Computer Vision and Analysis of the Carotid Artery

A thesis submitted to the University of Manchester for the degree of Master of Science in the Faculty of Medicine, Dentistry and Nursing

Michael James Pigram

Department of Imaging Science and Biomedical Engineering

October 1999

Contents

Figures

Abstract

Computer vision techniques have previously been applied to ultrasound images of a longitudinal section of the common carotid artery. These techniques have enabled tracking of the near and far wall, from which the distension waveform has been measured. However, they do not allow for translational movement of the artery. In this dissertation, a technique is described for tracking points on the walls of the common carotid artery, using ultrasound image sequences of the transverse section. Images were taken from the video output of ultrasound machines and converted into sequences of bitmaps. A computer program was written in IDL to produce an active shape model from a set of training images. This model gave a compact description of the shape of the artery walls and the grey scale variation present. In a second program, the model was applied to other image sequences, in which the artery walls were detected. A set of tools was created that displayed the motion of the artery cross-section as changed over time. These show the distension waveform as a three-dimensional surface (2D+T) and the translational motion of the artery. Comparisons were made with other studies of the properties of the common carotid artery, which demonstrated the accuracy of this method.

Acknowledgements

I would like to thank my supervisor, Mr Stephen Russell, whose original idea this project is based on, and also Mr David McHugh, both at Christie Hospital.

Many thanks to Miss Helena Bodill for her time and patience in acquiring the ultrasound images and testing my programs and to Mr Mo Baguneid for his ideas on clinical applications and enthusiasm for the work.

Thank you to the staff at Stepping Hill Hospital, for allowing the use of their ultrasound machine.

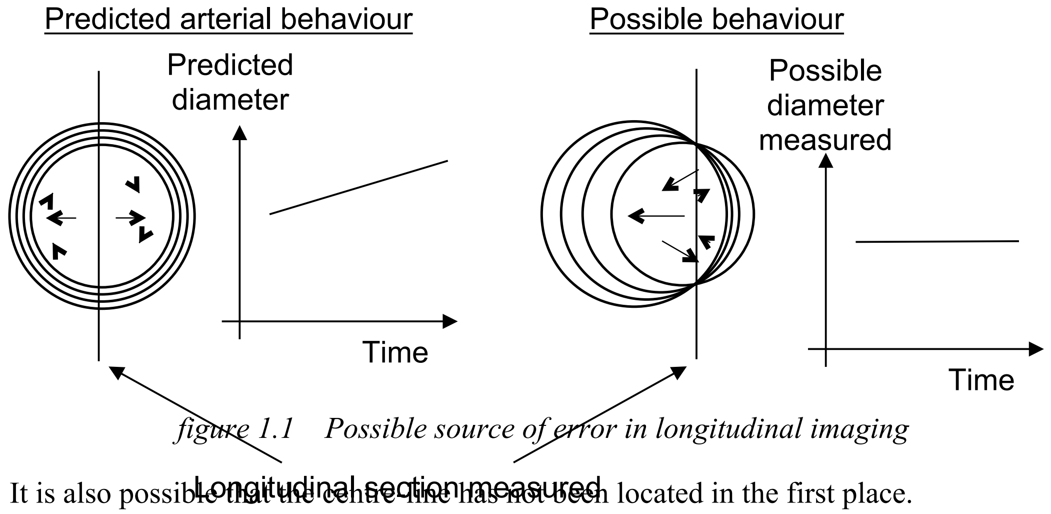
Preface

The author graduated from the University of Cambridge in 1994 with a BA degree in Mathematics. Since then he has worked in the Department of Pharmacy, Policy and Practice at the Keele University and, whilst studying for the MSc of which this dissertation is a part, at North Western Medical Physics.

Chapter 1: Introduction

Aims

Many studies have investigated the properties of the common carotid artery but, to my knowledge, no-one has studied the artery cross-section *in vivo*, using non-invasive techniques. Previous studies have looked at a longitudinal section of the artery, from which translation movement of the artery cannot be ascertained, though this movement may cause errors in such studies. If the centre-line through the artery has been correctly located, the artery may well move during a cardiac cycle:



Predicted arterial behaviour
Possible behaviour
Predicted diameter
Time
Possible diameter measured
Time
Longitudinal section measured

figure 1.1   Possible source of error in longitudinal imaging

It is also possible that the centre-line has not been located in the first place.

A further problem of longitudinal imaging is the finite beam width, as the artery walls curve over the width of the beam, there will be a small amount of blurring of the detected interfaces.

My aim was to produce a computer program that could process a sequence of ultrasound images of the cross-section of the common carotid artery. It should be able to detect the inner wall, and possibly the outer wall, of the artery in each frame of the sequence. The program should then calculate parameters of the artery that are clinically relevant and display these parameters as they change over time.

Arteries

The arteries are the major vessels carrying blood away from the heart and in all cases, except for the pulmonary artery, their job is to efficiently transport oxygenated blood to the capillaries where nourishment can be distributed to the tissues. The heart pumps blood in pulses, forcing 70cm^3 into the arteries approximately every 0.8 seconds. The beating of the heart causes blood pressure to range from 70 to 120 mm Hg under normal conditions [1].

Of particular interest here are the common carotid arteries. These are the major arteries supplying blood to the head. The left and right common carotid arteries run up the neck, either side of the oesophagus.

figure 1.2 Structure *t of the arterial system*

Arterial physiology

Artery walls can be divided into three distinct layers [2]:

1. Tunica intima, the inner layer of the wall. The inner most part is a layer of endothelial cells, which provide a smooth surface causing minimal resistance to blood flow. Further out is a layer of connective tissue containing elastic fibres and the outer part of the intima is the internal elastic membrane, which is a thick layer of elastic fibres.

2. Tunica media, the middle and thickest layer of the wall. This consists of concentric sheets of smooth muscle tissue, with collagen fibres holding it together and connecting it to the inner and outer layers. Another layer of elastic fibres, the external elastic membrane, surrounds these muscle cells. Contraction of the muscle cells causes the artery to contract.

3. Tunica adventitia (or tunica externa), the outer layer of the wall. This is a layer of connective tissue, mostly collagen fibres with a few elastic fibres, which protects the artery from damage and keeps the blood warm on the journey to the peripheries.

figure 1.3 Structure of an artery wall

When the left ventricle of the heart pumps blood into the systemic circulation, the system is said to be in systole. Elastic arteries, such as the common carotid artery, are stretched during systole, stretching the elastic fibres in the tunica media. When blood is entering the left ventricle from the left atrium and so blood is not being pumped into the systemic circulation, the system is said to be in diastole. During diastole the systemic blood pressure drops, allowing the elastic fibres in the tunica media to relax, so the diameter of the artery reduces. The expansion and contraction of the elastic arteries, dampens the pressure changes, so that blood flow in the arterioles and capillaries is continuous.

Measures of elasticity

As mentioned previously, arteries expand radially during systole and contract during diastole. Elasticity is an important measure in determining how healthy arteries are. Elastic moduli are defined as follows [3]:

Modulus = (change of pressure) / (fractional change of dimensions) *equation 1.1*

There are three common methods of measuring intra-arterial pressure:

- Invasively using a catheter with a pressure sensitive tip fed into the artery. This gives the most accurate measure of pressure but introducing a catheter into the blood stream always carries the risk of damaging the artery or of infection. So, unless the catheter is also to be used for another purpose, this technique is hard to justify.

- Non-invasively, using a solid-state high fidelity pressure transducer placed on the skin directly over the carotid artery. This gives a relative measure of pressure, which can be calibrated using mean brachial artery pressure (measured simply with a sphygmomanometer). This is the least traumatic method of acquiring continuous carotid artery blood pressure measurements, but as the transducer must be pressed against the skin, the motion of the artery may be affected.

- Estimation – non-invasive measures of systolic and diastolic blood pressure using a device attached to the finger have been shown to give a reasonably accurate estimate of intra-arterial pressure. Parati et al. [4] recorded average discrepancies between finger and intra-arterial blood pressure during 30 minute studies of 6.5 +/- 2.6 mm Hg and 5.4 +/- 2.9 mm Hg for systolic and diastolic blood pressure respectively (typical blood pressures were 125 mm Hg systolic and 75 mm Hg diastolic [8]). Alternatively, estimates for systolic and diastolic blood pressure in the common carotid artery can simply be set equal to brachial artery pressure – Salomaa et al. [5] concluded that for middle aged people examined at rest, this is a fair assumption.

The dimension measured in calculating carotid artery wall stiffness is usually inter-adventitial diameter (distance between near and far wall media-adventitia boundaries) [6], [7], [8], [9].

Bella et al. [6] quoted three useful measures of elasticity:

Midwall Peterson's elastic modulus: $$E_{pmid} = \frac{P_s - P_d}{Mid_s - Mid_d} \times Mid_d$$ *equation 1.2*

Midwall Young's elastic modulus: $$E_{mid} = \frac{P_s - P_d}{Mid_s - Mid_d} \times \frac{Mid_d}{FW_d}$$ *equation 1.3*

Midwall stiffness index β: $$Mid\ \beta = \frac{\log_e\left(P_s / P_d \right)}{Mid_s - Mid_d} \times Mid_d$$ *equation 1.4*

Where subscripts $_s$ and $_d$ refer to systolic and diastolic measurements, *P* is blood pressure, *Mid* is midwall diameter and *FW* is far wall thickness.

The equations above assume a linear relationship between pressure and arterial distension. Hayashi et al. [10] have validated this assumption for blood pressure in the range 60-160 mmHg. Above this range they suggest describing the modulus of elasticity by:

$$E_{inc} = \frac{\delta P}{\delta R_0} \frac{2(1-\nu^2) R_i^{\ 2} R_o}{(R_0^{\ 2} - R_i^{\ 2})}$$ *equation 1.5*

Where R_0, R_i and ν are the external radius, internal radius and Poisson's ratio, respectively. Assuming the artery wall to be an incompressible and isotropic material, they took Poisson's ratio to be 0.5.

The ARIC study

One of the largest surveys of common carotid artery elasticity and related variables was for the Atherosclerosis Risk in Communities (ARIC) study [8], [11]. Four groups (black and white, men and women) totalling 10,920 individuals all aged between 45 and 64 were measured for the study with the aim of relating common carotid artery elasticity with intimal-medial thickness (IMT). Two ultrasound measurements were taken: the IMT (distance between blood-intima and media-adventitia boundaries) using longitudinal B-mode ultrasound and the inter-adventitial diameter using echo tracking.

The main finding of the study was that, over the range of people studied, thicker common carotid artery walls did not relate to lower elasticity, except for the thickest 10%. The measurements of artery dimensions were as follows:

Type of B-Mode measurement	Mean (mm)	Sample size
Mean artery diameter	7.72	1,663
Minimum lumen diameter	6.23	911
Maximum near-wall thickness	0.76	967

Maximum far-wall thickness	0.80	1,377

Table 1.1 Common carotid artery dimensions

Arterial distension waveform

Stadler et al. [12] used three different ultrasound techniques to measure the distension of the right common carotid artery over time. They found that echo tracking gave the highest resolution but, due to its poor reproducibility, was no better than B-mode imaging of a longitudinal section of the artery (M-mode imaging was significantly worse than both). The normalised waveforms produced by the three methods are shown in figure 1.4, where E, B and M are echo tracking, B-mode imaging and M-mode imaging, respectively. As might be expected, the artery expands rapidly (corresponding with the R wave of the cardiac cycle [13]) and then slowly contracts (due to the relaxation of the elastic fibres).

figure 1.4 *tid artery*

By calculating the power spectrum, they found that all frequencies above 15 Hz were at least 40 dB down from the fundamental frequency (approximately 1 Hz). From this, they concluded that no significant information would be lost by sampling the images at 30 Hz.

Arterial pathology

Arteriosclerosis

This is a thickening and toughening of the artery wall, which contributes to a large proportion of deaths in the Western World, through coronary artery disease and by leading to strokes [2]. The major form of the disease in the elastic arteries is atherosclerosis. This is caused by damage to the endothelial lining of the artery, allowing lipid deposits to form in the tunica media. The cells of the artery wall phagocytize the lipids making a mass of fatty tissue (a plaque), which presses against the lining. If the plaque breaks through the lining into the artery, platelets in the blood will stick to it, increasing its size. By reducing the effective cross sectional area of the artery, the plaque can cause turbulent flow of the blood. If some of the platelets break off, the blood can carry them until they get stuck, possibly causing a stroke.

One study [7] has related the stiffness of the common carotid artery with the degree of atherosclerosis present. On a group of 465 patients, they measured blood pressure, arterial diameter (using echo tracking), and its pulsatile change to determine the stiffness parameter β (equation 1.4). They were able to make post-mortem examinations on 30 of the patients over the next 60 to 370 days, which provided 60 common carotid artery specimens (both left and right were used). The arteries were graded by the extent of atherosclerosis as shown in figure 1.5.

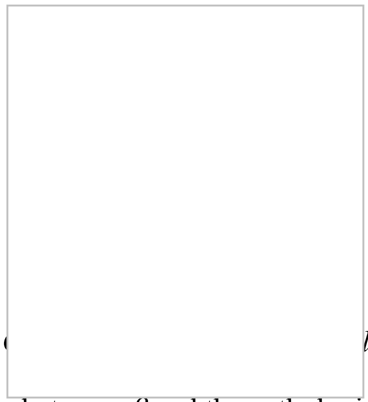


figure 1.5 *lassification*

There was a strong correlation between β and the pathological atherosclerosis grade.

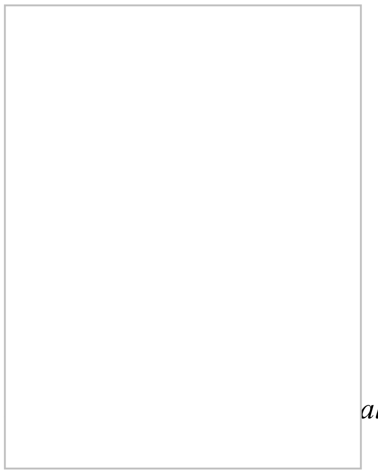


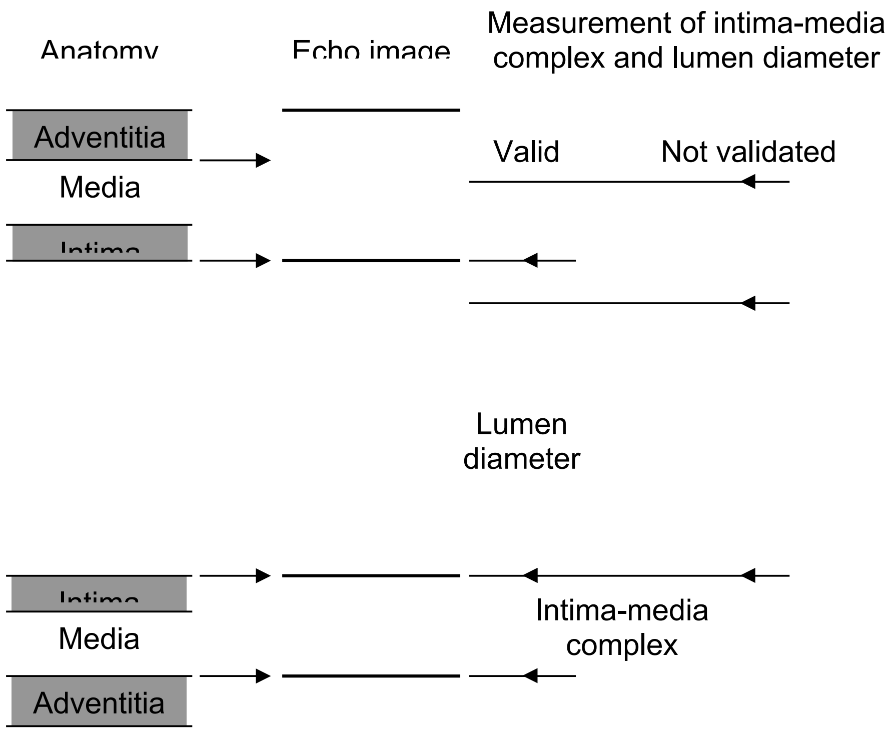
figure 1.6 Correlation *al atherosclerosis grade*

Ultrasound

Ultrasound echoes are generated at the interface of materials with differing acoustic impedance. In biological material this occurs at two levels. Within a single type of tissue echoes may be produced, mainly by cell walls, the strength of which depends on how smoothly the acoustic impedance changes across the tissue. This property gives different types of tissue an individual appearance. In the highest resolution images (such as those achieved with the ATL HDI5000) actual tissue structure can be seen. The strongest echoes, though, are from the interfaces between different tissues. Whilst the interface may be sharply defined and so considered not to have a thickness, the echo from the interface will have a finite thickness (or depth). This is due to the axial resolution of the ultrasound machine and so is largely dependent on the frequency used.

The position of an object in an ultrasound image is determined by the leading edge of the echo because, as mentioned earlier, the echo from an interface will have a finite depth.

figure 1.7 Differences between anatomy and echo image



Computer Vision

Digital images are represented within a computer as an array: for an 8-bit grey-scale image a 2-D array of integers 0-255 is used. Each value represents the grey-scale intensity of each pixel with 0 for black and 255 for white. The distribution of pixel intensities in an image can be seen by plotting a histogram showing the frequency of each intensity level.

Point processing

The pixel intensities in an image are often grouped together, so the majority of pixels fall into a couple of small intensity ranges (usually light pixels in the foreground and dark pixels in the background). Histogram equalisation changes the image so that there is an equal number of pixels of all intensities, thus making the intensity histogram flat. This helps people to see objects in the image more clearly as it tends to even out the contrast. In computer vision histogram equalisation can be used in pre-processing to standardise the intensities of images so for example objects can be recognised even if they are darker than usual as long as the background is also darker than usual.

Segmentation

For an object to be recognised it must in some way be separated out from the other objects and the background in the image. In the very simplest case an object may be brighter than everything else and so can be segmented by thresholding the image. In the output image only the pixels that are above a certain intensity, and so are bright enough to be representing the object, are shown. Often using point processing the object recognition problem can be reduced to the use of thresholding, following a sequence of point processing operations.

Model based vision

Many object recognition problems cannot be solved by the use of simple image processing and segmentation alone. Objects can be viewed in many different poses, under many conditions and can be partially obscured by other objects or by noise in the image. Under these conditions it is still easy for a human observer to pick out the object so how can computers be made to do this? The answer is that knowledge based processing must be used. The object recognition algorithm must contain information about the shape of the object to be recognised and information about the range of conditions under which the object will be used.

Snakes – Active Contour Models

The Active Contour Model developed by Kass et al. [14] is an energy minimising spline. This is a flexible curve, which can be thought of as subject to various forces and constraints. The spline is constrained by being attributed an energy function which is to be minimised. The minimisation occurs by the spline wriggling around until it reaches the lowest energy state – hence the name ‘snake’. The energy function has 3 components:

$$\begin{aligned} E^*_{snake} &= \int_0^1 E_{snake}(v(s))ds \\ &= \int_0^1 E_{\text{int}}(v(s)) + E_{image}(v(s)) + E_{con}(v(s)) \quad ds \end{aligned} \qquad \textit{equation 1.6}$$

The length along the snake is parameterised by $v(s) = (x(s),y(s))$.

- E_{int} - Internal energy. This relates to the first and second order differentials along the snake and affects the way in which the snake can bend.

- E_{image} - Image energy. This determines the how the snake is attracted to features in the image - a weighting is used to set the strength of the snake's attraction to lines, edges and line terminations.

- E_{con} - External constraint energy. Because the first 2 constraints work over a short range, the snake may fall into a local minimum. This constraint is used to enable the user to affect the global positioning of the snake and so find the global minimum.

Any type of energy function can be applied by the user to push or pull the snake around.

To stop the snake from getting caught in local minima, the image can be blurred using a Gaussian mask with the level of blurring reduced as the snake homes in on the desired feature.

Unlike some other models, this method does not generally use training to give it prior knowledge, so it does not contain any global shape constraints. It will therefore pick out the most clearly defined object, or the one that the user pushes it towards, rather than the one most like a desired shape. Because of this, once an object has been segmented, it cannot be automatically labelled.

The use of energy minimisation appears to be a very good method of attracting a model to features in an image and has been widely used in fitting contours to objects in medical images. One example is the use of the ADDER method [15], which has been applied successfully to high resolution, intra-arterial ultrasound images of the carotid artery. The rotational scanning used produces an image with a continuous artery wall, which is absent in externally measured ultrasound images. The drawback of this method is the invasive measurement.

One of the problems of using a snake model with ultrasound images is the high level of noise and gaps in object boundaries [16]. Chen et al. [17] suggest applying the snake model to a distance map of the image, which they claim, removes the speckle noise and produces a closed boundary. Another solution to this problem is to use further information to constrain the model to a particular shape.

Active Shape Models

For many types of object, an example of that object will conform to a general shape. Such an example can be recognised by the position of landmark points on it [18]. Point distribution models are used to describe objects like this by giving a statistical description of the relative positioning of these landmark points.

An active shape model is a type of point distribution model, which is given a very compact description through the use of eigen-vectors. This method of modelling is described by Cootes et al. [19], [20] and is the basis of the method I have used here. The model is determined through the use of a set of training data. Points are placed in equivalent positions on all of the images, from which the average position of each point can be calculated along with the covariance matrix. The principal eigen-vectors of the covariance matrix (those associated with the largest eigen-values) are calculated and using these a large proportion of the variation can be described with a fraction of the original data. Because eigen-vectors are used we are guaranteed that this is the most compact representation of a point distribution model.

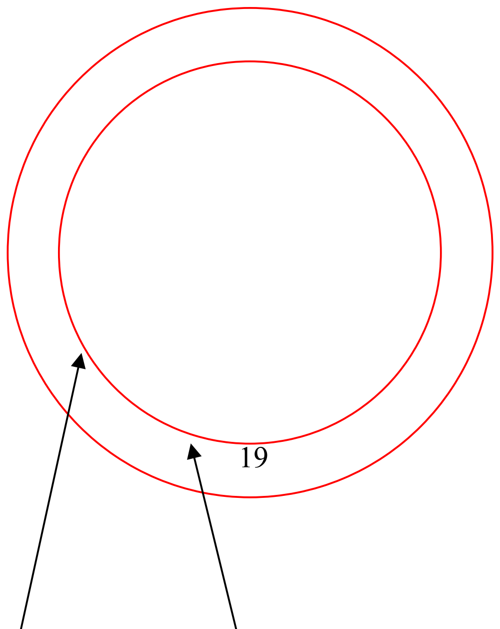
The compactness of the description of point distribution allows for other model features to be included, for example the grey-scale variations around the model boundary. To do this it is necessary to look at the grey-scale variations along lines normal to the boundary in all of the training images and for every point in the model to calculate a mean profile and also the principal eigen-vectors of the covariance matrix of the greyscale values along the profiles (this will be described in more detail in Chapter 2).

Having created a model based on a minimal set of eigen-vectors relating to shape and grey-scale variation, it can be used in image searches. To do this, an algorithm is applied which, given an initial positioning of the model, calculates how the points of the model should be moved to improve the fit, then calculates how the parameters should be changed (within the global shape constraints) to do this. Function minimisation is used to attract the model to image features in the same way as with snakes but the model constraints keep it from being attracted to invalid local minima.

This method has been widely applied to object recognition in 2-D images and has also been applied to tracking objects over time in both 2-D and 3-D.

Object Recognition in Ultrasound Images

Until recently, many of the object recognition / border detection algorithms applied to ultrasound images have been highly tailored to the particular application. The reason for this comes down to the peculiar nature of ultrasound imaging. Nearly all imaging modalities require little or no user interaction, so the recognition method must be able to pick out an object from a general scene, whereas with ultrasound, user interaction is an integral part of the imaging process so the object to be detected is the major feature of the image. It may seem from this, that object recognition would be easier in ultrasound images, but the converse is true. Ultrasound imaging requires so much interaction because to get a clear image, not only must the probe be carefully positioned, but also the image contrast must be adjusted at a range of depths. Still, this leaves an image in which only the horizontal components of tissue interfaces are clear, the grey level for the same tissue over an area can vary widely and also the contrast between the same two types of tissue can vary widely. This is easily demonstrated by one of the images acquired for this research:



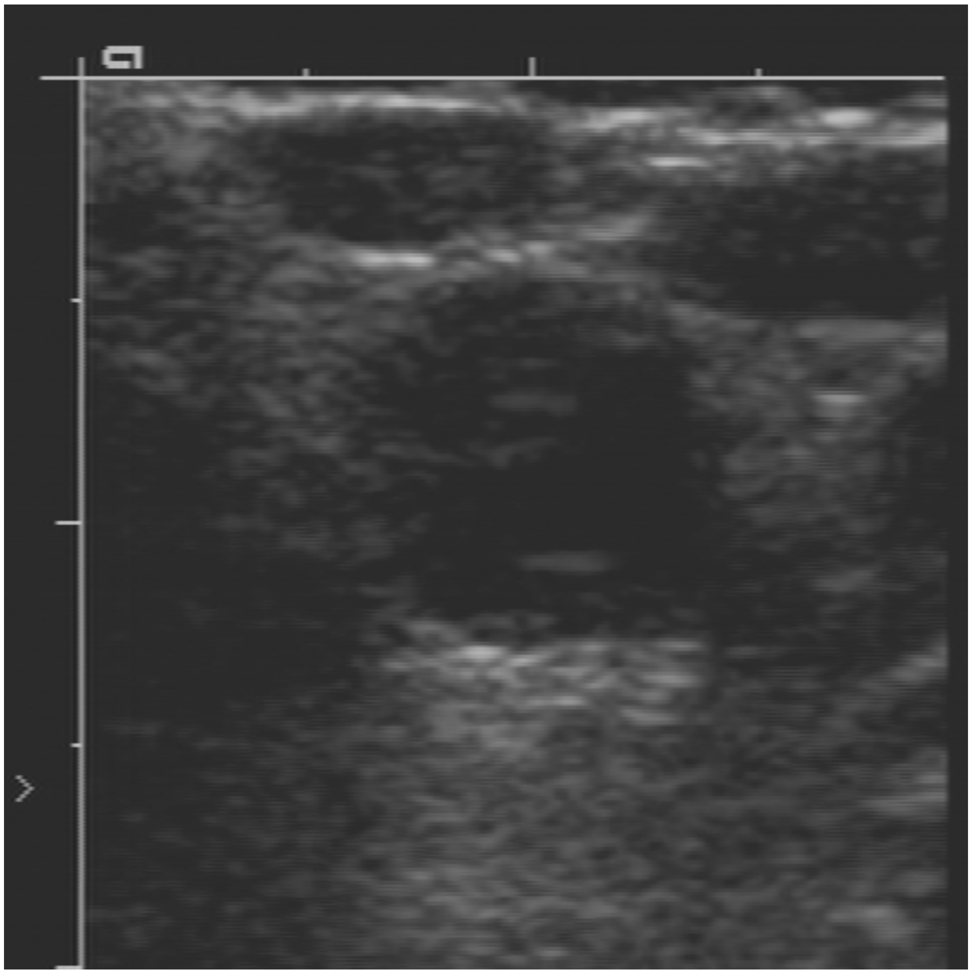


figure 1.8 Typical ultrasound image of the common carotid artery

In figure 1.8 the lumen (inner part, containing blood) can be seen clearly, but although its contents are homogeneous and should not produce any echoes, artefacts can be seen. The approximate position of the inner and outer artery wall is highlighted in red, parts of the walls can be clearly seen, especially the near and far walls (the top and bottom). The side walls are harder to pick out, and there are complete gaps at the 4 and 8 o'clock positions, but with the knowledge that the walls are approximately circular it is still quite easy to estimate where the walls should be. There is a very wide variation in grey-scale intensity all around the artery wall, with the far wall particularly bright. This echo enhancement is due to the low attenuation in the artery, which causes the signals from deeper structures, such as the far wall, to be over amplified [21].

None of these problems in artery wall detection are encountered when imaging the artery longitudinally or radially, using a probe inside the artery. The disadvantages of these methods, pointed out earlier, make it worthwhile trying to detect the artery walls in a cross-section.

Chapter 2: Method

Equipment used and its configuration

Ultrasound equipment

The first set of images was acquired using an Acuson 128 at the Christie Hospital in Manchester. For the second set an ATL HDI 5000 at Stepping Hill Hospital was used and for all subsequent imaging an Acuson 128 at Manchester Royal Infirmary was used. A 7MHz linear array transducer was used with all three machines

Machine settings

The images captured at Christie Hospital and at Stepping Hill Hospital were of my colleagues and myself. In acquiring these images I was able to learn more about the capabilities of these machines and the built in programs for vascular imaging [22]. Using the Acuson 128 there are 3 main settings that affect image quality:

1. Pre-processing, which controls how sharp or smooth the borders are in the image. Higher pre-processing values produce images with sharper edges.

2. Persistence, which affects the degree of temporal smoothing. High persistence reduces noise in slow moving images, but will produce blurring in fast moving images.

3. Post-processing, which affects the relationship between echo amplitude and pixel intensity. The range of post-processing curves available is shown in figure 2.1.

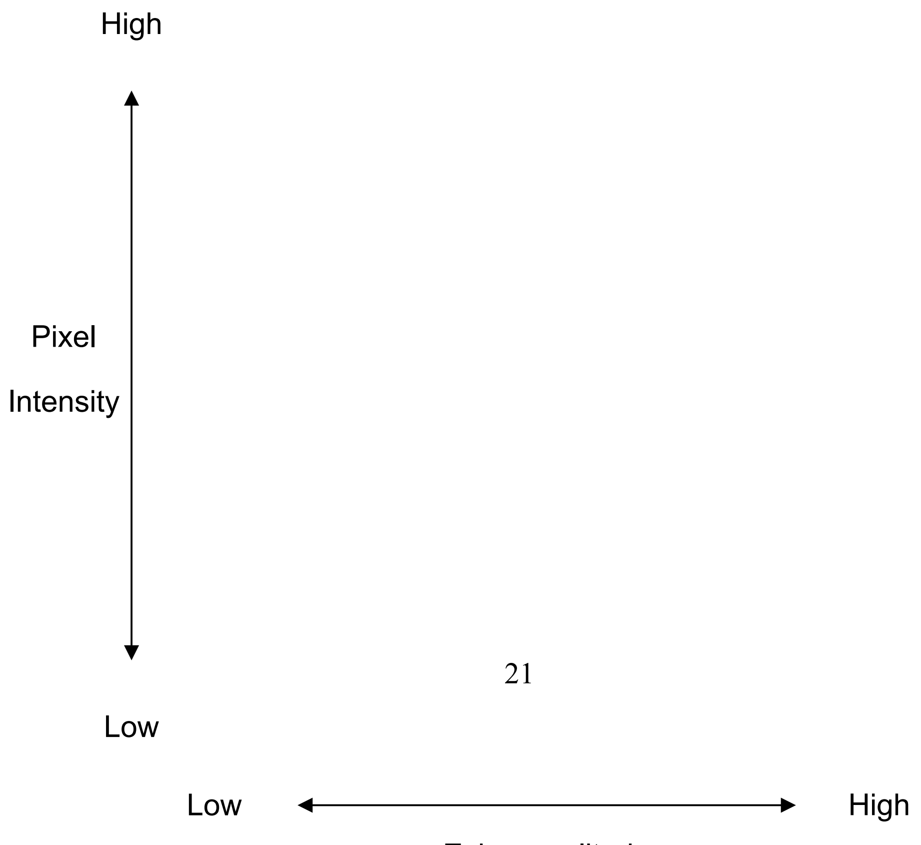


figure 2.1 Acuson 128 Post-processing curves

Initially I tried using my own settings to get as clear an image as possible. I used a pre-processing value that gave medium sharp edges, zero persistence as I was trying to detect fast moving edges and the POST 0 post-processing curve to give a linear relationship. The Acuson 128 has a pre-defined setting for carotid artery imaging, which is designed to work with colour Doppler imaging. The parameters for this are: 1 for pre-processing, 2 for persistence and 4 for post-processing. There was no discernible difference between image obtained using either setting, so I decided to use the pre-defined carotid artery settings from then on, as an ultrasonographer would be likely to use these settings too.

Computer hardware

The PC used was a dual Pentium II 350Mhz machine with 64Mb RAM single, 9Gb hard drive, 15inch colour monitor running windows NT4 service pack 4. To acquire images from the ultrasound machine, the monochrome video output was connected via a BNC cable to a Leutron Picport video capture card. Video Savant software was used for direct streaming of captured images onto the drive. This was crucial in order to be able to store the information on the drives at sufficient speed. To store the information two low voltage differential (LVD) 4Gb SCSI drives were used. Individually these supported a theoretical data stream rate of 80Mbits per sec per drive and when used as a striped set therefore had a theoretical data stream rate of 160 Mbits per second appearing as a single 8Gb drive. A separate IDE drive was used for the operating system, applications and to store formatted images on. Following the video acquisition, the frames were catalogued and converted to sequences of uncompressed 8-bit grey-scale bitmaps. The majority of images were 700*450 pixels and typically the sequences captured were 6 seconds long. At 25 fps this equates to 45Mb per sequence, so it was very useful to have the relatively high processing power to quickly process the data into images. Because of the large volumes of data created, the image sequences were copied onto CD.

Choice of computer software

There was a wide range of computer languages to choose from, with which to write an image-processing program. The factors I considered in making my choice were: how quickly I could develop programs in the language, the functionality of the language in the area of image processing, the appearance of a finished program to an end user and the speed of operation of a program written in the language. The most suitable languages that were available to me were IDL, PV-Wave, Matlab and Visual C++. The first three languages are of a similar style, being very high level languages designed mainly for easy and fast manipulation of large arrays. This makes then highly suitable for image processing as well as other ‘number crunching’ applications such as mathematical modelling. All three contain pre-compiled, optimised functions and procedures, which can be used to perform most of the operations required. Large libraries of uncompiled programs are also supplied with these languages and further programs are available for free on the Internet. Matlab and IDL appear to have a better selection of these library programs available than PV-Wave, as they seem to have a greater number of users with the 'freeware' ethos. Unfortunately Matlab scripts cannot be compiled, so do not run as efficiently as programs written in the other languages. The feature that made IDL my favourite of these high level languages is the ability to produce a graphical user interface (GUI), which makes for a much more user friendly end program than the command line driven Matlab and PV-Wave.

Visual C++ is very different from the other languages, being based on the fairly low-level object-oriented language C++. On top of the standard features of C++ it contains support for producing a 'Windows' interface. There are a great many C++ library routines available for image processing, which combined with a visual user interface can make very powerful, easy to use programs. Visual C++ also has the advantage that programs can be compiled to produce stand-alone executable files, which do not require the user to own a copy of the language (none of the others support this).

This brought the choice down to either Visual C++ or IDL. I would have preferred to program using Visual C++ as the end product could have been distributed for free, but the pressures of time forced me to choose IDL, as development times are much quicker.

Data Acquisition

Patient sample and Measurement conditions

The measurements taken at Stepping Hill Hospital, using the ATL HDI500 ultrasound machine, were not under standard conditions. The three people measured (myself included) were sitting down and not properly rested before the scans. We all scanned ourselves, none of us having previous experience using that machine.

The majority of images acquired for this project were of 9 patients at Manchester Royal Infirmary, who were attending the Vascular Laboratory clinic and very kindly agreed to be scanned for this project. The scans were taken over a period of two days, all by the same ultrasonographer. Permission was not sought to record any of the details of the patients, as I only wished to gain a range of images and did not intend relating any findings to the actual patients. At the same time and under the same conditions, scans were taken of myself, as I was significantly younger than the patients were and expected to have healthy arteries. All of these subjects had both their left and right carotid arteries scanned whilst lying down and had all rested for at least 10 minutes beforehand.

The ultrasound machine was already connected to the computer described earlier, so when the ultrasonographer was ready, it was simply a matter of pressing a button on the computer to start the recording and stop it approximately 6 seconds later. At the same time a log was made of the artery position – left/right, common/bifurcation/distal. This technique meant that no further demands were placed on the patients, except for the extra time required (approximately 3 minutes per patient).

The computer vision program

I identified three main tasks for the computer program: to create the vision model using a set of training images, to apply the model to new images and to produce an analysis of images. As the analysis follows on directly from the applying the model, these two tasks were grouped together into one program. The code written for these programs is included on an IBM formatted 3½" disc. Files in the format '*.sav' are IDL binary files which act like executable programs when a run-time copy of IDL is included. Files in the format '*.pro' are uncompiled IDL source files which to compile require the full version of IDL but can be viewed and edited in any text processing package. Further installation instructions are included in the 'readme.txt' file on the disc.

Much of the program is an interpretation of the method described by Cootes et al. [19].

Creating the model

I have written a program to enable the user to take sample bitmap images and create an Active Shape Model (ASM) based on them. There are 5 files containing the code, named 'CreateModel*.pro', the compiled run-time program is called 'CreateModel.sav'.

Shape spaces

There are three shape-spaces dealt with here, it is important to understand what is described in each, and how they relate to each other.

In the 'real world' shape space, the coordinates (X,Y) of a point are given as the column and row in the image array to which they relate, and the shape $\underline{X}$ is given by $\underline{X} = (X_0,Y_0,\ldots,X_{N-1},Y_{N-1})^T$.

In the 'normalised' space the coordinates (X,Y) are translated and scaled to (x,y) such that objects in normalised space are centred to (0,0) and have mean radius 1.

$$\underline{X} = s \cdot \underline{x} + \underline{Xc} \qquad \textit{equation 2.1}$$

where $\underline{Xc} = (Xc,Yc,Xc,Yc,\ldots,Xc,Yc)^T$ and s is a scaling factor such that:

$$\sum_{i=0}^{N-1} x_i = \sum_{i=0}^{N-1} y_i = 0 \qquad \textit{equation 2.2}$$

$$\frac{1}{N}\sum_{i=0}^{N-1}\sqrt{x_i^{\,2} + y_i^{\,2}} = 1 \qquad \textit{equation 2.3}$$

A shape in normalised space is described by the vector $\underline{x} = (x_0,y_0,\ldots,x_{N-1},y_{N-1})^T$.

Other implementations of this model allow for rotation between $\underline{X}$ and $\underline{x}$. I chose not to allow rotation for two reasons:

1. Modelling arteries, the rotational symmetry does not call for it.

2. From, for example, figure 2.4 it is clear that ultrasound images do not contain rotational symmetry in the grey scale, but the cause of the asymmetry is the imaging modality itself. Pixel intensity is related to echo strength and echoes are stronger from interfaces that are seen as horizontal in the image, therefore higher intensities will be seen at the top and bottom of the artery than at the sides – rotating the artery will make no difference to this.

In 'model' space, shapes are described by vectors $\underline{b} = (b_1, b_2, \ldots, b_{Tp})^T$. The relationship between $\underline{b}$ and $\underline{x}$ is given later in equation 2.6.

Filtering

Once the user has selected a sequence of bitmap images, not necessarily continuous, a range of filtering options can be applied. If chosen, the filters are applied in the order: histogram equalisation (explained on page 21), thresholding, smoothing and lastly edge enhancement. When thresholding is applied, pixels with an intensity value below the value set by the 'Threshold' slider are set to zero intensity. The smoothing method is a simple box car average in which each pixel in the output image is given an intensity value equal to the average intensity in a square neighbourhood the width of which is defined by the 'Smooth' slider. The Sobel operator is used for edge enhancement, in this copies of the image array are separately convolved with the neighbourhood operators X and Y and the absolute values of the resulting arrays are added.

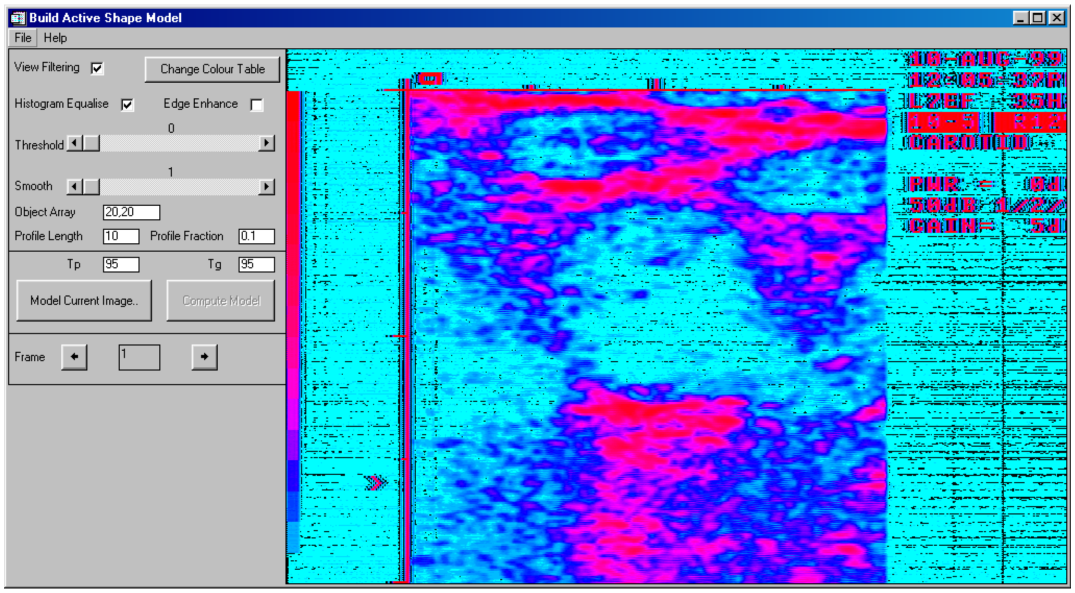
$$X = \begin{pmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{pmatrix} \qquad Y = \begin{pmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{pmatrix} \quad \Rightarrow \quad G = |X \quad I| + |Y \quad I|$$

equation 2.4

Where *I* is the original image, *G* is the filtered image and o is the convolution operator.

Included with the filtering operators is the option to change the colour table used to represent the image array. Usually a linear black-white table is used as this is the representation used on ultrasound machines, but if wished one of many built-in tables can be used and modified. This can be useful if most of the pixel intensities are in a small range as a number of different colours can be squeezed into that range – similar to histogram equalisation.

With the filtering options set, the image can be viewed either filtered and with the chosen colour table or unfiltered and with the standard colour table. In either case, both the filtered and unfiltered images are stored and later two models are created, one based on each.

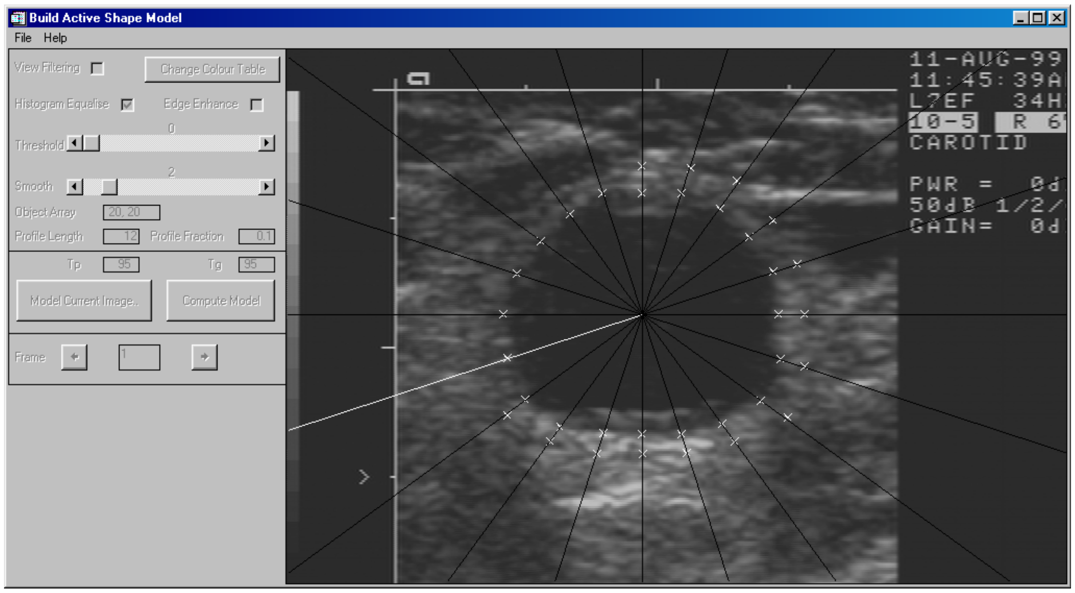


View Filtering [✓]
Histogram Equalise [✓]
Edge Enhance [ ]
Threshold 0
Smooth 1
Object Array 20,20
Profile Length 10
Profile Fraction 0.1
Tp 95
Tg 95
Frame 1

figure 2.2 Interface for building the ASM – demonstration of filtering options

Plotting model points

The program has been written to be applicable to a wide range of active shape modelling tasks. It will allow the model to contain any number of separate objects, each object being represented by any number of points. The user enters the number of points in each object in a comma-separated list in the 'Object Array' box and from this information the program reads in the array and calculates the total number of model points, *N*. In figures 2.2 and 2.3 there are two objects, the inner and outer walls, each containing 20 points. In figure 2.4 there is only one object – the inner wall. In this application of the program, some of the generality has been reduced to help the user. It has been assumed that, as the objects to be modelled are artery walls, the model points will be distributed with radial symmetry and that the objects will be concentric. When the modelling starts, the user is prompted to plot a point in the centre of the artery and from this radial guide-lines are drawn, with the guide-line for the next point being highlighted as in figure 2.3. The user then plots a point where the guide-line crossed the artery wall being modelled.



View Filtering ☐
Histogram Equalise ✓
Edge Enhance ☐
Threshold 0
Smooth 2
Object Array 20, 20
Profile Length 12
Profile Fraction 0.1
Tp 95
Tg 95
Frame 1

figure 2.3 Plotting points in the training images to build the ASM

As each model point is plotted, the co-ordinates of the points (X,Y) are stored in a 2*N element array, $\underline{X}$ (referred to in the program code as *bigX*).

Calculating profiles

Once the last model point has been plotted, the program calculates the intensity profiles along a line through each point, in fact profiles are calculated for both the filtered and unfiltered images. Cootes et al. [19] suggested calculating profiles of the derivative of the grey-scale image as this gives invariance to the scaling of the grey levels. Unfortunately calculating the derivative in ultrasound images only enhances the high level of noise, causing this method to fail. It was hoped that using histogram equalisation would give some of the invariance desired.

The value 'Profile Length' sets the number of pixels towards the centre of the object and away from the centre of the object for which the profile is calculated, so including the model point itself, the total length of the profile is 2* 'Profile Length'+1. If the model were to be built based upon images taken at different resolutions then a profile might reach to the centre of the artery in one image but only reach one third of the way in another image so one profile could not be compared directly with another. To overcome this problem, the value 'Profile Fraction' is entered. This sets the distance towards the centre and away from it that the profile should cover (for example 1/10th of the radius). If the distance to be covered by the profile is less than the number of pixels to be sampled, then some pixels will be sampled more than once. Conversely, if the distance to be covered by the profile is greater than the number of pixels to be sampled, then some pixels will not be sampled (this can be seen in figure 2.4 in which the profile lines are broken).

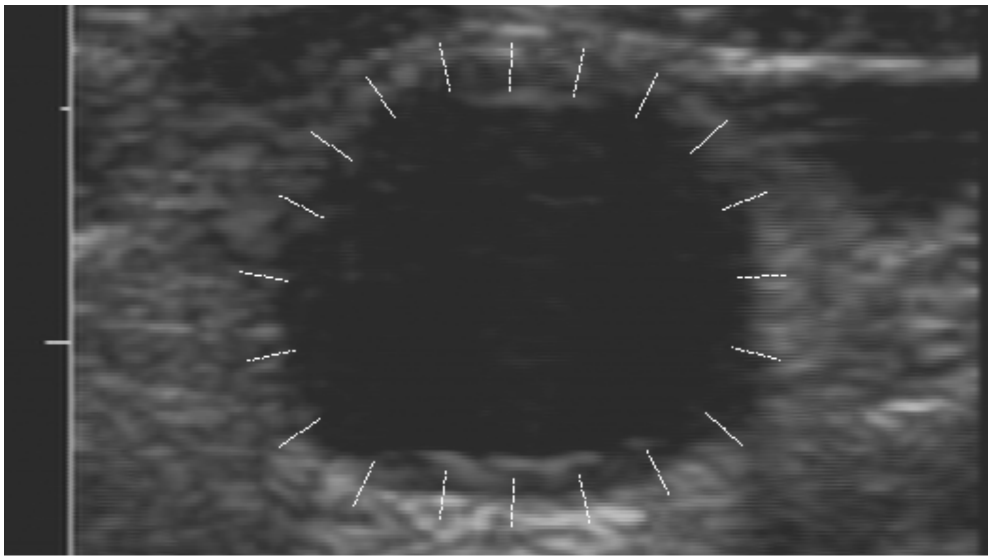


figure 2.4 Profiles plotted on an image to create a model of only the inner wall

The reason for dividing the model points into those contained in separate objects becomes obvious when calculating the direction for each profile. The profile direction chosen for a point is perpendicular to the line joining the point's neighbours. The neighbouring points are generally taken to be those plotted before and after a particular point with the exception that the last point in an object is a neighbour of the first point in the same object and not the first point in the next object. For example if the model contains two objects each represented by four points: 1A, 1B, 1C, 1D, 2A, 2B, 2C, 2D, then the corresponding array of neighbours to one side is 1B, 1C, 1D, 1A, 2B, 2C, 2D, 2A to the other side is 1D, 1A, 1B, 1C, 2D, 2A, 2B, 2C. Example objects with correct and incorrect profile directions are shown below.

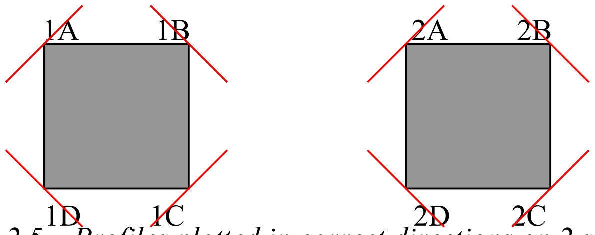


figure 2.5 Profiles plotted in correct directions on 2 separate objects

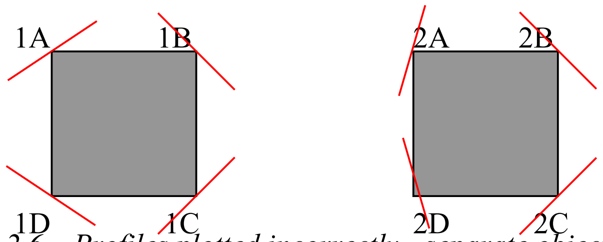


figure 2.6 Profiles plotted incorrectly - separate objects not defined

The data calculated for the profiles in the frame that has just been modelled are appended to two arrays containing data for all the frames modelled so far (*SeriesProfiles* for the unfiltered image and *fSeriesProfiles* for the filtered image).

The program calculates the position of the centre of the artery given by the mean *X* and *Y* co-ordinates and average distance between each point and the centre. These values are then used to normalise the co-ordinates, translating $\underline{X}$ to $\underline{x}$. Here $\underline{x}$ is the array *CurrentModel*. This array is appended to an array containing the model points for all of the frames modelled so far, *SeriesModel*.

Computing the model

After modelling any number of frames, the user can choose for the model to be calculated. At this point the program reads in the user-defined values *Tp* and *Tg* (these are set as percentages, to be explained later).

Simplest to calculate is $\underline{\bar{x}}$ *(Xbar)*, the average position (in normalised space) of each point over all of the frames modelled.

To determine how the position of one point in the model affects the position of all of the other points a 2*N* by 2*N* covariance matrix is calculated (there are 2*N* variables, *N* points each with 2 co-ordinates). With more than a few points in the model, this becomes an inefficient way of describing the point distribution, so the matrix is transformed such that the rows are orthogonal – they are eigen-vectors. The rows are then sorted according to the size of the corresponding eigen-values (creating the matrix *Pn*). Each eigen-vector explains a mode of variation of the model's shape in shape space, so any point distribution in the training set, $\underline{x}$, can be described by the vector $\underline{b_{2N}}$ where:

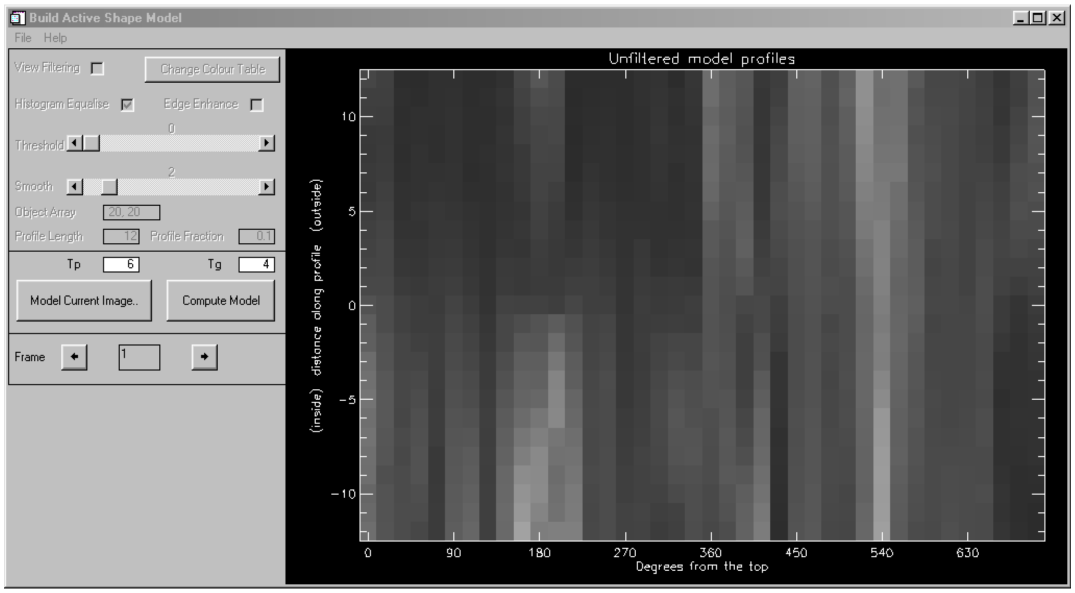
$$\underline{x} = \underline{\bar{x}} + Pn\,\underline{b_{2N}} \qquad \textit{equation 2.5}$$

Each eigen-value represents the amount of the total variation its associated eigen-vector explains. The total variation is explained by all 2*N* eigen-vectors, but in most cases the majority of the variation can be explained by only a few principal eigen-vectors. Here the principal eigen-vectors, which explain *Tp*% of the variability, are stored in a smaller matrix *Px*, the corresponding eigen-values are stored as *lambdaX.* The value in the 'Tp' box is then changed to show how many principal eigen-vectors there are.

$$\underline{x} \approx \underline{\bar{x}} + P_x \underline{b_{Tp}} \qquad \textit{equation 2.6}$$

$\underline{b_{Tp}}$ is a vector whose length is the value 'Tp' just calculated.

The same method is then applied to the profiles through each point in the model for the unfiltered image and filtered image. For each point in the model, the average profile is calculated by simply taking the mean. These profiles are stored in an array $\overline{G}$ (called *Gbar* in the program) (prefixed by *f* or *n* for images which have been filtered or not). The average profile for a series of unfiltered images is shown in figure 2.7.



View Filtering ☐
Histogram Equalise ☑ Edge Enhance ☐
Threshold 0
Smooth 2
Object Array 20, 20
Profile Length 12 Profile Fraction 0.1
Tp 6 Tg 4
Frame 1

figure 2.7 Unfiltered profiles of the computed model

Using the same method for calculating the matrix *Pn*, a matrix *Pgj* is calculated for the profile through each model point, with all of the *Pgj* matrices forming a matrix *Pg*, which explains the total variability of the profiles through all of the model points.

$$\underline{G_j} = \underline{\overline{G}_j} + Pg_j \, \underline{b_{2PL+1}} \qquad \textit{equation 2.7}$$

$\underline{G_j}$ is a profile through point j in the training set.

$\underline{\overline{G}_j}$ is the average profile through point j in the model.

$\underline{b_{2PL+1}}$ is a (2*ProfileLength+1) long vector.

The matrix *Pg* is then reduced so that, on average, the remaining eigen-vectors explain *Tg*% of the intensity variation in the profiles. The *N* sets of corresponding eigen-values, λ_g, are stored as *LambdaG*. The value in the 'Tg' box is then changed to show how many principal eigen-vectors there are in *Pg*.

Once a model has been calculated it can be saved or re-calculated using different values of *Tp*% and *Tg*%. The saved model can be restored, either in the same program so that more data can be added to it, or in another program in which the model can be applied to new images.

Displaying shape variation

I added a tool to the program, which, once the model has been calculated, displays the two major modes of shape variation that the model encompasses. It is recommended

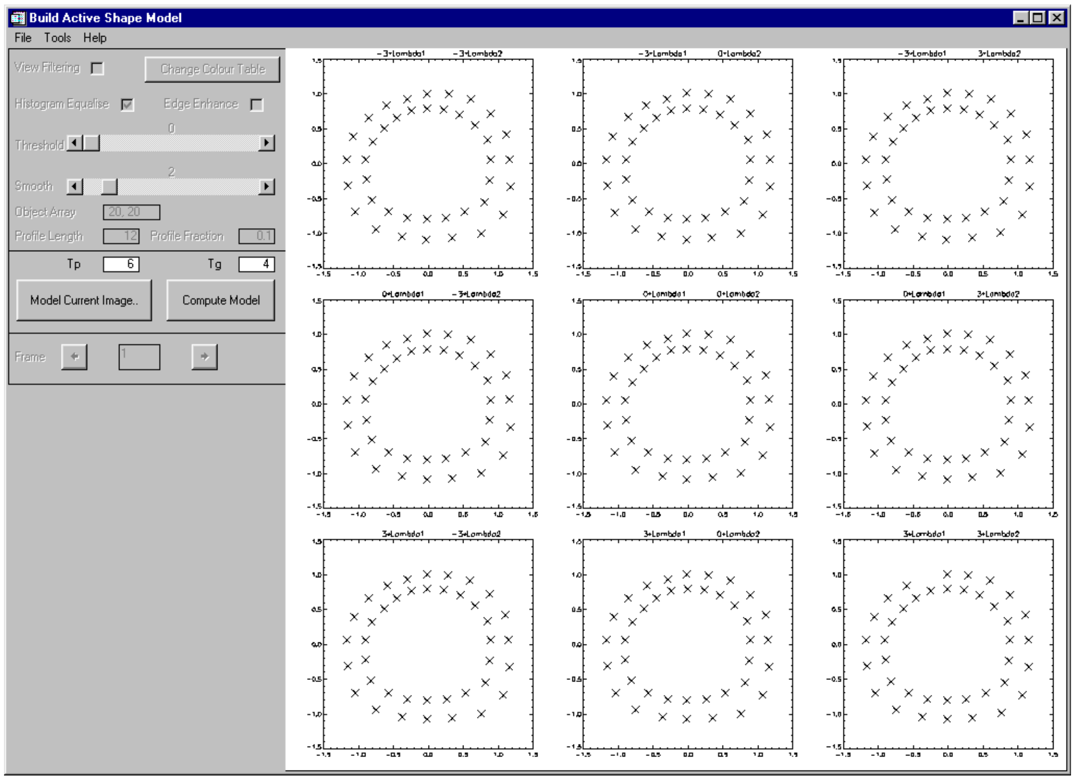
[19] that for a shape to be considered a valid example of the model, the values b_k should satisfy the equation:

$$-3\sqrt{\lambda_k} \le b_k \le 3\sqrt{\lambda_k}$$ *equation 2.8*

Where b_k is the kth element of $\underline{b}$

In other words, each descriptor of a shape (b_k) should lie within three standard deviations of the mean.

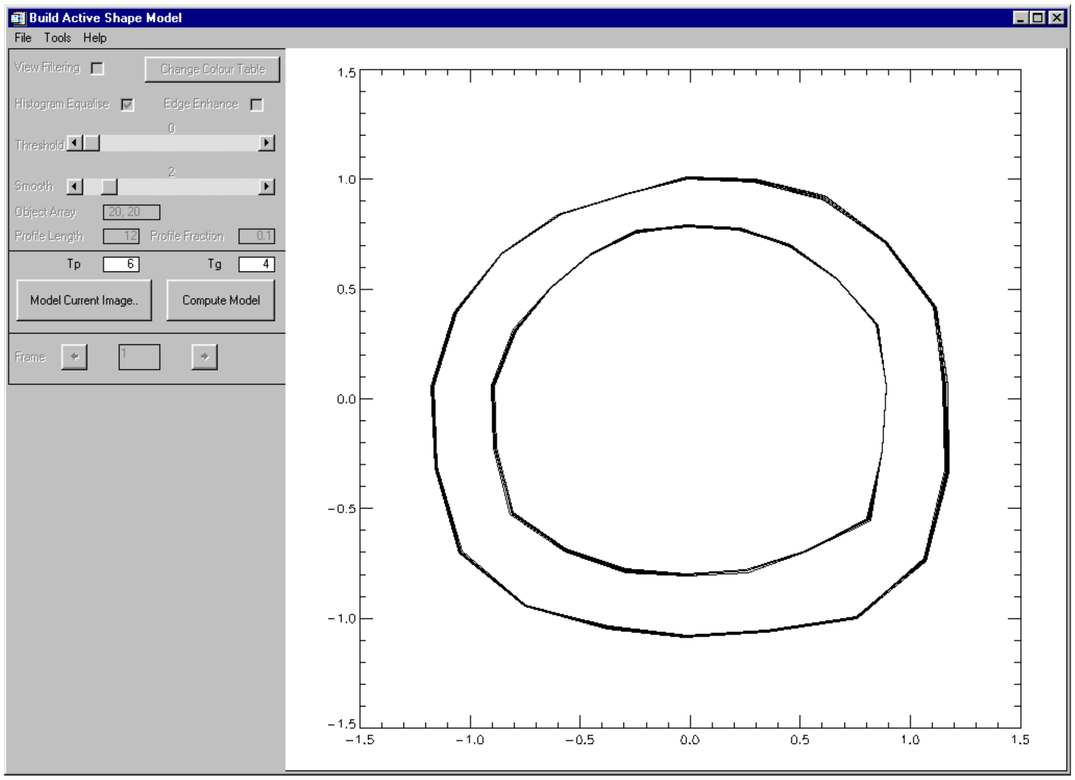
The tool displays the effect of setting $b_k = -3\lambda_k$, 0, $+3\lambda_k$ for k=1,2. An example is shown in figure 2.8.



View Filtering [ ]
Histogram Equalise [✓]
Edge Enhance [ ]
Threshold 0
Smooth 2
Object Array 20, 20
Profile Length 12
Profile Fraction 0.1
Tp 6
Tg 4
Frame 1

figure 2.8 Effect of varying the 2 main parameters in the model

Because, in the case of modelling arteries, the shape of the inner and outer walls varies little from circular, it is hard to see the variation in figure 2.8, so I also designed the program to plot the same 9 variants superimposed on each other:



View Filtering ☐

Histogram Equalise ☑ Edge Enhance ☐

Threshold 0

Smooth 2

Object Array 20, 20

Profile Length 12 Profile Fraction 0.1

Tp 6 Tg 4

Frame 1

figure 2.9 Effect of varying the 2 main parameters in the model

Applying the model

A separate program has been written, which applies an ASM to one or more images and then provides tools to analyse the outcome. There are 6 files containing the code, named ‘ApplyModel*.pro’, the compiled run-time program is called ‘ApplyModel.sav’.

In the program, the user is allowed to select any of the models that have been created. Selecting a model loads into memory the average point distributions and profiles and the matrices of eigen-vectors describing the modes of variation of these distributions. The user can also use a standard file selection dialog box to pick a selection of bitmap images (not necessarily continuous) from a directory. Never more than one bitmap is loaded into memory at one time, instead, all of the filenames are stored in an alphanumerically sorted list and only the bitmap currently on view is loaded.

If wished the user can select all of the images in the current directory at the press of a button. This is useful because it allows the user to select only a few frames from an image sequence to see how well the chosen model works and if the user is happy then it is easy to select the whole sequence and apply the model to every frame.

The user is given the choice of applying either the model created using unfiltered training images or the model created using filtered training images. If filtering is chosen, then the same filtering parameters are applied to current images as to the training images.

Before fitting the model to the images, the user must supply the program with two parameters, *Dmax* and *Mdist*, the relevance of which will be explained later.

Obtaining an initial estimate

The user can choose to apply the model to only the currently displayed image or to all images. In either case the model fitting procedure must be supplied with a guide to where the objective shapes are located. There are various global search techniques that could be employed to do this [19], but for two reasons I chose to let the user indicate the initial position. The first reason is simplicity of programming – the very little effort involved in indicating the approximate position outweighed the extra programming required. The second reason is that the jugular vein is located very close to the common carotid artery and may appear in the same image. An experienced ultrasonographer can easily distinguish the two, but this would place too high a demand on the global search algorithm. To indicate the starting point for the search, the user simply has to plot a diameter across the artery (one point on either side). The program estimates the position of the centre of the artery and its radius and applies these translation and scaling parameters to the vector $\underline{\bar{x}}$ to calculate the initial position, $\underline{X} = \underline{\bar{X}}$. If the model is to be fitted to multiple frames then this method is used for the first frame, but for subsequent frames the shape position calculated in the previous frame is used as the starting point (as the video sequences are captured at 25 fps, the frame-to-frame differences are small).

__Improving the model fit__

The current shape estimate $\underline{X}$ fits the shape model but may not be the best fit in terms of grey scale. For each point in $\underline{X}$, a search is made along the normal to the boundary to find the point at which the profile centred on it best fits the profile described by the model. Profiles are calculated for points in the image up to *Dmax* points in either direction from the current estimate (a total of 2**Dmax*+1 profiles are calculated for each point in the shape). Each of the profiles is of length *Tg* and is calculated in the same way as when creating the model. The profiles are then translated into 'model' space by:

$$\underline{b}_g = P_g^{\,T}(\underline{G}_{new} - \underline{\overline{G}}) \qquad \textit{equation 2.9}$$

A measure of how well each profile fits the model is given by *F*:

$$F = \sum_{j=1}^{Tg} \frac{b_{gj}^{\;2}}{\lambda_j} + \frac{2R^2}{\lambda_{Tg}} \qquad \textit{equation 2.10}$$

where $R^2 = \left(\underline{G}_{new} - \underline{\overline{G}}\right)^T \left(\underline{G}_{new} - \underline{\overline{G}}\right) - \underline{b}_g^{\,T}\underline{b}_g$ *equation 2.11*

The profile that gives the smallest value of *F* is the one which best fits the model, the distance of the centre point of this profile to the previous estimate for the point is d_{best}. A suggested movement of the model point, *dX*, is then calculated:

$|d\underline{X}| = 0$ if $|d_{best}| \leq 0.5$

$|d\underline{X}| = 0.5*d_{best}$ if $0.5 < |d_{best}| \leq \mathrm{d}_{\max}$ *equation 2.12*

It may not be possible to move the shape to $\underline{X}+d\underline{X}$, as may not be within the bounds of the model. We therefore try to find the closest point to this in 'model' space, but have to get there via 'normalised' space. From equation 2.1:

$$s(1+ds)[\underline{x}+d\underline{x}] + (\underline{Xc}+d\underline{Xc}) = \underline{X} + d\underline{X} = \mathrm{s}\underline{x} + \underline{Xc} + d\underline{X} \qquad \textit{equation 2.13}$$

Rearrange to get:

$$d\underline{x} = \frac{1}{s(1+ds)}\left(s\underline{x} + d\underline{X} - d\underline{Xc}\right) \qquad \textit{equation 2.14}$$

Using equation 2.6, we find that

$$d\underline{b} \approx P_x^{\,T} d\underline{x} \qquad \textit{equation 2.15}$$

This gives a new value for $\underline{b}$: $\underline{b}' = \underline{b}+d\underline{b}$

The Mahalanobis distance, *Dm*, is a measure of how well the shape described by $\underline{b}'$ fits the model:

$$Dm^2 = \sum_{k=1}^{Tp} \frac{b_k^{\ 2}}{\lambda_k}$$ *equation 2.16*

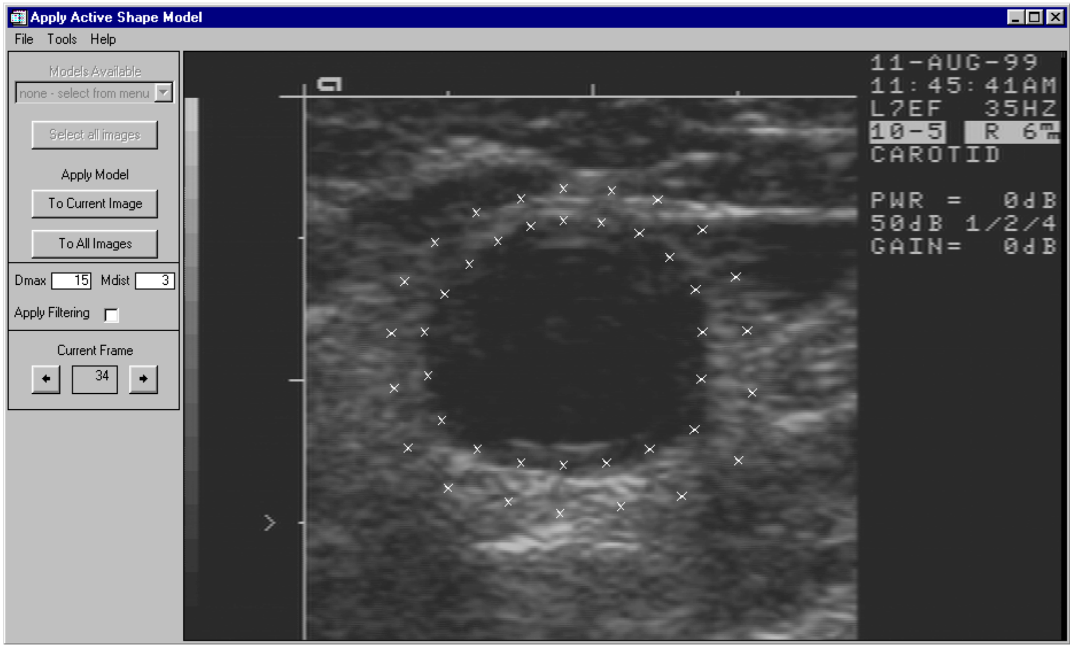
If the shape is a poor fit to the model, Dm>3.0, then $\underline{b}'$ is scaled:

$$\underline{b}'' = \underline{b}' \cdot \left(\frac{3.0}{Dm} \right)^{1/2}$$ *equation 2.17*

With a new, valid shape descriptor, $\underline{b}$, the scalar s and the vectors $\underline{x}$, $\underline{Xc}$ and $\underline{X}$ are updated.

If $Dm \geq Mdist$ (the user selected value), then the current shape estimate is not a good enough fit to the model and the process using the updated values. After 50 iterations, or if $Dm < Mdist$, the algorithm stops, with the program moving on to the next image if so desired. I chose to make the program stop after 50 iterations as it is unlikely that a good fit will be found if it has not already been found during the first 50 iterations.

One frame out of a sequence of 156 is shown in figure 2.10, with 20 model points fitted to both the inner and outer walls.



Dmax 15 Mdist 3

Current Frame 34

figure 2.10 Computed position of the model points in one frame

Analysis tools

Because the program allows the user to find the position of a number of objects as they move through a sequence of images, I have included a range of tools that display how parameters of the shapes change over time.

Calculating the image scale

The dimensions of the shapes located by the program are in pixels and therefore depend on the resolution at which the images were acquired. No numerical information is supplied with the images as to the scaling, but the horizontal and vertical tick marks on the axes (shown in figure 2.10) are known to be 5mm apart. I have created a tool in the program that lets the user covert the scaling from pixels to millimetres. The user is prompted to click the left mouse button on two adjacent tick marks and from this the program calculates the scaling factor to convert from pixels to millimetres. The array containing the shape location in each frame is not changed, so if the user is not happy with the scaling, it can simply be re-calculated. If the scaling has been set, the graphs displayed by the other tools will convert dimensions to millimetres (as in figure 2.12), otherwise the dimensions will be displayed in pixels (as in figure 2.11).

Cross-sectional area and diameter

Previous ultrasound studies of the common carotid artery have measured the diameter of the artery from the near wall to the far wall. To enable a comparison of the performance of this program with these studies, I produced a tool that calculates the diameter of the artery in the vertical direction in the images. For each object that has been located, the program simply finds the maximum and minimum Y coordinates for each frame and plots a graph of the difference – shown in figure 2.11.

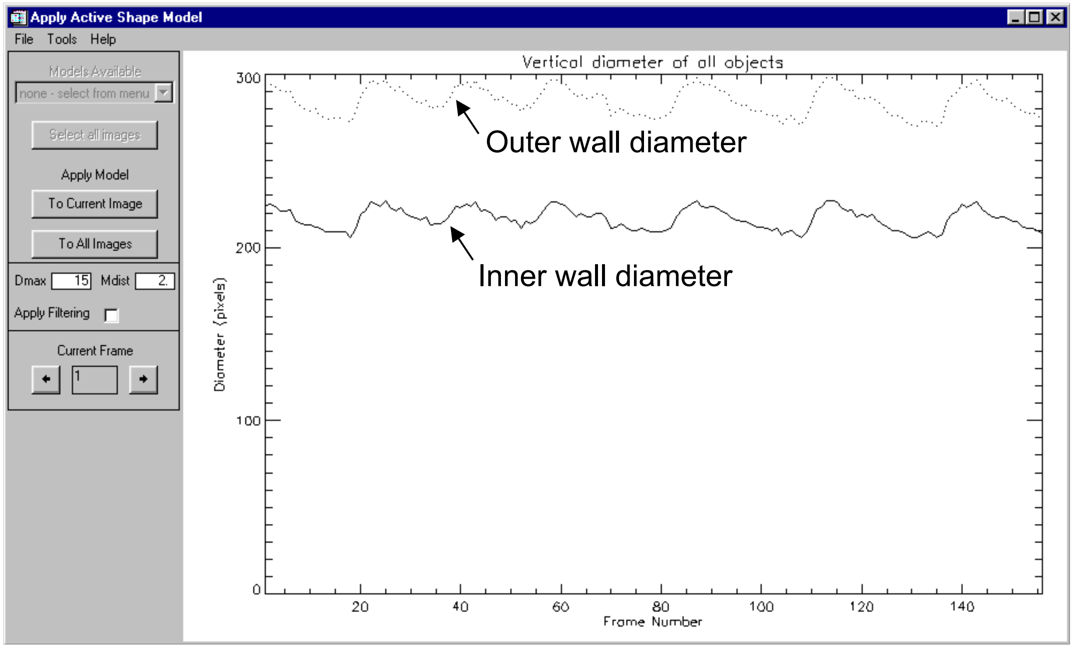


figure 2.11 Vertical diameter of the artery over time

I also produced a similar tool that plots the horizontal diameter over time and a tool that plots the maximum diameter over time. To plot the maximum diameter, the program finds the two points that are furthest apart for each object in the current image and then finds the distance between these points in all images. The resulting graphs may well differ, depending on the frame that was being viewed when the maximum diameter tool was applied (the maximum diameter may be in the vertical direction in one frame and in the horizontal direction in the next).

A more useful, and significantly less error prone, tool is one that calculates the cross-sectional area of the artery. It actually calculates the area of the polygon described by each object. There will be a small difference between the area of the polygon and the actual area of the artery due to the finite number of points in the polygon – if 20 points are used, the difference in area between a circle and an inscribed duodecagon is less than 2%. In most cases this small error will not matter because it is consistent and the relative change in cross-sectional area is of more interest than the absolute measure. If the model contains two objects, the inner and outer walls, then the program will also calculate the difference between the areas – the area of the wall.

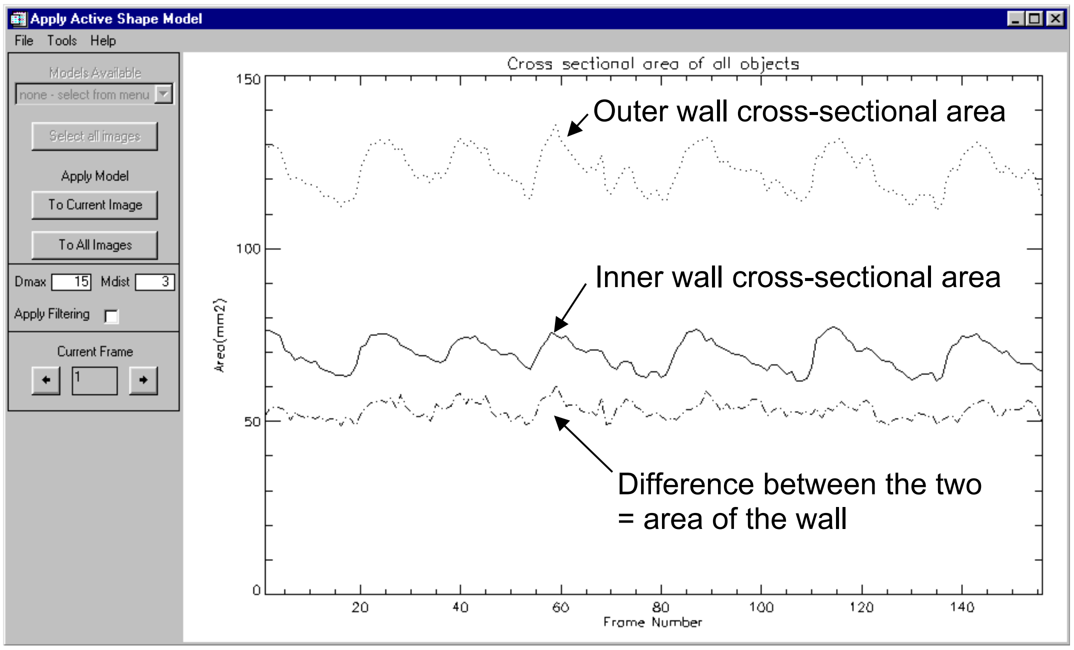


figure 2.12 Cross sectional area of the artery over time

Centre line

It may be of clinical use to know how the artery moves as a whole, relative to the transducer. I therefore produced a tool, which calculates the average X and Y coordinates for each object in each frame and plots these centre lines over time. As it can be difficult to interpret a 3-D line plot such as this, I have included projections of the plot that show movement in only the X or Y direction.

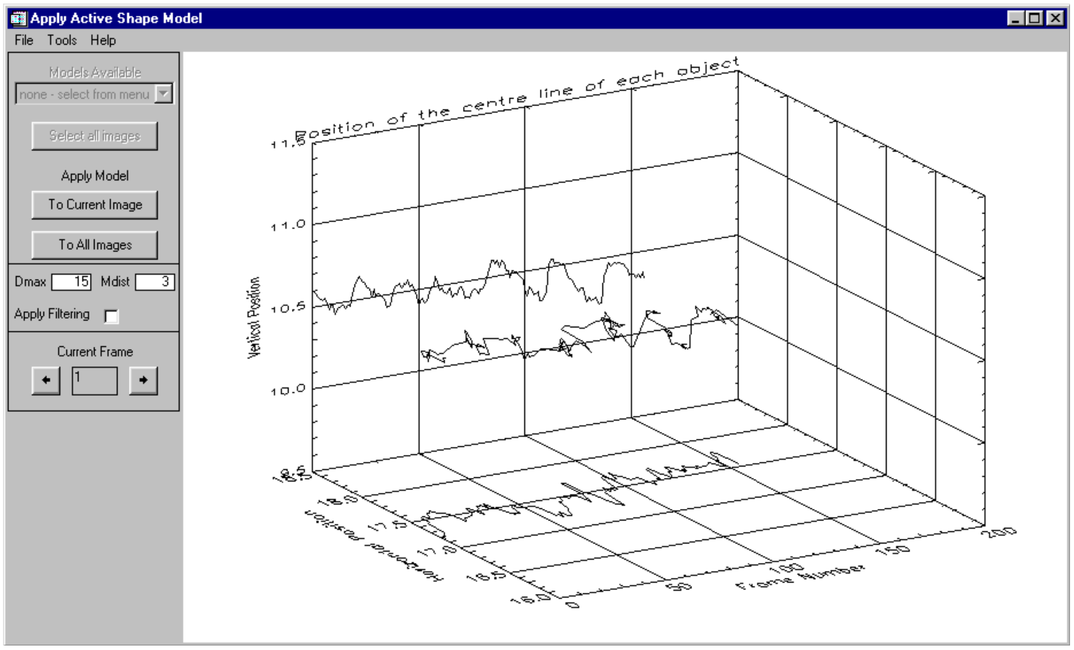
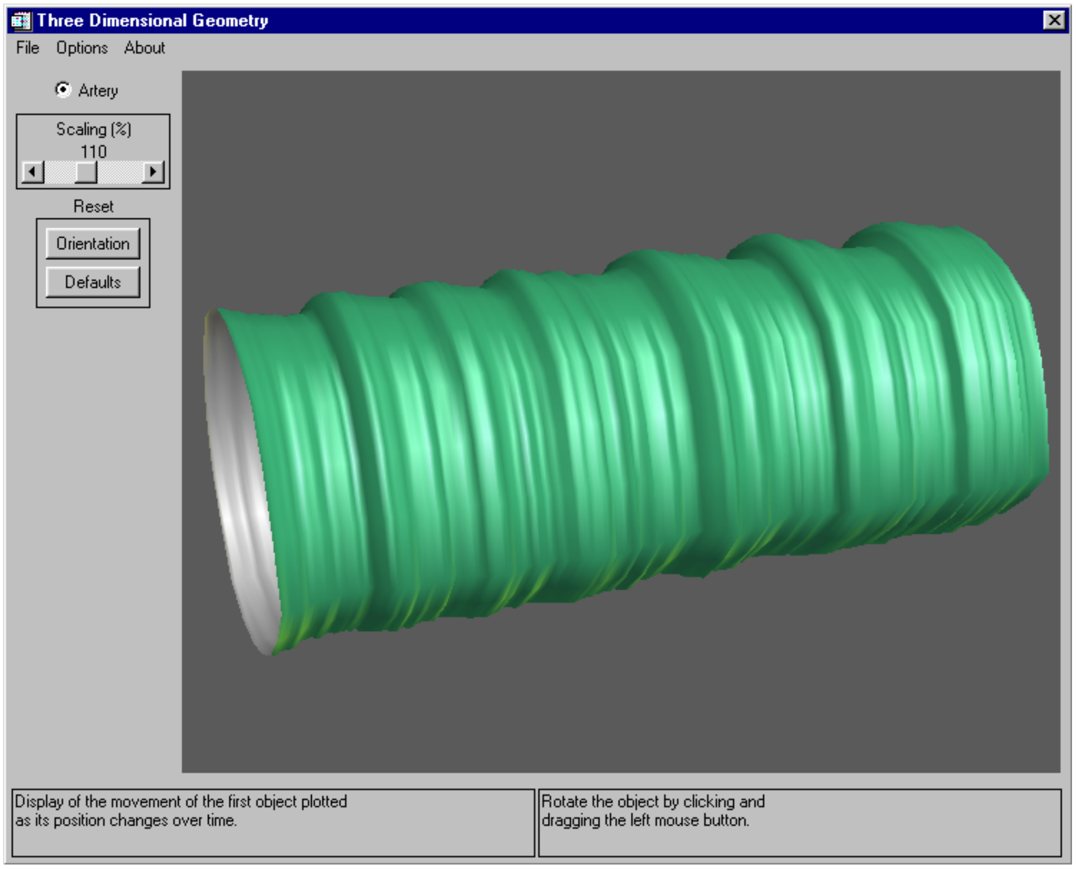


figure 2.13 Movement of the artery centre (projected onto x and y axes) over time

Artery wall position over time

The other tools display some of the parameters of the artery in a quantifiable manner, but it is also useful to see how the whole artery changes over time (not just the centre line or the diameter). Having seen that IDL is capable of producing excellent rendered images of 3D objects, and that these objects can be rotated by the user in real time, I decided to produce a tool that would display the inner wall of the artery. There was already a demo program, supplied with the IDL software that could read certain data files and then draw 3D objects for interactive viewing. I determined the format of the data required by this program and converted the data describing the inner artery wall into that format. One of the variables required was a list of (X,Y,Z) coordinates for all of the vertices, this simply required a rearrangement of the shape array. The other major variable required was a list of polygons that describe the surface. This is a list in the format (n, V_1, V_2, .., V_n), where $V_1..V_n$ are the indices of the vertices making up each n-sided polygon. When the tool is selected, the program calculates these variables and sends them to a modified version of the demo program (shown in figure 2.14).



Scaling (%) 110

Display of the movement of the first object plotted as its position changes over time.

Rotate the object by clicking and dragging the left mouse button.

figure 2.14 Rendered representation of the artery inner wall over time

Chapter 3: Results

Functionality of the programs

It was intended that the final set of computer programs produced would have the level of functionality such that they would be of use to ultrasonographers and vascular consultants.

Due to time constraints, it was only possible to demonstrate the programs to Mr Baguneid (Consulting Vascular Surgeon at Manchester Royal Infirmary), rather than to train him to use them. He was able to follow the method of the programs and thought that the tools would be of use in a variety of situations.

It was possible to spend more time with the Miss Bodill (Ultrasonographer at Manchester Royal Infirmary), training her to use the programs. She had no problems in using the programs, though many of the settings were chosen under my guidance, as the relevance of the parameters *Tp*, *Tg*, *Dmax* and *Mdist* required too deep an understanding of the method. When creating a shape model, using the images she had previously acquired, she found it easy to plot model points on the inner wall of the artery, and was happy to interpolate the boundary to plot points where the boundary was ill-defined. Even with her trained eye, she found it difficult to plot model points on the outer wall, as in most images it was unclear where the boundary lay.

Efficiency of the method

I have used the *CreateModel* program to model over 100 images, most containing 40 points (20 to model the inner wall and 20 to model the outer wall). With practice, the model points can be place on the training images in less than 3 seconds per point. Placing points at this rate, the points are positioned to within 1 pixel of the ideal position (when an ideal position can be determined). The points can be placed so easily because the user only has to perform a one-dimensional search for the ideal position – the position being constrained to lie on the highlighted guide line.

I plotted model points in a series of 20 images, each image from the same video sequence, but 6 frames apart (for more variability). I set *ObjectArray* = [20,20] and *ProfileLength* = 10. I saved the model data after plotting model points on 5, 7, 9, 12, 15, 17 and 20 images. For each of these sets of data, I set the program to calculate the model so that it would explain 99.9%, 99%, 98%, 97%, 96%, 95%, 90%, 85%, 80% and 75% of the variability in the positioning and greyscale. That is, I set 'Tp' and 'Tg' to the above values. It is clear that if more images are used to produce the model, then more eigen-vectors will be needed to explain the variability:

		Number of images modelled						
		5	7	9	12	15	17	20
		Number of eigen-vectors used (max. possible = 80)						
Percentage of variability explained	99.9 %	4	6	8	11	14	16	19
	99 %	4	6	8	11	14	15	18
	98 %	4	6	8	11	13	14	16
	97 %	4	6	8	10	12	13	15
	96 %	4	6	7	10	12	13	14
	95 %	4	6	7	9	11	12	14
	90 %	4	5	6	8	9	9	11
	85 %	3	4	5	6	7	8	8
	80 %	3	4	5	6	6	6	7
	75 %	3	3	4	5	5	5	5

Table 3.1 Number of eigen-vectors used to explain shape variation

		Number of images modelled						
		5	7	9	12	15	17	20
		Number of eigen-vectors used (max. possible = 21)						
Percentage of variability explained	99.9 %	4	6	8	10	12	13	15
	99 %	4	5	6	8	9	9	10
	98 %	4	5	5	6	7	7	8
	97 %	3	4	5	6	6	6	7
	96 %	3	4	4	5	5	6	6
	95 %	3	4	4	5	5	5	5
	90 %	2	3	3	3	4	4	4
	85 %	2	3	3	3	3	3	3
	80 %	2	2	2	2	2	2	3
	75 %	2	2	2	2	2	2	2

Table 3.2 Number of eigen-vectors used to explain grey-scale variation

Output from the programs

When trying to locate the artery walls in a new set of images, a new model can be created using a sample of those images; or a general model, that has been created earlier, can be applied. It is important for an ASM to contain the variability required to find all of the valid shapes in the image sequence, but if it contains too much variability, then invalid shapes may also be found.

Using a specialised ASM

The simplest way to create such an ASM that will work with a sequence of images is to use a representative sample of the images as the training set.

To demonstrate this, I used the model described above, based on 20 training images from the total sequence of 351 images (14 seconds), to find the inner artery wall in the whole sequence. The parameters used were *Dmax* = 15, *Mdist* = 3.0, filtering was not applied. In each frame, the program positioned the 40 points as well as could be done by hand (figure 3.1 shows a frame not included in the training set). The cross-sectional area calculated by the program changes smoothly from one frame to the next (figure 3.2). On a PC based on an Intel Celeron 300A with 64Mb RAM running Windows 95, the program took a total of 25 minutes (= 4.3 seconds per frame) and required an average of 48.8 iterations per frame (= 0.088 seconds per iteration). The program was run again with *Mdist* = 5.0 giving 14.1 iterations per frame (= 0.118 seconds per iteration. The difference in time per iteration can be explained by the time taken to load the image and initialise the variables, which is averaged of 14.1 iterations rather than 48.8 frames. These figures give timings of 0.075 seconds per iteration + 0.601 seconds initialisation time per iteration for this model.

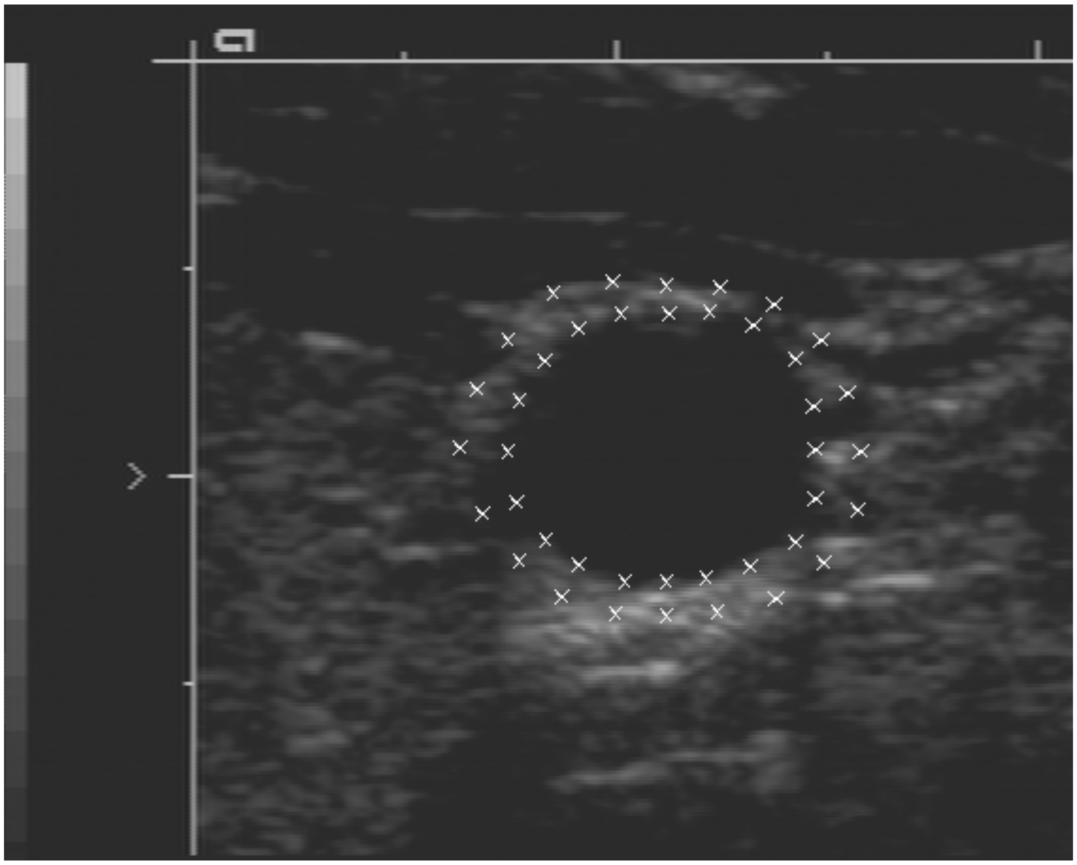


figure 3.1 Computed position of the model points in one frame

Points between which the diameter is maximised are indicated by the arrows

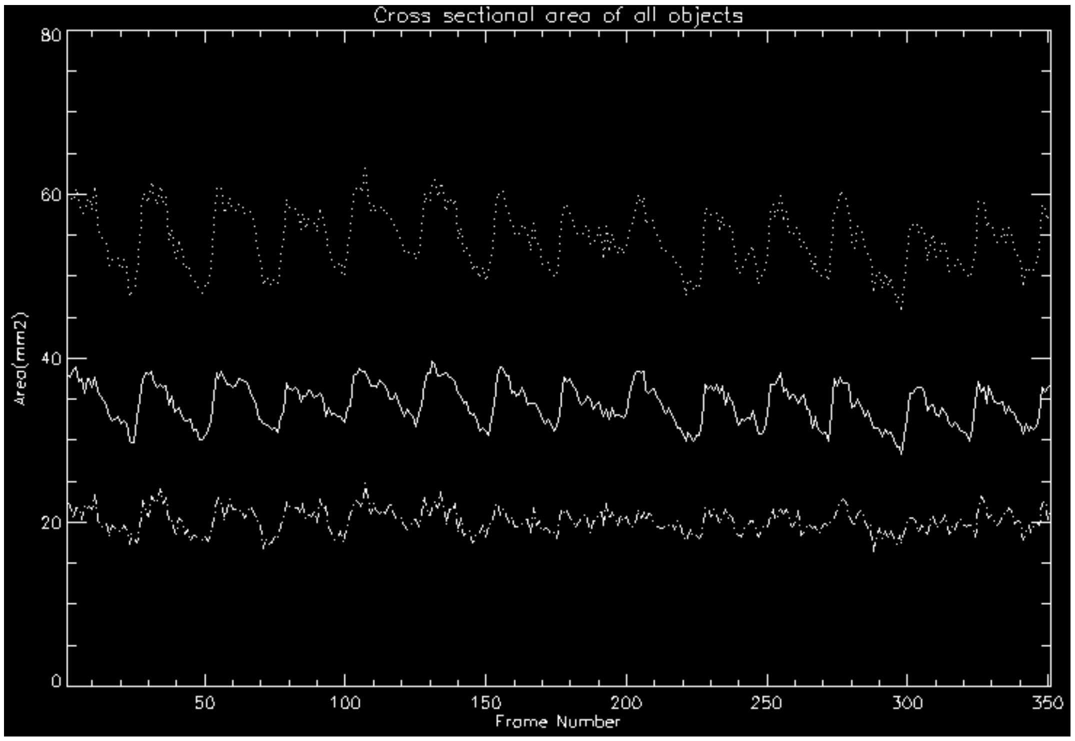


figure 3.2 Cross sectional area of the artery over time

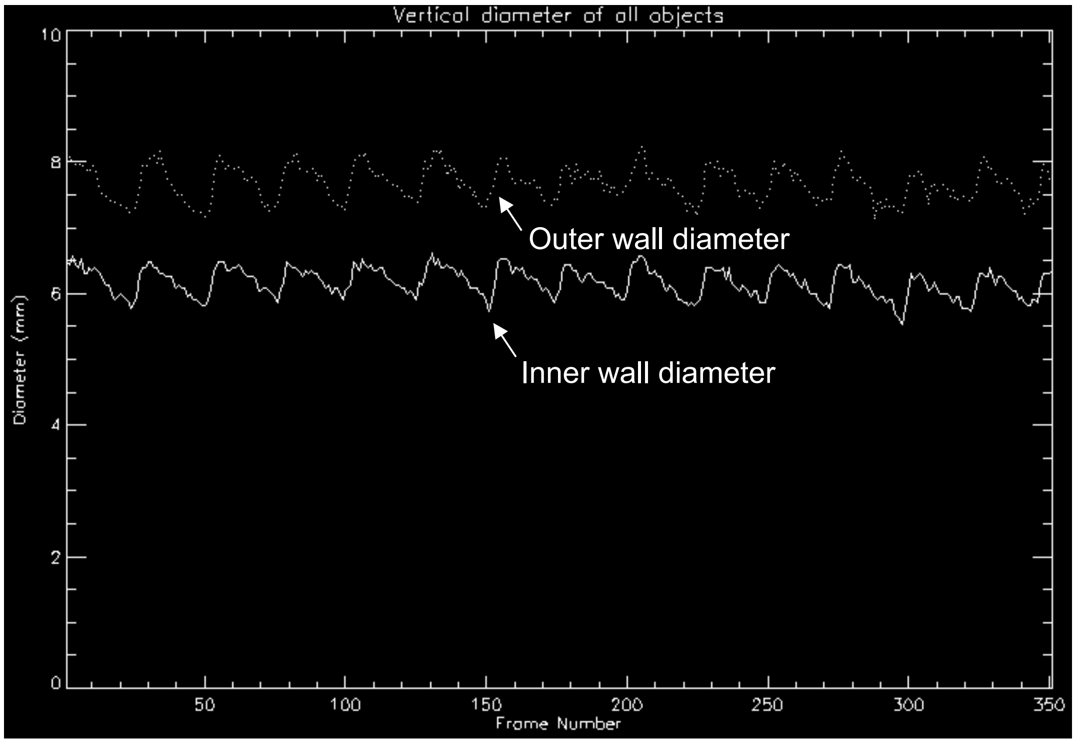


figure 3.3 Vertical diameter of the artery over time

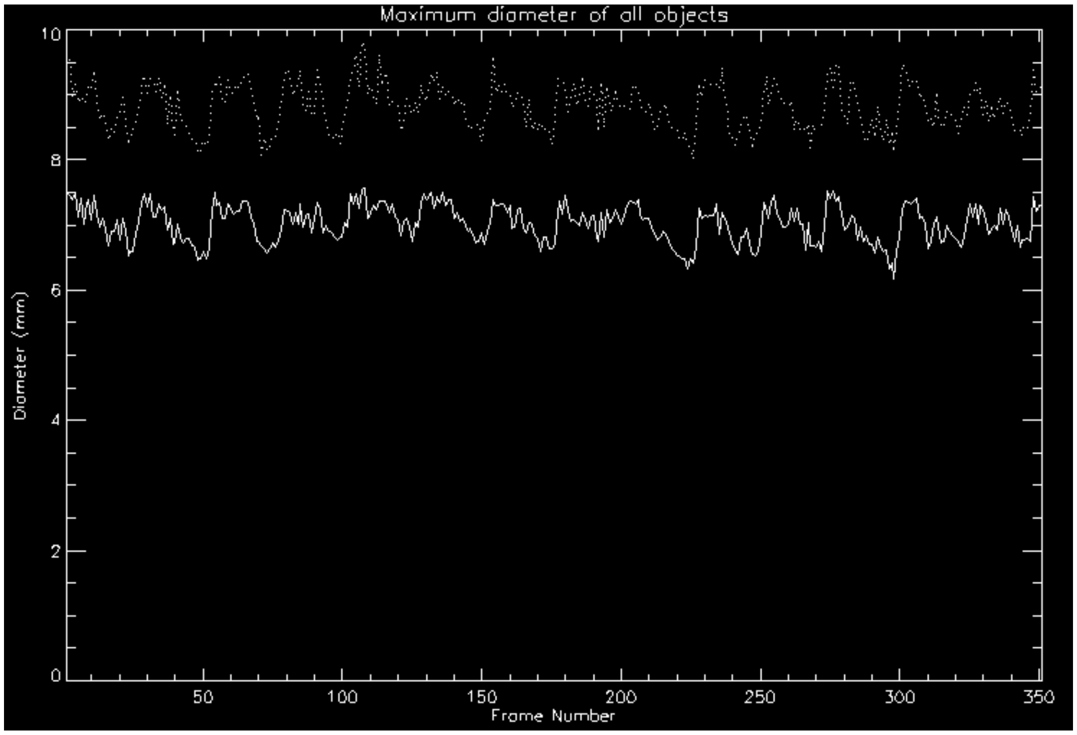


figure 3.4 Maximum diameter of the artery over time

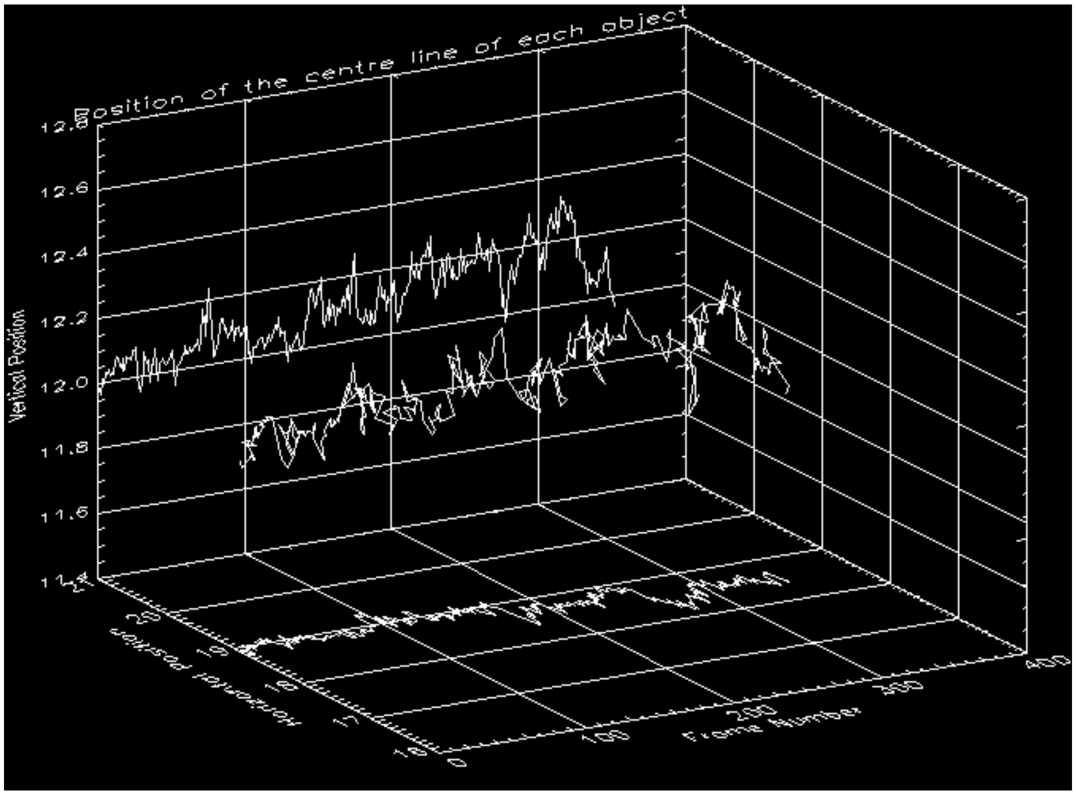


figure 3.5 Movement of the artery centre (projected onto x and y axes) over time

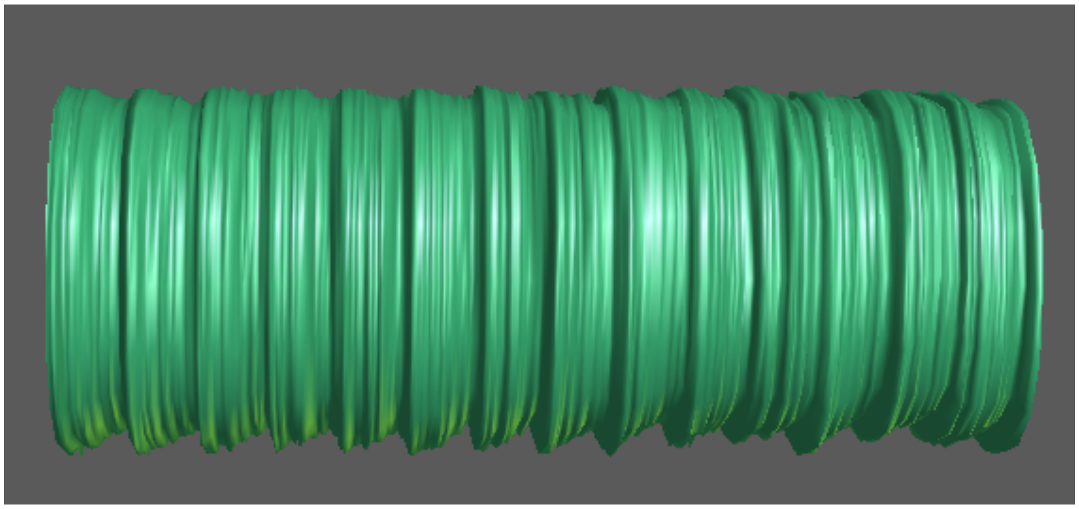


figure 3.6 Rendered representation of the artery inner wall over time

The distension waveform of the artery clearly follows the sawtooth pattern, described by Stadler et al. [12] (page 18), but this method also shows the way the artery distends in two dimensions (figure 3.6) and how the artery moves as a whole (figure 3.5). Using the interactive 3D view, and rotating the object, it is possible to get a good feel for shape being represented. In this example, when the artery is forced to expand it mainly expands downwards, away from the ultrasound transducer (perhaps because pressure from the transducer stops it moving upwards). The projection of the vertical motion of the centre-line in figure 3.5 shows this motion as an inverted sawtooth.

The maximum diameter measured (figure 3.4) does not show the sawtooth pattern as clearly as the vertical diameter (figure 3.3); the points that define the maximum diameter are also the points that lie on the least clearly defined sections of the wall and so positioning of the points will be less accurate.

In this case, the vertical diameter (determined from only 2 points) appears to follow the sawtooth pattern better than cross-sectional area (determined from 20 points). Although the vertical diameter is useful as a comparison with other studies, a more useful measure is the average diameter, derived from the cross-sectional area:

$$\textit{average diameter} = 2 \times \sqrt{\frac{\textit{area}}{\pi}} \qquad \textit{equation 3.1}$$

The cross-sectional area data for the inner artery wall was imported from the IDL application into the signal-processing package Santis. Using equation 3.1, the average diameter was calculated:

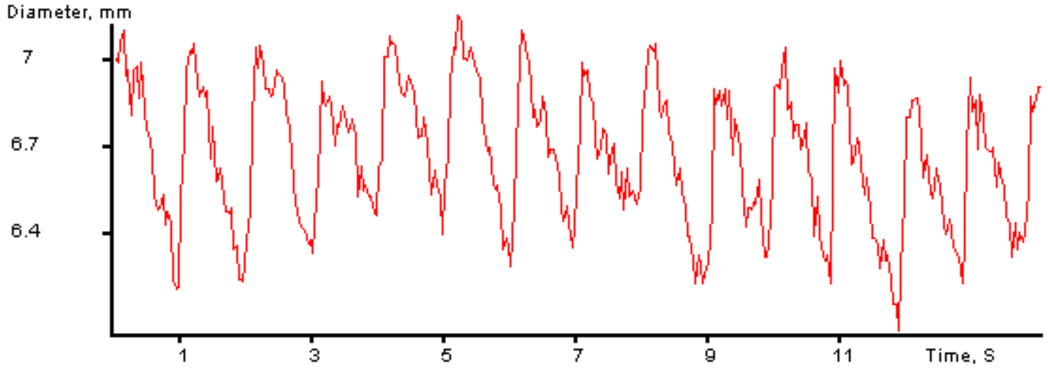


figure 3.7 Average diameter of the artery inner wall over time

This can then be normalised to remove DC offset and linear drift:

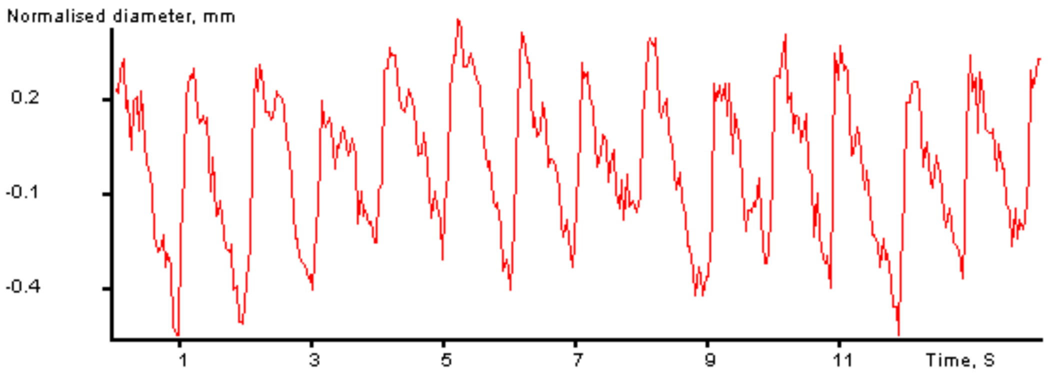


figure 3.8 Normalised average diameter of the artery inner wall over time

The frequency spectrum of the signal can then be calculated:

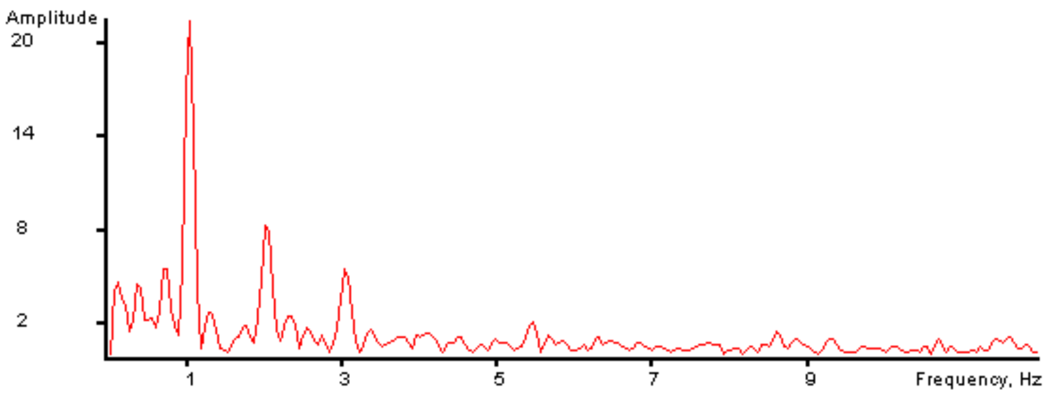


figure 3.9 Frequency spectrum of the artery inner wall diameter over time

The equation of a sawtooth wave is:

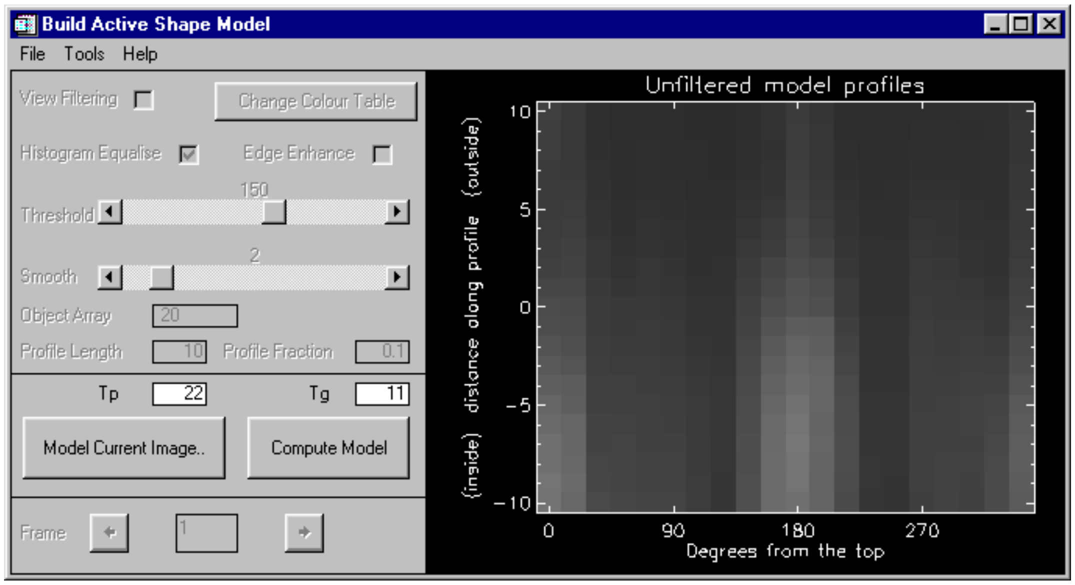
$$y = \sum_{k=1}^{\infty} \frac{\sin(kx)}{k}$$ *equation 3.2*

So if the fundamental frequency is 1Hz and has amplitude 1, then for a perfect sawtooth, the amplitude of the harmonic frequencies will be $\frac{1}{2}, \frac{1}{3}, \frac{1}{4}, \frac{1}{5}, \ldots$

The leading edge of the waveform cannot physically be vertical (the artery cannot expand infinitely quickly), but from the Fourier decomposition of the measured waveform, equation 3.2 is a good first approximation. Stadler et al. [12] described the waveform in greater detail.

Using a general ASM

Creating a new ASM for each set of images is very time consuming, so I tried to create a more generalised model, using a range of training images. The images used were from sequences showing 6 different common carotid arteries (5 people). From each sequence, 7 images were modelled, each 6 frames on from the last. I chose to model only the inner wall of the artery as the appearance of the outer wall varies greatly between image sequences.


Build Active Shape Model

View Filtering ☐
Histogram Equalise ✓
Edge Enhance ☐
Threshold 150
Smooth 2
Object Array 20
Profile Length 10
Profile Fraction 0.1
Tp 22
Tg 11
Frame 1

figure 3.10 Unfiltered profiles of the computed model

The values of the model parameters and the average grey-scale profiles can be seen in figure 3.10. The eigen-vectors selected explain 99% of the variability in shape and grey-scale profiles in the training images. The range of shapes created by varying the 2 most significant eigen-vectors is shown in figure 3.11.

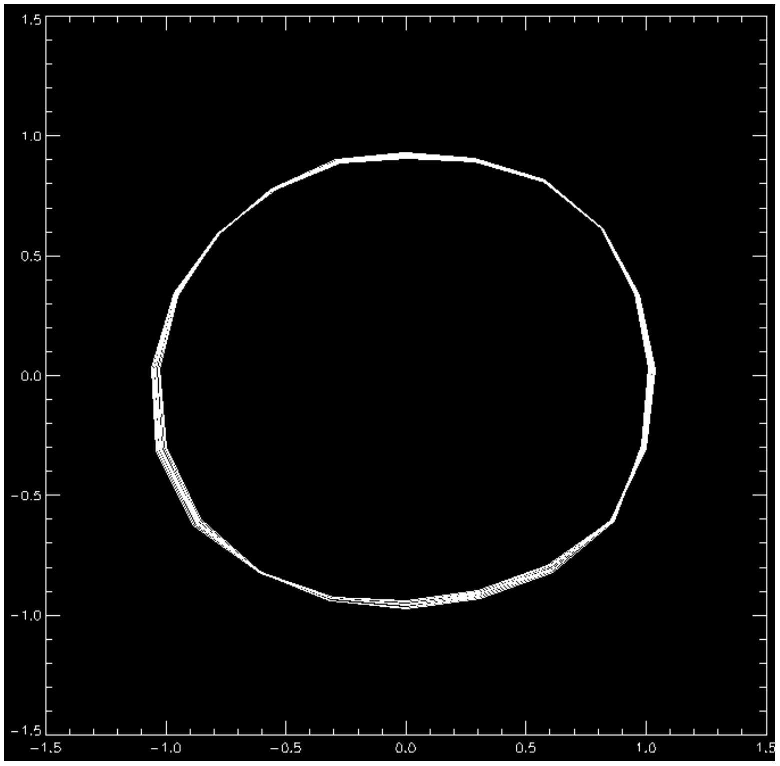


figure 3.11 Range of shapes determined by first 2 eigen-vectors

Having created this general model, there was no clear way to see how good it would be, except to try it on other images. Unfortunately this model failed rapidly, but a model based on the first 12 images did work. The following 8 figures show the result of applying this model to a sequence of 163 images of an artery that had not been used in creating the model. The values of the parameters were *Tp* = 11, *Tg* = 8, *Dmax* = 15, *Mdist* = 5.0 and filtering was not used. On the same PC as, the program took a total of 3 minutes 10 seconds (= 1.2 seconds per frame) and required an average of 14.2 iterations per frame (= 0.082 seconds per iteration). Applying the same model to the same images with *Mdist* = 4.0, 50 iterations were taken per frame (= 0.059 seconds per frame). These figures give timings of 0.050 seconds per iteration + 0.450 seconds per iteration initialisation for this model.

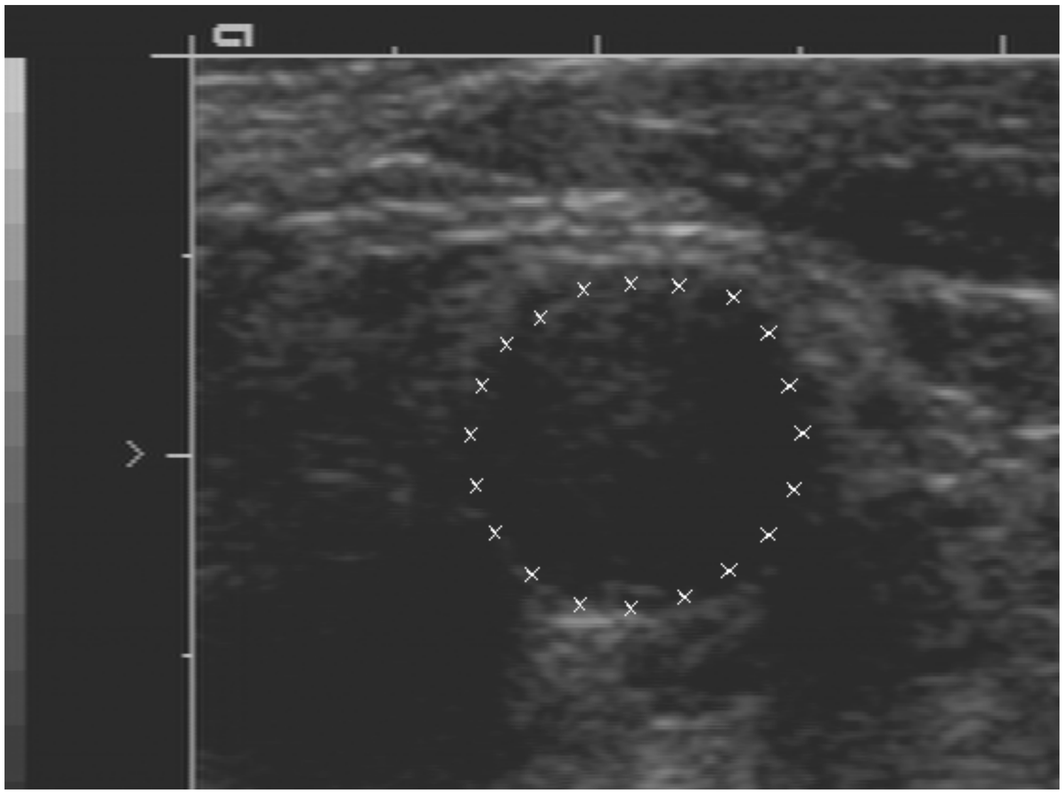


figure 3.12 Computed position of the model points in one frame

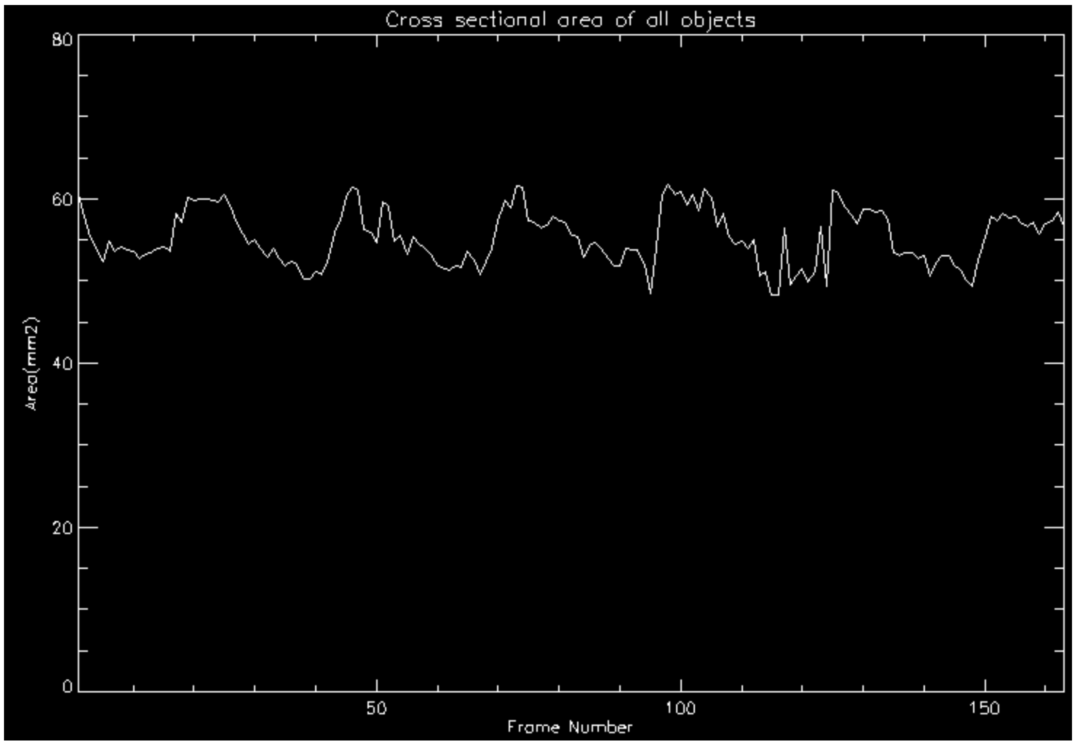


figure 3.13 Cross sectional area of the artery over time

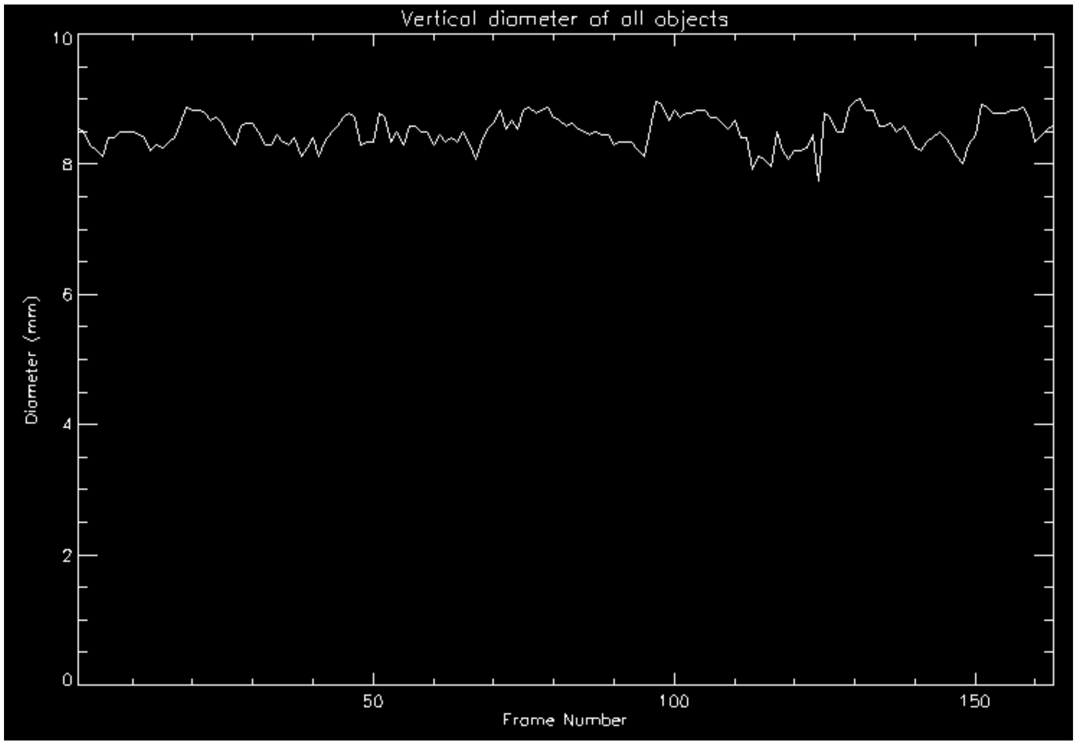


figure 3.14 Vertical diameter of the artery over time

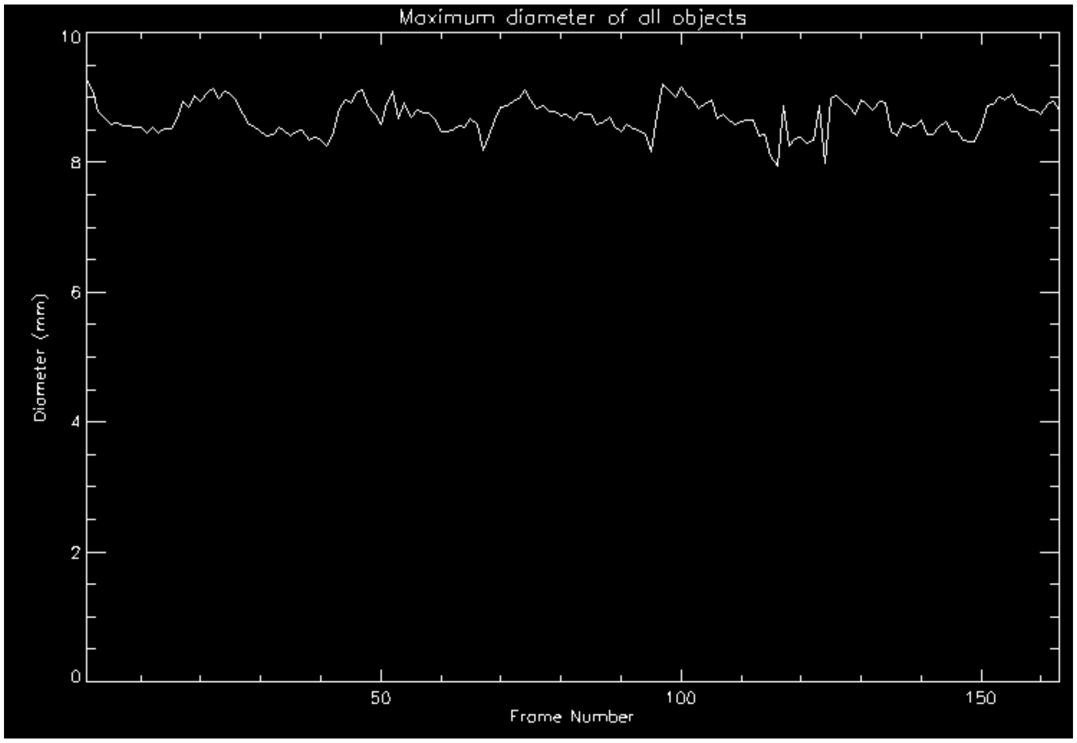


figure 3.15 Maximum diameter of the artery over time

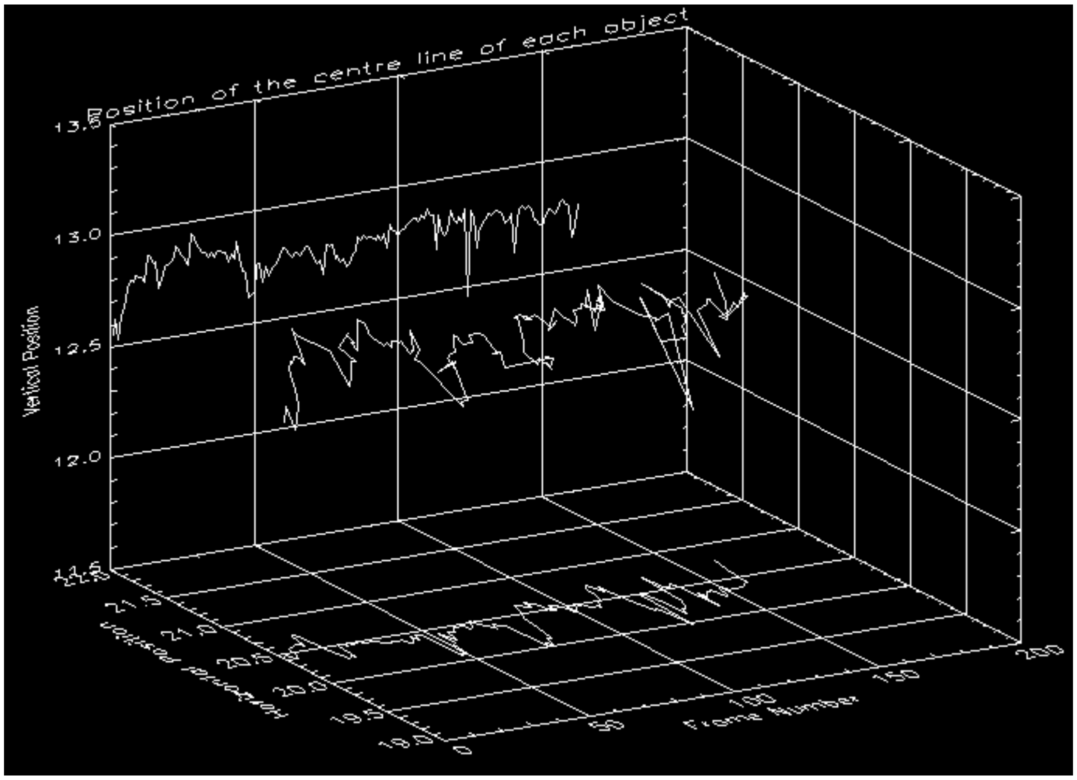


figure 3.16 Movement of the artery centre (projected onto x and y axes) over time

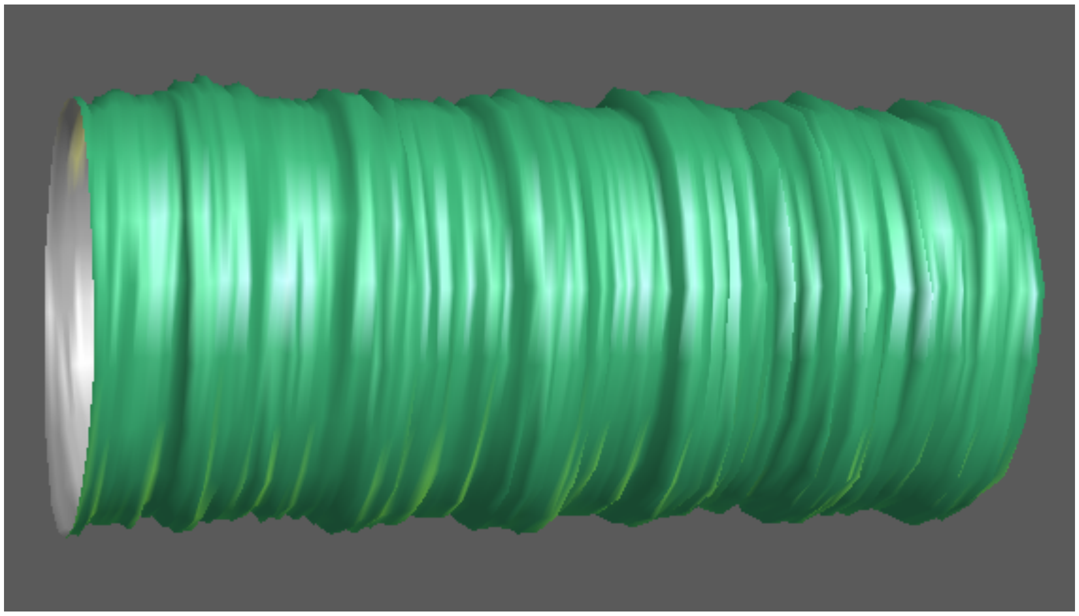


figure 3.17 Rendered representation of the artery inner wall over time

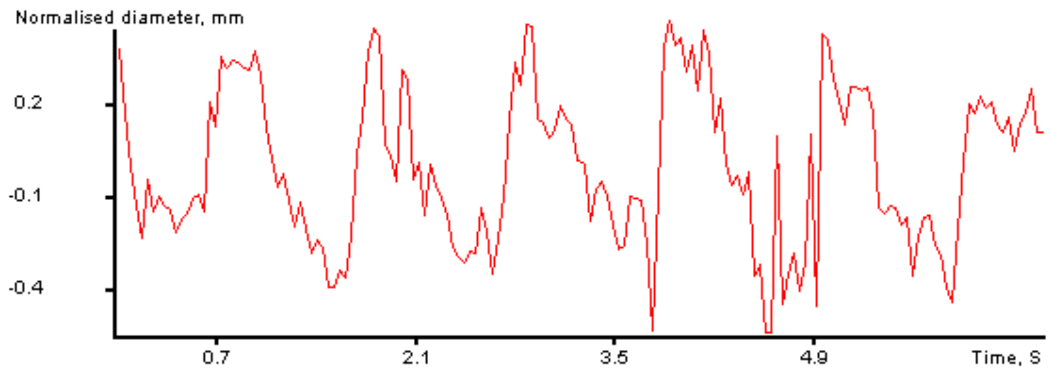


figure 3.18 Normalised average diameter of the artery inner wall over time

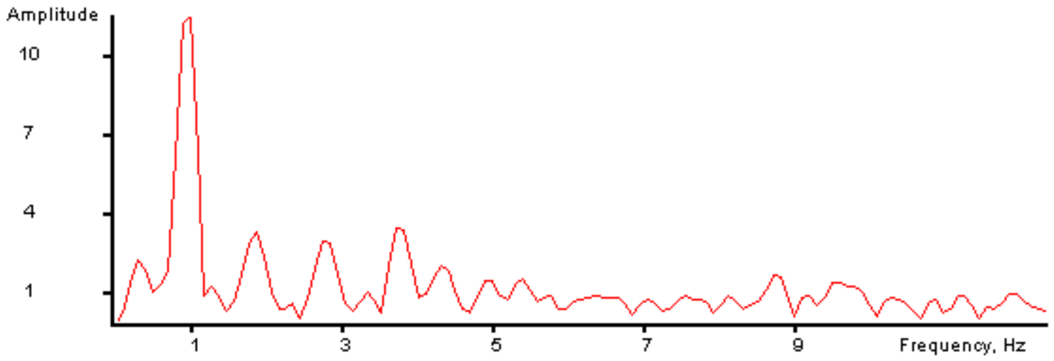


figure 3.19 Frequency spectrum of the artery inner wall diameter over time

It can be assumed that the motion of the artery is not as erratic as figure 3.18 suggests. It is clear that the general motion of the artery has been followed, as the sawtooth pattern is still evident, but the generalised model does not find the inner wall of the artery as accurately as a specialised model.

In my initial attempt at using this model with the same image sequence, I selected *Mdist*=3.0. The program was able to find the artery wall for approximately 100 images before failing. The way in which the program failed is shown in figure 3.20. I have applied many models to many image sequences and, in every case where the program failed, it failed in this way. Occasionally the program would place the model points further out than the boundary and further out than it had done in the majority of frames, but it never failed completely by expanding the shape to the edge of the image.

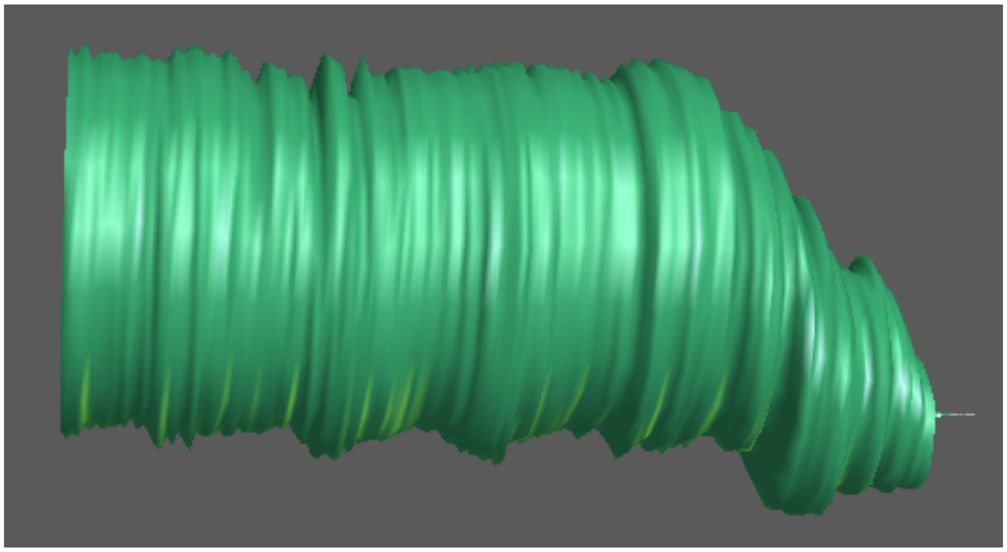


figure 3.20 Failure of the program

Comparison of results with other studies

For the waveform shown in figure 3.7 the relative diameter change was 10.8 ± 2.0%, which agrees with published findings [13] for a person of the same age and sex (9.6 ± 2.4%).

For the waveform shown in figure 3.18 the relative diameter change was 10.0 ± 1.5%, but the age and sex of the patient were not recorded, so a comparison cannot be made.

Chapter 4: Conclusion

Two computer programs have been created. The first can read in a sequence of bitmap images and allows the user to create an active shape model, which models the shape and grey-scale variation of objects in the images. The second program can apply an active shape model to a new sequence of images and locate examples of the modelled shape, tracking its movement from one image to the next.

Image sequences were taken by an ultrasonographer of the transverse section through both the left and right common carotid arteries of patients. The same person ran the programs, creating an active shape model of the inner wall of the artery and then applied the model to other images. Once the artery wall had been detected, tools within the program were used to display the motion of the artery. This practical application of the method showed it to be accurate and user friendly. The programs were considered by a Consulting Vascular Surgeon to have valuable practical applications, especially if measurements of blood pressure were included to enable the calculation of arterial compliance. The three-dimensional display of arterial motion (2D+T) is a novel (to my knowledge) and valuable method.

It was clear to see when a model gave a good fit to the images, as when the fit was poor, the outline found would collapse to a point. The data from the program showed that when a model does fit the images, the diameter, diameter change and distension waveform measured agree with previous studies.

It would be nice to quantify the accuracy of the method. Larger studies applying computer vision techniques to medical images have been able to compare the shapes found by their programs with the shapes found by human experts. Unfortunately this is very time consuming and was beyond the scope of this project.

Chapter 5: Discussion

Basic improvements to the method

In the *CreateModel* program, the user is able to select image filtering parameters. When the model is calculated, two models are actually calculated, one of the unfiltered images and one of the filtered images. In the *ApplyModel* program the user is then able to choose which of these models to use. The model of filtered images rarely performed better and often performed worse than the model of unfiltered images, even though the filtering parameters were selected to enhance the artery walls. It would be worthwhile producing a standardised method of filtering. The user would then be able to 'mix' models together to produce a larger, more generalised model (the *ObjectArray*, *ProfileLength* and *ProfileFraction* would need to be the same).

The reasons for the method failing (as in figure 3.20) should be investigated further. As the method only fails by placing the shape inside the true boundary, an improvement may be made by changing the initial estimate for each frame. Currently the initial estimate is the position of the final estimate in the previous frame. This could be expanded by say 5 pixels and then the program would 'relax' the shape back to the boundary.

Applications of the method

Calculating arterial compliance

As discussed in the introduction, elasticity (or compliance) is an important property of the common carotid artery (and other major arteries). There are numerous measures of elasticity, but all of them require maximum and minimum measurements of artery diameter (inner, outer or midwall). Blood pressure measurements are also required, for which there are non-invasive techniques, described in Chapter 1. This method can be used to calculate the maximum and minimum average diameter.

It may be possible to extend the method to calculate the compliance of segments of the artery wall, which would be of interest because sections of the artery wall in which plaque is developing would be expected to be less compliant than healthy sections.

Matching graft compliance with artery compliance

One of the most valuable and reliable applications of this program would be in matching the compliance of intra-arterial grafts with the artery. If a section of the artery has weakened, a graft can be placed inside to strengthen it. Normally the artery will expand and contract under the varying pressure of the blood. It is important for the graft to have the same elasticity as the rest of the artery. If it is more elastic it will not provide support, if it is less elastic it will loosen from the artery and blood will be able to get behind it. The compliance of the graft can be measured mechanically and the compliance of the artery can be calculated using the method described earlier.

Producing a volume image of the artery

Three-dimensional ultrasound imaging is made possible by attaching a positioning device to an ultrasound transducer. With this, the position in space of each image slice is known. Gill et al [23] have used this equipment to produce a volume image of the carotid artery, which they segmented using a simple balloon model, with an equation of motion constraining the position of each vertex.

Three-dimensional ultrasound equipment is very specialised, and most hospital ultrasound departments do not have access to it. I believe it would be possible to produce a three dimensional image of linear structures, such as the common carotid artery, with only the equipment and programs I have used here. A skilled ultrasonographer can move the ultrasound transducer along the length of the common carotid artery at a constant speed, whilst maintaining it at a constant angle. A sequence of images can therefore be captured, which approximates a volume image; though it won't represent a single point in time. To be able to visualise three-dimensional objects in a volume image, they need to be segmented. Because the images are spatially almost continuous, shapes found in one frame will be located nearby in the next frame, just as with a time series of images fixed in a single plain. If an active shape model is created from training images that cover the full range of variability along the length of the artery, then it can be applied to such a sequence of images, and will detect the position of the artery walls along the length scanned. The three dimensional rendered image, produced by the *ApplyModel* program, will then show distance as well as time along the Z axis. The length of the artery in such an image will appear to pulsate because of the time factor in acquiring the volume image, but this would also occur in normal 3D imaging. By this technique, parts of the length of the artery may appear to be compressed and other parts may appear to be stretched (if the scanning is not smooth enough), but there is a solution to this: Viéville et al. [24] describe the method of auto-calibration of just such a sequence of images. This method requires a continuous surface through the images, which we have, from which it is able to determine how the position of one frame is related to those either side of it. A more accurate volume image can then be built up as each pixel in the sequence of frames can be related to a voxel in the volume image.

Calculation of percentage stenosis

In the majority of ultrasound investigations of the carotid artery, the degree of stenosis (narrowing of the artery) is noted at a number of positions along the length of the artery. It may be possible to use this program as the basis of an automated system for calculating the degree of stenosis. The valid shapes describing the lumen are roughly circular, according to the ASM. If some of the inner artery wall cannot be seen because a plaque is extending into the artery, the program will not follow the outline of the plaque, because that would create an invalid shape, but will place the model points approximately where they boundary should be. If too much of the true boundary is obscured, the program will not be able to fit a shape to it. For example if the artery is evenly coated with plaque, the program will not be able to find the wall, but may find the plaque-lumen interface instead or may simply fail. In such a case, the outer wall could be searched for and the inner wall estimated to be a given fraction in from it. Once the inner wall has been detected, it should be possible to segment the plaque by thresholding and then calculate the percentage stenosis.

Other computational methods

An improvement to the boundary detection method would be to treat the sequence of images as a volume, rather than discrete frames. In the same way that an ASM defines the relationship between different points in space, an ASM could also define the constraints on a point as it moves through time, but to do this a large number of frames would need to be modelled. Rather than training the model with the behaviour over time, equation 3.2 or the more advanced equations described Stadler et al. [12] could be used. Jacob et al. [25] have applied such a method, which is particularly good at tracking objects with periodic movement. Their application was to echocardiographic sequences.

References

[1] Ganong WF: Review of Medical Physiology (18th Edition). Appleton & Lange. 1997.

[2] Martini R: Fundamentals of Anatomy and Physiology (4th Edition). Prentice Hall. 1999.

[3] Flowers BH, Mendoza A: Properties of Matter. John Wiley & Sons Ltd. 1978.

[4] Parati G, Casadei R, Groppelli A, Di Renzo M, Mancia G: Comparison of finger and intra-arterial blood pressure monitoring at rest and during laboratory testing. Hypertension 1989; 13 (6 Pt 1): 647-655.

[5] Salomaa V, Riley W, Kark JD, Nardo C, Folsom AR: Non-insulin dependent diabetes mellitus and fasting glucose concentrations are associated with arterial stiffness indexes: The ARIC study. Circulation 1995; 91: 1432-1443.

[6] Bella JN, Roman MJ, Pini R, Schwartz JE, Pickering TG, Devereux RB: Assessment of arterial compliance by carotid midwall strain-stress relation in normotensive adults. Hypertension. 1999 March; 33(3): 787-792.

[7] Wada T, Kodaira K, Fujishiro K, Maie K, Tsukiyama E, Fukumoto T, Uchida T, Yamazaki S: Correlation of Ultrasound Measured Common Carotid Artery Stiffness with Pathological Findings. Arteriosclerosis and Thrombosis 1994 March; 14(3): 479-482.

[8] Riley WA, Evans GW, Sharrett AR, Burke GL, Barnes RW: Variation of common carotid artery elasticity with intimal-medial thickness: the ARIC Study. Atherosclerosis Risk in Communities. Ultrasound-Med-Biol. 1997; 23(2): 157-164.

[9] Riley WA, Barnes RW, Evans GW, Burke GL: Ultrasonic Measurement of the Elastic Modulus of the Common Carotid Artery. Stroke. 1992; Vol 23: 952-956.

[10] Hayashi K, Handa H, Nagasawa S, Okumura A, Moritake K: Stiffness and Elastic Behaviour of Intracranial and Extracranial Arteries. J. Biomechanics. 1980; Vol 13: 175-184.

[11] Riley WA, Barnes RW, Bond MG, Evans GW, Chambles LE, Heiss G: High-Resolution B-Mode Ultrasound Reading Methods in the Atherosclerosis Risk in Communities (ARIC) Cohort. J. Neuroimaging. 1991; Vol 1: 168-172.

[12] Stadler RW, Taylor JA, Lees RS: Comparison of B-mode, M-mode and echo-tracking methods for measurement of the arterial distension waveform. Ultrasound-Med-Biol. 1997; 23(6): 879-887.

[13] Hansen F, Mangell P, Sonesson B, Lanne T: Diameter and compliance in the human common carotid artery - variations with age and sex. Ultrasound-Med-Biol. 1995; 21(1): 1-9.

[14] Kass M, Witkin A, Terzopoulos D: Snakes: active contour models. Proc. 1st Int. Conf. on Comput. Vision. IEEE Press, New York. 1987; 259-268.

[15] Finet G, Maurincomme E, Reiber JH, Savalle L, Magnin I, Beaune J: Evaluation of an automatic intraluminal edge detection technique for intravascular ultrasound images. Jpn-Circ-J. 1998 Feb; 62(2): 115-21.

[16] Giachetti A: On-line analysis of echocardiographic image sequences. Med-Image-Anal. 1998 Sep; 2(3): 261-84.

[17] Chen C-M, Horng-Shing Lu H, Lin Y-C: A new ultrasound image segmentation algorithm based on an early vision model and discrete snake model. SPIE conference 1998.

[18] Sonka M, Hlavac V, Boyle R: Image Processing, Analysis and Machine Vision. Brookes/Cole Publishing Company, California, 1999.

[19] Cootes T, Hill A, Taylor C, Haslam J: Use of active shape models for locating structures in medical images. Imaging and Computer Vision. 1994 July; 12(6): 355-365.

[20] Cootes TF, Taylor CJ, Cooper DH, Graham J: Active Shape Models – Their Training and Application. Computer Vision and Image Understanding. 1995 January; 61(1): 38-59.

[21] Fish P: Physics and Instrumentation of Diagnostic Medical Ultrasound. John Wiley & Sons. 1990.

[22] Accuson 128XP: General Imaging and Shared Services Applications User Manual.

[23] Gill JD, Ladak HM, Steinman DA, Fenster A: Development and evaluation of a semi-automatic 3D segmentation technique of the carotid arteries from 3D ultrasound images. SPIE conference 1999.

[24] Viéville T, Lingrand D, Dourthe O: Auto-calibration of Echographic Slide Sequences. From the Internet. 1996.

[25] Jacob G, Noble JA, Mulet-Parada M, Blake A: Evaluating a robust contour tracker on echocardiographic sequences. Medical Image Analysis. 1999; 3(1): 65-75.