**PROJECT Report**

|  |  |  |  |
| --- | --- | --- | --- |
| **PROJECT TITLE** | Brain Tumor Detection | | |
| **SUBMITTED BY** | Muhammad Zeeshan Khalid | **SUBMITTED TO** | Muhammad Sharif Malik |
| **PHONE / EMAIL** |  | **RECEIVER  PHONE / EMAIL** |  |
| **DATE  SUBMITTED** | December 16, 2021 | **PROJECTED  START DATE** |  |

|  |
| --- |
| Provide Name, Title... |
| **PROJECT SPONSOR** |
| Muhammad Zeeshan Khalid |
| **FUNDING SPONSOR** |
| Mhammad Zeeshan Khalid |
| **PROJECT OWNER** |
| Muhammad Zeeshan Khalid |
| **PROPOSAL FACILITATOR** |
| Muhammad Zeeshan Khalid |

|  |
| --- |
| **Basic Tool** |
| We are doing this project on MATLAB. |

|  |
| --- |
| **Overview** |
| We have done following work:   * Taken an MRI image of brain from user. * Resized input image. * Applied RGB Filter. * Applied Anisotropic Diffusion. * Obtained Threshold value in project through Hit and Trial technique. * Separating the Threshold-Area image from input image. * Separating the High-Dense-Area and putting Boundary box in the image * Output Images with previous task. * Show the tumor alone in separated image using threshold values * Show the tumor outline on another image * Show the tumor outline in the separate image * Applied the Prewit-filter also, to show the tumor region in another view |
| **Description** |
| In this project, require image path is firstly given to program and then it is resized to appropriate size 256x256. After resizing the image smoothing filter has been applied. There can be various types of filters to remove noise, but here we have used the Anisotropic diffusion filter because this filter will not make the edges of picture muddy, since it is tumor detection project, proper finding of tumor with its clear edges is primary goal.  From the formula of anisotropic diffusion:  Here all the anisotropic diffusion equation depends on ‘c’. Further details about this filter will be given, if required.  After filtering, for reducing the chance of exception, we again put our diffused image in uint8 format and also it is checked that image should not be in RGB because further processes are done on grayscale images. These tasks are done using uint8(image\_input) and image\_input=rgb2gray(image\_input) functions.  Then the maximum and minimum value of input image are obtained and then taken an average for threshold, since average was not giving us right answer therefore, we incremented X (obtained through Hit & Trial) for right threshold. These tasks are done using min() and max() functions.  After that, using nested loop of X\*Y coordinate which is equal to 256X256. The threshold values are extracted from the original image. Now from the newly obtained image from pre-processing, further operations are fulfilled in which regionprops( ) function is used to find Solidity, Area, BoundingBox properties for an image. Now from these properties Tumor image is extracted separately.  Now from the extracted tumor image, we obtained the eroded part of the image. The main reason to find the eroded part is to make the outline around the tumor according to shape to shape and from these tasks, another separate image could be extracted which have the tumor shape\_outline marked on it. These above tasks are done using the concept of segmentation by getting some threshold values using Hit & Trial Techniques, if-else conditions and nested loops.  Finally, after all these procedures, we added Prewit-filter and Canny-filter which was studied in the course. This filter was added on tumor region picture and then it was paired to original MRI image, fir clearer and better output view.  Then after these things we got our final outputs. |
| **Code** |
| %%%%%%%%%%%%%%%%%image\_inputut-image%%%%%%%%%%%%%%%%%%%%%%%%%%%%  disp("Taking image as image\_input. . . . . . ");  s=imread('C:\Users\Toshiba\OneDrive\Desktop\dip task3\brain1.jpg');  figure;  imshow(s);  title('\bf Input image','FontSize',20,'color','black');    %%%%%%%%%%%%%%%%%RGB & resizing%%%%%%%%%%%%%%%%%%%%%%%%%%%%  disp("converting to RGB. . . . . .");  disp("resizing image. . . . . . .");  s=imresize(s,[256,256]);  %s=label2rgb(im2gray(s));  figure;  imshow(label2rgb(im2gray(s)));  title('\bf RGB','FontSize',20,'color','black');    %%%%%%%%%%%%%%%%%Applying filter%%%%%%%%%%%%%%%%%%%%%%%%%%%%  %%%%%%%%%%%%%%%%%ANISTROPIC DIFUSSION%%%%%%%%%%%%%%%%%%%%%%%  disp("Aplying the filter. . . . . . .");  image\_input = imdiffusefilt(s);  disp("Anistropic diffusion applied. . . . . . .");  image\_input = uint8(image\_input);%just reapplying uint-8, incase there is no unit-8, therefore%0-255==2^8//dealing in grayscale  %css=size(image\_input,3);    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%TASK-3%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%  %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%    if size(image\_input,3)>1%double checking that should not be RGB  image\_input=rgb2gray(image\_input);  end  %thresholding  css=image\_input(:);%css=256\*256  sout=image\_input;    %getting maximum value from the 256\*256  mx\_value=max(image\_input(:));%max(css)//max returns the maximum color intensity present in image as parameter  %getting minimum value from the 256\*256  mn\_value=min(image\_input(:));  %getting average value from the obtained data  Avrg=(mx\_value+mn\_value)./2;%(.) or without (.)  %applying threshold value, it is changed with respect to hit&Trial technique  threshHold=Avrg+45;    sizeCheck=size(image\_input,2);  %getting the size value first, from first dimension and it will be 256,  %as we resized the image earlier, then performing nested loop  for x=1:1:size(image\_input,1)  for y=1:1:size(image\_input,2)  if image\_input(x,y)>threshHold  %if size will be greater than our threshold, then we put as 1  sout(x,y)=1;  else  %if size will be lesser than our threshold, then we put as 0  sout(x,y)=0;  end  end  end    %regionprop() will get the properties of image, which we required  %logical(sout) returns the total size which is 256\*256  %Soldidity=area's pixels from the input image  %Area=area from the image\_inputut image  %BoundingBox=position and its size from the input image w.r.t(areaTotal)  stats=regionprops(logical(sout),'Solidity','Area','BoundingBox');  %Getting density  Image\_density=[stats.Solidity];  %Getting Area  Image\_area=[stats.Area];  %Applying pre-processing before actual processes  Image\_high\_dense\_area=Image\_density>0.6;  %seperating the area which is more dense  mx\_area=max(Image\_area(Image\_high\_dense\_area));  %apply labeling  tumor\_label=find(Image\_area==mx\_area);  %converting the image in BlacknWhite of size 256\*256  label=bwlabel(sout);  %returning true values of ismember(a,b) in tumor  tumor=ismember(label,tumor\_label);  %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%  %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%  if mx\_area>100%%%puttiing another threshold  figure;  imshow(tumor)  title('tumor alone','FontSize',20);  else  h = msgbox('No Tumor!!','status');  %disp('no tumor');  return;  end  %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%  range = 5;  rad = floor(range);  [p,q] = size(tumor);  filledImage = imfill(tumor, 'holes');%%negkecting blckish region inside hole  for i=1:p  for j=1:q  x\_1=i-rad;  x\_2=i+rad;  y\_1=j-rad;  y\_2=j+rad;  if x\_1<1  x\_1=1;  end  if x\_2>p  x\_2=p;  end  if y\_1<1  y\_1=1;  end  if y\_2>q  y\_2=q;  end  ErodedPart(i,j) = min(min(filledImage(x\_1:x\_2,y\_1:y\_2)));  end  end  figure  imshow(ErodedPart);  title('eroded image','FontSize',20);  tumorOutline=tumor;  tumorOutline(ErodedPart)=0;  figure;  imshow(tumorOutline);  title('Tumor Outline','FontSize',20);  %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%  rgb = image\_input(:,:,[1 1 1]);  R = rgb(:,:,1);  R(tumorOutline)=255;  G = rgb(:,:,2);  G(tumorOutline)=0;  B = rgb(:,:,3);  B(tumorOutline)=0;  OutlineInsertedTumor(:,:,1) = R;  OutlineInsertedTumor(:,:,2) = G;  OutlineInsertedTumor(:,:,3) = B;  figure;  imshow(OutlineInsertedTumor);  title('Detected Tumer','FontSize',20);    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%  BW2 = edge(image\_input,'Prewitt');  figure;imshowpair(BW2,tumor)  title('PREWIT FILTER','FontSize',20);  BW2 = edge(image\_input,'Prewitt');  figure;imshowpair(BW2,tumor)  title('PREWIT FILTER','FontSize',20);  %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%  %making box on image, after pre-processing    %getting bounding box around high dense area~(detected tumor)  box = stats(tumor\_label);  %making boundary box area size of max area(pre-processing)  wantedBox = box.BoundingBox;  %%%%%%%%%%%%%%%%%%%DONE task#1%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%    subplot(241);imshow(s);title('Input image','FontSize',20);  subplot(242);imshow(image\_input);title('Filtered image','FontSize',20);  subplot(243);imshow(s);title('Bounding Box','FontSize',20);hold on;rectangle('Position',wantedBox,'EdgeColor','red');hold off;  subplot(244);imshow(tumor);title('Tumor alone','FontSize',20);  subplot(245);imshow(tumorOutline);title('Tumor Outline','FontSize',20);  subplot(246);imshow(OutlineInsertedTumor);title('Dectected','FontSize',20);  subplot(247);imshowpair(BW2,tumor);title('PREWIT FILTER','FontSize',20);  subplot(248);imshow(label2rgb(im2gray(s)));title('RGB image','FontSize',20); |
| **Output Images** |
| Output Images are as Follows: |