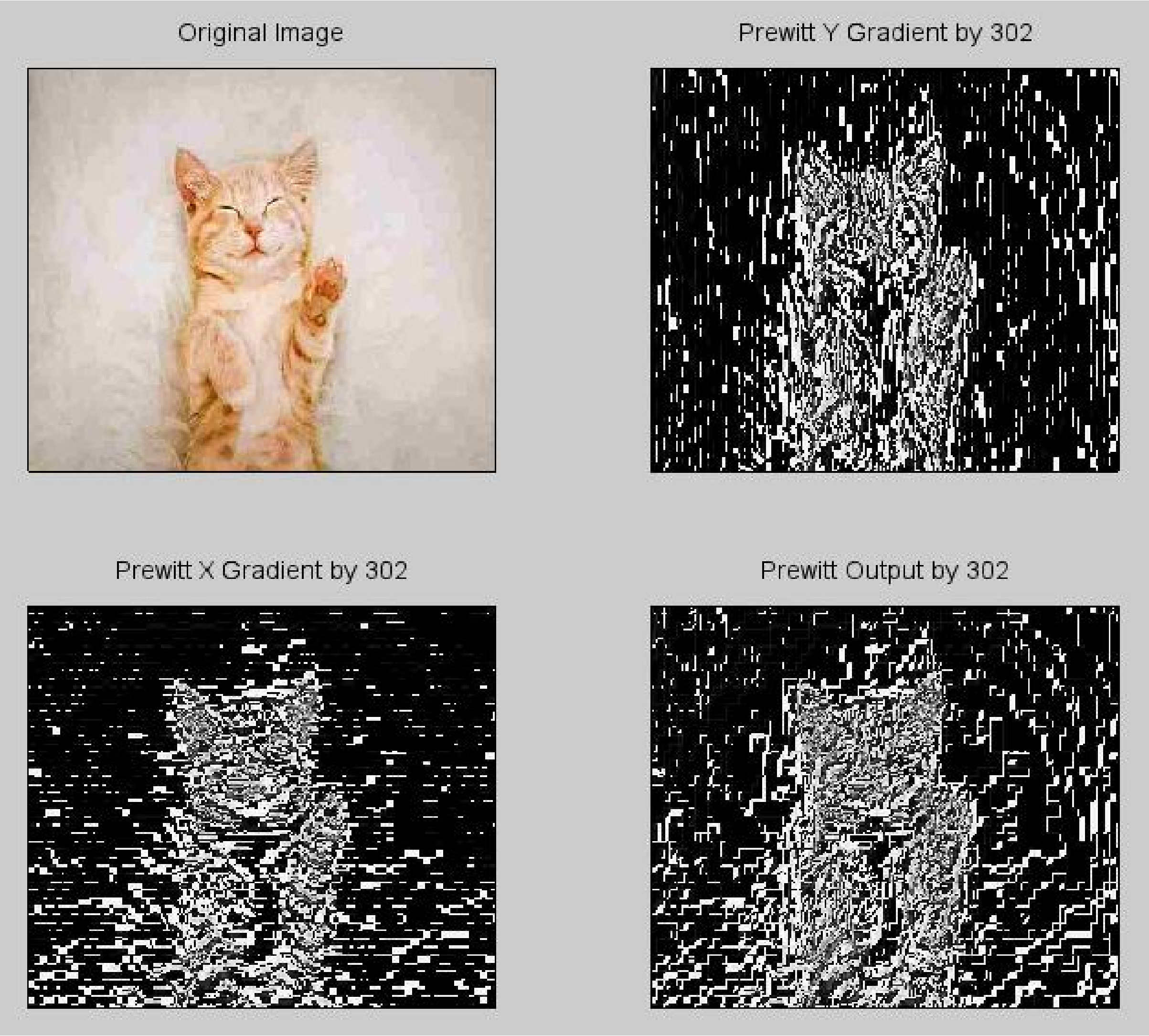
Practical 9 (Prewitt & Sobel)

Code:

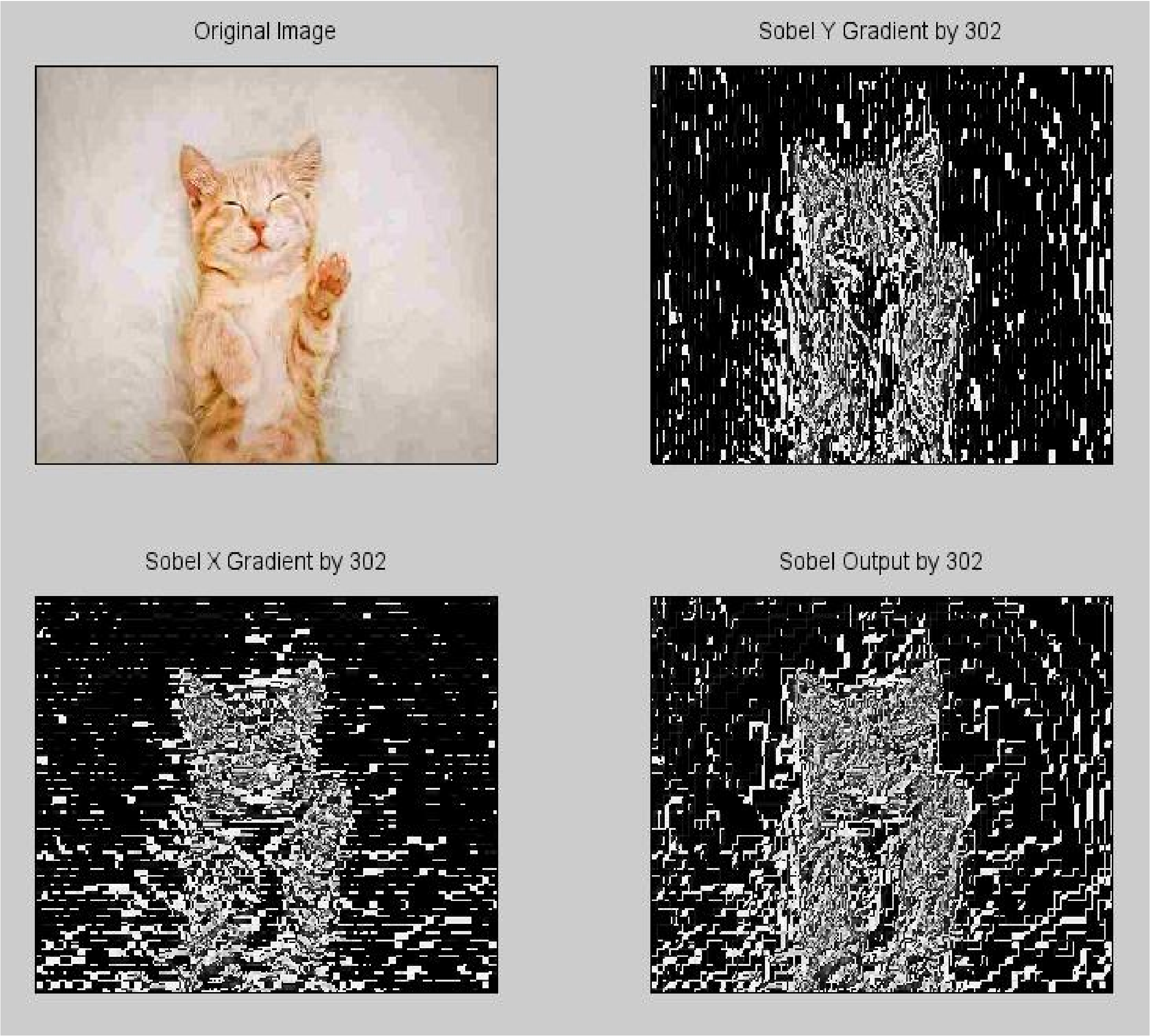
```
clc;
clear all;
a=imread("F:\dip\cute_11.jpg");
a=double(a); //[m,n]=size(a);
//Code For Prewitt
w1=[-1 -1 -1; 0 0 0; 1 1 1];
w2=[-1 0 1; -1 0 1; -1 0 1];
[row col]=size(a);
for x=2:row-1
    for y=2:col-1
        a1(x,y)=[w1(1)*a(x-1,y-1)+w1(2)*a(x-1,y)+w1(3)*a(x-1,y+1)+w1(4)
*a(x,y-1)+w1(5)*a(x,y)+w1(6)*a(x,y+1)+w1(7)*a(x+1,y-1)+w1(8)*a(x+1,y)+
w1(9)*a(x+1,y+1)];

        a2(x,y)=w2(1)*a(x-1,y-1)+w2(2)*a(x-1,y)+w2(3)*a(x-1,y+1)+w2(4)*
a(x,y-1)+w2(5)*a(x,y)+w2(6)*a(x,y+1)+w2(7)*a(x+1,y-1)+w2(8)*a(x+1,y)+w
2(9)*a(x+1,y+1);
      end
end
//Code For Sobel
s1=[-1 -2 -1; 0 0 0; 1 2 1];
s2=[-1 0 1; -2 0 2; -1 0 1];
[row col]=size(a);
for x=2:row-1
    for y=2:col-1
        b1(x,y)=[s1(1)*a(x-1,y-1)+s1(2)*a(x-1,y)+s1(3)*a(x-1,y+1)+s1(4)*a(
x,y-1)+s1(5)*a(x,y)+s1(6)*a(x,y+1)+s1(7)*a(x+1,y-1)+s1(8)*a(x+1,y)+s1(9)*a
(x+1,y+1)];

        b2(x,y)=s2(1)*a(x-1,y-1)+s2(2)*a(x-1,y)+s2(3)*a(x-1,y+1)+s2(4)*a(x,
y-1)+s2(5)*a(x,y)+s2(6)*a(x,y+1)+s2(7)*a(x+1,y-1)+s2(8)*a(x+1,y)+s2(9)*a(x
+1,y+1);
      end
End

figure(1)
subplot(2,2,1);imshow(uint8(a));title("Original
Image");subplot(2,2,2);imshow(uint8(a1));title("Prewitt Y Gradient by
302");subplot(2,2,3);imshow(uint8(a2));title("Prewitt X Gradient by 302");
outPrewitt=a1+a2;
subplot(2,2,4);imshow(uint8(outPrewitt));title("Prewitt Output by 302");
```

```
figure(2)
subplot(2,2,1);imshow(uint8(a));title("Original Image");
subplot(2,2,2);imshow(uint8(b1));title("Sobel Y Gradient by 302");
subplot(2,2,3);imshow(uint8(b2));title("Sobel X Gradient by 302");
outSobel=b1+b2;
subplot(2,2,4);imshow(uint8(outSobel));title("Sobel Output by 302");
```

Output for ***Prewitt***:



Output for ***Sobel***:

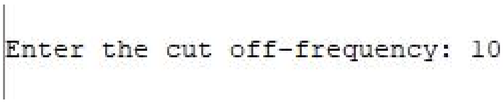


Practical 10 (GaussianLPF)

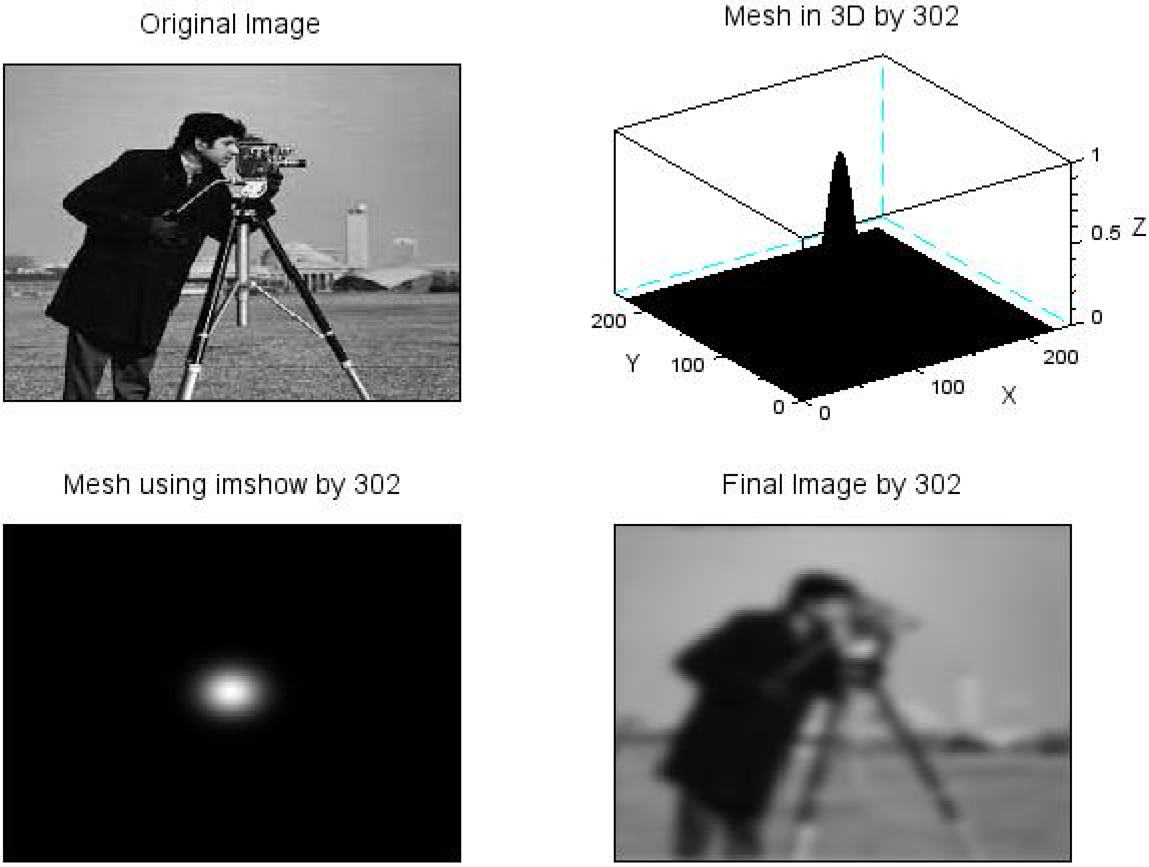
Code:

```
//Gaussian LPF
clc;
a=imread("C: \DIP-PRAC\cameraman.jpg");
//convert to gray image
a=rgb2gray(a);
a=double(a);
//get row, col in c(1) and c(2)
c=size(a);
N=c(1);
D0=input('Enter the cut off-frequency: ');
//creatiomn of Ideal-LPF
for u=1:1:c(1)
	for v=1:1:c(2)
		//calcuation of distance between (u,v) from centre
		Dx=((u-N/2)^2 + (v-N/2)^2)^0.5;
		D=Dx*Dx;
		H(u,v)=exp(-D / (2*D0*D0));
	end
end
//Find 2D DFT of image
vv=fft2(a);
vc=fftshift(vv);
//Scaler multiplication = convolution in spatial domain
x=vc.*H;
X=abs(ifft(x));
//plot graph
subplot(2,2,1); imshow(uint8(a));    title("Original Image");
subplot(2,2,2); mesh(H);                         title("Mesh in 3D by 302");
subplot(2,2,3); imshow(H);                    title("Mesh using imshow by 302");
subplot(2,2,4); imshow(uint8(X));    title("Final Image by 302");
```

Output:



```
Enter the cut off-frequency: 10
```



Original Image

Mesh in 3D by 302

Mesh using imshow by 302

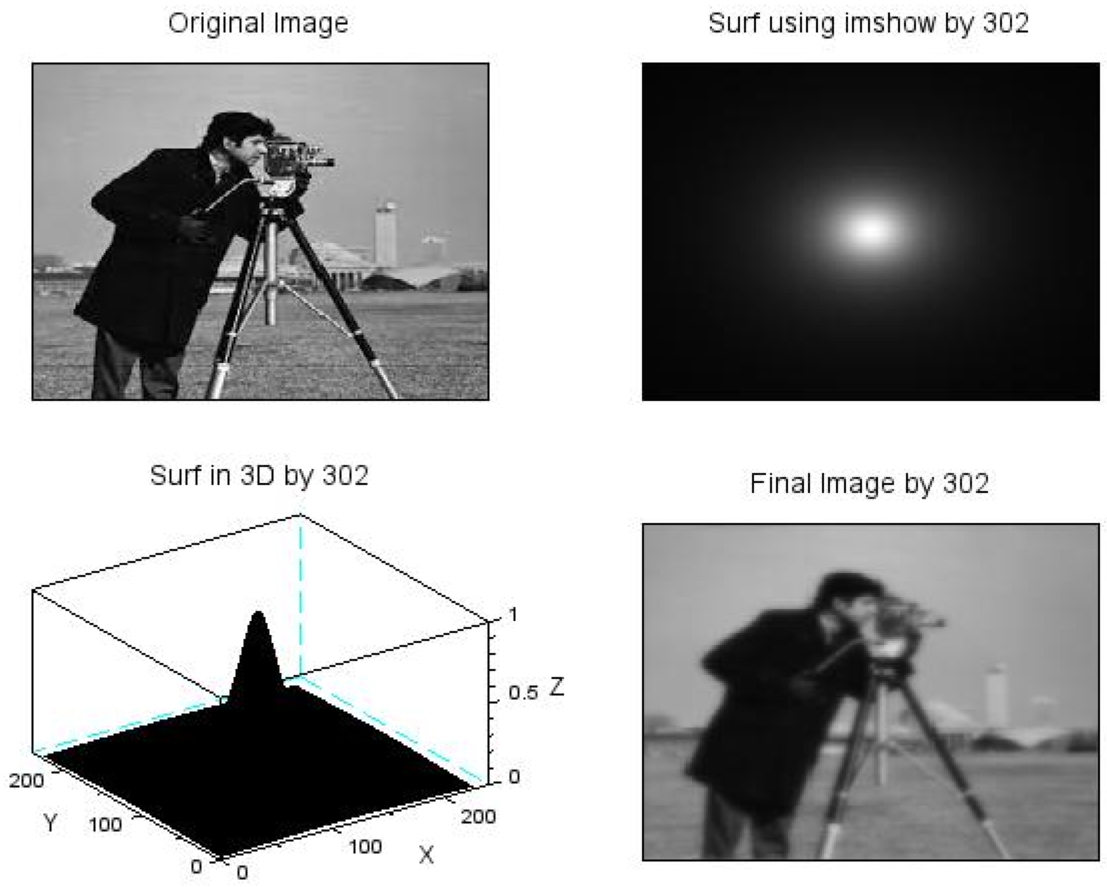
Final Image by 302

Practical 11 (Butterworth)

Code:

```
//Program to demonstrate butterworth low pass filter
//Read the input image
clc;
original1=imread("C:\Users\admin\Documents\DIP-PRAC\cameraman.jpg");
original=rgb2gray(original1);
original=double(original);
[m,n]=size(original);
//Set the cut off frequency
fc=20;
//Specifying the filter order
N=1;
//Finding the center of image
a=round(m/2);
b=round(n/2);
//Defining the filter kernel
//i and j are dimensions of input image
H=zeros(m,n);
for i=1:m
    for j=1:n
        d=((i-a)^2+(j-b)^2)^0.5;
        H(i,j)=1/(1+((d/fc)^(2*N)));

     end
end
//Input image to be shifted from spatial domain to frequency domain
original_freq=fftshift(fft2(original));
//H is filter function   ,multiplication in frequency domain is noting but convolution of image and apply the butterworth LPF
applpf=(original_freq).*H;
finalout=abs(ifft(applpf));
subplot(2,2,1); imshow(original1); title("Original Image");
subplot(2,2,2); imshow(H); title("Surf using imshow by 302");
subplot(2,2,3); surf(H); title("Surf in 3D by 302");
subplot(2,2,4); imshow(uint8(finalout)); title("Final Image by 302");
```

Output:

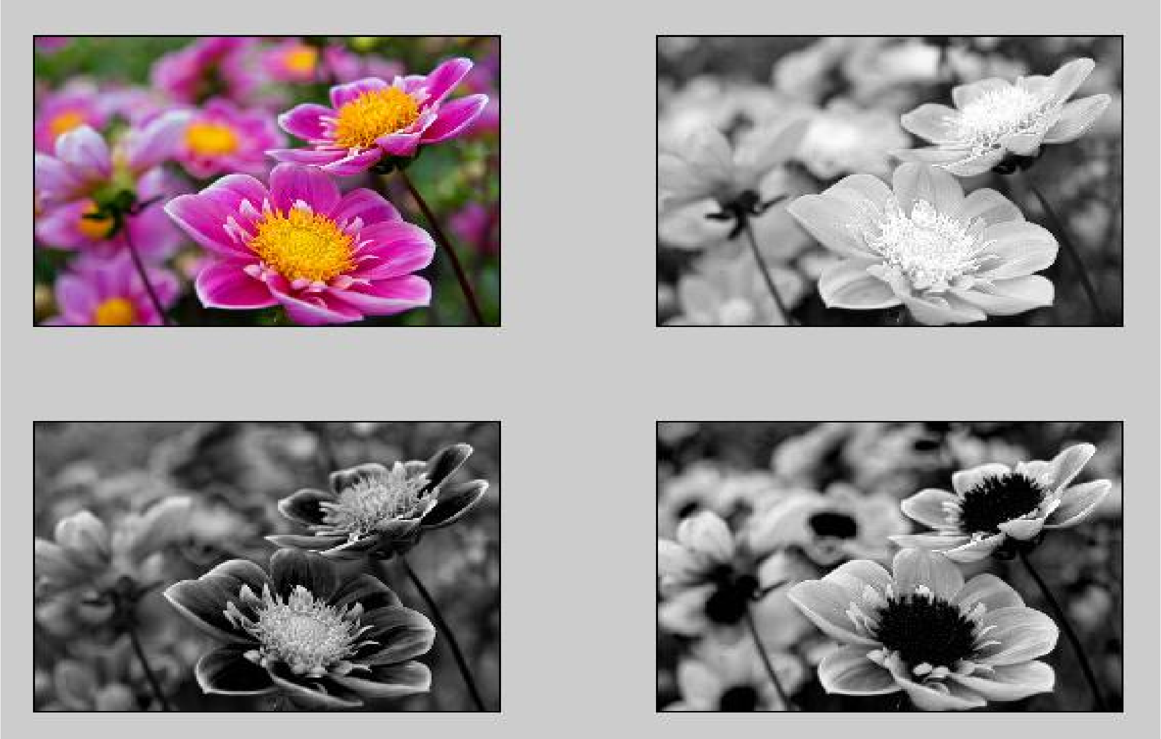

Original Image
Surf using imshow by 302
Surf in 3D by 302
1
0.5
Z
0
200
Y
100
0
0
100
X
200
Final Image by 302

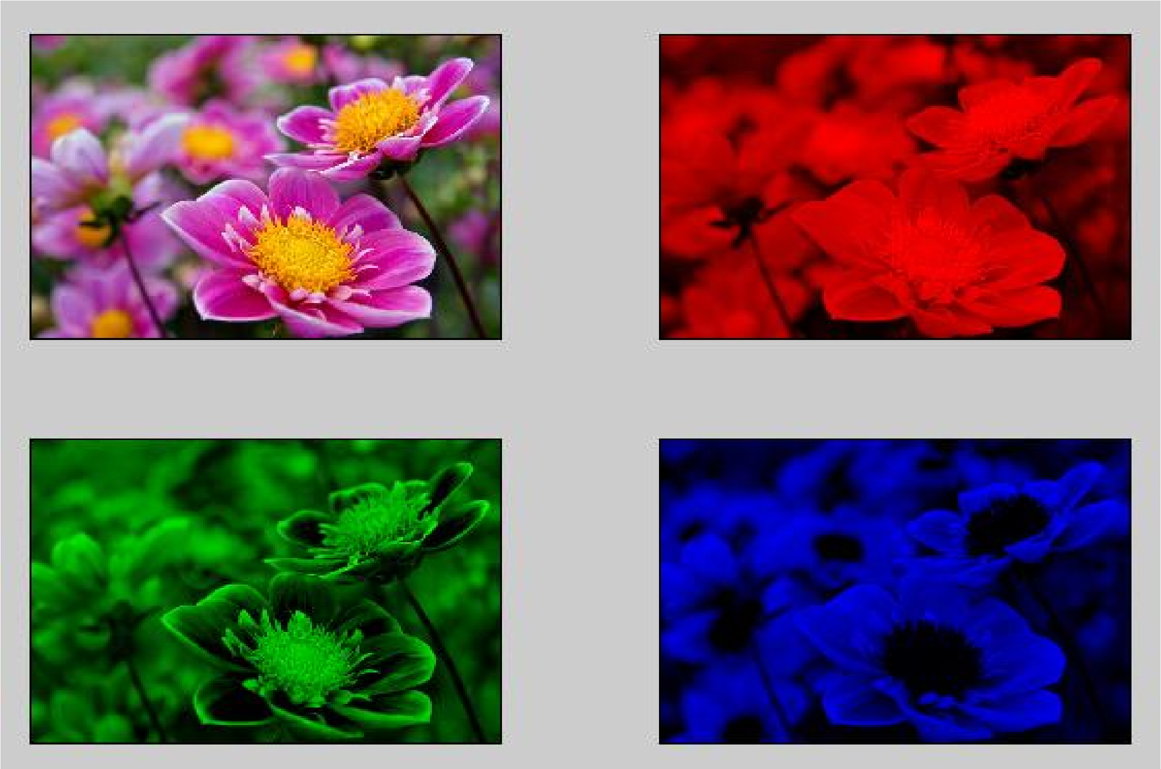
Practical 12 (Color Model)

Code:

```
clc;
a=imread("C:\Users\admin\Documents\DIP-PRAC\lavender.jpg");
a=double(a);
[row   col dim]=size(a);
red=a(:,:,1);//gives grey scale image of red plane
green=a(:,:,2);
blue=a(:,:,3);
plane=zeros(row, col);
RED=cat(3,red,plane,plane);//ensures that red is 24 bit
GREEN=cat(3,plane,green,plane);
BLUE=cat(3,plane,plane,blue);

figure(1);
subplot(2,2,1);
imshow(uint8(a));
subplot(2,2,2);
imshow(uint8(red));
subplot(2,2,3);
imshow(uint8(green));
subplot(2,2,4);
imshow(uint8(blue));

figure(2);
subplot(2,2,1);
imshow(uint8(a));
subplot(2,2,2);
imshow(uint8(RED));
subplot(2,2,3);
imshow(uint8(GREEN));
subplot(2,2,4);
imshow(uint8(BLUE));
```

Output:



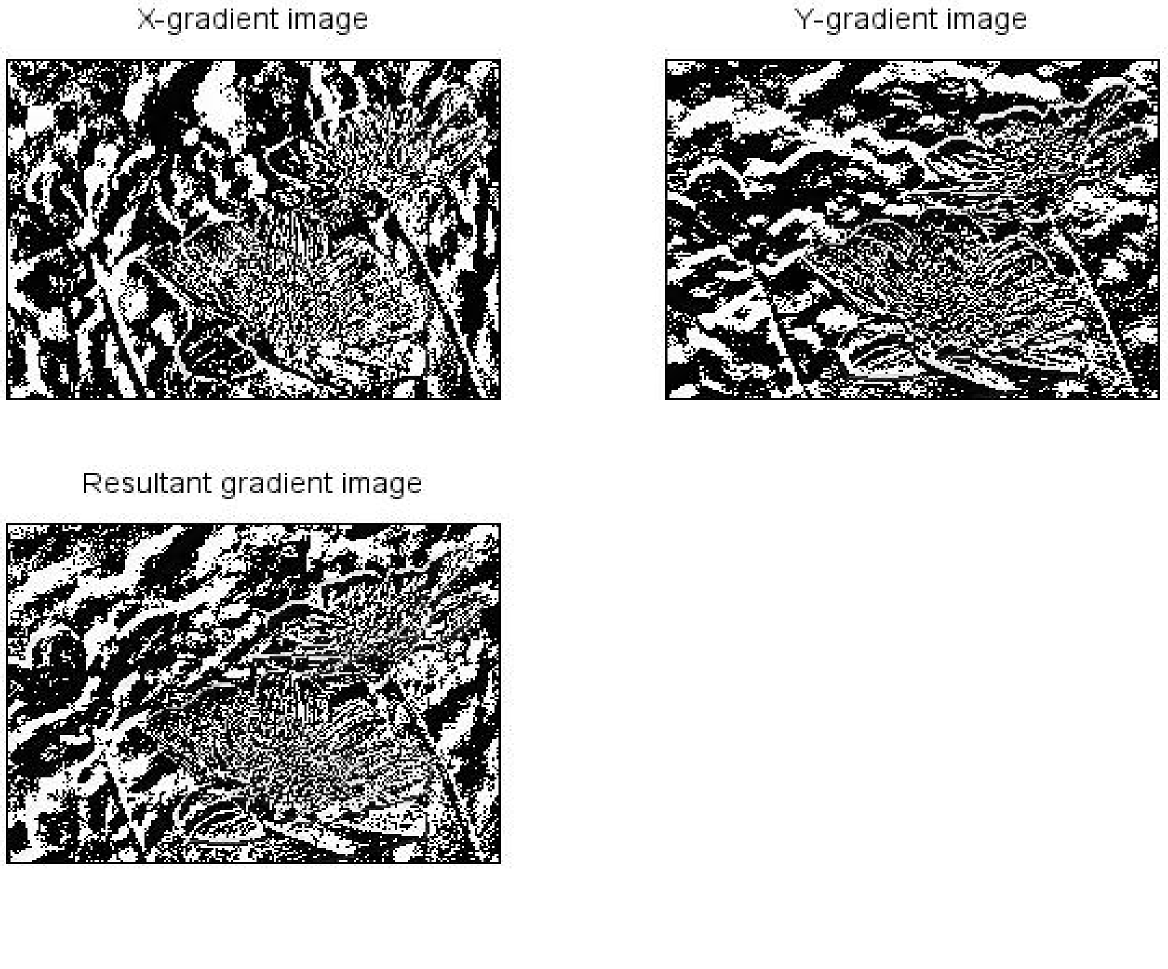


Practical 13 (Edge Detection)

Code for *Edge Detection using Ordinary operator*:

```
clc;
a=imread('f:\dip\lavender.jpg');
//convert to gray image
a=rgb2gray(a);    a=double(a);
//get row, col in c(1) and c(2)
[row col]=size(a);    //Ordinary operators
w1=[1 0; -1 0]; w2=[1 -1; 0 0];
for x=2:1:row-1
      for y=2:1:col-1
            a1(x,y)=w1(1)*a(x,y) + w1(2)*a(x,y+1) + w1(3)*a(x+1,y) +
w1(4)*a(x+1,y+1);
            a2(x,y)=w2(1)*a(x,y) + w2(2)*a(x,y+1) + w2(3)*a(x+1,y) +
w2(4)*a(x+1,y+1);
     end
end     a3=a1+a2;
subplot(2,2,1);imshow(uint8(a1))title('X-gradient image')
subplot(2,2,2);imshow(uint8(a2))title('Y-gradient image')
subplot(2,2,3);imshow(uint8(a3))title('Resultant gradient image')
```

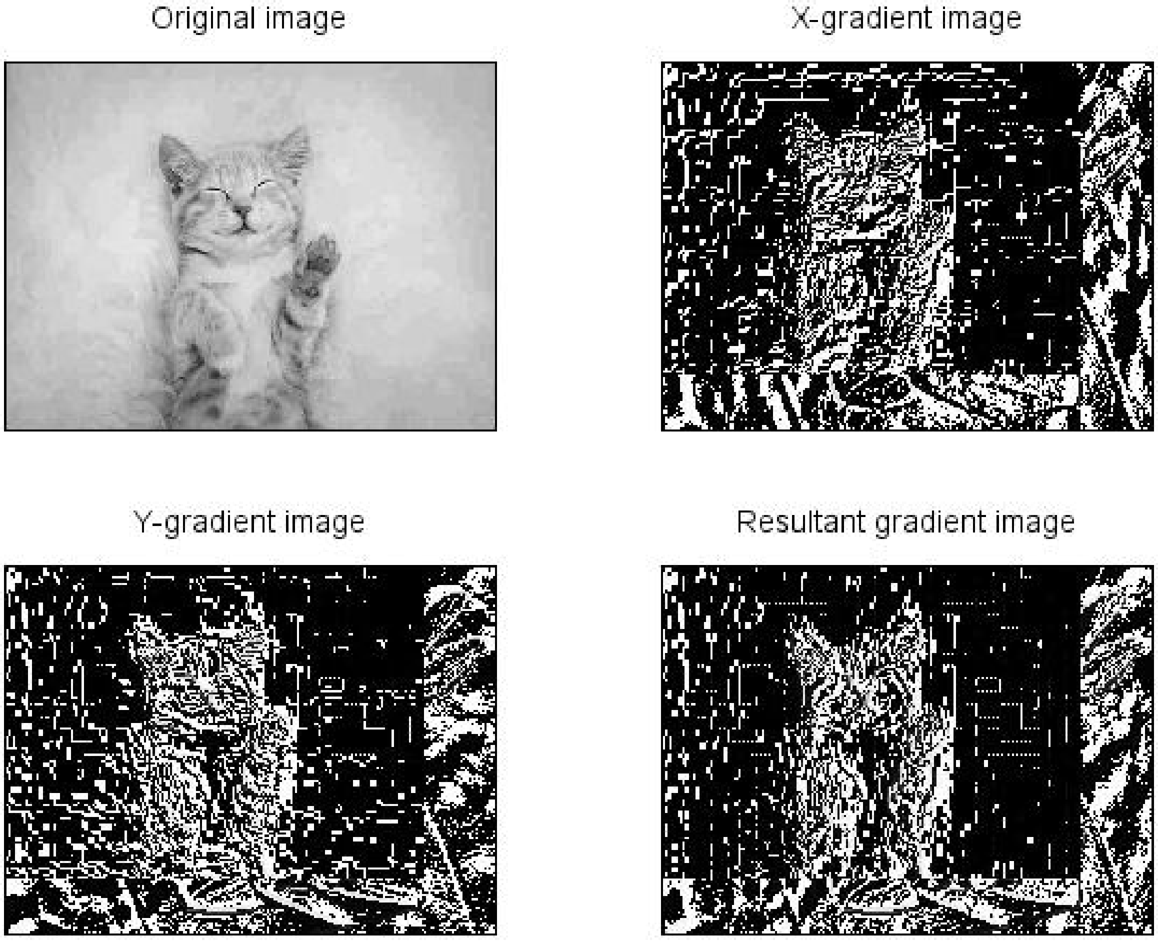
Output:



Code for *Edge Detection using Roberts operator*:

```
clc;//read image
a=imread('f:\dip\cameraman.jpg');
//convert to gray image
a=rgb2gray(a) ;a=double(a);
//get row, col in c(1) and c(2) [row col]=size(a);
//Roberts operators w1=[1 0; 0 -1]; w2=[0 1; -1 0];
for x=2:1:row-1
    for y=2:1:col-1
        a1(x,y)=w1(1)*a(x,y) + w1(2)*a(x,y+1) + w1(3)*a(x+1,y) +
w1(4)*a(x+1,y+1);
        a2(x,y)=w2(1)*a(x,y) + w2(2)*a(x,y+1) + w2(3)*a(x+1,y) +
w2(4)*a(x+1,y+1);
    endend
a3=a1+a2;
subplot(2,2,1);imshow(uint8(a))title('Original image')
subplot(2,2,2);imshow(uint8(a1))title('X-gradient image')
subplot(2,2,3);imshow(uint8(a2))title('Y-gradient image')
subplot(2,2,4);,imshow(uint8(a3))title('Resultant gradient image')
```

Output:



Original image

X-gradient image

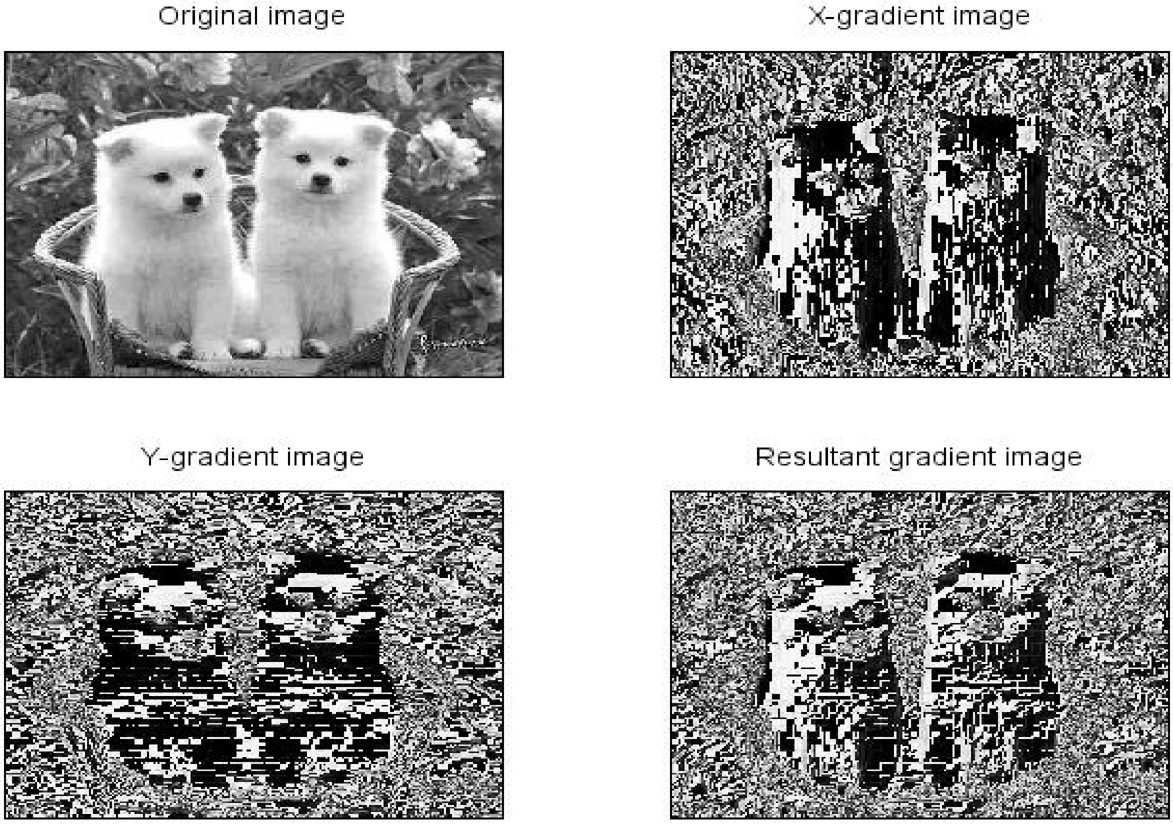
Y-gradient image

Resultant gradient image

Code for *Edge Detection using Prewitts operator*:

```
clc;
//read image    a=imread('f:\dip\cute2.jpg');
//convert to gray image    a=rgb2gray(a); a=double(a);
//get row, col in c(1) and c(2) [row col]=size(a);
//Prewitts operators
w1=[-1 -1 -1; 0 0 0; 1 1 1]; w2=[-1 0 1; -1 0 1; -1 0 1];
for x=2:1:row-1
      for y=2:1:col-1
         a1(x,y)=w1(1)*a(x-1,y-1) + w1(2)*a(x-1,y) + w1(3)*a(x-1,y+1) +
w1(4)*a(x,y-1)+w1(5)*a(x,y) + w1(6)*a(x,y+1) +w1(7)*a(x+1,y-1) +
w1(8)*a(x+1,y) + w1(9)*a(x+1,y+1);

         a2(x,y)=w2(1)*a(x-1,y-1) + w2(2)*a(x-1,y) + w2(3)*a(x-1,y+1) +
w2(4)*a(x,y-1)+w2(5)*a(x,y) + w2(6)*a(x,y+1) +w2(7)*a(x+1,y-1) +
w2(8)*a(x+1,y) + w2(9)*a(x+1,y+1);
         end
end      a3=a1+a2;
subplot(2,2,1);imshow(uint8(a))title('Original image')
subplot(2,2,2);imshow(uint8(a1))title('X-gradient image')
subplot(2,2,3);imshow(uint8(a2))title('Y-gradient image')
subplot(2,2,4);imshow(uint8(a3))title('Resultant gradient image')
```

Output:



Original image

X-gradient image

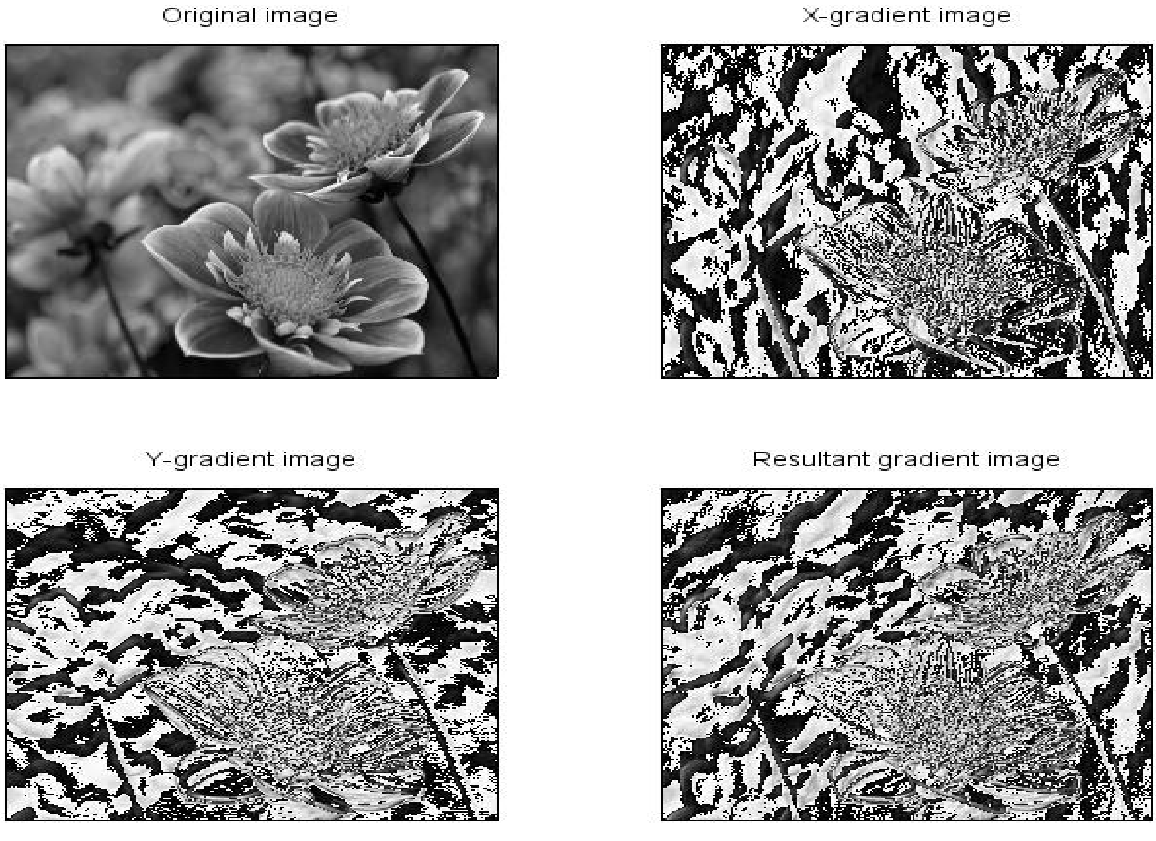
Y-gradient image

Resultant gradient image

Code for *Edge Detection using Sobel operator*:

```
clc;
//read image a=imread('f:\dip\lavender.jpg');
//convert to gray image a=rgb2gray(a); a=double(a);
//get row, col in c(1) and c(2) [row col]=size(a);
//Sobel operators
w1=[-1 -2 -1; 0 0 0; 1 2 1]; w2=[-1 0 1; -2 0 2; -1 0 1];
for x=2:1:row-1
      for y=2:1:col-1
         a1(x,y)=w1(1)*a(x-1,y-1) + w1(2)*a(x-1,y) + w1(3)*a(x-1,y+1) +
w1(4)*a(x,y-1)+w1(5)*a(x,y) + w1(6)*a(x,y+1) +w1(7)*a(x+1,y-1) +
w1(8)*a(x+1,y) + w1(9)*a(x+1,y+1);

         a2(x,y)=w2(1)*a(x-1,y-1) + w2(2)*a(x-1,y) + w2(3)*a(x-1,y+1) +
w2(4)*a(x,y-1)+w2(5)*a(x,y) + w2(6)*a(x,y+1) +w2(7)*a(x+1,y-1) +
w2(8)*a(x+1,y) + w2(9)*a(x+1,y+1);
         end
end   a3=a1+a2;
subplot(2,2,1);imshow(uint8(a))title('Original image')
subplot(2,2,2);imshow(uint8(a1))title('X-gradient image')
subplot(2,2,3);imshow(uint8(a2))title('Y-gradient image')
subplot(2,2,4);imshow(uint8(a3))title('Resultant gradient image')
```

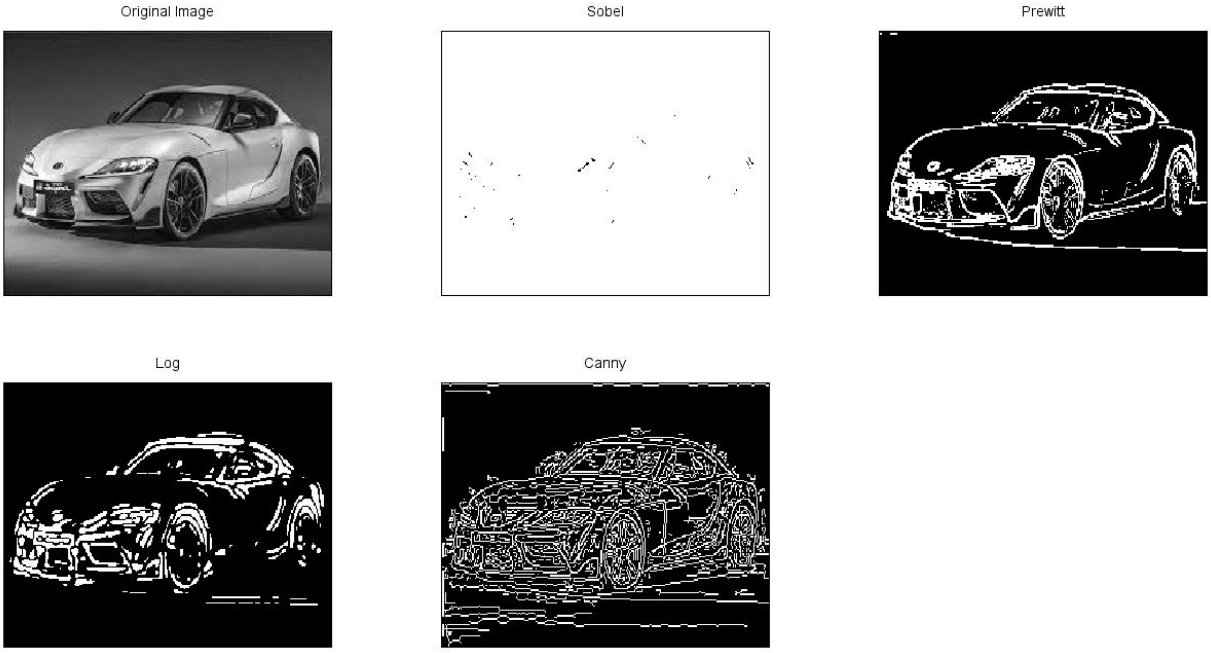
Output:



Code for *Edge Detection using Different Edge detectors*:

```
close ;
clc ;
a = imread ('d:\dip\supra.jpg');
a = rgb2gray(a);
c = edge (a, 'sobel');
d = edge (a, 'prewitt');
e = edge (a, 'log');
f = edge (a, 'canny');

subplot(2, 3, 1); imshow(a) title ('Original Image')
subplot(2, 3, 2); imshow(c) title ('Sobel')
subplot(2, 3, 3); imshow(d) title('Prewitt')
subplot(2, 3, 4); imshow(e) title ('Log')
subplot(2, 3, 5); imshow(f) title ('Canny')
```

Output:


Original Image
Sobel
Prewitt
Log
Canny