**ADIP WARM UP ASSIGNMENT**

**ROHAN N KALPAVRUKSHA PES1201802830**

**ROSHAN N KALPAVRUKSHA PES1201802834**

**SURIT PES1201800589**

1. **PROBLEM STATEMENT**

Perform addition of two images using Matlab.

**THEORY**

This operator takes as input two identically sized images and produces as output a third image of the same size as the first two, in which each pixel value is the sum of the values of the corresponding pixel from each of the two input images.

**CODE**

clc

clear all

close all

I = imread('rice.png');

J = imread('cameraman.tif');

for i=1:1:256

for j=1:256

output(i,j)=(I(i,j)+J(i,j));

end

end

imshow(output);

**OUTPUT**



**OBSERVATION**

The two images have been overlapped.

1. **PROBLEM STATEMENT**

Spot the difference between the two images shown below (hint: use image subtraction)

** **

**Figure 1 Figure 2**

**THEORY**

Image subtraction or pixel subtraction is a process whereby the digital numeric value of one pixel or whole image is subtracted from another image. This is primarily done for one of two reasons – levelling uneven sections of an image such as half an image having a shadow on it, or detecting changes between two images

**CODE**

clc

clear all

close all

warning off;

x=imread('Figure1.jpg');

y=imread('Figure2.jpg');

g=size(x);

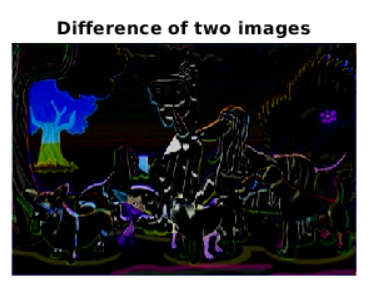
y=imresize(y,[g(1),g(2)]); % returns image B that has the number of rows and columns specified by the two-

figure;

imshow(x-y);

title('Difference of two images');

**OUTPUT**

****

**OBSERVATION**

The difference between the two images is spotted with a colour patch.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

**ADIP ASSIGNMENT - 1**

**ROHAN N KALPAVRUKSHA PES1201802830**

**ROSHAN N KALPAVRUKSHA PES1201802834**

**SURIT PES1201800589**

1. **PROBLEM STATEMENT**

Prune the skeletonised image given below to remove the unwanted branches on Matlab**.**

****

**THEORY**

Pruning is used as a complement to the [skeleton](https://en.wikipedia.org/wiki/Topological_skeleton) and thinning algorithms to remove unwanted parasitic components.

**CODE**

clc

clear all

close all

ima=imread('skel.jpg');

BW = im2bw(ima,0.5);

skel= bwmorph(BW,'skeleton',Inf);

B = bwmorph(skel, 'branchpoints');

E = bwmorph(skel, 'endpoints');

[y,x] = find(E);

B\_loc = find(B);

Dmask = false(size(skel));

for k = 1:numel(x)

D = bwdistgeodesic(skel,x(k),y(k));

distanceToBranchPt = min(D(B\_loc));

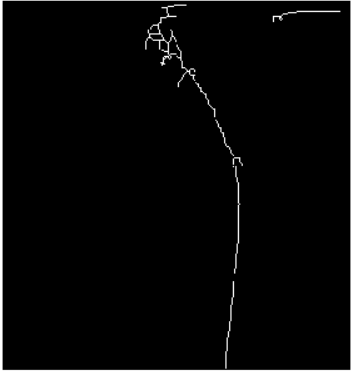
Dmask(D < distanceToBranchPt) =true;

end

skelD = skel - Dmask;

imshow(skelD);

**OUTPUT**

****

**OBSERVATION**

The primary root of the image is obtained by removing the smaller unwanted branches

1. **PROBLEM STATEMENT**

Perform segmentation using the Watershed algorithm in Maltab

****

**THEORY**

The watershed transforms find "catchment basins" or "watershed ridge lines" in an image by treating it as a surface where light pixels represent high elevations and dark pixels represent low elevations. The watershed transform can be used to segment contiguous regions of interest into distinct objects.

**CODE**

clc

clear all

close all

I = imread('coins.jpg');

I1 =imtophat(I, strel('disk',50)); %tophat filtering

level=graythresh(I1);

BW=im2bw(I1,level);

C=~BW;

D=-bwdist(C);

D(C)=-Inf;

L=watershed(D);

Wi=label2rgb(L,'hot','w');

im=I;

im(L==0)=0;

imshow(im);

**OUTPUT**

****

**OBSERVATION**

A green colour boundary line has been imposed on the original image surrounding the individual objects therefore segmenting the image by watershed algorithm.

1. **PROBLEM STATEMENT**

Perform image compression using the JPEG algorithm.

****

**THEORY/ALGORITHM**

With Input Image perform,

1. A colour transform
2. A 2D discrete cosine transform on 8x8 blocks
3. A quantization (filtering) stage
4. Huffman encoding.
5. Finally, a compressed image is returned in the .jpg file format.

**CODE**

clc

clear all

close all

x = imread('vege.jpg');

F=dct(double(x));

G=dct(F);

ff=idct(G);

fff=idct(ff);

[r,c] = size(x);

DF=zeros(r,c);

DFF= DF; % full factorial design

IDF=DF;

IDFF=DF;

depth=4;

N=1;

for i=1:N:r

for j=1:N:c

f=x(i,j);

df=dct2(f);

DF(i:i+N-1,j:j+N-1)=df;

dff=idct2(df);

DFF(i:i+N-1,j:j+N-1)=dff;

df(N:-1:depth+1,:)=0;

df(:,N:-1:depth+1)=0;

IDF(i:i+N-1,j:j+N-1)=df;

dff=(idct2(df));

IDFF(i:i+N-1,j:j+N-1)=dff;

end

end

A=DFF/255;

imwrite(A,'abc1.tif');

B=IDFF/255;

imshow(B);

imwrite(B,'abc2.tif');

**OUTPUT**





**OBSERVATION**

Size of the original image was 52KB it has reduced to 36KB after jpeg compression

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*