**east west university**

**Final Lab (Assignment)**

**Department:** **Computer Science and Engineering**

**Course Title:** Digital Image Processing

**Course Code:** CSE438

**Section No:** 02

**Submitted To**:

Dr. Ahmed Wasif Reza

Professor, Department of CSE

Additional Director, Institutional Quality Assurance

Cell (IQAC)

**Submitted By**:

Name: Apurba Roy Ajay

ID: 2018-3-60-063

**1. Apply Gaussian noise to Figure 1, and then use the following to restore the image:**

**i. Geometric Mean filter**

**ii. Harmonic Mean filter**

**iii. Contra-harmonic Mean filter**

**Ans:**

**Code-**

img = imread('438image1.png'); % Read the image

img = rgb2gray(img); % Convert it to grayscale

GI=imnoise(img,'gaussian');

Gg=im2double(GI);

Kr=3;

Kc=3;

Ord = 2;

%Geometric Mean filter

GM=exp(imfilter(log(Gg),ones(Kr,Kc),'replicate')).^(1/(Kr\*Kc));

%Harmonic Mean filter

HM=(Kr\*Kc)./imfilter(1./(Gg+eps),ones(Kr,Kc),'replicate');

%Contra-harmonic Mean filter

CM = imfilter(Gg.^(Ord+1), ones(Kr,Kc), 'replicate') ./ (imfilter(Gg.^(Ord), ones(Kr,Kc), 'replicate') + eps);

%Display

subplot(2,3,1), imshow(img); title('Input Orginal');

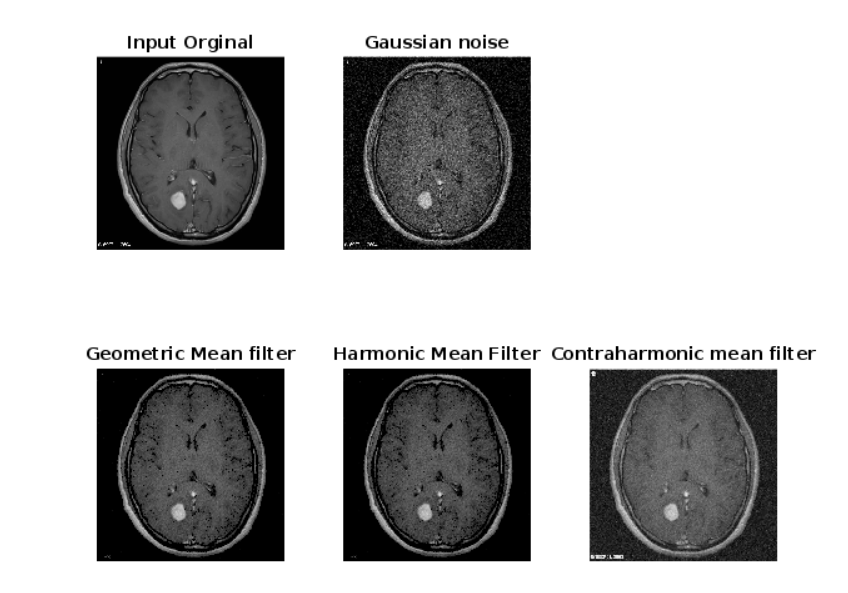
subplot(2,3,2), imshow(GI); title('Gaussian noise');

subplot(2,3,4), imshow(GM);title('Geometric Mean filter')

subplot(2,3,5), imshow(HM);title('Harmonic Mean Filter')

subplot(2,3,6), imshow(CM); title('Contraharmonic mean filter');

**Output:**



**2. Apply Gaussian noise to Figure 1, and then use the following order statistic filters to restore the image:**

**i. Median filter**

**ii. Maximum filter**

**iii. Minimum filter**

**iv. Midpoint filter**

**v. Alpha-trimmed filter**

**vi. Trimmed filter**

**Ans:**

**Code-**

img = imread('438image1.png'); % Read the image

img = rgb2gray(img);

% Add Gaussian noise to the image

img\_noise = imnoise(img, 'gaussian', 0, 0.01);

% Median filter

med\_filt = medfilt2(img\_noise, [5, 5]);

% Maximum filter

max\_filt = ordfilt2(img\_noise, 25, ones(5,5));

% Minimum filter

min\_filt = ordfilt2(img\_noise, 1, ones(5,5));

% Midpoint filter

mid\_filt = ordfilt2(img\_noise, 13, ones(5,5));

% Alpha-trimmed filter

alpha\_trim\_filt = imfilter(img\_noise, fspecial('unsharp',0.2), 'replicate');

alpha\_trim\_filt = imsubtract(img\_noise, alpha\_trim\_filt);

alpha\_trim\_filt = img\_noise - alpha\_trim\_filt;

% Trimmed filter

trimmed\_filt = ordfilt2(img\_noise, 5, ones(5,5));

%Display

subplot(2,3,1), imshow(med\_filt); title('Median Filter');

subplot(2,3,2), imshow(max\_filt); title('Maximum Filter');

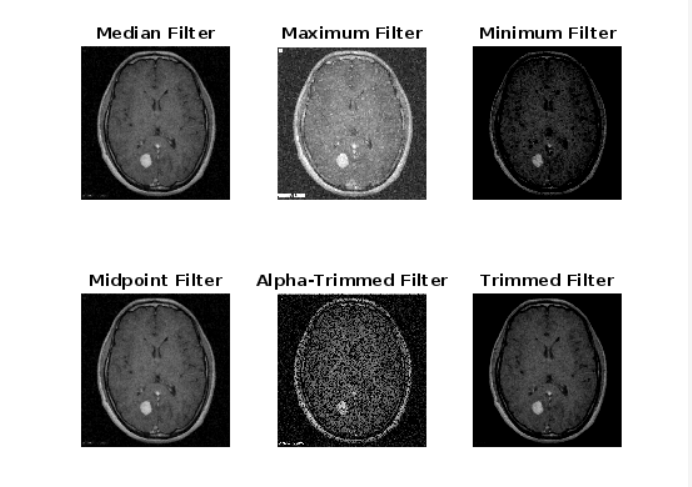
subplot(2,3,3), imshow(min\_filt); title('Minimum Filter');

subplot(2,3,4), imshow(mid\_filt); title('Midpoint Filter');

subplot(2,3,5), imshow(alpha\_trim\_filt); title('Alpha-Trimmed Filter');

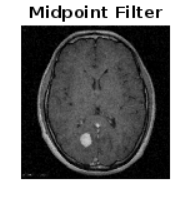
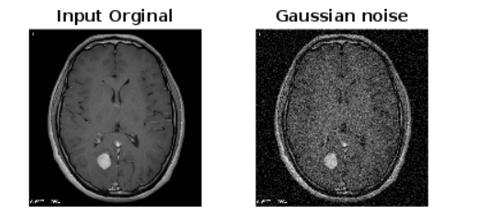
subplot(2,3,6), imshow(trimmed\_filt); title('Trimmed Filter');

**Output:**



**3.By observing and comparing each of the outputs, determine which filter restores the image closest to its original state. Mention the reasoning behind your observation and choose the most suitable image for the following problems.**

**Ans:**

****

By observing and comparing each of the outputs from question 1 and 2, Midpoint filter restores the image closest to its original state than other filters in my scenario. It is because, midpoint filter selects the midpoint, which is the average of the minimum and maximum values. It is very effective in removing Gaussian noise and uniform noise.

For those reasons, Midpoint filter restore the images perfectly.

**4. Detect the tumor from the image from Problem 3 using the segmentation approaches listed below:**

**(Outline the segmented object to highlight the tumor. You can crop the image for**

**accurate segmentation.)**

**i) Similarity approaches:**

**a) Local/Regional Thresholding**

**b) Global Thresholding**

**c) Variable Thresholding**

**d) Dynamic/Adaptive Thresholding**

**ii) Discontinuity approaches: Edge Detection (Sobel, Canny, Prewitt)**

**Ans:**

**Code-**

img = imread('438image1.png'); % Read the image

gray\_img= rgb2gray(img);

% Apply Gaussian filter

filtered\_img = imgaussfilt(gray\_img, 3);

% Perform local thresholding using Otsu's method

thresh = graythresh(filtered\_img);

bw\_img\_LT = imbinarize(filtered\_img, thresh);

% Perform global thresholding using a fixed threshold value

thresh = 0.6;

bw\_img\_GT = imbinarize(filtered\_img, thresh);

% Perform variable thresholding using adaptive thresholding

bw\_img\_VT = adaptthresh(filtered\_img, .1);

% Perform dynamic/adaptive thresholding

bw\_img\_AT = imbinarize(filtered\_img, 'adaptive', 'Sensitivity', 0.5);

% Perform edge detection using the Sobel operator

sobel\_img = edge(filtered\_img, 'sobel');

subplot(2,3,1), imshow(img ); title('Unsharp');

subplot(2,3,2), imshow(bw\_img\_LT); title('a) Local/Regional');

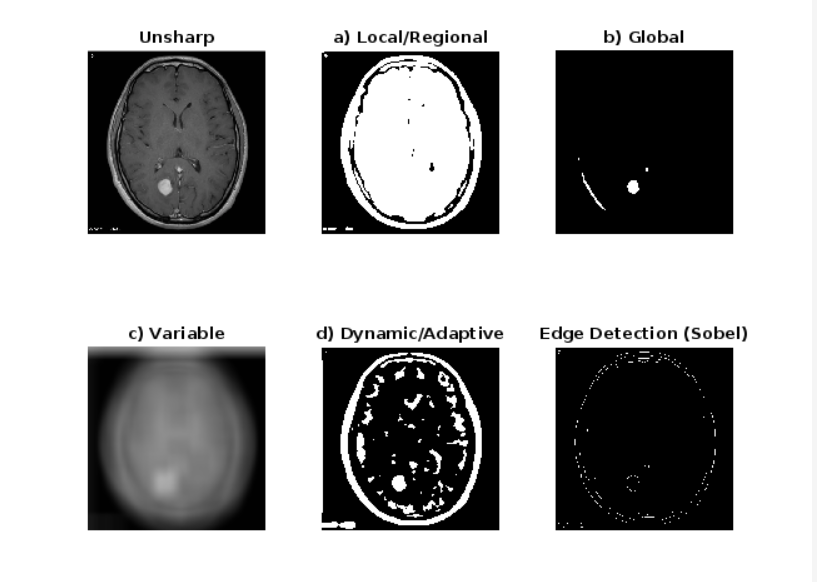
subplot(2,3,3), imshow(bw\_img\_GT); title('b) Global');

subplot(2,3,4), imshow(bw\_img\_VT ); title('c) Variable');

subplot(2,3,5), imshow(bw\_img\_AT); title('d) Dynamic/Adaptive');

subplot(2,3,6), imshow(sobel\_img ); title('Edge Detection (Sobel)');

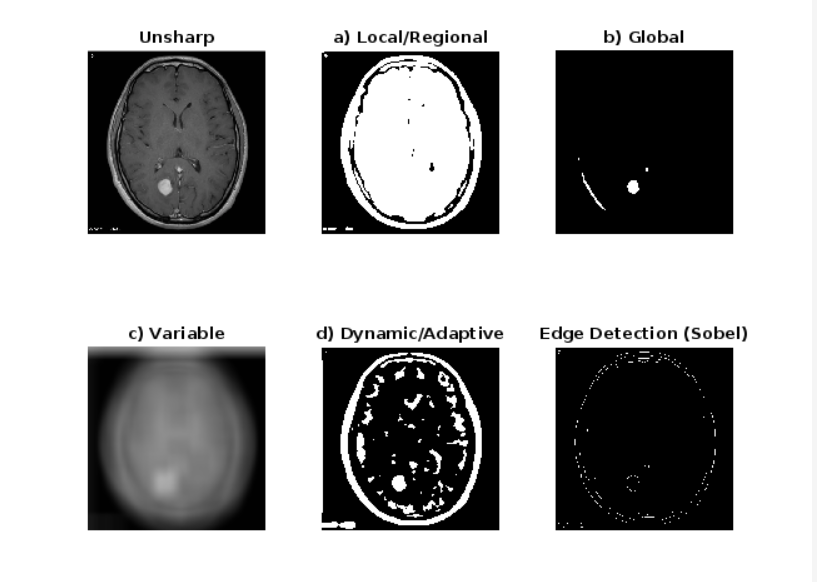
**Output:**



Here in the images, we can see that the global thresholding and dynamic/ adaptive thresholding can identify the tumor (round shape) perfectly. Specially the adaptive thresholding can highlight all the edges including the tumor. So, in my case, the adaptive has worked better than other segmentation processes including global thresholding. Because global thresholding only highlighted the round shape, it could miss the tumor in another scenario.

**5. Show how the Similarity and Discontinuity techniques compare.**

**Ans:**



Similarity and discontinuity techniques are two main types of image segmentation approaches. Similarity techniques are based on grouping pixels with similar properties or characteristics, such as color or texture, into regions. On the other hand, discontinuity techniques are based on detecting edges or boundaries between different regions of an image.

In the case of tumor detection, similarity techniques can be useful when the tumor has different characteristics or properties than the surrounding tissue. For example, if the tumor is darker or brighter than the surrounding tissue, thresholding techniques can be used to segment the tumor based on differences in intensity values. However, if the tumor is similar in intensity to the surrounding tissue, these techniques may not be effective.

Discontinuity techniques, such as edge detection, can be useful in detecting the boundary between the tumor and the surrounding tissue. By identifying the edges or boundaries, it is possible to segment the tumor from the surrounding tissue. However, these techniques may not be effective if the boundary between the tumor and surrounding tissue is not well-defined or if there is noise in the image.

In general, both similarity and discontinuity techniques can be useful in image segmentation, and the choice of technique will depend on the specific characteristics of the image and the object of interest. A combination of different techniques may also be used for more accurate segmentation. But, in my case, the Similarity technique (adaptive thresholding) has worked better.

**6. Segment the tumor from Figure 2 by using:**

**i. Region growing approach**

**ii. Region Splitting and Merging approach**

**Ans:**

**i. Region growing approach:**

function J=regiongrowing (I,x,y,reg\_maxdist)

if(exist('reg\_maxdist','var')==0), reg\_maxdist=0.2; end

if(exist('y','var')==0), figure, imshow(I,[]); [y,x]=getpts; y=round(y(1)); x=round(x(1)); end

J = zeros(size(I));

Isizes = size(I);

reg\_mean = I(x,y);

reg\_size = 1;

neg\_free = 10000; neg\_pos=0;

neg\_list = zeros(neg\_free,3);

pixdist=0;

neigb=[-1 0; 1 0; 0 -1;0 1];

diff = 01;

while(pixdist<reg\_maxdist && reg\_size<numel(I) && diff ~=0)

num1 = sum(sum(reg\_size));

% Add new neighbors pixels

for j=1:4,

% Calculate the neighbour coordinate

xn = x +neigb(j,1); yn = y +neigb(j,2);

% Check if neighbour is inside or outside the image

ins=(xn>=1)&&(yn>=1)&&(xn<=Isizes(1))&&(yn<=Isizes(2));

% Add neighbor if inside and not already part of the segmented area

if(ins&&(J(xn,yn)==0))

neg\_pos = neg\_pos+1;

neg\_list(neg\_pos,:) = [xn yn I(xn,yn)]; J(xn,yn)=1;

end

end

if(neg\_pos+10>neg\_free), neg\_free=neg\_free+10000; neg\_list((neg\_pos+1):neg\_free,:)=0; end

dist = abs(neg\_list(1:neg\_pos,3)-reg\_mean);

[pixdist, index] = min(dist);

J(x,y)=2; reg\_size=reg\_size+1;

reg\_mean= (reg\_mean\*reg\_size + neg\_list(index,3))/(reg\_size+1);

x = neg\_list(index,1); y = neg\_list(index,2);

neg\_list(index,:)=neg\_list(neg\_pos,:); neg\_pos=neg\_pos-1;

num2 = sum(sum(reg\_size));

diff = num2-num1;

end

J=J>1;

I = rgb2gray(im2double(imread('438image2.png')));

J=imadjust(I,[],[],0.5);

J=imgaussfilt(J);

J= regiongrowing(J,0.2);

imshowpair(I,J,'montage')

**output:**

A close-up of a brain scan

Description automatically generated with low confidence

**ii. Region Splitting and Merging approach:**

function g = splitmerge(f, mindim, fun)

Q = 2.^nextpow2(max(size(f)));

[M, N] = size(f);

f = padarray(f, [Q - M Q - N], 'post');

S = qtdecomp(f, @split\_test, mindim, fun);

Lmax = full(max(S(:)));

g = zeros(size(f));

MARKER = zeros(size(f));

for K = 1:Lmax

[vals, r, c] = qtgetblk(f, S, K);

if ~isempty(vals)

for I = 1:length(r)

xlow = r(I); ylow = c(I);

xhigh = xlow + K - 1; yhigh = ylow + K - 1;

region = f(xlow:xhigh, ylow:yhigh);

flag = feval(fun, region);

if flag

g(xlow:xhigh, ylow:yhigh) = 1;

MARKER(xlow, ylow) = 1;

end

end

end

end

g = bwlabel(imreconstruct(MARKER, g));

g = g(1:M, 1:N);

function v = split\_test(B, mindim, fun)

v(1:k) = false;

for I = 1:k

quadregion = B(:, :, I);

if size(quadregion, 1) <= mindim

v(I) = false;

continue

end

flag = feval(fun, quadregion);

if flag

v(I) = true;

end

end

I2 = imread('438image2.png');

g = splitmerge(I2,8,@predicate);

figure,subplot(1,2,1);imshow(I2,[]);

subplot(1,2,2);imshow(g,[]);

A close-up of a brain

Description automatically generated with low confidence

**7. Segment the tumor from Figure 2 by using Marker Controlled Watershed segmentation.**

**Ans:**

x=imbinarize(rgb2gray(imread('438image2.png')));

subplot(2,2,1);

imshow(x);

title(' Original Image ');

a=x;

x=~x;

ms=bwdist(x);

ms=255-uint8(ms);

subplot(2,2,2);

imshow(ms);

title('After applying Distance Transformation ');

hs=watershed(ms);

ws=hs==0;

subplot(2,2,3);

imshow(a | ws);

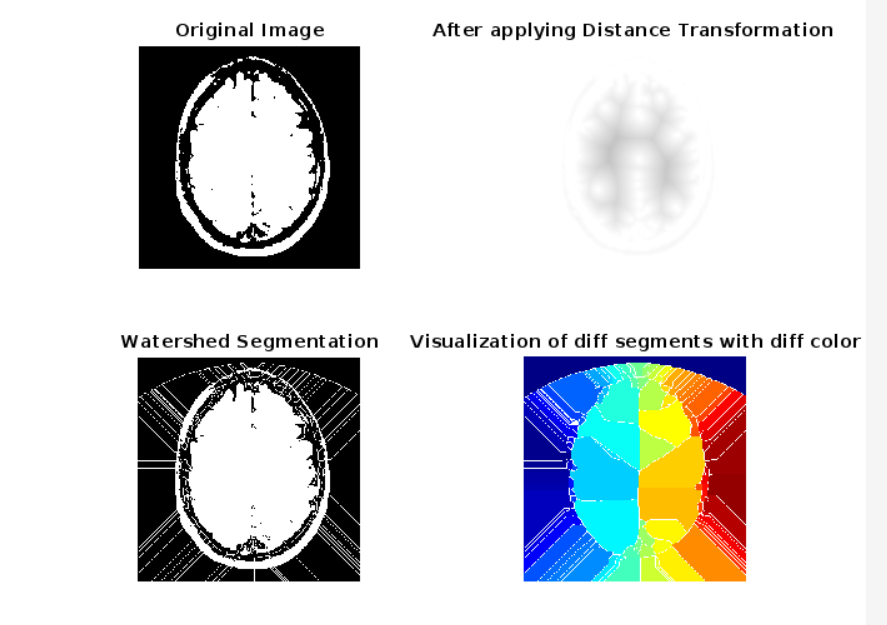
title(' Watershed Segmentation ');

subplot(2,2,4);

imshow(label2rgb(hs));

title(' Visualization of diff segments with diff color ');

**Output:**

****

**8. Segment the tumor from Figure 2 by using Quadtree Segmentation.**

**Ans:**

I = imread('438image2.png');

I=rgb2gray(I);

Ifilt = medfilt2(I,[8 8]);

[S] = qtdecomp\_var(I,10);

[Sfilt] = qtdecomp\_var(Ifilt);

blocks = repmat(uint8(0),size(S));

for dim = [256 128 64 32 16 8 4 2 1];

numblocks = length(find(S==dim));

if (numblocks > 0)

values = repmat(uint8(1),[dim dim numblocks]);

values(2:dim,2:dim,:) = 0;

blocks = qtsetblk(blocks,S,dim,values);

end

end

blocks(end,1:end) = 1;

blocks(1:end,end) = 1;

blocks\_filt = repmat(uint8(0),size(Sfilt));

for dim = [128 64 32 16 8 4 2 1];

numblocks = length(find(Sfilt==dim));

if (numblocks > 0)

values = repmat(uint8(1),[dim dim numblocks]);

values(2:dim,2:dim,:) = 0;

blocks\_filt = qtsetblk(blocks\_filt,Sfilt,dim,values);

end

end

blocks\_filt(end,1:end) = 1;

blocks\_filt(1:end,end) = 1;

figure;subplot(221);imshow(I); title('Input image');

subplot(222);imshow(Ifilt); title('Filtered image');

subplot(223);imshow(blocks,[]);title('Quad tree input image with var weight');

subplot(224);imshow(blocks\_filt,[]);title('Quad tree filtered image without var weight');

**output:**

A picture containing screenshot, stitch, pattern

Description automatically generated

**9. Generate a binary mask of the tumor from Figure 2 using any segmentation method of your choice, then apply:**

**i. Morphological Dilation**

**ii. Morphological Erosion**

**Ans:**

rgb = imread('438image2.png');

I = rgb2gray(rgb);

gmag = imgradient(I);

L = watershed(gmag);

Lrgb = label2rgb(L);

se = strel('disk',20);

Io = imopen(I,se);

Ie = imerode(I,se);

Iobr = imreconstruct(Ie,I);

Ioc = imclose(Io,se);

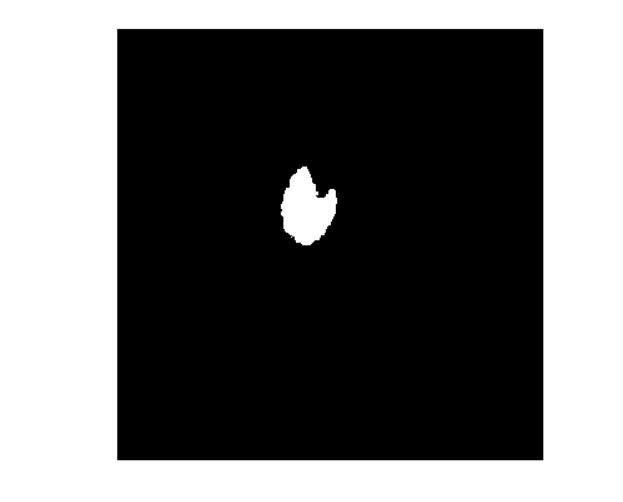
Iobrd = imdilate(Iobr,se);

Iobrcbr = imreconstruct(imcomplement(Iobrd),imcomplement(Iobr));

Iobrcbr = imcomplement(Iobrcbr);

fgm = imregionalmax(Iobrcbr);

figure,imshow(fgm)



**i. Morphological Dilation-**

bw =imread('438image2.png');

% Define the structuring element

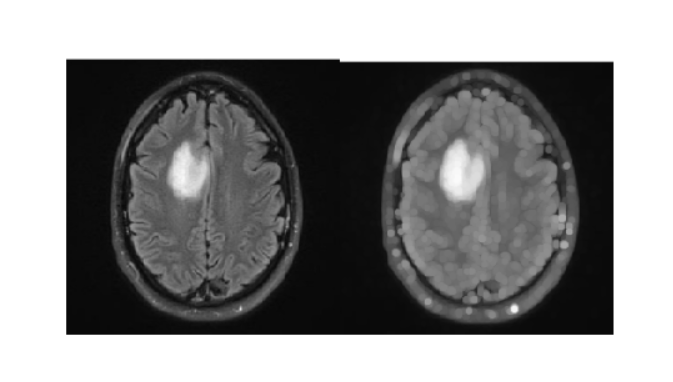
se = strel('disk', 5); % A disk-shaped structuring element with radius 5

% Perform morphological dilation

bw\_dilated = imdilate(bw, se);

% Display the original and the dilated images side by side

imshowpair(bw, bw\_dilated, 'montage');



**ii. Morphological Erosion**

bw =imread('438image2.png');

% Define the structuring element

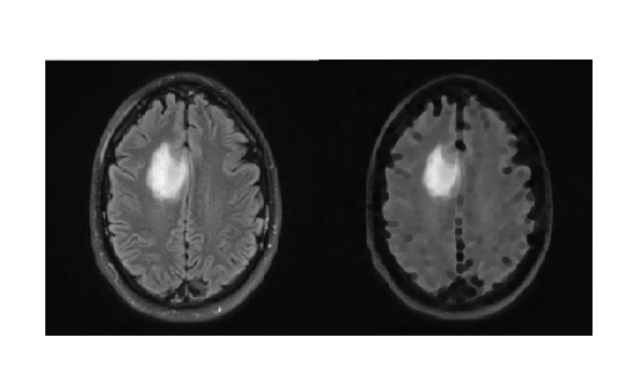
se = strel('disk', 5); % A disk-shaped structuring element with radius 5

% Perform morphological erosion

bw\_eroded = imerode(bw, se);

% Display the original and the eroded images side by side

imshowpair(bw, bw\_eroded, 'montage');



**10. Apply the Hough transform to Figure 3 and draw the detected lines on the original image.**

**Ans:**

img = imread('438image3.png');

grayImg = rgb2gray(img);

edgeImg = edge(grayImg, 'Canny');

[H,theta,rho] = hough(edgeImg,'Theta',-90:0.5:89.5);

peaks = houghpeaks(H,10,'Threshold',0.3\*max(H(:)));

lines = houghlines(edgeImg,theta,rho,peaks,'FillGap',10,'MinLength',20);

imshow(img);

hold on;

for k = 1:length(lines)

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

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

end

**output:**

