**ASSIGNMENT - 2**

1.

**1.1 Morphological Operations:**

Erosion: Erosion of a binary image with a structuring element gives another binary image with edge pixels eroded away. Erosion operation simply shrinks the image . operation includes comparing the elements of the image with the structuring element ,if they are foreground pixels they are left. But if any of the image pixels is background pixels, then image is set to background pixel. Here foreground and background pixels are obtained by initial thresholding operation. Erosion preserves convex shape and tend to round concave shapes.

Dilation: Dilation is an opposite operation to erosion. In dilation the structuring elements are compared to the background pixels. If any neighbouring pixels are found to be in foreground then the pixels value is assigned foreground pixel value. Dilation preserves concave shapes and the tend to round convex shapes.

Opening: Opening is erosion followed by dilation. It tends to smoothen the image.

Closing : closing is dilation followed by erosion. Unlike erosion , dilation fills gaps, eliminates small holes and fuses narrow breaks.

**1.2 Procedure(dilation):**

* First the RGB image is converted to a grey scale image
* Later a 0.5 thresholding is applied.
* After thresholding, dilation operation is performed by a disk shaped structured element of radius -1.
* Now the original image (grey image) is subtracted from the above step result image.
* Resultant image contains 2 grey levels with edges.

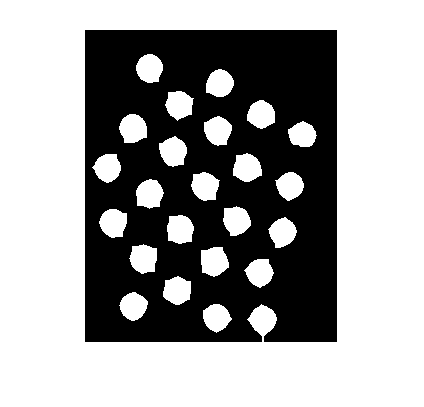
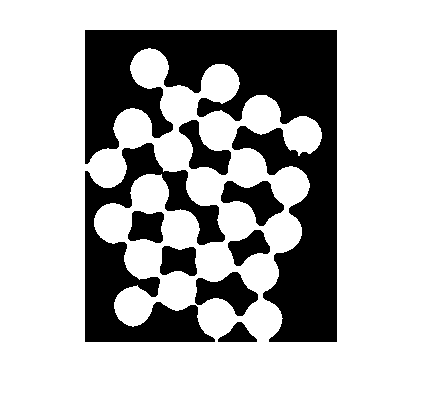
**Procedure(Erosion):**

* First RGB image is read and converted to a 2d grey image.
* After the complement of the image is obtained.
* Now a structuring element of dish shape and radius of 5 is created.
* Now the complemented image is eroded using this disk – an image is small joining lines between coins is formed.
* Now second erosion is performed again with the same structuring element to remove those joints.

**1.3 Results(dilation):**



**Results(erosion):**



**1.4 Code(dilation):**

image=imread('C:\Users\Aditya\Desktop\matlab\morphological\_proc\lena.tiff');

%%thresholding

image\_grey=im2bw(image,0.5);

%% structuring element

struc\_el=strel('disk',3);

image\_res3=imdilate(image\_grey,struc\_el);

struc\_el=strel('disk',1);

image\_res5=imdilate(image\_grey,struc\_el);

%%subtraction

imshow(image\_res5-image\_grey);

**Code(erosion):**

image=imread('C:\Users\Aditya\Desktop\matlab\morphological\_proc\circles.jpg');

%% grey image

image\_bw=im2bw(image);

%% complemented image

image\_ng=imcomplement(image\_bw);

%% structuring element

strel1=strel('disk',5);

%% first erosion

image1=imerode(image\_ng,strel1);

%% second erosion

image2=imerode(image1,strel1);

**2.**

**2.1 Hough Transform:**

Hough transform is used to find edges of image which have some discontinuity and also for detecting of lines and predefined shapes. However Generalized Hough Transform is used to detect shapes of any arbitrary size. This algorithm transforms the image in image space to hough space .

Image space is identified by x and y co-ordinates and hough space is identified by slope and intercept.

A set of points in image space corresponds to a line in hough space, using this many lines or curves are formed for the image in image space. If we find the local maixma of points in hough space, these points corresponds to lines in images space.

**2.2 Procedure:**

* First the RGB image is converted to gray scale image.
* Then a Canny edge filter is applied on the above image.
* The size of the image in rows and columns is found out and an accumulator array is formed by them.
* accumulator array is initialized wit zero’s initially, we here use a 3D array (x,y,r), because the circle contains radius
* then for each pixel in row and column, and for radius ranging from 70 to 100 identify the edge pixels(==1) and transform them to a polar domain. Forming a,b from x,y,r,theta .
* theta ranges from 0 to 360
* as we are doing a mathematical operation to obtain a,b the size of accumulator array can be greater than x,y. so , it is checked for this using some conditional loops.
* For positive values , accumulator array is incremented by one (one single vote is given)
* After obtaining the accumulator array (took a long time in matlab)
* We have to find the local maxima , the point where circle centre is likely to be located .
* After finding the maxima , the coordinates of that maxima are the center point of the circle.

**2.3 Results:**

(502,90); (460,186) ;(358,69)

**2.4 Code:**

image=imread('C:\Users\Aditya\Desktop\matlab\morphological\_proc\test\_samp.jpg');

inputimage = rgb2gray(image);

edge\_image = edge(inputimage, 'canny');

r1=100;

[x, y] = size(edge\_image);

acc = zeros(x,y,r1);

for xi=1:x

for yj=1:y

if edge\_image(xi, yj) == 1 %% this is an edge point

for r=70:100

for t=0:360

a=x-r\*cos(t\*pi/180);

b=y-r\*sin(t\*pi/180);

a=round(a);

b=round(b);

if a>x

acc=zeros(a,y,r1);

end

if b>y

acc=zeros(x,b,r1);

end

if a>=x && b>=y

acc=zeros(a,b,r1);

end

if a>0 && b>0

acc(a,b,r)=acc(a,b,r)+1;

end

end

end

end

end

end

[M,I]=max(acc(:));

[x,y,z]=ind2sub(size(acc),I);

temp=sort( acc, 'descend' )