LAB-1

Q. Implement a program to convert image to grayscale.

Program

```
clc
close all
clear all
A=imread("lab1.png");
subplot(1,2,1);
imshow(A);
title("Color Image of Car")
B=rgb2gray(A);
#figure
subplot(1,2,2);
imshow(B)
title("Grayscale Image of Car");
```

LAB 2

Q. Implement a program to perform the negative linear transformation.

Program

```
clear all;
close all;
clc;
pkg load image;
img = imread("lab2.png");
figure;
imshow(img);
title("Original Image");
grayscale_img = rgb2gray(img);
figure;
imshow(grayscale_img);
title("Grayscale Image");
imwrite(grayscale_img, "lab2_grayscale.png");
negative_img = 255 - grayscale_img;
figure;
imshow(negative_img);
title("Negative Image");
imwrite(negative_img, "lab2_negative.png");
```

LAB 3

Q. Implement a program to perform the Log transformation.

```
clear all;
close all;
clc;
a = imread('lab3.png');
subplot(2, 2, 1);
imshow(a);
title('Original Image');
b = im2double(a);
s1 = 1 * log(1 + b);
s1 = uint8(s1 * 255 / max(s1(:))); % Proper scaling
subplot(2, 2, 2); imshow(s1);
title('c=1');
s2 = 10 * log(1 + b);
s2 = uint8(s2 * 255 / max(s2(:))); % Proper scaling
subplot(2, 2, 3);
imshow(s2);
title('c=12');
s3 = 200 * log(1 + b);
s3 = uint8(s3 * 255 / max(s3(:))); % Proper scaling
subplot(2, 2, 4);
imshow(s3);
title('c=201');
```

LAB-4

Q. Implement a program to perform the Gamma Transformation.

Program

```
clear all;
close all;
clc;
pkg load image;
a = imread('lab4.png');
subplot(2, 2, 1);
imshow(a);
title('Original Image');
b = im2double(a);
a1 = 1; % Enter the value for a1 here
gamma1 = 0.5; % Enter the value for gamma1 here
a2 = 2; % Enter the value for a2 here
gamma2 = 1; % Enter the value for gamma2 here
a3 = 3; % Enter the value for a3 here
gamma3 = 1.5; % Enter the value for gamma3 here
% Apply gamma correction for the first set of values
s1 = (a1 * (b .^ gamma1)) * 256;
s1 = uint8(s1);
subplot(2, 2, 2);
imshow(s1);
title('Image 1');
```

```
s2 = (a2 * (b .^ gamma2)) * 256;
s2 = uint8(s2);
subplot(2, 2, 3);
imshow(s2);
title('Image 2');

% Apply gamma correction for the third set of values
s3 = (a3 * (b .^ gamma3)) * 256;
s3 = uint8(s3);
subplot(2, 2, 4);
imshow(s3);
title('Image 3');
```

LAB 5

Q. Implement a program to perform Bit plane Slicing.

```
clear all;
close all;
clc;
pkg load image;
i=imread("lab6.jpeg");
b0=double(bitget(i,1));
b1=double(bitget(i,2));
b2=double(bitget(i,3));
b3=double(bitget(i,4));
b4=double(bitget(i,5));
b5=double(bitget(i,6));
b6=double(bitget(i,7));
b7=double(bitget(i,8));
subplot(3,3,1);imshow(i);title('Original Image');
subplot(3,3,2);imshow(b0);title('BIT PLANE 0');
subplot(3,3,3);imshow(b1);title('BIT PLANE 1');
subplot(3,3,4);imshow(b2);title('BIT PLANE 2');
subplot(3,3,5);imshow(b3);title('BIT PLANE 3');
subplot(3,3,6);imshow(b4);title('BIT PLANE 4');
subplot(3,3,7);imshow(b5);title('BIT PLANE 5');
subplot(3,3,8);imshow(b6);title('BIT PLANE 6');
subplot(3,3,9);imshow(b7);title('BIT PLANE 7');
```

Lab-6

Q. Implement a program to perform Histogram Equalization.

Program

```
clear all; % clear ALL VARIABLES
close all;%close all figures
clc; %clear command window
%import image package
pkg load image;
% read image for Image Enhancement
I = imread('lab6.png');
subplot(4,2,1); imshow(I);title('Original Image');
g = rgb2gray(I);
subplot(4,2,5);imshow(g);title('Gray Image');
J= imadjust(g,[0.3,0.7],[]);
subplot(4,2,3);imshow(J);title('Enhanced Image');
D=imadjust(I,[0.2 0.3 0; 0.6 0.7 1],[]);
subplot(4,2,4);imshow(D);title('Enhanced Image 2');
%Histogram and Histogram Equalization
subplot(4,2,7); imhist(g);title('Histogram of Gray Image');
m=histeq(g);
subplot(4,2,6);imshow(m);title('Equalized Image');
subplot(4,2,8);imhist(m);title('Histogram of Equalized Image');
```

LAB -7

Q. Implement a program to perform Convolution Filtering.

Program

```
clear all;
close all;
clc;
pkg load image;
 i=imread('lab7.png');
 i=i(:,:,1);subplot(2,2,1);imshow(i);title('Original Image');
 a=[0.001 0.001 0.001;0.001 0.001 0.001;0.001 0.001 0.001];
 r=conv2(a,i);
 subplot(2,2,2);imshow(r);title('filtered image');
 b=[0.005 0.005 0.005;0.005 0.005 0.005;0.005 0.005 0.005];
 r1=conv2(b,i);
 subplot(2,2,3);imshow(r1);title('filtered image2');
```

LAB -8

Q. Implement a program to perform Median Filter.

Program

```
clear all;
close all;
clc;
pkg load image;
i=imread('lab8.png');
k=rgb2gray(i);
j=imnoise(k,'salt & pepper',0.05);
f=medfilt2(j,[3,3]);
f1=medfilt2(j,[10,10]);
subplot(3,2,1); imshow(i);title('Original Image');
subplot(3,2,2); imshow(k);title('Gray Image');
subplot(3,2,3); imshow(j);title('Noise added Image');
subplot(3,2,4); imshow(f);title('3*3 Image');
subplot(3,2,5); imshow(f1);title('10*10 Image');
```

LAB- 9

Q. Implement a program to perform Fast Fourier Transformation.

Program

```
clear all;%clear all variables
close all;%close all figures
clc; %clear command window
%import image package
pkg load image;
I=im2double(imread('lab9.png'));
f1=fft(I);
f2=fftshift(f1);

subplot(2,2,1); imshow(abs(f1)); title('Frequency Spectrum');
subplot(2,2,2); imshow(abs(f2)); title('Centered Spectrum');
f3=log(1+abs(f2));
subplot(2,2,3); imshow(f3); title('log(1+abs(f2))');
I=fft2(f1);
I1=real(I);
subplot(2,2,4); imshow(I1); title('2-D FFT');
```

LAB-10

Q. Implement a program to apply Gradient Filter.

Program

```
clear all;%clear all variables
close all;%close all figures
clc; %clear command window
pkg load image;
input_image = imread('lab10.png');
if size(input_image, 3) == 3
  input_image = rgb2gray(input_image); % Convert to grayscale if the image is RGB
endif
sobel_x = [-1 0 1; -2 0 2; -1 0 1];
sobel_y = [-1 -2 -1; 0 0 0; 1 2 1];
gradient_x = imfilter(double(input_image), sobel_x);
gradient_y = imfilter(double(input_image), sobel_y);
gradient_magnitude = sqrt(gradient_x.^2 + gradient_y.^2);
figure, imshow(input_image), title('Original Image');
figure, imshow(gradient_x, []), title('Gradient in X direction');
figure, imshow(gradient_y, []), title('Gradient in Y direction');
figure, imshow(gradient_magnitude, []), title('Gradient Magnitude');
```