

DEPARTMENT OF COMPUTER ENGINEERING & APPLICATIONS

DIP LAB RECORD

**Name :** Baibhavi Pandey

**University Roll No. :** 2115000281

**Subject Name :** Digital Image Processing Lab

**Subject Code :** BCSE 0101

**Course :** B.Tech.

**Year :** III

**Semester :** V

**INDEX**

|  |  |  |
| --- | --- | --- |
| **Exp.**  **No.** | **Program** |  |
| 1 | Introduction with the MATLAB Command. |  |
| 2 | Read the image and display it. |  |
| 3 | Image manipulation and pseudocolor composition. |  |
| 4 | Bit plane Slicing |  |
| 5 | Perform Log Transformation, Negative Transformation, Power Log Transformation |  |
| 6 | Perform Histogram Equalization on image. |  |
| 7 | Find Four, Diagonal and Eight Neighbours of image. |  |
| 8 | Spatial Filtering – Smoothing: Avg, Min, Max, Median, Wt. Avg |  |
| 9 | Sharpening filter- laplacian |  |
| 10 | Perform erosion and dilation on image |  |
| 11 | Perform Closing(Dilation Followed by Erosion) on Image. |  |
| 12 | PerformPoint Detection and Line Detection on image. |  |
| 13 | Canny Edge Detection |  |
|  |  |  |

**Experiment -1**

**Objective:-** Introduction with the MATLAB Command.

**Commands:-**

1. clc - Clear Command window

2. clearall - clear the workspace

3. date - displays current date

4. linspace - creates regularly spaced rectory

5. Mat – returns largest element

6. Min - returns smallest element

7. size – computes array size

8. ones – creates an array of ones

9. zeros – creates an array of zeros

10. subplot – creates plot in subwindows.

**Experiment -2**

**Objective:-** Read the image and display it.

**Implement:-**

image = imread("cameraman.tif");

imshow(image)

**Output:-**



**Experiment -3**

**Object:** Image manipulation and pseudocolor composition.

clear all;

close all;

clc;

img3=imread("matlab2.jpeg");

subplot(2,3,1);

imshow(img3);

img31=rgb2gray(img3);

subplot(2,3,2);

imshow(img31);

img32=imresize(img31,0.2);

subplot(2,3,3);

imshow(img32);

img33=imresize(img31,2);

subplot(2,2,3);

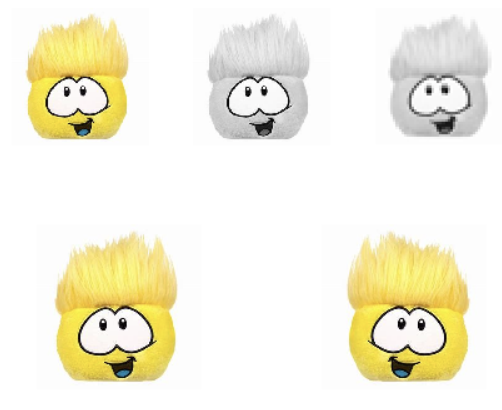
imshow(img3);

subplot(2,2,4);

img34=fliplr(img3);

imshow(img34);

**Output:-**



**Experiment -4**

**Objective:-** Bit plane Slicing on Image.

**Implement:-**

close all;

clc;

a=imread("mat.jpg");

a=rgb2gray(a);

b1=bitget(a,1);

subplot(1,8,1);imshow(b1); title('original');

b2=bitget(a,2);

subplot(1,8,2);imshow(b2); title('Bit Plane1');

b3=bitget(a,3);

subplot(1,8,3);imshow(b3); title('Bit Plane2');

b4=bitget(a,4);

subplot(1,8,4);imshow(b4); title('Bit Plane3');

b5=bitget(a,5);

subplot(1,8,5);imshow(b5); title('Bit Plane4');

b6=bitget(a,6);

subplot(1,8,6);imshow(b6); title('Bit Plane5');

b7=bitget(a,7);

subplot(1,8,7);imshow(b7); title('Bit Plane6');

b8=bitget(a,8);

subplot(1,8,8);imshow(b8); title('Bit Plane7');

c=(b6\*32)+(b7\*64)+b8\*128;

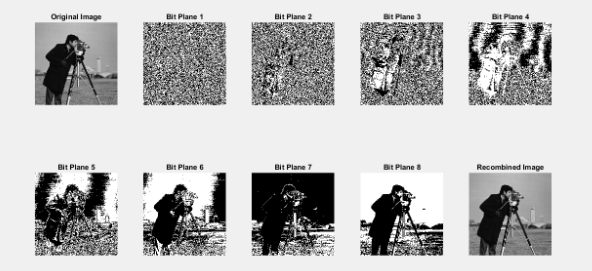
c=uint8(1.138\*c);

subplot(2,2,1);

imshow(c);

title('Bit Plane8');

**Output:-**



**Experiment -5.1**

**Objective:-** Perform Log Transformation on Image.

**Implement:-**

image = imread('cameraman.tif');

subplot(1,2,1),

imshow(image),

title("original image");

r = double(image);

c = 40;

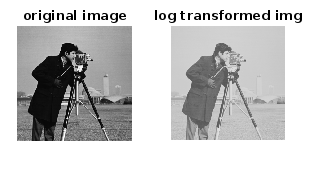
s = c \* log(1 + r);

image1 = uint8(s);

subplot(1,2,2), imshow(image1),

title("log transformed img");

**Output:-**



**Experiment -5.2**

**Objective:-** Perform Negative Transformation on Image.

**Implement:-**

image = imread('cameraman.tif');

subplot(1,2,1),

imshow(image),

title("original image");

image1=image;

for row=1:size(image,1)

for col=1:size(image,2)

image1(row,col)=255-image(row,col);

end

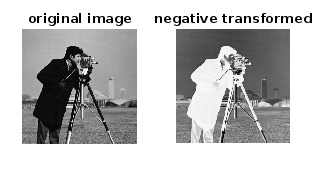
end

subplot(1,2,2),

imshow(image1),

title("negative transformed");

**Output:-**



**Experiment -5.3**

**Objective:-** Perform Power Law Transformation on Image.

**Implement:-**

image = imread('cameraman.tif');

subplot(1,2,1),

imshow(image),

title("original image");

r = double(image);

c = 100;

g = 0.5;

s = c \* log(1 + r)^g;

t = 255/(c\*log(256)^g);

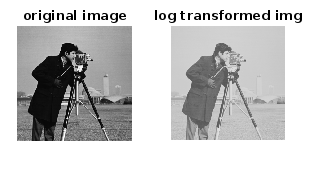
image1 = uint8(t\*s);

subplot(1,2,2),

imshow(image1),

title("power log transformed");

**Output:-**



**Experiment -6**

**Objective:-** Perform Histogram Equalization on image.

**Implement:-**

image=imread('cameraman.tif');

[n,m]=size(image);

for i=1:256

t(i,1)=i-1;

t(i,2)=0;

end

for i=1:n

for j=1:m

pix=image(i,j)+1;

t(pix,2)=t(pix,2)+1;

end

end

for i=1:256

t(i,3)=t(i,2)/(256\*256);

end

t(1,4)=t(1,3)

for i=2:256

t(i,4)=t(i-1,4)+t(i,3);

end

for i=1:256

t(i,5)=t(i,4)\*255;

end

for i=1:256

t(i,6)=round(t(i,5));

end

for i=1:256s(i,1)=i-1;

s(i,2)=0;

end

for i=1:256

pix=t(i,6)+1;

s(pix,2)=s(pix,2)+t(i,2);

end

image2=image;

for i=1:n

for j=1:m

image2(i,j)=t(image(i,j)+1,6);

end

end

subplot(3,2,1),

imshow(image),

title('Original');

subplot(3,2,2),

imshow(image2),

title('Manual Transform');

subplot(3,2,3),

bar(t(:,1),t(:,2)),

title('Histogram');

subplot(3,2,4),

bar(s(:,1),s(:,2)),

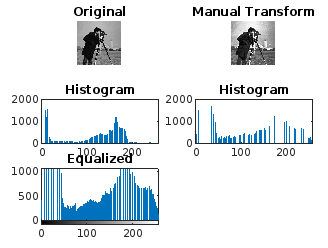
title('Histogram');

subplot(3,2,5),

imhist(image2),

title("Equalized");

**Output:-**

****

**Experiment -7**

**Objective:-** Find Four, Diagonal and Eight Neighbours of image.

**Implement:-**

image = [1, 2, 3, 4; 5, 6, 7, 8; 9, 10, 11, 12; 13, 14, 15, 16];

x = input('X-axis: ');

y = input('Y-axis: ');

[n, m] = size(image);

fprintf('n: %d, m: %d\n', n, m);

FN = NaN(1, 4);

EN = NaN(1, 8);

DN = NaN(1, 4);

if x > 1

FN(1) = image(x - 1, y);

end

if y > 1

FN(2) = image(x, y - 1);

end

if x < n

FN(3) = image(x + 1, y);

end

if y < m

FN(4) = image(x, y + 1);

end

if x > 1 && y > 1

DN(1) = image(x - 1, y - 1);

end

if x < n && y < m

DN(2) = image(x + 1, y + 1);

end

if x < n && y > 1

DN(3) = image(x + 1, y - 1);

end

if x > 1 && y < m

DN(4) = image(x - 1, y + 1);

end

fprintf('Four Neighbors: %s\n', mat2str(FN));

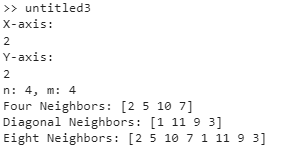
fprintf('Diagonal Neighbors: %s\n', mat2str(DN));

EN(1:4) = FN(1:4);

EN(5:8) = DN(1:4);

fprintf('Eight Neighbors: %s\n', mat2str(EN));

**Output:-**



**Experiment -8.1**

**Objective:-** Spatial Filtering: Smoothening - Max.

**Implement:-**

close all;

clear all;

%max

i1 = imread("bean.jpg");

filter = ones(3);

i1 = cast(i1,'double');

i2 = i1;

for i = 2:255

for j = 2:255

s = max(i1(i-1:i+1, j-1:j+1));

i2(i,j) = max(s);

end

end

subplot(1,2,1);

imshow(uint8(i1));

title('Original');

subplot(1,2,2);

imshow(uint8(i2));

title('Max');

**Output:-**



**Experiment -8.2**

**Objective:-** Spatial Filtering: Smoothening - Min.

**Implement:-**

close all;

clear all;

i1 = imread("bean.jpg");

filter = ones(3);

i1 = cast(i1,'double');

i2 = i1;

for i = 2:255

for j = 2:255

s = min(i1(i-1:i+1, j-1:j+1));

i2(i,j) = min(s);

end

end

subplot(1,2,1);

imshow(uint8(i1));

title('Original');

subplot(1,2,2);

imshow(uint8(i2));

title('Min Operation');

**Output:-**



**Experiment -8.3**

**Objective:-** Spatial Filtering: Smoothening - Median.

**Implement:-**

close all;

clear all;

i1=imread("bean.jpg");

filter=[1,2,1;2,3,2;1,2,1];

i1 = cast(i1, 'double');

i2=i1;

s=0;

for i=2:255

for j=2:255

s=i1(i-1:i+1,j-1:j+1).\*filter(1:3,1:3);

s1=sum(s(:));

i2(i,j)=s1/15;

end

end

subplot(1,2,1);

imshow(uint8(i1));

title('Original');

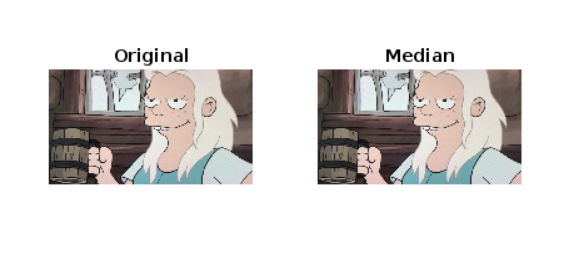
i2 = uint8(i2);

subplot(1,2,2);

imshow(uint8(i2));

title('Median');

**Output:-**



**Experiment -8.4**

**Objective:-** Spatial Filtering: Smoothening - Average.

**Implement:-**

close all;

clear all;

i1=imread("bean.jpg");

filter=ones(3);

i1 = cast(i1, 'double');

i2=i1;

s=0;

for i=2:255

for j=2:255

s=i1(i-1:i+1,j-1:j+1);

s1=sum(s(:));

i2(i,j)=s1/9;

end

end

subplot(1,2,1);

imshow(uint8(i1));

title("Original");

subplot(1,2,2);

imshow(uint8(i2));

title("After Average");

**Output:-**



**Experiment -8.5**

**Objective:-** Spatial Filtering: Smoothening – Weighted Average.

**Implement:-**

close all;

clear all;

i1=imread("bean.jpg");

filter=ones(3);

i1 = cast(i1, 'double');

i2=i1;

s=0;

for i=2:255

for j=2:255

s=i1(i-1:i+1,j-1:j+1);

s1=sum(s(:));

i2(i,j)=s1/16;

end

end

subplot(1,2,1);

imshow(uint8(i1));

title("Original");

subplot(1,2,2);

imshow(uint8(i2));

title("After Wt. Average");

**Output:-**



**Experiment -9**

**Objective:-** Sharpening filter -laplacian

**Implement:-**

clear all;

close all;

clc;

originalImage = imread('matlab2.jpeg');

grayImage = rgb2gray(originalImage);

laplacianFilter = fspecial('laplacian');

laplacianResult = imfilter(grayImage, laplacianFilter);

subplot(1, 2, 1)

imshow(grayImage);

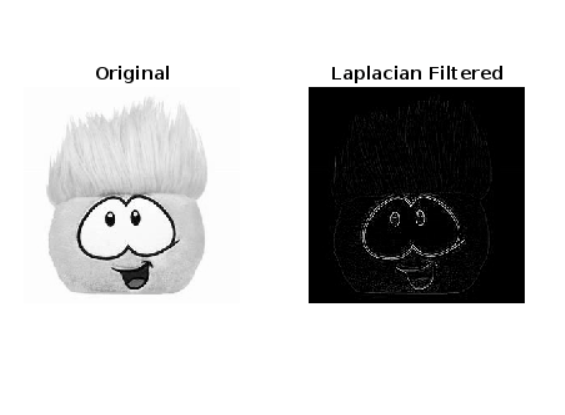
title('Original');

subplot(1, 2, 2),;

imshow(laplacianResult);

title('Laplacian Filtered');

**Output:-**



**Experiment -10.1**

**Objective:-** Perform Dilation on Image.

**Implement:-**

clear all;

close all;

clc;

% erosion %

img=imread("matlab2.jpeg");

img=rgb2gray(img);

subplot(2,4,1);

imshow(img);

title('original')

binaryImage = im2bw(img);

subplot(2,4,2);

imshow(binaryImage);

title('binary');

se = strel('disk', 3);

subplot(2,4,3);

imshow(double(getnhood(se)), 'InitialMagnification', 'fit');

title('Struc Ele');

erodedImage = imerode(binaryImage, se);

subplot(2,4,4);

imshow(erodedImage);

title('Eroded Image');

% dilation %

subplot(2,4,5);

imshow(img);

title('original')

binaryImage = im2bw(img);

subplot(2,4,6);

imshow(binaryImage);

title('binary');

se = strel('disk', 3);

subplot(2,4,7);

imshow(uint8(getnhood(se)), 'InitialMagnification', 'fit');

title('Struc ele');

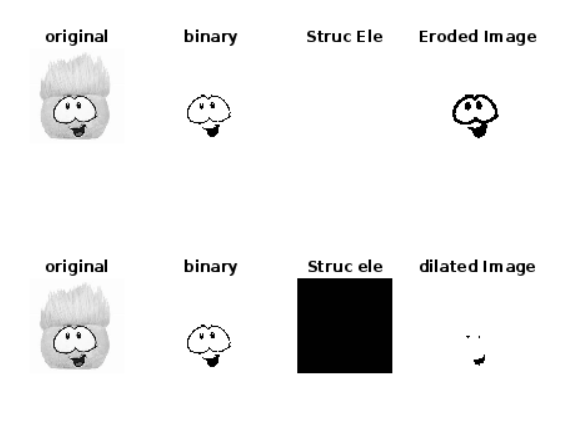
dilatedImage = imdilate(binaryImage, se);

subplot(2,4,8);

imshow(dilatedImage);

title('dilated Image');

**Output:-**



**Experiment -11**

**Objective:-** Perform Closing(Dilation Followed by Erosion) on Image.

**Implement:-**

originalImage = imread('matlab2.jpeg');

grayImage = rgb2gray(originalImage);

se = strel('disk', 5);

closingResult = imclose(grayImage, se);

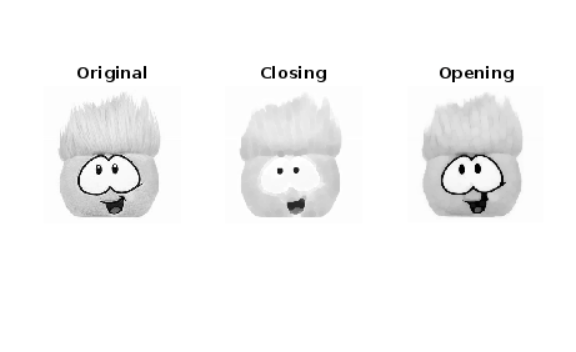
openingResult = imopen(grayImage, se);

subplot(1, 3, 1), imshow(grayImage), title('Original');

subplot(1, 3, 2), imshow(closingResult), title('Closing');

subplot(1, 3, 3), imshow(openingResult), title('Opening');

**Output:-**



**Experiment -12.1**

**Objective:-** PerformPoint Detection on image.

**Implement:-**

clear all;

close all;

clc;

%point detection

originalImage = imread('matlab2.jpeg');

grayImage = rgb2gray(originalImage);

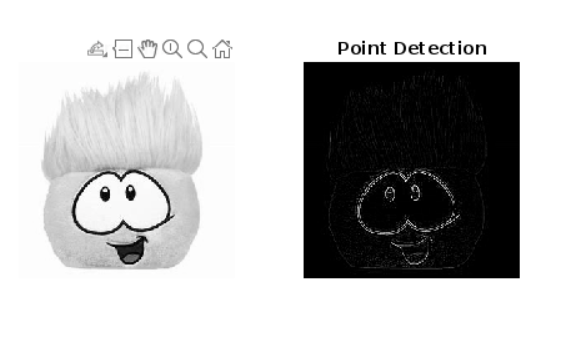
pointDetectedImage = imfilter(grayImage, ...

fspecial('laplacian')

subplot(1, 2, 1), imshow(grayImage), title('Original');

subplot(1, 2, 2), imshow(pointDetectedImage), title('Point Detection');

**Output:-**



**Experiment -12.2**

**Objective:-** Line Detection

**Implement:-**

clear all;

close all;

clc;

originalImage = imread('matlab2.jpeg');

grayImage = rgb2gray(originalImage);

BW = edge(grayImage, 'canny');

[H, T, R] = hough(BW);

P = houghpeaks(H, 5);

lines = houghlines(BW, T, R, P);

figure;

subplot(1, 2, 1), imshow(grayImage), title('Original');

subplot(1, 2, 2), imshow(grayImage), title('Line Detection');

hold on;

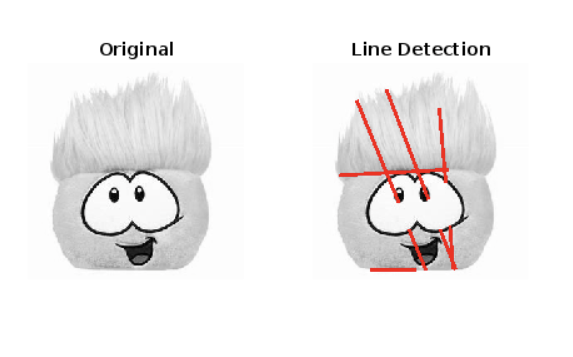
for k = 1:length(lines)

xy = [lines(k).point1; lines(k).point2];

plot(xy(:,1), xy(:,2), 'LineWidth', 2, 'Color', 'red');

end

hold off;

**Output:-**

**Experiment -13**

**Objective:- Canny Edge Detection**

**Implement:-**

clear all

close all

clc

originalImage = imread('amazon.jpg');

grayImage = rgb2gray(originalImage);

edgeImage = edge(grayImage, 'canny');

figure;

subplot(1, 2, 1);

imshow(grayImage);

title('Original Image');

subplot(1, 2, 2);

imshow(edgeImage);

title('Canny Edge Detection');

set(gcf, 'Position',get(0, 'Screensize'));

**Output:-**

