POORNIMA

Program - 11

Objective: Edge detection

Theory: The most powerful edge-detection technique that edge provide is the canny method. The canny method in that others used two different types of thresholding levels to detect strong and weak edges

Tool: MATLAB

Code

```
I = rgb2gray ("flower.jpg")
subplot (2,4,1)
imshow (I);

title ("Gray Scale Image");
J = edge (I, 'Sobel');
subplot (2,4,2);
imshow (J);
title ("Sobe");

K = edge (I, 'Prewitt');
subplot (2,4,3);
imshow (K);
title ("Prewitt");
```

Page No.....

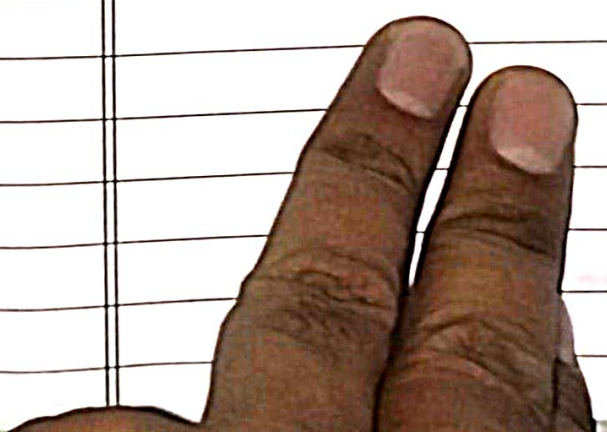
POORNIMA

```
L = edge(I, 'Roberts');
subplot (2,4,4),
imshow(L);
title ("Robert");

M = edge(I, 'log');
subplot (2,4,5);
imshow (M);
title ("log");


M = edge (I', 'zero');
subplot (2,4,6);
imshow (M);
title ("zero cross");

N = edge (I', 'canny');
subplot (2,4,7)
imshow (N);
title ("canny");
```



POORNIMA

Program = 11.1

Object → Edge detection

Tool → MATLAB

Code:

```
I = double (imread ('image.jpg'));

In = I;
mask = [1, 0, -1 ; 1, 0, -1 ; 1, 0, -1];

mask = fliped (mask);
mask = flipud (mask);

for i = 2 ; size (I, 1) - 1

    for j = 2 ; size (I, 2), - 1

        neighbours_matrix = mask. * In (i-1 ; i <1, j-1;
                                        j+1);

        avg_value = sum ( neighbours_matrix (:));

        I(i, j) = avg_value ;

    end
    end
figure, imshow (uint8(I)) ;
```

Page No.....

POORNIMA

Program - 11.2

Object → Edge detection

Tool → MATLab

Code

```
I = double ((imread ('image1.jpg')));

In = I;

mask = [1, 1, 1; 0, 0, 0; -1, -1, -1];
mask = flipud (mask);
mask = fliplr (mask);

for i = 2 : size (I, 1) - 1
    for j = 2 : size (I, 2) - 1

neighbour_matrix = mask * In (i-1 : i+1 : j-1 ; j+1);
avg value = sum (neighbour_matrix (:));

I (i, j) = avg_value;

end
end
    figure, imshow (uint8 (I));
```

Page No.....

POORNIMA

Program - 11.3

Object → Edge detection

Tool → MATLAB

```
I = double ((imread ('image1.jpg')));
In = I

mask = [0,-1,-1; 1,0,-1; 1,1,0];
mask = flipud (mask);
mask = fliplr (mask);

for i=2 : size (I,1)-1
    for j=2 : size (I,2)-1

  neighbours_matrix = mask .* In (i-1 : i+1, j-1, j+1);
  avg_value = sum (neighbour_matrix (:));
   I (i,j) = avg_value)
end
end

figure, imshow (uint 8(I));
```

PURNIMA

Program 11.4

Object: Edge detection

Tool: MATLAB

```
I = double ((imread ('image1.jpg')));
In = I;

mask = [1,1,1; 0,0,0; -1,-1,-1];
mask = flipud (mask);
mask = fliplr (mask);

for i=2 : size (I,1) -1
    for j=2 : size (I,2) -1

neighbour_matrix = mask. * In (i-1 : i+1, j-1 : j+1)

avg_value = sum (neighbour_matrix (:));

I (i,j) = avg_value;
end
end

figure, imshow (uint8 (I));
```

POORNIMA

Program 16.5

Object -> Edge detection

Tool -> MATLAB

```
I = double (imread ('image1.jpg'));
Im = 7;
mask1 = [1,0,-1; 1,0,-1; 1,0,-1];
mask2 = [1,1,1; 0,0,0; -1,-1,-1];
mask3 = [0,-1,-1; 1,0,-1; 1,1,0];
mask4 = [1,1,0; 1,0,-1; 0,-1,-1];

mask1 = flipud (mask1);
mask1 = fliplr (mask1);

mask2 = flipud (mask2);
mask2 = fliplr (mask2);

mask3 = flipud (mask3);
mask3 = fliplr (mask3)

mask4 = flipud (mask4);
mask4 = fliplr (mask4);

for i = 2 : size (I,1) - 1
    for j = 2 : size (I,2) - 1
```

Page No.....

POORNIMA

```
neighbour_matrix 1 = mask1 .* In(i-1,i+1,j-1,j+1);
 avg_value = Sum( neighbour_matrix 1(:));

neighbour_matrix * 2=mask2.In(i-1,i+1,j-1,j+1)
 avg_value = Sum (neighbour_matrix 2(:));

neighbour_martrix 3 = mask3.In(i-1, i+1, j-1, j+
 avg_value = sum( neighbour_matrix 3(:));

neighbour_matrix 4 = mask4 In(i-1, i+1, j-1, j+1)
 avg_value = Sum (neighbour_matrix 4(:));

I(i,j) = max( (avg_value 1. avg-value 2, avg-value 3,
                avg_value 4));

end
  end

figure, imshow (uint8(I));
```

POORNIMA

Program - 12

Objective →

Code

```
clc;
clear all;
close all;

im = imread ('                    ');

FT = fft2 (double (im));
FTS = fftshift (FT);
FTSG = log (FTS + 1);
figure;

imshow (abs (FTSG), []);
[m, n] = size (im);
t = 0: pi / 10: 2 * pi;

xc = (m + 150) / 2;
yc = (n - 150) / 2;
figure,
  subplot (2,2,1)
imshow (im);
```

```
for k=1:3
  r = 200;
  r1 = 50*k;

xcc= r*cos(t)+xc;
ycc = r* sin(t)+yc;
xcc1= r1*cos(t)+xc;
ycc1=r1* sin(t)+yc;

mask = poly2mask(double(xcc),double(ycc),m,n);

mask(mask1)=0;

FT2 = FTS;
FT2(mask)=0;

imshow(abs(output,[]);
end
```

POORNIMA

Program -> 13

Objective -> To understand & implement matlab program for Gaussian filtering.

Code ->

```
I = imread ('                    ')

I = double (I);
fil_Knl = ones (3.3)/9

[I_h, I_w] = size (I);

[ fil_h, fil_w] = size (fil_knl);

out_img = uint8 (zeros (I_h, I_w));

P1 = floor ( fil_h (2));
P2 = floor ( fil_w (2));

for i = 1 : I_h
        for j = 1 : I_w

    roi = pad_img (i : i + fil_h - 1, j : j + fil_w

    out_img (i, j) = sum (sum ( roi. * fil_knl))

end end
```

Page No.....

```
figure

subplot (2,1,1);
imshow (uint8(I));

title ('Original');

subplot (2,1,2);
imshow (out_img);
title ('filtered image');
```

Program → 14.1

Objective → To understand & implement Matlab program for dilated image or eroded image.

```
I = imread ("flower.jpg");
subplot (2,3,1);
imshow (I);
title (" Original Image");

se = strel (" line ", 7,7);
dilate = imdilate (I,se);
subplot (2,3,2);
imshow (dilate);
title (" Dilate Image");

erode = imerode (I, se);
subplot (2,3,3);
imshow (erode);
title (" eroded image ");

open = imopen ( I, se);
subplot (2,3,4);
imshow (open);
title ('openet Image');
```

```
close = imclose (I,se);
subplot (2,3,5);
imshow (close);
title ('Closed Image');
```

Program → 14.2

Objective → To understand and implement matlab program for gaussian Noise or Average fillter.

Code

```
clc;
close all;
A = imread ("image.jpeg");
subplot (3,3,1);
imshow (A);
title ('Original image");

a = strel ('reactangle', [10,20]);
di = imdilate (A,a);
imshow (3,3,2)
imshow (di);
title ('dilated image');
```

POORNIMA

```
e = imnoise(di,'gaussian');
subplot (3,3,3);
imshow (e);
title ('Gaussian noise')

a1 = fspecial ('average' 2);
avgfilt = imfilter(e,a1,
subplot (3,3,4)
imshow (avg filt),
title ('average filter );

m = mean2(di);
```