Ex No : 1 Extraction of color components from RGB color image

Program :

```
I = imread('lenna.png');
r = size(I, 1);
c = size(I, 2);

R = zeros(r, c, 3);
G = zeros(r, c, 3);
B = zeros(r, c, 3);

R(:, :, 1) = I(:, :, 1);
G(:, :, 2) = I(:, :, 2);
B(:, :, 3) = I(:, :, 3);

figure, imshow(uint8(R));
figure, imshow(uint8(G));
figure, imshow(uint8(B));

rgbImage = imread('flower.png');

redChannel = rgbImage(:,:,1); % Red channel
greenChannel = rgbImage(:,:,2); % Green channel
blueChannel = rgbImage(:,:,3); % Blue channel

allBlack = zeros(size(rgbImage, 1), size(rgbImage, 2), 'uint8');

just_red = cat(3, redChannel, allBlack, allBlack);
just_green = cat(3, allBlack, greenChannel, allBlack);
just_blue = cat(3, allBlack, allBlack, blueChannel);

recombinedRGBImage = cat(3, redChannel, greenChannel, blueChannel);

subplot(3, 3, 2);
imshow(rgbImage);
title('Original RGB Image')
```

```
subplot(3, 3, 4);
imshow(just_red);
title('Red Channel in Red')

subplot(3, 3, 5);
imshow(just_green)
title('Green Channel in Green')

subplot(3, 3, 6);
imshow(just_blue);
title('Blue Channel in Blue')

subplot(3, 3, 8);
imshow(recombinedRGBImage);
title('Recombined to Form Original RGB Image Again')
```

Ex No : 2 Image enhancement using pixel operation

Program :

A. Linear Transformation

```
clc;
clear all;
close all;
pic=imread('grape.jpg');
subplot(1,2,1)
imshow(pic)
[x,y,z]=size(pic);
if(z==1);
else
  pic=rgb2gray(pic);
end
max_gray=max(max(pic));
max_gray=im2double(max_gray);
pic=im2double(pic);
for i=1:x
  for j=1:y
```

```
    pic_negative(i,j)=max_gray-pic(i,j);
  end
end
subplot(1,2,2)
imshow(pic_negative)
```

B. Logarithmic Transformation

```
clc; clear all; close all;
f=imread('grape.jpg');
g=rgb2gray(f);
c=input('Enter the constant value, c = ');
[M,N]=size(g);
    for x = 1:M
      for y = 1:N
        m=double(g(x,y));
        z(x,y)=c.*log10(1+m);
      end
    end
imshow(f), figure, imshow(z);
```

C. Power Law Transformation

```
clear all
close all
RGB=imread('grape.jpg');
I=rgb2gray(RGB);
I=im2double(I);
[m n] = size(I);
c = 2;
g =[0.5 0.7 0.9 1 2 3 4 5 6];
for r=1:length(g)
for p = 1 : m
  for q = 1 : n
    I3(p, q) = c * I(p, q).^ g(r);
  end
end
```

```
figure, imshow(I3);
title('Power-law transformation');
xlabel('Gamma='),xlabel(g(r));
end
```

Ex No : 3   Image enhancement using histogram equalization.

Program :

```
close all
I = imread('pout.tif');
imshow(I)
figure, imhist(I)
I2 = histeq(I);
figure, imshow(I2)
figure, imhist(I2)
imwrite (I2, 'pout2.png');
imfinfo('pout2.png')
```

Ex No : 4   Filtering an image using averaging low pass filter in spatial domain and median filter.

Program :

AVERAGING LOW PASS FILTER

```
clc
clear all;
close all;

i=imread('grape.jpg');
a = rgb2gray(i);

b=imnoise(a,'salt & pepper',0.1);
c=imnoise(a,'gaussian');
d=imnoise(a,'speckle');

h1=1/9*ones(3,3);
h2=1/25*ones(5,5);
```

```
b1=conv2(b,h1,'same');
b2=conv2(b,h2,'same');
c1=conv2(c,h1,'same');
c2=conv2(c,h2,'same');
d1=conv2(d,h1,'same');
d2=conv2(d,h2,'same');

figure;
subplot(2,2,1);
imshow(a);
title('original image');

subplot(2,2,2);
imshow(b);
title('Salt & Pepper');

subplot(2,2,3);
imshow(uint8(b1));
title('3X3 Averaging filter');

subplot(2,2,4);
imshow(uint8(b2));
title('5X5 Averaging filter');

figure;
subplot(2,2,1);
imshow(a);
title('original image');

subplot(2,2,2);
imshow(c);
title('Gaussian');

subplot(2,2,3);
imshow(uint8(c1));
title('3X3 Averaging filter');
```

```
subplot(2,2,4);
imshow(uint8(c2));
title('5X5 Averaging filter');

figure;
subplot(2,2,1);
imshow(a);
title('original image');

subplot(2,2,2);
imshow(d);
title('Speckle');

subplot(2,2,3);
imshow(uint8(d1));
title('3X3 Averaging filter');

subplot(2,2,4);
imshow(uint8(d2));
title('5X5 Averaging filter');
```

MEDIAN FILTER

```
clc;
clear all;
close all;
a = imread('grape.jpg');
I = rgb2gray(a);
J = imnoise(I,'salt & pepper',0.02);
K = medfilt2(J);

figure;
subplot(1,3,1);
imshow(I);
title('Original image');
subplot(1,3,2)
imshow(J);
title('Noisy image');
```

```
        subplot(1,3,3);
        imshow(K);
        title('Median filtered image');
```

Ex No : 5 Sharpen an image using 2-D laplacian high pass filter in spatial domain.

Program :

```
i = imread("grape.jpg");
subplot(2,2,1);

a =imshow("grape.jpg");
title("Original image");

a= rgb2gray(i);
Lap=[0 1 0; 1 -4 1; 0 1 0];

a1 = conv2(a,Lap,'same');
a2 = uint8(a1);

subplot(2,2,2);
imshow(abs(a-a2),[])
title("Laplacian filtered image");

lap=[-1 -1 -1; -1 8 -1; -1 -1 -1];

a3=conv2(a,lap,'same');
a4=uint8(a3);
subplot(2,2,3);

imshow(abs(a+a4),[])
title("High boost filtered image");
```

Ex No : 6 Smoothing of an image using low pass filter and high pass filter in frequency domain (Butterworth LPF and HPF)

Program :

% MATLAB Code | Butterworth Low Pass Filter

```
clc;
clear all;
close all;
a = imread("grape.jpg");
input_image = rgb2gray(a);
[M, N] = size(input_image);
FT_img = fft2(double(input_image));
n = 2;
D0 = 20;

u = 0:(M-1);
v = 0:(N-1);
idx = find(u > M/2);
u(idx) = u(idx) - M;
idy = find(v > N/2);
v(idy) = v(idy) - N;

[V, U] = meshgrid(v, u);
D = sqrt(U.^2 + V.^2);
H = 1./(1 + (D./D0).^(2*n))
G = double(H).*double(FT_img);

output_image = real(ifft2(double(G)));

subplot(2, 1, 1), imshow(input_image),
title("Original Image");
subplot(2, 1, 2), imshow(output_image, [ ]);
title("Butterworth lowpass filtered Image");
```

```
% MATLAB Code | Butterworth High Pass Filter

        a = imread("grape.jpg");
        input_image= rgb2gray(a);

        [M, N] = size(input_image);
        FT_img = fft2(double(input_image));

        n = 2;

        D0 = 10;
        u = 0:(M-1);
        v = 0:(N-1);
        idx = find(u > M/2);
        u(idx) = u(idx) - M;
        idy = find(v > N/2);
        v(idy) = v(idy) - N;

        [V, U] = meshgrid(v, u);
        D = sqrt(U.^2 + V.^2);
        H = 1./(1 + (D0./D).^(2*n));
        G = H.*FT_img;

        output_image = real(ifft2(double(G)));

        subplot(2, 1, 1), imshow(input_image),
        title("Original Image");
        subplot(2, 1, 2), imshow(output_image, [ ]);
        title("Butterworth highpass filtered Image");
```

Ex No : 7 Program for morphological image operations-erosion, dilation, opening & closing

Program :

```
% Morphological image operations - Erosion
         originalBW = imread('cameraman.tif');
```

```
    se = strel('line',5,40);
    erodedBW = imerode(originalBW,se);
    figure, imshow(originalBW);
    title("Original Image");
    figure, imshow(erodedBW)
    title("Eroded Image");

% Morphological image operations - Dilation
    originalBW = imread('text.png');
    se = strel('line',9,50);
    dilatedBW = imdilate(originalBW,se);
    figure, imshow(originalBW),
    title("Original Image");
    figure, imshow(dilatedBW)
    title("Dilated Image");

% Morphological image operations - Opening
    original = imread('cameraman.tif');
    se = strel('disk',3);
    afterOpening = imopen(original,se);
    figure, imshow(original),
    title("Original Image");
    figure, imshow(afterOpening,[])
    title("Image after opening");

% Morphological image operations - Closing
    originalBW = imread('circles.png');
    figure, imshow(originalBW);
    title("Original Image");
    se = strel('disk',6);
    closeBW = imclose(originalBW,se);
    figure, imshow(closeBW);
    title("Image after closing");
```

Ex No : 9 Program for image compression using Huffman coding

Program :

```
symbols = 1:6;
p = [.5 .125 .125 .125 .0625 .0625];
dict = huffmandict(symbols,p);
inputSig = randsrc(100,1,[symbols;p]);
code = huffmanenco(inputSig,dict);
sig = huffmandeco(code,dict);
isequal(inputSig,sig)
binarySig = de2bi(inputSig);
seqLen = numel(binarySig)
binaryComp = de2bi(code);
encodedLen = numel(binaryComp)
inputSig = {'a2',44,'a3',55,'a1'}
dict = {'a1',0; 'a2',[1,0]; 'a3',[1,1,0]; 44,[1,1,1,0]; 55,[1,1,1,1]}
enco = huffmanenco(inputSig,dict);
sig = huffmandeco(enco,dict)
isequal(inputSig,sig)
```

Exp: 10. Pattern Classification Methods.

PROGRAM:

```
[x,t] = iris_dataset;
net = patternnet(10);
net = train(net,x,t);
view(net)
y = net(x);
perf = perform(net,t,y);
classes = vec2ind(y);
```