Distribution Agreement

In presenting this thesis as a partial fulfillment of the requirements for a degree from Emory University, I hereby grant to Emory University and its agents the non-exclusive license to archive, make accessible, and display my thesis in whole or in part in all forms of media, now or hereafter known, including display on the world wide web. I understand that I may select some access restrictions as part of the online submission of this thesis. I retain all ownership rights to the copyright of the thesis. I also retain the right to use in future works (such as articles or books) all or part of this thesis.

Signature:

____________________	4/18/2014
Shannon P. Buckley	Date

Image Recognition of Coronary Stents for Automatic Construction of Patient Specific Models

by

Shannon Buckley

Adviser

Alessandro Veneziani

Department of Mathematics and Computer Science

Dr. Alessandro Veneziani
Adviser

Dr. David Zureick-Brown
Committee Member

Dr. Jed Brody
Committee Member

Dr. Shun Yan Cheung
Committee Member

4/18/14

Image Recognition of Coronary Stents for Automatic Construction of Patient Specific Models

by

Shannon Buckley

Adviser
Alessandro Veneziani

An abstract of
A thesis submitted to the Faculty of Emory College of Arts and Sciences
of Emory University in partial fulfillment
of the requirements of the degree of
Bachelor of Sciences with Honors

Department of Mathematics and Computer Science

2014

Abstract

Image Recognition of Coronary Stents for Automatic Construction of Patient Specific Models

By Shannon Buckley

Professor Veneziani and his team of graduate students have been working on the problem of modeling the effects that cardiovascular disease and the medical procedures employed to cure it have on the fluid dynamic process of the cardiovascular system. One of the newer solutions to this pressing disease is to insert a medical structure, called a stent, into the artery where a blockage is occurring. The modeling of this procedure requires the creation of a 3D model of the stent, which is then used in the algorithms. For real patients this data has previously been collected by manually recording the locations of the stent structures found in intravascular images of the patient's arteries. To speed up this data collection process and provide more accurate data, we have created a MATLAB algorithm that uses image recognition software to automatically identify stent structures in the intravascular images and record their positions in the artery.

Image Recognition of Coronary Stents for Automatic Construction of Patient Specific Models

by

Shannon Buckley

Adviser
Alessandro Veneziani

A thesis submitted to the Faculty of Emory College of Arts and Sciences
of Emory University in partial fulfillment
of the requirements of the degree of
Bachelor of Sciences with Honors

Department of Mathematics and Computer Science

2014

Contents

1 Introduction 1

1.1 Preface 1

2 Related Work 5

2.1 Atherosclerosis and Its Consequence 5

2.2 Stents 9

3 Image Recognition of Stent Structures 14

3.1 Introduction 14

3.2 OCT Imaging 15

3.3 Stent Recognition Algorithm 17

3.3.1 Step 1 18

3.3.2 Step 2 20

3.3.3 Step 3 22

3.3.4 Step 4 25

3.3.5 Step 5 27

3.3.6 Step 6 29

3.3.7 Step 7 29

3.3.8 Step 8 31

3.4 Using the Algorithm 33

4 Looking Forward **34**

4.1 Code 34

4.2 Application 38

4.3 Conclusion 42

5 Acknowledgements **44**

List of Figures

1.1 Different stages of the study of vascular problems through CFD (from [2]). 4

2.1 Sections of a healthy artery (left) and a strongly diseased artery (right) (courtesy of M. Tuveri). 6

2.2 Cholesterol buildup in the intima (courtesy of BruceBlaus via Wikipedia). 8

2.3 Collapsed (right) and Expanded (left) Stents (Courtesy of Frank C. Muller via Wikipedia). 10

2.4 A set of CT images of the human brain (courtesy of Wikipedia). 13

3.1 An OCT image of a patients artery with a stent. Notice the virtual representation of the stent in the top left part of the image. (courtesy of Dr. Guagliumi). 16

3.2 Original OCT image. Notice the struts of the stent. 19

3.3 Grayscale image. 21

3.4 Grayscale image with gamma adjusted contrast 23

3.5 Binary Image 24

3.6 Inverted binary image 26

3.7 Noise reduced inverted binary image. 27

3.8 Original OCT image with the centroids that were found. . . . 32

4.1 Figures courtesy of Boyi Yang. 39

4.2 Figures courtesy of Boyi Yang. 40

4.3 Preliminary results showing the pressure and velocity magniture at different areas of the blood vessel. Notice the visibility of the stent geometry in the models. (Figures courtesy of Boyi Yang). 41

4.4 Preliminary results showing the wall shear stress and and velocity magniture at different areas of the blood vessel. Notice the visibility of the stent geometry in the models. (Figures courtesy of Boyi Yang). 41

4.5 The automated results capture about 33% of the stent structures present as shown in this comparison between the algorithm's results on the left and the manually recorded results on the right. 43

Chapter 1

Introduction

1.1 Preface

In humans, the cardiovascular system is the mechanism through which the body takes in oxygen from the air and delivers it to the organs via blood and a network of arteries and veins. The heart pumps the oxygen-rich blood to the organs through arteries and the oxygen-depleted blood returns through veins. The oxygen-depleted blood is then pumped to the lungs where it is filled with oxygen again and returned to the heart. The cycle repeats like this indefinitely until we die. This system is a vital component of a functioning human being, and thus if it develops issues that prevent it from working correctly, they must be fixed or we risk the possibility of death. Under normal circumstances arteries are able to adapt in order to meet the demands of changing blood flow and pressure; however, sometimes a significant blockage can develop which causes a depletion of the necessary oxygen-rich blood from an organ downstream. Severe cases can lead to such devastating consequences as a stroke or necrosis of the organs. As a result, much study has been focused on the workings of the cardiovascular system in order to better understand its functions and how it reacts to vascular diseases. One of the main approaches has been the study of hemodynamics, the study of fluid dynamics in blood. The complexity of the vascular system makes an analytical approach to hemodynamics tricky and very involved. But using Computational Fluid Dynamics (CFD), a subset of fluid mechanics using algorithms and numerical methods to solve problems involving fluid flow, has allowed for a study of vascular diseases in real clinical applications. Its versatility and ability to work with real vascular geometries make it an integral part of this field. Beyond analytical and numerical approaches, there is the ”experimental” one (in vitro) and the use of animals (in vivo). CFD complements these approaches. As shown in Fig. 1.1, doctors, engineers, mathematicians and physiologists can all work together using CFD to better understand physiological and pathological phenomena along with surgical planning.

The set up of the numerical model, which is at the heart of the interaction, requires two main inputs: the mathematical description of the problem, and the specific individual data. The latter of which will be the main topic of this thesis.

We will discuss the common but dangerous pathology, Atherosclerosis, and one of the medical procedures used to remedy it: the placement of a stent. Then we will discuss how CFD can model this situation in real clinical applications before we present the results of our research.

All of the work presented in this thesis puts emphasis on the large and medium vessels because the development and effects of diseases are more relevant there. [1]

[1] Basic reference for this short introduction is [1].

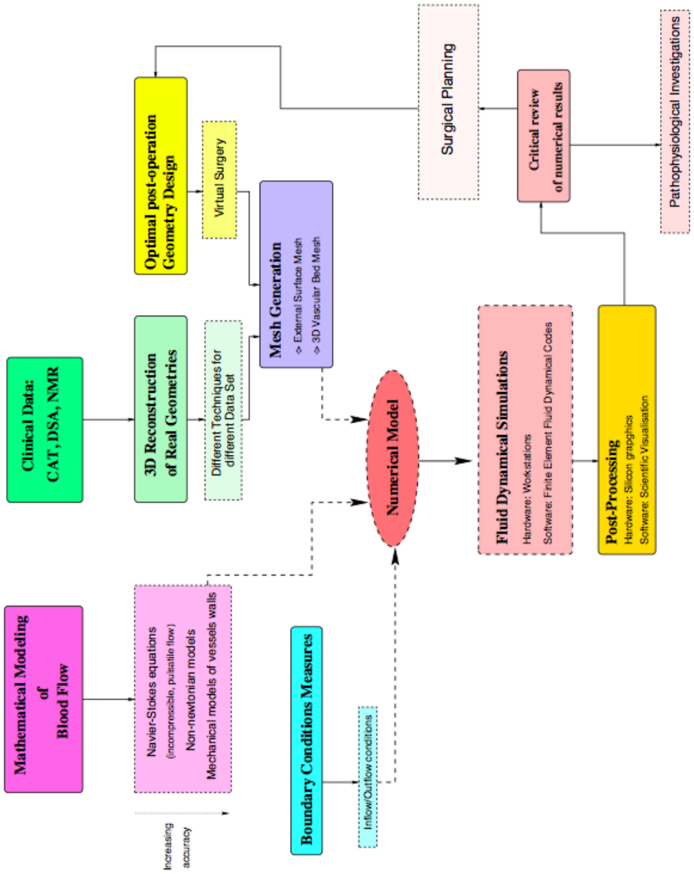

Mathematical Modeling of Blood Flow
Navier-Stokes equations (incompressible, pulsatile flow)
Non-newtonian models
Mechanical models of vessels walls
Increasing accuracy
Boundary Conditions Measures
Inflow/Outflow conditions
Clinical Data: CAT, DSA, NMR
3D Reconstruction of Real Geometries
Different Techniques for different Data Set
Optimal post-operation Geometry Design
Virtual Surgery
Mesh Generation
-> External Surface Mesh
-> 3D Vascular Bed Mesh
Numerical Model
Fluid Dynamical Simulations
Hardware: Workstations
Software: Finite Element Fluid Dynamical Codes
Post-Processing
Hardware: Silicon grapghics
Software: Scientific Visualisation
Critical review of numerical results
Surgical Planning
Pathophysiological Investigations

Figure 1.1: Different stages of the study of vascular problems through CFD (from [2]).

Chapter 2

Related Work

2.1 Atherosclerosis and Its Consequence

Atherosclerosis, which is commonly known to be a very widespread disease in the developed world, is the result of plaque forming underneath the intima, or innermost layer of an artery or vein. The plaque buildup causes a thickening of the walls which can increase blood pressure by significantly reducing the blood's ability to flow through that portion of the vessel. The buildup also reduces the blood vessel's elasticity which further decreases the amount of blood allowed through the vessel. If the buildup becomes severe enough, a total blockage of the vessel can occur leaving the downstream organs without

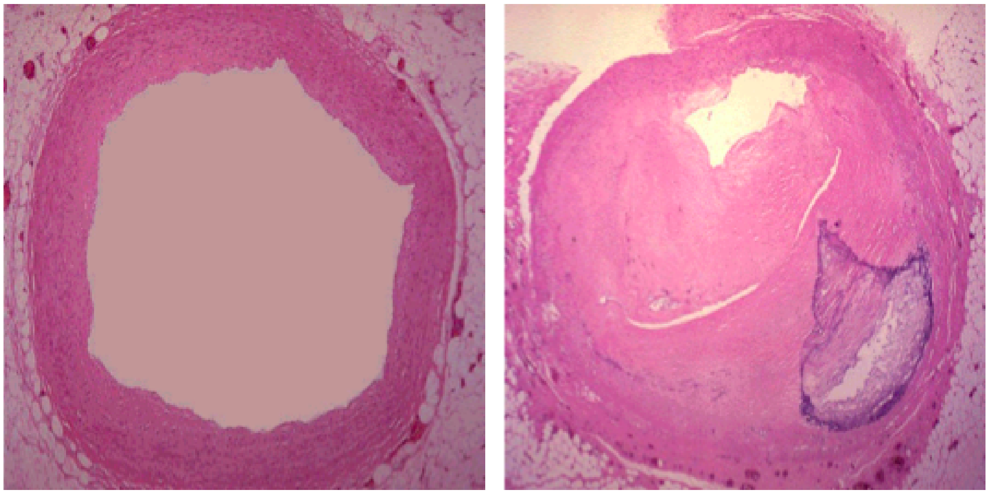


Figure 2.1: Sections of a healthy artery (left) and a strongly diseased artery (right) (courtesy of M. Tuveri).

their oxygen supply. This is what happens when a person suffers from a stroke or a heart attack.

In more detail, Low-Density Lipoprotein (LDL), a lipoprotein in the blood carrying cholesterol, can move through the blood vessel wall and become stuck under the intima. Once there, the LDL particles can become oxidized by free radicals, which are atoms or molecules with an unpaired covalent bond. Oxidized LDL causes an inflammatory reaction from the cells in the vessel wall which in turn triggers an immune response from the body. This response is to send specialized white blood cells, called macrophages, to digest and dispose of the LDL particles. The macrophages, however, are unable to process the oxidized LDL and in time rupture. This release a large amount of oxidized cholesterol in the intima. With more oxidized LDL in the vessel wall, the body sends more macrophages, continuing the cycle. This process repeats itself until the blood vessel is inflamed enough to make the surrounding muscular layer of the vessel's wall form an aneurysm around the plaque in order to contain it. An aneurysm is a bulge in the wall of a blood vessel that is usually filled with blood, but can also be filled with other substances such as cholesterol plaque.

While the strength of the muscular layer itself is not compromised, the aneurysms are weak and easily ruptured due to a low level of elasticity. When they rupture, the lipids contained within them spill out into the blood vessel and cause stenosis, i.e. narrowing of the lumen. Another result of the plaque's presence is that a calcification forms between the lipids and the vascular muscle cells. As the calcium deposits accumulate, the lumen narrows and restricts blood flow. This is just a basic description of the disease so that the reader may gain enough of an understanding to follow the procedures and results presented later in this thesis. [1]

[1] Basic reference for this short background section is [1].

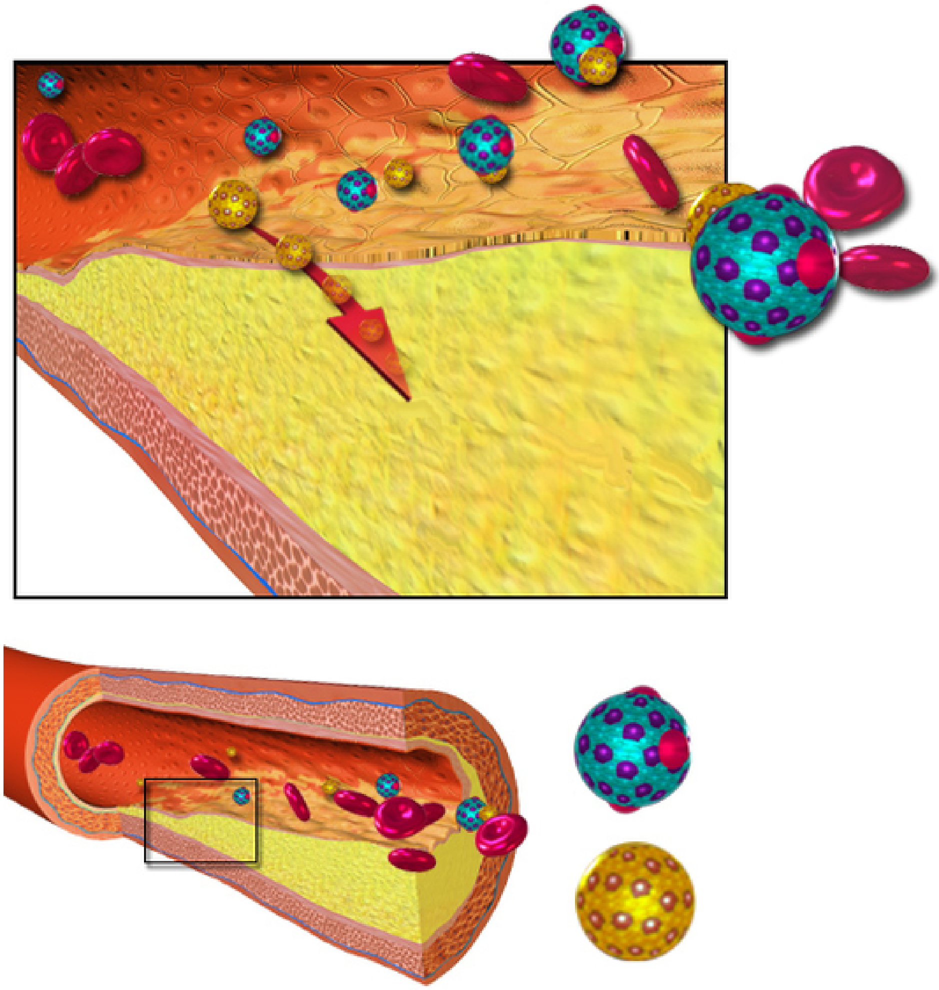


Figure 2.2: Cholesterol buildup in the intima (courtesy of BruceBlaus via Wikipedia).

2.2 Stents

As we have seen, Atherosclerosis is a potentially deadly disease that builds up over time inside the blood vessels. Once a blockage of the lumen becomes severe enough to cause dangerous health issues, treatment is required.

There are multiple surgical treatments available:

1. *Angioplasity*: This is the mechanical enlargement of the blood vessel to increase lumen size. A deflated balloon on a catheter is inserted into the vessel at the site of stenosis and inflated to expand the passageway. One of the dangers of this procedure is that fragmented plaque can float away and block the vessel at a point downstream.

2. *Stent Placement*: A stent is a structure made from a cylindrical steel net that is inserted into the vessel via catheter. Once at the site of stenosis, the stent is expanded by a ballon and used to keep the lumen dilated so blood can flow more easily. See Fig. 2.3.

3. *Endoarterectomy*: This is an extremely invasive procedure by which the plaque is surgically separated from the lining of the artery's wall. A side effect of this treatment is damage to the arterial wall.

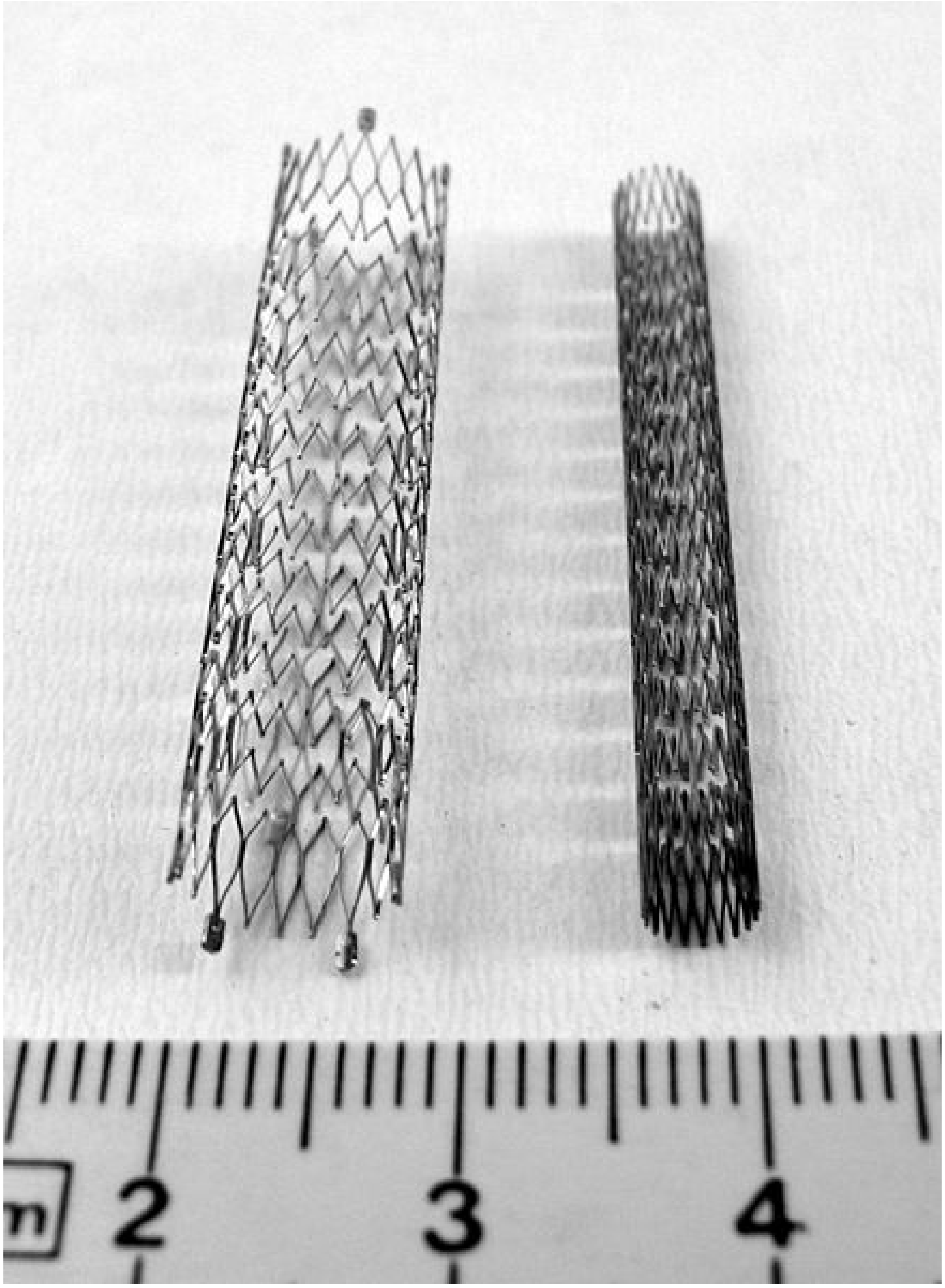


Figure 2.3: Collapsed (right) and Expanded (left) Stents (Courtesy of Frank C. Muller via Wikipedia).

4. *By-pass*: A by-pass is also an extremely invasive surgery that creates an alternate path for the blood around the stenosis.

The procedure that this research focuses on is stent placement. In hemodynamics with CFD, we wish to model the blood flow of human beings so that we can analyze patterns and better understand the complexity of the cardiovascular system. Part of this analysis includes modeling the effects and repercussions that the placement of a stent has on the blood flow of a patient. We wish to know many things such as: the resulting mechanical actions of the blood caused by a foreign object in the blood vessel, the changes in pressure felt by the vessel's walls, the effects on circulation downstream, and how long the stent will treat the stenosis. A comprehensive understanding of these factors coordinated with efforts by doctors and physicians can help make this surgery not only more effective, but also safer for the patients that must undergo it.

Originally, generic numerical models were created using in vitro experiments to collect data for the algorithms. While they helped foster a much deeper understanding of hemodynamics, every patient is unique and thus a more specific model was needed in order to gain the most benefit for clinical purposes. Modern technology has made this possible. Imaging techniques for reconstructing individual geometrical morphologies include Digital Subtraction Angiographies (DSA) and Computed Tomography scans (CT scans). DSA images are acquired by exposing specific areas of the body to x-rays while injecting the blood vessels with a contrast medium to gain anatomical positioning of the blood vessels. CT scans use computer processed x-rays to create virtual slices of areas of interest in the body which allow us to see inside without cutting the person open. See Fig. 2.4 for an example of a set of virtual slices of the human brain. When stacked, these slices can combine to make a 3D model of the geometry we want for our computations.

Now that the reader possesses a working knowledge of the issue at hand and the method by which we attempt to solve it, we can move on to the main research of this paper.[2]

[2] Basic reference for this short introduction is [1].

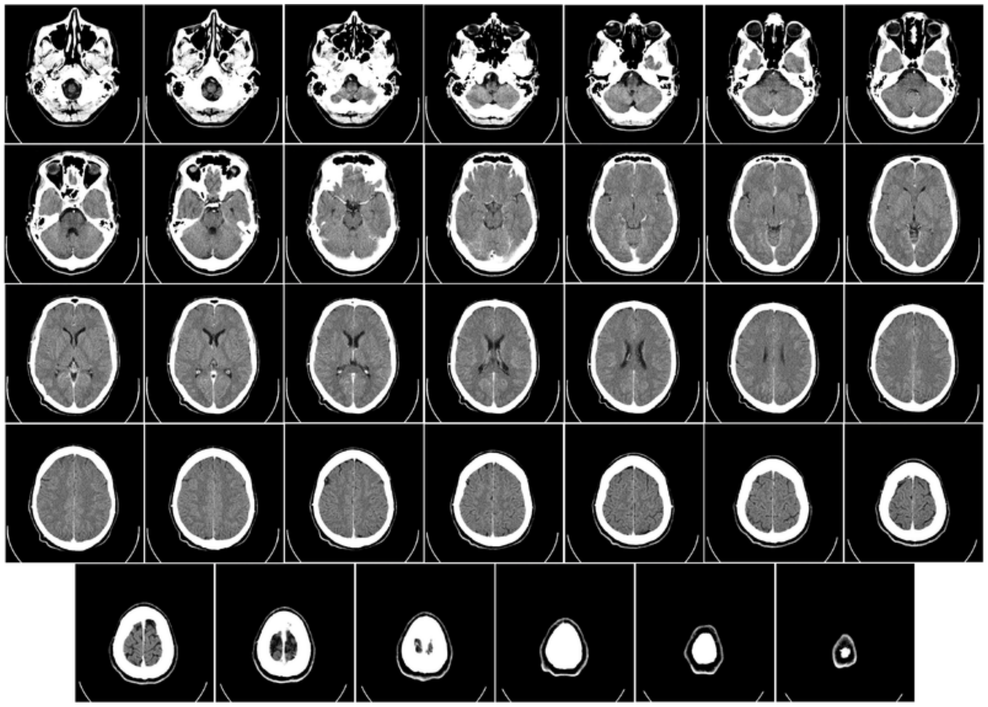


Figure 2.4: A set of CT images of the human brain (courtesy of Wikipedia).

Chapter 3

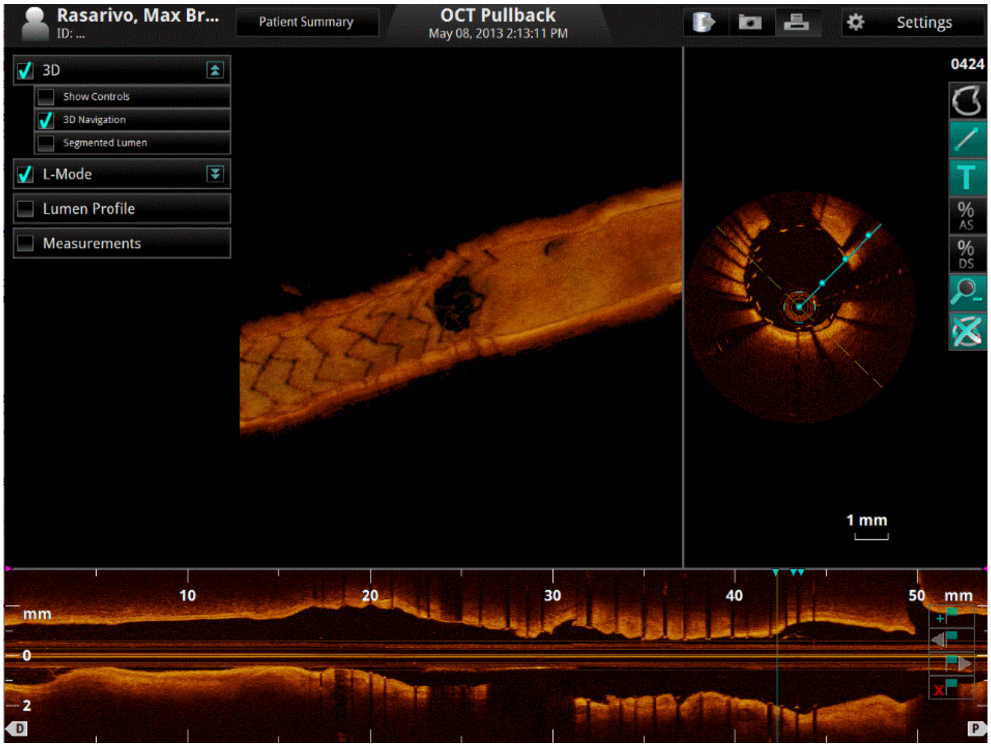
Image Recognition of Stent Structures

3.1 Introduction

In order to model the placement of a stent in a real patient from medical images, our algorithms require the creation of a 3D geometry of the stent. For real patients, this data has previously been collected by manually recording the locations of the stent structures found in intravascular images of the patient's arteries. To speed up this data collection process and to provide more accurate data, we have created a MATLAB® algorithm that uses image recognition software to automatically identify stent structures in the arthroscopic images. The algorithm then records their positions in the artery to be compiled into a 3D geometry.

3.2 OCT Imaging

Optical Coherence Tomography (OCT) imaging is the intravascular imaging process used to gain cross-sectional images of the patient's blood vessels. It is a minimally invasive procedure by which a probe is inserted into the blood vessel via a small incision in the arterial wall. In short, the probe scatters near-infrared light rays throughout the blood vessel and using an echoing technique similar to ultrasound imaging, collects the reflected rays into a high-resolution cross-sectional image of the artery. Just as in CT scans, these two dimensional images can be compiled into a 3D stack which is then used for image processing. Fig. 3.1 is an example of an OCT image with a virtual representation of the stent inside the blood vessel.



☑ 3D
□ Show Controls
☑ 3D Navigation
□ Segmented Lumen
☑ L-Mode
□ Lumen Profile
□ Measurements

Figure 3.1: An OCT image of a patients artery with a stent. Notice the virtual representation of the stent in the top left part of the image. (courtesy of Dr. Guagliumi).

3.3 Stent Recognition Algorithm

Our stent recognition algorithm is a program that takes in a cross-sectional OCT image of a patient's artery as an input and returns the locations of stent structures that it identifies in the image. It is written in a special computer language for a software called MATLAB® . MATLAB® is an invaluable computer software for computational mathematics created by the company Mathworks®. The software uses its own computer language to enable the coding of functions, algorithms and programs for a wide range of numerical purposes. In our case, we used MATLAB®'s language to write our image processing algorithm which we can then use MATLAB® to run.

Our code is modified from a publicly shared shape recognition program [3]. These are the steps it goes through in order to identify the stent structure locations in the artery:

1. Read image into program

2. Convert image from rgb to gray

3. Find grayscale threshold and convert image to binary image

4. Invert the binary image

5. Find boundaries

6. Determine object properties

7. Plot stent boundaries with centroids on original image

8. Return array of centroid values

3.3.1 Step 1

The first step of our algorithm is to read in the OCT image. We do this with a simple sequence of code:

```
function W = Classify(ImageFile)
RGB = imread(ImageFile);
```

Where ImageFile is the OCT image you wish to use.

The first line brings the image in as a parameter and then the function imread() [6] saves the image as an rgb image variable named RGB, where rgb stands for red, green blue. Each pixel of this image contains three sets of information: hue, saturation and intensity values. Hue values are the values by which the color of an object is classified in reference to the color spectrum.

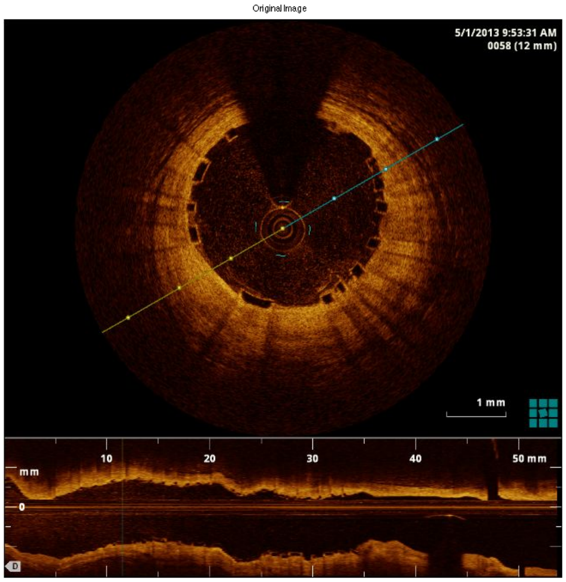


Figure 3.2: Original OCT image. Notice the struts of the stent.

Saturation values are the colorfulness of a color. A high saturation value corresponds to brighter colors, whereas a low value means the color appears dark. Intensity values are given in a range from 0 to 1, where 0 represents a complete absence of light (black) and 1 represents a complete presence of light (white). Step 2 will then use the image variable RGB as its parameter variable.

3.3.2 Step 2

The next sequence of code converts the rgb image into a grayscale image and adjusts the grayscale contrast to bring out more detail of the stent structures for recognition:

```
GRAY = rgb2gray(RGB);
J = imadjust(GRAY,[0 0.5],[0.5 1],.75);
```

Here, the function rgb2gray() [7] is a MATLAB® function that takes an rgb image, such as our RGB, as its parameter and eliminates the hue and saturation information leaving behind a grayscale image that contains only the intensity information. It then returns the image as the image variable

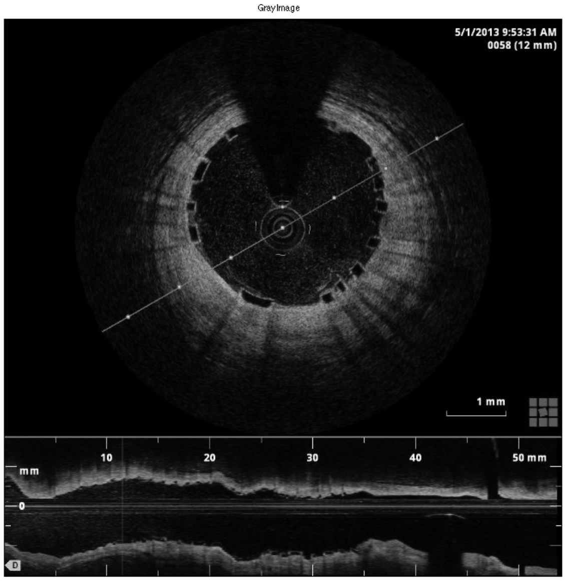


Figure 3.3: Grayscale image.

GRAY. The resulting image is what one normally thinks of as a black and white image.

All of the pixels are displayed solely with different shades of gray containing a value from 0 to 1.

The MATLAB® function imadjust(GRAY,[0 0.5],[0.5 1],.75) [8] is an intensity value adjusting function used to bring out more detail from an image. It has four input parameters and returns the adjusted image as an image variable. The first parameter is your image variable that you wish to adjust. In our code, the greyscale image GRAY is used. The next two parameters are the range of intensity values from the input image you wish to adjust and the the range of intensity values you wish to map them to for the output image. For example, in our code we wish to map the values in the range 0 to 0.5 to values in the range 0.5 to 1. These ranges were chosen through experimentation in order to adjust the image to a brighter contrast and bring out the finer details that were previously lost due to lower intensity values. The final parameter is a gamma value that specifies the shape of the curve used to map the input intensity values to the output intensity values. If gamma is equal to 1, the curve is linear. If gamma is less than 1, the curve is weighted towards higher intensity values. If gamma is greater than 1, the curve is weighted towards lower intensity values. In our code, we found through experimentation, that a gamma value of .75 brings out the best mapping for our image recognition purposes. The new image variable J is our grayscale adjusted image that Step 3 uses for the conversion to a binary image.

3.3.3 Step 3

In this step, the program creates an intensity value threshold with which it can then convert the grayscale image into a binary image:

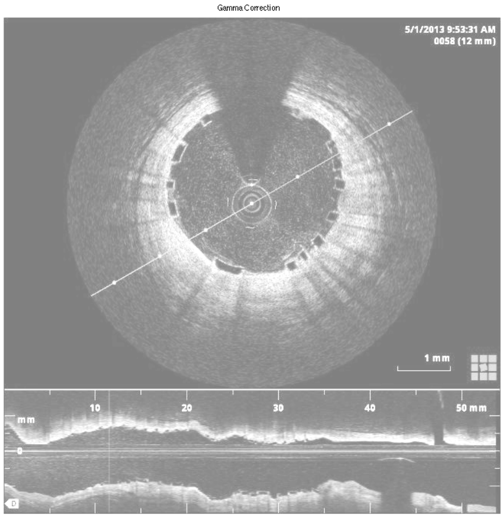


Figure 3.4: Grayscale image with gamma adjusted contrast

```
threshold = graythresh(J);

BW = im2bw(J, threshold);
```

graythresh() [9] is a MATLAB® function that uses Otsu's method [4] to find a conversion threshold of a grayscale image. In our code, the graythresh() function reads in the gamma corrected grayscale image J and then returns its threshold value as the variable threshold. The next function, im2bw() [10] reads in an input image and a threshold value which it applies to the input

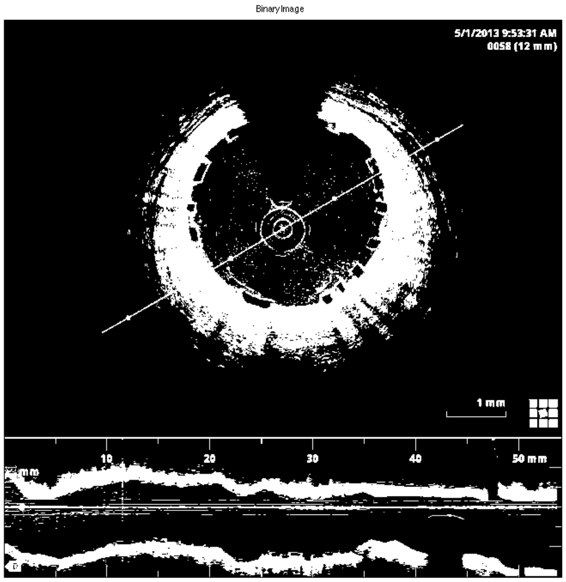


Figure 3.5: Binary Image

image, converting every pixel with an intensity value below the threshold to black, and every pixel with an intensity value above the threshold to white. It then returns the input image as a binary image variable. A binary image is one in which the the pixels contain either the value 1 for white or the value 0 for black. Here the input image is again the gamma corrected grayscale image J, and the threshold is the one created using the function graythresh(). The resulting binary image is returned as the image variable BW.

3.3.4 Step 4

Step 4 is a relatively simple step but very important for the purposes of our program. Here the binary image is inverted and then background noise is eliminated:

```
BW = ~ BW;
bw = areaopen(BW, 50);
```

The $\sim$ symbol inverts all of the pixels' intensity values turning 1's into 0's and 0's into 1's. This step is necessary for the shape recognition function to find the boundaries of the stents. The function finds contiguous pixels with intensity values of 0 and uses them to trace the boundary of an object. Originally, the stent outlines had intensity values of 1 so for the algorithm to recognize them, we need to invert the image and give the values of 0.

The areaopen() [11] function is used to reduce background noise in our image. It takes in a binary image and a threshold number. It uses this threshold to convert any grouping of contiguous white pixels, whose total number is less than the threshold, to black. Our code uses the inverted binary image BW, and a threshold of 50 pixels. Thus any group of white

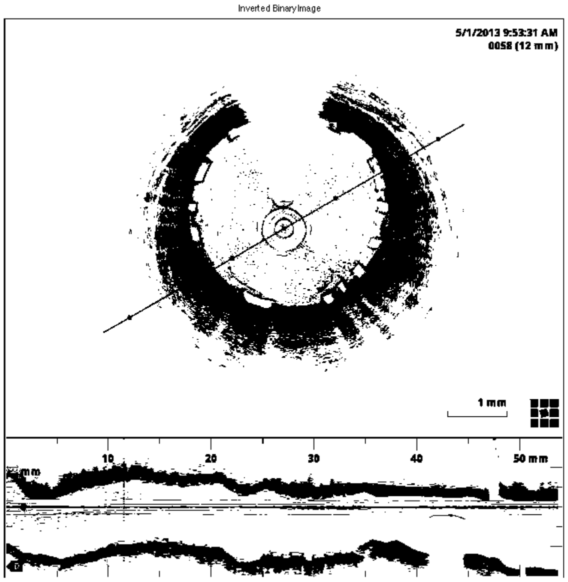


Figure 3.6: Inverted binary image

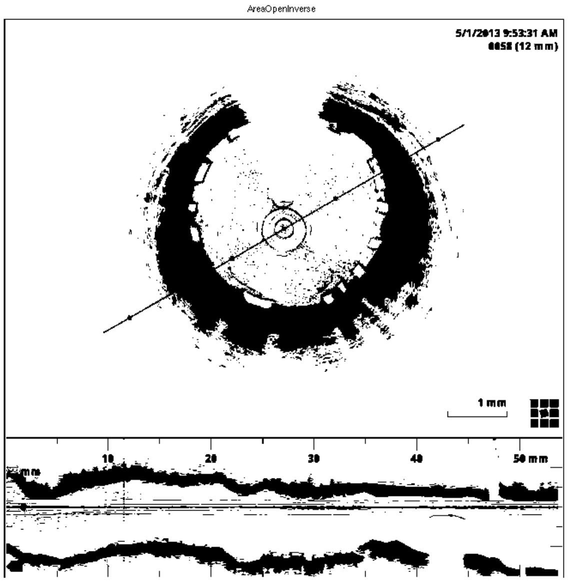


Figure 3.7: Noise reduced inverted binary image.

pixels composed of fewer than 50 pixels will be converted to black. We know that a stent structure's interior will be composed of more than 50 pixels and so any object composed of fewer, is unnecessary for our computations and is just background noise.

3.3.5 Step 5

At this point we now have the image in the format that we want and can apply the MATLAB® boundary finding function:

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

The bwboundaries() [12] function uses a binary image as its input parameter. The image must have the pixels of objects contain the value 1 and background pixels contain the value 0. In our code, this is the inverted binary image bw. The function then returns two matrices containing the boundary information. The first matrix is a cell array with Z cells, where Z is the number of objects found in the image. Each cell contains a Y-by-2 matrix where each row contains the row and column coordinates of a boundary pixel. Y is the number of boundary pixels found for an object. In our code, we designate this matrix as B. The second matrix is a label matrix, which we designate as L. This is a two-dimensional array of nonnegative integers that represents contiguous regions in the image. Region k is the set of all elements of L that have the value k (Reference MATLAB$^{®}$ Page). This array is then used in the next step to compute the properties of the objects that were found. The 'no holes' parameter of this function is used to specify that we wish for the algorithm to not search for objects within other objects since there will be no stents inside other stents in our image.

MATLAB® does not publish the exact code it uses for the bwboundaries() function. But for a more in-depth look at boundary tracing in image processing see [5].

3.3.6 Step 6

This is another one line sequence of code that is essential in our program:

```
STATS = regionprops(L, 'all');
```

The regionprops() [13] function takes a label matrix, L, and computes a set of properties about each labeled object. It returns these properties in a P-by-1 array, STATS, where P is the number of objects labeled in L. The property we are concerned with is the centroid of the object, which we will use as our location of the stent structure for our creation of the 3D geometry.

3.3.7 Step 7

At this point we could just return the centroid pixel locations that we have found, but for the purpose of understanding how successful our program was, we use this step to display the boundaries of the objects we found, and their centroids on the original image. There is only one part of this step that requires explaining:

```
for i : 2 : length(STATS)
    if STATS(i).Extent > .35 &&
    STATS(i).MinorAxisLength/STATS(i).MajorAxisLength > .3
        centroid = STATS(i).Centroid;
        plot(centroid(1),centroid(2), 'w0');
    end
end
```

Here we use a for loop to move from one entry of the P-by-1 array, STATS, to the next. At each entry we use an if statement to check different properties of the objects we found in an attempt to further reduce noise. If the object meets the property requirements, we print it on our original image. The first property we check is the extent of an object. The extent is a scalar that represents the ratio of pixels in the interior of the object, to pixels in the total bounding box of the object. The total bounding box is the smallest rectangle that contains the object. We know that the stent structures we are looking for should be roughly rectangular, and thus have a relatively high extent ratio. Requiring an object to have an extent of at least .35 in order to be recorded, lowers the probability of finding objects that aren't stents. The second property we check is the ratio of the minor and major axis lengths of the smallest ellipse that can be drawn around the object. Again, we know that the stent structures should be roughly rectangular and thus have a relatively high ratio. Thus, requiring an object to have a ratio of at least .3 in order to be recorded, further lowers the probability of finding objects that aren't stents. The values of .35 and .3 were found through extensive experimentation to produce the greatest number of stents recognized while at the same time having the greatest effect on noise reduction.

No other part of this step requires explaining, so we will leave this part of the code for viewing at the end of this paper in a complete presentation of our algorithm.

3.3.8 Step 8

Finally we return the centroid pixels in an array to the user so that they may be used in the 3D construction of our geometry.

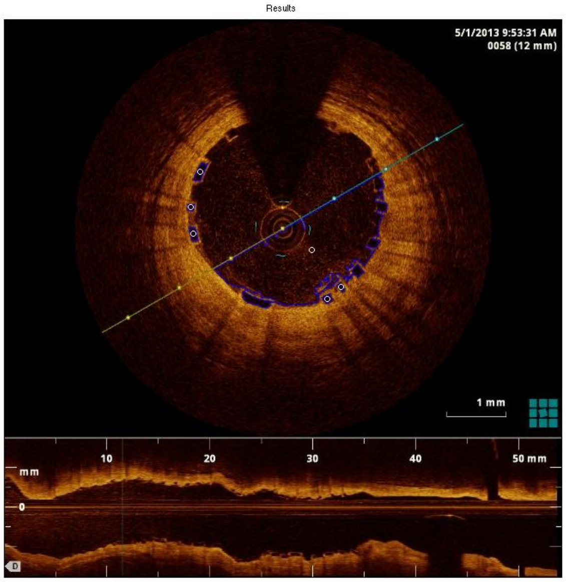


Figure 3.8: Original OCT image with the centroids that were found.

3.4 Using the Algorithm

Using the algorithm is a fairly straightforward process. You simply load the the OCT images you want to process into your MATLAB working directory. Then enter the command:

```
W = Classify(’imagename’)
```

The function will then process the image and return an n-by-2 matrix W containing the centroids of all the n stents that were found. The first column will contain the column location of the centroid and the second column will contain the row location of the centroid. The function will also display a results image showing all the stents that were identified with their centroids plotted. Run this command for every image you wish to process.

Chapter 4

Looking Forward

4.1 Code

The following is the completed code along with comments marked by the % symbol:

```
function W = Classify(ImageFile)
% Step 1: Read image in
RGB = imread(ImageFile);

% Step 2: Convert image from rgb to gray
```

```
GRAY = rgb2gray(RGB);


% Use image adjust with Gamma Correction to change greyscale contrast
% to pick up more detail of stent structures.
J = imadjust(GRAY,[0 0.5],[0.5 1],.75);


% Step 3: Threshold the image and convert the image to black and white
%in order to prepare for boundary tracing using bwboundaries().
threshold = graythresh(J);
BW = im2bw(J, threshold);


% Step 4: Invert the Binary Image
BW =∼ BW;


%bwareaopen gets rid of white space so its betrer to use on the inverse
%image to erase the noise.
bw = bwareaopen(BW, 55);


% Step 5: Find the boundaries. Concentrate only on the exterior boundaries.
% Option ’noholes’ will accelerate the processing by preventing
% bwboundaries from searching for inner contours.
[B, L] = bwboundaries(bw, ’noholes’);

%The original B captures the outline of the entire image as its first
%object so we want to remove that.
B = B(2:length(B));

% Step 6: Determine object’s properties.
STATS = regionprops(L, ’all’); % we need ’BoundingBox’ and ’Extent’

%Display final image.
figure,
imshow(RGB),
title(’Results’);
hold on

%Shows all the boundaries documented in the Cell Array B from
%bwboundaries().
```

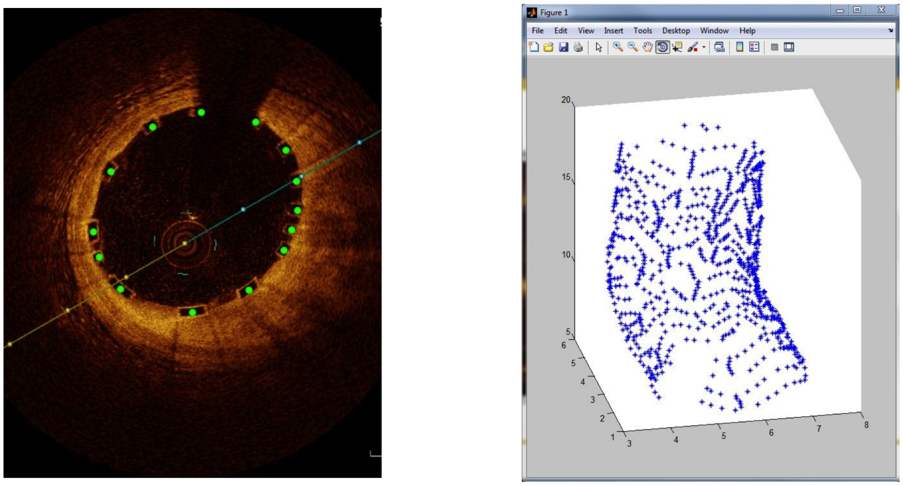
```
for i = 1 : length(B)
    G = B{i};
    if length(G) ≤ 150 &&
    length(G) > 15 %Don’t plot large or small areas
        X = G(:, 2);
        Y = G(:, 1);
        plot(X, Y);
    end
end


%Used to plot centroids of shapes that have been recognized.
for i = 2 : length(STATS)
    if STATS(i).Extent > .35 &&
    STATS(i).MinorAxisLength/STATS(i).MajorAxisLength > .3
        centroid = STATS(i).Centroid;
        plot(centroid(1),centroid(2),’wO’);
    end
end
```

```
%Return matrix W containg centroid loctions.
W = zeros((length(STATS)-1),2);
for i = 1 : length(W)
    if STATS(i+1).Extent > .35 &&
    STATS(i+1).MinorAxisLength/STATS(i+1).MajorAxisLength > .3
        W(i,1) = STATS(i+1).Centroid(1);
        W(i,2) = STATS(i+1).Centroid(2);
    end
end
return[1]
```

4.2 Application

One of Professor Veneziani's graduate students, Boyi Yang, is currently working on the CFD problem for the placement of stents in patients. His work, relavent to this thesis, consists of compiling patient-specific 3D models of stents which he then uses in the computation of factors such as: the resulting mechanical actions of the blood, the changes in pressure felt by the

[1] Code modified from publicly shared shape recognition program [3].



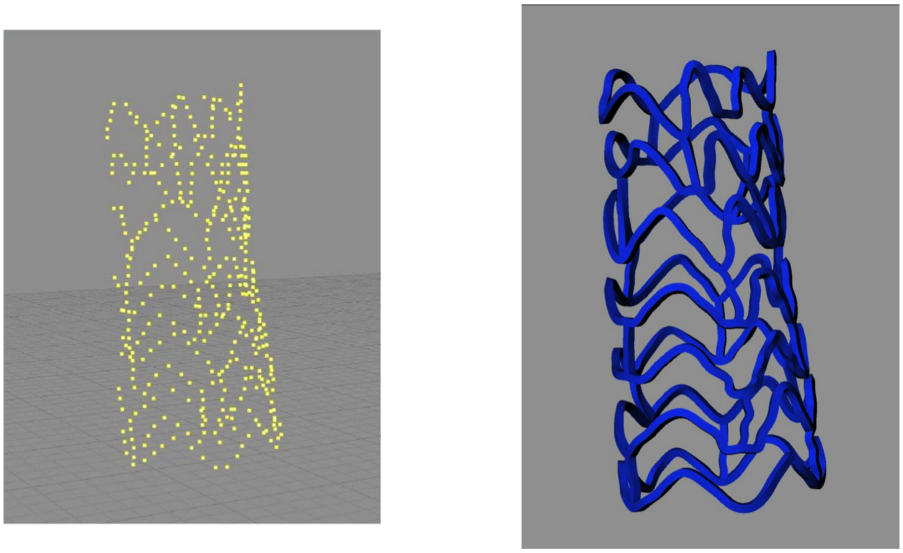
(a) Collection of Data.

(b) Data Plotted.

Figure 4.1: Figures courtesy of Boyi Yang.

vessel's walls, the effects on circulation downstream, and for how long the stent will treat the stenosis. Just like the code presented in this thesis, Boyi uses OCT images to collect data describing the locations of the stent structures. However, Boyi currently collects this information by hand, which is a very tedious job and the reason for our data collection algorithm. If this process can be automated, much more of Boyi's time can be saved for the important job of analyzing the CFD results. After the data collection, he then compiles this information into the 3D plot which he will use for his model. Fig. 4.1 illustates these two stages of the process.

Once he has his data plot, he constructs the 3D patient-specific stent

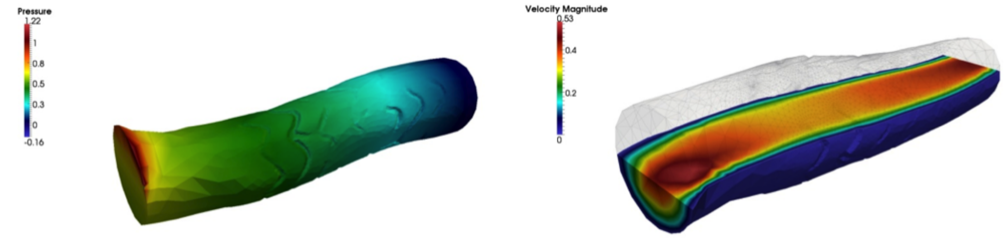


(a) Recorded Data.

(b) Stent Geometry.

Figure 4.2: Figures courtesy of Boyi Yang.

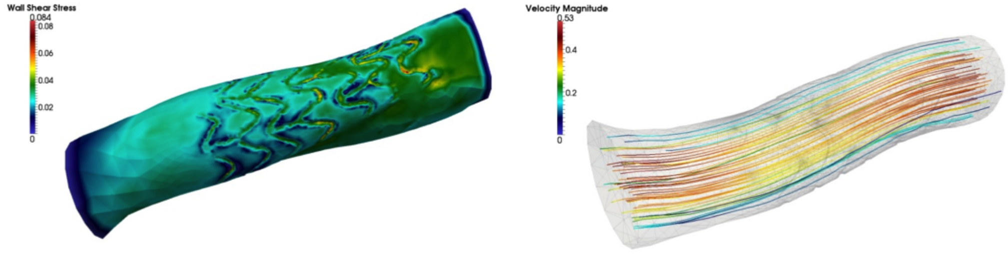
geometry and applies the CFD algorithms. Fig. 4.2 shows the construction of the geometry and Figs. 4.3 and 4.4 show some preliminary results using the geometry. As you can see in Figs. 4.3 and 4.4, the stent geometry is visible in the walls of the artery. These results are the ones used to analyze the cardiovascular disease Atherosclerosis. It is through this process that insights into the solutions and disease itself can be gained.



(a) Pressure.

(b) Velocity Magnitude.

Figure 4.3: Preliminary results showing the pressure and velocity magnitude at different areas of the blood vessel. Notice the visibility of the stent geometry in the models. (Figures courtesy of Boyi Yang).



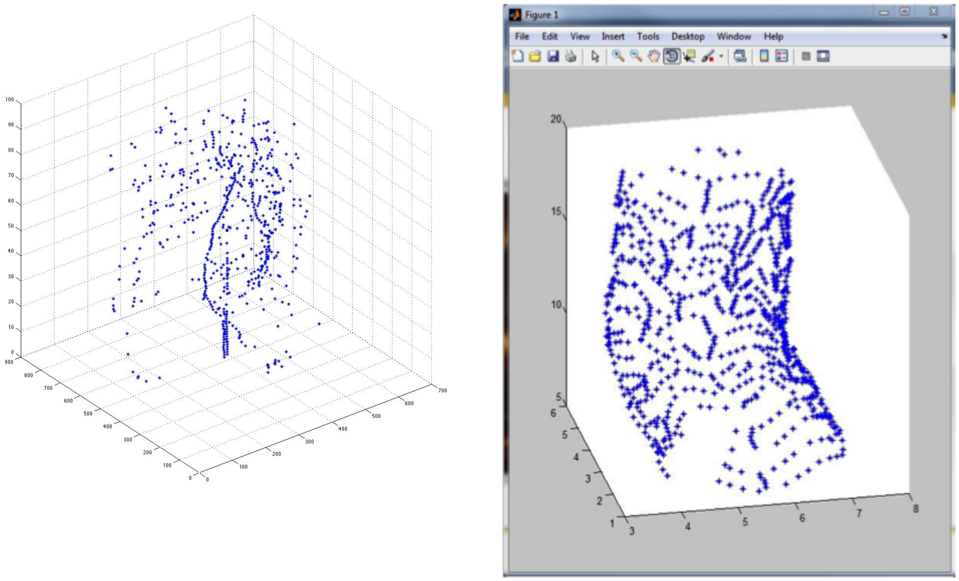
(a) Wall Shear Stress.

(b) Velocity Magnitude.

Figure 4.4: Preliminary results showing the wall shear stress and and velocity magniture at different areas of the blood vessel. Notice the visibility of the stent geometry in the models. (Figures courtesy of Boyi Yang).

4.3 Conclusion

As we have seen, the human cardiovascular system is an extremely important yet very complicated sytem. It is required for keeping us alive but is succeptable to such diseases as Atheroclerosis. The result of this disease going untreated is usually deadly and thus there is a large field of research being conducted in order to better treat this problem. Professor Veneziani and his team use CFD to model the hemodynamics of this disease and its treatments, which include the placement of stents. On this team, Boyi Yang is the lead researcher on the placementment of stents and our algorithm is the first ever attempt at automating the stent location data collection process. Currently the collection process is done manually and is very tedious. It also cannot be exactly replicated. This means that when there are many patients to analyze, a large amount of time is required to create the 3D patient-specific models. It also means that if the stent location data were to be collected multiple times, the exact same locations might not be recorded which could have significant effects on the results of the CFD analysis. As seen in Fig. 4.5, our algorithm consistently captures about 1/3 of the locations of the stent structures present in the OCT images with very few errors. While this does not record all of the data necessary to construct the 3D geometries,



(a) Results Recorded Using Image Processing Algorithm.

(b) Results Recorded by Hand.

Figure 4.5: The automated results capture about 33% of the stent structures present as shown in this comparison between the algorithm's results on the left and the manually recorded results on the right.

it is a great first step towards full automation. In this scenario, given the lengthy process of manual data collection, a reduction of time of 1/3 will allow for significantly more patients to be analyzed and significantly more knowledge to be gained about treating this deadly disease. It also creates a very useful stepping stone for any future researchers to use in the pursuit of full automation.

Chapter 5

Acknowledgements

Foremost, I would like to express my gratitude to my thesis advisor Professor Alessandro Veneziani for his continuous support of my thesis, his patience and knowledge of the subject.

Secondly, I would like to thank Boyi Yang for his guidance through this process. His research is the basis for my thesis and without his direction I would not have been able to complete my work.

Bibliography

[1] Veneziani, Alessandro. (1998). *Mathematical and Numerical Modelling of Blood Flow Problems.* Ph.D. Thesis. University of Milan: Italy.

[2] Tuveri M., Veneziani, Alessandro. Tini-Brunozzi P., Formaggia L., Zanetti G., and Brotzu G. *Simulazioni numeriche in geometrie vascolari reali sane e stenotiche. Prospettive di un sistema integrato di Supporto alla Ricerca e alla Pratica Clinica. Presentato al XVIII Convegno della Societa Italiana di Patologia Vascolare, Roma, Italia, 13 Dicembre 1996.*

[3] Samieh, Ahmed. "Shape Recognition." *File Exchange.* Math Works, 03 July 2007. Web. 02 Mar. 2014.

[4] Otsu, Nobuyuki. "A Threshold Selection Method from Gray-Level Histograms". *IEEE Transactions on Systems, Man and Cybernetics* 9.1 (1979): 62-66. Print.

[5] Gonzales, Rafael C., Richard E. Woods, and Steven L. Eddins. *Digital Image Processing Using MATLAB*. Berkshire: McGraw Hill, 2011. Print.

[6] "Documentation Center." *Read Image from Graphics File*. Mathworks, n.d. Web. 16 Apr. 2014.

[7] "Documentation Center." *Convert RGB Image or Colormap to Grayscale*. Mathworks, n.d. Web. 16 Apr. 2014.

[8] "Documentation Center." *Adjust Image Intensity Values or Colormap*. Mathworks, n.d. Web. 16 Apr. 2014.

[9] "Documentation Center." *Global Image Threshold Using Otsu's Method*. Mathworks, n.d. Web. 16 Apr. 2014.

[10] "Documentation Center." *Convert Image to Binary Image, Based on Threshold*. Mathworks, n.d. Web. 16 Apr. 2014.

[11] "Documentation Center." *Remove Small Objects from Binary Image*. Mathworks, n.d. Web. 16 Apr. 2014.

[12] "Documentation Center." *Trace Region Boundaries in Binary Image*. Mathworks, n.d. Web. 16 Apr. 2014.

[13] "Documentation Center." *Measure Properties of Image Regions.* Mathworks, n.d. Web. 16 Apr. 2014.