EAST WEST UNIVERSITY
Department of Computer Science and Engineering

Lab-report:09

Course Name: Digital Image Processing
Course Code: CSE438
Section No: 03

Submitted To:

Prof. Dr. Ahmed Wasif Reza

Department of Computer Science and Engineering

East West University

Submitted by:

Student's ID: 2020-3-60-012

Student's Name: Sadia Islam Prova

Date of submission: 24-5-24

Problem 1: Segment the tumor from Figure 1 by using:

i. Region growing approach

ii. Region Splitting and Merging approach

Code:

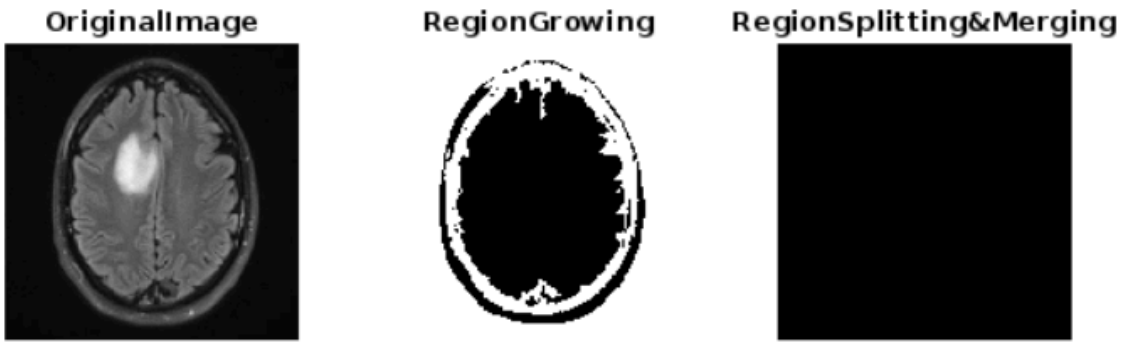
```
image = imread('tumor.png');
seed_point = [100, 100];
threshold_rg = 20;
segmented_image_rg = region_growing(image, seed_point, threshold_rg);
segmented_image_rsm = region_splitting_merging(image);
figure;
subplot(1,3,1);
imshow(image);
title('OriginalImage');
subplot(1,3,2);
imshow(segmented_image_rg);
title(['RegionGrowing']);
subplot(1,3,3);
imshow(segmented_image_rsm);
title('RegionSplitting&Merging');
function segmented_image = region_growing(image, seed_point, threshold)

    segmented_image = zeros(size(image));

    queue = seed_point;

    connectivity = [-1, -1; -1, 0; -1, 1; 0, -1; 0, 1; 1, -1; 1, 0; 1, 1];

    while ~isempty(queue)
```

```
        current_pixel = queue(1,:);
        queue(1,:) = [];

        if current_pixel(1) >= 1 && current_pixel(1) <= size(image, 1) && ...
           current_pixel(2) >= 1 && current_pixel(2) <= size(image, 2)

            if segmented_image(current_pixel(1), current_pixel(2)) == 0 && ...
               abs(image(current_pixel(1), current_pixel(2)) -
image(seed_point(1), seed_point(2))) <= threshold

                segmented_image(current_pixel(1), current_pixel(2)) = 1;

                for i = 1:size(connectivity, 1)
                    neighbor = current_pixel + connectivity(i,:);
                    queue = [queue; neighbor];
                end
            end
        end
    end
end
function segmented_image = region_splitting_merging(image)

    segmented_image = zeros(size(image));

    region = struct('pixels', [], 'mean_intensity', 0);
    region.pixels = [1, 1];
    region.mean_intensity = image(1, 1);
```

```
   split_threshold = 10;


   merge_threshold = 20;


   segmented_image = split_merge(region, image, segmented_image,
split_threshold, merge_threshold);

end

function segmented_image = split_merge(region, image, segmented_image,
split_threshold, merge_threshold)

    if std2(region.pixels) > split_threshold


       subregions = split(region, image);




       for i = 1:length(subregions)

           segmented_image = split_merge(subregions(i), image, segmented_image,
split_threshold, merge_threshold);

       end

   else


       mergeable = is_mergeable(region, segmented_image, merge_threshold);

       if mergeable


           segmented_image = merge(region, segmented_image);

       else


           segmented_image(region.pixels) = 1;

       end

   end

end

function subregions = split(region, image)
```

```
end
function mergeable = is_mergeable(region, segmented_image, merge_threshold)


   mergeable = false;
end
function segmented_image = merge(region, segmented_image)


end
```

Output:



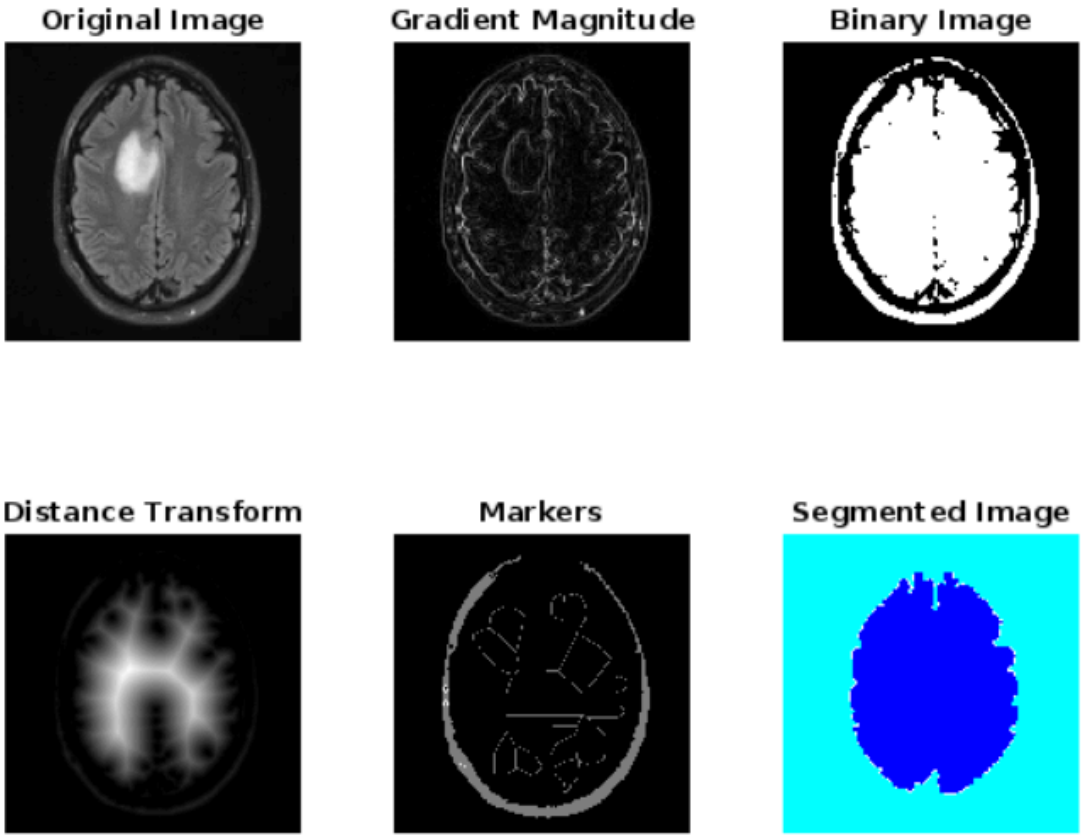
Problem 2: Segment the tumor from Figure 1 by using Marker Controlled Watershed segmentation.

Code:

```
gray_img = imread('tumor.png');
figure;
subplot(2, 3, 1);
imshow(gray_img);
title('Original Image');
gradient_magnitude = imgradient(gray_img, 'sobel');
bw = imbinarize(gray_img);
```

```
bw = bwareaopen(bw, 20);
distance_transform = bwdist(~bw);
DL = watershed(distance_transform);
bg_markers = DL == 0;
cc = bwconncomp(bw);
markers = false(size(gray_img));
markers(cc.PixelIdxList{1}) = true;
fg_markers = bwlabel(markers);
markers = fg_markers + bg_markers;
L = watershed(gradient_magnitude);
L(~bw) = 0;
L2 = imimposemin(gradient_magnitude, markers);
ws = watershed(L2);
segmented_img = label2rgb(ws, 'jet', 'w', 'shuffle');
subplot(2, 3, 2);
imshow(gradient_magnitude, []);
title('Gradient Magnitude');
subplot(2, 3, 3);
imshow(bw);
title('Binary Image');
subplot(2, 3, 4);
imshow(distance_transform, []);
title('Distance Transform');
subplot(2, 3, 5);
imshow(markers, []);
title('Markers');
subplot(2, 3, 6);
imshow(segmented_img);
title('Segmented Image');
```

Output:



Problem 3: Segment the tumor from Figure 1 by using Quadtree Segmentation.

Code:

```
gray_img = imread('tumor.png');
gray_img = im2double(gray_img);
threshold = 0.1;
segmented_img = quadtreeSegmentation(gray_img, threshold);
figure;
subplot(1, 2, 1);
```

```
imshow(gray_img);
title('Original Image');
subplot(1, 2, 2);
imshow(segmented_img);
title('Segmented Image (Quadtree)');
function segmented_img = quadtreeSegmentation(img, threshold)

   segmented_img = zeros(size(img));



   if std(img(:)) > threshold

       [rows, cols] = size(img);
       mid_row = floor(rows / 2);
       mid_col = floor(cols / 2);



       quad1 = img(1:mid_row, 1:mid_col);
       quad2 = img(1:mid_row, mid_col+1:end);
       quad3 = img(mid_row+1:end, 1:mid_col);
       quad4 = img(mid_row+1:end, mid_col+1:end);



       segmented_img(1:mid_row, 1:mid_col) = quadtreeSegmentation(quad1,
threshold);
       segmented_img(1:mid_row, mid_col+1:end) = quadtreeSegmentation(quad2,
threshold);
       segmented_img(mid_row+1:end, 1:mid_col) = quadtreeSegmentation(quad3,
threshold);
       segmented_img(mid_row+1:end, mid_col+1:end) =
quadtreeSegmentation(quad4, threshold);
   else


   end
```

```
end
```

Output: