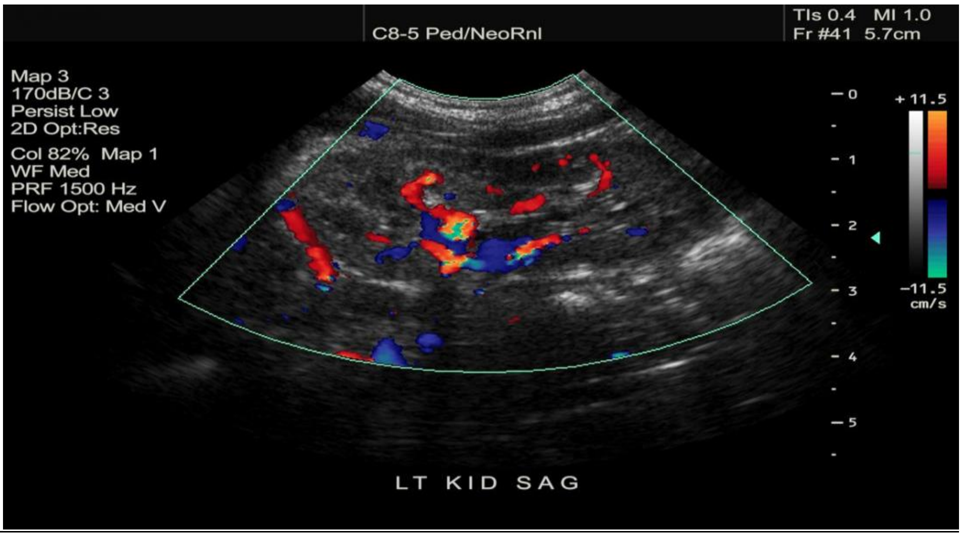
BIOMEDICAL IMAGING
A PROJECT REPORT
ECE-1023



ULTRASOUND IMAGING

PRESENTED BY:

NAME: IMMANUEL VARGHESE KOSHY
REG. NO: 20BEC0315
immanuelvarghese.koshy2020@vitstudent.ac.in

NAME: SRISHTI CHOPRA
REG. NO: 20BML0046
srishti.chopra2020@vitstudent.ac.in

PREFACE

Ultrasound uses high frequency sound waves to observe inside the body. It is captured in real time. Ultrasound waves are emitted from the piezoelectric crystal of the ultrasound transducer and this depending on the acoustic impedance of different materials produces different levels of black and white images.

In ultrasound , a transducer is placed directly on the skin over a thin layer of gel so that the ultrasound waves are transmitted from the transducer through the gel into the body.

Ultra sound imaging is based on the reflection of the waves from a organ or body part. The strength of the sound signal and the time taken by it to travel and return to the transducer is used to recreate an ultrasound image.

Now coming to the software side ultrasound imaging , different algorithms are used by different companies but the basic concept behind of them remains the same. Now for the purpose of illustrating an example, we have taken a Matlab based code structure.

The purpose of this example is the contour extraction of a region of interest in ultrasonic images. An active contour model gained using a gradient vectors.

Now for the main program to execute or run, we need various function modules used to perform different operations.

PROGRAMME CODE: (MATLAB)

```
close all
clear all
clc

% Read the image
[I,map] = imread('2.jpg');
Ioriginal = I;
% Anisotropic Diffusion
I = AnisotropicDiffusion(0.01,0.002);
disp(' Press any key to continue');
pause();

% Compute its edge map
disp(' Computing edge map ...');
[Ex,Ey] = gradient(I);
f = sqrt(Ex.*Ex+Ey.*Ey);
% f = 1 - I/255;
% Compute the GVF of the edge map f

disp(' Compute GVF ...');
[u,v] = GVF(f, 0.02, 100);
% [u,v] = GGVF(f,0.002,100);
disp(' Nomalizing the GVF external force ...');
mag = sqrt(u.*u+v.*v);
px = u./(mag+1e-10); py = v./(mag+1e-10);
% px = u;
% py = v;

% display the results
figure(2);
subplot(221); imdisp(I); title('test image');
subplot(222); imdisp(f); title('edge map');
```

```
[fx,fy] = gradient(f);
subplot(223); quiver(fx,fy);
axis off; axis equal; axis 'ij';     % fix the axis
title('edge map gradient');
subplot(224); quiver(px,py);
axis off; axis equal; axis 'ij';     % fix the axis
title('normalized GVF field');

% snake deformation
disp(' Press any key to start GVF snake deformation');
pause;
figure(2); subplot(221); cla;%cla= clear current axes
% colormap(gray(64)); image(((1-f)+1)*40);
imshow(I);

axis('equal', 'off');
[x,y] = snakeinit(I,1);
snakedisp(x,y,'r')

for i=1:25,
   [x,y] = snakedeform(x,y,0.01,0,1,0.5,px,py,5);
   [x,y] = snakeinterp(x,y,2,0.5);
snakedisp(x,y,'r')
   title(['Deformation in progress,  iter = ' num2str(i*5)])
   pause(1);
end

disp(' Press any key to display the final result');
pause;
cla;
%colormap(gray(64)); image(((1-f)+1)*40);
imshow(Ioriginal);
axis('equal', 'off');
snakedisp(x,y,'r')
title(['Final result,  iter = ' num2str(i*5)]);
```

ANISOTROPIC DIFFUSION :-

This module is used to smooth the image while preserving the edge integrity of the image. Its also called Perona- Malik Diffusion.

“”

```
k=k1; lambda=lambda1; N=90;
%weight number control smooth level of image [img,map] =
imread('2.jpg');
disp(' Computing edge map ...'); img = rgb2gray(img);
img = im2double(img); figure();
subplot(121);imshow(img);title('The original image') [m n]=size(img);
imgn=zeros(m,n); for i=1:N

for p=2:m-1 for q=2:n-1
```

%present divergence of pexil, calculate partial derivative in four directions, partial difference in different directions
%if it changes a lot, it is edgy. it should be preserve NI=img(p-1,q)-img(p,q);
SI=img(p+1,q)-img(p,q);
EI=img(p,q-1)-img(p,q);
WI=img(p,q+1)-img(p,q);

%heat conductivity coefficients in four directions, the more
%change in each direction, the less the value is. cN=exp(-NI^2/(k*k));

cS=exp(-SI^2/(k*k)); cE=exp(-EI^2/(k*k)); cW=exp(-WI^2/(k*k));

imgn(p,q)=img(p,q)+lambda*(cN*NI+cS*SI+cE*EI+cW*WI);
%the new value after diffuse calculation end
end

img=imgn; end
I = imgn;
subplot(122);imshow(imgn);title('The processed image') “”
This helps the further functions run more efficiently. The diffusion process uses a gaussian model. The same model as the below given is used in code for above.

$$\frac{\partial I}{\partial t} = \operatorname{div}\left(c(x,y,t)\nabla I\right) = \nabla c \cdot \nabla I + c(x,y,t)\Delta I$$

GVF:-

This modules used to create a gradient vector flow. This is usually used for object tracking and edge detection. It uses a computer vision framework introduced by Chenyang Xu and Jerry L. Prince.

“”

```
[fx,fy] = gradient(f); u = fx; v = fy;
SqrMagf = fx.*fx + fy.*fy;


% to  solve for the GVF u,v for i=1:ITER

u = u + 4*mu*del2(u) - SqrMagf.*(u-fx); v = v + 4*mu*del2(v) -
SqrMagf.*(v-fy);
```

“”

IMDISP:-

This is a Matlab function that combines the functionality of IMAGE, IMAGESC, IMSHOW and MONTAGE. Images are placed in subplots. It scales the dynamic range of an image and displays it.

“”

```
function imdisp(I)

x = (0:255)./255;
grey = [x;x;x]'; minI = min(min(I));
maxI = max(max(I));
I = (I-minI)/(maxI-minI)*255; image(I); axis('square','off');
colormap(gray(256));
```

“”

Here the division by 255 signifies converting the 0-255 range to a 0.0-1.0 range. Usual RGb images have a 0-255{ 256 bit} range.

SNAKE DEFORM || SNAKE DISP || SNAKE INDEX || SNAKE INIT || SNAKE INTERP :-

Snake Is the basis for a type of image segmentation. It is mainly used for identifying various surfaces or living or nonliving objects from an image.

Now Snake utilises something known as active contours. Its a segmentation method that uses “ energy forces “to separate pixels our internet from an image.

Its mainly used to mark the smooth shapes in images and to construct contours for edge regions.

The model’s primary function is to identify and outline the target object for segmentation.

The general mathematical equation this follows is

$$E_{\text{snake}}^{*} = \int_{0}^{1} E_{\text{snake}}(\mathbf{v}(s))\,ds = \int_{0}^{1} \left(E_{\text{internal}}(\mathbf{v}(s)) + E_{\text{image}}(\mathbf{v}(s)) + E_{\text{con}}(\mathbf{v}(s))\right) ds$$

OUTPUT IMAGES

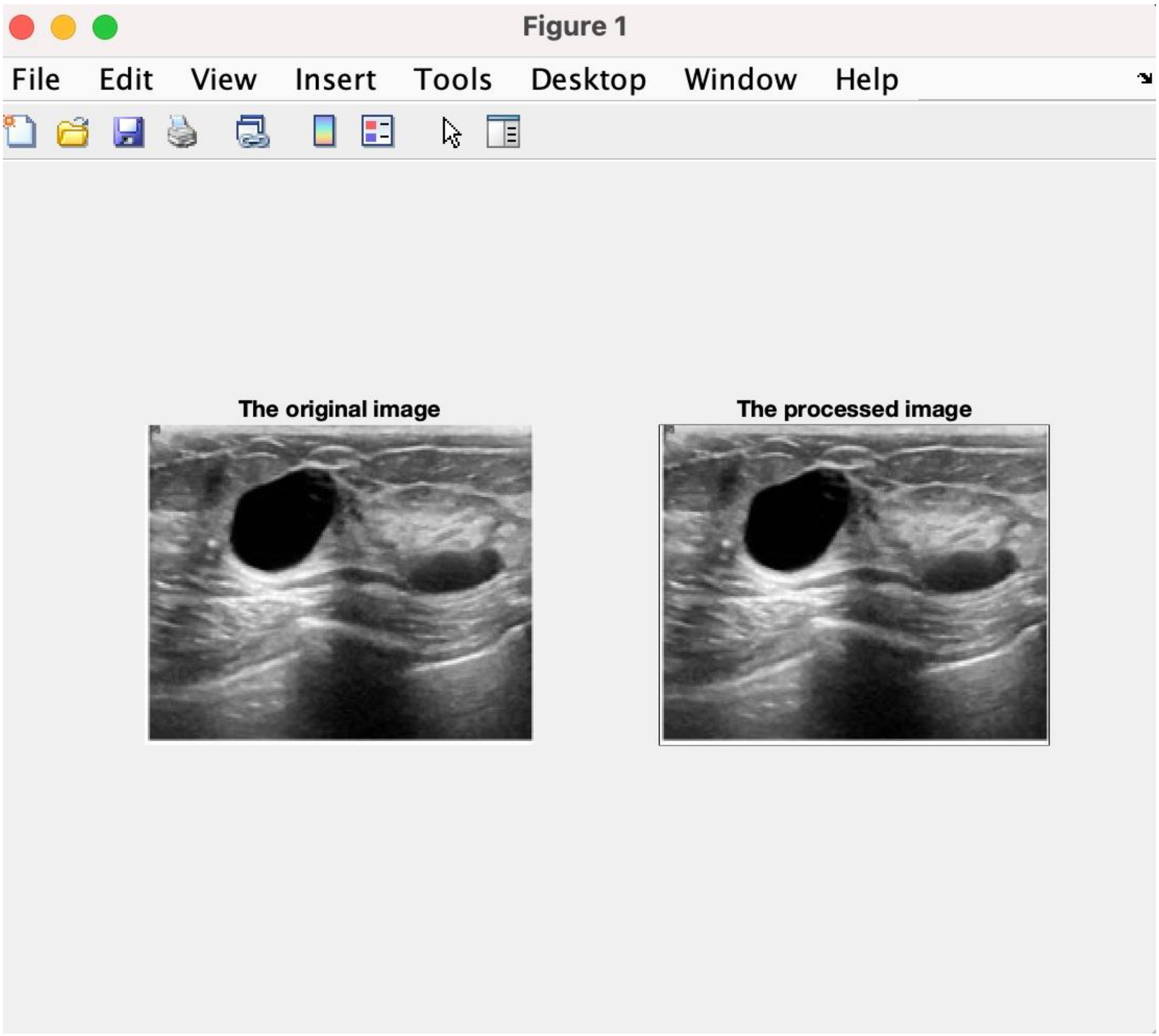


Figure 1
File Edit View Insert Tools Desktop Window Help
The original image
The processed image

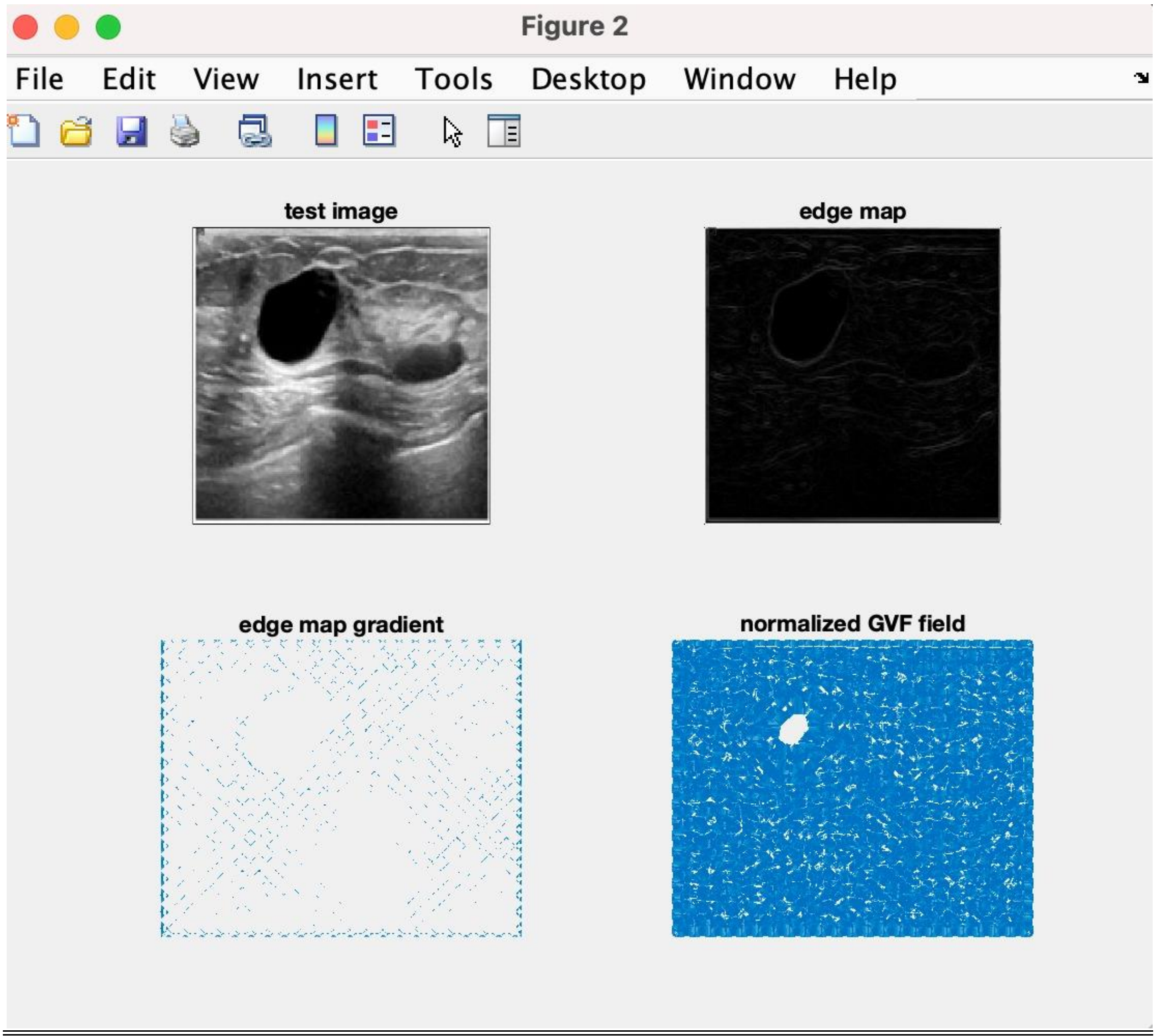


Figure 2
File
Edit
View
Insert
Tools
Desktop
Window
Help
test image
edge map
edge map gradient
normalized GVF field

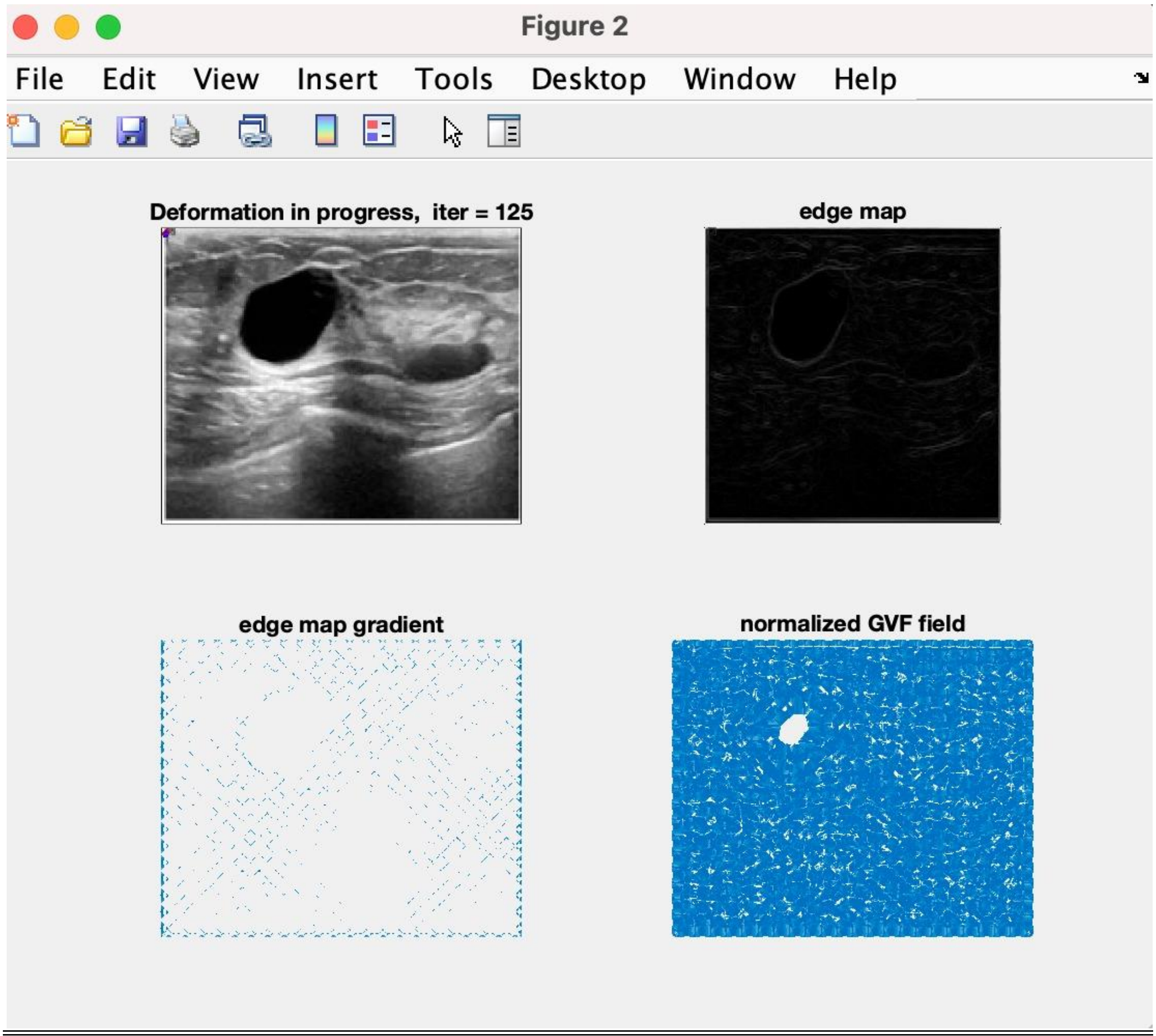


Figure 2
File
Edit
View
Insert
Tools
Desktop
Window
Help
Deformation in progress, iter = 125
edge map
edge map gradient
normalized GVF field

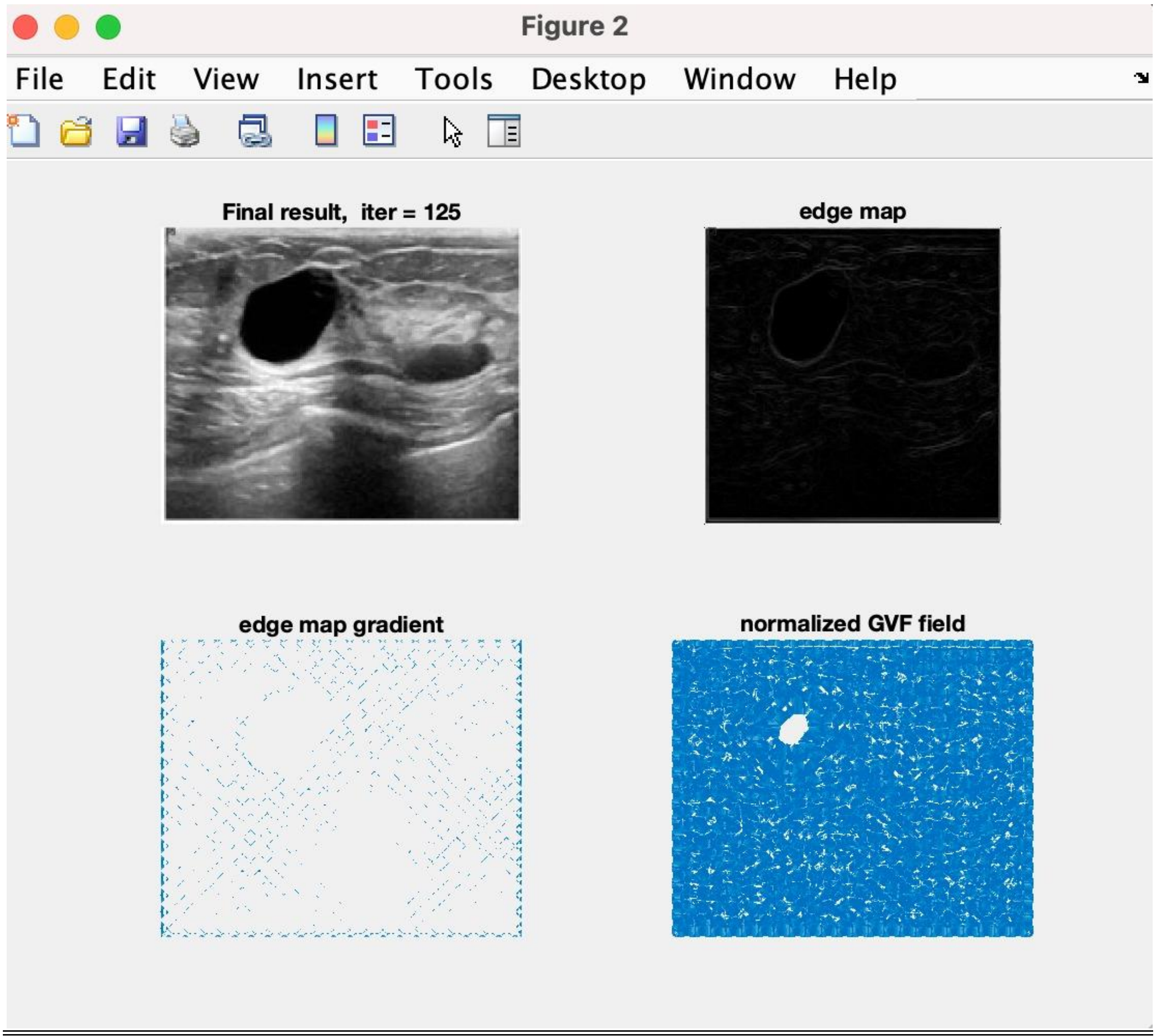


Figure 2
File
Edit
View
Insert
Tools
Desktop
Window
Help
Final result, iter = 125
edge map
edge map gradient
normalized GVF field

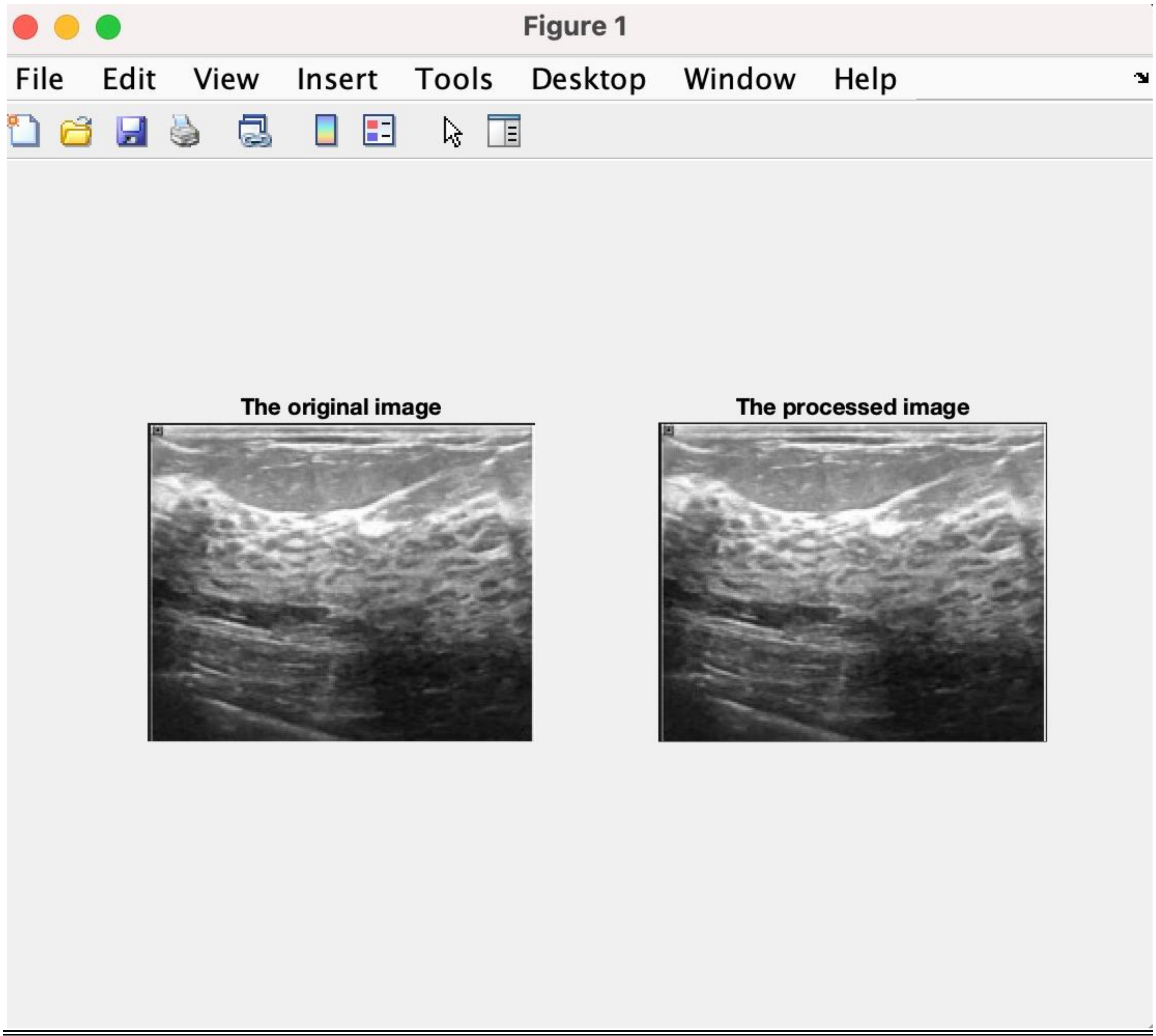
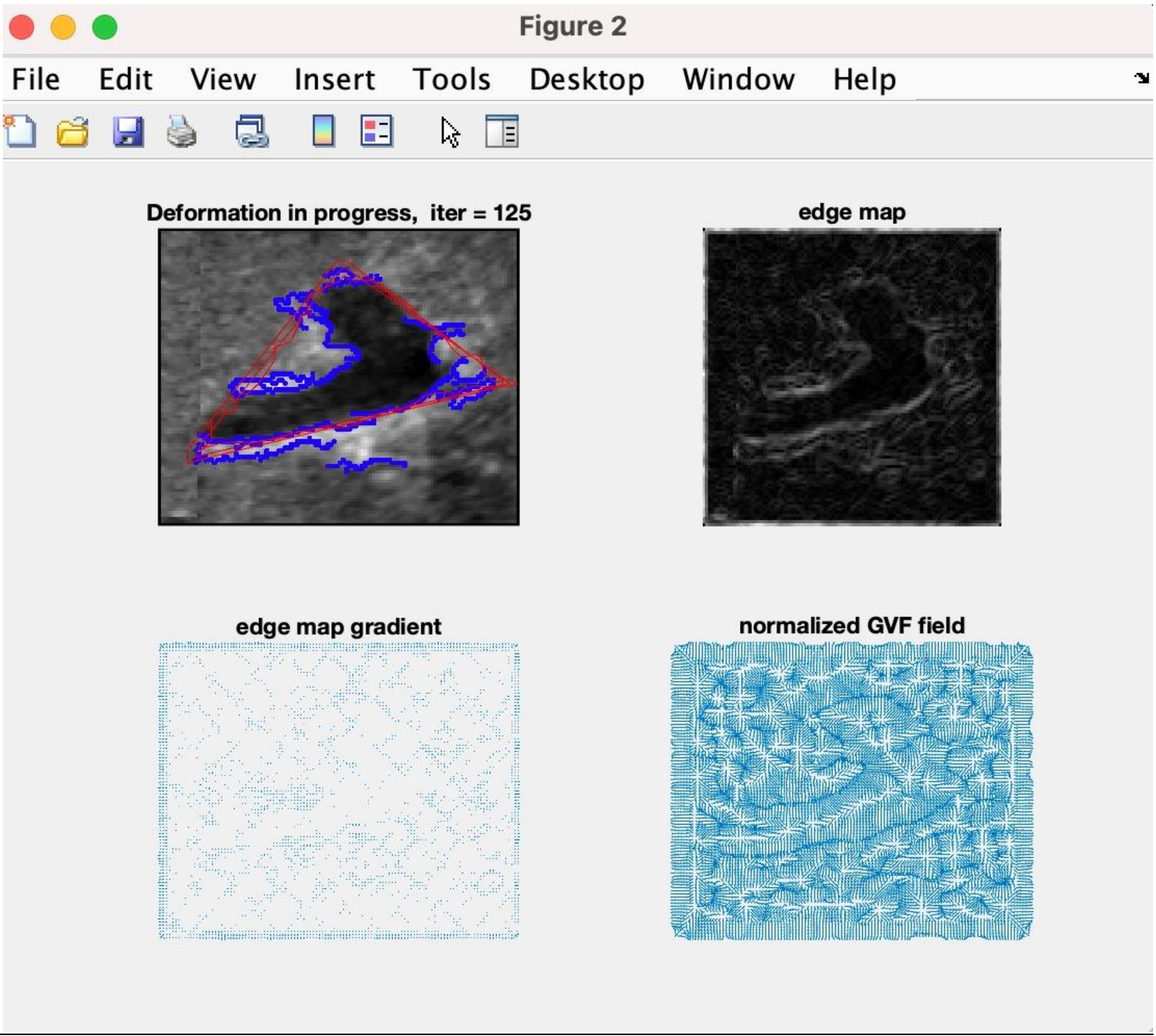


Figure 1
File Edit View Insert Tools Desktop Window Help
The original image
The processed image



- All the modules/ functions required to perform the image processing were coded and saved in the same matlab folder as the main program along with the images that need to be processed.
- Lower iter value means that the mass of cancer has shrunk
- The above given pictures are the expected output for the ultrasound scan run through a segmentation algorithm. Now a lower iter value after each scan would mean the cancerous cells have shrunk.
- As you can see the edges have been detected , image smoothened, GVF fields have been created and an iteration value has been outputted

REFERENCES

- Perona P, Malik J. Scale-space and edge detection using anisotropic diffusion[J]. IEEE Transactions on pattern analysis and machine intelligence, 1990, 12(7): 629-639.
- Kass M, Witkin A, Terzopoulos D. Snakes: Active contour models[J]. International journal of computer vision, 1988, 1(4): 321-331.
- Yang-xu C. Snakes, shapes, and gradient vector flow[C]// Inter-national Conference on Image Processing. 2002, 9(2): 17-820.