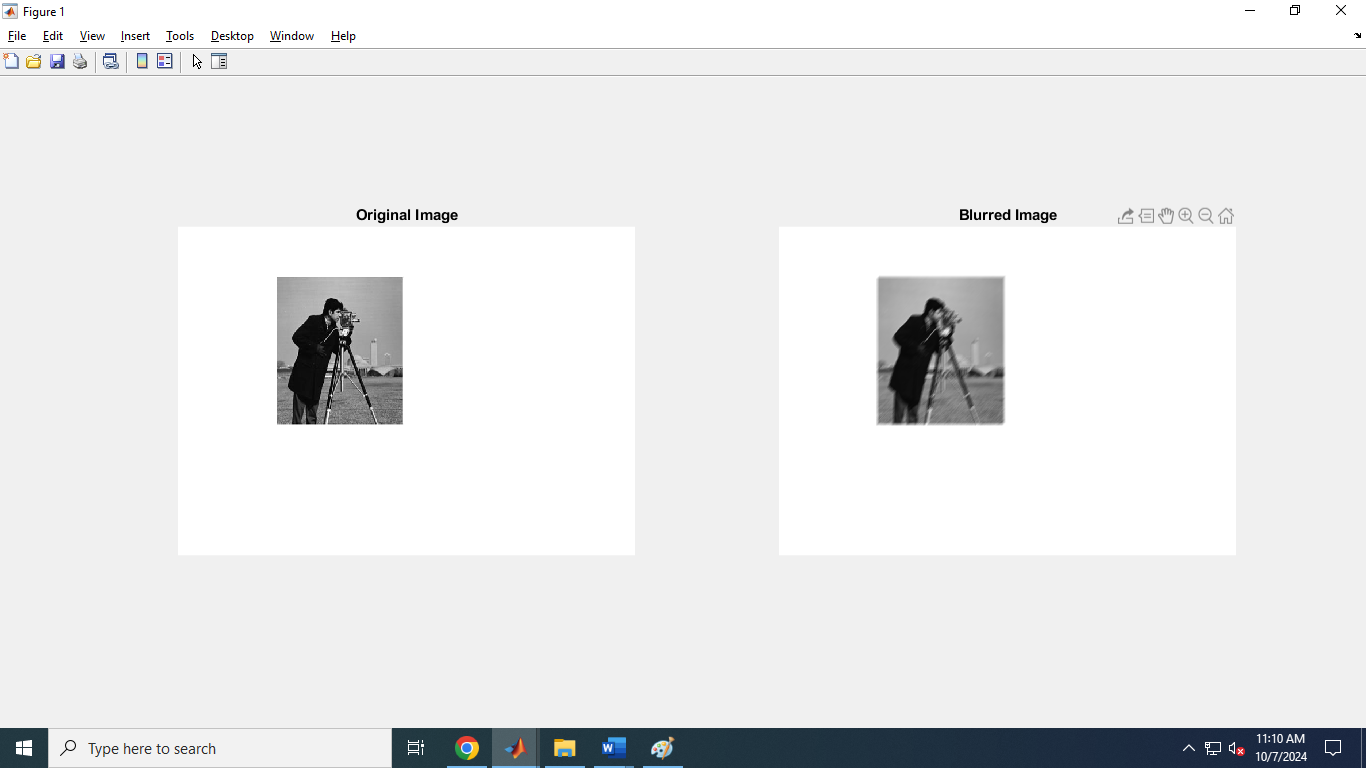
restoredImage = uint8(restoredImage);

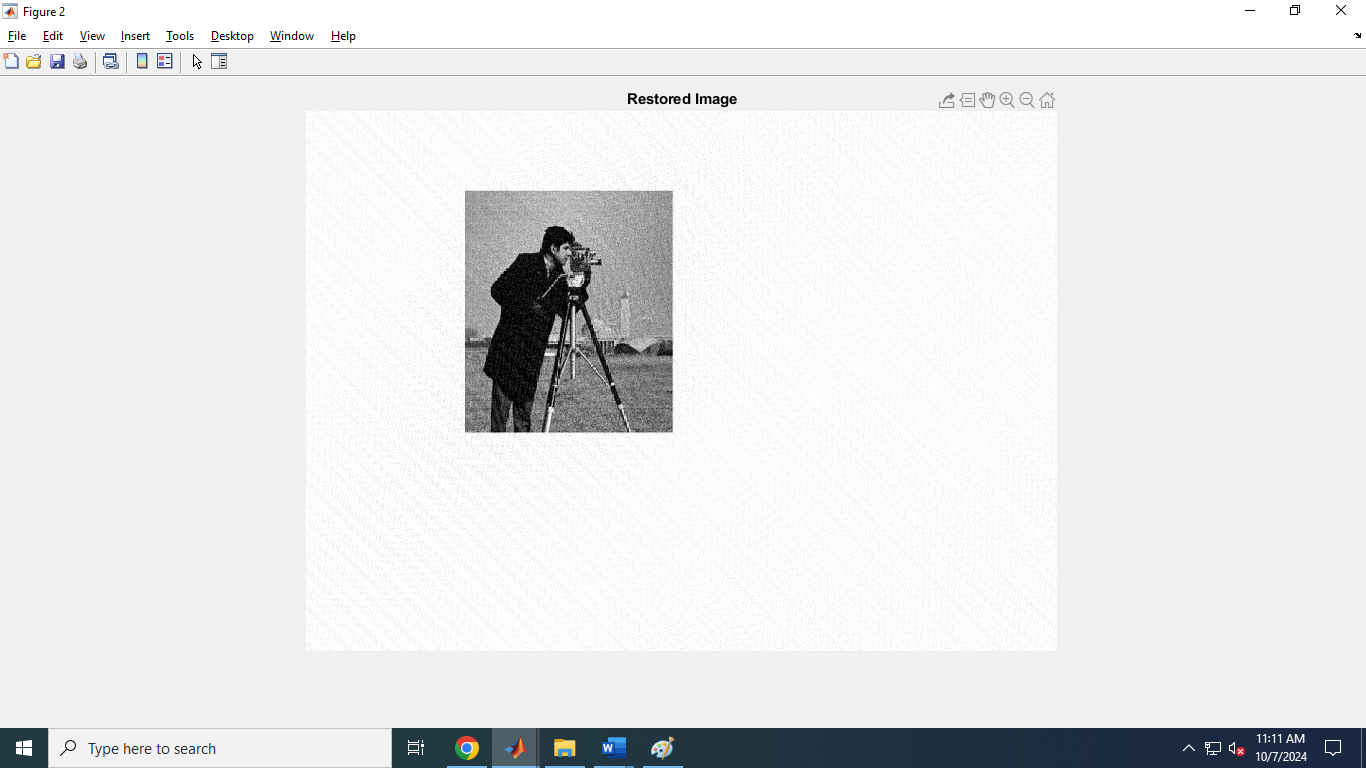
figure;

imshow(restoredImage);

title('Restored Image');

**Output:**





**CODING:**

image = imread('monika.jpg');

imshow(image);

title('Original Image');

if size(image, 3) == 3

image = rgb2gray(image);

end

filterSize = 5;

sigma = 2;

h = fspecial ('gaussian', [filterSize filterSize], sigma);

filtered image = imfilter(image, h, 'replicate');

figure;

imshow(filtered\_image, []);

title('Filtered Image with Gaussian Blur');

**Output:**





**Roberts operator**

**CODING:**

img = imread('d.png');

grayImg = rgb2gray(img);

robertsX = [1 0; 0 -1];

robertsY = [0 1; -1 0];

edgesX = imfilter(double(grayImg), robertsX, 'replicate');

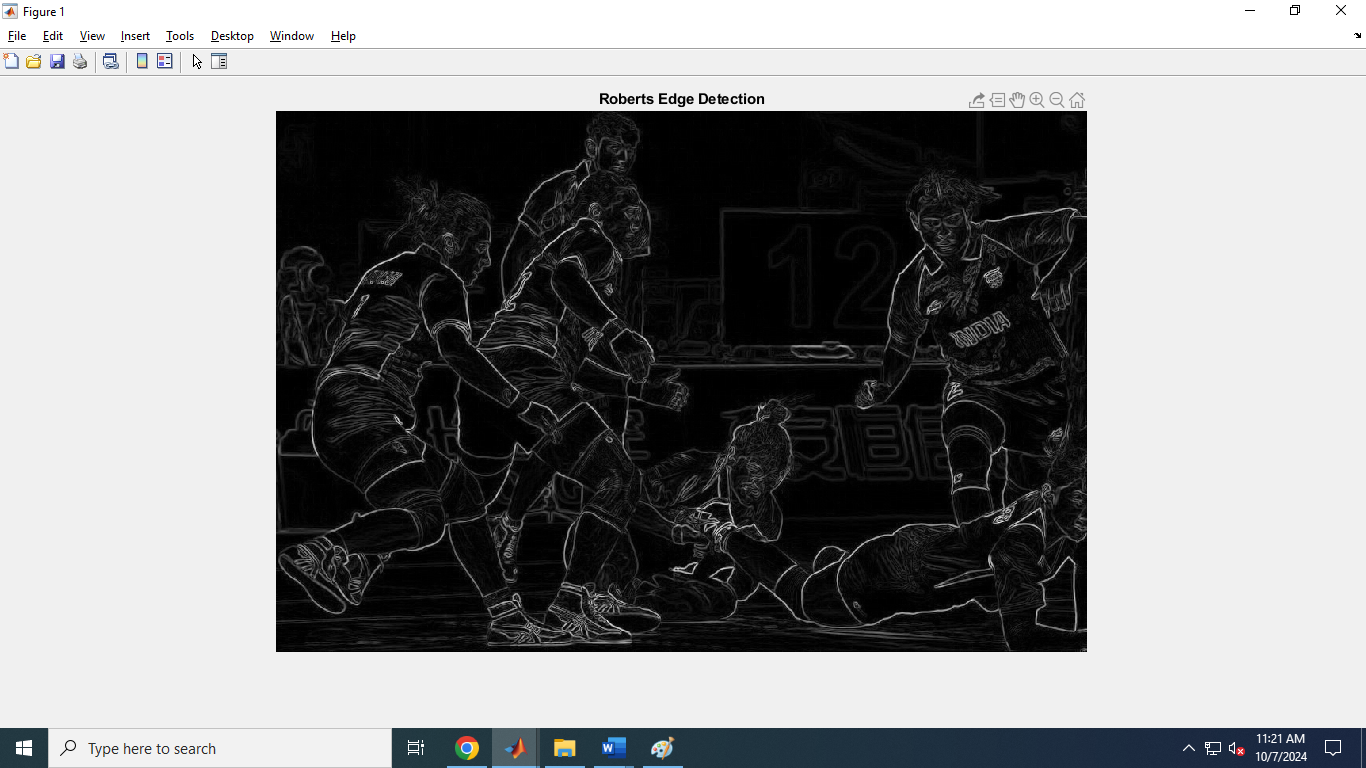
edgesY = imfilter(double(grayImg), robertsY, 'replicate');

edges = sqrt(edgesX.^2 + edgesY.^2);

imshow(edges, []);

title('Roberts Edge Detection');

**Output:**



**Prewitts operator**

**CODING:**

img = imread('d.png');

grayImg = rgb2gray(img);

prewittX = [-1 0 1; -1 0 1; -1 0 1];

prewittY = [-1 -1 -1; 0 0 0; 1 1 1];

edgesX = imfilter(double(grayImg), prewittX, 'replicate');

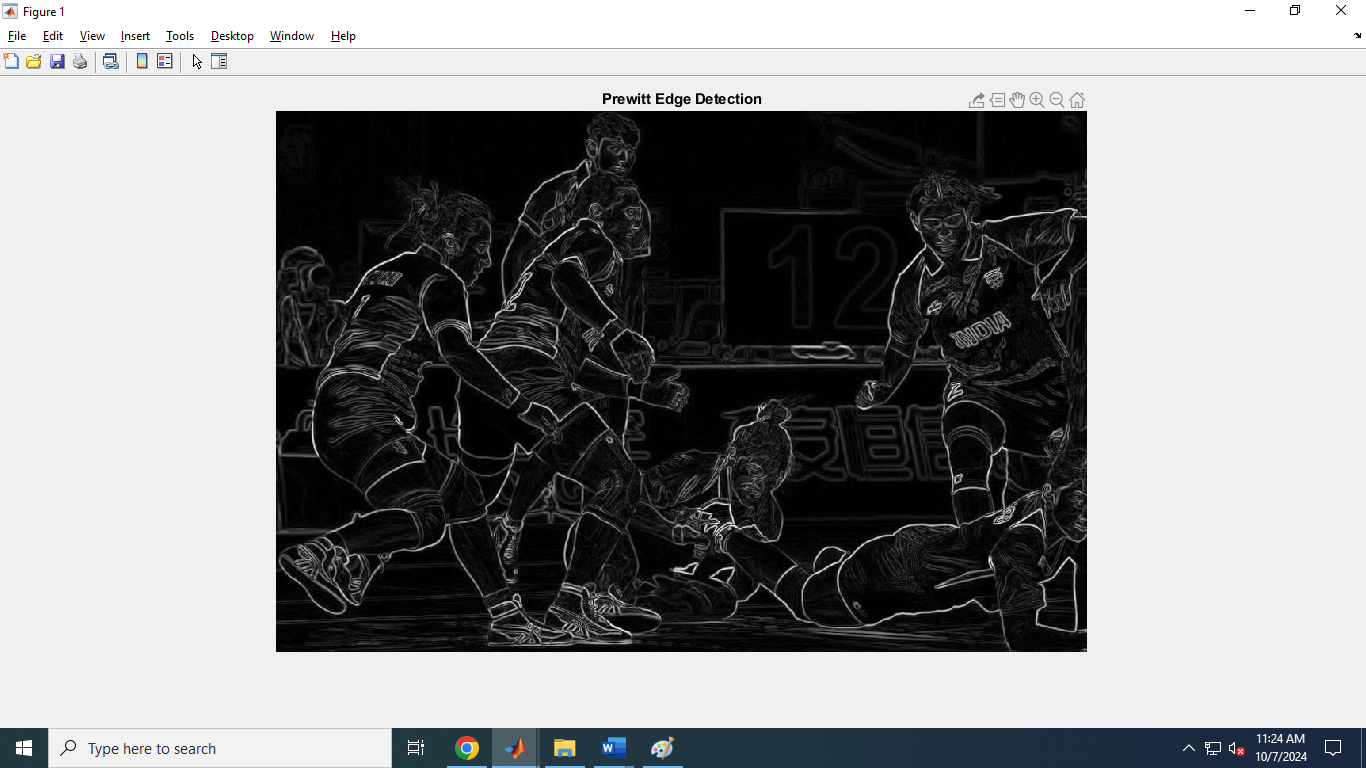
edgesY = imfilter(double(grayImg), prewittY, 'replicate');

edges = sqrt(edgesX.^2 + edgesY.^2);

imshow(edges, []);

title('Prewitt Edge Detection');

**Output:**



**Sobel operators**

**CODING:**

img = imread('d.png');

grayImg = rgb2gray(img); sobelX = [-1 0 1; -2 0 2; -1 0 1];

sobelY = [-1 -2 -1; 0 0 0; 1 2 1];

edgesX = imfilter(double(grayImg), sobelX, 'replicate');

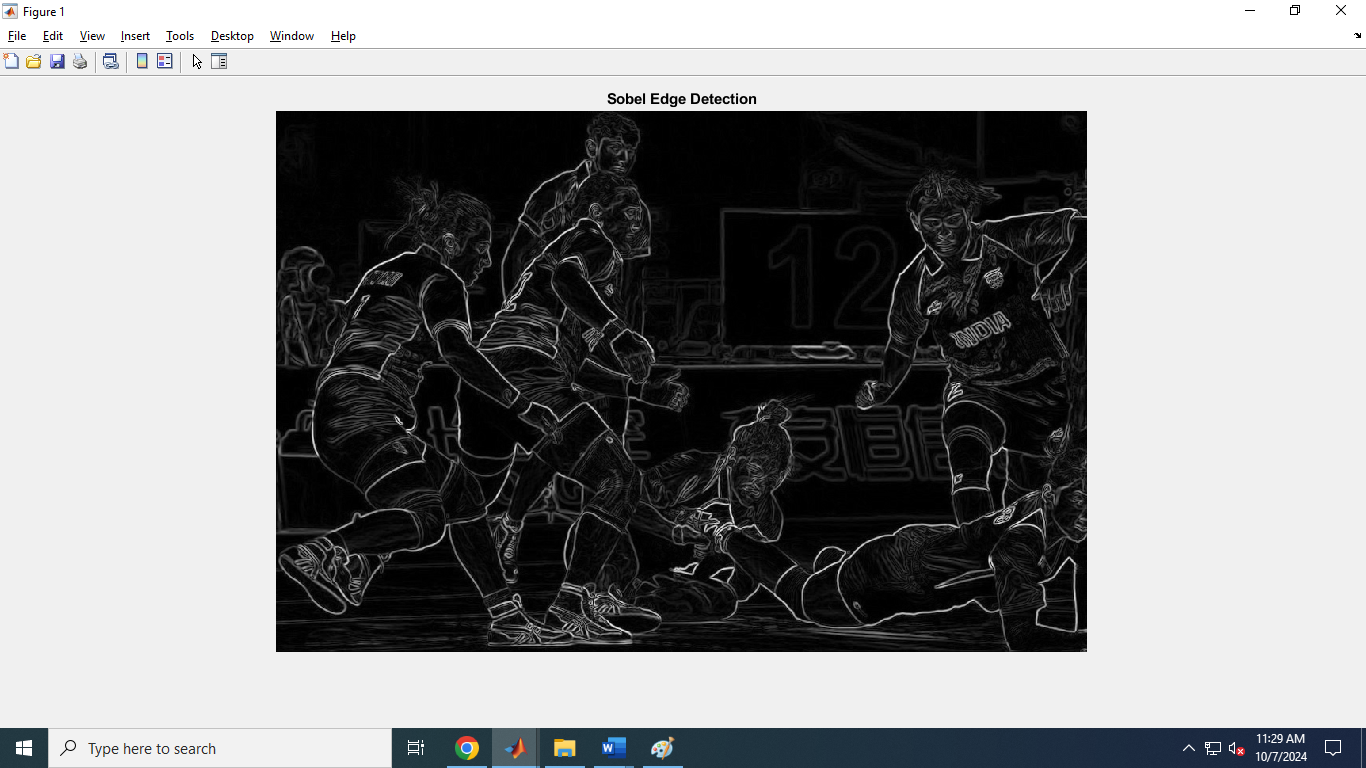
edgesY = imfilter(double(grayImg), sobelY, 'replicate');

edges = sqrt(edgesX.^2 + edgesY.^2);

imshow(edges, []);

title('Sobel Edge Detection');

**Output:**



**CODING:**

originalImage = imread('moni.png');

originalImage = rgb2gray(originalImage);

figure;

imshow(originalImage);

title('Original Image');

dctImage = dct2(originalImage);

threshold = 20;

compressedImage = dctImage;

compressedImage(abs(compressedImage) < threshold) = 0;

reconstructedImage = idct2(compressedImage);

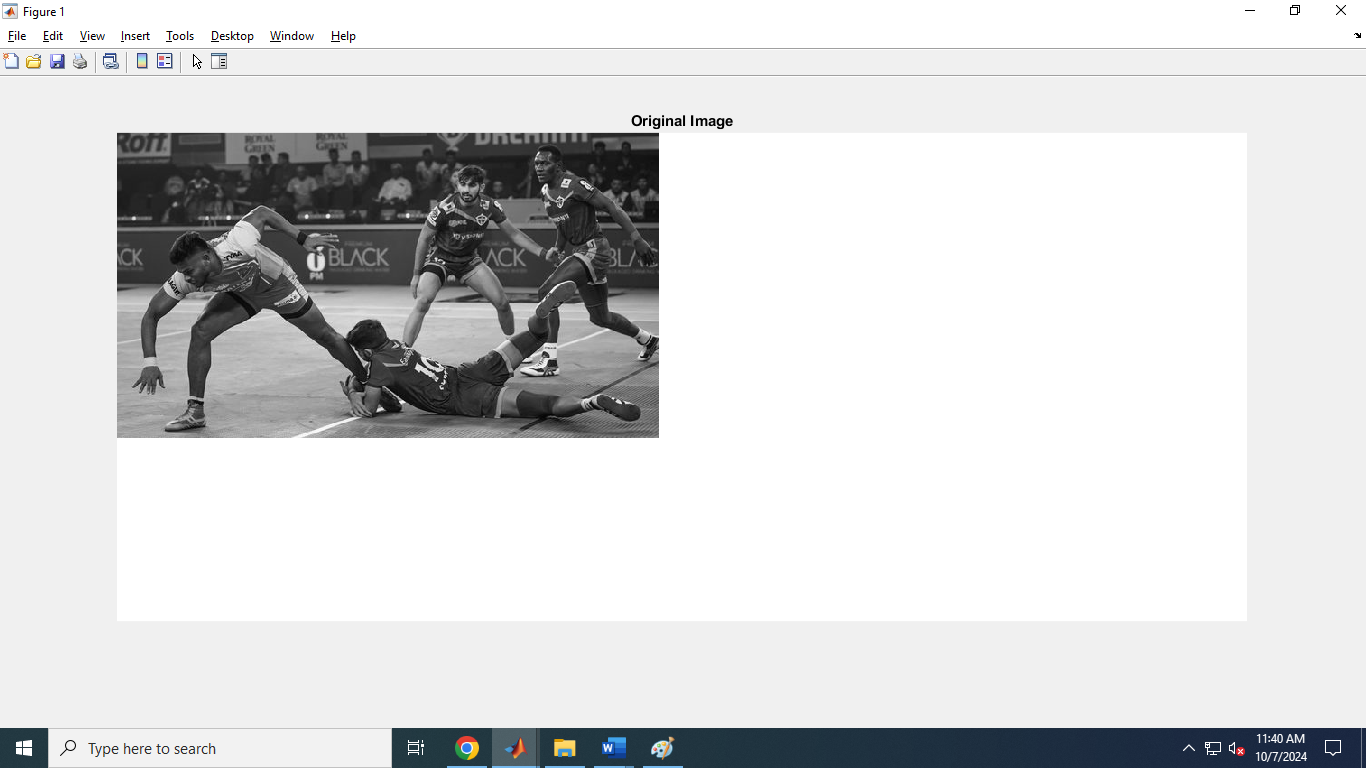
reconstructedImage = uint8(reconstructedImage);

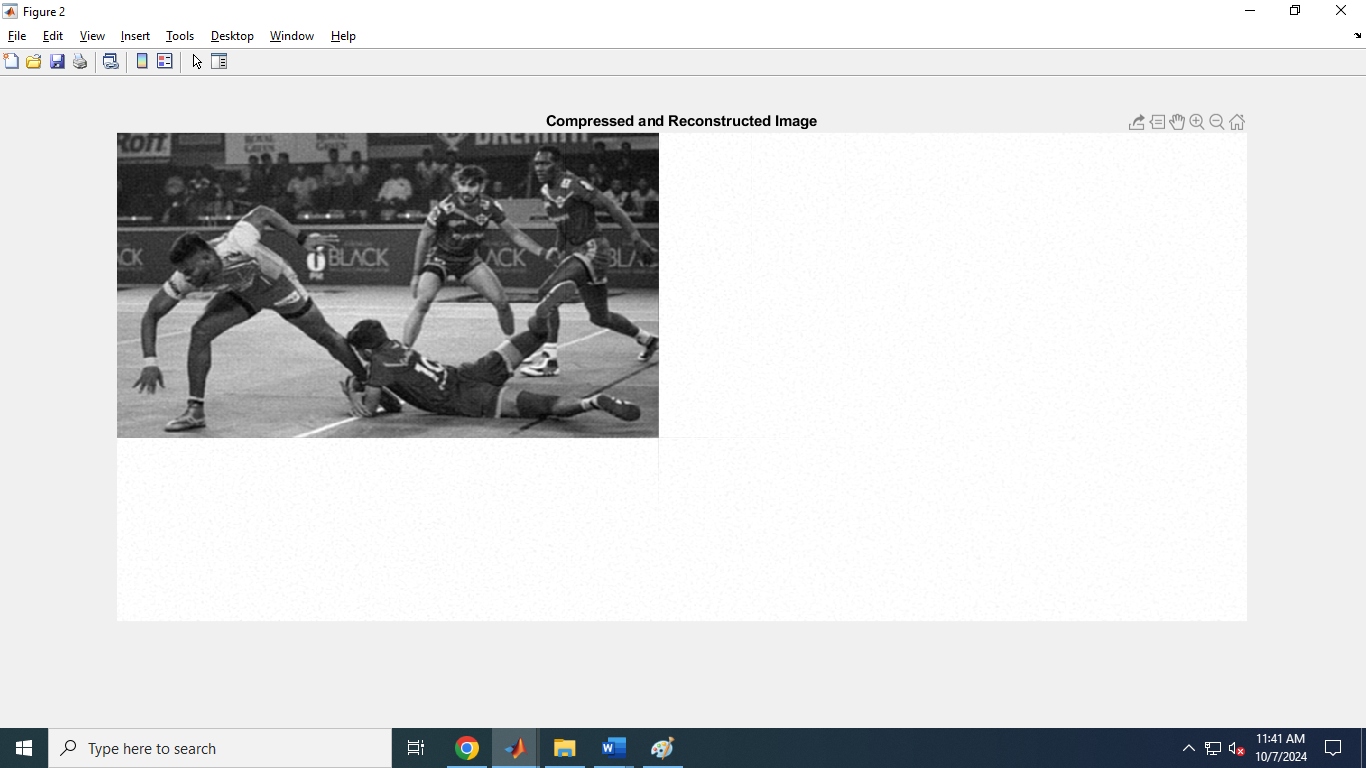
figure;

imshow(reconstructedImage);

title('Compressed and Reconstructed Image');

**Output:**





**CODING:**

I=imread('price.png');

background=imopen(I,strel('disk',15));

Ip = imsubtract(I,background);

imshow(Ip,[])

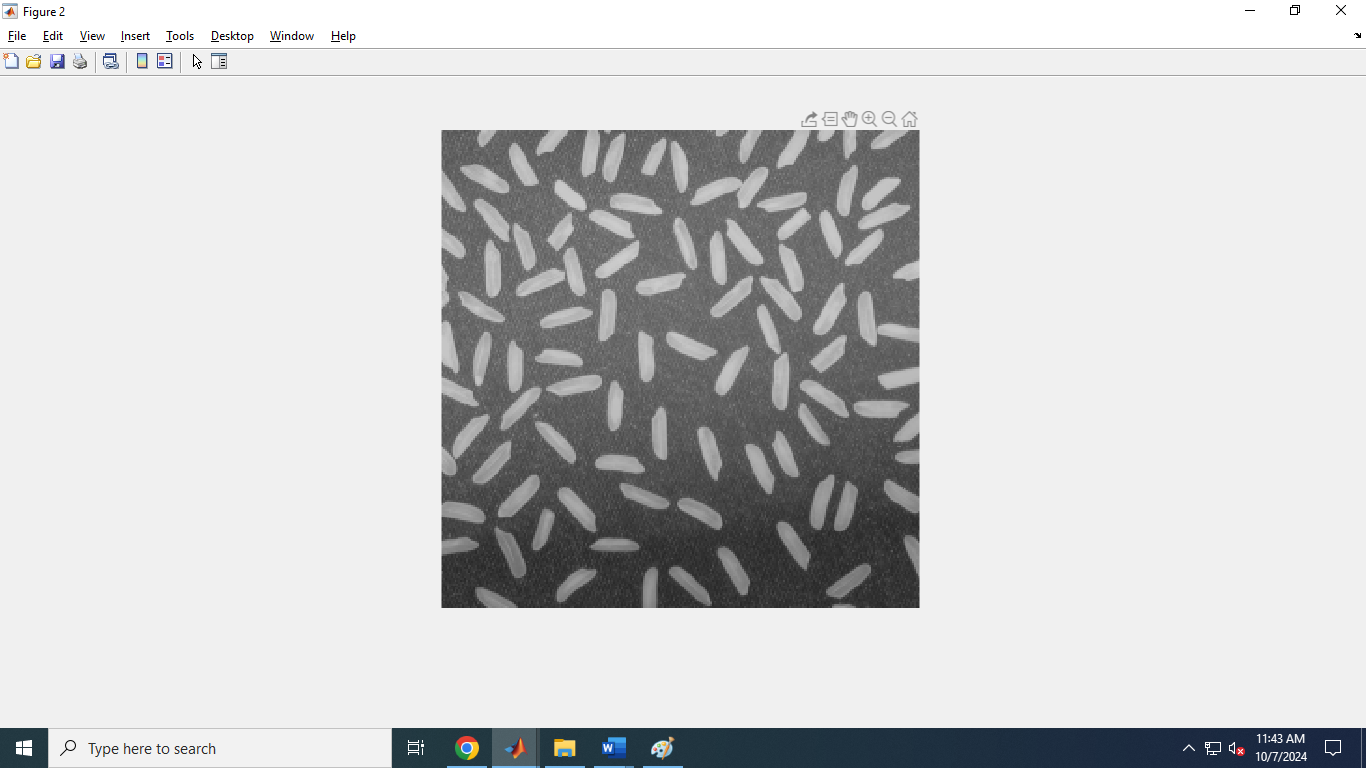
I=imread('rice.png');

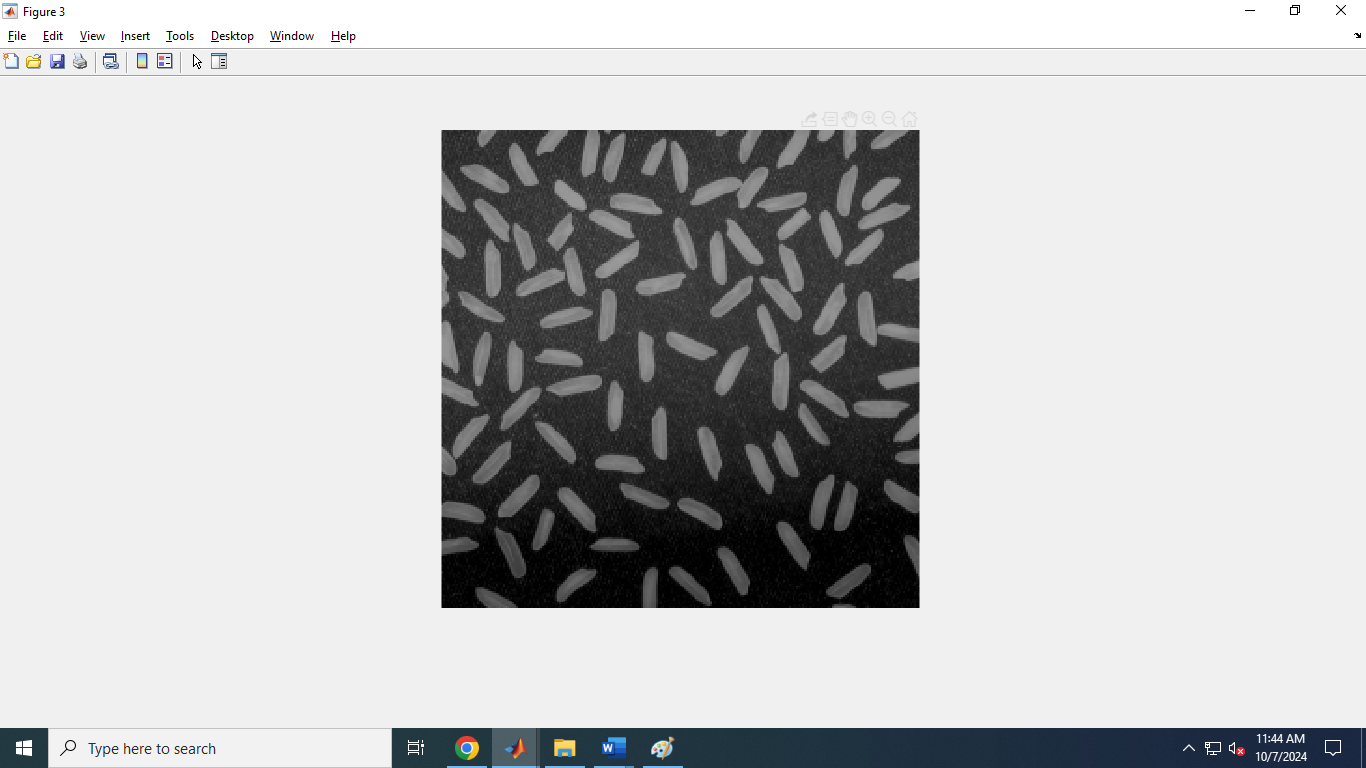
Iq = imsubtract(I,50);

figure,imshow(I),

figure,imshow(Iq)

**Output:**





**CODING:**

binaryImage = imread('black.png');

binaryImage = im2bw(binaryImage);

figure;

imshow(binaryImage);

title('Original Binary Image');

SE = strel('disk', 2);

erodedImage = imerode(binaryImage, SE);

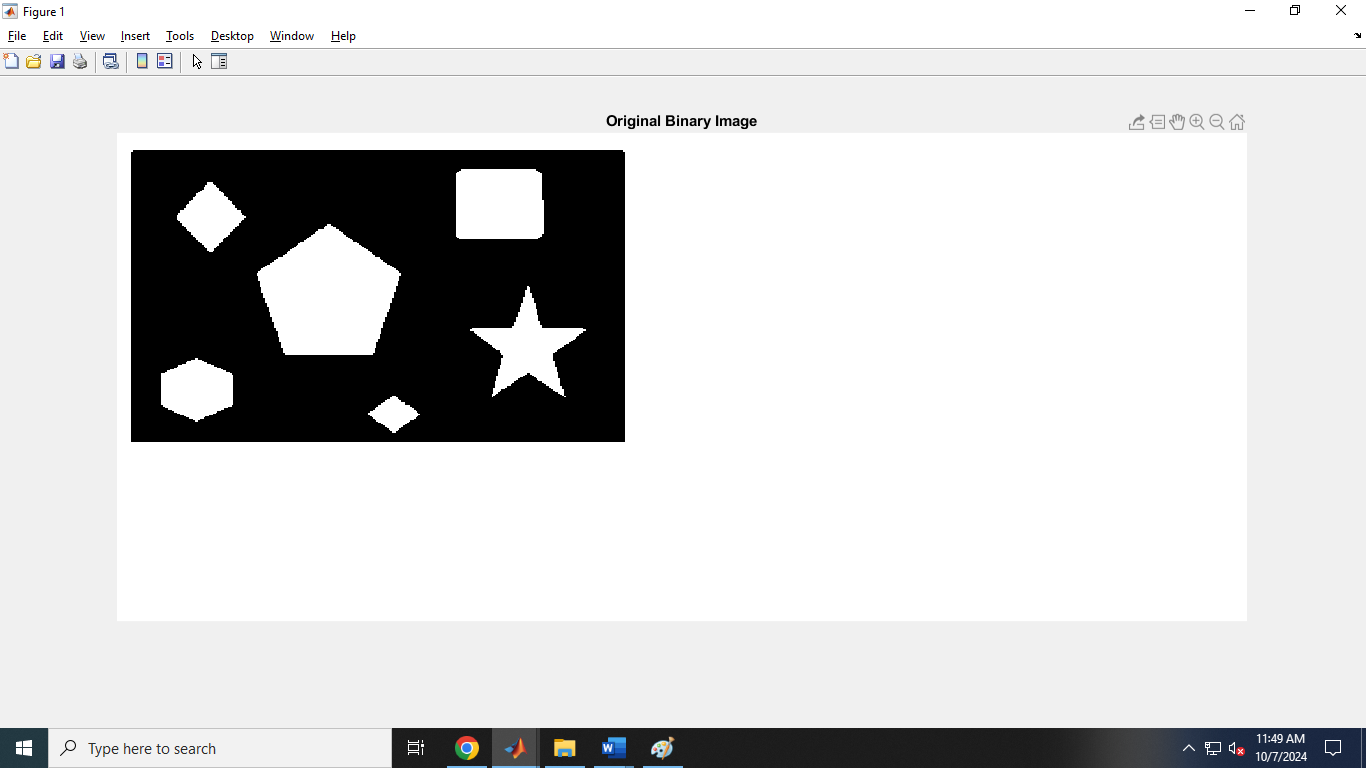
boundaryImage = binaryImage - erodedImage;

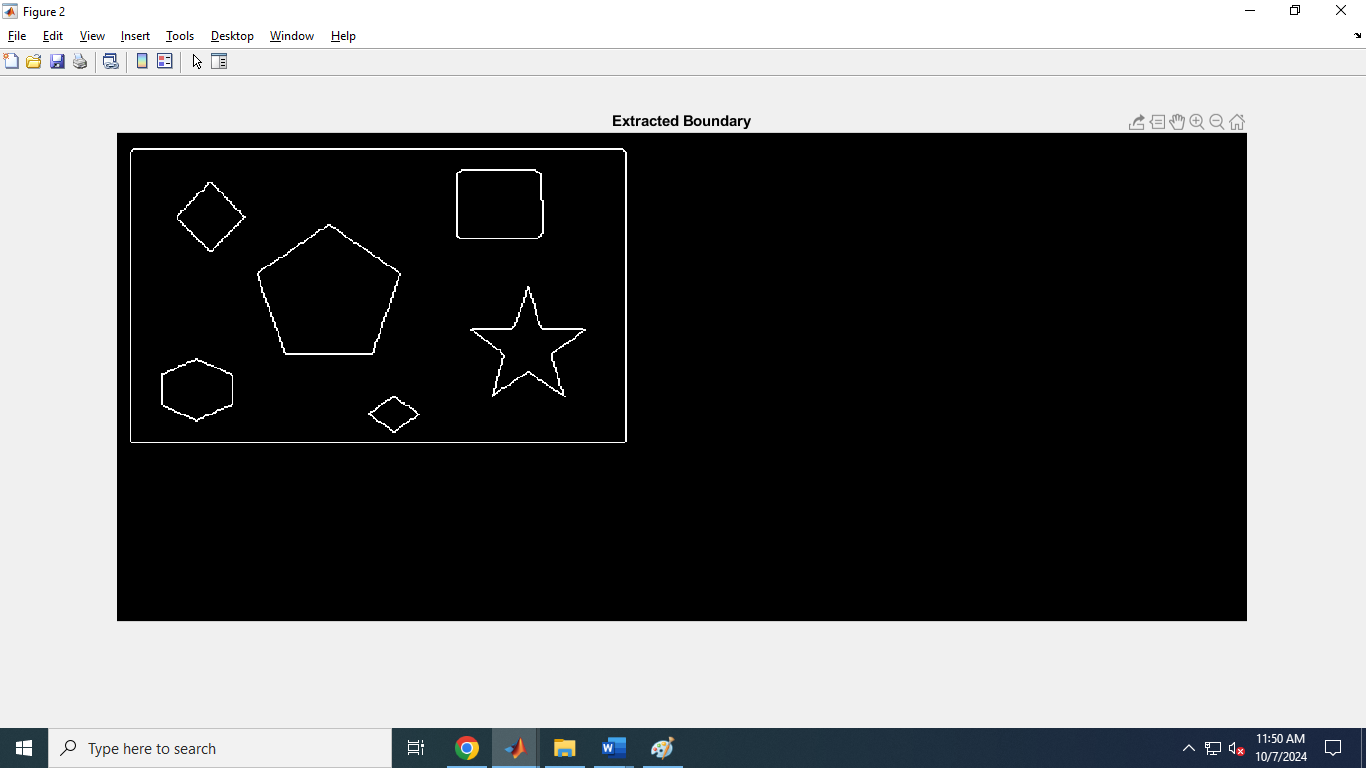
figure;

imshow(boundaryImage);

title('Extracted Boundary');

**Output:**





**CODING:**

image = imread('moni.jpg');

threshold\_value = graythresh(image);

threshold\_image = im2bw(image,threshold\_value);

subplot(1,2,1),imshow(threshold\_image);

subplot(1,2,2), imshow(image);

**Output:**

