IDENTIFYING BREAST CANCER IN MAMMOGRAM IMAGES USING IMAGE PROCESSING TECHNIQUES

A PROJECT REPORT PRESENTED BY

M. M. D. SAMARANAYAKE
(S/16/533)

to the Board of Study in
DEPARTMENT OF STATISTICS AND COMPUTER SCIENCE

in partial fulfillment of the requirement
for the award of the degree of

B.Sc. (Honours) in Computer Science

of the

UNIVERSITY OF PERADENIYA
SRI LANKA
2020

DECLARATION

I hereby declare that I carried out the work reported in this project report. It describes my own independent research results except where due references have been made in the text. No part of this project report/thesis has been submitted earlier or concurrently for the same or any other degree.

18.11.2020 ……

Date

……

Signature of the Candidate

ABSTRACT

Breast cancer is the second common cancer worldwide after lung cancer, the fifth common cause of cancer death, and the primary cause of cancer death in women. There are about 100 times more new cases of breast cancer in women than in men every year. Doctors know that breast cancer happens when some breast cells begin to grow abnormally. These cells divide more rapidly than healthy cells do and keep on gathering, forming a lump or mass. Cells may spread (metastasize) through breasts to lymph nodes or other body parts. Breast cancer mostly begins with cells in the milk-producing ducts (invasive ductal carcinoma). Breast cancer may also start in the glandular tissue called lobules (invasive lobular carcinoma) or in other cells or tissue within the breast. Researchers have identified hormonal, lifestyle, and environmental factors that may increase the risk of breast cancer. But it's not clear why some people who have no risk factors develop cancer, yet other people with risk factors never do. Breast cancer is likely caused by a complex interaction of the genetic makeup and the environment which people live. Beginning in the 1980s, screening mammography led to sweeping improvements in the early detection of breast cancer. Mammography is a specialized medical imaging that uses a low-dose x-ray system for scanning the breasts. A mammography exam helps in the early discovery and diagnosis of breast cancer in women. Mammogram image segmentation is useful in detecting the breast cancer regions (tumors/lumps); hence, better diagnosis.

The common difficulty in identifying breast cancer using mammogram images is finding the exact location of the tumor and calculating its diameter. It takes great experience for a doctor to identify the exact location of a tumor where the doctor may misjudge sometimes. I have created a graphical user interface using tkinter and OpenCV to solve this problem. The programming language used for this purpose is Python. In this project, I've applied enhanced thresholding based approach for Mammograms' image enhancement. Moreover, here I've used morphological operations in order to enhance and identify the location of the tumor from the mammogram image helping physicians to easily detect the breast cancer into different Mammograms. The result is enhanced wise effect on breast cancer qualitative detection into Mammograms, helping physicians better diagnose. Moreover, I've used contours to identify the area and calculate the diameter of actual size mammogram images, finally stating the tumor stage corresponding to the TNM staging method used by Doctors [8]. This manual thresholding and morphological operations have the advantage of not only reducing processing time.

ACKNOWLEDGEMENT

I express my sincere thanks to the consultants and facilitators of the Department of Computer Science and Statistics of the University of Peradeniya for providing us the opportunity to experience and research in OpenCV with Python and to successfully complete a project.

It is with great pleasure I acknowledge Mrs. Amirthavarshini Mahadevan and Mr. Lakshitha Amarasinghe for the guidance, constructive feedback, and supportive lectures. I would like to extend my sincere gratitude to Dr. Ruwan Nawarathna, the Head of the Department of Statistics and Computer Science, for providing the facilities for the completion of this project successfully.

I would be remiss if I didn't acknowledge the valuable sacrifices made by my parents in order to complete my project and for their encouragements and blessings.

My sincere gratitude goes to the Department of Statistics and Computer Science for giving me this opportunity to reach into the depths of image processing in the field of computer science. It has been a great achievement for me and my career ahead.

Finally, thank you, my fellow colleagues for helping me throughout the project, sharing ideas and the support rendered in developing this project, and all those who assisted me in various ways but names have been unmentioned.

TABLE OF CONTENTS

CHAPTER 1 1

INTRODUCTION 1
1.1 Problem Statement 1
1.2 Objective 2
1.3 Approach 2

CHAPTER 2 3

LITERATURE REVIEW 3

CHAPTER 3 5

METHODOLOGY 5
3.1 Image acquisition and GUI creation 6
3.1.1 Image Acquisition 6
3.1.2 Setting up the OpenCV environment 8
3.1.3 Browsing and Resizing the Image 9
3.2 Pre-Processing 10
3.2.1 Gray Scaling 10
3.2.2 Thresholding the image 11
3.2.3 Using morphological transformations 12
3.2.4 Cropping the image 12
3.3 Image Segmentation 14
3.3.1 Finding Contours around the tumors 14
3.3.2 Draw contours around the tumor found 14
3.3.3 Find circles on the drawn contours and get the diameters 14
3.3.4 Calculating the diameters and the areas 15
3.4 Finding Tumor stage using TNM staging 17
3.4.1 TNM staging 17
3.4.2 Finding the stage of tumor using the diameters 20
3.5 Testing the developed procedure for other mammograms 21

CHAPTER 4 23

RESULTS AND DISCUSSION 23
4.1 Results and Discussion Overview 23
4.2 Complete Code of the Project 24
4.2.1 Browsing and resizing the image 24
4.2.2 Image Enhancement 25
4.2.3 Image Segmentation 25
4.2.4 Tumor Staging 27
4.3 Assumptions 27

CHAPTER 5 28

CONCLUSION 28

REFERENCES 29

TABLE OF FIGURES

Figure 1 Steps 5
Figure 2: Graphical User Interface 6
Figure 3 : Mammogram of a breast with tumor 6
Figure 4: Mammogram with breast cancer tumor 7
Figure 5: Right breast with tumor 7
Figure 6: Breast with a minor tumor 8
Figure 7: Importing Libraries 8
Figure 8: Browsing and resizing the image 9
Figure 9 Storing image in the GUI Label 9
Figure 10: Browsed image displayed 9
Figure: 11 Normal breast mammogram image 10
Figure: 12 Breast with an abnormality of dense tissue predicted to be a tumor 10
Figure 13: Converting the color image to grayscale 10
Figure 14: Grayscale image of the selected mammogram 11
Figure 15: Thresholding the image 11
Figure 16: Thresholded grayscale image 11
Figure 17: Scale bars used in the GUI to control the kernel values 12
Figure 18: Getting the kernel values off the scale and morphological operations are done 12
Figure 19: Cropping the image 12
Figure 20: Storing image in the GUI Label 13
Figure 21: After Morphological transformation the cropped image is displayed 13
Figure 22 Finding contours 14
Figure 23: Drawing Contours 14
Figure 24: Drawing circles and populating the array 14
Figure 25: Storing image in the GUI Label 15
Figure 26: GUI representation of the segmented image in the 3rd panel 15
Figure 27: Diameter Calculation 15
Figure 28: Message displayed if there is no tumor found 16
Figure 29: Configuring the messages to the labels 16
Figure 30: GUI with the displayed diameter and area 16
Figure 31: TNM Staging 19
Figure 32: Finding tumor stage 20
Figure 33: Final results displayed 20
Figure 34: Mammogram with tiny development of a tumor 21
Figure 35: Mammogram with T2 stage tumor 21
Figure 36: Mammogram with a T3 stage tumor 21
Figure 37: Mammogram with T2 stage tumor 22
Figure 38: Mammogram with a T2 stage tumor 22
Figure 39: Mammogram with T2 stage tumor 22
Figure 40: breast with tumor (left) and the resultant tumor with contours (right) 23
Figure 41: breast with tumor (left) and the resultant tumor with contours (right) 23
Figure 42: Code with Browse function 24
Figure 43: Enhance Image function 25
Figure 44: First part of the segmentation function 25
Figure 45: ELSE part which contains the IP techniques 26
Figure 46: Code for tumor staging 27

LIST OF ABBREVIATIONS

ex : Example 11
GUI : Graphical User Interface 2

CHAPTER 1
INTRODUCTION

1.1 Problem Statement

Breast cancer is the most well-known cancer affecting women worldwide. Over recent decades, the incidence of breast cancer has risen globally. Studies have demonstrated that the most significant increment will be among women in developing countries, most of whom live in the Asian region. More than one diagnostic modality used for breast cancer screening, like computed tomography, resonance magnetic imaging, and ultrasound. All of these modalities aren't enough to detect all lesions of the breast in spite of their characteristics such as high sensitivity of soft tissue detection. Digital mammography is prevalent in detecting breast lesions than other diagnostic modalities.

Many types of techniques are used for mammography image detection such as edge-based transform, smoothing, noise removal, and extraction techniques. Therefore, it is necessary to have an efficient method to enhance and improve the mammography images to identify the tumor's size and location.

The methodology implemented in this project gives you the enhanced image with the tumor's location, and finally, the diameter, area and tumor stage will be displayed in the user interface. This enhancement will come in handy for the Doctors to identify and diagnose breast cancer and its stage.

1.2 Objective

This project's main objective is to locate and identify the tumor in breast cancer using mammogram images. Further, to develop a simple approximation method to find the diameter and the area of the tumor in the images.

Objectives Specified

- Preprocess and enhance the image
- Identify the location of the tumor
- Calculate the size of the tumor (Area Calculation)
- Determine the stage of the tumor in TNM staging [8]

1.3 Approach

- Acquire the necessary mammogram images needed.
- Creation of the Graphical User Interface.
- Browsing and loading the images to the GUI.
- Applying thresholding to get a separate tumor area from the image.
- Applying morphological operations and getting a clear segmentation of the tumor, reducing the noise around it.
- Finding contours around the segmented objects.
- Use contours to draw around the tumors to separate the biggest and the dense area of the tumor.
- Use circular contours on them to calculate the diameter of the tumor.
- Populating an array of all the diameters calculated and getting the highest value of it will give the diameter of the exact tumor.
- Extending the project to assess the tumor stage according to the TNM staging tool.

CHAPTER 2
LITERATURE REVIEW

The research works done in breast cancer detection which follows either image processing and image segmentation techniques used with machine learning algorithms on the extracted features to classify the images as breast cancerous or non-breast cancerous. Some of the existing breast cancer detection methods are described in this section.

A Survey paper on breast cancer diagnosis using image processing techniques by Mussarat Yasmin, Muhammad Sharif, and Sajjad Mohsin highlights the available breast cancer detection techniques based on image processing and provides an overview of the affordability, reliability, and outcomes of each technique [1].

Biomedical research on breast cancer detection using image enhancement and segmentation algorithms carried out by Yousif M.Y Abdallah shows the contrast improvement, noise lessening, texture scrutiny, and portioning algorithms which aim to augment and hone the image intensity and eliminate noise from the images. The assortment factor of augmentation depends on the backdrop tissues and type of breast lesions; hence, some lesions gave better improvement than the rest due to their density [2].

A research article on mammogram images thresholding for breast cancer detection using different thresholding methods by Moumena Al-Bayati and Ali El-Zaart concludes that Otsu thresholding method is the best in determination of tumor cells [3].

A research paper by D. Ponraj, M. Jenifer, P. Poongodi and J. Manoharan, "A Survey on the Preprocessing Techniques of Mammogram for the Detection of Breast Cancer," reviews existing approaches of preprocessing in mammographic images [4].

A review article by Abdullah-Al Nahid[1] and Yinan Kong "Involvement of Machine Learning for Breast Cancer Image Classification" concludes that machine learning techniques and tools have been a tremendous success so far, and this success has gained an extra impetus with the involvement of deep-learning techniques [9].

An article citation by M. Melloul and L. Joskowicz, "Segmentation of Micro Calcification in X-Ray Mammograms Using Entropy Thresholding," describes a new algorithm for micro calcification segmentation in mammographic x-ray images [10].

An article citation by A. Hoyer and W. Spiesberg, on "Computerized Mammogram Processing," takes an approach based on local adaptive threshold segmentation to extract contour and a new smoothing approach founded on Fourier descriptors are introduced [11].

A biomedical research carried out by T. Lau and W. Bischoff, on "Automated Detection of Breast Tumors Using the Asymmetry Approach," shows that this method can improve the sensitivity and reliability of systems for automated detection of breast tumors [12].

Another biomedical research by Yin and M. Giger on Computerized Detection of Masses in Digital Mammogram: Analysis of Bilateral Subtraction Images indicates that the scheme is potentially useful as an aid to radiologists in the interpretation of screening mammograms [13].

There are some sophisticated methodologies for finding shapes of different rice grains and classifying them using image processing and machine learning techniques. All of the above methods involve some sophisticated technologies in image processing.

CHAPTER 3
METHODOLOGY

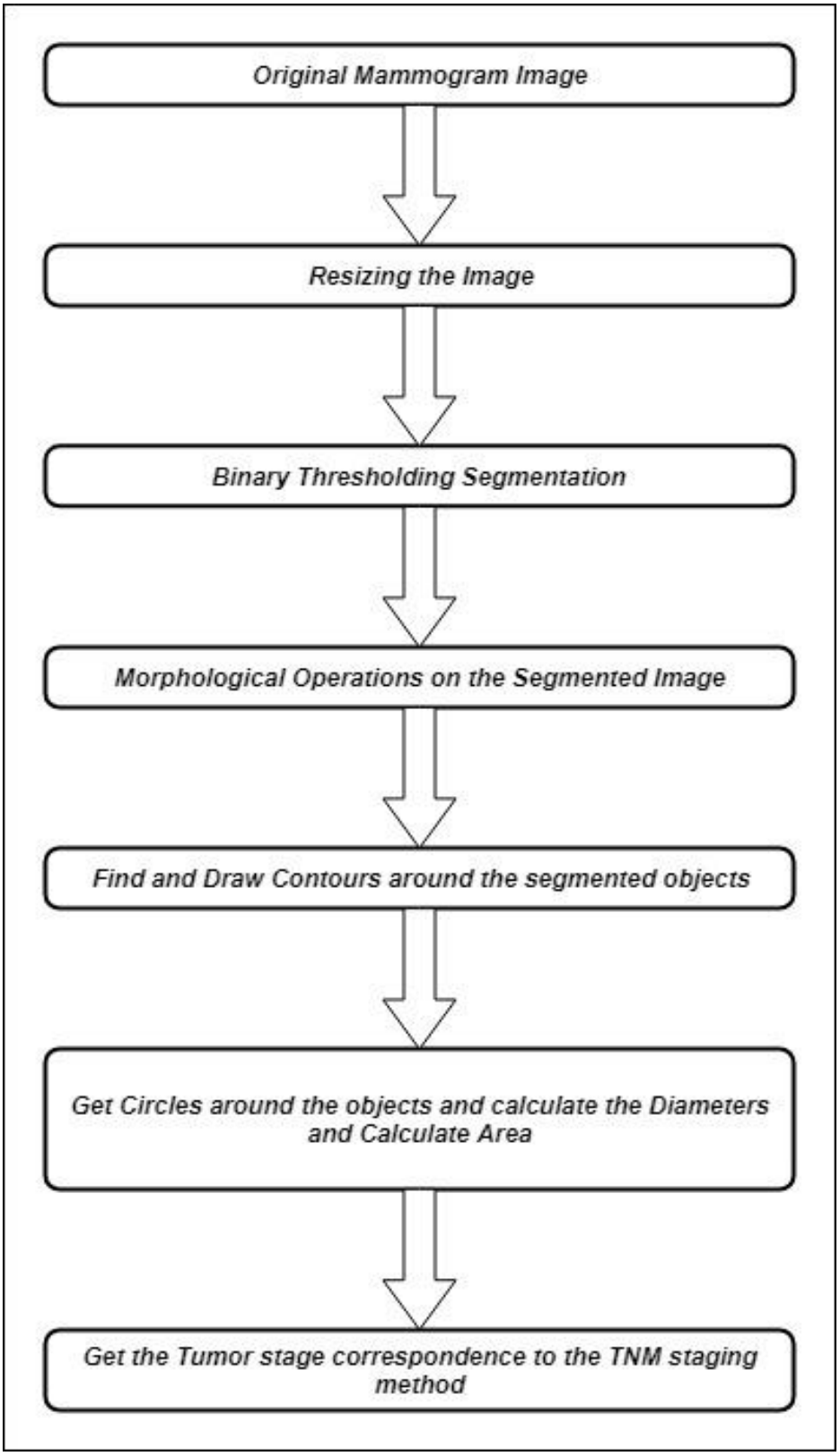

Figure 1 Steps

3.1 Image acquisition and GUI creation

3.1.1 Image Acquisition

For this project, I have used sample google images downloaded from the internet. Image to image, the color conditions vary, and thus thresholding values and kernels used for morphological transformations had to be changed. Since image processing has to be done for individual images separately, I created a GUI to control the kernel values for the morphological operations separately. Shown below is the GUI created and some of the images used in this project.

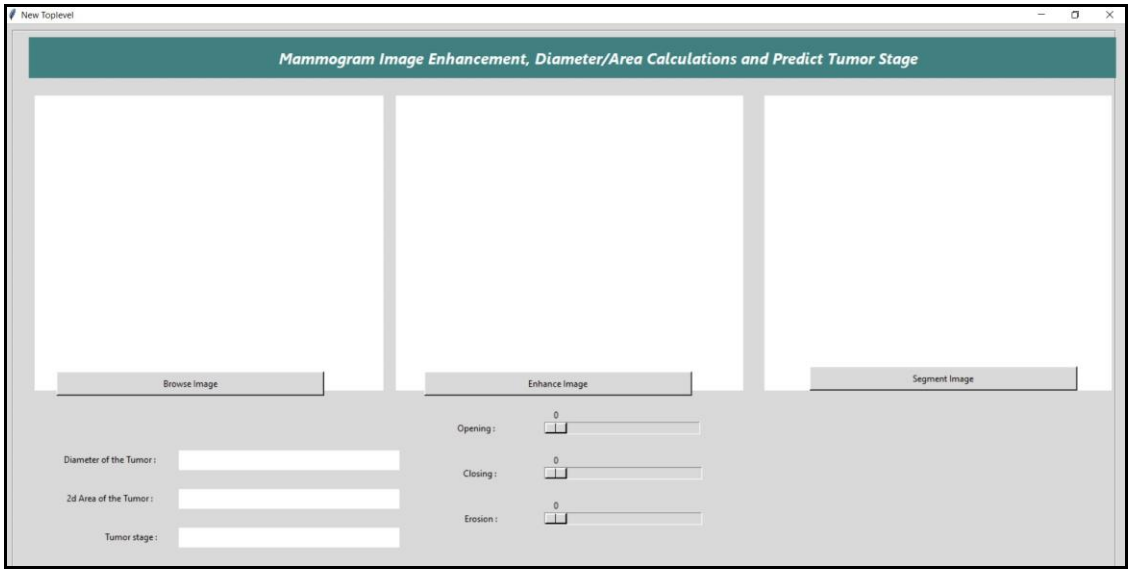

Figure 2: Graphical User Interface

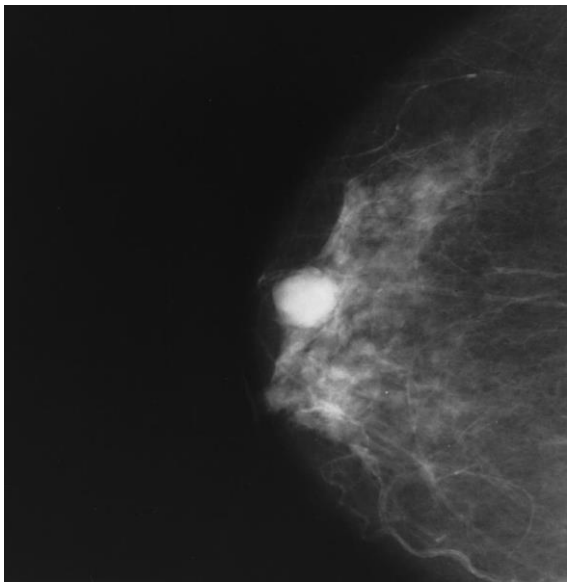

Figure 3 : Mammogram of a breast with tumor

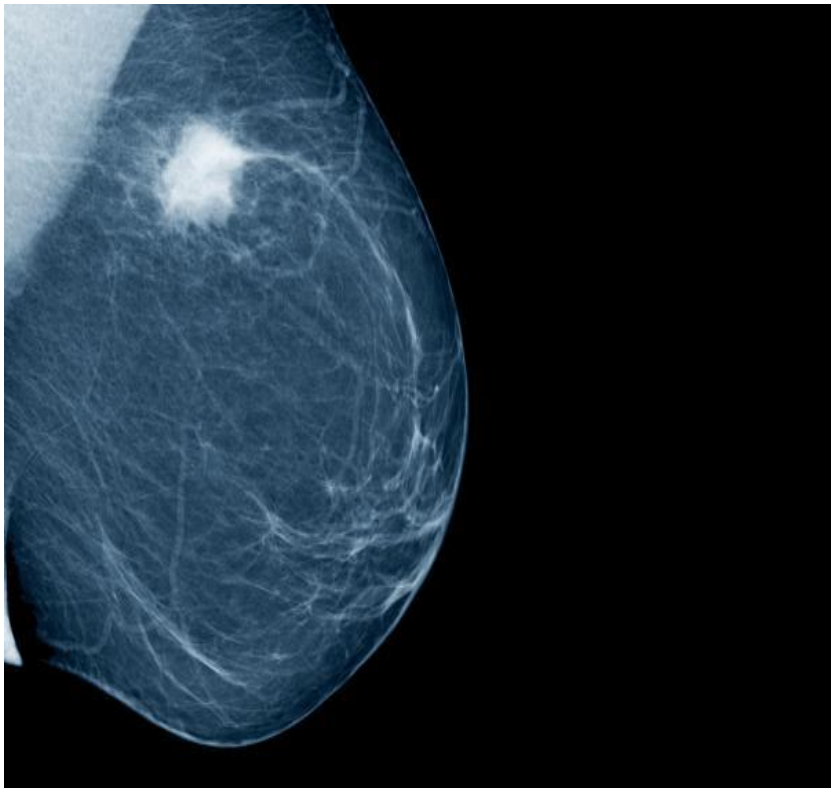

Figure 4: Mammogram with breast cancer tumor

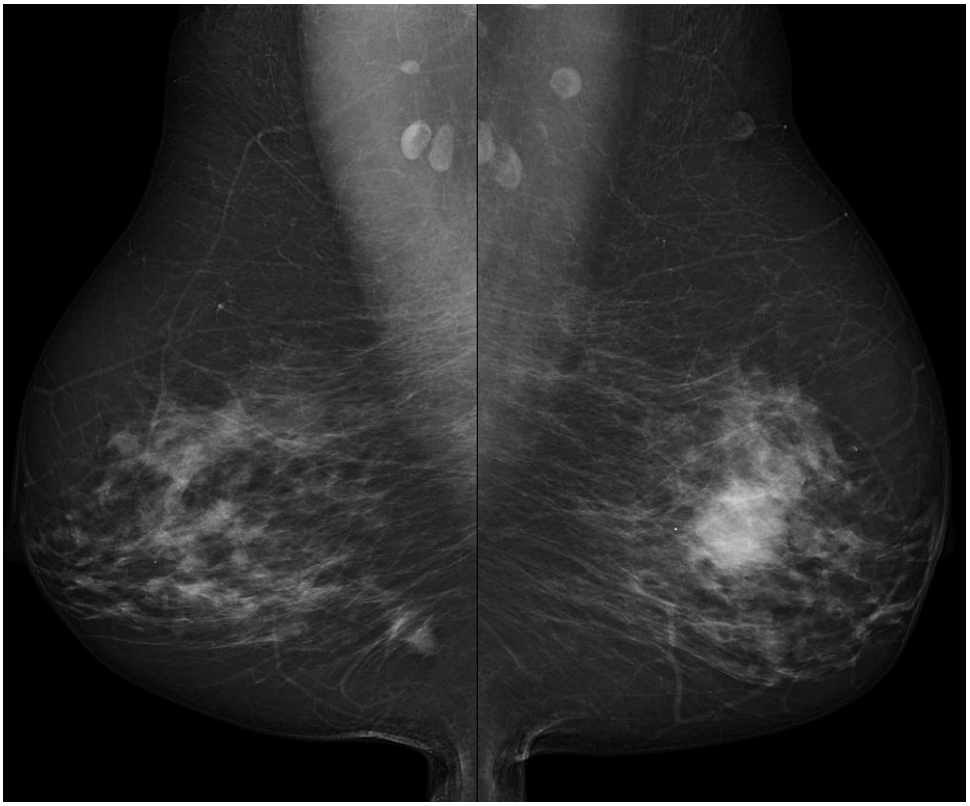

Figure 5: Right breast with tumor

Figure 6: Breast with a minor tumor

3.1.2 Setting up the OpenCV environment

Initially, OpenCV has to be installed on the PC to continue this project [6]. I have used Python as the language for development.

Following libraries were imported to run the project in Python.

- sys
- cv2
- numpy
- tkinter
- math
- pillow

```
import sys
import cv2
import numpy as np
import tkinter as tk
import tkinter.ttk as ttk
import math
from PIL import Image, ImageTk
from tkinter import filedialog
```

Figure 7: Importing Libraries

3.1.3 Browsing and Resizing the Image

Using the GUI, the image was selected using the browse image function. After importing the libraries, the image of interest was loaded as a color image. It was read as a color image because in the final output the contours were drawn in the original color image in black to give a better contrast and clear output. The image was resized to fit the label of the GUI, and the resized percentage is calculated to be used in the area calculation of the tumor at the end.

```
    # -------------------------------------Image browser function-----------------------------
def browse(self):
    img = Image.open(filedialog.askopenfilename())
    val = img.size
            # Actual size to resized image percentage calculation
    calc_area_percentage = ((477*404)/(val[0]*val[1]))*100

            # Resizing the image
    resized = img.resize((477,404), Image.ANTIALIAS)
```

Figure 8: Browsing and resizing the image

The image is converted to the ImageTk format in order to be stored in the display label of the GUI.

```
            # convert the image to ImageTk format
    imgtk = ImageTk.PhotoImage(image=resized)

            # storing image in the  browselabel
    self.browselabel.image = imgtk
    self.browselabel.configure(image=imgtk)
```

Figure 9 Storing image in the GUI Label

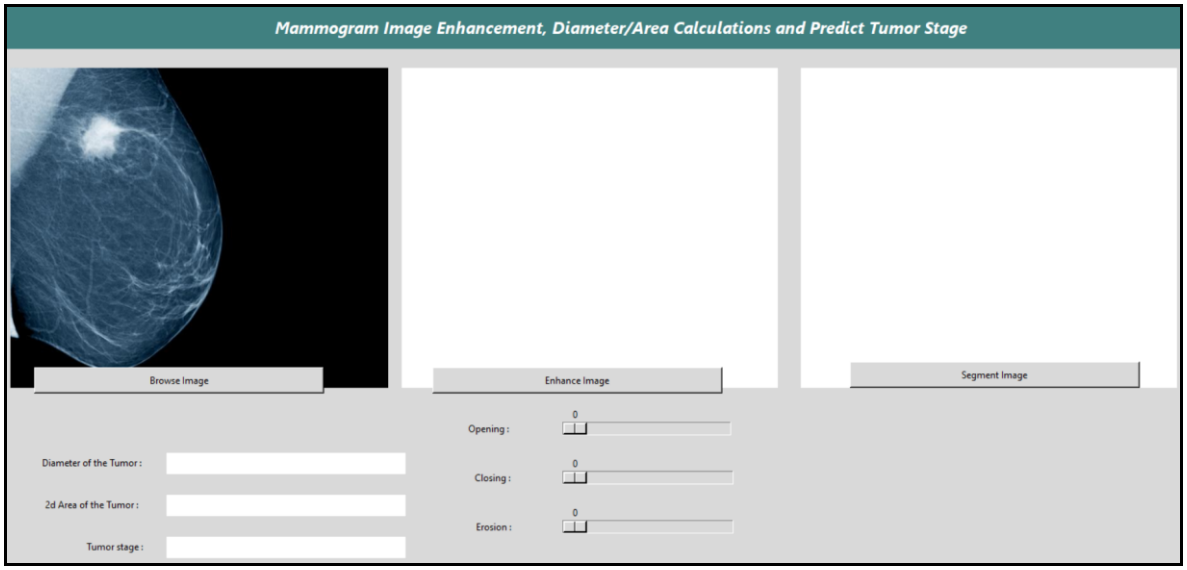

Figure 10: Browsed image displayed

3.2 Pre-Processing

Before detecting breast cancer tumors via image processing techniques, the browsed image needs to undergo several filtering to make the image easier to process and generate more accurate results.

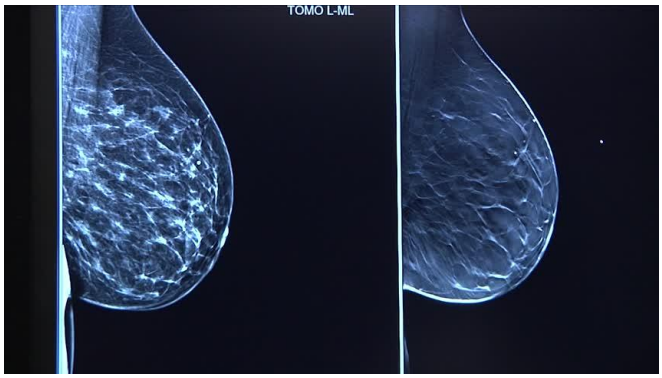

Figure: 11 Normal breast mammogram image

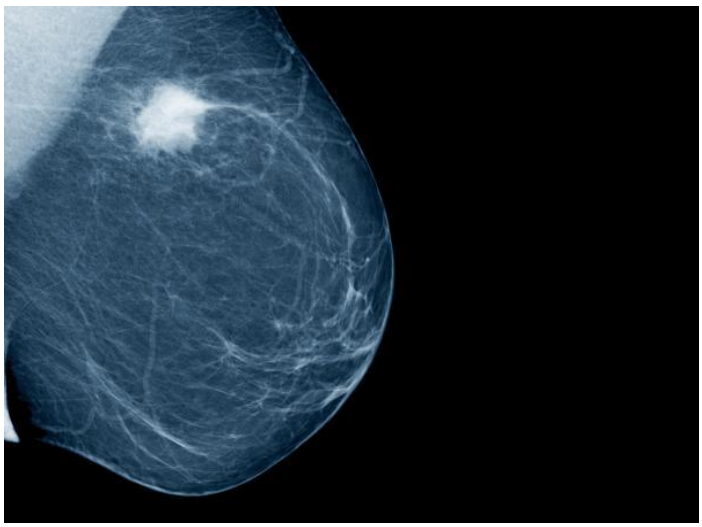

Figure: 12 Breast with an abnormality of dense tissue predicted to be a tumor

3.2.1 Gray Scaling

The image is converted to grayscale to reduce color complexity and noise. It is done by rejecting hue and saturation information and using the following equation.

$$(0.2989 * R) + (0.5870 * G) + (0.1140 * B)$$

```
    # ----------------------------------Image enhancement function--------------------
def enhance_img(self,img,area):
            # convert image to grayscale image
    resized_gray = cv2.cvtColor(np.float32(img), cv2.COLOR_BGR2GRAY)
```

Figure 13: Converting the color image to grayscale

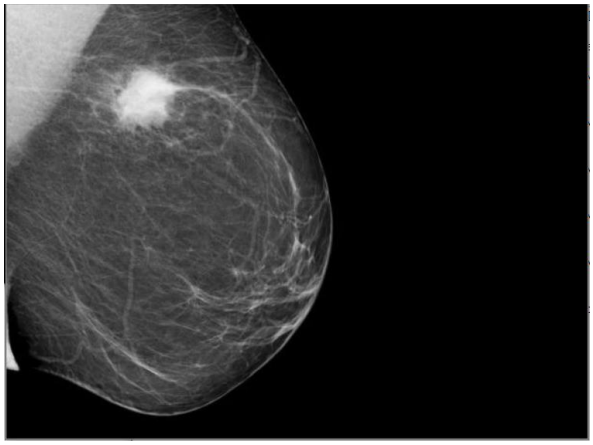

Figure 14: Grayscale image of the selected mammogram

3.2.2 Thresholding the image

When thresholding, if the pixel value is greater than the threshold value, that pixel is assigned a certain value (black and white if Binary thresholding is done and other colors for color images). Thresholding allows identifying the regions we are interested in [5]. For ex- the mammogram's denser area shows the tumor with breast cancer to be separated from the rest of the mammogram.

```
        # apply thresholding
thresh,img_bin = cv2.threshold(resized_gray,145,255,cv2.THRESH_BINARY)
```

Figure 15: Thresholding the image

Figure 16: Thresholded grayscale image

3.2.3 Using morphological transformations

Morphological transformations are done typically to binary images. For morphological operators, a structuring element or a kernel is needed. Based on our selected structuring element, the output image can vary [6]. I've created the GUI to adjust the kernel values by the user to get the expected output image without noise and a clear area of the tumor. In this I've used opening, closing, and erosion to get the expected output. Opening is the process of erosion followed by dilation which was helpful in noise removal. Closing is used in order to close small holes inside the foreground objects, or the small black points on it. I've considered erosion as an extra tool to get the best output image.

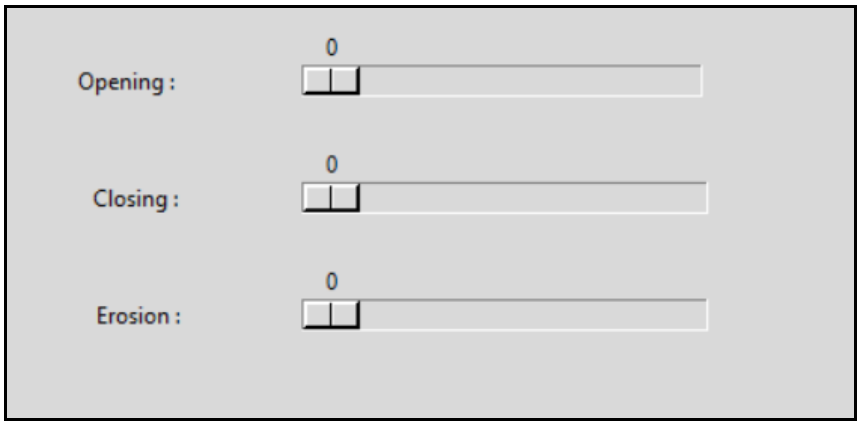

Figure 17: Scale bars used in the GUI to control the kernel values

```
        # getting opening, closing, erosion kernel values from the scale bars in the GUI
val1 = self.Scale1.get()
val2 = self.Scale2.get()
val3 = self.Scale3.get()

        # IMPLEMENTING KERNELS
kernel = np.ones((val1,val1), np.uint8)
kernel2 = np.ones((val2,val2), np.uint8)

        # APPLYING MORPHOLOGICAL OPERATIONS TO THE IMAGE
opening = cv2.morphologyEx(img_bin, cv2.MORPH_OPEN, kernel)
closing = cv2.morphologyEx(opening, cv2.MORPH_CLOSE, kernel2)
erosion = cv2.erode(closing,cv2.getStructuringElement(cv2.MORPH_ELLIPSE,(val3,val3)))
```

Figure 18: Getting the kernel values off the scale and morphological operations are done

3.2.4 Cropping the image

The image after the morphological operations is cropped in order to get only the tumor part without other tissue areas

```
        # CROP IMAGE TO EXTRACT THE TUMOR PART
crop_img = erosion[50:400, 60:477]
```

Figure 19: Cropping the image

The image is converted to the ImageTk format to be stored in the GUI display label.

```
        # convert the image to ImageTk format
image1 = Image.fromarray(crop_img)
imgtk = ImageTk.PhotoImage(image=image1)

        # storing image in the  enhanceimage label
self.enhanceimage.image = imgtk
self.enhanceimage.configure(image=imgtk)
```

Figure 20: Storing image in the GUI Label

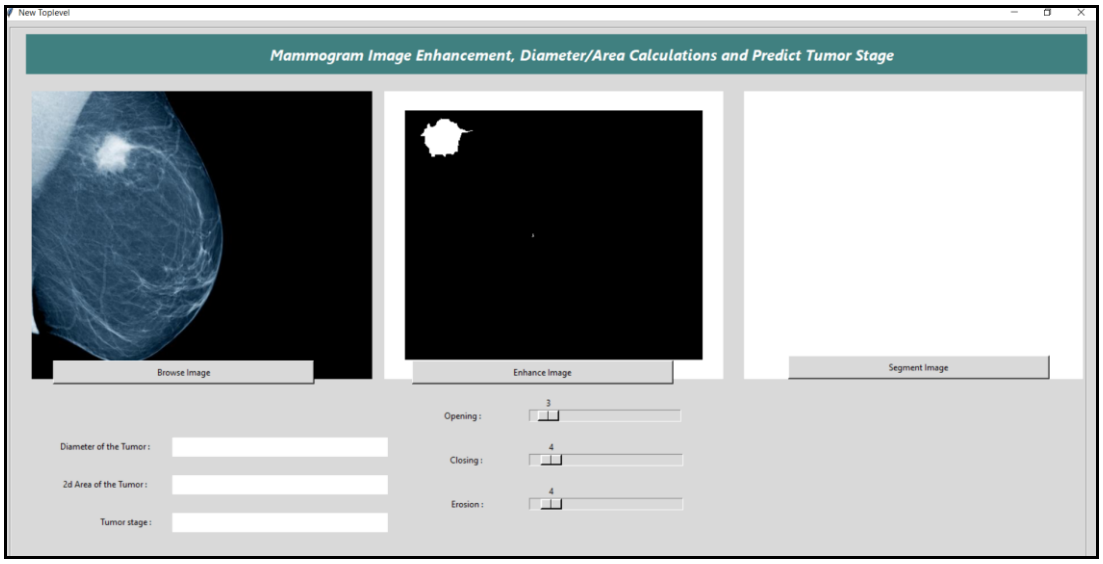

Figure 21: After Morphological transformation the cropped image is displayed

3.3 Image Segmentation

3.3.1 Finding Contours around the tumors

Contours can be explained simply as a curve joining all the continuous points (along the boundary), having the same color or intensity. The contours are a useful tool for shape analysis and object detection and recognition [7].

```
# find contours
contours,hierarchy = cv2.findContours(thresh,2,1)  # include another variable before 'contuors' if you get an error saying
                                                   # "ValueError: not enough values to unpack (expected 3, got 2)"
```

Figure 22 Finding contours

3.3.2 Draw contours around the tumor found

Contours are drawn around the foreground objects to pinpoint the tumor accurately.

```
cnt = contours
                # draw contours on the image
cv2.drawContours(thresh, cnt, -1, (36, 255, 12), 2)
```

Figure 23: Drawing Contours

3.3.3 Find circles on the drawn contours and get the diameters

Around the drawn contours, circles are drawn in order to get the diameters of the tumors. So the circles are drawn using a for loop and is used to populate an array with the diameters. Since a single pixel is equal to 0.26458333mm the radiuses populated will be multiplied by this value to get the radius in millimeters.

```
    arr = [0 for i in cnt]

                # finding the radius of the segmented objects
    for i in range (len(cnt)):
        (x,y),radius = cv2.minEnclosingCircle(cnt[i])
        center = (int(x),int(y))
        radius = int(radius)
        cv2.circle(thresh,center,radius,(0,255,0),2)
                # populating the diameters of the objects to arr
        arr[i] = radius*0.26458333
```

Figure 24: Drawing circles and populating the array

The image is converted to the ImageTk format to be stored in the GUI display label.

```
# convert the image to ImageTk format
image1 = Image.fromarray(thresh)
imgtk = ImageTk.PhotoImage(image=image1)
            # storing image in the  segImg label
self.segImg.image = imgtk
self.segImg.configure(image=imgtk)
```

Figure 25: Storing image in the GUI Label

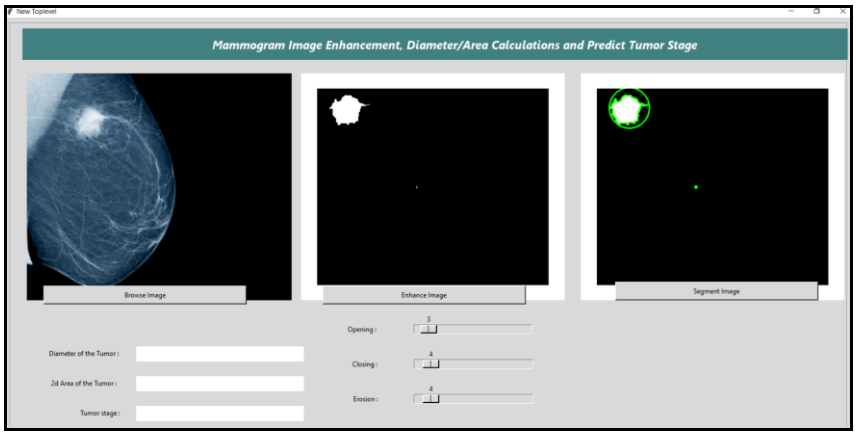

Figure 26: GUI representation of the segmented image in the 3rd panel

3.3.4 Calculating the diameters and the areas

The diameter is calculated using the radius of the circles drawn on the tumors. The populated array is sorted in descending order. The tumor will be the largest object found among the foreground objects after preprocessing. So this will be the tumor we need for our calculations. The first element is taken from the array since it will be the highest value (radius) among the drawn contours. First, the area is calculated for the cropped image using the area formulae for circles.

$$\text{Area of a Circle } = \pi r^2 \text{ (r = radius of the circles drawn on contours)}$$

In order to calculate the actual diameter of the original image, the area calculated will be multiplied by the inverse of the resized percentage calculated at [3.3.1] section. And then, the diameter is calculated by the radius taken from the actual area.

```
            # sorting the array in descending order
arr.sort(reverse=True)

            # find the area of the biggest tumor in the image
area_circle = math.pi * (arr[0]*arr[0])
            # finding the area for the actual sized image
actual_circle_area = area_circle*(100/area)
            # calculating the diameter from the actual circle area
diameter_result1 = round(math.sqrt((actual_circle_area/math.pi))*2, ndigits=2)

diameter_result2 = "(Actual image) Diameter of the biggest tumor is " + str(diameter_result1)  + "mm"
```

Figure 27: Diameter Calculation

Diameter and the areas calculated will be displayed in the GUI. If there was no tumor found a message will be displayed. If the tumor is found, the other part of the code will be executed.

```
if np.mean(thresh) == 0:
                # convert the image to ImageTk format
    image1 = Image.fromarray(thresh)
    imgtk = ImageTk.PhotoImage(image=image1)

                # storing image in the  segImg label
    self.segImg.image = imgtk
    self.segImg.configure(image=imgtk)

                # return messages to the following labels
    self.diameter.configure(text="No Tumor Detected")
    self.area.configure(text="No Tumor Detected")
    self.tumorstage.configure(text="No Tumor Detected")
```

Figure 28: Message displayed if there is no tumor found

The results will be displayed by configuring the labels with the messages.

```
            # storing the diameter in the diameter label
self.diameter.configure(text=diameter_result2)

result = round(actual_circle_area, ndigits=2)
resultant_area = "(Actual image) Area of the biggest tumor is " + str(result) + "mm"

            # storing the area of the tumor in the area label
self.area.configure(text=resultant_area)
```

Figure 29: Configuring the messages to the labels

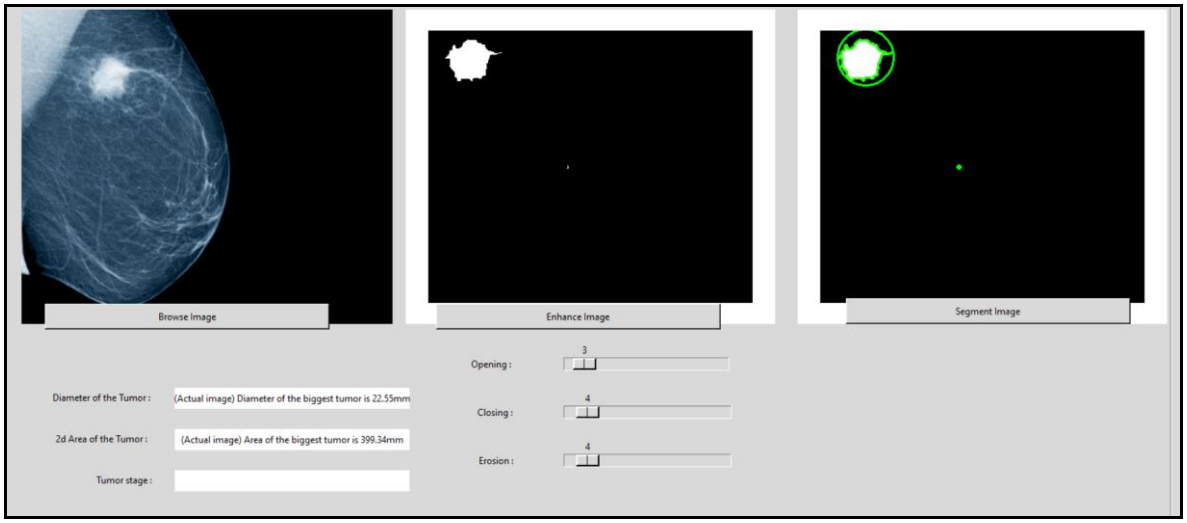

Figure 30: GUI with the displayed diameter and area

3.4 Finding Tumor stage using the TNM staging

3.4.1 TNM staging

The most common tool that doctors use to describe the stage is the TNM system [8]. Doctors use the results from diagnostic tests and scans to answer these questions:

- Tumor (T): How large is the primary tumor in the breast? What are its biomarkers?
- Node (N): Has the tumor spread to the lymph nodes? If so, where, what size, and how many?
- Metastasis (M): Has the cancer spread to other parts of the body?

The results are combined to determine the stage of cancer for each person.

There are 5 stages of breast cancer: stage 0 (zero), which is non-invasive ductal carcinoma in situ (DCIS), and stages I through IV (1 through 4), which are used for invasive breast cancer. The stage provides a common way of describing cancer, so doctors can work together to plan the best treatments.

Staging can be clinical or pathological. Clinical staging is based on the results of tests done before surgery, including physical examinations, mammograms, ultrasound, and MRI scans. Pathological staging is based on what is found during surgery to remove breast tissue and lymph nodes. The results are usually available several days after surgery.

Tumor (T)

Using the TNM system, the "T" plus a letter or number (0 to 4) is used to describe the size and location of the tumor. Tumor size is measured in centimeters (cm). A centimeter is roughly equal to the width of a standard pen or pencil. Stage may also be divided into smaller groups that help describe the tumor in even more detail. Specific tumor stage information is listed below.

TX: The primary tumor cannot be evaluated.

T0 (T plus zero): There is no evidence of cancer in the breast.

Tis: Refers to carcinoma in situ. The cancer is confined within the breast tissue ducts and has not spread into the surrounding tissue of the breast. There are 2 types of breast carcinoma in situ:

Tis (DCIS): DCIS is non-invasive cancer, but if not removed, it may develop into an invasive breast cancer later. DCIS means that cancer cells have been found in breast ducts and have not spread past the layer of tissue where they began.

Tis (Paget's): Paget disease of the nipple is a rare form of early, non-invasive cancer that is only in the skin cells of the nipple. Sometimes Paget disease is associated with invasive breast cancer. If there is invasive breast cancer, it is classified according to the stage of the invasive tumor.

T1: The tumor in the breast is 20 millimeters (mm) or smaller in size at its widest area. This is a little less than an inch. This stage is then broken into 4 sub stages depending on the size of the tumor:

- T1mi is a tumor that is 1 mm or smaller.
- T1a is a tumor that is larger than 1 mm but 5 mm or smaller.
- T1b is a tumor that is larger than 5 mm but 10 mm or smaller.
- T1c is a tumor that is larger than 10 mm but 20 mm or smaller.

T2: The tumor is larger than 20 mm but not larger than 50 mm.

T3: The tumor is larger than 50 mm.

T4: The tumor falls into 1 of the following groups:

T4a means the tumor has grown into the chest wall.

T4b is when the tumor has grown into the skin.

T4c is cancer that has grown into the chest wall and the skin.

T4d is inflammatory breast cancer.

Like for tumor staging, there are several other stagings in for Node and Metastasis examinations. I have focused my results on the tumor stagings.

		Stage	Primary tumour (T)*	Regional lymph node status (L)	Distant metastasis (M)
T- Tumour		**0**	Tis	N0	M0
T1	Tumour ≤ 2 cm	**I**	T1	N0	M0
T2	Tumour ≥ 2 cm but ≥ 5 cm		T0	N1	M0
T3	Tumour ≥ 5 cm	**IIA**	T1	N1	M0
T4	Tumour of any size with direct extension to chest wall or skin		T2	N0	M0
N- Lymph node		**IIB**	T2	N1	M0
N0	No cancer in regional node		T3	N0	M0
N1	Regional movable metastasis	**III A**	T0	N2	M0
N2	Non-movable regional metastases		T1	N2	M0
N3	Cancer in the internal mammary lymph nodes		T2	N2	M0
M- Metastasis			T3	N1/N2	M0
M0	No distant metastases	**III B**	T4	Any N	M0
M1	Distant metastases	**III C**	Any T	N3	M0
		IV	Any T	Any N	M1

Criteria for staging breast tumours according to the UICC ICD-10 TNM classification.

*Size measurements are for the tumour's greatest dimension.

Figure 31: TNM Staging

3.4.2 Finding the stage of tumor using the diameters

The code below will show how the stages found according to the conditions for tumor staging

```
    # --------------Tumor stage identifying function--------------
def tumor_st(self,diameter):
    if diameter == 0:
        self.tumorstage.configure(text="Tumor Stage T0")

    elif diameter > 0 and diameter <= 2:
        if diameter > 0 and diameter <= 0.1:
            self.tumorstage.configure(text="Tumor Stage T1mi")
        elif diameter > 0.1 and diameter <= 0.5:
            self.tumorstage.configure(text="Tumor Stage T1a")
        elif diameter > 0.5 and diameter <= 1:
            self.tumorstage.configure(text="Tumor Stage T1b")
        else:
            self.tumorstage.configure(text="Tumor Stage T1c")

    elif (diameter > 2) and (diameter <= 5):
        self.tumorstage.configure(text="Tumor Stage T2")
    elif diameter > 5:
        self.tumorstage.configure(text="Tumor Stage T3")
    else:
        self.tumorstage.configure(text="Tumor Stage T4")
    sys.stdout.flush()
```

Figure 32: Finding tumor stage

Final results will be displayed in the GUI.

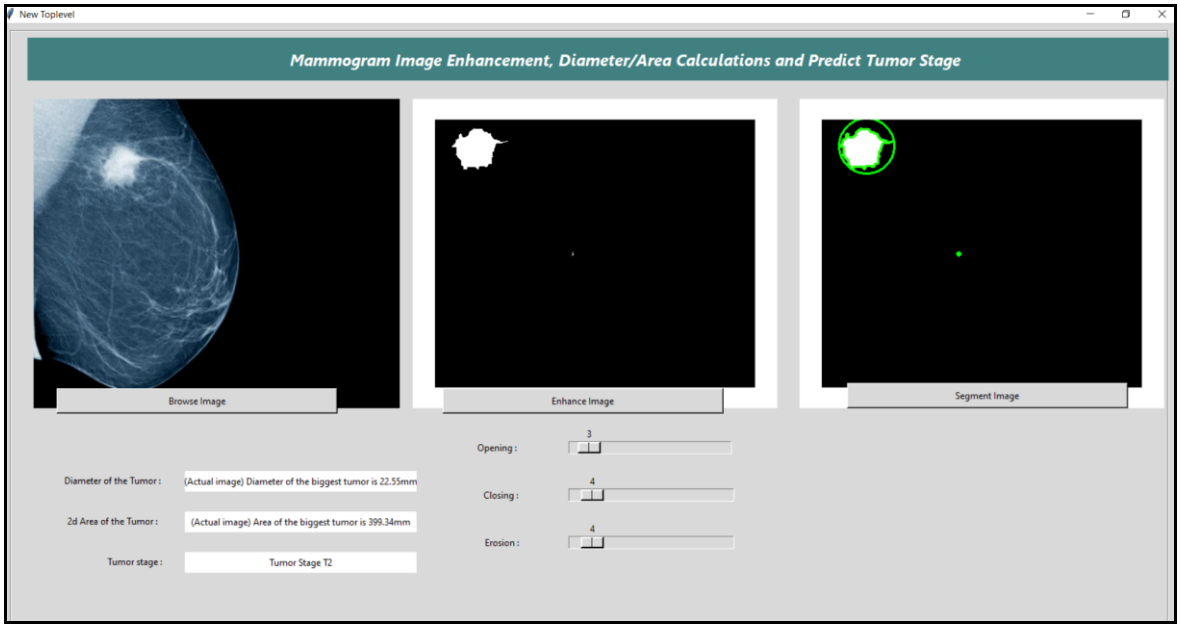

Figure 33: Final results displayed

3.5 Testing the developed procedure for other mammograms

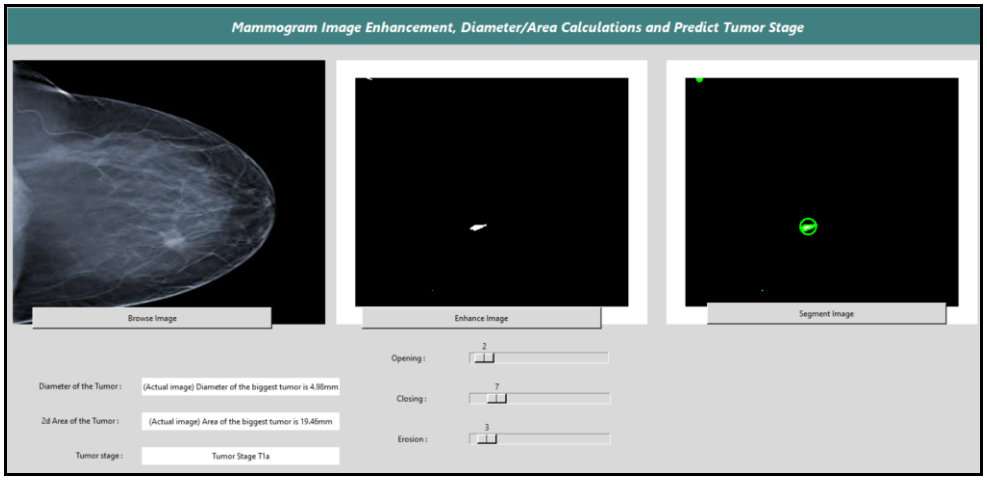

Figure 34: Mammogram with tiny development of a tumor

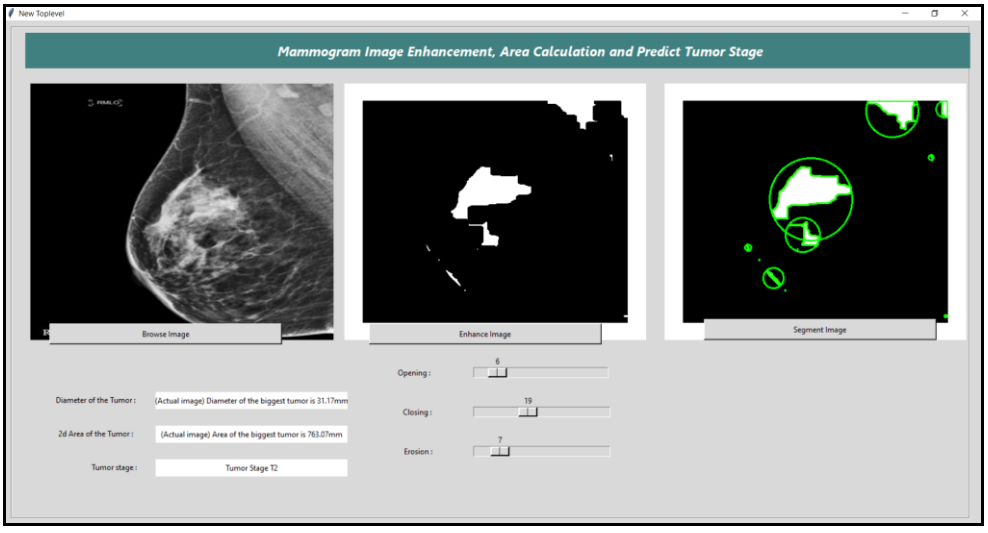

Figure 35: Mammogram with T2 stage tumor

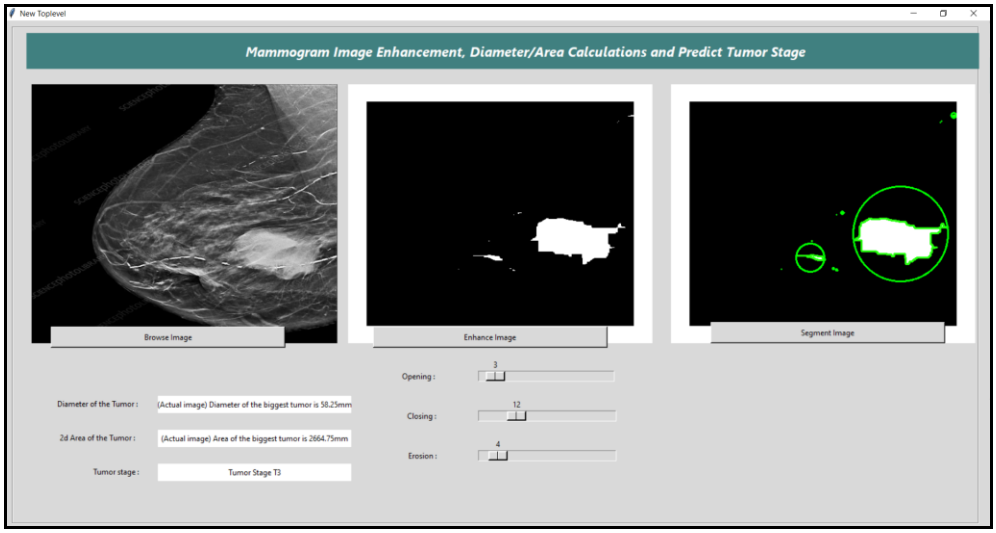

Figure 36: Mammogram with a T3 stage tumor

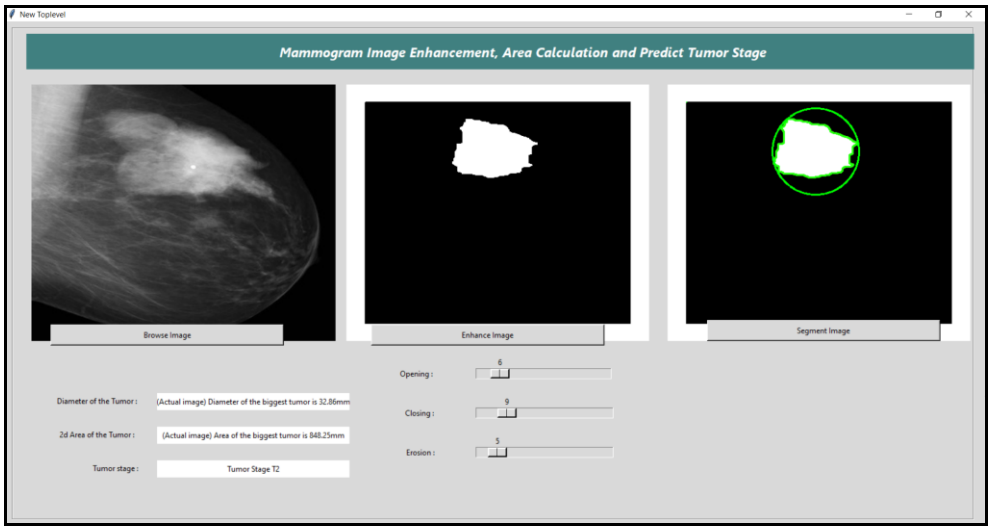

Figure 37: Mammogram with T2 stage tumor

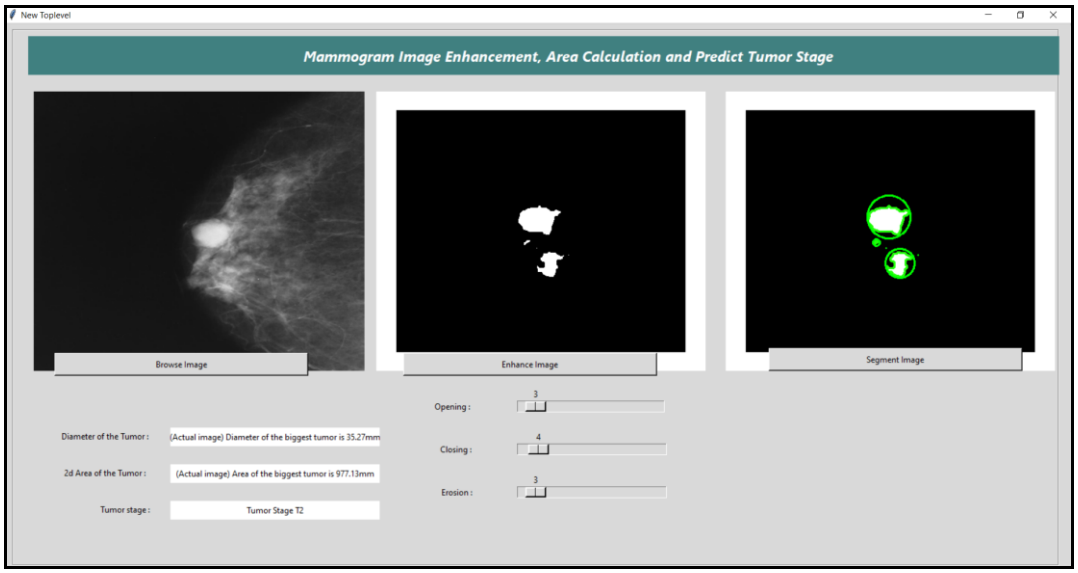

Figure 38: Mammogram with a T2 stage tumor

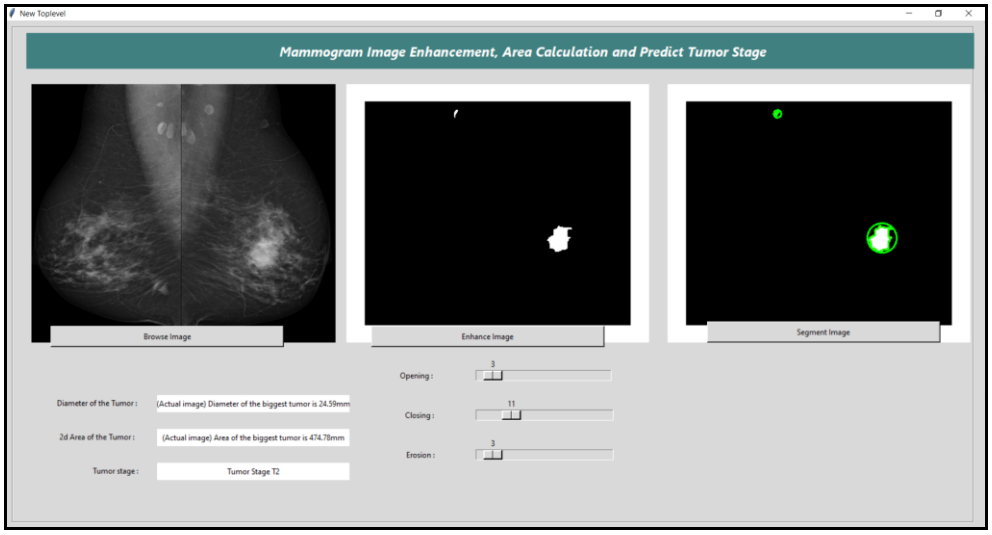

Figure 39: Mammogram with T2 stage tumor

CHAPTER 4
RESULTS AND DISCUSSION

4.1 Results and Discussion Overview

Using the GUI created to do this project, approximated results will be given to the user who inputs mammogram images with breast cancer. The results are made assuming that the collected images are of original size mammograms; this can be used by doctors with real mammogram images taken in the radiography department of the hospitals. Some of the results are shown below.

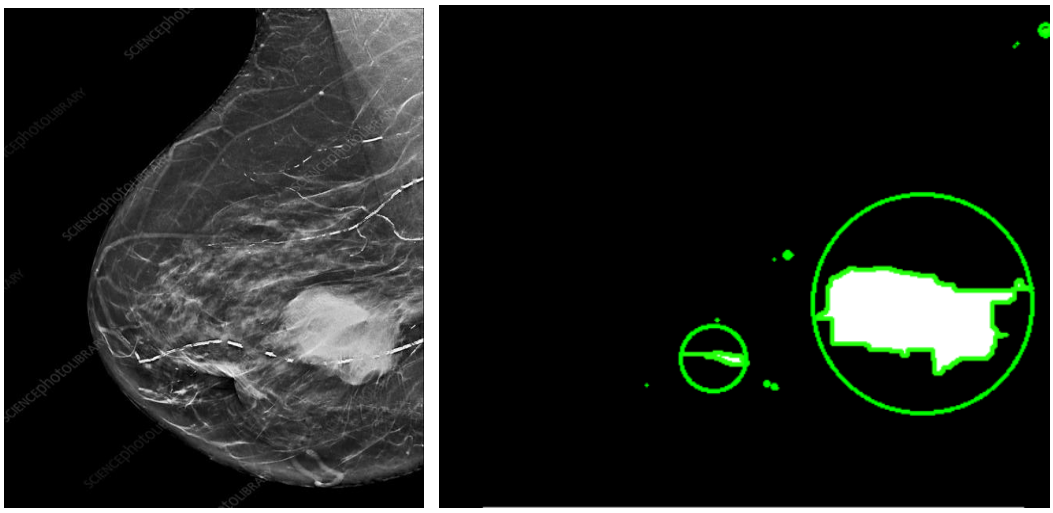

Figure 40: breast with tumor (left) and the resultant tumor with contours (right)

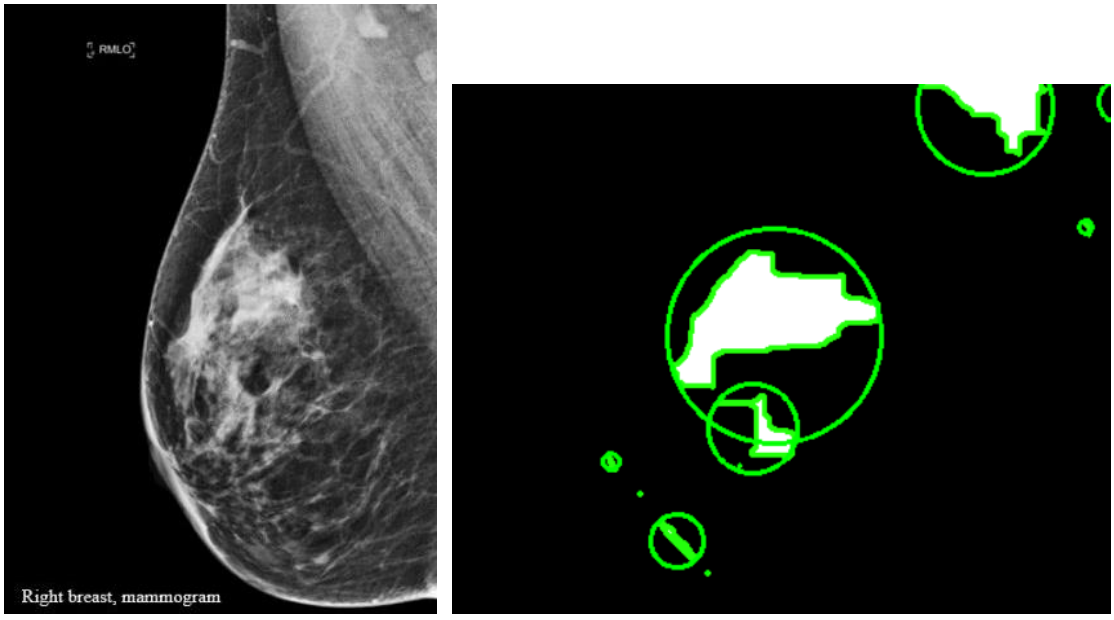

Figure 41: breast with tumor (left) and the resultant tumor with contours (right)

In hereafter preprocessing the left image and after all the morphological operations, contours are drawn to segment the tumor area. As you can see, the largest circle shows the tumor therefore, I used an array to store all the radiuses given by the circles and make it in descending order so the first element would provide the diameter of the tumor.

4.2 Complete Code of the Project

4.2.1 Browsing and resizing the image

In here I have calculated the resized percentage from the original image and it has been passed in to the enhanced image function. Enhanced image function is called in this browse function

```
    # ------------------------------------Image browser function-----------
def browse(self):
    img = Image.open(filedialog.askopenfilename())
    val = img.size
            # Actual size to resized image percentage calculation
    calc_area_percentage = ((477*404)/(val[0]*val[1]))*100

            # Resizing the image
    resized = img.resize((477,404), Image.ANTIALIAS)

            # convert the image to ImageTk format
    imgtk = ImageTk.PhotoImage(image=resized)

            # storing image in the  browselabel
    self.browselabel.image = imgtk
    self.browselabel.configure(image=imgtk)

            # calling enhance_img function to enhance the image
    self.enhance_img(resized,calc_area_percentage)
    sys.stdout.flush()
```

Figure 42: Code with Browse function

4.2.2 Image Enhancement

```
    # ----------------------------------Image enhancement function----------------------------
def enhance_img(self,img,area):
            # convert image to grayscale image
    resized_gray = cv2.cvtColor(np.float32(img), cv2.COLOR_BGR2GRAY)

            # apply thresholding
    thresh,img_bin = cv2.threshold(resized_gray,145,255,cv2.THRESH_BINARY)

            # getting opening, closing, erosion kernel values from the scale bars in the GUI
    val1 = self.Scale1.get()
    val2 = self.Scale2.get()
    val3 = self.Scale3.get()

            # IMPLEMENTING KERNELS
    kernel = np.ones((val1,val1), np.uint8)
    kernel2 = np.ones((val2,val2), np.uint8)

            # APPLYING MORPHOLOGICAL OPERATIONS TO THE IMAGE
    opening = cv2.morphologyEx(img_bin, cv2.MORPH_OPEN, kernel)
    closing = cv2.morphologyEx(opening, cv2.MORPH_CLOSE, kernel2)
    erosion = cv2.erode(closing,cv2.getStructuringElement(cv2.MORPH_ELLIPSE,(val3,val3)))

            # CROP IMAGE TO EXTRACT THE TUMOR PART
    crop_img = erosion[50:400, 60:477]

            # convert the image to ImageTk format
    image1 = Image.fromarray(crop_img)
    imgtk = ImageTk.PhotoImage(image=image1)

            # storing image in the  enhanceimage label
    self.enhanceimage.image = imgtk
    self.enhanceimage.configure(image=imgtk)

            # writing the enhanced image to the working directory
    cv2.imwrite("img.png", crop_img)

            # calling the image segmenting function
    self.segment_img(area)
    sys.stdout.flush()
```

Figure 43: Enhance Image function

The code here uses image processing techniques to enhance the image and separate the tumor from the breast. Image segmentation function is called within this function, passing the resized area percentage from the previous function

4.2.3 Image Segmentation

```
    #----------------------------------------Image segmentation function---------------------
def segment_img(self,area):
    thresh = cv2.imread('img.png',0)
            # check if the image is black without any detected tumors in white objects

    if np.mean(thresh) == 0:
                    # convert the image to ImageTk format
        image1 = Image.fromarray(thresh)
        imgtk = ImageTk.PhotoImage(image=image1)

                    # storing image in the  segImg label
        self.segImg.image = imgtk
        self.segImg.configure(image=imgtk)

                    # return messages to the following labels
        self.diameter.configure(text="No Tumor Detected")
        self.area.configure(text="No Tumor Detected")
        self.tumorstage.configure(text="No Tumor Detected")
```

Figure 44: First part of the segmentation function

Here I have used an IF condition to identify the image has foreground (white) objects and return that there's no tumor detected. The rest of the code inside the ELSE where the function draws all the contours and calculates the diameter of the tumor detected, and the Tumor staging function is called at the end.

```
else:
                # find contours
    contours,hierarchy = cv2.findContours(thresh,2,1)  #include another variable before 'contuors' if you get
                # convert image to grayscale image
    thresh = cv2.cvtColor(thresh, cv2.COLOR_GRAY2RGB)
    cnt = contours
                # draw contours on the image
    cv2.drawContours(thresh, cnt, -1, (36, 255, 12), 2)

    arr = [0 for i in cnt]

                # finding the radius of the segmented objects
    for i in range (len(cnt)):
        (x,y),radius = cv2.minEnclosingCircle(cnt[i])
        center = (int(x),int(y))
        radius = int(radius)
        cv2.circle(thresh,center,radius,(0,255,0),2)
                # populating the diameters of the objects to arr
        arr[i] = radius*0.26458333

    # convert the image to ImageTk format
    image1 = Image.fromarray(thresh)
    imgtk = ImageTk.PhotoImage(image=image1)
    # storing image in the  segImg label
    self.segImg.image = imgtk
    self.segImg.configure(image=imgtk)
    # sorting the array in descending order
    arr.sort(reverse=True)
    # find the area of the biggest tumor in the image
    area_circle = math.pi * (arr[0]*arr[0])
    # finding the area for the actual sized image
    actual_circle_area = area_circle*(100/area)
    # calculating the diameter from the actual circle area
    diameter_result1 = round(math.sqrt((actual_circle_area/math.pi))*2, ndigits=2)
    diameter_result2 = "(Actual image) Diameter of the biggest tumor is " + str(diameter_result1)  + "mm"

    # storing the diameter in the diameter label
    self.diameter.configure(text=diameter_result2)
    result = round(actual_circle_area, ndigits=2)
    resultant_area = "(Actual image) Area of the biggest tumor is " + str(result) + "mm"

                # storing the area of the tumor in the area label
    self.area.configure(text=resultant_area)
                # calling the tumor_st function
    self.tumor_st(float(diameter_result1/10))
sys.stdout.flush()
```

Figure 45: ELSE part which contains the IP techniques

4.2.4 Tumor Staging

According to the tumor staging, if conditions are used to give the output. Here the diameter value is passed from the previous function.

```
    # --------------Tumor stage identifying function--------------
def tumor_st(self,diameter):
    if diameter == 0:
        self.tumorstage.configure(text="Tumor Stage T0")

    elif diameter > 0 and diameter <= 2:
        if diameter > 0 and diameter <= 0.1:
            self.tumorstage.configure(text="Tumor Stage T1mi")
        elif diameter > 0.1 and diameter <= 0.5:
            self.tumorstage.configure(text="Tumor Stage T1a")
        elif diameter > 0.5 and diameter <= 1:
            self.tumorstage.configure(text="Tumor Stage T1b")
        else:
            self.tumorstage.configure(text="Tumor Stage T1c")

    elif (diameter > 2) and (diameter <= 5):
        self.tumorstage.configure(text="Tumor Stage T2")
    elif diameter > 5:
        self.tumorstage.configure(text="Tumor Stage T3")
    else:
        self.tumorstage.configure(text="Tumor Stage T4")
    sys.stdout.flush()
```

Figure 46: Code for tumor staging

4.3 Assumptions

- Assumptions were made for the images collected. Considering them to be real size mammogram images as they were only available in google images.

CHAPTER 5
CONCLUSION

In this project, a Graphical User Interface and a method were developed to identify tumors in breast cancer mammogram images. All the objectives of this project were fulfilled. Using OpenCV in Python, this project was developed for processing the images of breast cancer mammograms to pinpoint and locate tumors, making it easier for doctors. Diameters and areas were calculated to find the tumor stage that supports this project for further development.

REFERENCES

[1] Mussarat Yasmin, Muhammad Sharif and Sajjad Mohsin, "Survey Paper on Diagnosis of Breast Cancer Using Image Processing Techniques" Research Journal of Recent Sciences, Vol. 2(10), 88-98, October (2013)

[2] Yousif M.Y Abdallah1*, Sami Elgak1, Hosam Zain2, Mohammed Rafiq3, Elabbas A. Ebaid4, Alaeldein A. Elnaema5, "Breast cancer detection using image enhancement and segmentation algorithms", Biomedical Research 2018; 29 (20): 3732-3736.

[3] Moumena Al-Bayati, Ali El-Zaart, "Mammogram Images Thresholding for Breast Cancer Detection Using Different Thresholding Methods", Advances in Breast Cancer Research Vol.2 No.3, June 27, 2013

[4] D. Ponraj, M. Jenifer, P. Poongodi and J. Manoharan, "A Survey on the Preprocessing Techniques of Mammogram for the Detection of Breast Cancer," Journal of Emerging Trends in Computing and Information Sciences, Vol. 2, No. 12, 2011, pp. 656-664. http://www.cisjournal.org

[5] https://docs.opencv.org/master/d7/d4d/tutorial_py_thresholding.html

[6] https://docs.opencv.org/master/d9/d61/tutorial_py_morphological_ops.html

[7] https://docs.opencv.org/master/d4/d73/tutorial_py_contours_begin.html

[8] https://emedicine.medscape.com/article/2007112-overview

[9] Abdullah-Al Nahid[1] and Yinan Kong "Involvement of Machine Learning for Breast Cancer Image Classification", Review Article Hindawi Computational and Mathematical Methods in Medicine Volume 2017, Article ID 3781951, 29 pages https://doi.org/10.1155/2017/3781951

[10] M. Melloul and L. Joskowicz, "Segmentation of Micro Calcification in X-Ray Mammograms Using Entropy Thresholding," CARS/Springer, Berlin, 2002.

[11] A. Hoyer and W. Spiesberg, "Computerized Mammogram Processing," Phillips Technical Review, Vol. 38, 1979, pp. 347-355.

[12] T. Lau and W. Bischoff, "Automated Detection of Breast Tumors Using the Asymmetry Approach," Computers and Biomedical Research, Vol. 24, No. 3, 1991, pp. 273-295. doi:10.1016/0010-4809(91)90049-3

[13] Yin and M. Giger, "Computerized Detection of Masses in Digital Mammogram: Analysis of Bilateral Subtraction Images," Medical Physics and Biology, Vol. 28, 1991, pp. 955-963.

[14] U. Bick and M. Giger, "Automated Segmentation of Digitized Mammograms," Academic Radiology, Vol. 2, No. 1, 1995, pp. 1-9. doi:10.1016/S1076-6332(05)80239-9

[15] Y. Qiaoa, Q. Hua, G. Qiana, S. Luob and W. Nowinskia, "Thresholding Based on Variance and Intensity Contrast," Pattern Recognition, Vol. 40, No. 2, 2007, pp. 596-608. doi:10.1016/j.patcog.2006.04.027