2018

Active Contours Implementation

Steven D. Oberhelman
University of Rhode Island, soberhelman@my.uri.edu

ACTIVE CONTOURS IMPLEMENTATION

BY

STEVEN D. OBERHELMAN

A THESIS SUBMITTED IN PARTIAL FULFILLMENT OF THE

REQUIREMENTS FOR THE DEGREE OF

MASTER OF SCIENCE

IN

ELECTRICAL ENGINEERING

UNIVERSITY OF RHODE ISLAND

2018

MASTER OF SCIENCE THESIS

OF

STEVEN D. OBERHELMAN

APPROVED:

Thesis Committee:

Major Professor Frederick J. Vetter

Richard Vaccaro

Manbir Sodhi

Nasser H. Zawia

DEAN OF THE GRADUATE SCHOOL

UNIVERSITY OF RHODE ISLAND

2018

ABSTRACT

Many image processing problems require the detection of objects in an image. In problems such as these, active contours are widely used to extract features of interest from an image or model for computer analysis. Many examples of active contour applications come from biomedical image processing, where both healthy and diseased areas of the body can be detected, counted, and/or measured; however, a number of other applications of active contours exist outside of the biomedical field. This paper will survey the state of active contours and deliver software that can be used to explore the various nuances of active contour application. The topics in this paper include an explanation of internal energy and external energy using the Marr method, Gradient Vector Flow (GVF) method, and Vector Field Convolution (VFC) method. This paper will focus on 2-D active contours, but almost all of these methods can also be used to fit an active surface to 3-D objects. In the appendix, a Matlab program is provided that implements these methods.

ACKNOWLEDGMENTS

I would like to sincerely thank my major professor, Dr. Fred Vetter, for all the support and guidance he has given throughout this work. I am grateful for the opportunity to work with him on this project. My committee members, Dr. Richard Vaccaro, and Dr. Monbir Sodhi, also deserve recognition for dedicating their time to reviewing and providing meaningful feedback for my thesis. I would also like to thank my defense chair, Dr. Chengzhi Yuan.

I would like to thank my parents for their love and for supporting me in every way possible and to Vanessa for all of her love, support and proofreading. Additionally I would like to thank all of my friends and coworkers for their support throughout the year and to Dr. Koch for his professional guidance and constant support.

TABLE OF CONTENTS

ABSTRACT ii

ACKNOWLEDGMENTS iii

TABLE OF CONTENTS iv

LIST OF FIGURES vi

LIST OF TABLES viii

CHAPTER

1 Introduction 1

2 Active Contour Mechanics 5

2.1 Internal Energy 5

2.2 Edge map 6

2.3 External Energy 8

2.4 Kass, Witkin, and Terzopoulos (KWT) Method 11

3 External Energy 13

3.1 Gradient Vector Flow (GVF) 13

3.2 Vector Field Convolution 15

4 Active Contours Software 18

4.1 Active Contours Software Introduction 18

4.2 Active Contours Software Performance 21

5 Future Work 26

5.1 Splines and Models 26

Page

5.2 Active Surfaces 26

5.3 Active Contour Merging, Ballooning Force and GVF Boundary Conditions 27

5.4 Generalized Gradient Vector Flow 27

5.5 Tracking 28

5.6 Stopping Criteria 28

5.7 Clustering 28

LIST OF REFERENCES 29

APPENDIX

A Matlab script 31

B Internal Energy Derivation 38

C Gradient Vector Flow Derivation 39

BIBLIOGRAPHY 41

LIST OF FIGURES

Figure		Page
1	A noisy star image.	2
2	The edge map of the noisy star image.	2
3	The initial active contour on the edge map.	3
4	The active contour after a few iterations. The red points are the most recent iteration of the active contour. The blue lines are past active contours.	3
5	The final active contour.	4
6	Using the program developed in this work to determine the edges of a heart slice	4
7	(Upper Left) Image with row 110 highlighted in red. (Upper Right) Pixel values of image row 110 with grayscale pixel values between 0 and 1. (Lower Left) Edge map of image row 110 using Equation 3. (Lower Right) External Energy of image row 110 using the KWT method [1].	7
8	The external energy using the Marr method. The black line is the edge map of the image and the blue vectors indicate the direction and magnitude of the external force. The left panel uses a Gaussian blur with a standard deviation of 10 while the right panel uses a Gaussian blur with a standard deviation of 4. Note how the Marr method that uses Gaussian blur with a large standard deviation does not create large external energies near the edge.	10
9	The external force using the GVF with $\mu = 0.3$. The black line is the edge map of the image and the blue vectors indicate the direction and magnitude of the external force.	15
10	(Left Panel) VFC vector field kernel using the magnitude function from Equation 12, $\epsilon = 0.01$ and $\gamma = 2$. (Right Panel) VFC vector field kernel using the magnitude function from Equation 13, $\xi = 2$.	16

Figure **Page**

11 The external force using the VFC with $\xi = 6$ in Equation 13. The black line is the edge map of the image and the blue vectors indicate the direction and magnitude of the external force. . . . 17

12 Demonstration of Active Contour Program. 18

13 Image to test the program developed in this work. 21

14 Marr external energy test using the test image in Figure 13 . . . 23

15 VFC external energy test using the test image in Figure 13 . . . 24

16 GVF external energy test with an initial external energy using the Marr approach using the test image in Figure 13. 25

LIST OF TABLES

Table		Page
1	List of variables	20

CHAPTER 1

Introduction

An active contour is a thin curve on an image that can delineate a desired feature. Active contours are used in image processing to extract features of interest from an image or model for computer analysis. An active contour is defined as a list of ordered points or as a continuous parametric function $(x(s), y(s))$ where the range of s is defined; typically, $s\epsilon[0, 1]$. The active contour forms to the edge of an object by minimizing a summation of energies which gives the active contour its properties. The two main energies are internal and external energy, but others like a ballooning force [2] can be included [3] [1].

To demonstrate an active contour, consider the image in Figure 1. We then generate the edge map which is explained in section 2.2. The edge map is large when it is near a potential edge (Figure 2). We begin with an initial contour as shown in Figure 3. Then we step it through an iterative process that minimizes the internal and external energy, which moves the active contour points. As we run the active contour though this process, the points will begin the line up with the edges of the image as shown in Figure 4. After running the active contour through enough iterations, the points will hopefully line up with the edges of the image as shown in Figure 5.

Active contours are used to determine the edges of an object of interest. An example of this approach is given in Figure 6 where an active contour is being used to find the edges of a heart slice. Other application of active contours has been determining the size of patients' lungs from MRI scans [4] and determining the volume of a mouse's heart ventricle over time [5]. Successive images can be used to track objects moving though an image as demonstrated by Action and Ray [6].

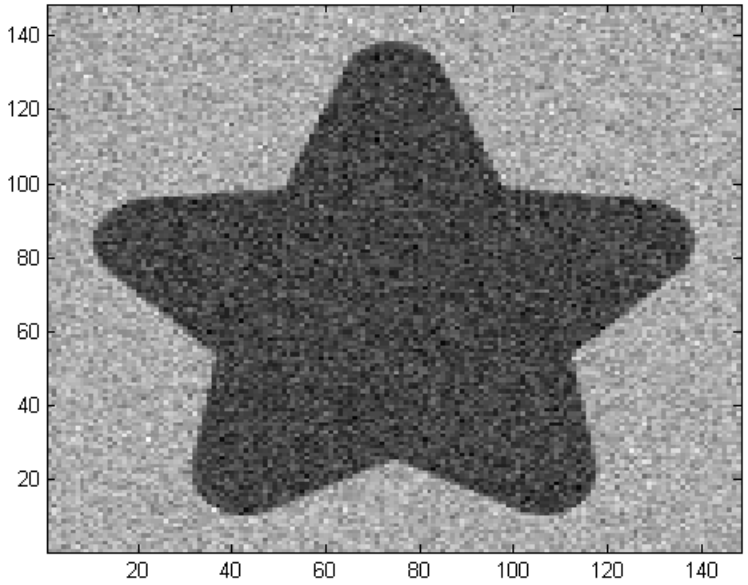

Figure 1. A noisy star image.

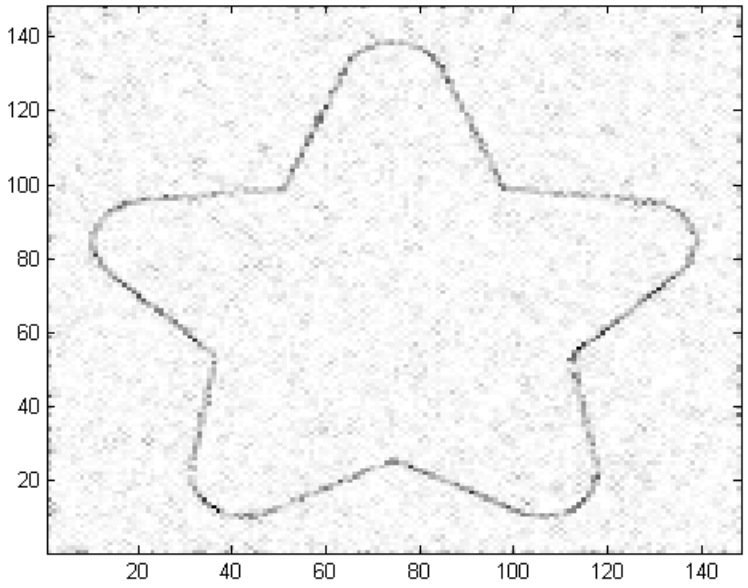

Figure 2. The edge map of the noisy star image.

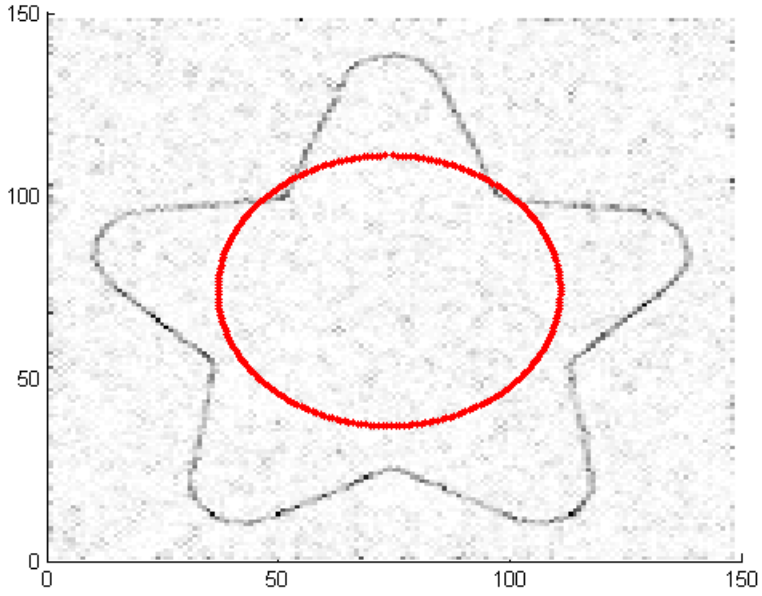

Figure 3. The initial active contour on the edge map.

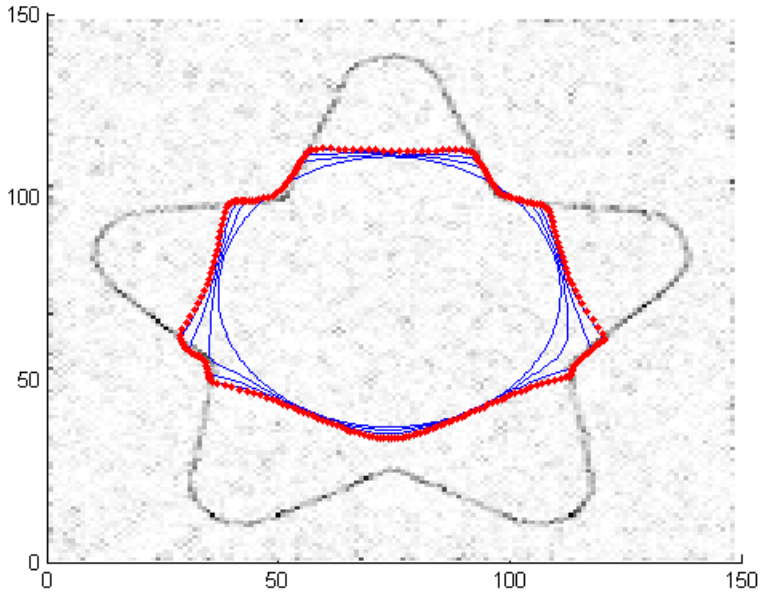

Figure 4. The active contour after a few iterations. The red points are the most recent iteration of the active contour. The blue lines are past active contours.

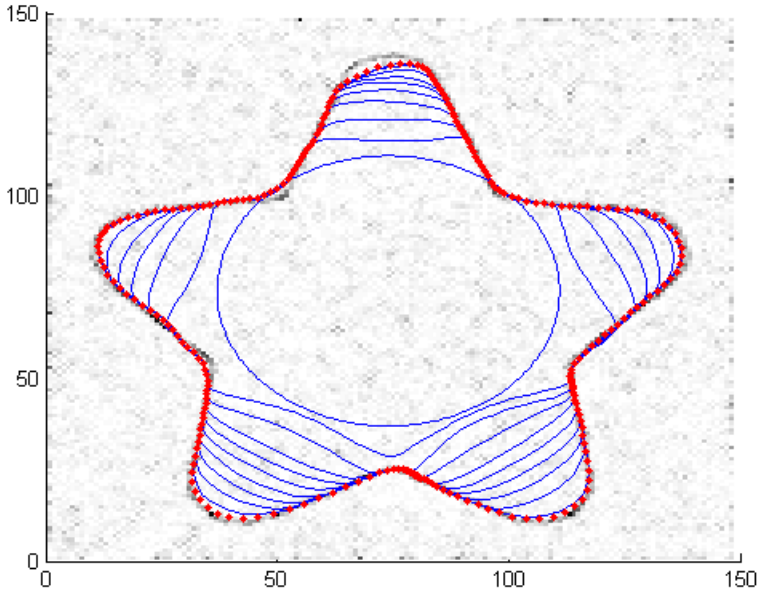

Figure 5. The final active contour.

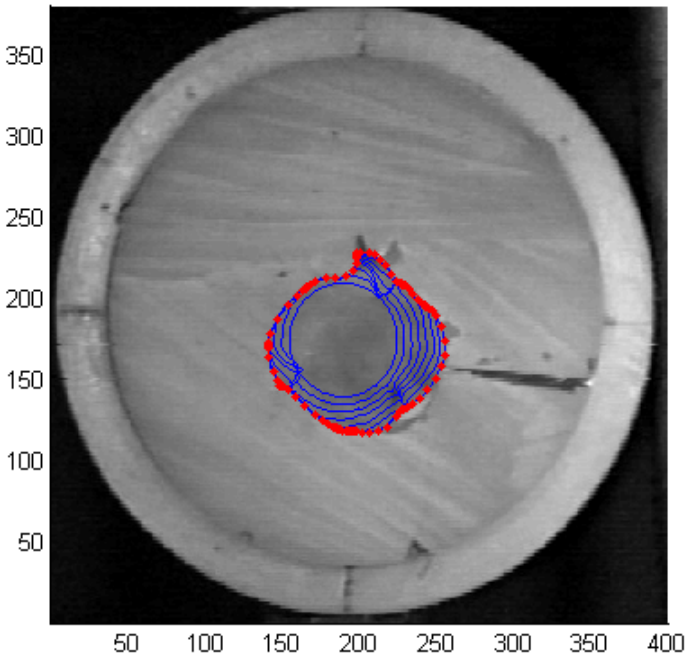

Figure 6. Using the program developed in this work to determine the edges of a heart slice

CHAPTER 2

Active Contour Mechanics

2.1 Internal Energy

The internal energy is defined in such a way that it is minimized when the active contour is smooth. This will prevent the resulting contour from being jagged, which makes sense since most edges of interest are smooth. There are two common ways of representing an active contour and each way has a different way of defining the internal energy. The first way is an active contour can use a list of points to define the active contour and the internal energy is minimizing an approximation of the first and second derivative of the contour. This is the method used in this paper. The second method is by using splines or parametric equations to represent the active contour [7] [1]. The internal energy of the active contour is then created from the constraints placed on it by the spline.

Since we are using a list of points for the active contour, we will define by two scaling factors which weigh the first and second derivatives of the active contour to create the internal energy. Conventionally, these weighting factors are α for the first derivative and β for the second derivative. The Terzopoulos (KWT) active contour model defines the internal energy as [1]:

$$E_{\text{int}}(X,Y) = \frac{1}{2}\int_0^1 \alpha\left[\left|\frac{dX}{ds}\right|^2 + \left|\frac{dY}{ds}\right|^2\right] + \beta\left[\left|\frac{d^2X}{ds^2}\right|^2 + \left|\frac{d^2Y}{ds^2}\right|^2\right] ds \qquad (1)$$

This definition of the active contours results in an internal energy that is minimized when the first and second derivative are minimized, which occurs when the active contour is smooth.

With the internal energy defined, we can calculate the internal energy force. This force needs to "push" the active contour to reach a state with minimal internal energy. This is done by taking the derivative of the internal energy with respect

to the equation of the active contour. Just like the derivative of a function, the derivative in respect to an equation produces a gradient which can be used to find a direction to a local minima. Taking the derivative of this internal energy in Equation 1 results in the following internal forces (Derivation in Appendix B):

$$F_{\text{int X}}(X) = \alpha\frac{\partial^2 X}{\partial s^2} - \beta\frac{\partial^4 X}{\partial s^4} \tag{2a}$$

$$F_{\text{int Y}}(Y) = \alpha\frac{\partial^2 Y}{\partial s^2} - \beta\frac{\partial^4 Y}{\partial s^4} \tag{2b}$$

2.2 Edge map

Before looking at the external energy of an image, it is worth looking at the edge map, $I_{\mathbf{edge}}(x, y)$. The edge map is a two dimensional set of values that measure the change in the gradient in the image. The assumption being made here is that there is a quick change in color between the object and the background that occurs at the edge of the object. Many papers define the negative edge map as [1]:

$$I_{\text{edge}}(x, y) = |\nabla I(x, y)|^2 \tag{3}$$

A demonstration of the edge map is given in Figure 7. Instead of focusing on the whole image, the figure will demonstrate the edge map on a one dimensional case. The upper right panel of Figure 7 shows the pixel grayscale value between 0 (white) and 1 (black) from pixel row 110 of the upper left panel. The edge map of this row is shown in the lower left panel of Figure 7. As we can see, the peaks of the edge map at pixel index 50 and 79 correspond with the large change in pixel values which correspond with the edges of the image.

There are other methods that exist to find an edge map of the image. One of these approaches is the Canny filter which optimizes the edge map detection based on assumptions on the size of the object of interest and the amount of noise present in the image [8].

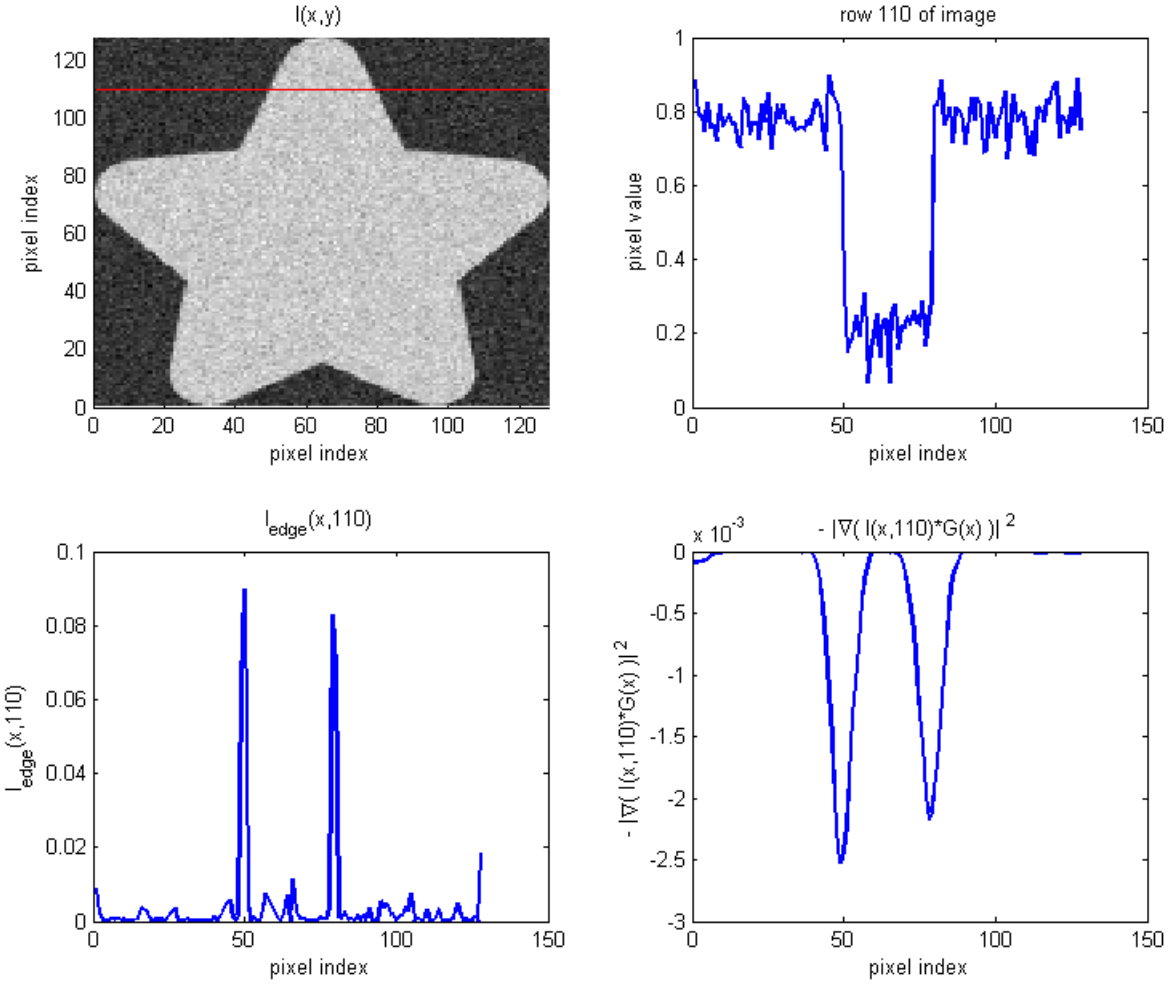

Figure 7. (Upper Left) Image with row 110 highlighted in red. (Upper Right) Pixel values of image row 110 with grayscale pixel values between 0 and 1. (Lower Left) Edge map of image row 110 using Equation 3. (Lower Right) External Energy of image row 110 using the KWT method [1].

2.3 External Energy

The external force "pushes" the active contour points to the edge of the object of interest. There are many different ways of defining this external energy. This section is going to look at one of the earliest methods of determining the external energy, the Marr method [9].

The Marr method was developed by D. Marr [9]. The Marr method uses a definition of the external energy and then finds the derivative of this function to find the external force. The external energy in this case is defined in such a way that it is minimized when the active contour sits near the edge of the feature of interest:

$$E_{\text{ext}}(x,y) = -\Big|\nabla\big[G(x,y) * I(x,y)\big]\Big|^2 \tag{4}$$

where $I(X,Y)$ is the image, $G(x,y)$ is a Gaussian blur, $*$ is the 2-D convolution operation, ∇ is the vector gradient operation, and $|\cdot|$ is the magnitude of the vector. To understand this approach, we first notice that if we remove the Gaussian blur, then this equation would be the same as the negative edge map (Equation 3). There are two effects that the Gaussian blur convolution has on this equation. First, the Gaussian blur spreads out the sharp changes in gradient so that active contour points that are further away from the edge will be "pulled" towards that edge (note the difference in width around pixels 50 and 79 between the two spikes in the bottom panels of Figure 7) [10]. Second, if the image is slightly blurry, the edge can span over multiple pixels with a weaker gradient. The Gaussian blur will combine these gradients in such a way that their total energy will be near the same as a sharp edge that has a large gradient between a few pixels.

An easy way to visualize this phenomenon is to picture the one dimensional version of this external energy as shown in Figure 7. The bottom right panel in Figure 7 shows the results of using the Marr method on row 110 of pixels shown

in the upper right panel. The Gaussian blur used in the bottom right panel has a standard deviation of 5. As mentioned before, the edge is located at pixel indexes 50 and 79. These are the points where the Marr method is minimized but with the added benefit of the wide slow monotonic slope on either side. We can use this definition of the external energy to find an equation for the external force by taking the negative gradient of the external energy. This will "push" the active contour points towards the minima, which hopefully is an edge in the image. The resulting equation for the external force is:

$$\mathbf{F}_{\text{ext}}(x,y) = \nabla\left|\nabla\left[G(x,y) * I(x,y)\right]\right|^2 \tag{5}$$

Since the external force is a vector, it can be broken up into x and y components. In this paper, $F_{\text{ext X}}(x,y)$ is the x component of the external force vector and $F_{\text{ext Y}}(x,y)$ is the y component of the external force vector.

One thing to note about this approach is that the range is limited in the Marr method. For example if the active contour point in the lower right panel of Figure 7 began at pixel index 130, the gradient of the external energy would be near zero. This would mean that there would be almost no external force on that active contour point. One way to fix this is to use a wider Gaussian blur which would cause the points from further away to be drawn in, but then the forces near the edge will be weakened [10]. This can be seen in the 2-D visualization of the Marr method shown in Figure 8. Note how the vectors in the left panel are really small in the immediate vicinity of the edge while the vectors in the right panel are large in the immediate vicinity of the edge. One way to overcome this issue is to use multiple external energy maps when updating the active contour. The first external energy maps will use a wide Gaussian blur (high σ) to "pull" the active contour points nearby and final external energy maps will use a narrow Gaussian blur (low σ) to "pull" the active contour points to the edge [10]. This change in

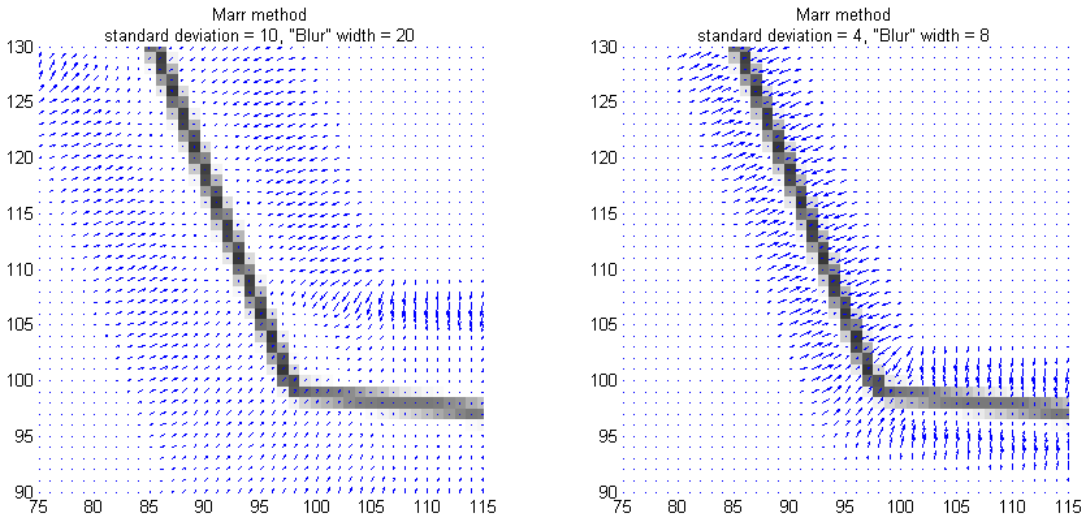

Figure 8. The external energy using the Marr method. The black line is the edge map of the image and the blue vectors indicate the direction and magnitude of the external force. The left panel uses a Gaussian blur with a standard deviation of 10 while the right panel uses a Gaussian blur with a standard deviation of 4. Note how the Marr method that uses Gaussian blur with a large standard deviation does not create large external energies near the edge.

the external energy will increase the effective range and accuracy of the external force, but it is still limited in how far the force for an edge can reach before being lost in the noise. There are other ways of defining the external energy force, and more of these approaches will be discussed in the next chapter.

Another thing to note about this approach is that the external energy is typically pre-calculated at the pixel locations of the image but the active contour points can fall anywhere on the image, even between pixels. This means that when the active contour is being updated, the external forces on the active contour points will have to be interpolated. For this paper and for the program developed in this work, the bi-linear interpolation is used.

2.4 Kass, Witkin, and Terzopoulos (KWT) Method

By combining these definitions of the internal and external energy, the KWT active contour approach is created [1]. For the rest of this paper, the KWT active contour approach will be called the KWT method which is defined as [10] [11]:

$$\frac{\partial X}{\partial \tau} = F_{\text{int X}} + F_{\text{ext X}} = \alpha \frac{\partial^2 X}{\partial s^2} - \beta \frac{\partial^4 X}{\partial s^4} + F_{\text{ext X}}(X, Y) \tag{6a}$$

$$\frac{\partial Y}{\partial \tau} = F_{\text{int Y}} + F_{\text{ext Y}} = \alpha \frac{\partial^2 Y}{\partial s^2} - \beta \frac{\partial^4 Y}{\partial s^4} + F_{\text{ext Y}}(X, Y) \tag{6b}$$

To solve the KWT method we use a finite difference definition for the first and second derivatives in Equation 6. Then evolve the solution iteratively using a time step zeta (ζ). The resulting equations are shown in Equation 7 [1]. The subscript denotes the pixel location and the superscript denotes the time.

$$\begin{aligned} \frac{X_i^{t+1} - X_i^t}{\zeta} &= \alpha(X_{i+1}^t - 2X_i^t + X_{i-1}^t) \\ &\quad - \beta(X_{i+2}^t - 4X_{i+1}^t + 6X_i^t - 4X_{i-1}^t + X_{i-2}^t) + F_{\text{ext X}}(X_i^t, Y_i^t) \end{aligned} \tag{7a}$$

$$\begin{aligned} \frac{Y_i^{t+1} - Y_i^t}{\zeta} &= \alpha(Y_{i+1}^t - 2Y_i^t + Y_{i-1}^t) \\ &\quad - \beta(Y_{i+2}^t - 4Y_{i+1}^t + 6Y_i^t - 4Y_{i-1}^t + Y_{i-2}^t) + F_{\text{ext Y}}(X_i^t, Y_i^t) \end{aligned} \tag{7b}$$

The iterative approach to solve this is then defined in Equation 8. The time step zeta (ζ) can be increased to reduce the number of iterations required to reach the minimum energy, but then the active contour points can oscillate around the minima. Oscillations occur more often when using external energy methods that have strong forces around the edges [2]. This effect can be reduced by reducing

the time step in later iterations.

$$X_i^{t+1} = X_i^t + \zeta\Big[\alpha(X_{i+1}^t - 2X_i^t + X_{i-1}^t) - \beta(X_{i+2}^t - 4X_{i+1}^t + 6X_i^t - 4X_{i-1}^t + X_{i-2}^t) + F_{\text{ext X}}(X_i^t, Y_i^t)\Big] \tag{8a}$$

$$Y_i^{t+1} = Y_i^t + \zeta\Big[\alpha(Y_{i+1}^t - 2Y_i^t + Y_{i-1}^t) - \beta(Y_{i+2}^t - 4Y_{i+1}^t + 6Y_i^t - 4Y_{i-1}^t + Y_{i-2}^t) + F_{\text{ext Y}}(X_i^t, Y_i^t)\Big] \tag{8b}$$

CHAPTER 3

External Energy

There are many different ways to define the external force used in active contours. The Marr approach was described in the previous chapter. In this chapter, we will explore two other ways of defining the external energy and take a look at the properties these methods have.

3.1 Gradient Vector Flow (GVF)

The Gradient Vector Flow (GVF) is a method that uses the solution to a pair of decoupled linear partial differential equations to solve for the external force [12]. These equations are designed to add a smoothing function to the external force to extend the range of strong external forces into areas that have weak external energies. This solves that problem with the Marr method where the initial definition of the external force was limited in range by the Gaussian function used. The GVF method uses a smoothing weight μ to scale the smoothing effect to the amount of noise in the image. For images with more noise, a larger μ is used [12]. The smoothing weight μ is bounded between 0 and 1 for stability.

Since this approach is an iterative process, it requires an initial external energy definition. Any external energy can be used [12]. In the program developed in this work, the Marr approach to the external force is used to initialize the GVF method. The Gaussian blur used in the program has variance $\sigma = 1$.

The GVF method minimizes the following equation to smooth the external

force [12] :

$$\varepsilon = \frac{1}{2}\int\int \mu\left[\left(\frac{dF_{\text{ext X}}(x,y)}{dx}\right)^2 + \left(\frac{dF_{\text{ext X}}(x,y)}{dy}\right)^2 + \left(\frac{dF_{\text{ext Y}}(x,y)}{dx}\right)^2 + \left(\frac{dF_{\text{ext Y}}(x,y)}{dy}\right)^2\right] + \left|\mathbf{F}_0(x,y)\right|^2\left|\mathbf{F}_{\text{ext}}(x,y) - \mathbf{F}_0(x,y)\right|^2 dxdy \tag{9}$$

where the initial external force is $\mathbf{F}_0(x,y)$ and the external force being solved for is $\mathbf{F}_{\text{ext}}(x,y)$.

As we can see from this equation, in the areas where the external force is weak, $\left|\mathbf{F}_0(x,y)\right|^2$ is near zero and the equation is mainly a smoothing function. In areas with high external force, $\left|\mathbf{F}_0(x,y)\right|^2$ is large and the equation is dominated by $\left|\mathbf{F}_{\text{ext}}(x,y) - \mathbf{F}_0(x,y)\right|^2$ which is minimized when $\mathbf{F}_{\text{ext}}(x,y) \approx \mathbf{F}_0(x,y)$.

The solution to minimize the above equation can be solved with calculus of variations to develop an iterative solution (Derivation in Appendix C). The resulting discrete equations are shown in Equation 10. Since these equations are decoupled, they can be solved independently. This means that the GVF is easily expanded to additional dimensions and it can easily be processed in parallel. The external force created using the GVF method with $\mu = 0.3$ is shown in Figure 9.

$$F_{\text{ext X}}(x,y)_{n+1} = \left[1 - \left|\mathbf{F}_0(x,y)\right|^2\right]F_{\text{ext X}}(x,y)_n + \frac{\mu}{4}\nabla^2 F_{\text{ext X}}(x,y)_n + \left|\mathbf{F}_0(x,y)\right|^2 F_{0\text{ X}}(x,y) \tag{10a}$$

$$F_{\text{ext Y}}(x,y)_{n+1} = \left[1 - \left|\mathbf{F}_0(x,y)\right|^2\right]F_{\text{ext Y}}(x,y)_n + \frac{\mu}{4}\nabla^2 F_{\text{ext Y}}(x,y)_n + \left|\mathbf{F}_0(x,y)\right|^2 F_{0\text{ Y}}(x,y) \tag{10b}$$

where ∇^2 is the Laplace operator approximated as:

$$\nabla^2 F_{\text{ext X}}(x,y)_n = F_{\text{ext X}}(x+1,y)_n + F_{\text{ext X}}(x,y+1)_n + F_{\text{ext X}}(x-1,y)_n + F_{\text{ext X}}(x,y-1)_n - 4F_{\text{ext X}}(x,y)_n \tag{11}$$

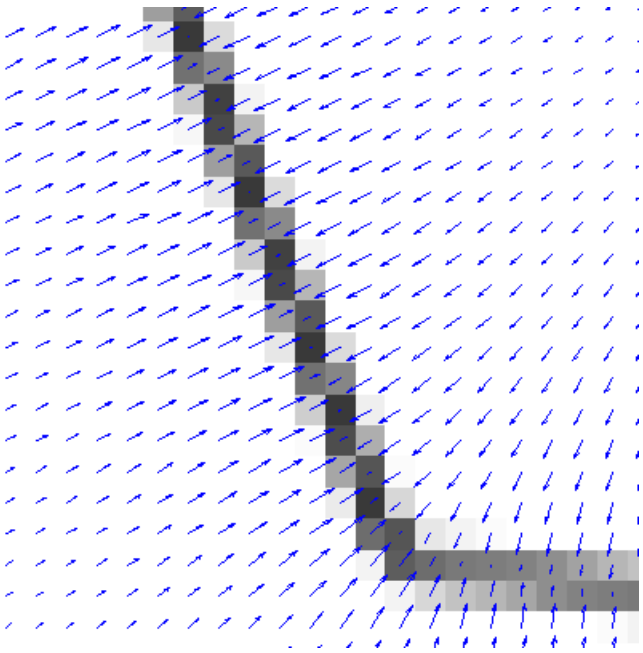

Figure 9. The external force using the GVF with $\mu = 0.3$. The black line is the edge map of the image and the blue vectors indicate the direction and magnitude of the external force.

3.2 Vector Field Convolution

Another method of defining the external force is using Vector Field Convolution (VFC). This approach takes the edge map from of the image and convolves it with a vector field kernel. This vector field is defined by the user, but it needs to have specific properties [13] such that all of the vectors point towards the center of the vector field kernel and the magnitude of the vectors closer to the center are greater than the magnitude of the vectors at the edges. This produces a definition for the external force where the external vectors are a weighted sum of the surrounding edge map with closer edges and larger edge maps are weighed more. Since the vector field kernel is defined by the user, they have a lot of control over the field of influence of a single value on the edge map. In [13], Li proposes two

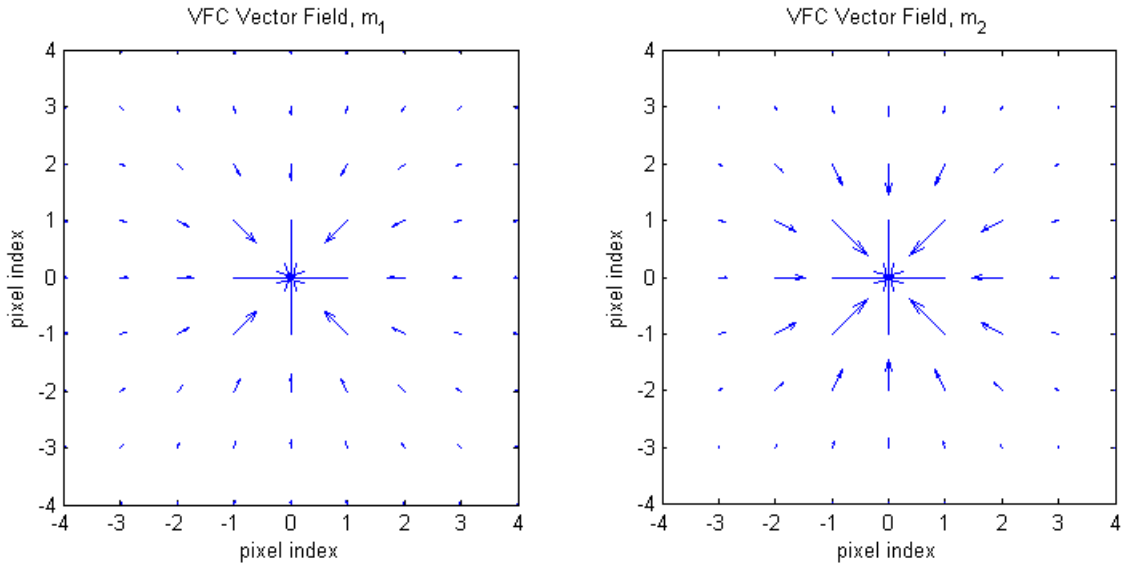

Figure 10. (Left Panel) VFC vector field kernel using the magnitude function from Equation 12, $\epsilon = 0.01$ and $\gamma = 2$. (Right Panel) VFC vector field kernel using the magnitude function from Equation 13, $\xi = 2$.

types of vector magnitude functions:

$$m_1(x, y) = (r + \epsilon)^{-\gamma} \tag{12}$$

$$m_2(x, y) = \exp(-r^2/\xi^2) \tag{13}$$

where r is the distance from the center of the kernel to the current pixel, ϵ is a small number to avoid dividing by zero and γ and ξ scale the field of influence. An image with more noise should use a vector field kernel with a larger field of influence (smaller γ, larger ξ) to smooth out the noise [13]. Figure 10 shows an example of two vector fields by using the two magnitude functions from Equations 12 and 13.

The program developed in this work uses only Equation 13 to implement the VFC. Using this method produces an external vector field like the one shown in Figure 11.

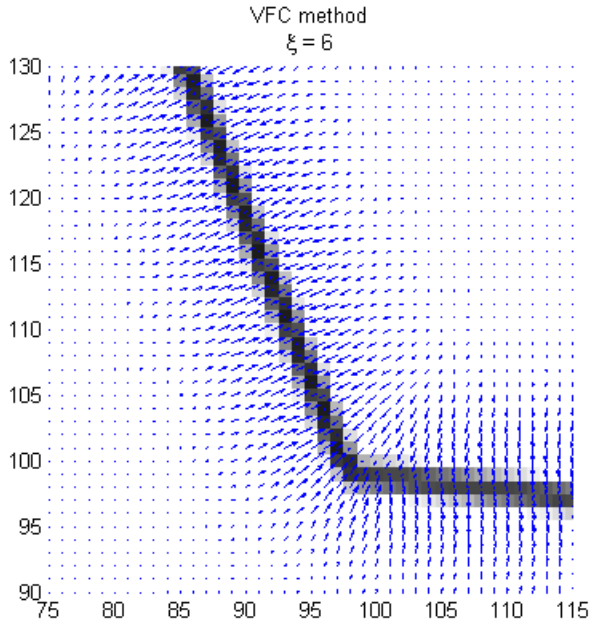

Figure 11. The external force using the VFC with $\xi = 6$ in Equation 13. The black line is the edge map of the image and the blue vectors indicate the direction and magnitude of the external force.

CHAPTER 4

Active Contours Software

4.1 Active Contours Software Introduction

The software in this work is written in Matlab 2015b using GUIDE for the graphical user interface (GUI). The Matlab code is included in Appendix A with all of the GUIDE initializations and callbacks omitted. This program has the ability to load an image, convert it to grayscale, set up an initial active contour on the image, and show the active contour evolution using the KWT approach and the Gradient Vector Flow approach. The field descriptions are provided in Table 1.

An example of this interface is shown in Figure 12. The blue lines are the intermediate active contours and the red dots are the resulting active contour nodes.

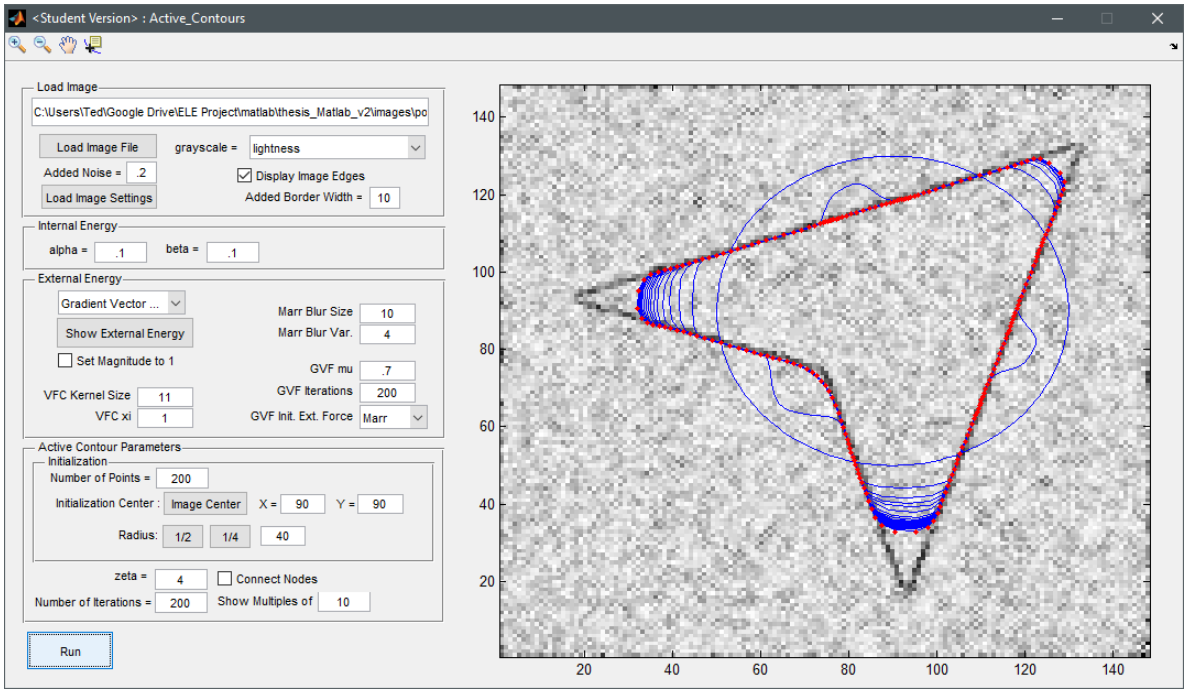

Figure 12. Demonstration of Active Contour Program.

Field	Description
Load Image	
select file	Option to type the image location to use
Load Image File	Uses a file picker to load an image
grayscale	Select the method to convert color images into grayscale
Added Noise	Add noise to the image as Gaussian noise (value is sigma). The range is then resized to between 0 and 1. The picture does not need to be square
Display Image Edges	Display the edge map instead of the image
Add Border Width	Add a border around the image with the provided width. The surrounding boarder is white which will add an edge to the image if the background of the image is not white. This is in case there is an image with an edge that skims the border.
Load Image Settings	Reload the settings in the Load Image section. This will also reseed the added noise.
Internal Energy	
alpha (α)	Scales the first derivative of the active contour
beta (β)	Scales the second derivative of the active contour
External Energy	
(pulldown menu)	Choose the method for the external energy (Marr, Vector Field Convolution, or Gradient Vector Field)
Set Magnitude to 1	Keep the angle of the external force calculated and set the magnitude of all vectors to 1. (Vectors of zero magnitude remain at 0 magnitude)
VFC Kernel Size	Change the size of the vector field kernel that is applied in the Marr approach (Figure 10). Entering n results in an 2n by 2n vector field kernel.
VFC xi (ξ)	Scaling factor when using the VFC approach. This scaling the field of influence of the VFC. (Equation 13)
Marr Blur Kernel Size	Change the size of the Gaussian blur that is applied in the Marr approach (Equation 5). Entering n results in an 2n by 2n 2-D Gaussian blur.
Marr Blur Variance	Change the variance of the Gaussian blur that is applied in the Marr approach (Equation 5)
GVF mu (μ)	Scaling factor when using the GVF approach. Weights the trade off between smoothing and inital external force for use in noisy images. More noise, higher mu. $0 < mu < 1$ (Equation 10)
GVF Iterations	Max iterations used in calculating the Gradient Vector Field. (Equation 10)
GVF Init. Ext. Force	Chose which approach is used when initializing the GVF approach. (Equation 10)

Show External Energy	Displays the external energy on the image using vectors. It is recommended to use this with Display Image Edges
Other fields	
zeta (ζ)	Finite difference time step for active contour iterations
Connect Nodes	Connect the nodes in the last iteration of the active contour
Number of Iterations	Number of times to update the active contour
Show Multiples of	Only display the active contours that are multiples of this value
Run	Run the active contour using the settings provided in the other fields

Table 1: List of variables

This program processes grayscale images. If the image provided is not grayscale, there are a few different methods included to convert the color image into a grayscale image. There are methods of finding a weighted average of the red, green and blue values of a color image that give the greatest distinctions between the regions to make the edges more defined [7]. The three methods used in this work to convert color images to grayscale are lightness (Equation 14), average (Equation 15) and luminosity (Equation 16) [14]. Each of these methods convert the red, green and blue (RGB) values of the image into a grayscale image.

$$\text{lightness: } I(x,y) = \frac{\max(R,G,B) + \min(R,G,B)}{2} \tag{14}$$

$$\text{average: } I(x,y) = \frac{R+G+B}{3} \tag{15}$$

$$\text{luminosity: } I(x,y) = 0.21R + 0.72G + 0.07B \tag{16}$$

The "Show External Energy" button displays the external force vectors on top of the image. It is recommended to use the "Display Image Edges" in the Load Image section to clearly see where the edges are. The number of vectors displayed is always limited to no more than 200 in any row or column in order to not clutter the image or overload the plot. If there are more than 200 pixels in the image

(an therefor more than 200 external force vectors) the program will automatically down sample in both dimensions to every other pixel recursively until there is less than 200 external force vectors to display in both directions.

4.2 Active Contours Software Performance

To test the program developed in this work, a test image was used to get a feel for the performance using different methods. The following test image was created to test the active software program:

Figure 13. Image to test the program developed in this work.

The notable features of this test image is the protrusion on top, the blurred corner in the bottom left, and the hard corner on the right. The curved protrusion on top which can be difficult for active contours to form to because the external energy wants to "pull" the active contour points to one of the side walls [12]. This effect can also be seen in Figure 12. The bottom left corner of this image is blurred so there is no hard edge. The slow gradient will make the edge map weaker in this area which can affect the performance. Finally, the edge on the right has sharp

edges which can be difficult for active contours to form to because the internal energy will resist sharp changes [1]. The image will also have Gaussian noise (standard deviation = 0.1) added when it is processed to add some variability.

For each approach, there are parameters that can be tweaked and each will change the performance of the active contour. The application of the active contour will also have a big impact on performance. For example in tracking, the active contour will typically initialize in the next frame of the video displaced by the current calculated object velocity [6]. This means that the active contour should be close to the desired edge which is an easier case than if it is far away.

All of the following tests use a 300 point active contour initialized in the same location and iterated 300 times with time step, zeta, set to $\zeta = 2.0$. Every tenth active contour is shown in blue and the final active contour points are shown in red.

Using the Marr external energy force with a Gaussian blur variance of 7 produced the following results:

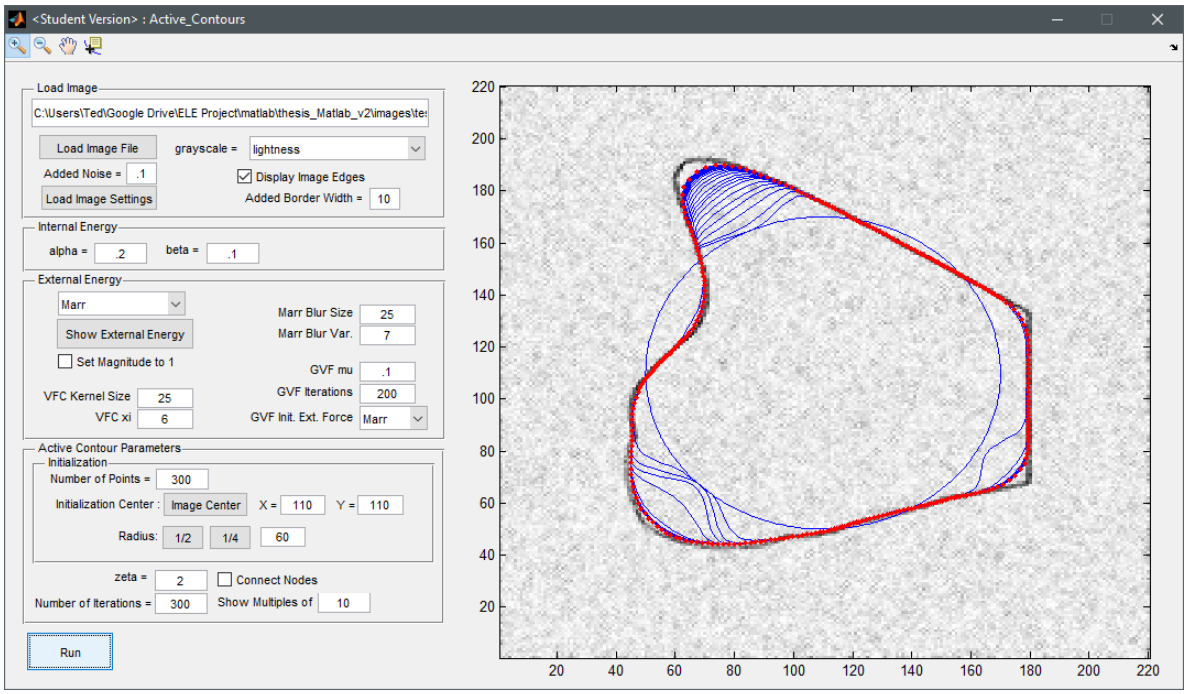

Figure 14. Marr external energy test using the test image in Figure 13

The high variance is needed to pull the active contours from further away. The high variance causes the corners to be rounded because the strength of the active contours is weak in the immediate vicinity of the edge (see Figure 8). The active contour is also very slow to move to the protrusion in the top of the image.

Using the VFC external energy force with Equation 13 and $\xi = 6$ produced the following results:

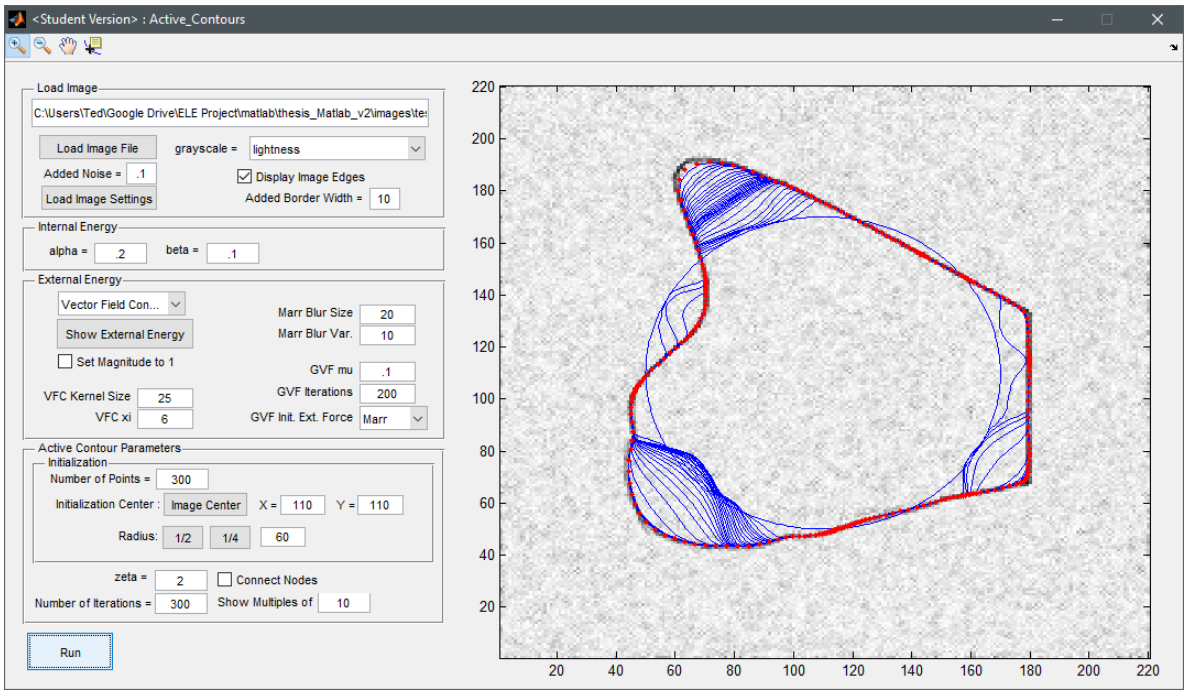

Figure 15. VFC external energy test using the test image in Figure 13

From the definition of the vector field kernel, the external energy increases its field of influence while maintaining a strong external force in the immediate vicinity of the edge (see Figure 11). This gives better active contour performance at the corners of the image and in the protrusion. The active contour has some trouble with the blurred edge but with enough iterations, the points eventually get "pulled" to the edge.

The figure below uses the GVF with a Marr initial external energy. The initial Marr external energy uses a standard deviation of 2. The initial external energy is iterated though the GVF approach 200 times with mu set to $\mu = 0.7$.

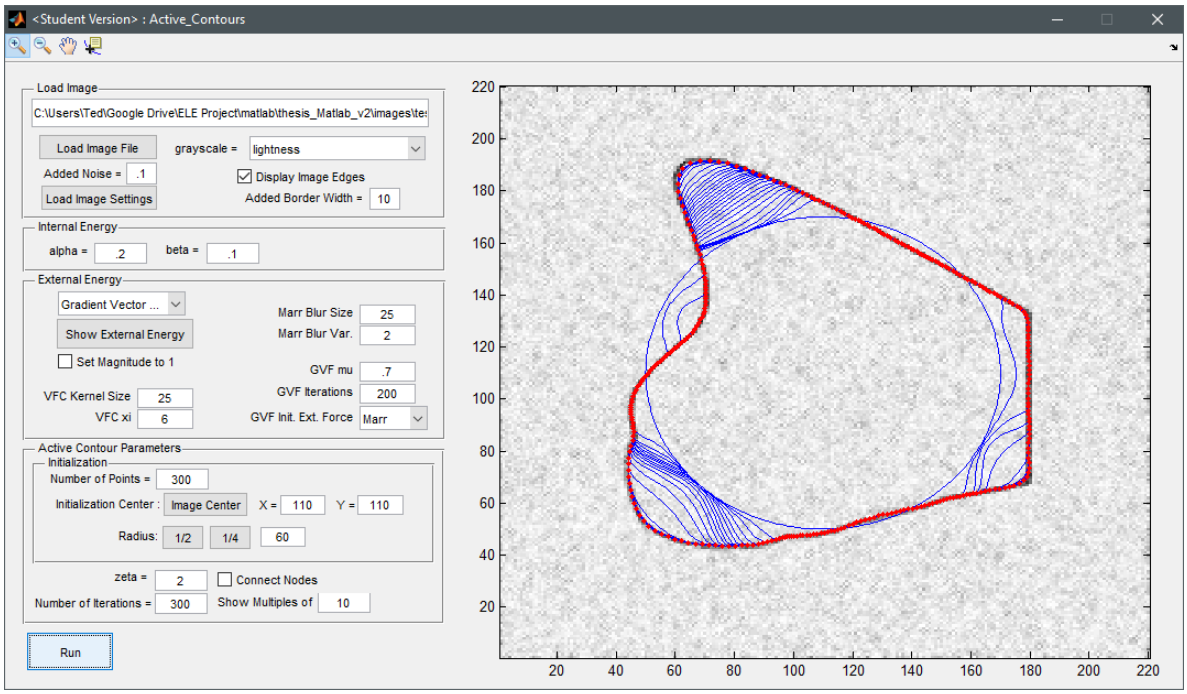

Figure 16. GVF external energy test with an initial external energy using the Marr approach using the test image in Figure 13.

This approach produced good results in the protrusion, in the blurred area, and in the hard corners. Both corners have good performance because the GVF has strong external energy forces in the immediate vicinity of the edges from using a Marr initial external energy using a Gaussian blur with a low standard deviation. The protrusion has good performance because the smoothing function of the GVF causes the strong external energy forces to spread into the center of the image where there is little initial external energy (as demonstrated in Figure 9).

CHAPTER 5

Future Work

This paper has established the foundation for a variety of active contour techniques. In this section, we will explore the different modifications that could be made to the active contour approaches explained in this paper.

5.1 Splines and Models

As explained in section 2.1, the internal energy of the active contour can be replaced with a spline that uses parametric equations to define a continuous curve [7] [1]. Depending on how the spline is defined, different restrictions can be placed on the active contour. For example, the spline chosen could have a continuous first or second derivative.

Another way to define the internal energy is by using a set of equations. This transforms the active contour methods into a way of matching a model defined by the equations to an image [7]. An example of this approach is to use a model for a person's eyes determine the direction of their gaze or to use a model of a person's mouth to determine if they are smiling [15].

5.2 Active Surfaces

All of the approaches explored in this paper can be expanded to three dimensions [1] [12] [13]. An active surface can be used to form to the edge of a 3-D object from 3-D data to extract information on size and volume. Active surfaces have been used to measure human talus cartilage from a stereophotographic image [16].

5.3 Active Contour Merging, Ballooning Force and GVF Boundary Conditions

The approaches explored in this paper work best when the object of interest has a known approximate location in the image. This allows the active contour to begin near that object and then iterate to reach the edges of the object. If the approximate location of an object is unknown, then the active contour may not initialize in a location that will allow it to reach the object of interest. For example, if the active contour was initialized as a small circle that crossed a strong edge in the image, all of the active contour points could converge to that edge.

One way to solve this problem is to use many small active contours scattered throughout the image and add a non-overpowering expansion force to the external force. As the active contours expand, they will collide with one another in some areas and settle onto object edges in other areas. If two active contours collide, then they are held in place until the end of the active contour iterations. At the end of the active contour iterations, all of the collided active contours are combined into one large active contour that hopefully outlines the object of interest. One way of adding this expansion force is by setting boundary conditions on the GVF method to force the GVF solution to have an external force that points outward from the active contour [1] [4]. Another way of adding this expansion force is by adding a ballooning force, which is an additional force that is normal to the active contour [1].

5.4 Generalized Gradient Vector Flow

The Generalized GVF (GGVF) replaces μ and $\left|\mathbf{F}_0(x,y)\right|^2$ with user defined weighting functions [17]. The GGVF gives the user more control over the trade off between the smoothing function and the edge map to get better performance on images with gaps in the edge map and thin protrusion features [17].

5.5 Tracking

The approaches explored in this paper can be used to track an object in a series of images and keep track of where the active contour moves between images. To improve the ability to track an object, it is useful to add a estimate of the current object velocity and rotation to more accurately initialize the active contour in the next image [6].

5.6 Stopping Criteria

Instead of iterating an active contour a set number of times and hoping that it reached the edge of the object of interest, it would be beneficial to add a stopping criteria to check if the active contours have aligned with the object edges in the image. It may be possible to use the sum of the edge map values of all the active contour points as a stopping criteria. The edge map values of the active contour points would have to be bilinearly interpolated since the edge map is defined at the pixels of the image. As the active contour points line up with the edges of the image, the total edge map value would increase. Once all of the active contour points line up with the edge, the sum of the edge map values should not change significantly.

5.7 Clustering

Sometimes the points in an active contour will cluster together, either because the external force converges or because the points initialize near an edge that "pulls" them all in. It may be possible to avoid this by adding a force that makes the points spread apart from one another. Unlike the ballooning force, this force would be in parallel to the line connecting the active contour points instead of perpendicular to it. The magnitude of this force would have to be inversely proportional to the distance between the points to keep the points separated.

LIST OF REFERENCES

[1] S. T. Acton and N. Ray, *Biomedical Image Analysis: Segmentation.* Morgan and Claypool, 2009.

[2] L. D. Cohen, "On active contour models and balloons," *Computer Vision, Graphics, and Image Processing: Image Understanding*, vol. 53, no. 2, pp. 211–218, March 1991.

[3] J. Ivins and J. Porrill, *Everything You Always Wanted to Know About Snakes (But Were Afraid to Ask)*, Artificial Intelligence Vision Research Unit, University of Sheffield, July 1993.

[4] N. Ray, S. T. Acton, T. Altes, E. E. de Lange, and J. R. Brookeman, "Merging parametric active contours within homogeneous image regions for mri-based lung segmentation," *IEEE Transactions on Medical Imaging*, vol. 22, no. 2, pp. 189–199, February 2003.

[5] P. C. Tay, B. Li, C. D. Garson, S. T. Acton, and J. A. Hossack, "Left ventricle segmentation using model fitting and active surfaces," *Journal of Signal Processing Systems*, vol. 55, pp. 139–156, May 2008.

[6] S. T. Acton and N. Ray, *Biomedical Image Analysis: Tracking.* Morgan and Claypool, 2005.

[7] A. Blake and M. Isard, *Active Contours.* Springer-Verlag London Limited, 1998.

[8] J. Canny, "A computational approach to edge detection," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 8, no. 6, pp. 679–698, November 1986.

[9] D. Marr and E. Hildreth, "Theory of edge detection," *Proceedings of the Royal Society of London. Series B, Biological Sciences*, vol. 207, no. 1167, pp. 187–217, February 1980.

[10] M. Kass, A. Witkin, and D. Terzopoulos, "Snakes: Active contour models," *International Journal of Computer Vision*, vol. 1, no. 4, pp. 321–331, 1987.

[11] T. McInerney and D. Teropoulos, "Deformable models in medial image analysis: A survey," *Medical Image Analysis*, vol. 1, no. 2, pp. 91–108, 1996.

[12] C. Xu and J. L. Prince, "Snakes, shapes, and gradient vector flow," *IEEE Transactions on Image Processing*, vol. 7, no. 3, pp. 359–369, March 1998.

[13] B. Li and S. T. Acton, "Active contour external force using vector field convolution for image segmentation," *IEEE Transactions on Image Processing*, vol. 16, no. 8, August 2007.

[14] J. D. Cook. John D. Cook Consulting. "converting color to grayscale." August 2009. [Online]. Available: https://www.johndcook.com/blog/2009/08/24/algorithms-convert-color-grayscale/

[15] A. L. Yuille, P. W. Hallinan, and D. S. Cohen, "Feature extraction from faces using deformable templates," *International Journal of Computer Vision*, vol. 8, no. 2, pp. 99–111, 1992.

[16] B. Li, S. A. Millington, D. D. Anderson, and S. T. Acton, "Registration of surfaces to 3d images using rigid body surfaces," *Fortieth Asilomar Conference on Signal, Systems, and Computers*, pp. 416–420, 2006.

[17] C. Xu and J. L. Prince, "Generalized gradient vector flow external forces for active contours," *Signal Processing*, vol. 71, pp. 131–139, 1998.

APPENDIX A

Matlab script

```
% Omitted all GUIDE Callbacks and initializations

function img = DisplayImage(handles, force)
%check if the file has changed
persistent currentFileName currentImg

fileName = get(handles.selectedFile, 'String');
if ~strcmp(fileName, currentFileName) || force %image has changed or update is forced
    currentFileName = fileName;
    try
        [img, map] = imread(fileName);
    catch
        disp('unable to read file')
        return
    end
    img = flipud(img); %flip image due to image convention
    %convert to grayscale
    if ~isempty(map)
        img = ind2rgb(img,map);
    end

    img = double(img);
    if size(img,3)==3
        grayscaleMethod = cellstr(get(handles.grayscaleMethod,'String'));
        grayscaleMethod = grayscaleMethod{get(handles.grayscaleMethod,'Value')};
        switch grayscaleMethod
            case 'lightness'
                img = (max(img,[],3)+min(img,[],3))/2;
            case 'average'
                img = (img(:,:,1) + img(:,:,2) + img(:,:,3))/3;
            case 'luminosity'
                img = 0.21 * img(:,:,1) + 0.72 * img(:,:,2) + 0.07 * img(:,:,3);
        end
```

```
34      end
35
36      img = (img - min(img(:)))/(max(img(:))-min(img(:)));
37
38      % add Boarder
39      bw = str2double(get(handles.editAddBoarder,'String'));
40      if bw > 0
41          img = [ones(bw,size(img,2)+2*bw);ones(size(img,1),
                bw), single(img), ...
42              ones(size(img,1),bw);ones(bw,size(img,2)+2*bw)
                ];
43      end
44
45      %add noise
46      noise = str2double(get(handles.editAddedNoise,'String')
            );
47      if noise > 0
48          img = img + noise*randn(size(img));
49          % re-normalize
50          img = (img - min(img(:)))/(max(img(:) - min(img(:))
                ));
51      end
52
53      %store image for next time
54      currentImg = img;
55  else %image has not changed and update is not forced
56      img = currentImg;
57  end
58
59  %display image
60  if get(handles.displayEdges,'Value') %chesk if display
        edges is checked
61      [gradX, gradY] = gradient(img);
62      imgEdge = double(sqrt(gradX.^2 + gradY.^2));
63      imgEdge = imgEdge / max(imgEdge(:));
64      axes(handles.mainDisplay)
65      hold off, imagesc(imgEdge,[0,1]), set(gca,'YDir','
            normal')
66      colormap(flipud(gray)), hold on
67  else
68      axes(handles.mainDisplay)
69      hold off, imagesc(img,[0,1]), set(gca,'YDir','normal')
70      colormap(gray), hold on
71  end
```

```
72
73 function DisplayExternalEnergy(handles)
74 img = DisplayImage(handles, 0);
75 [Fx, Fy] = CalcExternalEnergy(handles, img);
76 scale = 1;
77 while max(size(Fx))>200
78     Fx = downsample(Fx,2);
79     Fx = downsample(Fx',2)';
80     Fy = downsample(Fy,2);
81     Fy = downsample(Fy',2)';
82     scale = scale*2;
83 end
84 [u,v] = meshgrid(1:scale:size(img,2),1:scale:size(img,1));
85 axes(handles.mainDisplay)
86 hold on, quiver(u, v, Fx, Fy), hold off
87
88 function StartActiveContours(handles)
89 img = DisplayImage(handles, 0);
90 [Fx, Fy] = CalcExternalEnergy(handles, img);
91
92 alpha = str2double(get(handles.alphaInput,'String'));
93 beta = str2double(get(handles.betaInput,'String'));
94 zeta = str2double(get(handles.zetaInput,'String'));
95 T = str2double(get(handles.itterNum,'String'));
96 TMulti = str2double(get(handles.itterMulti,'String'));
97 N = str2double(get(handles.numberOfPoints,'String'));
98
99 % Calculate the A matrix
100 A = sparse(1:N,1:N,6*beta+2*alpha) + sparse([2:N, 1, 1:N
    ],[1:N, 2:N, 1], -1*(4*beta + alpha)) ...
101     + sparse([3:N, 1, 2, 1:N],[1:N, 3:N, 1, 2],beta);
102 A = full(A);
103
104 % Initialize
105 Xpos = str2double(get(handles.editInitalX,'String'));
106 Ypos = str2double(get(handles.editInitalY,'String'));
107 radius = str2double(get(handles.editInitialRadius,'String')
    );
108
109 % generate the intial active contour as a circle
110 n = [1:N]'; %#ok<NBRAK>
111 X = radius * sin((n/(N+1))*(2*pi));
112 X = X + Xpos;
113 Y = radius * cos((n/(N+1))*(2*pi));
```

```
114 Y = Y + Ypos;
115
116 axes(handles.mainDisplay) %set the main display as active
117 hold on, plot([X; X(1)],[Y; Y(1)],'b') %plot the initial
    active contour
118 drawnow
119 for t = 1:T
120     X_force = interp2(Fx, X, Y);
121     Y_force = interp2(Fy, X, Y);
122     X = (eye(N)+zeta*A)^-1*(X+zeta*X_force);
123     Y = (eye(N)+zeta*A)^-1*(Y+zeta*Y_force);
124     X(X<0)=0;
125     Y(Y<0)=0;
126     X(X>size(img,2))=size(img,2);
127     Y(Y>size(img,1))=size(img,1);
128     if TMulti>0 && rem(t,TMulti)==0 && t~=T
129         hold on, plot([X; X(1)],[Y; Y(1)],'b') %if multiple
              of TMulti, plot in blue
130         drawnow
131     end
132 end
133 if get(handles.checkConnectNodes,'Value')
134     hold on, plot([X; X(1)],[Y; Y(1)],'.-g','linewidth',2,'
        MarkerEdgeColor','red','MarkerSize',8)
135 else
136     hold on, plot([X; X(1)],[Y; Y(1)],'.','MarkerEdgeColor'
        ,'red','MarkerSize',8)
137 end
138
139 function [Fx, Fy] = CalcMarrExternalEnergy(handles,img)
140 kSize = str2double(get(handles.editMarrBlurSize,'String'));
141 kVar = str2double(get(handles.editMarrBlurVar,'String'));
142 h = fspecial('gaussian',[kSize kSize],kVar);
143 imgBlur = imfilter(img,h,'replicate');
144 imgBlur = imgBlur/max(imgBlur(:));
145 [Xf, Yf] = gradient(imgBlur);
146 [Fx, Fy] = gradient(sqrt(Xf.^2 + Yf.^2));
147
148 %normilze to max force = 1
149 Fmax = max(abs(sqrt(Fx(:).^2 + Fy(:).^2)));
150 Fx = Fx./Fmax;
151 Fy = Fy./Fmax;
152
153
```

```
154 function [Fx, Fy] = CalcVFCExternalEnergy(handles,img)
155 kSize = str2double(get(handles.editVFCSize,'String'));
156 xi = str2double(get(handles.editVFCxi,'String'));
157 kSize = floor(kSize/2);
158
159 [u,v] = meshgrid(-1*kSize:kSize,-1*kSize:kSize);
160 angle = atan2(v,u);
161
162 r = sqrt(u.^2 + v.^2);
163 m = exp(-1*r.^2/xi^2);
164 m(r==0)=0;
165 % m = m/sum(m(:));
166
167 m_x = m.*cos(angle);
168 m_y = m.*sin(angle);
169
170 [gradX, gradY] = gradient(img);
171 edge = sqrt(gradX.^2 + gradY.^2);
172 edge = edge/max(edge(:));
173
174 Fx = imfilter(edge,m_x,'replicate');
175 Fy = imfilter(edge,m_y,'replicate');
176
177 %normilze to max force = 1
178 Fmax = max(abs(sqrt(Fx(:).^2 + Fy(:).^2)));
179 Fx = Fx./Fmax;
180 Fy = Fy./Fmax;
181
182 function [Fx, Fy] = CalcGVFExternalEnergy(handles,img)
183 mu = str2double(get(handles.editGVFmu,'String'));
184 itter = str2double(get(handles.editGVFIterations,'String'))
      ;
185
186 %Initialize External Force
187 initialExternalForce = cellstr(get(handles.popupmenuGVF,'
      String'));
188 initialExternalForce = initialExternalForce{get(handles.
      popupmenuGVF,'Value')};
189 switch initialExternalForce
190     case 'Marr'
191         [iFx, iFy] = CalcMarrExternalEnergy(handles,img);
192     case 'VFC'
193         [iFx, iFy] = CalcVFCExternalEnergy(handles,img);
194 end
```

```
195
196 u = iFx; v = iFy; % Initialize GVF with the gradient
197 sqrMagF = iFx.*iFx + iFy.*iFy; % Squared magnitude of the
    gradient field
198
199 delt = 0.5; % iterate at half speed for stability
200 for i=1:itter %Calculate the GVF external force
201     u = u + delt*(mu*0.25*myDel2(u) + sqrMagF.*(iFx - u));
202     v = v + delt*(mu*0.25*myDel2(v) + sqrMagF.*(iFy - v));
203 end
204
205 %normilze to max force = 1, min force = 0
206 Fmax = max(abs(sqrt(u(:).^2 + v(:).^2)));
207 Fx = u/Fmax;
208 Fy = v/Fmax;
209
210 function [Fx, Fy] = CalcExternalEnergy(handles, img)
211 ExternalEnergyType = cellstr(get(handles.
    externalEnergySelect,'String'));
212 ExternalEnergyType = ExternalEnergyType{get(handles.
    externalEnergySelect,'Value')};
213 switch ExternalEnergyType
214     case 'Marr'
215         [Fx, Fy] = CalcMarrExternalEnergy(handles,img);
216     case 'Vector Field Convolution'
217         [Fx, Fy] = CalcVFCExternalEnergy(handles,img);
218     case 'Gradient Vector Flow'
219         [Fx, Fy] = CalcGVFExternalEnergy(handles,img);
220 end
221 if get(handles.checkboxMagEqOne,'Value')
222     mag = sqrt(Fx.^2 + Fy.^2);
223     mag(mag==0)=1; %Fix zero magnitude force
224     Fx = Fx./mag;
225     Fy = Fy./mag;
226 end
227
228 function CenterInitialization(handles, img)
229 set(handles.editInitalX,'String',num2str(size(img,2)/2,'%.2
    g'));
230 set(handles.editInitalY,'String',num2str(size(img,1)/2,'%.2
    g'));
231
232 function [ B ] = myDel2( A )
233 %myDel2 is the discrete laplacian
```

```
234 %   This function performs the discrete laplacian described
        for the
235 %   function del2 under Algrithms.  This method does not
    use linear
236 %   interpolation at the edges.
237 [D1, D2] = size(A);
238 U = [zeros(1,D2);A(1:end-1,:)];
239 D = [A(2:end,:);zeros(1,D2)];
240 L = [zeros(D1,1),A(:,1:end-1)];
241 R = [A(:,2:end),zeros(D1,1)];
242
243 B = U + D + L + R - 4*A;
```

APPENDIX B

Internal Energy Derivation

From the internal energy (Equation 1):

$$E_{\text{int}}(X,Y) = \frac{1}{2}\int_0^1 \alpha\left[\left|\frac{dX}{ds}\right|^2 + \left|\frac{dY}{ds}\right|^2\right] + \beta\left[\left|\frac{d^2X}{ds^2}\right|^2 + \left|\frac{d^2Y}{ds^2}\right|^2\right]ds \tag{B.17}$$

We first separate the X and Y components:

$$E_{\text{int}}(X,Y) = \frac{1}{2}\int_0^1 \alpha\left|\frac{dX}{ds}\right|^2 + \beta\left|\frac{d^2X}{ds^2}\right|^2 ds + \frac{1}{2}\int_0^1 \alpha\left|\frac{dY}{ds}\right|^2 + \beta\left|\frac{d^2Y}{ds^2}\right|^2 ds \tag{B.18}$$

We can minimize the internal energy by minimizing both of the integrals in Equation B.18. The minimization can be done by applying the Euler - Lagrange Formula for a formula with a second derivative:

$$\int F(x,g,g',g'')dx \qquad \text{is minimized when:}$$
$$\frac{\partial F}{\partial g} - \frac{\partial}{\partial s}\frac{\partial F}{\partial g'} + \frac{\partial^2}{\partial s^2}\frac{\partial F}{\partial g''} = 0 \tag{B.19}$$

where g is a function of x and g' is the derivative of g in respect to x.

This is similar to how the minimum of a function, $g(x)$ occurs at a point where $g'(x) = 0$. To find the force that "pushes" the active contour points towards the minimum of Equation B.18, we use gradient decent. We will first focus on the first integral in Equation B.18 which will determine the force in the x direction.

$$F_{\text{int X}}(X,Y) = \frac{1}{2}\times -1 \times \left(\frac{\partial F}{\partial s} - \frac{\partial}{\partial s}\frac{\partial F}{\partial x'} + \frac{\partial^2}{\partial s^2}\frac{\partial F}{\partial x''}\right)$$
$$F_{\text{int X}}(X,Y) = \frac{1}{2}\times -1 \times \left(0 - \frac{\partial}{\partial s}\left(2\alpha\frac{\partial X}{\partial s}\right) + \frac{\partial^2}{\partial s^2}\left(2\beta\frac{\partial^2 X}{\partial s^2}\right)\right)$$
$$F_{\text{int X}}(X,Y) = \alpha\frac{\partial^2 X}{\partial s^2} - \beta\frac{\partial^4 X}{\partial s^4} \tag{B.20a}$$

Similarly for Y:

$$F_{\text{int Y}}(X,Y) = \alpha\frac{\partial^2 Y}{\partial s^2} - \beta\frac{\partial^4 Y}{\partial s^4} \tag{B.20b}$$

APPENDIX C

Gradient Vector Flow Derivation

From the GVF energy equation (Equation 9):

$$\varepsilon = \frac{1}{2}\int\int \mu\left[\left(\frac{dF_{\text{ext X}}(x,y)}{dx}\right)^2 + \left(\frac{dF_{\text{ext X}}(x,y)}{dy}\right)^2 + \left(\frac{dF_{\text{ext Y}}(x,y)}{dx}\right)^2 + \left(\frac{dF_{\text{ext Y}}(x,y)}{dy}\right)^2\right] + \left|\mathbf{F}_0(x,y)\right|^2\left|\mathbf{F}_{\text{ext}}(x,y) - \mathbf{F}_0(x,y)\right|^2 dxdy \tag{C.21}$$

This expands to:

$$\varepsilon = \frac{1}{2}\int\int \mu\left[\left(\frac{dF_{\text{ext X}}(x,y)}{dx}\right)^2 + \left(\frac{dF_{\text{ext X}}(x,y)}{dy}\right)^2 + \left(\frac{dF_{\text{ext Y}}(x,y)}{dx}\right)^2 + \left(\frac{dF_{\text{ext Y}}(x,y)}{dy}\right)^2\right] + \left|\mathbf{F}_0(x,y)\right|^2\Big(F_{\text{ext X}}(x,y) - F_{0X}(x,y)\Big)^2 \times \Big(F_{\text{ext Y}}(x,y) - F_{0Y}(x,y)\Big)^2 dxdy \tag{C.22}$$

We apply the same approach as in appendix B except we will use the Euler Lagrange for two variables:

$$\int\int F(x,y,g,g'_x,g'_y,h,h'_x,h'_y)dxdy \qquad \text{is minimized when:}$$

$$\frac{\partial F}{\partial g} - \frac{\partial}{\partial x}\frac{\partial F}{\partial g'_x} - \frac{\partial}{\partial y}\frac{\partial F}{\partial g'_y} = 0 \quad \text{and} \tag{C.23a}$$

$$\frac{\partial F}{\partial h} - \frac{\partial}{\partial x}\frac{\partial F}{\partial h'_x} - \frac{\partial}{\partial y}\frac{\partial F}{\partial h'_y} = 0 \tag{C.23b}$$

where g is a function of x and y, h is a function of x and y, and g'_x is the derivative of g in respect to x.

Using this, the GVF energy in the x direction is minimized when:

$$\left|\mathbf{F}_0(x,y)\right|^2\Big(F_{\text{ext X}}(x,y) - F_{0X}(x,y)\Big) - \mu\frac{d^2F_{\text{ext X}}(x,y)}{dx^2} - \mu\frac{d^2F_{\text{ext X}}(x,y)}{dy^2} = 0 \tag{C.24}$$

To solve this, we use the gradient descent resulting in the following:

$$\begin{aligned} &F_{\text{ext X}}(x,y)_{n+1} - F_{\text{ext X}}(x,y)_n = \\ &\quad -1 * \left[\left| \mathbf{F}_0(x,y) \right|^2 \Big(F_{\text{ext X}}(x,y)_n - F_{0x}(x,y) \Big) - \mu \frac{d^2 F_{\text{ext X}}(x,y)_n}{dx^2} \right. \\ &\quad \left. - \mu \frac{d^2 F_{\text{ext X}}(x,y)_n}{dy^2} \right] \end{aligned} \tag{C.25}$$

If we replace $\frac{d^2 F_{\text{ext X}}(x,y)_n}{dx^2} + \frac{d^2 F_{\text{ext X}}(x,y)_n}{dy^2}$ with the discrete Laplacian:

$$\begin{aligned} \nabla^2 F_{\text{ext X}}(x,y)_n = F_{\text{ext X}}(x+1,y)_n + F_{\text{ext X}}(x,y+1)_n + F_{\text{ext X}}(x-1,y)_n \\ + F_{\text{ext X}}(x,y-1)_n - 4F_{\text{ext X}}(x,y)_n \end{aligned}$$

Equation C.25 simplifies to:

$$\begin{aligned} F_{\text{ext X}}(x,y)_{n+1} = \Big[1 - \left| \mathbf{F}_0(x,y) \right|^2 \Big] F_{\text{ext X}}(x,y)_n + \frac{\mu}{4} \nabla^2 F_{\text{ext X}}(x,y)_n \\ + \left| \mathbf{F}_0(x,y) \right|^2 F_{0\,\text{X}}(x,y) \end{aligned} \tag{C.26a}$$

Similarly for Y:

$$\begin{aligned} F_{\text{ext Y}}(x,y)_{n+1} = \Big[1 - \left| \mathbf{F}_0(x,y) \right|^2 \Big] F_{\text{ext Y}}(x,y)_n + \frac{\mu}{4} \nabla^2 F_{\text{ext Y}}(x,y)_n \\ + \left| \mathbf{F}_0(x,y) \right|^2 F_{0\,\text{Y}}(x,y) \end{aligned} \tag{C.26b}$$

BIBLIOGRAPHY

Acton, S. T. and Ray, N., *Biomedical Image Analysis: Tracking.* Morgan and Claypool, 2005.

Acton, S. T. and Ray, N., *Biomedical Image Analysis: Segmentation.* Morgan and Claypool, 2009.

Blake, A. and Isard, M., *Active Contours.* Springer-Verlag London Limited, 1998.

Canny, J., "A computational approach to edge detection," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 8, no. 6, pp. 679–698, November 1986.

Cohen, L. D., "On active contour models and balloons," *Computer Vision, Graphics, and Image Processing: Image Understanding*, vol. 53, no. 2, pp. 211–218, March 1991.

Cook, J. D. John D. Cook Consulting. "converting color to grayscale." August 2009. [Online]. Available: https://www.johndcook.com/blog/2009/08/24/algorithms-convert-color-grayscale/

Ivins, J. and Porrill, J., *Everything You Always Wanted to Know About Snakes (But Were Afraid to Ask)*, Artificial Intelligence Vision Research Unit, University of Sheffield, July 1993.

Kass, M., Witkin, A., and Terzopoulos, D., "Snakes: Active contour models," *International Journal of Computer Vision*, vol. 1, no. 4, pp. 321–331, 1987.

Li, B. and Acton, S. T., "Active contour external force using vector field convolution for image segmentation," *IEEE Transactions on Image Processing*, vol. 16, no. 8, August 2007.

Li, B., Millington, S. A., Anderson, D. D., and Acton, S. T., "Registration of surfaces to 3d images using rigid body surfaces," *Fortieth Asilomar Conference on Signal, Systems, and Computers*, pp. 416–420, 2006.

Marr, D. and Hildreth, E., "Theory of edge detection," *Proceedings of the Royal Society of London. Series B, Biological Sciences*, vol. 207, no. 1167, pp. 187–217, February 1980.

McInerney, T. and Teropoulos, D., "Deformable models in medial image analysis: A survey," *Medical Image Analysis*, vol. 1, no. 2, pp. 91–108, 1996.

Ray, N., Acton, S. T., Altes, T., de Lange, E. E., and Brookeman, J. R., "Merging parametric active contours within homogeneous image regions for mri-based lung segmentation," *IEEE Transactions on Medical Imaging*, vol. 22, no. 2, pp. 189–199, February 2003.

Tay, P. C., Li, B., Garson, C. D., Acton, S. T., and Hossack, J. A., "Left ventricle segmentation using model fitting and active surfaces," *Journal of Signal Processing Systems*, vol. 55, pp. 139–156, May 2008.

Xu, C. and Prince, J. L., "Generalized gradient vector flow external forces for active contours," *Signal Processing*, vol. 71, pp. 131–139, 1998.

Xu, C. and Prince, J. L., "Snakes, shapes, and gradient vector flow," *IEEE Transactions on Image Processing*, vol. 7, no. 3, pp. 359–369, March 1998.

Yuille, A. L., Hallinan, P. W., and Cohen, D. S., "Feature extraction from faces using deformable templates," *International Journal of Computer Vision*, vol. 8, no. 2, pp. 99–111, 1992.