

---

*Implementation and Comparison of Four  
Different Boundary Detection Algorithms for  
Quantitative Ultrasonic Measurements of the  
Human Carotid Artery*

**Masters Thesis**

**By**

**Ghassan Hamarneh  
Rafeef Abu-Gharbieh**

**Department of Applied Electronics  
December 1996**

---



*Implementation and Comparison of Four  
Different Boundary Detection Algorithms  
for Quantitative Ultrasonic Measurements  
of the Human Carotid Artery*

**Masters Thesis**

**By**  
**Ghassan Hamarneh**  
**Rafeef Abu-Gharbieh**

**Supervisor**  
**Tomas Gustavsson**

**December 1996**

**Department of Applied Electronics**  
**Chalmers University of Technology**



# *Abstract*

Recently, B-mode ultrasonography has been used for non-invasive quantitative measurements of the intima-media thickness (IMT) as an early indicator of atherosclerosis. Previously, these measurements relied on human visual judgement for manually identifying the echo coordinates. The need for automatic boundary detecting schemes stems from the fact that accuracy and precision of IMT measurements are limited by the intra- and interoperator variability.

This Thesis examines four methods for automated boundary detection, and describes the application of these methods to IMT calculation.

The first method uses a dynamic programming approach to identify the boundary path that minimizes a certain cost function. The method includes training in order to optimize the parameter values used in the detection scheme.

The second method uses the criterion of maximum gradient, with sub-pixel resolution, for defining the boundary points.

The third method is based on a mathematical model describing the intensity profile perpendicular to the two boundaries defining the IMT. Training is performed to obtain some parameters which are then used to identify the boundary points.

The last method examined is based on defining a template representing the intensity profile across a boundary. Then, the image is searched for candidate profiles that match the template. Eventually, the profile that is most likely to correspond to the real boundary is chosen.

The Thesis also presents a quantitative and qualitative comparison of the four examined methods for boundary detection. It was shown that the dynamic programming method possessed superior performance to the other methods in many aspects, most important of which is the high correlation coefficient between the IMT results obtained from this method and the manually obtained reference values. The correlation coefficient was found to be 0.96, 0.94, 0.63, and 0.85 for the dynamic programming, the maximum gradient, the model-based, and the matched filter method, respectively.



## *Acknowledgements*

With this Masters Thesis coming to its completion point, we would like to sincerely thank all of those who had made it possible for us to work successfully throughout all the different stages of this project.

We would specially like to thank Dr. Tomas Gustavsson for his supervision and continuous support.

Our thanks also go to Quan Liang for the assistance he gave us.

*Ghassan Hamarneh  
Rafeef Abu-Gharbieh*

---

---

# Contents

<b>CHAPTER 1</b>	<i>Introduction</i>	<b>1-1</b>
<b>CHAPTER 2</b>	<i>Background</i>	<b>2-1</b>
	The ultrasound imaging technique	2-1
	Anatomical definitions	2-2
	Representation of images	2-4
<b>CHAPTER 3</b>	<i>The Dynamic Programming Method</i>	<b>3-1</b>
	General description and mathematical representation	3-1
	The dynamic programming algorithm	3-3
	<i>Calculating average intensity</i>	3-3
	<i>Calculating the intensity gradient</i>	3-4
	<i>Forward scanning</i>	3-4
	<i>Backward tracking</i>	3-6
	IMT calculation	3-8
	Effect of changing different parameters	3-8
	<i>Continuity weight</i>	3-8
	<i>Gradient weight</i>	3-8
	<i>Intensity weight</i>	3-8
	<i>Intensity operator length</i>	3-8
	<i>Adding a parallelism cost term</i>	3-9
	<i>Order of continuity dependency</i>	3-9
<b>CHAPTER 4</b>	<i>The Maximum Gradient Method</i>	<b>4-1</b>
	The maximum gradient algorithm	4-1
	<i>Determining an approximate echo boundary</i>	4-2
	<i>Determining the initial edge</i>	4-2
	<i>Eliminating the weak edges</i>	4-6



---



---

	The IMT calculation procedure	4-7
	Effect of changing different parameters	4-8
	<i>Number of manually chosen points</i>	4-8
	<i>Location of manually chosen points</i>	4-8
	<i>Length of normal search</i>	4-8
	<i>Extent of manual intervention for modification</i>	4-9
<b>CHAPTER 5</b>	<b><i>The Model-Based Method</i></b>	<b>5-1</b>
	The model-based algorithm	5-1
	<i>Training</i>	5-1
	<i>Detection</i>	5-10
	IMT calculation	5-10
	Effect of changing different parameters	5-11
	<i>Initial conditions for the fitting search</i>	5-11
	<i>The model</i>	5-11
<b>CHAPTER 6</b>	<b><i>The Matched Filter Method</i></b>	<b>6-1</b>
	The matched filter algorithm	6-1
	<i>Training</i>	6-1
	<i>Detection</i>	6-7
	IMT calculation	6-7
	Effect of changing parameters	6-8
	<i>Choice of matched filter template</i>	6-8
	<i>Choice of point on template that defines the boundary</i>	6-8
<b>CHAPTER 7</b>	<b><i>IMT Calculation</i></b>	<b>7-1</b>
	The boundary coordinates	7-1
	<i>The dynamic programming method</i>	7-1
	<i>The maximum gradient method</i>	7-2
	<i>The model-based method</i>	7-3
	<i>The matched filter method</i>	7-3
	Methods for IMT calculation	7-3
	<i>General method</i>	7-3

---

*Special procedure for the maximum gradient  
method 7-4*

**CHAPTER 8**      *Results and Comparisons 8-1*

Introduction 8-1  
Results 8-2  
Comparison 8-7

**CHAPTER 9**      *Discussion and Conclusions 9-1*

Comparison between the boundary detection  
methods 9-1  
    *Calculation complexity 9-1*  
    *Programming effort 9-2*  
    *Required training 9-2*  
    *Extent of manual intervention 9-3*  
    *Variability between users & degree of automation 9-3*  
    *Similarity to a true boundary 9-4*  
    *Closeness to manual results 9-5*  
Conclusions 9-6

**CHAPTER 10**      *Our Implementations 10-1*

Software description 10-1  
    *The dynamic programming method 10-1*  
    *The maximum gradient method 10-2*  
    *The model-based method 10-2*  
    *The matched filter method 10-3*  
Implementation specifications 10-3  
    *Implementation environment 10-3*  
    *Training 10-4*  
    *The dynamic programming method 10-4*  
    *The maximum gradient method 10-5*  
    *The model-based method 10-5*  
    *The matched filter method 10-7*  
Description of the MatLab® files 10-7  
    *The dynamic programming method 10-7*

---

*The maximum gradient method 10-8*

*The model-based approach 10-10*

*The matched filter method 10-14*

**CHAPTER 11**

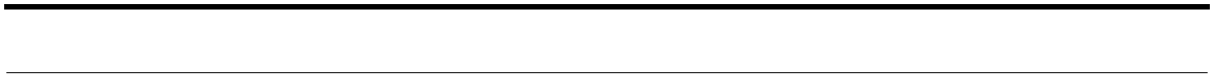
*Captures 11-1*

A look at the dynamic programming method  
implementation **11-2**

A look at the maximum gradient method  
implementation **11-5**

A look at the model-based method implementation  
**11-10**

A look at the matched filter method  
implementation **11-15**



---

Acquiring knowledge and information through image analysis techniques is becoming more and more important in these days. Recently, it has been shown that with high resolution B-mode ultrasonic images, vessel wall characteristics of the carotid arteries can be non-invasively assessed in an effective and accurate way. Non-invasive ultrasonography is presently used to assess the presence and extent of atherosclerosis. In order to do that, the carotid artery intima-media thickness (IMT) is measured and used as an early indicator of atherosclerosis in population-based studies<sup>1</sup>.

IMT is defined as the thickness between the lumen-intima and the media-adventitia boundaries in the carotid artery, and there is an obvious need for automatic boundary detection methods that are accurate and precise. Previously, these measurements relied on human visual judgement for manually identifying the boundary echo coordinates. The need for automatic boundary detecting schemes stems from the fact that accuracy and precision of IMT measurements are limited by the intra- and inter-operator variability<sup>2</sup>.

- 
1. see reference [9].
  2. see reference [7].

Unfortunately, many problems encounter the development of such automatic methods. Available images often possess weak echos, echo dropout and speckle noise<sup>1</sup> which make the detection of the boundaries, and thus the calculation of the IMT, a challenging task.

Different methods of boundary detection and IMT calculation were suggested earlier. Selzer *et al.*<sup>2</sup> reported a technique based on finding the maximal intensity gradient in a direction perpendicular to the vessel interfaces, i.e. the lumen-intima and the media-adventitia boundaries. In this way, a well defined measure of the IMT thickness can be obtained. There are, however, some disadvantages with this method. Because of frequent echo dropouts and scattering phenomena, points of maximal gradient do not always exist. Furthermore, in some cases these points do not relate to the true interface but to irrelevant structures located more deep into the vessel layer. A remedy to this would be to add a continuity criterion which forces the detected boundary to be smooth in the longitudinal direction. By combining intensity, gradient, and continuity criteria into a cost function and then minimize this function by some optimization procedure would be a plausible approach.

The aim of this Thesis is to examine four methods for boundary detection, and to test the application of these methods to B-mode ultrasonic images for the purpose of calculating the IMT.

In the following pages we try to outline briefly the topics covered by each chapter of this thesis.

Chapter 2 provides the reader with some basic background. Some useful medical terminology, a description of the ultrasound imaging technique, and an overview on image representation and visualization can be found in this chapter.

Chapter 3 investigates the dynamic programming procedure which minimizes a cost function for detecting boundary edges. This approach is the first of four boundary detection methods discussed in this thesis.

Chapter 4 investigates the second method which is based on defining the boundary edges to be those having the maximum gradient<sup>3</sup>. The method uses sub-pixel resolution to refine the results.

- 
1. see reference [6].
  2. see reference [7].
  3. which refers to the rate of change of the intensity.

---

---

Chapter 5 investigates a method based on modelling the intensity profile across the boundaries. It uses the models in a training process which results in some parameters that are used later on for detecting the boundaries in real situations.

Chapter 6 investigates the last method for boundary detection which searches for that part of the image that has an intensity profile which matches a predefined template. The template represents the intensity profile across the boundaries.

From now on, we will refer to the above four methods as:

1. *the dynamic programming method.*
2. *the maximum gradient method.*
3. *the model-based method.*
4. *the matched filter method.*

Chapter 7 treats the problem of IMT calculation from two sets of boundary coordinates.

Chapter 8 presents the calculated IMT values resulting from our implementations of all the boundary detection methods.

Chapter 9 discusses the results and presents some summaries and conclusions.

Chapter 10 and 11 contain more details and illustrations on our software implementations.





---

This chapter introduces the terminology and concepts related to this project.

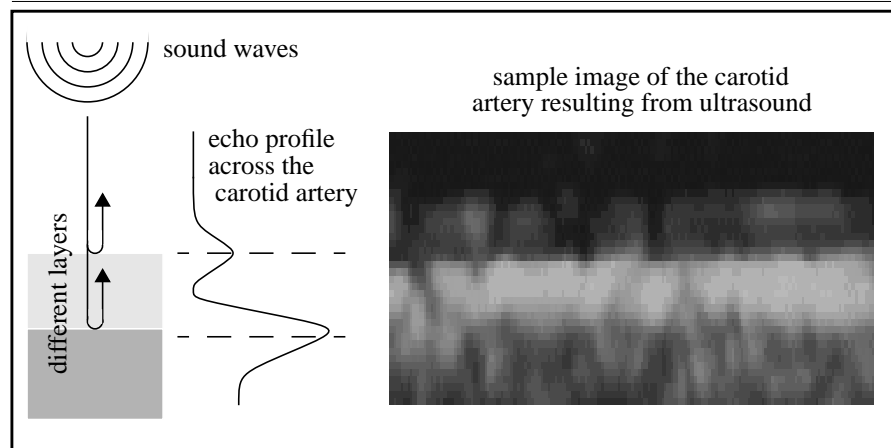
---

*The ultrasound imaging technique*

An ultrasound examination is a diagnostic procedure that uses very high frequency sound waves (1-10 MHz) to produce an image of many of the internal structures of the body. Ultrasound is widely used for it is non-invasive, safe, easy to use and relatively cheap.

Briefly, one can describe how ultrasound works by the following: When the ultrasound pulses propagate through the body and reach an interface between two tissues having different characteristics, i.e. different acoustic impedances, a certain amount of sound energy is reflected at the interface while the remainder propagates into the second tissue. We observe the reflected echoes and then use them to produce a suitable kind of display for the information we got about the structure of the examined region (See Figure 2.1).

**FIGURE 2.1. Producing ultrasound images**



The displayed data is a very helpful tool either for diagnosis or for conducting measurements. This thesis investigates some of the techniques used for quantitative analysis of the images produced via ultrasound.

---

### *Anatomical definitions*

Throughout this thesis, the interesting part of the body that we try to extract information about using ultrasonic images is a small part of the carotid artery. To be specific, we are interested in measuring the intima-media thickness (referred to as the IMT from now on). The next figure shows the different layers that constitute the artery wall and the IMT definition in a schematic diagram (See Figure 2.2), they are also shown within an ultrasound image (See Figure 2.3).

FIGURE 2.2. The artery structure

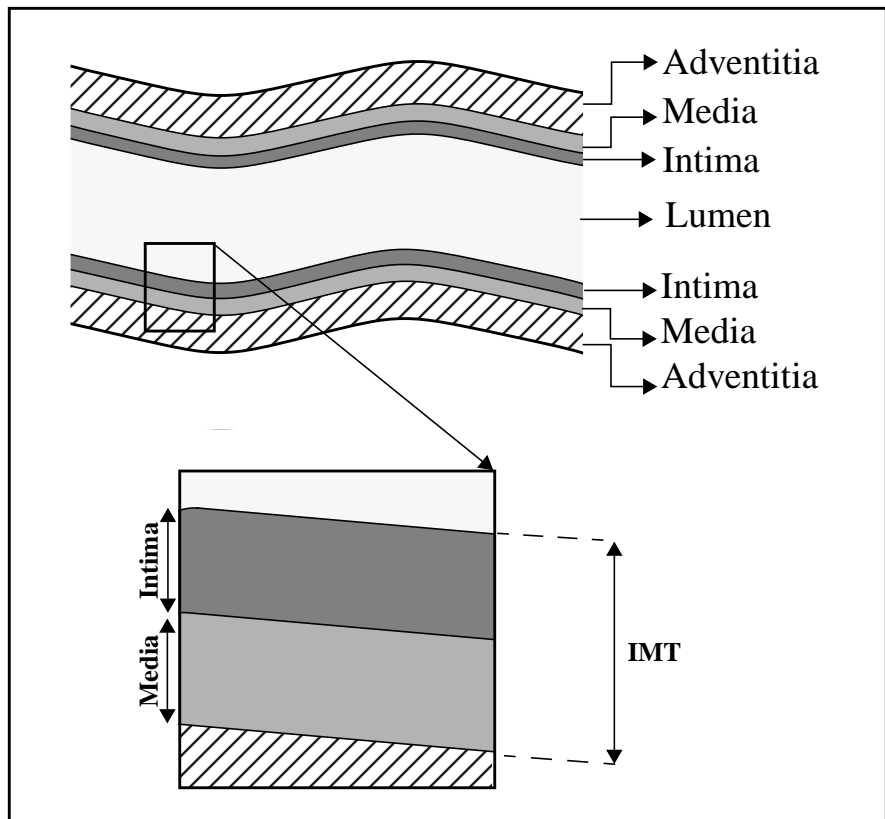
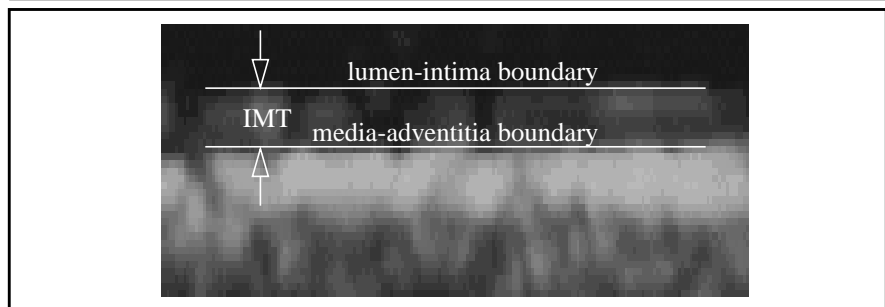


FIGURE 2.3. The artery structure in an ultrasonic image



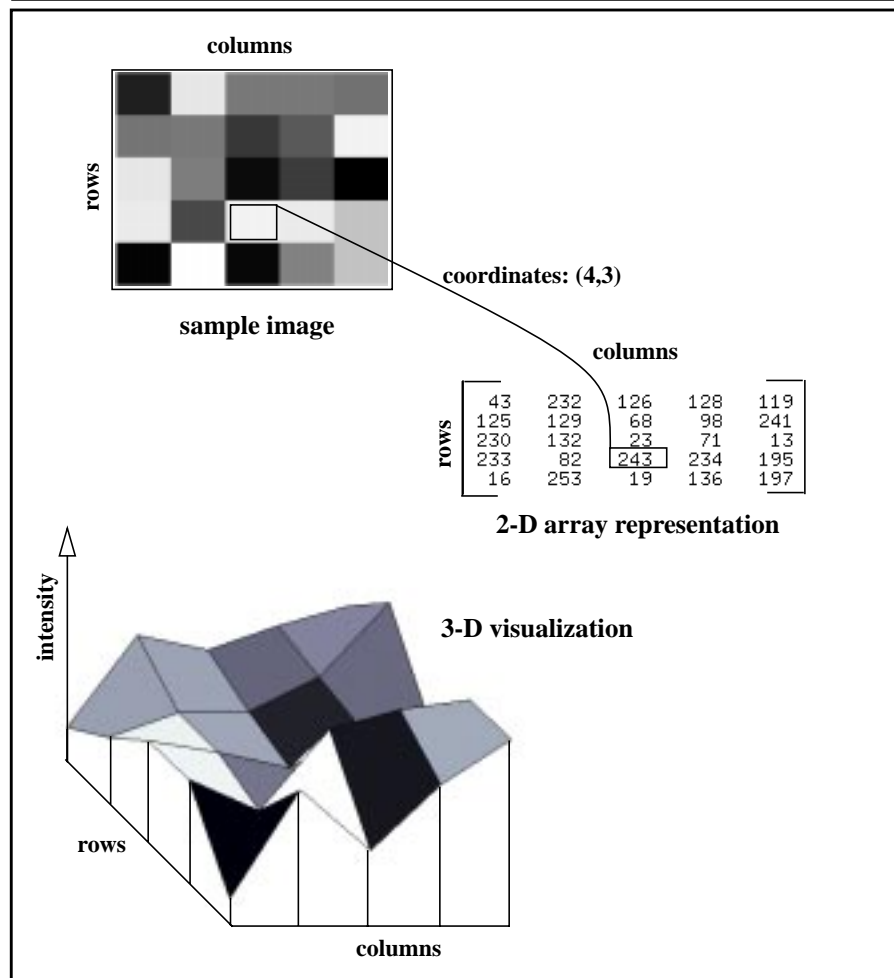
---

*Representation of images*

The images obtained from the ultrasound technique are<sup>1</sup> represented as a two dimensional array (2-D matrix) of grey level intensity values. This representation is essential for the analysis of the image data using software packages or simple programs<sup>2</sup>. For better visualization 3-D images are useful. An explanation of the representation and an illustration of the visualization of images are shown in Figure 2.4.

- 
1. after going through a number of operations, digitizing is an example.
  2. our implementations of the different algorithms that will be discussed later, were done using MatLab<sup>®</sup>.

FIGURE 2.4. 2-D matrix representation and 3-D visualization of a grey scale image



---

**Background**

---

## *The Dynamic Programming Method*

---

Dynamic programming tries to find the boundary by minimizing a certain cost function.

---

### *General description and mathematical representation*

This method is an optimal search technique for locating boundaries. Within the image studied, local measurements of the echo intensity, edge strength and candidate boundaries' discontinuity are extracted and included as weighted terms in a cost function. The algorithm then works on finding the boundary points that form a path minimizing the cost function.

The Image is scanned and all the possible boundary lines are considered polylines with N vertices represented as a vector  $p$  :

$$p = (p_1, p_2, \dots, p_{i-1}, p_i, \dots, p_N)$$

where

$p_{i-1}$  and  $p_i$  are horizontal neighbours.

$N$  is the horizontal length of a contour line.

The criteria for judging a polyline to be a valid boundary line is a cost function  $E_{sum}$  which is a sum of the local energies along this line. The optimum line is the one that minimizes the cost function:

$$E_{sum} = \sum_{i=1}^N E(p_i)$$

The local cost  $E(p_i)$  is a weighted sum of cost components:

$$E(p_i) = C_1 E_{discont}(p_{i-1}, p_i) - C_2 E_{int}(p_i) - C_3 E_{grad}(p_i)$$

where

$C_1, C_2, C_3$  are weighting factors<sup>1</sup>.

$E_{discont}$  is the value of polylines' discontinuity (measure of smoothness).

$E_{int}$  is the average brightness below the tested pixel.

$E_{grad}$  is the value of the intensity gradient (the rate of change of the intensity).

One can add other components to the cost function which may improve the result, such as parallelism. The added terms must only exploit features that generally characterize a boundary. In summary, strong intensity gradient, strong brightness and good continuity will give lower local cost. The desired boundary should have the global minimum cost.

By using the following dynamic programming algorithm, the optimum boundary is found without the need for exhaustive search of all the possible lines.

---

1. in an actual boundary detection situation, these values are obtained by training.



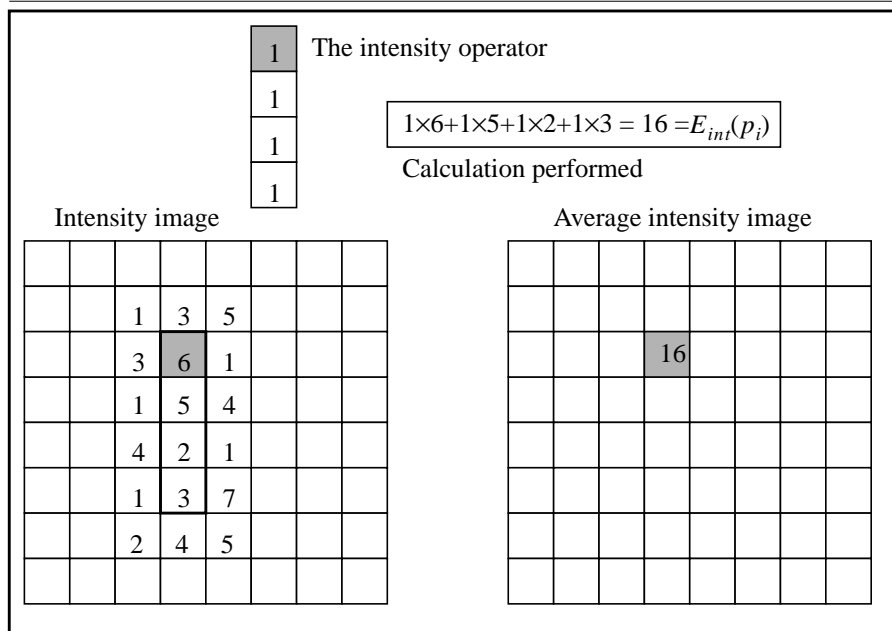
*The dynamic programming algorithm<sup>1</sup>*

The dynamic programming algorithm can be divided into four major steps. The first is calculating the average intensity. The second is calculating the intensity gradient. The third is forward scanning, and finally the backward tracking<sup>2</sup>.

**Calculating average intensity**

Given an image, for every pixel, the average brightness below it is calculated using a certain operator (See Figure 3.1). The larger the average brightness, the larger the intensity cost  $E_{int}$ . The operator is chosen to reflect the nature of the boundary we are searching for, which is a transition from a dark to a bright region. After applying the operator, an *average intensity matrix* is obtained.

**FIGURE 3.1. Example of applying an intensity operator**



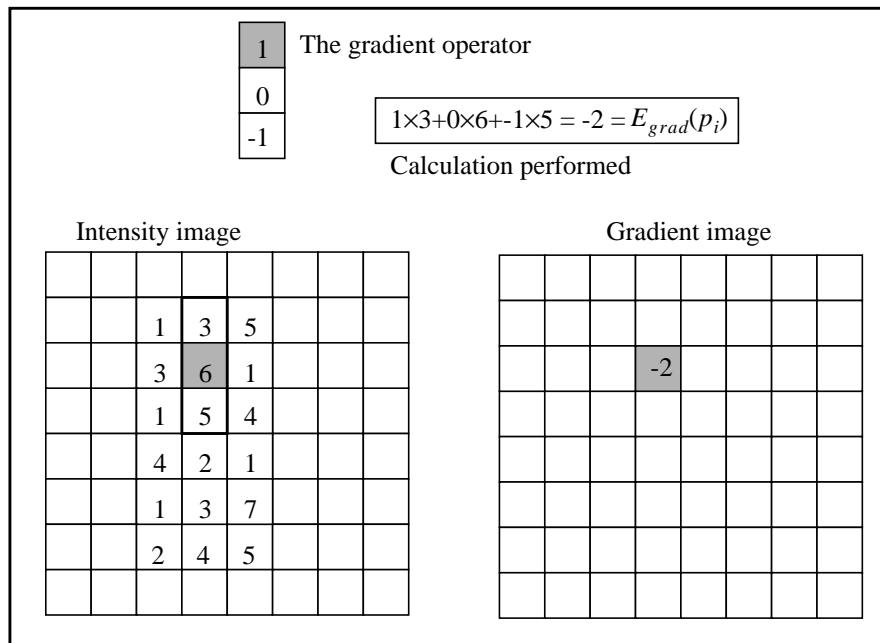
1. the examples shown here are merely simple ones that aim to illustrate the idea. For details on the actual implementation see chapter 10.
2. one should not forget the training required to obtain the weighting factors.

### Calculating the intensity gradient

A gradient operator is applied in a manner similar to applying the intensity operator (See Figure 3.2). A **gradient matrix** is then produced reflecting the change in intensity in a direction perpendicular to the boundary.

**FIGURE 3.2. Example of applying a gradient operator**

---



After calculating both the average intensity matrix and the intensity gradient matrix, the matrix  $-C_2E_{int} - C_3E_{grad}$  is found, which represents the contribution of intensity and gradient to the cost function.

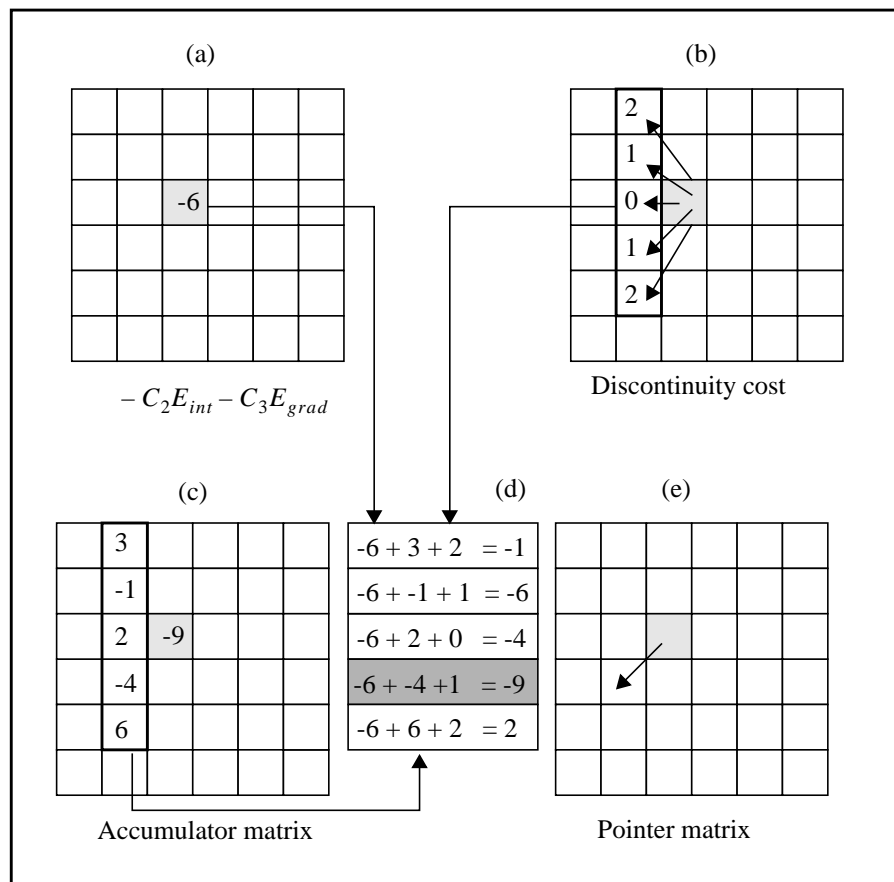
### Forward scanning

Scanning columns is now performed. The costs of moving from any of the pixels in the previous column to each pixel of the scanned column is calculated. This cost consists of the original intensity plus gradient cost of the scanned pixel (See Figure 3.3a), plus an additional cost resulting from discontinuity (See figure 3.3b), plus

the accumulated cost required to reach the point in the previous column (See Figure 3.3c).

The minimum of the costs is found (See Figure 3.3d) and this minimum cost is stored in an *accumulator matrix* (See Figure 3.3c) and the path it corresponds to is stored in a *pointer matrix* (See Figure 3.3e).

FIGURE 3.3. Forward scanning (accumulator and pointer matrices)

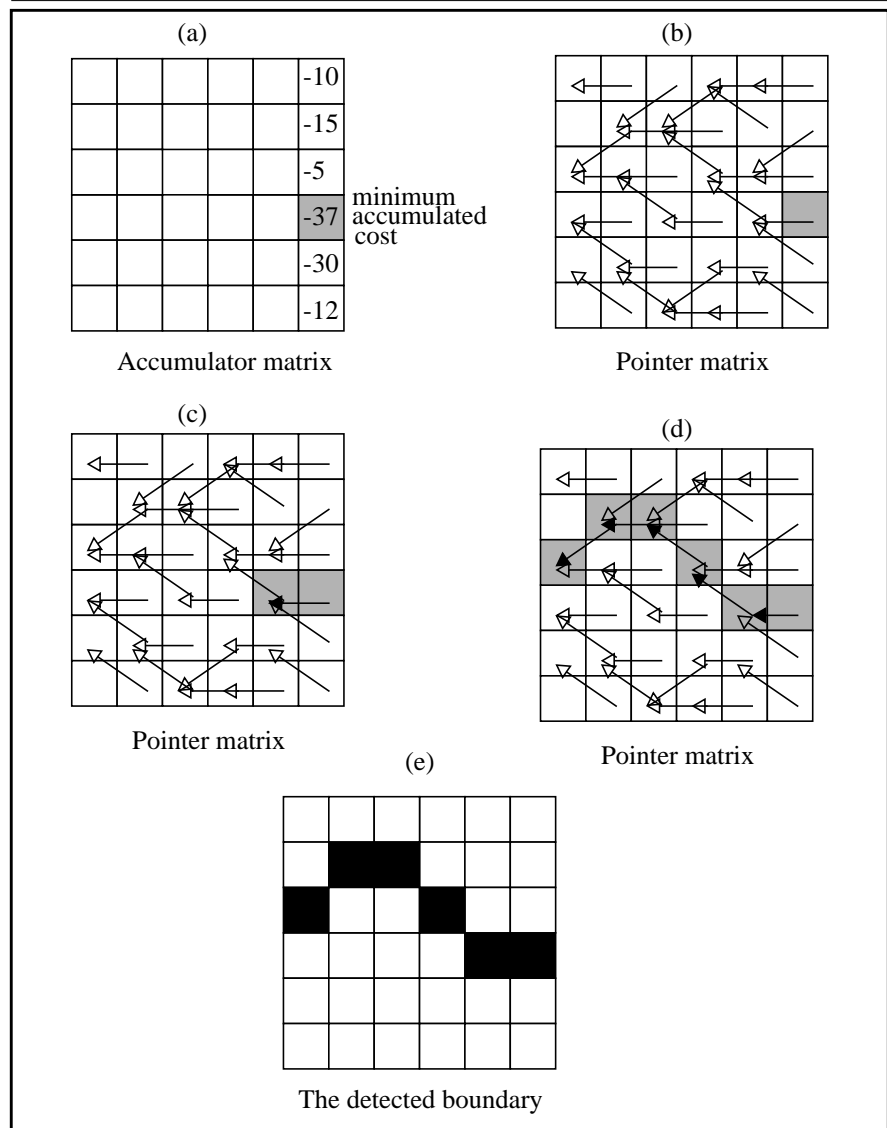


### **Backward tracking**

Tracking back is carried out after all the columns are scanned. Beginning at the point with minimum accumulated cost (See Figure 3.4a and 3.4b), tracking backwards is done according to the pointer matrix (See Figure 3.4c) until reaching the first column (See Figure 3.4d). As a result we end up with the detected boundary (See Figure 3.4e).

The above algorithm is applied once to detect the lumen-intima boundary, and another to detect the media-adventitia boundary.

FIGURE 3.4. Tracking back



---

### *IMT calculation*

At this point two sets of boundary coordinates are obtained. One set defines the lumen-intima boundary and the other defines the media-adventitia boundary. With these two sets, the IMT can be calculated according to the method described in chapter 7.

---

### *Effect of changing different parameters*

Changing different parameters in the dynamic programming method will result in detecting different boundaries. This section describes the effects of such changes.

#### **Continuity weight**

As the continuity weight increases, the detected boundary will tend to become closer to a straight line, and less jumps will appear between horizontal neighbouring pixels.

#### **Gradient weight**

As the gradient weight increases, the detected boundary will move towards points of maximum gradient<sup>1</sup> along the direction perpendicular to the boundary.

#### **Intensity weight**

As the intensity weight increases, the detected boundary will move towards the points surrounded by pixels with high intensity values<sup>2</sup>.

#### **Intensity operator length**

As the intensity operator length increases the detected boundary will tend to align with pixels surrounded (in the direction of the intensity operator) by longer runs of high intensity pixels.

- 
1. i.e. the points with maximum rate of change of the intensity
  2. by pixels we mean those which lie on the direction of the intensity operator (throughout its entire length)

### **Adding a parallelism cost term**

Assuming one boundary (ex. lumen-intima boundary) is detected, then adding a parallelism cost term, will force another detected boundary (ex. media-adventitia boundary) to be parallel to the first boundary to a certain extent determined by the parallelism weighting factor. This was seen to be hard to implement.

### **Order of continuity dependency**

If two horizontally neighbour pixels are different by  $n$  vertical pixels then for an  $m^{th}$  order continuity dependence, the discontinuity cost term will be proportional to  $n^m$ . By increasing this order of dependence, the same tendency encountered by increasing the continuity weight will be observed.





## *The Maximum Gradient Method*

---

In this chapter we will examine the second boundary detection method. This method defines the boundary point as the point of maximum gradient. Here the boundary point need not to be a certain pixel of the image, it can also be a sub-pixel if it is assumed to have the maximum gradient<sup>1</sup>.

---

### *The maximum gradient algorithm*

The algorithm can be easily summarized by the following three steps:

1. An approximate echo boundary is identified manually.
2. The approximate boundary is used to guide the computer edge finding algorithm to locate an initial edge.
3. The initial computer edges are tested for 'edge strength' and the weak edges are eliminated.

A detailed description of the method is given below.

---

1. the maximum gradient method is a modified version of the method presented in Reference [7].

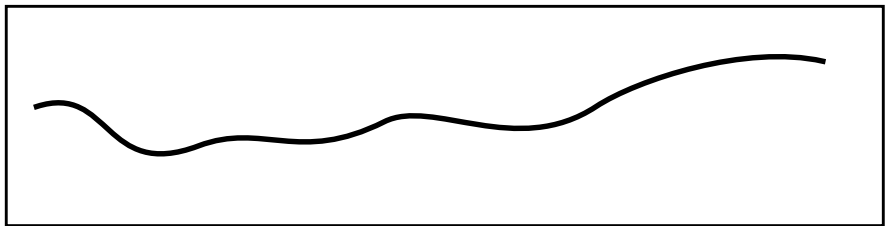
### **Determining an approximate echo boundary**

This is accomplished by manually choosing few points along the desired (and not clear) boundary (See Figure 4.1 and Figure 4.2). This can be easily done by pointing and clicking at the desired points with a mouse. These points are then connected by straight lines forming the first estimate of the boundary 'the approximate boundary' (See Figure 4.3).

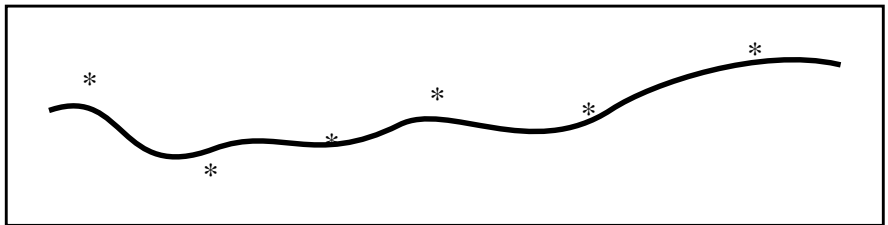
### **Determining the initial edge**

The algorithm searches for the edges in a direction perpendicular to the approximate boundary (See Figure 4.4), where at each point it examines 13 pixels in the image that lie along the line normal to the boundary at that point (See Figure 4.5) (the perpendicular was locally the same for all the points between any two selected-by-mouse points and was defined as the line normal to the linear segment connecting the two selected points). The examined points are equally taken from each side of the boundary (See Figure 4.6).

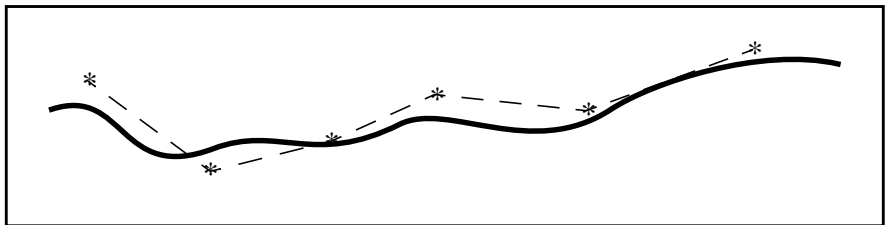
**FIGURE 4.1. The true boundary**



**FIGURE 4.2. The manually selected points**



**FIGURE 4.3. The approximate boundary resulting from connecting the points**



**FIGURE 4.4. Lines normal to the approximate boundary**

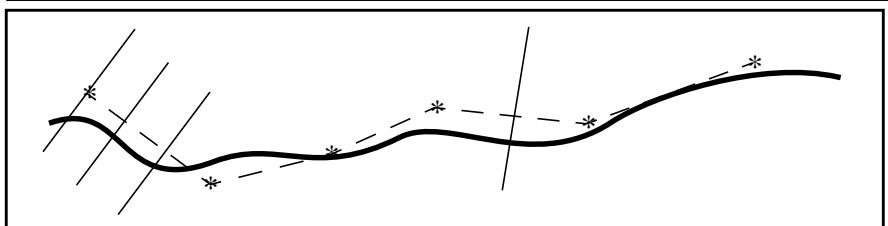


FIGURE 4.5. The 13 pixels examined on the normal

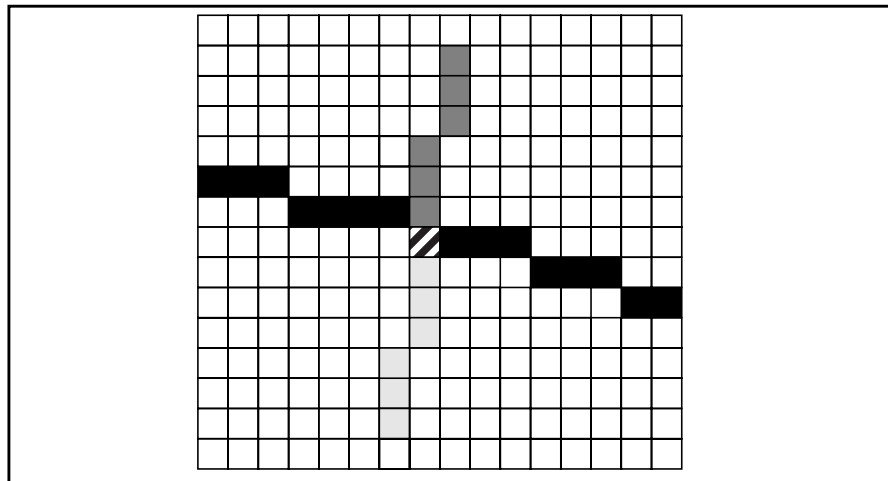
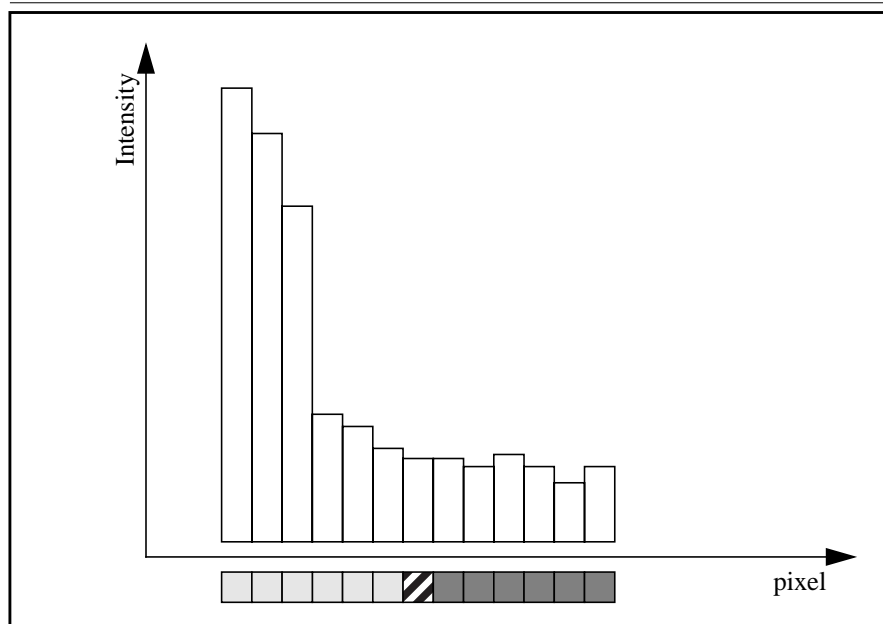
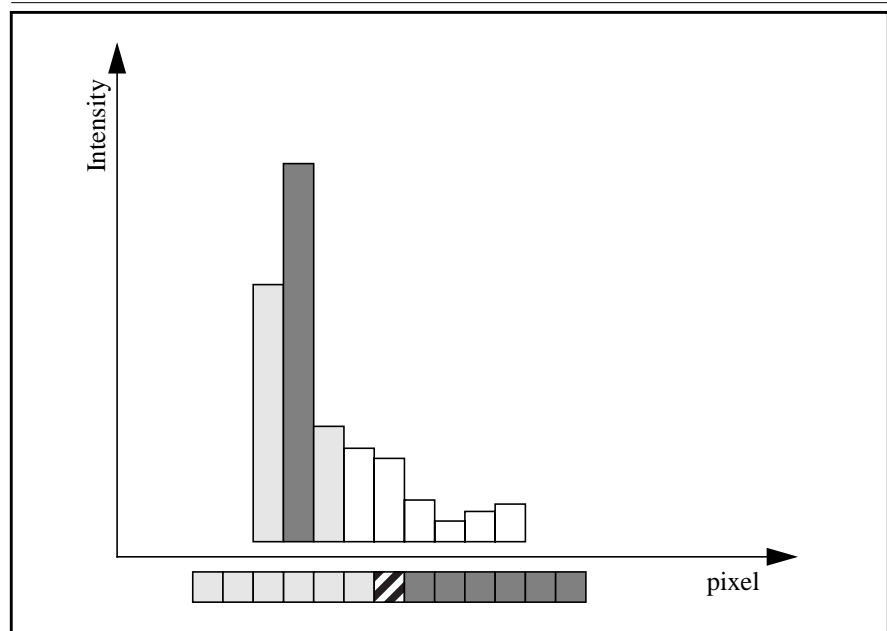


FIGURE 4.6. The intensity values at the 13 points



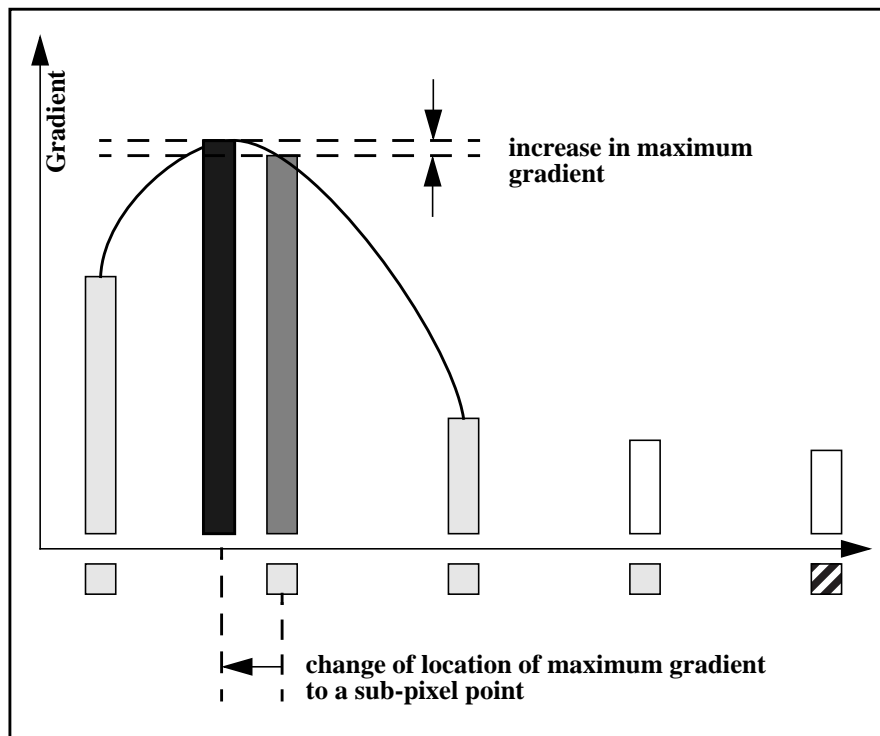
The initial boundary point is chosen to be the one at which the gradient (the rate of intensity change) of the pixels is maximum. The gradient at a particular pixel is computed as the derivative (at that pixel) of a second degree polynomial fit to five pixels centered on the pixel in question, the process is repeated for nine points starting from the third up to the eleventh (See Figure 4.7).

**FIGURE 4.7. The gradient values for the 9 inner pixels**



To find the sub-pixel maximum gradient location, the nine gradient values corresponding to the integer pixel locations are searched. This is done by fitting a parabola to the three gradient values centered around the largest value. The coordinates of the parabola's maximum is taken as the sub-pixel edge location. Now the points we located are considered to be those of an initial boundary (See Figure 4.8).

FIGURE 4.8. Demonstration of the effect of subpixel resolution



### Eliminating the weak edges

After the initial boundary edges are determined for all points, the gradient value for each edge is compared with the global maximum gradient value of all the initial boundary edges. Edge points with gradient values less than 20%<sup>1</sup> of the global maximum value are discarded while the remaining edges are considered acceptable.

The edge tracking process described above is applied to the lumen-intima and the media-adventitia echos.

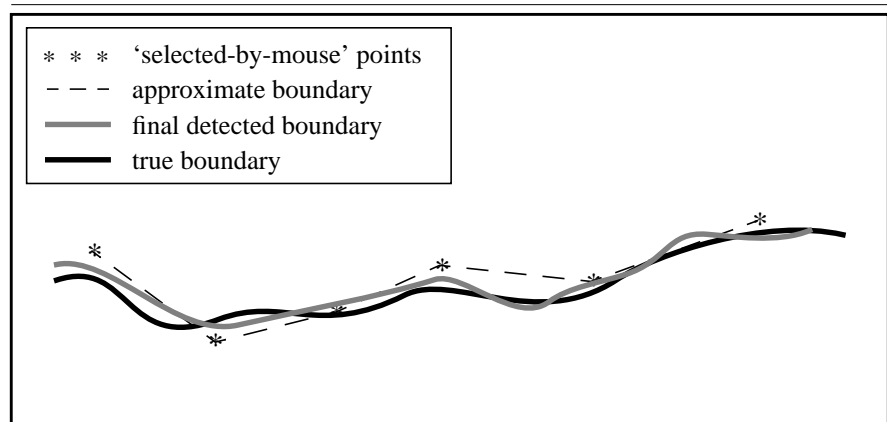
---

1. we followed the proposed numbers presented in reference [7].

The gaps in the lumen-intima boundary due to discarding weak edges are filled by linearly interpolated points from the nearest acceptable edges, meanwhile the gaps in the media-adventitia are not.

The progress of the algorithm is shown (See Figure 4.9).

**FIGURE 4.9. Progress of the algorithm**



---

### *The IMT calculation procedure*

At this point two sets of boundary coordinates are obtained. One set defines the lumen-intima boundary and the other defines the media-adventitia boundary. With these two sets, the IMT can be calculated according to the method described in chapter 7.

---

### *Effect of changing different parameters*

Changing different parameters in the maximum gradient method will result in detecting different boundaries. This section describes the effects of such changes.

#### **Number of manually chosen points**

By choosing more points manually, a more curved initial boundary can be defined, which means that it would be more likely that the final detected boundary would follow an extremely curved boundary (a hard-to-detect boundary).

#### **Location of manually chosen points**

Since the location of the manually chosen points will determine the piece-wise linear initial boundary which itself will guide the normal scanning procedure, and since the normal search for maximum gradient is limited (in our case to six pixels below and six pixels above the initial boundary), then choosing points far from the boundary will cause the boundary detection to fail and the algorithm will choose the point with maximum gradient just within the searching area which would be far from the desired boundary in this case.

#### **Length of normal search**

As the number of the pixels searched along the normal of the initial boundary is increased, this algorithm may find the maximum gradient farther from the initial boundary and this has pros and cons. Sometimes this far search will result in finding a point with maximum gradient which is not related to the desired boundary. On the other hand, this far search could result in finding the desired (correct) boundary point even if our initial estimation for its location was far from the real location.

To summarize, if the manually chosen points are chosen accurately and as close as possible to the expected true boundary, then the length of normal search is preferred to be small, meanwhile a rough choice of initial points requires farther normal search, but then one must take care not to detect boundary points belonging to another close-by boundary or to fall into fading regions which will result in a very discontinuous boundary.



**Extent of manual intervention for modification**

It is almost always the case that the detected boundary will have some points which are obviously wrong. This situation occurs more frequently when the image has more echo drop-out. In this case it is wise to manually modify these erroneous points. Although this manual intervention fixes the boundary (looking from the eyes of one operator), it increases the variability of the detected boundary between different operators.

---

The Maximum Gradient Method

---

## *The Model-Based Method*

---

In this chapter we will investigate the model-based approach for boundary detection. In this method we try to fit a parameterized model to the image intensity profile, and then using a previously defined criterion obtained from training, we choose a certain point on our model to be the detected boundary point.

---

### *The model-based algorithm*

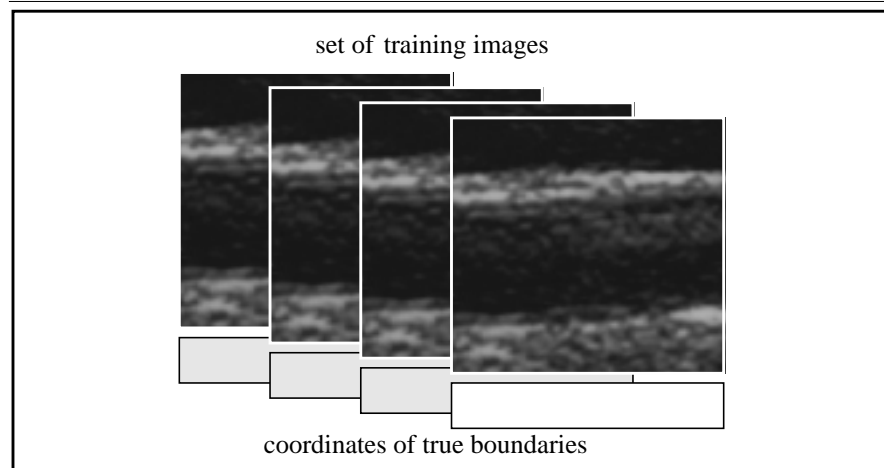
The model-based algorithm is divided into two major parts, the training part and the detection part.

#### **Training**

In this part we are actually training the parameter of the criterion that defines the boundary point on the model.

A set of images with assumed known true boundary are given, which we refer to as the set of training images (See Figure 5.1).

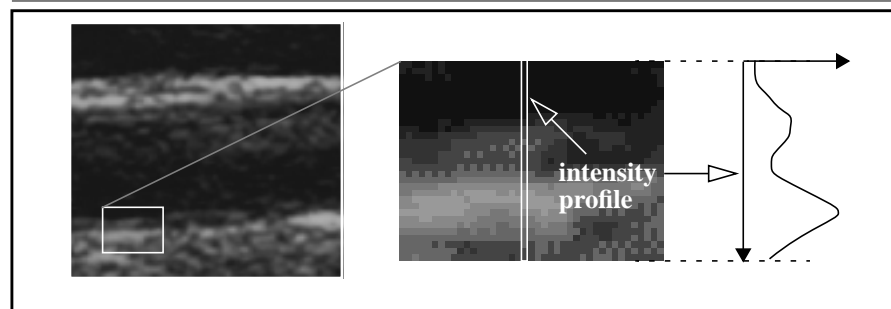
**FIGURE 5.1. The set of training images**



For each image in the training set, the intensity profiles perpendicular to the direction of the boundary path are found.

The interesting part that contains both the intima and media layers in all the intensity profiles is cut (See Figure 5.2).

**FIGURE 5.2. Extraction of the interesting intensity profile**



A parameterized model similar to the resulting profiles is then defined.

Here we investigated two types of parameterized models that can be used. The first one is a '*sum-of-two-gaussians*' model (See Figure 5.3), while the second one is a '*piece-wise linear*' one (See Figure 5.4).

The ‘sum-of-two-gaussians’ model can be described by the equation:

$$f(x) = p_1 + \left( p_2 \cdot e^{-\frac{(x-p_3)^2}{2 \cdot p_4}} + p_5 \cdot e^{-\frac{(x-p_6)^2}{2 \cdot p_7}} \right)$$

where

$x$  represents the row number.

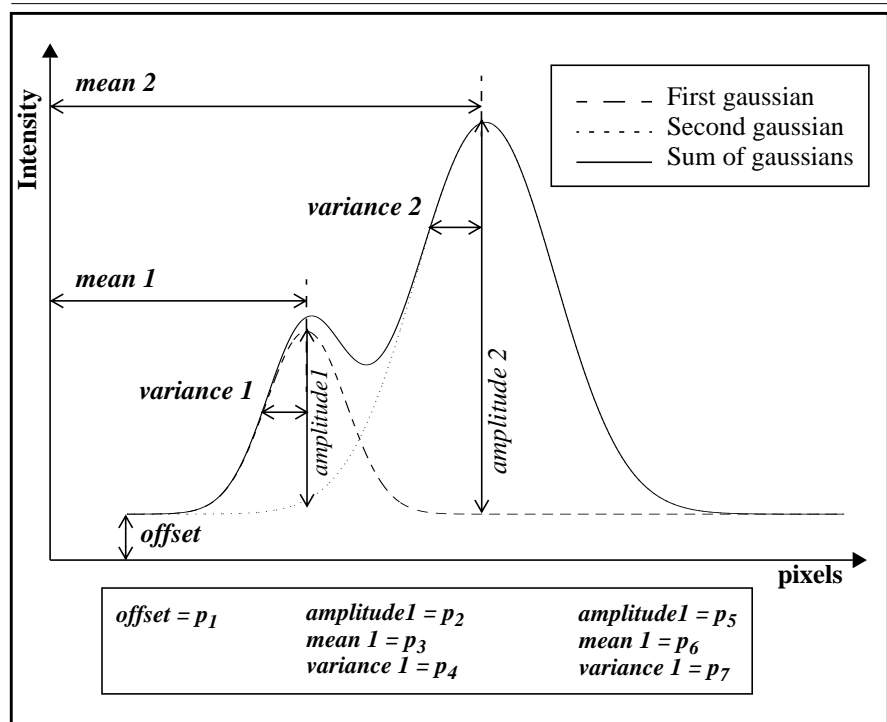
$p_1$  is the offset parameter.

$p_2, p_5$  are the amplitude parameters for the two gaussians.

$p_3, p_6$  are the mean parameters for the two gaussians.

$p_4, p_7$  are the variance parameters for the two gaussians.

**FIGURE 5.3. ‘Sum-of-two-gaussians’ model with its parameters**



On the other hand, the 'piece-wise linear' model can be represented by the equation:

$$f(x) = \left\{ \begin{array}{ll} p_1; & 0 \leq x \leq p_2 \\ p_3x + a; & p_2 \leq x \leq p_4 \\ p_5x + b; & p_4 \leq x \leq p_6 \\ c; & p_6 \leq x \leq p_7 \\ p_8x + d; & p_7 \leq x \leq p_9 \\ p_{10}x + e; & p_9 \leq x \leq p_{11} \\ f; & p_{11} \leq x \leq \text{last column} \end{array} \right\}$$

where

$x$  represents the row number.

$p_1$  is the starting offset parameter.

$p_2$  is the end of the constant offset parameter.

$p_3$  is the first up-slope parameter.

$p_4$  is the end of the first up-slope parameter.

$p_5$  is the first down-slope parameter.

$p_6$  is the end of the first down-slope parameter.

$p_7$  is the end of the middle constant level parameter.

$p_8$  is the second up-slope parameter.

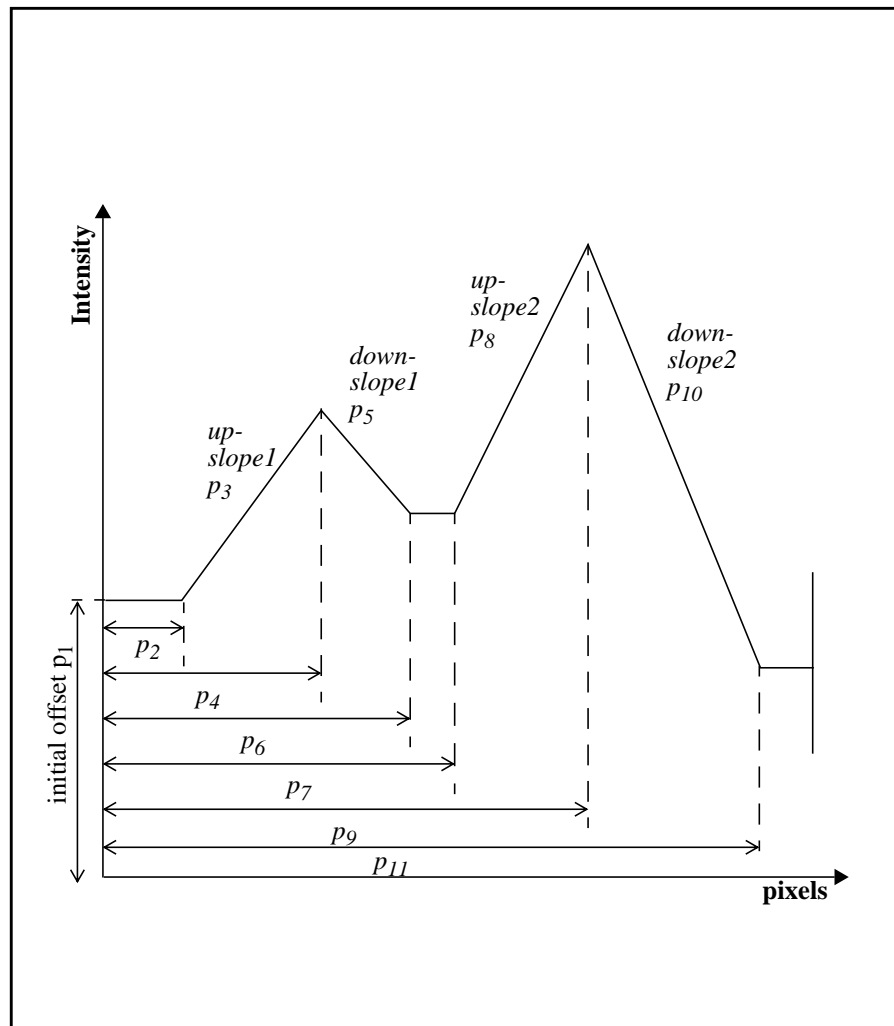
$p_9$  is the end of the second up-slope parameter.

$p_{10}$  is the second down-slope parameter.

$p_{11}$  is the end of the second down-slope parameter.

$a, b, c, d, e, f$  are constants needed for the continuity of the model.

FIGURE 5.4. Piece-wise linear model with its parameters



The parameters of the model which best fits each profile are found<sup>1</sup>. By best fit we mean the parameters that will give (or try to give) a model which has the minimum square error from the intensity profile. The square error to be minimized can be expressed for both types of models as:

$$\varepsilon = \sum_x (f(x) - i(x))^2$$

where

$f(x)$  is the model fitted to the intensity profile  $i(x)$ .

Illustrations of a model fit to an intensity profile can be seen for the ‘sum-of-two-gaussians’ model (See Figure 5.5) and for the ‘piece wise linear’ model (See Figure 5.6)

---

1. we used ***constr*** function from MatLab’s Optimization Toolbox. ***constr*** uses a Sequential Quadratic Programming (SQP) method. For more details see reference [8].



FIGURE 5.5. Fitting the 'sum-of-two-gaussians' model to an intensity profile

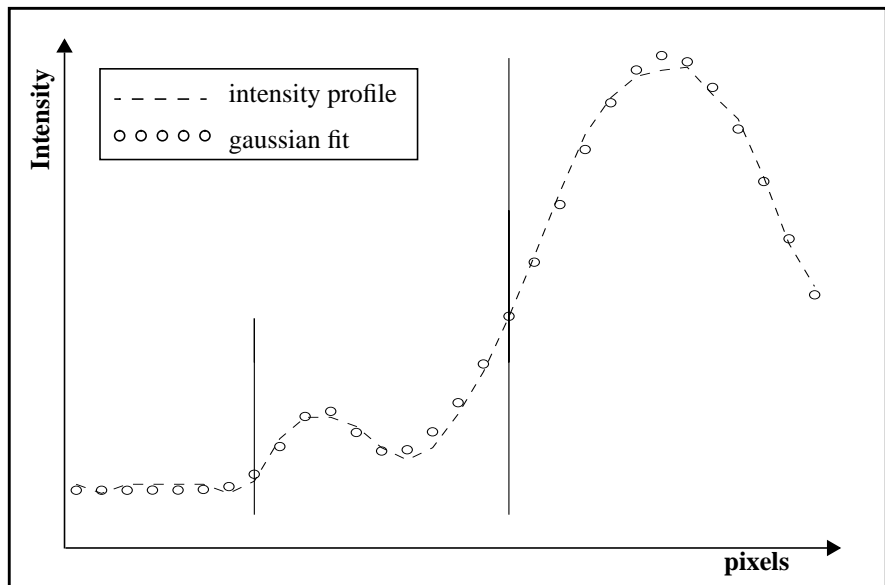
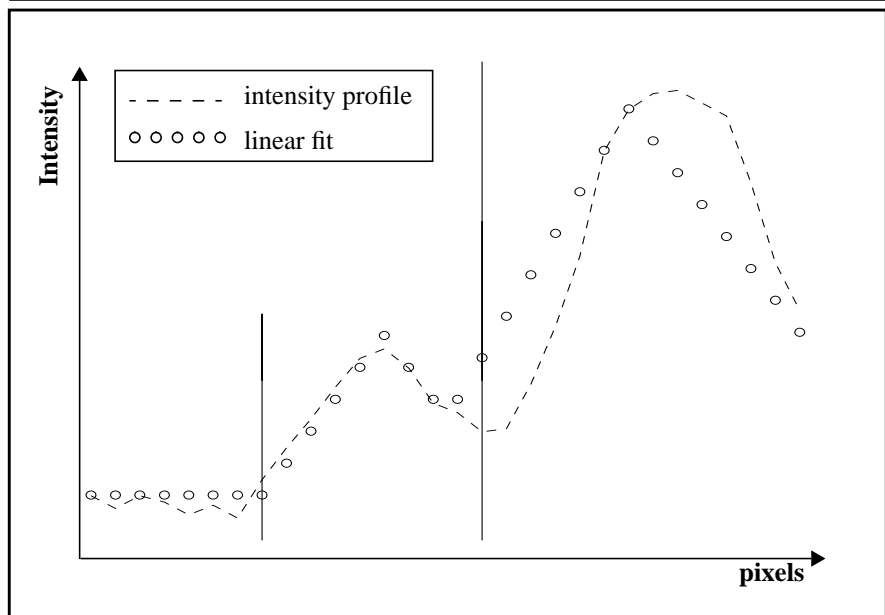


FIGURE 5.6. Fitting the 'piece-wise linear' model to an intensity profile

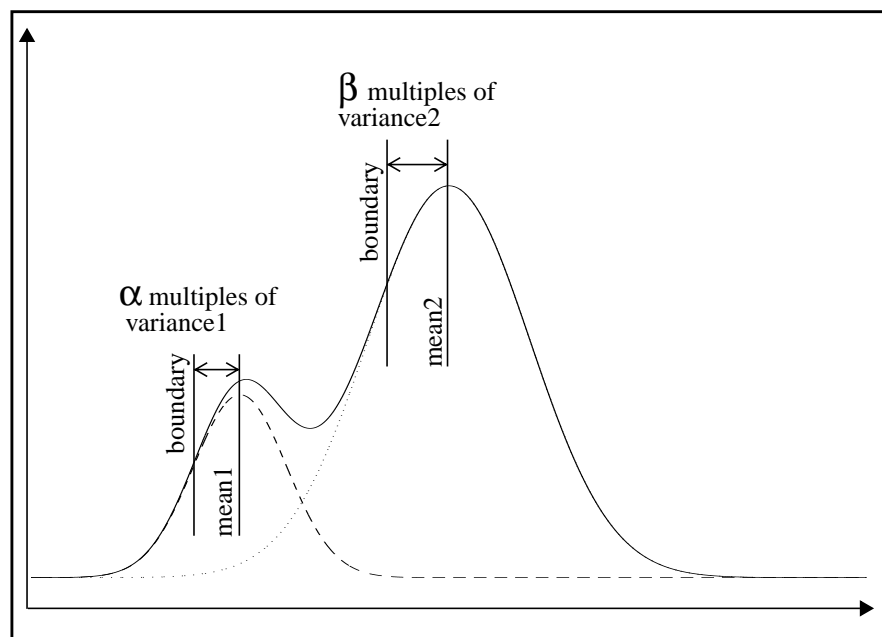


Since the true boundaries are known in the training set, the parameters of the criteria defining the true boundary are calculated.

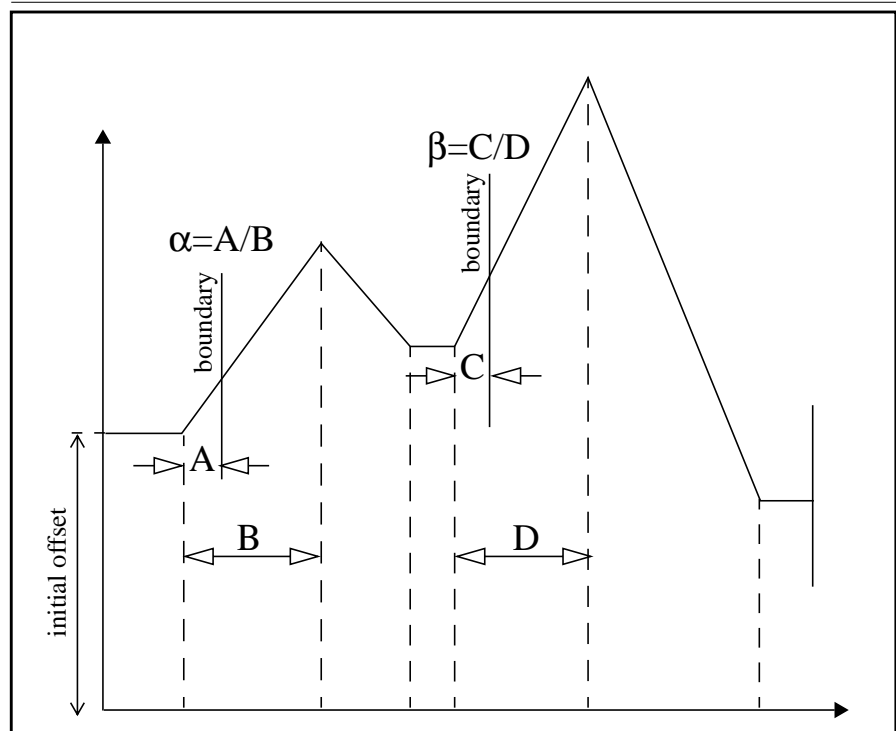
The parameter calculation is performed for all the intensity profiles of the set of training images. And hence a set of parameters are obtained, and by excluding unreasonable answers, and then averaging the remaining parameters, the parameter of the criterion defining the boundary is obtained. This parameter will be used in the actual detection step, described later on.

It should be noted that there are two criteria for each model, one for defining the lumen-intima boundary point on the model, and a similar one for defining the media-adventitia boundary (See Figure 5.7 and 5.8).

**FIGURE 5.7. Criteria defining the boundary for the ‘sum-of-two-gaussians’ model**



**FIGURE 5.8. Criteria defining the boundary for the ‘piece-wise linear’ model**



So, as a result of this training step we obtain (for a given model):

1. a parameter for the criterion that defines the lumen-intima boundary ( $\alpha$ ).
2. a parameter for the criterion that defines the media-adventitia boundary ( $\beta$ ).

### **Detection**

In this part an actual boundary detection is performed, using the results obtained from training.

We are given an image with unknown true boundary. The intensity profiles perpendicular to the boundary along the direction of the boundary path are found. Then the interesting part containing the intima and media layers is cut from each of the intensity profiles. The parameters of the model that best fit the profiles are then found.

Up to this point, the same steps applied in the training part are applied also in the detection part, but as we will see next, different routes are now taken.

The trained parameter of the criterion that defines the boundary point is applied to each intensity profile model and thus we obtain a boundary point on the profile.

More specifically, we had two trained parameters and thus we apply one parameter to detect the lumen-intima boundary point and the other to detect the media-adventitia boundary point.

---

### *IMT calculation*

At this point two sets of boundary coordinates are obtained. One set defines the lumen-intima boundary and the other defines the media-adventitia boundary. With these two sets we are able to calculate the IMT according to the method described in chapter 7.

---

### *Effect of changing different parameters*

Changing different parameters in the model-based method will result in detecting different boundaries. This section describes the effects of such changes.

#### **Initial conditions for the fitting search**

One of the steps of the model-based algorithm, is to fit a parameterized model to an intensity profile. In fact, this may be considered the most important step. In the case of the ‘sum-of-two-gaussians’ model there are 7 parameters that can be varied in order to find the best fit. In the case of the ‘piece-wise linear’ model there are even more degrees of freedom since there are 11 parameters. In order to fit the model, a search is done for the best set of parameters that minimize the square of the error between the model and the intensity profile. It was found that the fitted model relies heavily and is highly sensitive to the initial conditions. This is due to the large number of degrees of freedom in the searching process. This could cause the optimization process to end up on local minima of the error function and not on the global one. To summarize, we can say that the initial values should be chosen very carefully in order to get acceptable model fitting for a large number of images.

#### **The model**

It is obvious that the detected boundary depends on the model used, and the definition of the criterion defining the boundary. In our implementations we found that the ‘sum-of-two-gaussians’ model fits the intensity profile in a much better manner than the ‘piece-wise linear’ model. This can be explained by the fact that the sound pulses<sup>1</sup> are claimed (by the manufacturers of ultrasound imaging devices) to be sent with a gaussian envelope<sup>2</sup>, where this envelope is detected, after it is reflected by the body tissues, to produce the ultrasound images.

- 
1. meaning the waves used to produce the ultrasound images.
  2. which means that the shape of the wave is gaussian.



## *The Matched Filter Method*

---

This method investigates the use of one-dimensional spacial matched filtering for identifying boundaries.

---

### *The matched filter algorithm*

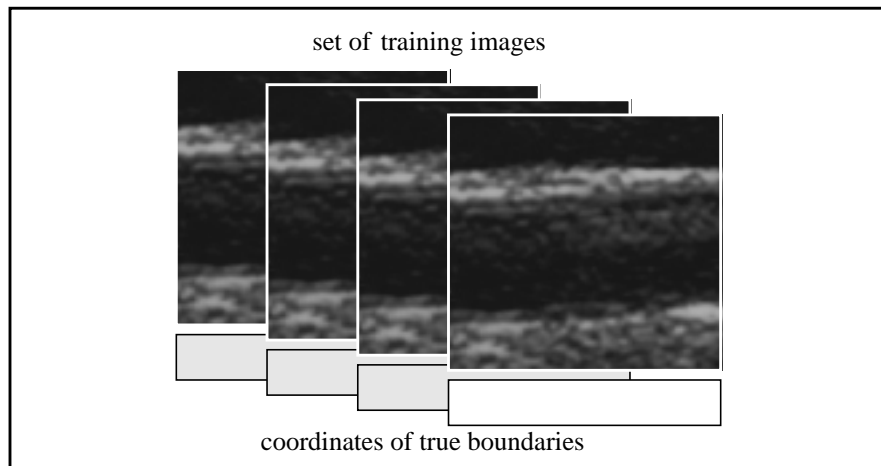
The matched filter method is based on a simple mathematical operation which is cross correlation of a reference signal (template) with the signal to be processed. The template is a model of the signal to be detected, which is, in our case, the intensity profile of the ultrasonic image, in the direction perpendicular to the boundary. The output of the correlation process is maximized when the input signal best matches the template.

The application of this method constitutes of two main steps; training and detection, which are described below.

#### **Training**

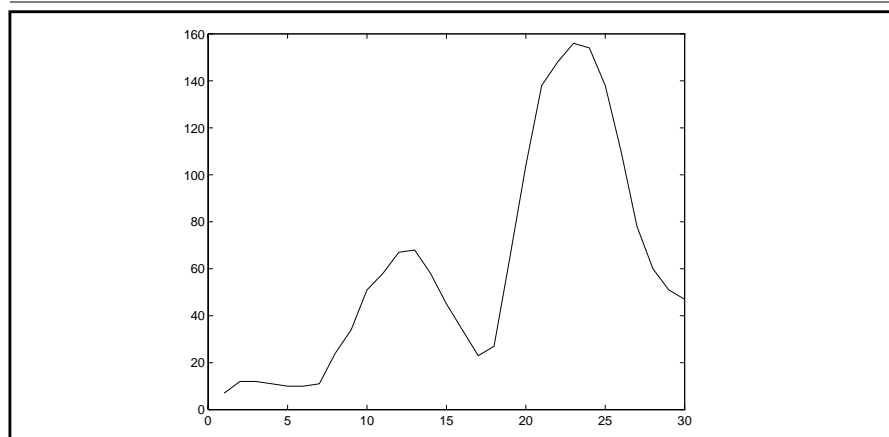
A set of ultrasound images is given with associated coordinates representing the assumed true boundary (See Figure 6.1).

**FIGURE 6.1. The set of training images**



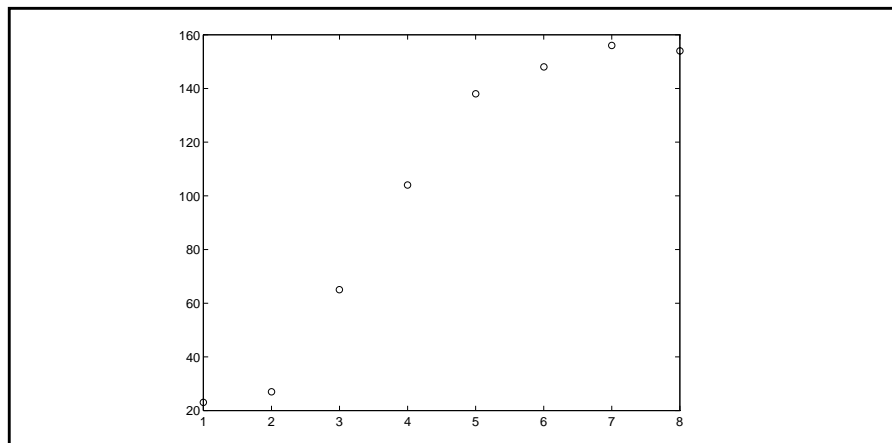
For each image, all its intensity profiles (See Figure 6.2) are convolved with a pre-defined matched filter (a template of intensity values representing a typical boundary profile) (See Figure 6.3). The output of the convolution (See Figure 6.4) is then smoothed to remove small undesired peaks (See Figure 6.5).

**FIGURE 6.2. An intensity profile**

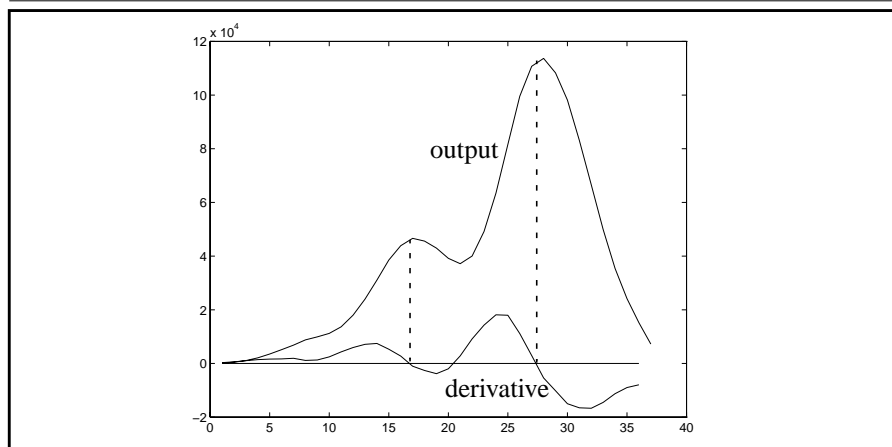




**FIGURE 6.3. A matched filter**

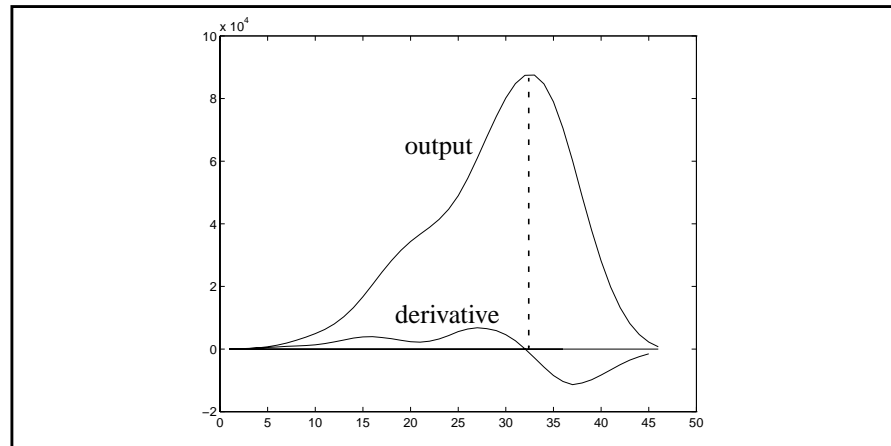


**FIGURE 6.4. Output of convolving the matched filter with the profile**



**FIGURE 6.5. Result of smoothing the convolution output**

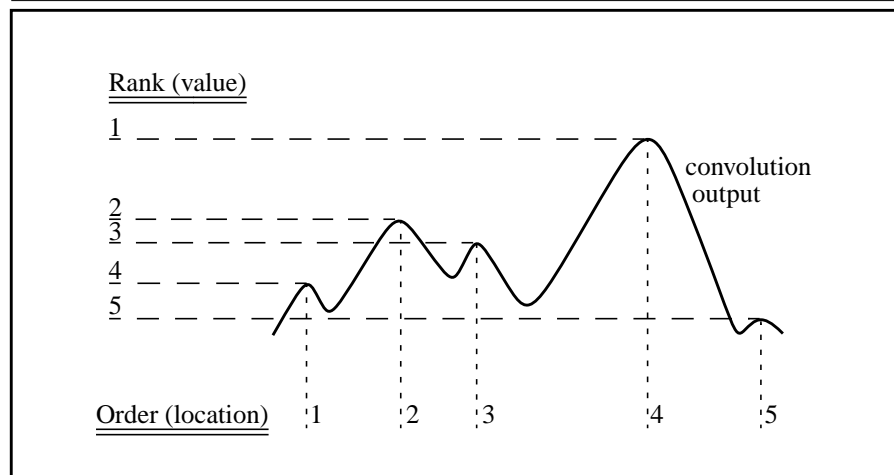
---



Since the true boundary point for each profile is known in the training set, the peak from the smoothed convolution output, which is closest to it is identified.

The closest peak is then studied, and its rank among the other peaks is determined, i.e. whether it is the maximum peak in value, or the second maximum, and so on. The order of the peak is also evaluated, i.e. whether it is the first peak resulting from the convolution, or is the second, and so on (See Figure 6.6).

FIGURE 6.6. The rank and order properties of a peak



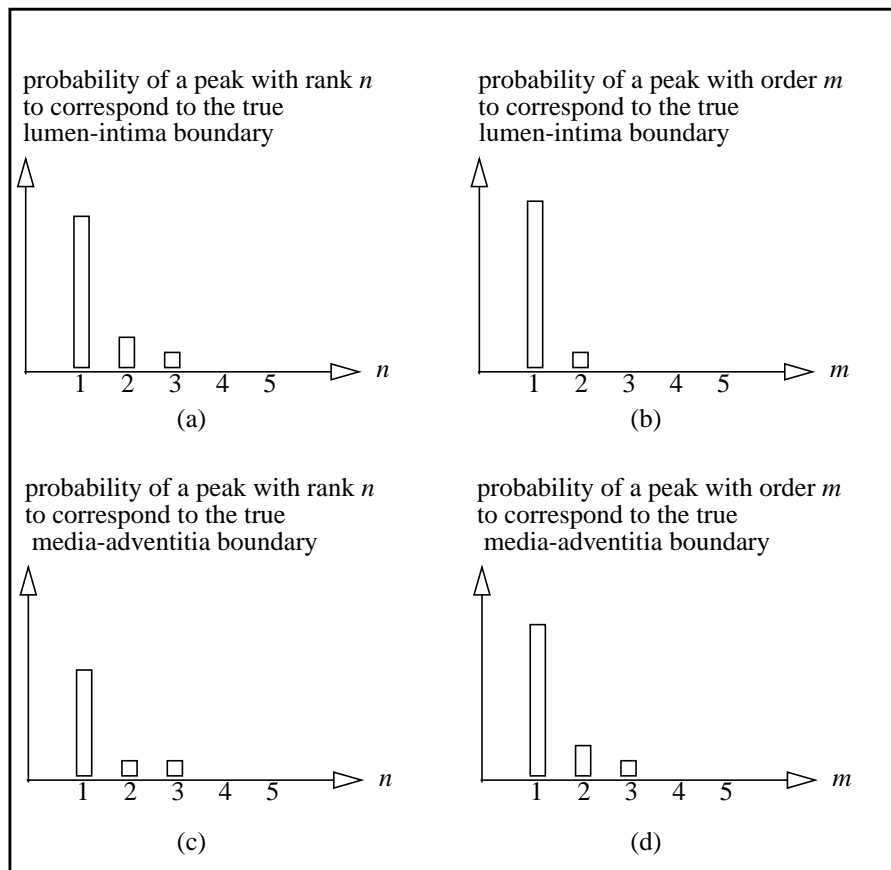
At this point a histogram representing the occurrences of peaks is established and updated for each smoothed convolution output. If the peak closest to the true boundary was the  $n^{\text{th}}$  maximum and had the  $m^{\text{th}}$  location, then the probability of the true boundary corresponding to a peak with those specifications is increased.

The true boundary in the previous discussion corresponds to the media-adventitia boundary. The same is done for the lumen-intima boundary after cutting the image at the location of the true media-adventitia boundary. This is because the later boundary is stronger and is always easily detected in the first run of convolution.

So, as a result of the training step we end up with four histograms representing the following probabilities:

1. probability of the  $n^{\text{th}}$  maximum peak (in value) to correspond to the true lumen-intima boundary (See Figure 6.7a).
2. probability of the  $m^{\text{th}}$  peak (in location) to correspond to the true lumen-intima boundary (See Figure 6.7b).
3. probability of the  $n^{\text{th}}$  maximum peak (in value) to correspond to the true media-adventitia boundary (See Figure 6.7c).
4. probability of the  $m^{\text{th}}$  peak (in location) to correspond to the true media-adventitia boundary (See Figure 6.7d).

FIGURE 6.7. Typical histograms resulting from training



### **Detection**

In this part, it is required to detect the two boundaries in order to calculate the intima-media thickness.

An image is given, and it is required as a first step to find the media-adventitia boundary. Each profile of the image is convolved with the same matched filter used in training. The output of the convolution is smoothed in the same way as in the training. All the peaks of the smoothed convolution output are found. Then each peak is associated with a rank and order values, i.e. a certain peak is the  $n^{th}$  maximum peak and the  $m^{th}$  peak in location.

The probability of each peak being the peak that corresponds to the true boundary is then found. This probability can be restated for a peak with rank  $n$  and order  $m$  by the probability of a peak with rank  $m$  and order  $n$  being the true peak. And by assuming that the rank of the peak is independent of the order of the peak, we can calculate the probability as the multiplication of the two probabilities which in turn are obtained from the results of the training step.

At this point, all peaks are candidates to represent the true boundary point, but the peak with maximum probability is taken as the true boundary peak.

Now the image is cut down to the detected media-adventitia boundary. The same detection procedure is now applied but for the lumen-intima boundary.

---

### *IMT calculation*

At this point two sets of boundary coordinates are obtained. One set defines the lumen-intima boundary and the other defines the media-adventitia boundary. With these two sets we are able to calculate the IMT according to the method described in chapter 7.

---

### *Effect of changing parameters*

#### **Choice of matched filter template**

The shape of the matched filter template should reflect the shape of the boundary intensity profile<sup>1</sup>. The form of transition should also reflect the form of transition of the boundary profile. So, we used two matched filter templates, one for the lumen-intima boundary and another for the media-adventitia boundary, since they differ in the form of transition. It follows that as the matched filter template changes, then the probability of matching the template to new different boundaries increases.

On the other hand, the length of the matched filter is also very important. It should be sufficiently long to contain the necessary information reflecting the transition region. Also it should be short enough to obtain distinguished matches when two intensity profiles similar to the template are close to each other. The later point is essential in our case, since the lumen-intima and media-adventitia boundaries are rather close.

#### **Choice of point on template that defines the boundary**

The point on the template that defines the boundary would not affect the calculated IMT if the same matched filter is used to detect both the lumen-intima and the media-adventitia boundaries. This results from the fact that there will be the same shift in the row coordinates for both boundaries. But since we stated the need for a different template for each boundary, then the edge defining point should be chosen carefully. It could be chosen as the point with maximum intensity change (maximum gradient), or more practically, the definition of this point should be obtained from training on a set of images with known boundaries<sup>2</sup>.

- 
1. i.e. a transition from a low intensity region to high intensity one.
  2. in our implementation we used training to choose the point that defines the boundary.

---

This chapter begins by discussing some aspects of the boundaries found by the different boundary detection methods. After that it presents different approaches for calculating the IMT.

---

*The boundary coordinates*

To calculate the thickness between two boundaries, the coordinates of the two boundaries' points must be known. Here, we will summarize the forms of boundary points that result from applying the different boundary detection methods discussed previously.

**The dynamic programming method**

In the dynamic programming approach, the coordinates of both the lumen-intima and the media-adventitia boundary pixels (resulting from the detection) have the following characteristics:

- They are integer-valued pixel coordinates.
- There is one row value for each column (remember that our images are matrices now).
- All the columns have a row value.

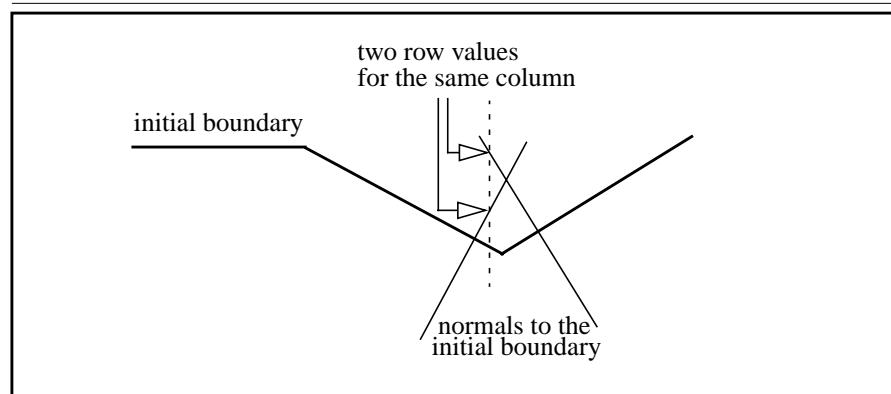
This is a direct result from the way the dynamic programming method detects the boundary, which is by finding one point for each and every column, that makes the total cost function minimum.

### The maximum gradient method

For the maximum gradient method, the situation is different. The coordinates are characterized by the following:

- They are real-valued (remember the sub-pixel resolution).
- It may not have a row value (for the detected boundary) at all columns, due to the procedure of cancelling boundary points having a maximum gradient less than 20% of the global maximum.
- It could have more than one row value for a specific column (See Figure 7.1).

**FIGURE 7.1. Having more than one row value for a specific column in the boundary coordinates of the maximum gradient method.**





### The model-based method

In the model-based approach, and in both types of modelling: the ‘sum-of-two-gaussians’ and the ‘piece-wise linear’ models, the coordinates of the detected boundary have the same characteristics as those resulting from dynamic programming method. This results from the fact that the modelling is done for all adjacent parallel profiles along the desired region of the image.

### The matched filter method

In the matched filter method the coordinates of the boundaries detected, share the same characteristics as both in the dynamic programming method and the model-based method. This results from the fact that the matched filter is convolved with each and every column of the image, and the boundary points are detected at a certain location within the matched filter template.

---

## *Methods for IMT calculation*

Here we present some methods for calculating the IMT.

### General method<sup>1</sup>

This method is not unique to a specific method of boundary detection and can be used as an alternative whenever it suits to do that.

- Fit a line  $L_1$  to the lumen-intima boundary, then find its slope  $S_1$ .
- Fit a line  $L_2$  to the media-adventitia boundary, then find its slope  $S_2$ .
- Find the line  $L_3$  which has the average slope  $S = \frac{(S_1 + S_2)}{2}$ .
- Find the line that is perpendicular to  $L_3$ , which naturally has the slope

$$S_p = -\frac{1}{S}.$$

---

1. This method was suggested by us and was used for calculating the IMT throughout all our implementations. It was found to give good results and to be accurate enough.

- For each media-adventitia edge, go perpendicularly up in a direction similar to that of  $L_3$  until we intersect  $L_1$ . Mark the intersection point  $P_{intersect}$ .
- Look for the closest lumen-intima edge to the intersection point and pair it with the corresponding media-adventitia point.
- Calculate the IMT distance as the distance between each paired points.
- The average IMT is computed as the mean distance between the edge point pairs.

### **Special procedure for the maximum gradient method<sup>1</sup>**

This method is only applicable to the maximum gradient method, due to the way it works and the form of coordinates resulting from detecting the boundaries. As has been stated previously, the media-adventitia boundary coordinates includes only acceptable points, with the unacceptable points excluded, meanwhile the lumen-intima includes acceptable points as well as interpolated points replacing the unacceptable points. The procedure is carried out as follows:

- For each acceptable media-adventitia edge, a line is drawn that is locally perpendicular to the media-adventitia edges and intersects the lumen-intima boundary.
- If the lumen-intima edge closest to this line was originally an acceptable edge (i.e. not an interpolated edge), it is paired with the media-adventitia edge and the IMT distance is calculated.
- If the nearest lumen-intima edge is an interpolated edge, the media-adventitia edge under consideration is assumed not to have a matching edge and is ignored. The process is then repeated for the next media-adventitia edge.
- The average IMT is computed as the mean distance between the acceptable edge point pairs.

---

1. this method is suggested in reference [7].

---

In this chapter we will present the results of calculating the IMT.

---

*Introduction*

The coordinates of the lumen-intima and the media-adventitia boundaries were obtained using all of the four methods of boundary detection: dynamic programming, maximum gradient, model-based, or matched filtering. With the coordinates available we were able to calculate the IMT.

The next important step was to define a criterion for evaluating our results. We used IMT values obtained by manual detection carried out by experts as our reference. This was the only solution to our problem since there exists no other *known* reference. We should point out that these manually-obtained values could be inaccurate but they are the only ones available.

One of the other testing criteria that were previously suggested<sup>1</sup> is to use a lucite step wedge phantom. Unfortunately, we did not have such equipment and thus were

---

1. see reference [7].

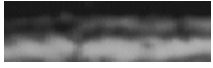
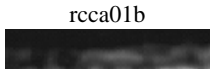
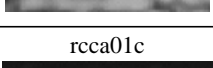
not able to judge our results using such reference. One can also argue that such a method for testing is not suitable in our case. The reason lies in the fact that our goal is to measure the IMT from noisy and unclear images which is not quite the same as measuring the thickness of a well defined and clear layer in the phantom.

---

### Results





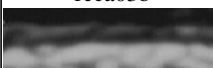
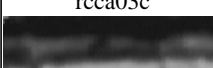
The following table summarizes the results of the IMT calculation for the four methods<sup>1</sup> (See Table 8.1).

**TABLE 8.1. Results of IMT calculation<sup>a</sup>**

Image	Dynamic Programming		Maximum Gradient		Model Based		Matched Filter		Manual Results	
		<i>average</i>		<i>average</i>		<i>average</i>		<i>average</i>		<i>average</i>
rcca01a 	0.814	0.82	0.842	0.85	0.753	0.66	0.694	0.68	0.74	0.75
rcca01b 	0.858		0.892		0.849		0.702		0.78	
rcca01c 	0.775		0.826		0.377		0.649		0.72	

<sup>a</sup> IMT values are expressed in mm




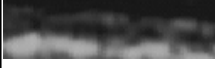

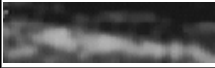

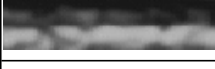

TABLE 8.1. Results of IMT calculation<sup>a</sup>

Image	Dynamic Programming		Maximum Gradient		Model Based		Matched Filter		Manual Results	
		<i>average</i>		<i>average</i>		<i>average</i>		<i>average</i>		<i>average</i>
rcca02a 	0.809	0.83	0.898	0.87	0.545	0.56	0.673	0.71	0.70	0.78
rcca02b 	0.804		0.826		0.446		0.709		0.81	
rcca02c 	0.876		0.874		0.691		0.732		0.84	
rcca03a 	0.752	0.78	0.811	0.80	0.748	0.60	0.663	0.66	0.72	0.70
rcca03b 	0.787		0.788		0.515		0.641		0.69	
rcca03c 	0.787		0.785		0.530		0.663		0.70	

<sup>a</sup> IMT values are expressed in mm



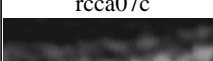

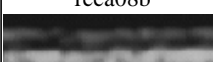
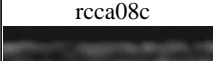
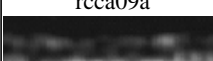
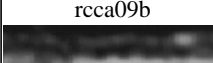
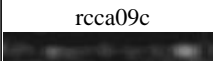
1. the results of the model-based method are obtained from using the 'sum-of-two-gaussians' model because it was easier to obtain a better fit with fixed initial conditions over all the images.

TABLE 8.1. Results of IMT calculation<sup>a</sup>

Image	Dynamic Programming	average	Maximum Gradient	average	Model Based	average	Matched Filter	average	Manual Results	average
		0.701	0.73	0.746	0.76	0.652	0.64	0.641	0.65	0.62
	0.726	0.774		0.655		0.675		0.66		
	0.750	0.768		0.618		0.631		0.65		
	0.904	0.93	0.903	0.91	1.058	1.00	0.729	0.70	0.89	0.91
	0.990		0.890		0.849		0.673		0.99	
	0.885		0.947		1.085		0.710		0.84	
	0.628	0.61	0.689	0.67	0.648	0.67	0.612	0.61	0.61	0.59
	0.588		0.657		0.483		0.599		0.56	
	0.624		0.674		0.876		0.609		0.59	

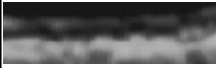


<sup>a</sup> IMT values are expressed in mm

TABLE 8.1. Results of IMT calculation<sup>a</sup>

Image	Dynamic Programming	average	Maximum Gradient	average	Model Based	average	Matched Filter	average	Manual Results	average
rcca07a 	0.771	0.78	0.890	0.88	0.732	0.65	0.710	0.68	0.75	0.76
rcca07b 	0.758		0.809		0.641		0.657		0.73	
rcca07c 	0.818		0.922		0.575		0.689		0.79	
rcca08a 	0.770	0.78	0.843	0.86	0.706	.69	0.707	0.70	0.73	0.78
rcca08b 	0.762		0.870		0.661		0.692		0.82	
rcca08c 	0.828		0.864		0.709		0.707		0.78	
rcca09a 	0.802	0.84	0.883	0.89	0.872	0.86	0.730	0.66	0.78	0.80
rcca09b 	0.805		0.878		0.796		0.633		0.79	
rcca09c 	0.907		0.908		0.922		0.626		0.82	

<sup>a</sup> IMT values are expressed in mm

TABLE 8.1. Results of IMT calculation<sup>a</sup>

Image	Dynamic Programming	average	Maximum Gradient	average	Model Based	average	Matched Filter	average	Manual Results	average
rcca10a 	0.785	0.77	0.802	0.81	0.637	0.73	0.647	0.67	0.75	0.73
rcca10b 	0.751		0.817		0.811		0.665		0.73	
rcca10c 	0.783		0.798		0.756		0.695		0.71	

<sup>a</sup> IMT values are expressed in mm



*Comparison*

The following table summarizes some important results that can be used to judge the performance of the different boundary detection methods. The results are based on the averaged values of every three images. (See Table 8.2).

**TABLE 8.2. Statistical parameters for describing the performance of the different boundary detection methods**

	Dynamic Programming	Maximum Gradient	Model-Based	Matched Filter
the average intima-media thickness: $\overline{IMT}$ (average of manual IMT results = 0.7430)	0.7866	0.8291	0.7065	0.6721
correlation coefficient with the manual IMT results: $\rho_{MR,X}$	0.9615	0.9402	0.6307	0.8520
average difference (manual-auto): $\bar{\epsilon}$	-0.0436	-0.0861	0.0365	0.0709
standard deviation of the difference: $\sigma_{\epsilon}$	0.0242	0.0316	0.1017	0.0634
coefficient of variation <sup>a</sup> : $CV = \frac{\sigma_{\epsilon} / \sqrt{2}}{\text{pooled mean}} \cdot 100\%$	2.2384	2.8450	9.9269	6.3359

a. the pooled mean is the overall mean of the manual and the automatic averaged IMT values. For more details see reference [1].

---

## Results and Comparisons

---

## *Discussion and Conclusions*

---

This chapter compares the different boundary detection methods, shows their pros and cons, and presents our conclusions from this study.

---

### *Comparison between the boundary detection methods*

This section describes and compares the performance of the four boundary detection methods. The comparison is based on different criteria for judging.

#### **Calculation complexity**

- ***Dynamic programming:*** this method is fast for that the calculations involved can be done in a very efficient way so as to save time.
- ***Maximum gradient:*** this method needs a moderate amount of calculations, thus the time needed to apply the algorithm is also moderate.
- ***Model-based:*** this method needs a great deal of calculations in order to find the models that best fit the intensity profiles. Therefore, it takes time to get the result of applying this algorithm on an image. In addition, we found it very hard to define some 'generally' good initial conditions for the optimization function

which was used to find the best model fitting a certain intensity profile. We have seen that the initial starting points for the process of optimization had a great influence on the final result, especially in the case of the ‘piece-wise linear’ model. This results in a difficulty which is trying to find a set of values that can be used in general for different sets of images.

- **Matched filter:** this method is fast. The calculations needed are quite few and thus it is time efficient. It may be that the hardest point is to find a good representative template to be used as the matched filter that can be used in a general sense.

### **Programming effort**

- **Dynamic programming:** this method needs a moderate amount of effort in order to write the software. It has many details for each step of the algorithm which might be complicated if the program is to be efficient.
- **Maximum gradient:** this method involves quite a large number of operations. Therefore, the programming needed to develop the algorithm takes lots of effort, especially that the algorithm has many details that should be considered.
- **Model-based:** this method needs quite an effort for developing the software especially if one wants to make the program robust and general. In the ‘piece-wise linear’ model it was difficult to find initial conditions that give appropriate fit for an acceptable number of images, on the other hand this task was easier in the ‘sum-of-two-gaussians’ model.
- **Matched filter:** this algorithm requires a moderate amount of effort in the programming process. The idea is basic, so there isn’t a lot of complications.

### **Required training**

- **Dynamic programming:** this method needs lots of training on a very large set of images. The reason lies in the fact that it searches for an approximately optimum weights for the different contributors to the cost function.
- **Maximum gradient:** no training is required in this method.
- **Model-based:** this method needs lots of training in order to get an approximately optimum values for the parameters used to define the criteria for detecting the boundary points<sup>1</sup>.

---

1. for a clearer description of these parameters refer to Figure 5.7 and Figure 5.8.

- **Matched filter:** this method needs training too. The first reason is that it is important to find the histograms for the order and rank of the detected peaks resulting from the convolution process. This is needed to identify the most probable true boundary point. The second reason for training is to get an approximately optimum template that best represents the intensity transition at the boundary points.

### Extent of manual intervention

- **Dynamic programming:** This method needs manual interventions only in rare cases. The fact that it gives weights to many different factors characterizing a real boundary makes the results very reasonable. In any case, the program we used gives the possibility for manual modification of the detected boundary whenever needed.
- **Maximum gradient:** this method often gives unreasonable results for certain boundary points. The reason is that it does not give any importance to the continuity of the detected boundary and thus it results in huge jumps between the boundary points particularly in the regions of the image where echo drop-out is strong. It should be mentioned that our developed software has a mechanism enabling the user to modify the erroneous points manually<sup>1</sup>.
- **Model-based:** this method may give erroneous boundary points if the intensity profiles of the analysed image were very different than the proposed model. In this case, a mechanism for fixing such errors must be used to eliminate the obvious unreasonable points.
- **Matched filter:** this method may give erroneous boundary points if the analysed image suffers from strong echo drop-out so that the intensity profiles are very different from the template used as the matched filter. Here also, there is a need for a certain mechanism for fixing errors.

### Variability between users & degree of automation

- **Dynamic programming:** this method is nearly a 100% automatic algorithm. The only manual intervention, which may lead to variability between users, can happen if the operator thinks that certain detected boundary points are not acceptable. The occurrence of wrong detection is so rare that it could be considered insignificant. Nevertheless, this may give rise to some variability between operators if their standards for a correct point are quite different.

---

1. see chapter 10 for details on the implementation.

- **Maximum gradient:** this method involves manual procedures in a wider sense. First, the user has to define the initial approximate boundary around which the algorithm searches for the true boundary. Second, the absence of a continuity factor in the algorithm results in wrong detected points quite often. The previous two points lead to the fact that this algorithm can be considered a semi-automatic method (not a fully automated one). We can say from the discussion above, that there is more variability between users when this method is used.
- **Model-based:** this method is nearly fully automated. Nevertheless, there still exists some cases where obvious errors have to be compensated for by means of manual modifications. This, of course, may result in variability between users in some limited cases.
- **Matched filter:** here, the same discussion as in the model-based method above applies.

### **Similarity to a true boundary**

- **Dynamic programming:** this method gives very reasonable results nearly all the time. The detected boundary always has the characteristics of a true boundary. This results from the way the method works which is to take account for all important features of a true edge. In summary, this method gives results that are very similar to a true boundary.
- **Maximum gradient:** this algorithm defines the peak as the point of maximum intensity which is not true all the time. This assumption leads to obvious errors when the ultrasound images are of bad quality. The algorithm also ignores the fact that a true boundary does not have jumps between neighbouring edges. Therefore, it is quite often that the method gives a result that is not similar enough to what a true boundary looks like. Thus, it is quite often needed to modify the output manually so as to eliminate bad points.
- **Model-based:** the results are usually good. Nevertheless, it may happen that the results are poor if the analysed images cannot be modelled good enough with the suggested model. In such cases, some points can also be obviously wrong since they violate the continuity of the boundary.
- **Matched filter:** here, the same discussion as in the model-based method above applies.

### Closeness to manual results

This is probably the most important criteria of all. Therefore, this aspect should be given a higher degree of importance or in other words should be given higher weight<sup>1</sup>. The closeness to the manual results is best viewed through the correlation coefficient.

- **Dynamic programming:** the IMT results obtained from this method<sup>2</sup> had the highest correlation coefficient with the manual IMT results  $\rho \approx 0.96$ .
- **Maximum gradient:** the correlation was the second highest  $\rho \approx 0.94$ .
- **Model-based:** the correlation was the least  $\rho \approx 0.63$ .
- **Matched filter:** the correlation was the third highest  $\rho \approx 0.85$ .

- 
1. For a close look on the results obtained from each method, and for comparing them to the reference manual results, the reader is referred to chapter 8.
  2. It can also be noted that the IMT results obtained by the dynamic programming method, in our case, were systematically higher than the manual results. This can be explained by the fact that the manually detected points in the lumen-intima boundary is sometimes chosen closer to the maximum intensity point than the maximum gradient point. This is simply because sometimes the maximum gradient point is one of low intensity that the operator can not see.

---

### *Conclusions*

In the following lines, we try to summarize our experience regarding the different boundary detection methods.

- ***Dynamic programming:*** this method seems to be a very good one. It is relatively fast, nearly fully automatic, gives reasonable results and it is the only method that accounts for the continuity of the detected boundaries<sup>1</sup>.
- ***Maximum gradient:*** no training is required for this method. Unacceptable boundaries are observed with images with high echo drop-out. It requires excessive manual data entry and intervention to obtain acceptable results.
- ***Model-based:*** the training takes a lot of time. The fitting is highly dependent on initial conditions. Very good fits were obtained only in the 'sum-of-two-gaussians' model. Sometimes the results were irrelevant in some bad images. We believe that it needs further modifications.
- ***Matched filter:*** training time is acceptable. Resulting boundaries are also acceptable. Promising and further investigation and refining is suggested.

---

1. it should be noted that our using of an already developed software to test the method, made it hard for us to fairly judge the difficulties or complexities encountered in producing such a program. Our implementation of the dynamic programming algorithm was not fully completed so as to be able to use it for IMT calculation simply because this was already developed in the department before.



---

This chapter focuses on our implementations. It presents a general description of the software, some detailed specifications and a list of the file names followed by a brief description.

---

*Software description*

**The dynamic programming method**

We implemented the software needed for illustrating the use of dynamic programming for boundary detection. This software features:

- Ability to change the width of intensity operator.
- Ability to control the weight of the continuity cost.
- Ability to control the weight of the gradient.
- Ability to control the weight of the average intensity.
- Ability to choose linear, quadratic or cubic continuity cost dependence.
- Ability to choose a simple, complex gradient operators, or use MatLab<sup>®</sup>'s gradient function.

- The ability to choose different images for experimenting.
- An easy to use graphical user interface (GUI).

### **The maximum gradient method**

We implemented the software needed for illustrating the use of the maximum gradient method for boundary detection and IMT calculation. The program features:

- Ability to select the intima, media, or adventitia boundaries as the one to be detected.
- Ability to specify that the required boundary has either a bright area above it and a dark area below it or vice versa.
- Ability for the user to pick initial points on the required boundary by clicking on them using a mouse.
- Ability to choose which thickness<sup>1</sup> is to be calculated.
- Ability to draw up to three boundaries simultaneously on the original image.
- Ability to browse throughout the user's directories for loading different images.
- Ability to manually modify the automatically detected boundary by using simple mouse clicks.
- An easy to use GUI.

### **The model-based method**

We implemented the software needed to illustrate the model-based method for boundary detection and IMT calculation. The software features:

- Having a gaussian modelling demo.
- Having a linear modelling demo.
- Ability to perform gaussian training on arbitrary images.
- Ability to perform linear training on arbitrary images.
- Ability to perform fitting of a gaussian model to the profiles of an image.
- Ability to perform fitting of a linear model to the profiles of an image.
- Ability to detect the boundaries in the modelled images.

---

1. meaning either the IMT or the lumen diameter.

---

## Implementation specifications

---

- Ability to calculate the IMT.
- Ability to use parameters obtained from training or specified by the user as a criterion for identifying the boundary.
- An easy to use GUI.

### **The matched filter method**

We implemented the software needed to illustrate the matched filtering method for boundary detection and IMT calculation. The software features the:

- Possibility for training in order to obtain the histogram of the rank of the peak that is most likely to be the true lumen-intima boundary peak.
- Possibility for training in order to obtain the histogram of the rank of the peak that is most likely to be the true media-adventitia boundary peak.
- Possibility for training in order to obtain the histogram of the order of the peak that is most likely to be the true lumen- intima boundary peak.
- Possibility for training in order to obtain the histogram of the order of the peak that is most likely to be the true media-adventitia boundary peak.
- Ability to apply the matched filter to the profiles of an image.
- Ability to detect the boundary based on a maximum likelihood criterion.
- Ability to calculate the IMT.
- An easy to use GUI.

---

## *Implementation specifications*

Below are some specifications, about our different implementations.

### **Implementation environment**

Our software implementations were done using MatLab<sup>®</sup>, version 4.2c under Unix, using a Sun work station.

### Training<sup>1</sup>

Since we had a limited number of ultrasound images<sup>2</sup> over which we had to perform both training and test detection, we had to use a special strategy to do that in an efficient way. For each image, we performed a separate training procedure that uses all the other images available in our image set. This method is called the '*leave one out*' or the '*jack-knife*' method. This procedure is popular in cases like ours where tests are to be carried out using small sets of images.

### The dynamic programming method

Results of the IMT calculations<sup>3</sup> were obtained using a software developed earlier by Quan Liang from the applied electronics department. The following describe some relevant specifications of the software.

- Calculating the gradient matrix, was done by fitting a 3<sup>rd</sup> order local (5×5) fitting surface and taking the slope of the centre point of this surface in a direction perpendicular to the boundary. The gradient values obtained were truncated. The truncation value was chosen to avoid detecting a boundary with high gradient but within a region of high intensity that does not reflect a boundary transition region.
- For calculating the intensity matrix, the mean of 4 or 8 pixels below the desired pixel was found when detecting the lumen-intima or the media-adventitia boundary, respectively.
- In order to consider the effect of a curved vessel, the discontinuity cost was evaluated with respect to reference axis having the same curvature of the vessel. This means that horizontally neighbouring pixels have a zero discontinuity cost if both are at the same distance from the reference. So, two pixels in the same row could have a non-zero discontinuity cost, and two pixels in different rows could have zero cost (See Figure 8.1).

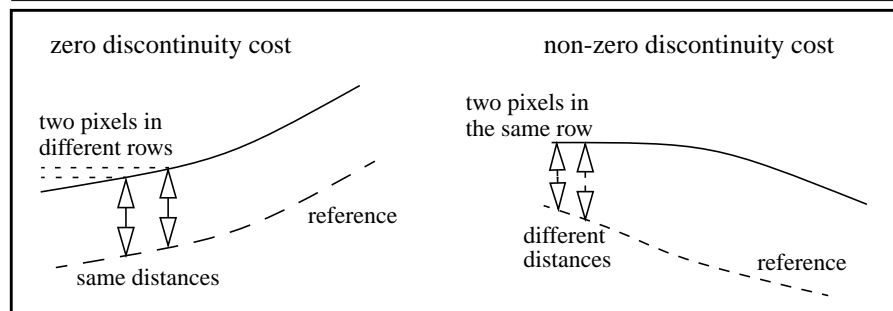
---

1. this was used in the model-based and the matched filter methods only. The program we used for the dynamic programming method had already been trained previously.

2. we used a set of 30 images.

3. see chapter 8.

**FIGURE 8.1. More on discontinuity cost**



### The maximum gradient method<sup>1</sup>

For the size of the searching region on the normal lines<sup>2</sup>, we used 13 pixels where half of them were above the initial boundary while the other half were below it.

After the detection of the points with maximum gradient, we found the global maximum, then we discarded any detected point that had a gradient value below 20% of the global maximum value.

### The model-based method

For the ‘sum-of-two-gaussians’ model, we used the following values as the parameter defining the lumen-intima boundary  $\alpha$ . Also the values used as the parameter defining the media-adventitia boundary  $\beta$  are also listed<sup>3</sup> (See Table 10.1).

- 
1. the chosen numbers were suggested in reference [7].
  2. by normal we mean the normal on the initial boundary defined by the manual choosing of the approximate edge point locations.
  3. for a clear definition of these parameters refer to Figure 5.7.

**TABLE 10.1. List of parameters defining the intima-media and the media-adventitia boundaries in the ‘sum-of-two-gaussians’ model.**

Image	$\alpha$	$\beta$
rcca01a	0.8366	0.3469
rcca01b	0.8361	0.3443
rcca01c	0.8273	0.3504
rcca02a	0.8410	0.3465
rcca02b	0.8347	0.3500
rcca02c	0.8374	0.3470
rcca03a	0.8319	0.3469
rcca03b	0.8264	0.3495
rcca03c	0.8536	0.3513
rcca04a	0.8424	0.3444
rcca04b	0.8330	0.3449
rcca04c	0.8393	0.3475
rcca05a	0.8374	0.3444
rcca05b	0.8382	0.3460
rcca05c	0.8279	0.3460
rcca06a	0.8273	0.3452
rcca06b	0.8284	0.3467
rcca06c	0.8387	0.3482
rcca07a	0.8347	0.3453
rcca07b	0.8378	0.3496
rcca07c	0.8387	0.3446
rcca08a	0.8339	0.3470
rcca08b	0.8300	0.3492
rcca08c	0.8333	0.3479
rcca09a	0.8385	0.3445
rcca09b	0.8371	0.3482
rcca09c	0.8350	0.3460

TABLE 10.1. List of parameters defining the intima-media and the media-adventitia boundaries in the ‘sum-of-two-gaussians’ model.

Image	$\alpha$	$\beta$
rcca10a	0.8281	0.3488
rcca10b	0.8290	0.3453
rcca10c	0.8332	0.3445
<b>Mean</b>	<b>0.8349</b>	<b>0.3469</b>
<b>Variance</b>	<b>3.2579e-05</b>	<b>4.1127e-06</b>

### The matched filter method

The matched filter for lumen-intima was part of a gaussian shape. The following values were used:

[15 16 20 29 43 58 65]

The matched filter for media-adventitia was also part of a gaussian shape. The following values were used:

[15 16 17 20 25 34 47 66 89 115 138 154 160]

---

## *Description of the MatLab® files*

Below we list the names of the MatLab files written for implementing the software for the dynamic programming, the maximum gradient, the model-based, and the matched filter methods. A brief descriptions of each file is provided.

### The dynamic programming method

#### ***DPstart.m***

Start-up point, clears all variables, closes all figures, builds up the graphical user interface and associates the controls with corresponding callback functions.

***DPall.m***

A function called when the user wants to apply the dynamic programming method to a specified image with certain parameters. This function results in displaying the detected boundary

***DPAbout.m***

Loads an information dialogue box, which can be closed by pressing the ‘ok’ button.

***NewImage.m***

Performs necessary tasks associated with changing the image on which the dynamic programming method is performed.

***rb1prsd.m, rb2prsd.m, rb3prsd.m***

These three functions take care of the operation of the radio buttons. i.e. to have only one active radio button at a time.

**The maximum gradient method**

***IMTstart.m***

Start-up point, clears all variables, closes all figures and calls the function “uiscreen” which loads the dialogue box for the implementation of the maximum gradient method program.

***RealMaxGrd.m***

Fits a second order polynomial to three points and finds the value and location of the maximum.

***browseimg.m***

This is a callback function executed when the user wants to browse throughout his directories for selecting an image to work on it.

***calcthick.m***

This is a callback function evaluated when the user wants to calculate the thickness between two boundaries he had chosen.

***cnct2pts.m***

This function connects two 2D points by a straight line.

***cnctinitpts.m***

This function connects a set of separated points piece wise linearly.



***drawthree.m***

This is a callback function invoked when we want to draw the boundaries. bnd1.mat and bnd2.mat and bnd3.mat should be saved on the current directory.

***fixbound.m***

This function displays the image with the boundary chosen for modification, and sets the figure ready to be picked at the points we want to modify.

***g9max2neibs.m***

This functions returns the maximum value of 9 values, its location, and the neighbouring 2 values to the maximum, for a set of L points.

***getRealXYCtr.m***

This function finds the real (x,y) coordinates for L points known by their location on the normal to a known line.

***grad5neibs.m***

This function finds the gradient at a point considering 2 points on each side of it. For L sets of 13 points so the gradient is only calculated for 9 out of 13 points.

***interpolate.m***

This function inspects a set of points defined by their (x,y) coordinates and having each a value for a property they share. “interpolate” selects those points which have a property value above 20% of the maximum property value (for all points), it also interpolates the values for the unacceptable ones (those with values below 20%).

***intersect.m***

Finds the common content of two vectors.

***modify.m***

This is a callback function executed when the user wants to start modifying a boundary manually.

***neg20.m***

Inspects a set of points defined by their (x,y) coordinates and having each a value for a property they share. “neg20” selects those points which have a property value above 20% of the maximum property value (for all points) and discards the remaining ones.

***nrm13pts.m***

Finds the 13 points along the normal on each pixel along the initial contour of L points.

***pointpicked.m***

This is a callback function evaluated when the user chooses a point as one of the initial points selected from the image.

***ptstomodify.m***

This is a callback function evaluated when the user clicks a point on the image for modifying the position of the boundary at that point.

***startpicking.m***

This is a callback function evaluated when the user wants to start picking on the image for selecting the initial points on the boundary.

***stopmodify.m***

This is a callback function executed when the user wants to stop the process of modifying a boundary manually, and to save his modifications.

***stoppicking.m***

This is a callback function evaluated when the user finishes entering the initial points and presses the stop picking push button.

***thickness.m***

This function calculates the thickness between two boundaries.

***uiscreen.m***

This function loads the dialogue box for the implementation of the maximum gradient method program.

**The model-based approach**

***startup.m***

This function is called when MatLab is started from the directory that contains this file. This function adds certain directories to the matlab path.

***G\_alpha.mat***

This is a matlab data file containing the parameter of the criterion defining the lumen-intima boundary for the 'sum-of-two-gaussians' model.

***G\_beta.mat***

This is a matlab data file containing the parameter of the criterion defining the media-adventitia boundary for the ‘sum-of-two-gaussians’ model.

***L\_alpha.mat***

This is a matlab data file containing the parameter of the criterion defining the lumen-intima boundary for the ‘piece-wise linear’ model.

***L\_alpha.mat***

This is a matlab data file containing the parameter of the criterion defining the media-adventitia boundary for the ‘piece-wise linear’ model.

***GMOV.mat***

This a matlab data file, containing necessary information for the movie function which is used within the gaussian demo.

***LoadG\_alpha.m***

This function is for loading the parameter of the criterion defining the lumen-intima boundary for the gaussian model.

***LoadG\_beta.m***

This function is for loading the parameter of the criterion defining the media-adventitia boundary for the gaussian model.

***gDemo.m***

Gaussian modelling demo. Uses saved frames and the movie command to present an animation.

***gblldemo.m***

For building the frames of the gaussian modelling demo.

***gfit.m***

Fitting the gaussian model to a set of data points.

***gfun.m***

The function to be minimized when fitting the gaussian model. Needed for the *constr* function in matlab.

***gget\_gIMG\_ubnd\_lbnd.m***

Given an image and the parameters of the criteria defining the boundaries, this function returns a gaussian-modelled image and both of the detected boundaries.

***ggetalpha.m***

Given the true boundary point and the gaussian model parameters, this function calculates the parameter of the criterion defining the lumen-intima boundary.

***ggetbeta.m***

Given the true boundary point and the gaussian model parameters, this function calculates the parameter of the criterion defining the media-adventitia boundary.

***grun.m***

Fits a gaussian model to all profiles of an image, and uses the parameters of the criteria defining the boundaries to calculate the IMT.

***gshow.m***

This function shows the original image, the gaussian-modelled image, and the detected boundaries.

***gtrain.m***

Trains the parameters of the criteria defining the boundaries for the gaussian model. This function requires a set of images with known true boundaries.

***togtrain.m***

Called when the user presses the training button and initiates the gaussian training.

***LMOV.mat***

This a matlab data file, containing necessary information for the movie function which is used within the linear demo.

***LoadL\_alpha.m***

This function is for loading the parameter of the criterion defining the lumen-intima boundary for the linear model.

***LoadL\_beta.m***

This function is for loading the parameter of the criterion defining the media-adventitia boundary for the linear model.

***IDemo.m***

Linear modelling demo. Uses saved frames and the movie command to present an animation.

***blddemo.m***

For building the frames of the linear modelling demo.

***lgfit.m***

Fitting the linear model to a set of data points.

***lfun.m***

The function to be minimized when fitting the linear model. Needed for the *constr* function in matlab.

***lget\_IIMG\_ubnd\_lbnd.m***

Given an image and the parameters of the criteria defining the boundaries, this function returns a linear-modelled image and both of the detected boundaries.

***lgetalpha.m***

Given the true boundary point and the linear model parameters, this function calculates the parameter of the criterion defining the lumen-intima boundary.

***lgetbeta.m***

Given the true boundary point and the linear model parameters, this function calculates the parameter of the criterion defining the media-adventitia boundary.

***lrn.m***

Fits a linear model to all profiles of an image, and uses the parameters of the criteria defining the boundaries to calculate the IMT.

***lshow.m***

This function shows the original image, the linear-modelled image, and the detected boundaries.

***ltrain.m***

Trains the parameters of the criteria defining the boundaries for the linear model. This function requires a set of images with known true boundaries.

***toltrain.m***

Called when the user presses the training button and initiates the linear training.

***MDLstart.m***

Starting point of the model-based boundary detection and IMT calculation program. Builds the graphical user interface and associates controls with callback functions.

***PressFitWBnd.m, PressFitWIMT.m***

Callback functions for radio buttons. The radio buttons are for specifying whether only the boundaries should be detected or also the IMT should be calculated.

***calcIMT.m***

Given the coordinates of two boundaries, this function calculates the thickness between them.

**The matched filter method**

***MATCHstart.m***

Starting point of the matched filter boundary detection and IMT calculation program. Builds the graphical user interface and associates controls with callback functions

***PressFindBnd.m, PressFindWIMT.m***

Callback functions for radio buttons. The radio buttons are for specifying whether only the boundaries should be detected or also the IMT should be calculated.

***PxlsBeforePkL.mat***

A matlab data file containing an integer which specifies the interna boundary point on the matched filter template.

***PxlsBeforePkU.mat***

A matlab data file containing an integer which specifies the media boundary point on the matched filter template.

***SMOOTHER.mat***

A matlab data file containing a vector of numbers that specify the shape of the matched filter.

***calcIMT.m***

Given the coordinates of two boundaries, this function calculates the thickness between them.

***getIndexOfClosest.m***

A function for finding the index of the peak which is closest to the true boundary given with the training image.

***getOrders.m***

Used for calculating the orders (locations) of the peaks after convolving the matched filter with the image profile.

***getPeaks.m***

Given a set of data points, this function returns another set containing the indices of the points having a peak value.

***getRanks.m***

Used for calculating the ranks (in value) of the peaks after convolving the matched filter with the image profile.

***mfilterL.mat***

A matlab data file containing the definition of the matched filter used for detecting the media boundary.

***mfilterU.mat***

A matlab data file containing the definition of the matched filter used for detecting the intima boundary.

***mrinL.m***

Starting point for detecting the media boundary.

***mrinU.m***

Starting point for detecting the intima boundary.

***mtrainL.m***

Training for the media boundary (building the media histograms).

***mtrainU.m***

Training for the intima boundary (building the intima histograms).

***orderhistL.mat***

A matlab data file containing the histogram values for the order of the peaks for the media boundary.

***orderhistU.mat***

A matlab data file containing the histogram values for the order of the peaks for the intima boundary.

***rankhistL.mat***

A matlab data file containing the histogram values for the rank of the peaks for the media boundary.

***rankhistU.mat***

A matlab data file containing the histogram values for the order of the peaks for the intima boundary.

***run.m***

Initiates the process of detecting a boundary using the matched filter method.

***setParam.m***

A matched filter designer function, in which the matched filter and the point on the filter where the boundary is detected are both specified for the intima and the media boundaries.

***startup.m***

Called when matlab is started from the directory containing this file. Adds some directories to matlab's path.

***toLtrain.m***

Starting point for media boundary training.

***toUtrain.m***

Starting point for the intima boundary training.

***vieworderhistL.m***

Loads the order of peaks histogram data and displays it. This function is for the media boundary.

***vieworderhistU.m***

Loads the order of peaks histogram data and displays it. This function is for the intima boundary.

***viewrankhistL.m***

Loads the rank of peaks histogram data and displays it, This function is for the media boundary.

***viewrankhistU.m***

Loads the rank of peaks histogram data and displays it, This function is for the intima boundary.

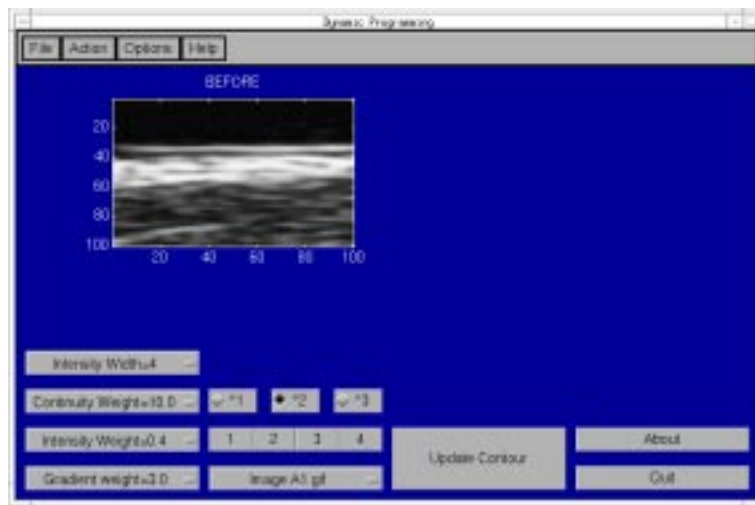


---

This chapter takes the reader in a tour through the different stages of running our implementations by presenting captured illustrations.

*A look at the dynamic programming method implementation*

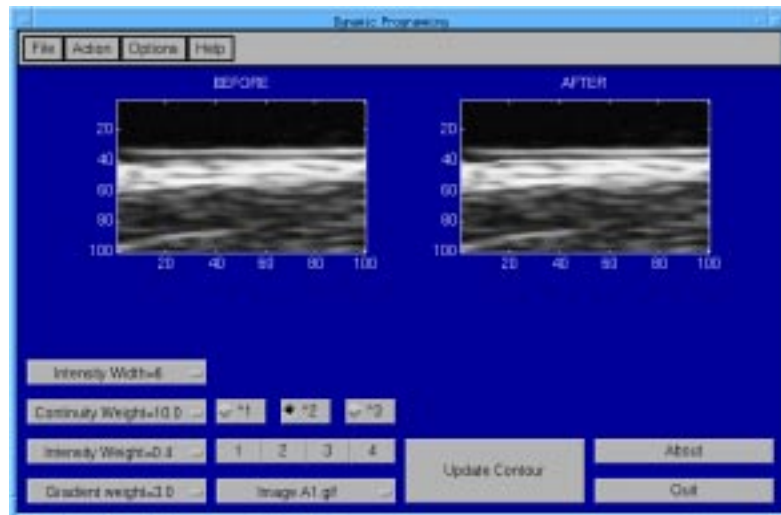
**FIGURE 11.1. The graphical user interface**



**FIGURE 11.2. Detecting a boundary with linear continuity dependence**



**FIGURE 11.3. Detecting the boundary with quadratic continuity dependence**



**FIGURE 11.4. Detecting the boundary with cubic continuity dependence**

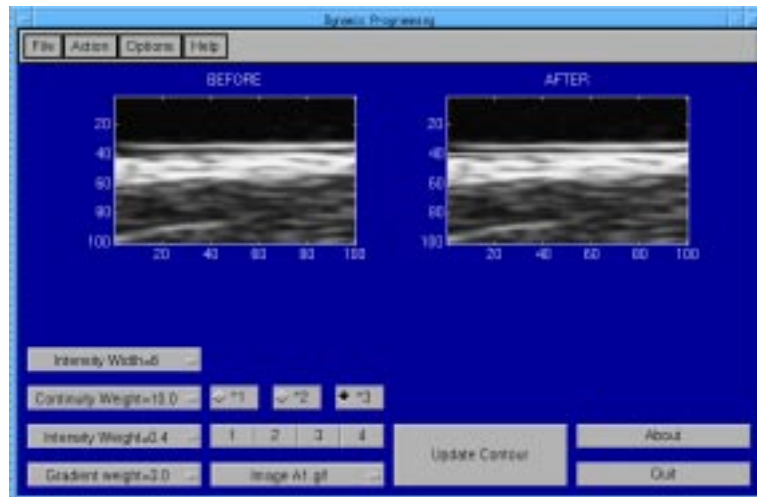
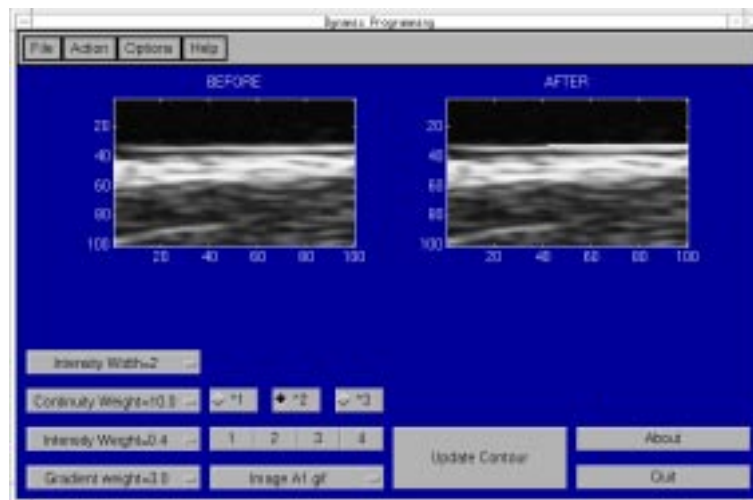


FIGURE 11.5. Result of clicking the 'About' button



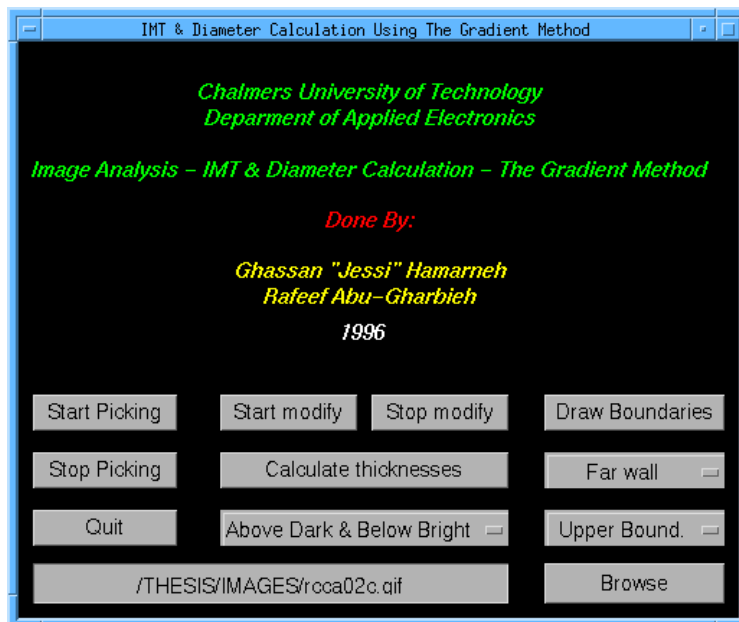
FIGURE 11.6. Decreasing the intensity width and detecting the intima boundary



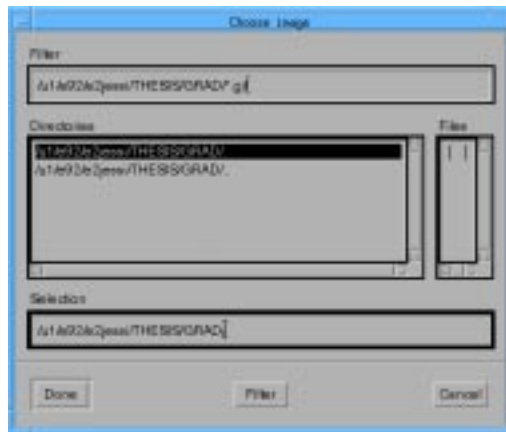
*A look at the maximum gradient method implementation*

**FIGURE 11.7.** The graphical user interface

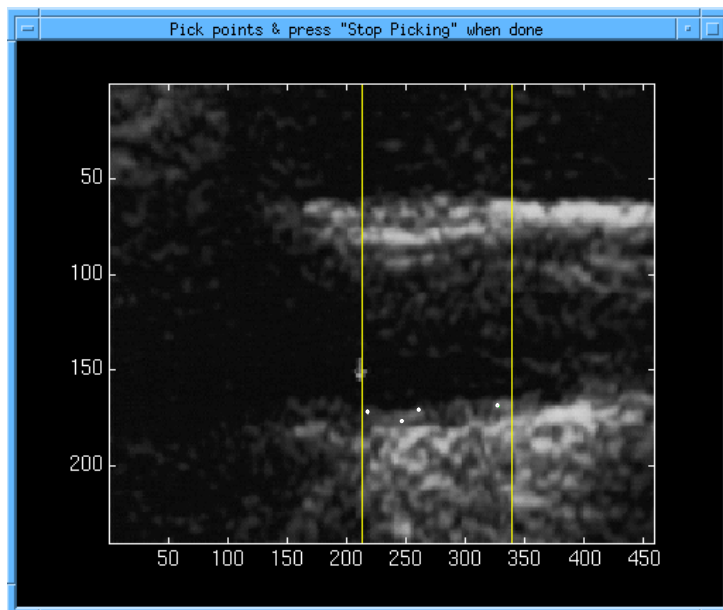
---



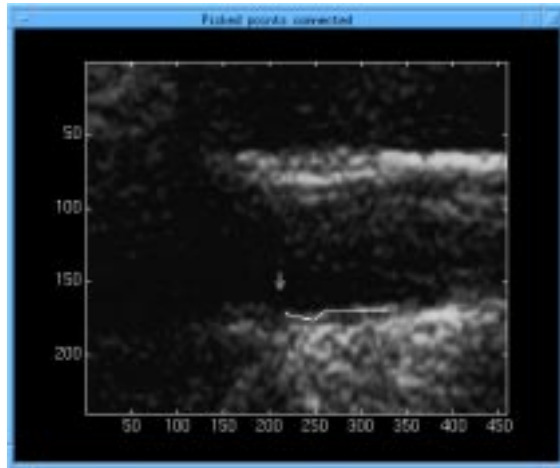
**FIGURE 11.8. Browsing to choose an image and apply the gradient method**



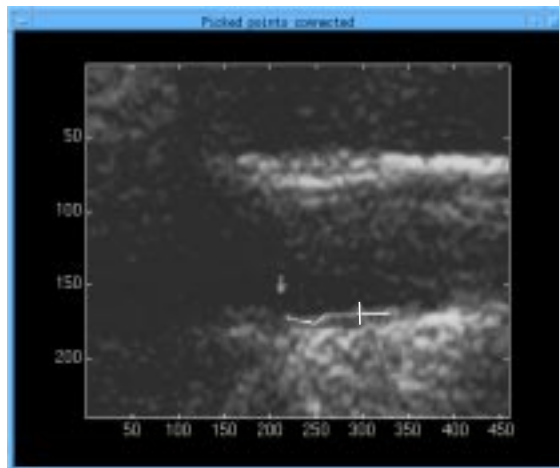
**FIGURE 11.9. Picking points with the mouse**



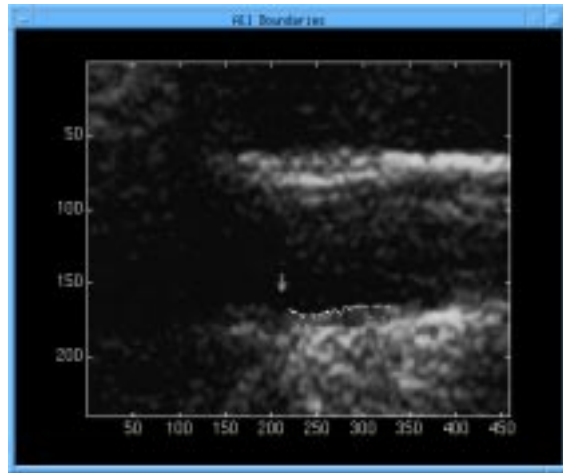
**FIGURE 11.10.** Constructing the initial boundary by connecting the points with straight lines



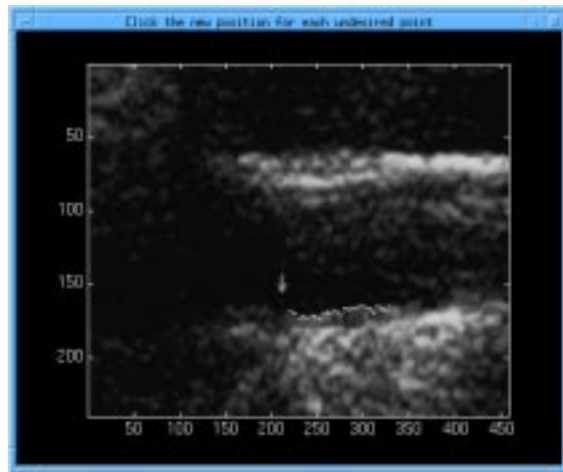
**FIGURE 11.11.** Scanning along the normal to each point



**FIGURE 11.12. Detecting the boundary**

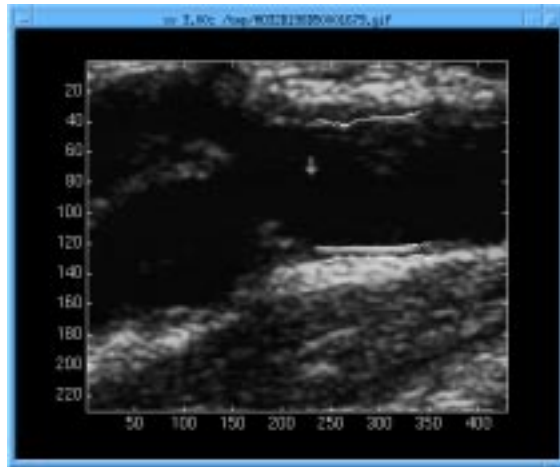


**FIGURE 11.13. Modifying the detected boundary**

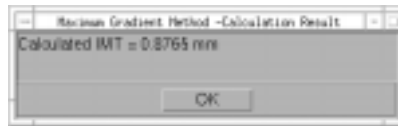




**FIGURE 11.14. Multiple boundaries detected**



**FIGURE 11.15. Result of calculating the intima-media thickness**



---

*A look at the model-based method implementation*

---

**FIGURE 11.16.** The graphical user interface

---



**FIGURE 11.17.** Result of calculating the intima-media thickness

---

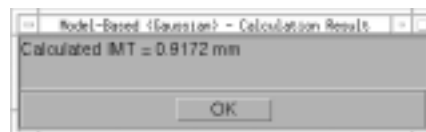


FIGURE 11.18. Modelling an intensity profile by the linear model

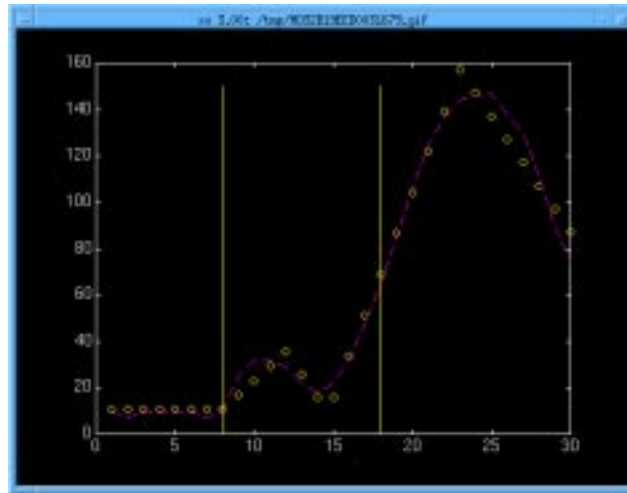


FIGURE 11.19. In the process of modelling an image by the linear model

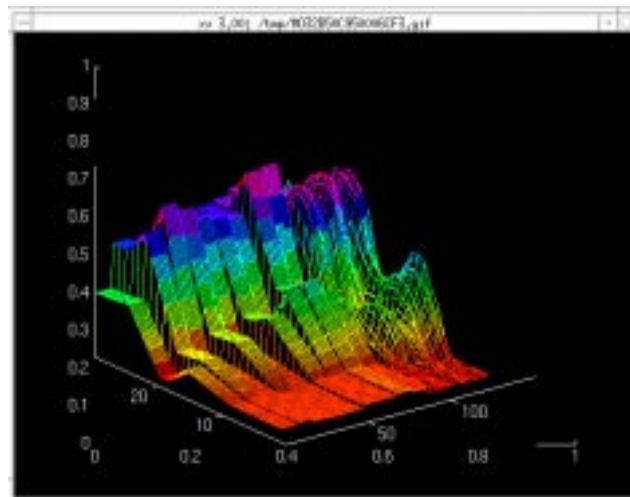


FIGURE 11.20. Modelling an intensity profile by the gaussian model

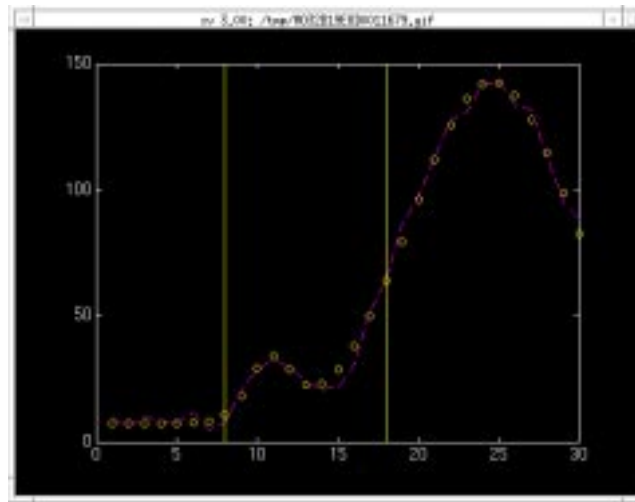


FIGURE 11.21. In the process of modelling an image by the gaussian model

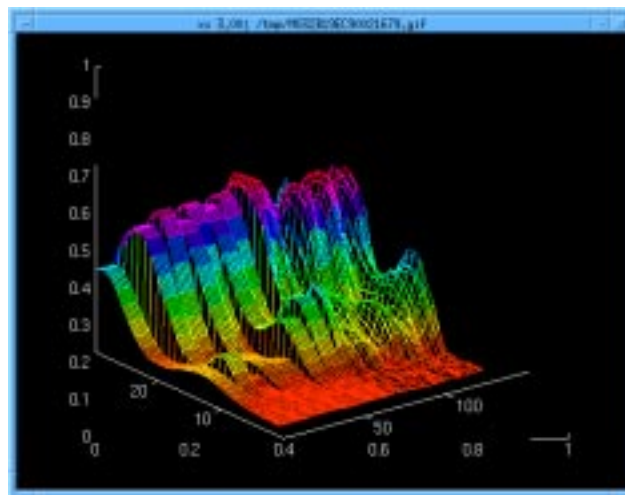


FIGURE 11.22. An intensity image visualized in 3D

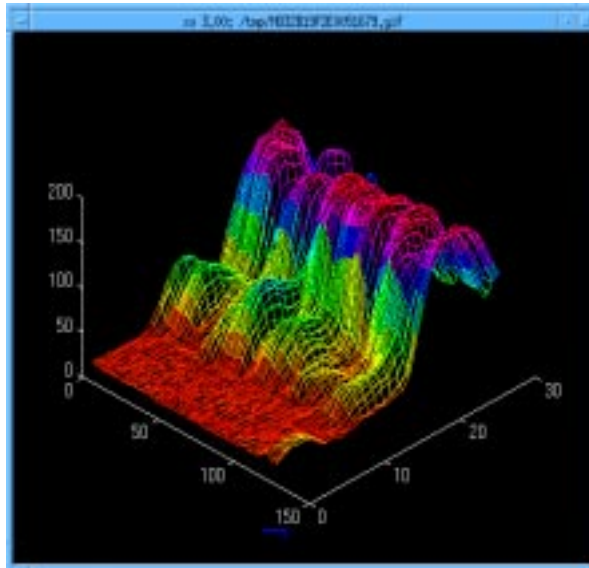


FIGURE 11.23. The gaussian model of the intensity image

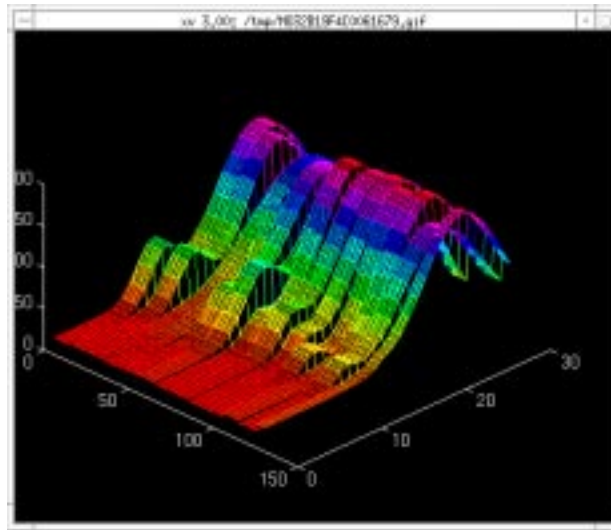
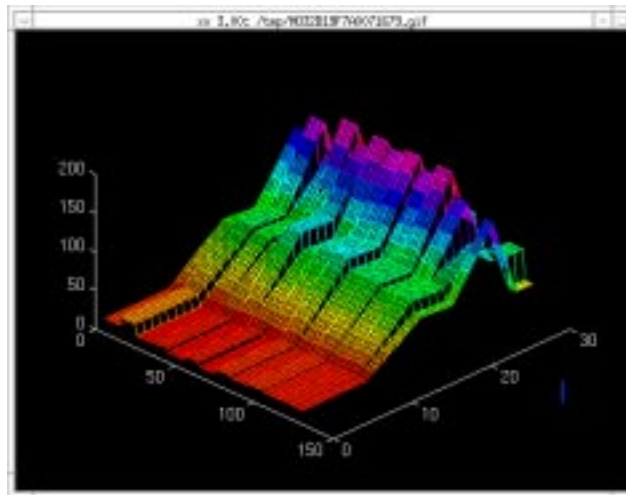
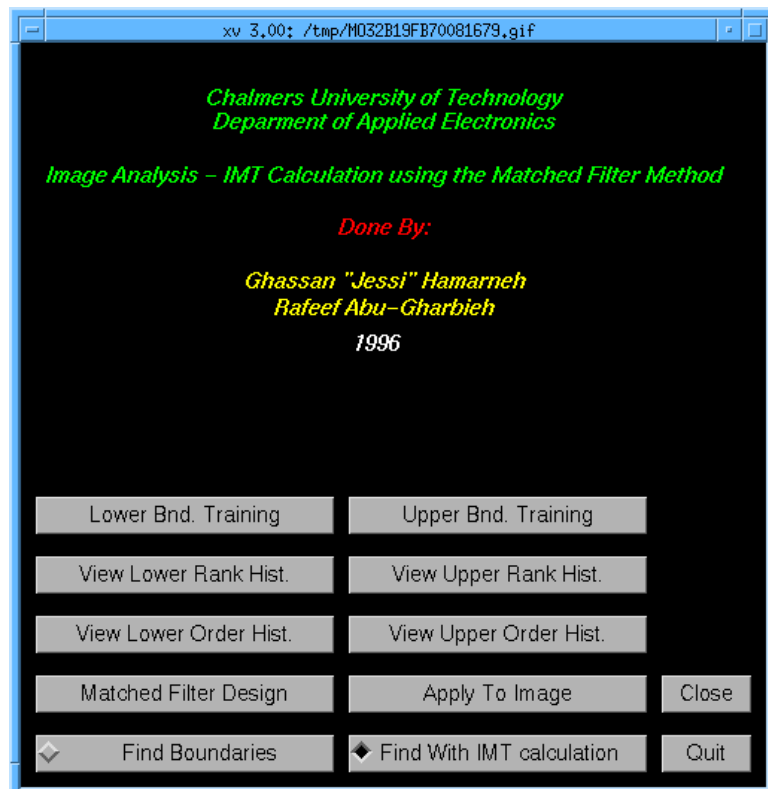


FIGURE 11.24. The linear model of the intensity image

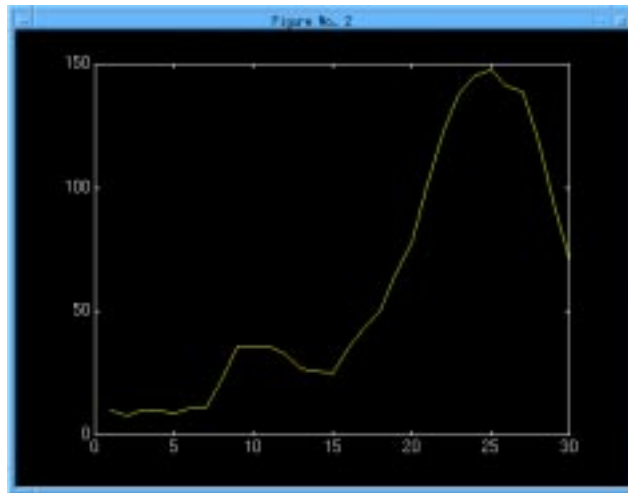


*A look at the matched filter method implementation*

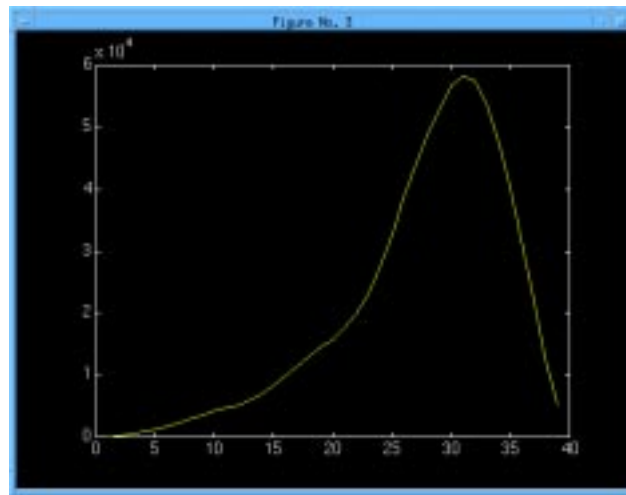
**FIGURE 11.25.** The graphical user interface



**FIGURE 11.26. An intensity profile**

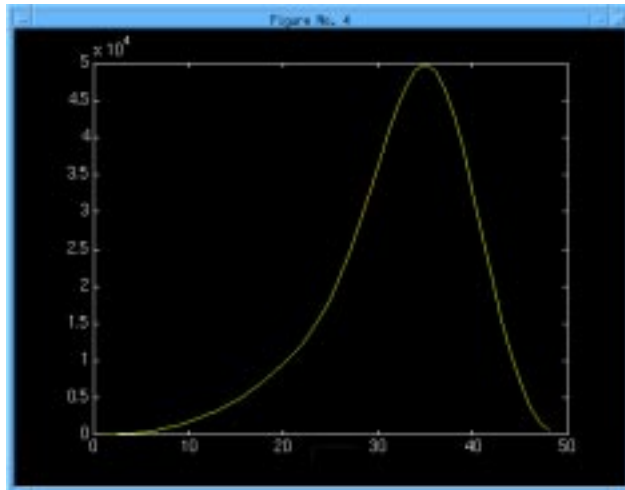


**FIGURE 11.27. Convolution of the intensity profile with a matched filter**





**FIGURE 11.28. Smoothing the result of the convolution**



**FIGURE 11.29. One of the histograms used for detecting the peak which is most likely to correspond to the true boundary**

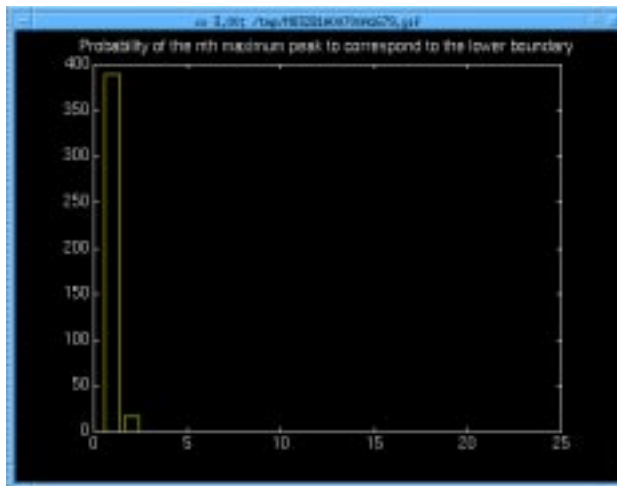


FIGURE 11.30. Result of detecting the intima and media boundaries

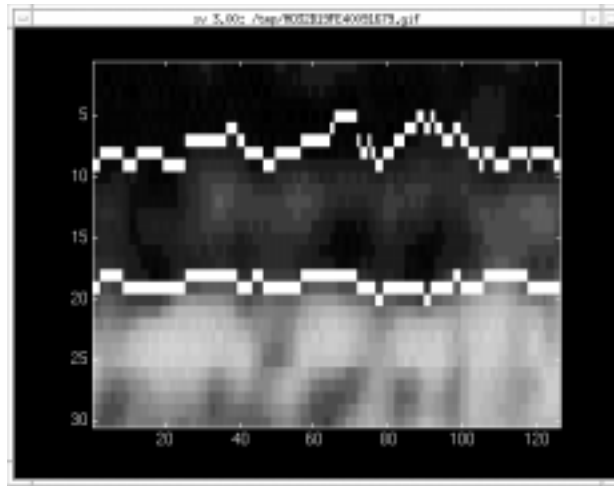
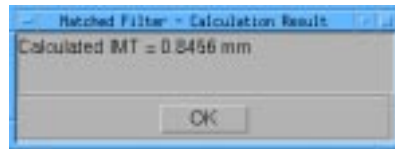


FIGURE 11.31. Result of calculating the intima-media thickness



---

---

## *References and Related Material*

---

### *References*

- [1] On the acquisition, analysis and display of echocardiographic image sequences. Tomas Gustavsson. School of Electrical & Computer Engineering, Chalmers University of technology, Göteborg, Sweden. Technical Report No. 209. 1991.
- [2] Cardiac Imaging and Image Processing. Steve M. Collins & David Skorton. McGraw-Hill Inc. 1986.
- [3] Edge detection with image enhancement via dynamic programming. M. Furst & P. Caines. McGill university, Montreal, Canada. Computer vision, graphics and image processing 33, 263-279. 1986.
- [4] Matched filter identification of left-ventricular endocardial borders in trans-esophageal echocardiograms. Paul Detmer, G. Bashein & Roy Martin. IEEE transactions on medical imaging, vol. 9, No. 4, December 1990.
- [5] Automated ultrasonic measurement of the carotid artery using dynamic programming. Tomas Gustavsson, Quan Liang, Inager Wendelhag & John Wikstrand. School of Electrical & Computer Engineering, Chalmers University of technology & Wallenberg laboratory for cardiovascular research, Göteborg, Sweden.
- [6] A dynamic programming procedure for automated ultrasonic measurement of the carotid artery. Tomas Gustavsson, Quan Liang, Inager Wendelhag & John Wikstrand. School of Electrical & Computer Engineering, Chalmers University of technology & Wallenberg laboratory for cardiovascular research, Göteborg, Sweden.
- [7] Evaluation of computerized edge tracking for quantifying intima-media thickness of the common carotid artery from B-mode ultrasound images. Robert Selzer, Hoard Hodis, Helenann Kwong-Fu, Wendy Mack, Paul lee, Chao-ran Liu & Ci-hua Liu.
- [8] MatLab's Optimization Toolbox User's Guide.
- [9] Wall thickness of the carotid artery as an indicator of generalized atherosclerosis. M.L.Bots, The Rotterdam Study, 1993.

---

---

### *Related material*

- [1] Model-based image interpretation using genetic algorithms. A.Hill and C.J. Taylor. Department of Medical Biophysics, University of Manchester, 1991.
- [2] Using dynamic programming for solving variational problems in vision. Amir A, Amini, Terry E. Weymouth and Ramesh C. Jain. IEEE, 1990.
- [3] Rational gain compensation for attenuation in cardiac ultrasonography. Hewlett E. Melton, Jr. and David J. Skorton. Department of Electrical and Computer Engineering and Internal Medicine, and Cardiovascular Centre. University of Iowa and the Iowa City VA medical centre, 1983.
- [4] Combined wall thickening of carotid and femoral arteries in patients with isolated hypercholesterolemia. J.Gariepy, A.Simon, M.Massonneau, J.Levenson and the PCVMETRA Group.
- [5] Quantitative ultrasonic tissue characterization with real time integrated backscatter imaging in normal human subjects and in patients with dilated cardiomyopathy. Z.Vered, B.Barzilai, G.A.Mohr. Department of Medicine and Physics, Washington University, School of Medicine, 1987.
- [6] Model-based interpretation of 3D medical images. A.Hill, A.Thornham, C.J.Taylor. Department of Medical Biophysics, University of Manchester. 1993.

# *Index*

## **Numerics**

2-D matrix representation 2-4  
3-D image visualization 2-4

## **A**

abstract 5  
accumulator matrix 3-5  
acknowledgements 7  
artery structure 2-3  
atherosclerotic 5  
automatic edge detection 5, 1-1  
automation 9-3  
average intensity 3-3

## **B**

background 2-1  
boundary coordinates 7-1

## **C**

carotid artery 2-2  
comparing boundary detection methods 9-1  
comparison between methods  
  calculation burden 9-1  
  closeness to manual results 9-5  
  degree of automation 9-3  
  extent of manual intervention 9-3  
  programming effort 9-2  
  required training 9-2  
  similarity to a true boundary 9-4  
  summary 9-6  
  variability between users 9-3  
continuity weight 3-8  
convolution 6-2  
coordinates 7-1  
cost function 5

## **D**

dynamic programming  
  abstract 5  
  algorithm 3-3  
  applying a gradient operator 3-4  
  applying an intensity operator 3-3

- backward tracking 3-6
- boundary coordinates 7-1
- changing parameters 3-8
- forward scanning 3-4
- implementation specifications 10-4
- mathematical representation 3-1
- matlab files 10-7
- software 10-1
- training 5

dynamic programming method

- conclusions 9-6

## G

- gradient operator 3-4
- gradient weight 3-8
- grey level intensity 2-4
- Gustavsson 7

## H

- histogram 6-5

## I

- image representation 2-4
- implementation environment 10-3
- implementation specifications 10-3
- IMT 5
- IMT calculation 7-3
- IMT calculation results 8-2
- initial conditions for the fitting
  - search 5-11
- intensity gradient 3-4
- intensity operator length 3-8
- intensity profile 5
- intensity weight 3-8
- interoperator variability 5
- intima media thickness 5

## J

- jack-knife 10-4

## L

- leave one out 10-4
- Liang 7, 10-4

## M

- matched filter 6-3
  - abstract 5

- algorithm 6-1
- boundary coordinates 7-3
- changing parameters 6-8
- definition 6-2
- implementation specification 10-7
- matlab files 10-14
- order of peak 6-4
- rank of peak 6-4
- software 10-3
- Training 6-1

Matched filter method

- conclusions 9-6

MatLab® 10-7

maximum gradient

- abstract 5
- algorithm 4-1
- approximate echo boundary 4-2
- boundary coordinates 7-2
- changing parameters 4-8
- eliminating the weak edges 4-6
- initial edge 4-2
- matlab files 10-8
- software 10-2
- sub-pixel resolution 4-5
- training 5-1

maximum gradient method

- conclusions 9-6

model-based

- abstract 5
- algorithm 5-1
- changing parameters 5-11
- detection 5-10
- implementation specification 10-5
- matlab files 10-10
- piece-wise linear 5-2
- software 10-2
- sum-of-two-gaussians 5-2
- training 5

Model-based method

- conclusions 9-6

## N

- non-invasive 5, 2-1

## O

- order of continuity dependency 3-9
- order property 6-5

**P**

parallelism 3-9  
phantom 8-1  
piece-wise linear 5-2  
pointer matrix 3-5

**Q**

Quan Liang 7, 10-4

**R**

rank property 6-5  
reference 8-1  
results 8-2  
results of IMT calculation 8-2

**S**

smoothing 6-2  
software description 10-1  
sound waves 2-1  
sub-pixel resolution 5  
sum-of-two-gaussians 5-2

**T**

terminology 2-1  
Tomas Gustavsson 7  
training 10-4  
two dimensional array 2-4  
typical histograms 6-6

**U**

ultrasonography 5  
ultrasound imaging technique  
    background 2-1  
    producing images 2-2

**V**

variability 5