**Detecting Tumor from MRI Images**

18AIS101J – Introduction to MATLAB for Artificial Intelligence

Submitted by:

Vasa Chandana-RA2011047010080

Soumya Mangal-RA2011047010088

Aluru Leela Rani-RA2011047010095

Shruti Iyengar-RA2011047010105

**MINI PROJECT**

BACHELOR OF TECHNOLOGY



**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING**

# **FACULTY OF ENGINEERING AND TECHNOLOGY**

**SRM INSTITUTE OF SCIENCE AND TECHNOLOGY**

**KATTANKULATHUR- 603 203**

**February 2021**

****

**BONAFIDE CERTIFICATE**

Certified to be the project report titled **“Detecting Tumor from MRI Images”** is the bonafide work done by Vasa Chandana (RA2011047010080), Soumya Mangal (RA2011047010088), Aluru Leela Rani (RA2011047010095), Shruti Iyengar (RA2011047010105) of AI B. Tech Degree course in the theory **18AIS101J – Introduction to MATLAB for Artificial Intelligence** during the academic year 2020-2021.

Signature of the Faculty

Dr.S.Thanga Revathi

AP/CSE

**Date of Submission: 21.02.21**

**ABSTRACT**

Brain Tumor is a fatal disease which cannot be confidently detected without MRI.In this project, it is attempted to detect if the patient’s brain has a tumor or not from MRI images using MATLAB simulation. We achieve this by applying concepts of GUI (Graphical User Interface) and Anisotropic Diffusion. Using the GUI, this program can use various combinations of segmentation, filters, and other image processing algorithms to achieve the best results. To pave the way for morphological operation on MRI image, the image was first filtered using Anisotropic Diffusion Filter to reduce contrast between consecutive pixels. After that the image was resized and utilizing a threshold value image was converted to a black and white image manually. This primary filter the plausible locations for tumor presence. On this semi processed image morphological operations have been applied and information on solidity and areas of the plausible locations was obtained. A minimum value of both of these characters has been determined from statistical average of different MRI images containing tumor. Then it was used to deliver final detection result. The larger goal of the project is to build a data base of 2D image data of tumor fromthe MRI images taken from different angles of a particular human and by analyzing them to point out the exact 3D location of the tumor. To fulfill this, 2D tumor detection and segmentation have been developed to better accuracy so that 3D detection can be more reliable. This is the primary target of the project.

**TABLE OF CONTENTS**

**CHAPTERNO. TITLE PAGE NO**

ABSTRACT 3

1. INTRODUCTION 5
2. PROJECT DESCRIPTION 6
3. CODE 8
4. SCREENSHOTS 18

CONCLUSION 22

REFERENCES 22

**INTRODUCTION:**

A brain tumor, known as an intracranial tumor, is an abnormal mass of tissue in which cells grow and multiply uncontrollably, seemingly unchecked by the mechanisms that control normal cells.

Histological grading, based on a stereotactic biopsy test, is the gold standard and the convention for detecting the grade of a brain tumor. The biopsy procedure requires the neurosurgeon to drill a small hole into the skull from which the tissue is collected. There are many risk factors involving the biopsy test, including bleeding from the tumor and brain causing infection, seizures, severe migraine, stroke, coma and even death. But the main concern with the stereotactic biopsy is that it is not 100% accurate which may result in a serious diagnostic error followed by a wrong clinical management of the disease. Tumor biopsy being challenging for brain tumor patients, non-invasive imaging techniques like Magnetic Resonance Imaging (MRI) have been extensively employed in diagnosing brain tumors. Therefore, development of systems for the detection and prediction of the grade of tumors based on MRI data has become necessary.

It needs to be detected at an early stage using MRI or CT scanned images when it is as small as possible because the tumor can possibly result to cancer. This project mainly focuses on detecting and localizing the tumor region existing in the brain by proposed methodology using patient's MRI images. The proposed methodology consists of three stages i.e. pre-processing, edge detection and segmentation. Pre-processing stage involves converting original image into a grayscale image and removes noise if present or crept in. This is followed by edge detection using few algorithms with image enhancement techniques. Next, segmentation is applied to clearly display the tumor affected region in the MRI images. Finally, the brain tumor is detected using anisotropic methods and other few concepts. Here we have used MATLAB for the development of the project.

**PROJECT DESCRIPTION:**

In the beginning, we propose a graphical interface in which the callback of each button translates all methods used from the programming code which contains functions Mfile. MATLAB software is used.The button (Load Image) located next to the original image window is to upload the picture from the database to be treated giving the image in the next window after noise and filter and extraction of the tumor. The button “Detect tumor” starts processing until the image is displayed after processing. As the tumor is found, there is a message that shows the highest result picture saying "Brain Tumor detected".

On the other hand, in case of no tumor, a message appears showing the highest result picture saying" No Tumor!!".

After implementation of the proposed algorithm on MATLAB, we obtain the desired results.

From a database given in Figure with different tumors locations and sizes, we will select a figure as input image. The predefined function is used to load an image. Then, we obtain two figures, the first with a uniform noise and the second with a salt and pepper noise to show the noise effect on image.

As next step, filtering technique for modifying or enhancing an image, is put to use. We use predefined functions to filter the image like solidity function, Area function and etc.

For the two types of noise cited previously, even when applying the filter severely, a little part of noise will be eliminated but a blur on image cannot be removed.

This filter realizes a weighted average of the values in the entire neighborhood: in the center, it is about a strong weight and low at the periphery. Another function develops an Anisotropic Diffusion method to filter the input image to get the Figureby removing all noise.

In the rest of work, we will use the Anisotropic Diffusion filter because it is the best in terms of filtering an image and removing noise from it. Also,Anisotropic Diffusion preserves the sharpness of edges better than Gaussian blurring. So, we choose to continue with Anisotropic Diffusion Filter function which is written to be applied to the image and we determine the amount of filter for each image to become clearer, this will show the image after being processed with filter Anisotropic Diffusion. In this.After application of the best filter selected, we refer to morphological operation which is based on segmentation technique. Morphology is a broad set of image processing operations that process images based on shapes. In a morphological operation, each pixel in the image is adjusted based on the value of other pixels in its neighborhood. The most basic morphological operations are dilation and erosion. We used the Erosion technique which replaces each pixel with the local minimum of the neighborhood around the pixel. The object operates on a stream of binary intensity values. This object uses a streaming pixel interface with a structure for frame control signals. In contrary, the dilation is block that replaces each pixel with the local maximum of the neighborhood around the pixel. The block operates on a stream of binary intensity values. This block uses a streaming pixel interface with a bus for frame control signals. This interface enables the block to operate independently of image size and format. In case of tumor, we must follow the steps cited next or display a message "No Tumor!!”. First, we will measure properties of image region using the function "regionprops" which has as output Stats. This contains area, Bounding Box and solidity. Next, we isolate the tumorin Figure 3. We will highlight the tumor using "Bounding box" knowing that Bounding boxes are imaginary boxes that go around objects, highlighting the area they cover, that will be shown in Figure 4. Then to complete the steps of morphological operation we obtain allfigures resulting from erosion and dilation techniques.

As last part, to confirm the detection we insert the outline in filtered image in red color.

Finally, the goal is achieved. To conclude, we try to enumerate all stages followed from the beginning after selecting the right filter and continue workingon it. The last Figure shows this succession of steps using a subplot. This is how the Brain tumor detection with anisotropic diffusion works.

This is the whole concept of Brain Tumor Detection using MRI Images. The same concept is used in the three codes namely, Graphical User Interface, Subplots and Anisotropic Diffusion.

Graphical User Interface code uses this concept to detect the tumor by availing GUI with two buttons for loading image and detecting tumor and then after processing the resulting image is displayed and “no tumor!!” text in a small dialog box if there’s no tumor in the input MRI image, else it displays theimage with a red boundary around the tumor, displaying “Tumor Detected” as the message.

**CODE:**

**GRAPHICAL USER INTERFACE**

functionvarargout = GUI(varargin)

% GUI MATLAB code for GUI.fig

% GUI, by itself, creates a new GUI or raises the existing

% singleton\*.

%

% H = GUI returns the handle to a new GUI or the handle to

% the existing singleton\*.

%

% GUI('CALLBACK',hObject,eventData,handles,...) calls the local

% function named CALLBACK in GUI.M with the given input arguments.

%

% GUI('Property','Value',...) creates a new GUI or raises the

% existing singleton\*. Starting from the left, property value pairs are

% applied to the GUI before GUI\_OpeningFcn gets called. An

% unrecognized property name or invalid value makes property application

% stop. All inputs are passed to GUI\_OpeningFcn via varargin.

%

% \*See GUI Options on GUIDE's Tools menu. Choose "GUI allows only one

% instance to run (singleton)".

%

% See also: GUIDE, GUIDATA, GUIHANDLES

% Edit the above text to modify the response to help GUI

% Last Modified by GUIDE v2.5 22-Jan-2021 18:17:27

% Begin initialization code - DO NOT EDIT

gui\_Singleton = 1;

gui\_State = struct('gui\_Name', mfilename, ...

'gui\_Singleton', gui\_Singleton, ...

'gui\_OpeningFcn', @GUI\_OpeningFcn, ...

'gui\_OutputFcn', @GUI\_OutputFcn, ...

'gui\_LayoutFcn', [] , ...

'gui\_Callback', []);

ifnargin&&ischar(varargin{1})

gui\_State.gui\_Callback = str2func(varargin{1});

end

ifnargout

[varargout{1:nargout}] = gui\_mainfcn(gui\_State, varargin{:});

else

gui\_mainfcn(gui\_State, varargin{:});

end

% End initialization code - DO NOT EDIT

% --- Executes just before GUI is made visible.

functionGUI\_OpeningFcn(hObject, eventdata, handles, varargin)

% This function has no output args, see OutputFcn.

% hObject handle to figure

% eventdata reserved - to be defined in a future version of MATLAB

% handles structure with handles and user data (see GUIDATA)

% varargin command line arguments to GUI (see VARARGIN)

% Choose default command line output for GUI

handles.output = hObject;

% Update handles structure

guidata(hObject, handles);

% UIWAIT makes GUI wait for user response (see UIRESUME)

% uiwait(handles.figure1);

% --- Outputs from this function are returned to the command line.

functionvarargout = GUI\_OutputFcn(hObject, eventdata, handles)

% varargout cell array for returning output args (see VARARGOUT);

% hObject handle to figure

% eventdata reserved - to be defined in a future version of MATLAB

% handles structure with handles and user data (see GUIDATA)

% Get default command line output from handles structure

varargout{1} = handles.output;

% --- Executes on button press in pushbutton1.

function pushbutton1\_Callback(~, eventdata, handles)

% hObject handle to pushbutton1 (see GCBO)

% eventdata reserved - to be defined in a future version of MATLAB

% handles structure with handles and user data (see GUIDATA)

global img1 img2

[path,nofile]=imgetfile();

ifnofile

msgbox(sprintf('No image found'),'Error','Error');

return

end

img1=imread(path);

img1=im2double(img1);

img2=img1;

axes(handles.axes1);

imshow(img1)

title('\fontsize{20}\color[rgb]{0.996,0.592,0.0} Brain MRI')

% --- Executes on button press in pushbutton2.

function pushbutton2\_Callback(hObject, eventdata, handles)

% hObject handle to pushbutton2 (see GCBO)

% eventdata reserved - to be defined in a future version of MATLAB

% handles structure with handles and user data (see GUIDATA)

global img1

axes(handles.axes2);

bw=im2bw(img1,0.7);

label=bwlabel(bw);

stats=regionprops(label,'Solidity','Area');

density=[stats.Solidity];

area=[stats.Area];

high\_dense\_area=density>0.5;

max\_area=max(area(high\_dense\_area));

tumor\_label=find(area==max\_area);

tumor=ismember(label,tumor\_label);

se=strel('square',5);

tumor=imdilate(tumor,se);

ifmax\_area>200

Bound=bwboundaries(tumor,'noholes');

imshow(img1);

holdon

fori=1:length(Bound)

plot(Bound{i} (:,2),Bound{i} (:,1),'y','linewidth',1.75)

end

title('\fontsize{20}\color[rgb]{0.996,0.592,0.0} Tumor detected');

holdoff

axes(handles.axes)

else

h=msgbox('No Tumor!!','Result');

return

end

**ANISOTROPIC DIFFUSION**

clc

closeall

clearall

%% input

[I, path] = uigetfile('\*.jpg','select an input image');

str=strcat(path,I);

s=imread(str);

figure;

imshow(s);

title('Input image', 'FontSize',20);

%% Filter

num\_iter = 10;

delta\_t=1/7;

kappa=15;

option=2;

disp('Preprocessing image kindly wait ...');

inp=anisodiff(s,num\_iter,delta\_t,kappa,option);

inp=uint8(inp);

inp=imresize(inp,[256,256]);

if size(inp,3)>1

inp=rgb2gray(inp);

end

figure;

imshow(inp);

title('Filtered Image','FontSize',20);

%%threshold

sout=imresize(inp,[256,256]);

t0=60;

th=t0+((max(inp(:))+min(inp(:)))./2);

fori=1:1:size(inp,1)

for j=1:1:size(inp,2)

ifinp(i,j)>th

sout(i,j)=1;

else

sout(i,j)=0;

end

end

end

%% Morphological Operation

label=bwlabel(sout);

stats=regionprops(logical(sout),'Solidity','Area','BoundingBox');

density=[stats.Solidity];

area=[stats.Area];

high\_dense\_area=density>0.5;

max\_area=max(area(high\_dense\_area));

tumor\_label=find(area==max\_area);

tumor=ismember(label,tumor\_label);

ifmax\_area>200

figure;

imshow(tumor)

title('tumor alone','FontSize',20);

else

h=msgbox('No Tumor !!','Result');

%disp('no tumor');

return;

end

%%Bounding box

box=stats(tumor\_label);

wantedBox=box.BoundingBox;

figure

imshow(inp);

title('Bounding Box','FontSize',20);

holdon;

rectangle('Position',wantedBox,'EdgeColor','y');

holdoff;

%% Getting Tumor Outline - image filling, eroding, subtracting

%erosion the walls by a few pixels

dilationAmount=5;

rad=floor(dilationAmount);

[r,c]=size(tumor);

filledImage=imfill(tumor,'holes');

fori=1:r

for j=1:c

x1=i-rad;

x2=i+rad;

y1=j-rad;

y2=j+rad;

if x1<1

x1=1;

end

if x2>r

x2=r;

end

if y1<1

y1=1

end

if y2>c

y2=c;

end

erodedImage(i,j)=min(min(filledImage(x1:x2,y1:y2)));

end

end

figure

imshow(erodedImage);

title('Eroded image','FontSize',20);

%% subtracting eroded image from original bw image

tumorOutline=tumor;

tumorOutline(erodedImage)=0;

figure;

imshow(tumorOutline);

title('Tumor Outline','FontSize',20);

%% Inserting the outline in filtered image in green color

rgb=inp(:,:,[1 1 1]);

red=rgb(:,:,1);

red(tumorOutline)=255;

green=rgb(:,:,2);

green(tumorOutline)=0;

blue=rgb(:,:,3);

blue(tumorOutline)=0;

tumorOutlineInserted(:,:,1)=red;

tumorOutlineInserted(:,:,2)=green;

tumorOutlineInserted(:,:,3)=blue;

figure

imshow(tumorOutlineInserted);

title('Detected Tumor','FontSIze',20);

%% Display together

figure

subplot(231);imshow(s);title('Input Image','FontSize',14);

subplot(232);imshow(inp);title('Filtered Image','FontSize',14);

subplot(233);imshow(inp);title('Bounding Box','FontSize',14);

holdon;rectangle('Position',wantedBox,'EdgeColor','y');hold off;

subplot(234);imshow(tumor);title('Tumor Alone','FontSize',14);

subplot(235);imshow(tumorOutline);title('Tumor Outline','FontSize',14);

subplot(236);imshow(tumorOutlineInserted);title('Detected Tumor','FontSize',14);

ANISODIFF

functiondiff\_im = anisodiff(im, num\_iter, delta\_t, kappa, foption)

fprintf('Removing noise\n');

fprintf('Filtering Completed !!');

% Convert input image to double.

im = double(im);

% PDE (partial differential equation) initial condition.

diff\_im = im;

% Center pixel distances.

dx = 1;

dy = 1;

dd = sqrt(2);

% 2D convolution masks - finite differences.

hN = [0 1 0; 0 -1 0; 0 0 0];

hS = [0 0 0; 0 -1 0; 0 1 0];

hE = [0 0 0; 0 -1 1; 0 0 0];

hW = [0 0 0; 1 -1 0; 0 0 0];

hNE = [0 0 1; 0 -1 0; 0 0 0];

hSE = [0 0 0; 0 -1 0; 0 0 1];

hSW = [0 0 0; 0 -1 0; 1 0 0];

hNW = [1 0 0; 0 -1 0; 0 0 0];

% Anisotropic diffusion.

for t = 1:num\_iter

% Finite differences. [imfilter(.,.,'conv') can be replaced by conv2(.,.,'same')]

nablaN = imfilter(diff\_im,hN,'conv');

nablaS = imfilter(diff\_im,hS,'conv');

nablaW = imfilter(diff\_im,hW,'conv');

nablaE = imfilter(diff\_im,hE,'conv');

nablaNE = imfilter(diff\_im,hNE,'conv');

nablaSE = imfilter(diff\_im,hSE,'conv');

nablaSW = imfilter(diff\_im,hSW,'conv');

nablaNW = imfilter(diff\_im,hNW,'conv');

% Diffusion function.

iffoption == 1

cN = exp(-(nablaN/kappa).^2);

cS = exp(-(nablaS/kappa).^2);

cW = exp(-(nablaW/kappa).^2);

cE = exp(-(nablaE/kappa).^2);

cNE = exp(-(nablaNE/kappa).^2);

cSE = exp(-(nablaSE/kappa).^2);

cSW = exp(-(nablaSW/kappa).^2);

cNW = exp(-(nablaNW/kappa).^2);

elseiffoption == 2

cN = 1./(1 + (nablaN/kappa).^2);

cS = 1./(1 + (nablaS/kappa).^2);

cW = 1./(1 + (nablaW/kappa).^2);

cE = 1./(1 + (nablaE/kappa).^2);

cNE = 1./(1 + (nablaNE/kappa).^2);

cSE = 1./(1 + (nablaSE/kappa).^2);

cSW = 1./(1 + (nablaSW/kappa).^2);

cNW = 1./(1 + (nablaNW/kappa).^2);

end

% Discrete PDE solution.

diff\_im = diff\_im+ ...

delta\_t\*(...

(1/(dy^2))\*cN.\*nablaN + (1/(dy^2))\*cS.\*nablaS + ...

(1/(dx^2))\*cW.\*nablaW + (1/(dx^2))\*cE.\*nablaE + ...

(1/(dd^2))\*cNE.\*nablaNE + (1/(dd^2))\*cSE.\*nablaSE + ...

(1/(dd^2))\*cSW.\*nablaSW + (1/(dd^2))\*cNW.\*nablaNW );

end

**USING SUBPLOTS**

Close all;

Clear all;

clc;

global img1 img2

[path, nofile]=imgetfile();

Ifnofile

msgbox(sprintf('Image not found !!!'),'Error','warning');

return

end

img1=imread(path);

img1=im2double(img1);

img2=img1;

bw=im2bw(img1,0.7);

label=bwlabel(bw);

stats=regionprops(label,'Solidity','Area');

density=[stats.Solidity];

area=[stats.Area];

high\_dense\_area=density>0.5;

max\_area=max(area(high\_dense\_area));

tumor\_label=find(area==max\_area);

tumor=ismember(label,tumor\_label);

se=strel('square',5);

tumor=imdilate(tumor,se);

ifmax\_area>200

figure(2);

subplot(1,3,1);

imshow(img1,[]);

title('Brain');

else

h=msgbox('No Tumor!!','Result');

return

end

subplot(1,3,2);

imshow(tumor,[]);

title('Tumor Alone');

[B,L]=bwboundaries(tumor,'noholes');

subplot(1,3,3);

imshow(img1,[]);

holdon

fori=1:length(B)

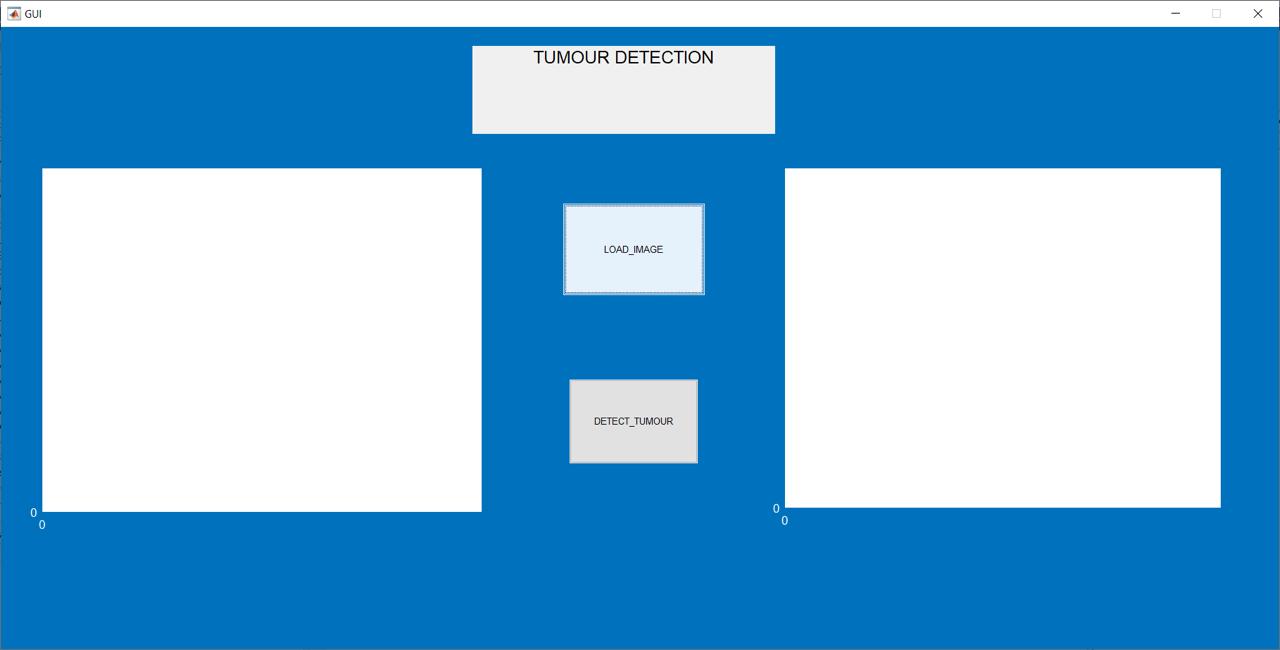
plot(B{i}(:,2),B{i}(:,1), 'y' ,'linewidth',1.45);

end

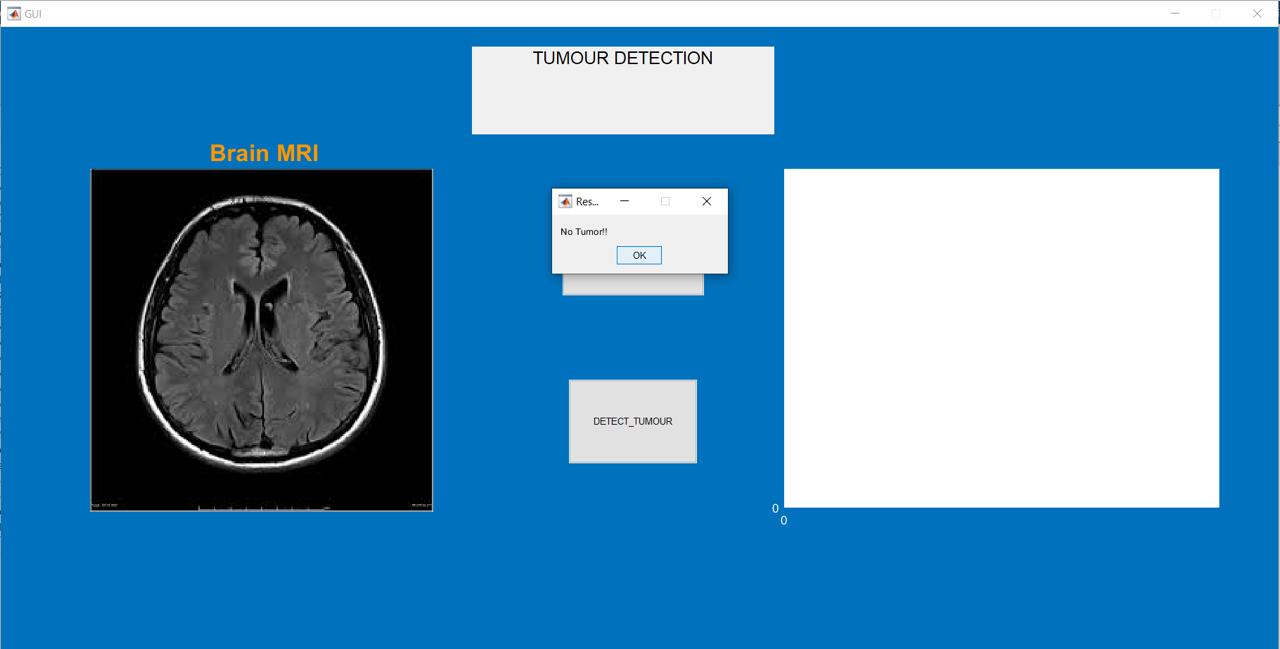
title('Detected Tumor');

holdoff;

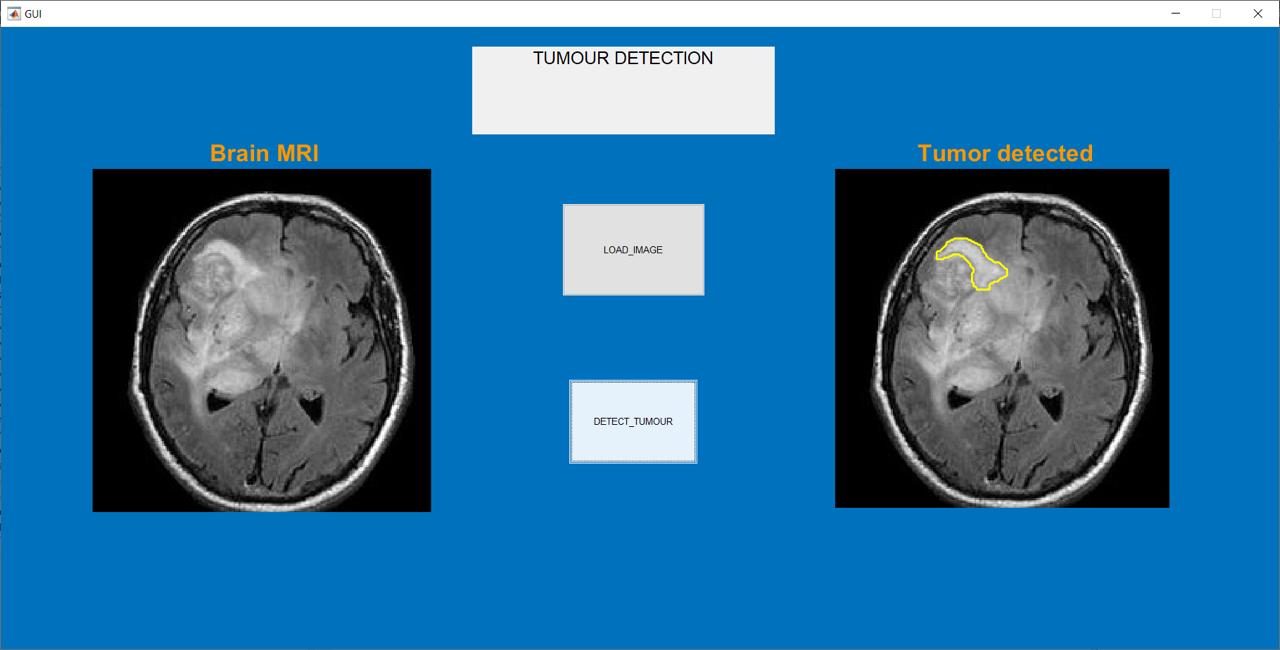
**OUTPUT:**



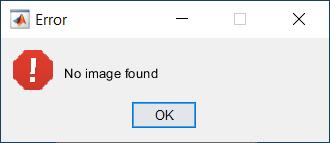
GUI for Brain Tumor Detection



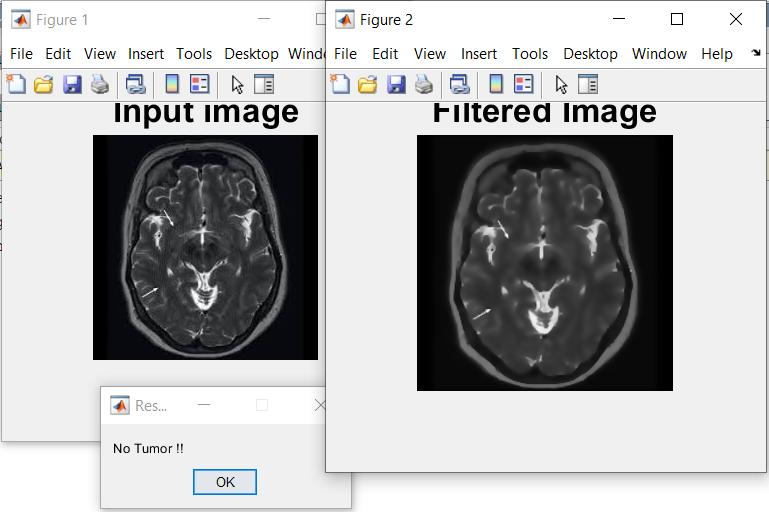
GUI for Brain Tumor Detection displaying “No Tumor!!”

****

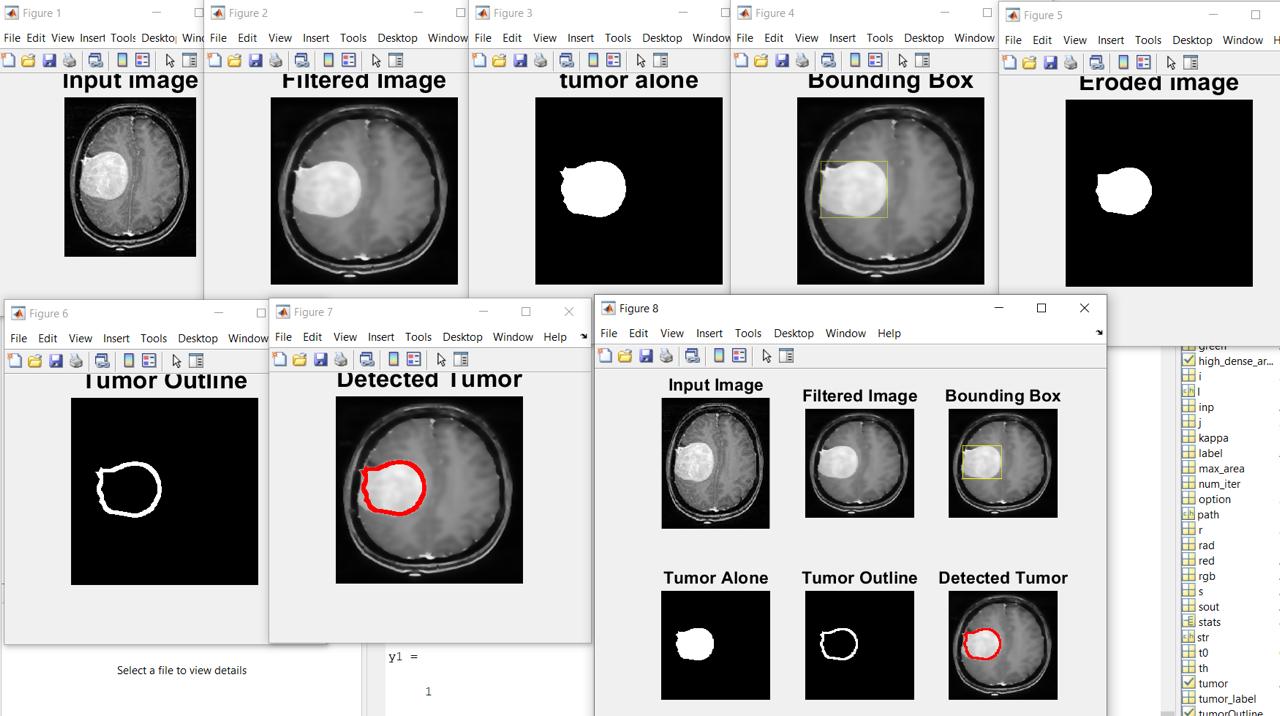
GUI for Brain Tumor Detection displaying “Tumor detected” with the resulting image



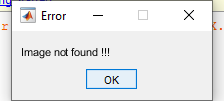
GUI for Brain Tumor Detection displaying “No image found”



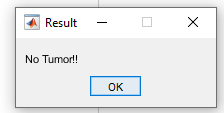
Brain Tumor Detection with Anisotropic Diffusion displaying “No Tumor!!”



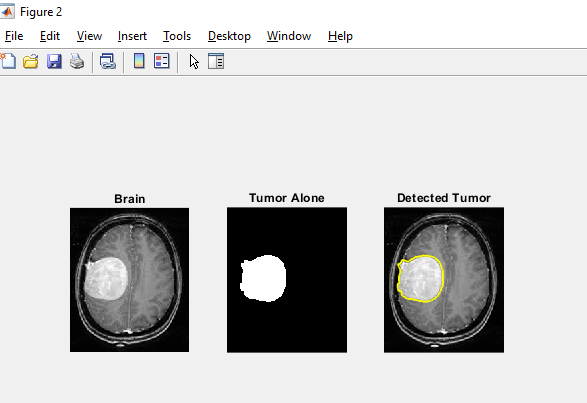
Brain Tumor Detection with Anisotropic Diffusion displaying “Detected Tumor” text and input image, filtered image, tumor alone image, Bounding Box image, Eroded image, Tumor Outline image and Detected Tumor image and the subplot showing each step of the process.



Brain Tumor Detection with subplots showing error when no image has been selected



Brain Tumor Detection with subplots showing “no Tumor!!!”



Brain Tumor Detection with subplots showing Tumor Alone image and Detected Tumor image

**CONCLUSION:**

This team has developed a project for brain tumor detection from MRI using MATLAB software. A selection of the best filter and methods sets our project apart compared to the standard algorithms. Image processing, a process where the image gets analyzed and processed intensively, is one of the significant points of the project. This project can make MRI image processing and tumor detection process faster and cheaper, thus offering an optimal solution for tumor detection, using Anisotropic Diffusion Filter and the morphological operators. As result, we offer modern medicine a glimpse of the future of tumor detections and the greater role of AI in health and wellness sectors.

**REFERENCES:**

1)mathworks.in

2)geeksforgeeks.org

3)[ece.northwestern.edu](http://www.ece.northwestern.edu)

4)webopedia.com

5)Github.com

6)Wikipedia

7)sciencedirect.com

8)himnickson.medium.com

9)datacarpentry.org

10)stackoverflow.com