UNIVERSITY OF SOUTHAMPTON

Image Processing in Echography and MRI

by

Bernardo S. Carmo

A thesis submitted in partial fulfillment for the degree of Doctor of Philosophy

in the
Faculty of Engineering, Science and Mathematics
School of Electronics and Computer Science

April 2005

UNIVERSITY OF SOUTHAMPTON

ABSTRACT

FACULTY OF ENGINEERING, SCIENCE AND MATHEMATICS
SCHOOL OF ELECTRONICS AND COMPUTER SCIENCE

Doctor of Philosophy

by Bernardo S. Carmo

This work deals with image processing for three medical imaging applications: speckle detection in 3D ultrasound, left ventricle detection in cardiac magnetic resonance imaging (MRI) and flow feature visualisation in velocity MRI.

For speckle detection, a learning from data approach was taken using pattern recognition principles and low-level image features, including signal-to-noise ratio, co-occurrence matrix, asymmetric second moment, homodyned k-distribution and a proposed specklet detector. For left ventricle detection, template matching was used. For vortex detection, a data processing framework is presented that consists of three main steps: restoration, abstraction and tracking. This thesis addresses the first two problems, implementing restoration with a total variation first order Lagrangian method, and abstraction with clustering and local linear expansion.

Contents

Acknowledgements

I would like to thank my supervisor, Dr. Adam Prügel-Bennett. This work was supported by Portuguese FCT grant BD/1318 and a UK EPSRC grant. Additional guidance by Dr. Andrew Gee and Dr. Richard Prager, University of Cambridge, and by Prof. G. Z. Yang and Y. H. P. Ng of Imperial College, London. I would also like to thank Dr. Steve Gunn, William Benfold, Harriet Osborn and my parents for their help and support.

Chapter 1

Introduction

A considerable amount of knowledge about human anatomy has been accumulated throughout the centuries through the application of invasive methods. Today, this type of procedure still has to be performed on patients for diagnostic purposes, for example biopsies to determine the malignancy of a tumour, or open-brain surgical attachment of electrodes for locating sources of epileptic seizures. Research is underway to replace these and other methods with non-invasive techniques, which do not require direct access to the body's interior.

1.1 Medical Imaging

Medical imaging consists of a set of technologies that enable physicians to "see" inside the human body [40, 7]. Current imaging modalities used in clinical practice include X-ray films and computerised tomography (CT), radioisotope imaging, magnetic resonance imaging (MRI) including velocity MRI and functional MRI, single-photon emission tomography (SPECT), magnetic source imaging, surface light scanning and ultrasound (US), including two-dimensional frames from array probes (B-scans) and intra-vascular probes (IVUS) [262]. General 3D medical imaging has been researched since the 1970s and used clinically for about 15 years [242]. Historically, the first applications of 3D medical imaging consisted of imaging bones in CT, soft-tissue in MRI, and interventional and surgery assistance. The objects of interest in 3D medical imaging may be rigid (e.g., bones), deformable (e.g., soft-tissue structures), static (e.g., skull), or dynamic (e.g. heart, joints). In many applications, the analysis may focus on several objects. For example, an MRI 3D study of a patient's head may focus on three 3D objects: the white matter, the gray matter, and the cerebrospinal fluid. The information sought via 3D medical imaging may be qualitative or quantitative in nature. In qualitative analysis, mostly visual data is required, and the predominant issues are how to extract and display such information for human visualisation. In quantitative studies, we seek to measure or estimate parameters about the objects' morphology or function. In many applications, extensive user involvement is required, for example in the form of manual delineation of organ contours in tomographic images.

3D medical image processing and analysis systems generally consist of the following modules: a) pre-processing, including restricting the volume of interest for efficient data storage, filtering to supress or enhance information, interpolation to modify the level of discretisation, registration to combine information from multiple sources or viewpoints, and segmentation to assign image points to anatomic features; b) visualisation, which can be slice-based if the given scene is broken up into 2D slices that are displayed, or volume-based if the 3D object information in the scene is directly rendered using various techniques; c) manipulation, used to simulate surgery procedures and to develop aids for therapy procedures, and include operations to cut, separate, move, mirror, stretch, compress and bend objects; d) analysis, which aims to quantify morphological/functional information about the objects in the scene, yielding data such as density measures, intensity statistics, velocity, distance, length, curvature, area volume and mechanical properties.

This thesis deals with the first stage, pre-processing of medical image data. It addresses segmentation, noise reduction and feature detection, and makes a contribution towards resolving some of the current issues in those areas. This facilitates subsequent processing. Visualisation is improved by noise reduction, as the displayed data contains more of the desired information. Noise reduction also improves the accuracy of analysis, as clinical measures are not misled by random signal fluctuations. Segmentation brings direct benefits to visualisation, as rendering can then display only segmented image sections of interest. This is, of course, possible with manual segmentation. However, fully or even semi-automatic segmentation generally result in significant time savings in reduced operator time. The benefits of this are obvious in a clinical setting, when required to handle hundreds of patients. For manipulation, a noise-reduced or segmented dataset is usually easier to manipulate than raw data, because considerable data reduction has taken place. If the manipulation is operator-driven, the improved rendering of segmented data makes the manipulation task easier and more intuitive. For analysis, segmented data can be analysed quicker or even immediately. Organ volumes from 3D scans, for example, are trivial to measure if the organs in question have already been segmented. Another application is the measurement of mechanical properties of the heart, which can be computed directly from a segmented dataset.

Visualisation, manipulation and analysis have already been implemented with a high degree of success by several commercial as well as publicly available software packages. For example, CMR Tools (Imperial College, London) is an application suite for analysing 2D and 3D MRI data. It has facilities for loading and saving data, sequence animations, distance measurements, region delineation, signal analysis inside regions, as well as volume rendering, isosurface selection, and cardiac clinical measurements such as volume, mass and ejection fraction. The Dutch company Medis has a range of products for Echography and MRI, starting from full direct visualisation of raw data to cardiovascular flow velocity analysis and semi-automated angiography. Stradx (Department of Engineering, Cambridge University) is an acquisition and analysis program for 3D Echography, containing facilities for any-plane reslicing, semi-automated segmentation, volume rendering and volume measurement. Medical scanner manufacturers also provide a set of software applications to be used in conjunction with the scanner. The facilities provided are generally basic but certified to be reliable for daily medical use.

Segmentation, in particular, continues to be a bottleneck in clinical analysis packages. Much operator time is still spent on manual delineation of organ or vessel contours, for further processing, because fully available automatic systems are still an area of research. In many applications of 3D ultrasound, a volume of interest must be found before rendering due to noise and occlusion, and in order to compute measurements within that volume. In 3D cardiac MRI, there is less of a visualisation requirement and more of analysis: the left ventricle must be found, in order for elasticity, extensibility and other vital measures to be taken. In 3D velocity MRI, a method of acquiring blood flow information inside the human heart and vessels, direct visualisation of the flow patterns is impossible without automatic flow analysis, because of the large size of the dataset, and occlusion by other image features. In time series acquisitions, there is an added tracking challenge since the structures of interest move with time. In order for analysis to be carried out, those features must be tracked reliably to ensure that the measurements are being taken consistently over the same flow feature. This is further complicated when features end and new ones appear, or when the borders between them are unclear. The tracking aspect of time series acquisitions is not addressed in this thesis.

1.2 Thesis Outline

This thesis details our research on three applications of pattern recognition to medical imaging. The first application is concerned with three-dimensional echography and the detection of speckle pattern in the data. Three-dimensional ultrasound has classically been performed with hand-held ultrasound probes which are either 3D probes *per se*, outputting a voxel array, or 2D probes, outputting a sequence of 2D pixel arrays that are then registered into a 3D voxel array by means of a position sensor attached to the probe. An emerging alternative is based on using information from the probe data itself for spatial registration, without the need for a separate position sensor. This information is conveyed by the image speckle, an apparently random signal resembling image noise that is actually generated and determined by the physical interation betwen signal and tissue. It has been found that ultrasound speckle pattern is proportional to probe displacement in such a way that pattern changes in two nearby slices correlate well with the distance between those slices. To implement a positioning system based on this principle, a reliable means of finding speckle in the image is necessary. An asessment of existing methods, and the proposal and assessment of a novel method, are presented in chapter 2.

Essential background theory is introduced in chapter 3 on the Magnetic Resonance Imaging (MRI) modality and its flow variant, Velocity MRI. This is in preparation for the following three chapters, which deal with our research on these modalities. Chapter 4 deals with the detection of the left ventricle in cardiac MRI. This acquision mode captures MRI slices of the heart at different phases of the cardiac cycle and can generate image frame sequences or even movies. More clinically relevant for the study of pathology, however, is the quatitative evaluation of flow and mechanical characteristics related to the mechanical pumping function of the heart. There are several relevant measures associated with the movement of the left ventricle, but at present the extraction of these measures suffers from the important bottleneck of segmentation of the ventricle borders. Extensive manual intervention by a qualified techician is required to pinpoint the location of the left ventricle in each frame for each new patient. We assess the current state of automation software in this field and present our findings in the study of an original template-based method.

Chapter 5 describes our work on the first step of a three-step framework for processing cardiac blood flow Velocity MRI data. The framework consists of restoration of the noise-corrupted flow field, abstraction to extract relevant features, and tracking to find correspondences between features in different time frames. The first step has been fully implemented using total variation and the first-order method, and its effectiveness assessed. Results are presented for its application to simulated 3D flow data, including simulated flow of blood inside the heart. The method was first successfully applied to 2D *in vivo* data by a different author (see [175, 176]).

The abstraction step deals primarily with the detection of vortices in the blood flow inside the heart, and is closely related to the clinical study of cardiac function following infarction, or heart attack. At the onset of disease, a vortex ring can be observed in the cardiac flow which increases in size and vorticity with the passing weeks, primarily a result of an enlarged left ventricle. It is important to detect this vortex ring and analyse its characteristics in patients, for diagnostic and also for research purposes. This presents a main problem for visualisation: flow surrounding the vortices will occlude these from the observer. Chapter 6, presents our solution to this which combines clustering, local linear expansion and automatic streamline selection.

Papers published

Carmo, B. S., Prager, R. W., Gee, A. H. and Berman, L. *Speckle detection for 3D ultrasound*, Ultrasonics 40, 129-132, 2002.

Ng, Y. H. P., Carmo, B. S., Prügel-Bennett, A. and Yang, G. Z. *A first order Lagrangian based variational approach for MR flow vector field restoration.* Proc. Computer Assisted Radiology and Surgery (CARS) 2003.

Ng, Y. H. P., Carmo, B. S. and Yang, G. Z. *Flow Field Abstraction and Vortex Detection For MR Velocity Mapping.* Proc. MICCAI 2003, Lecture Notes in Computer Science 2878:424-431.

Carmo, B. S., Ng, Y. H. P., Prügel-Bennett, A. and Yang, G. Z. *A data clustering and streamline reduction method for 3D MR flow vector field simulation.* Proc. MICCAI 2004, Lecture Notes in Computer Science 3216:451-458.

Chapter 2

Speckle Detection for 3D Ultrasound

In this section we describe our work and experiments involving a fully automatic segmentation method: a pattern classifier for detecting speckle in ultrasound images. Speckle detection is required for sensor-less recording of ultrasound (US) probe position through speckle decorrelation.

As we referred earlier, freehand 3-D ultrasound employs a procedure whereby the clinician is allowed to move the probe as in a normal B-mode examination. The probe's position is tracked by the system's hardware, which allows its software to position frames relative to each other in order to produce a 3-D dataset. Determination of probe movement can be achieved by several methods, including magnetic and optical position sensors. In both cases, a position beacon is attached to the handheld probe while a fixed detector assembly determines the beacon's placement in 3-D space. With appropriate calibration, the 3-D ultrasound system is able to label each B-scan with the position and orientation of the scan plane. Currently available positioning systems are expensive and complicated to install and use. They also have limited accuracy, which makes it highly desirable to replace them with more sensitive techniques, or otherwise to complement them with other sources of position data. Such an enhancement could increase the accuracy of organ and tumour volume measurement using freehand 3-D ultrasound, and also enable better 3-D reconstruction of images produced by the more recent high-definition ultrasound probes. According to current research, this may be achieved using information present in the images themselves, using speckle decorrelation [37, 225, 240].

Ultrasound images contain several types of features: anatomically-related structures, which result from the interaction between the ultrasound waves and tissue structures and boundaries; artifacts and noise, which are caused by hardware and software limitations and do not express any actually existing structures; and speckle, a globular

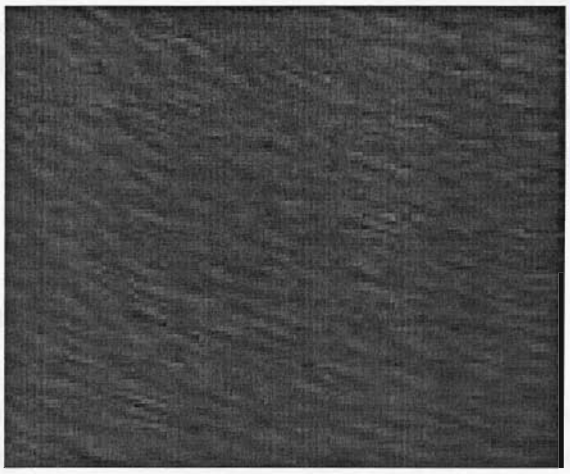


FIGURE 2.1: An ultrasound speckle pattern.

pattern generally found in areas of the image which correspond to homogeneous tissue (Figure 2.1).

Ultrasound speckle is an image structure that does not correspond to any existing structure in the tissue being imaged. In fact, because the tissue is homogeneous, a uniform brightness level would be expected to appear in the corresponding area of the image. Instead, a pattern is displayed. This pattern has been found to be determined by physical characteristics of the ultrasound transducer being used, as well as scattering properties of the insonified tissue volume.

If the ultrasound probe is held stationary during a scan, the pattern of speckle remains unchanged if the subject is still[1]. Two successive, closely-spaced B-scan frames of a freehand probe sweep contain areas of speckle with measureable correlation. The level of decorrelation which results from moving the probe in the elevational (out-of-plane) direction can be used to determine the extent of the displacement. Therefore, given the precise location of speckle areas in the image, and their correspondence in successive frames, it is possible to determine their level of decorrelation in order to calculate probe movement.

Speckle tracking and decorrelation methods have the potential to enhance performance or even eliminate the need for a separate position sensor, because they can provide the ultrasound system with position and orientation information gathered from data present in the B-scan frames alone. Commercially available systems that have replaced external position sensors employ proprietary techniques that involve some form of speckle decorrelation, combined with other methods such as Doppler spectra and echo timing using multiple-line probes [38, 81, 265, 266].

[1]In practice, there are contributions from random noise in the image. Also, the tissue inevitably moves slightly due to physiological processes such as blood flow. Here, we assume that the effect of these deviations is negligible.

In the next sections we present the theory of ultrasound speckle and its relationship with the optical theory of laser speckle. We also explain its relevance towards current research in speckle detection and classification for tissue identification. We then report on our own studies for detecting and tracking speckle regions, and we compare our results with previous techniques.

2.1 Theory of Speckle

Speckle appears in ultrasound images as a characteristic granular pattern, generally in areas corresponding to homogeneous tissue. The expected representation would be a uniform brightness level, but random-looking signal fluctuations appear instead. Because the actual tissue being imaged does not contain any structures resembling the granular pattern, speckle is usually classified as a B-scan artifact. It is sometimes an undesired feature, since it inhibits detection of low-contrast structures [5, 48, 99].

The cause of speckle is believed to be scattering by reflectors smaller than the *resolution cell* (the smallest region that can be resolved by the ultrasound beam). These reflectors are called *non-specular*; tissue boundaries, which are represented in B-scans by sharp lines, are called *specular reflectors.* At each non-specular reflector, a scattered wave front is emitted in all directions. Multiple wavefronts from the all the scatterers in each resolution cell reach the transducer with various phases, which results in the well-known phenomenon of *interference* (Figure 2.2). Because of the very small magnitudes involved, the combined contribution is unpredictable and must be treated statistically.

Speckle patterns are not random in the same sense as, for example, electrical noise, which is also present in B-scans. When an object is scanned twice in the same position under the same conditions, the resulting B-scans contain identical speckle patterns. Furthermore, the size of the speckle granules is approximately the same as the lateral and axial resolution of the scanner [28].

At present there are no well-defined criteria in the literature for systematically locating speckle in an ultrasound image - it is not possible to state with certainty whether a selected image region contains features which are attributable exclusively to speckle phenomena. What is known, however, is that speckle arises in image regions that correspond to parenchymal tissue[2], or any other medium which is homogeneous throughout (e.g. blood, placenta) and contains a critical concentration of sub-resolution particles called *scatterers.* Besides speckle, the image may also be corrupted by random hardware noise and acoustic artifacts. As we will see in the next section, many speckle detection algorithms are based on knowledge of the process of speckle formation, from which a characteristic signature is derived and used as a search criterion.

[2]Parenchyma is made of structurally unspecialised cells that have adapted to serve particular functions.

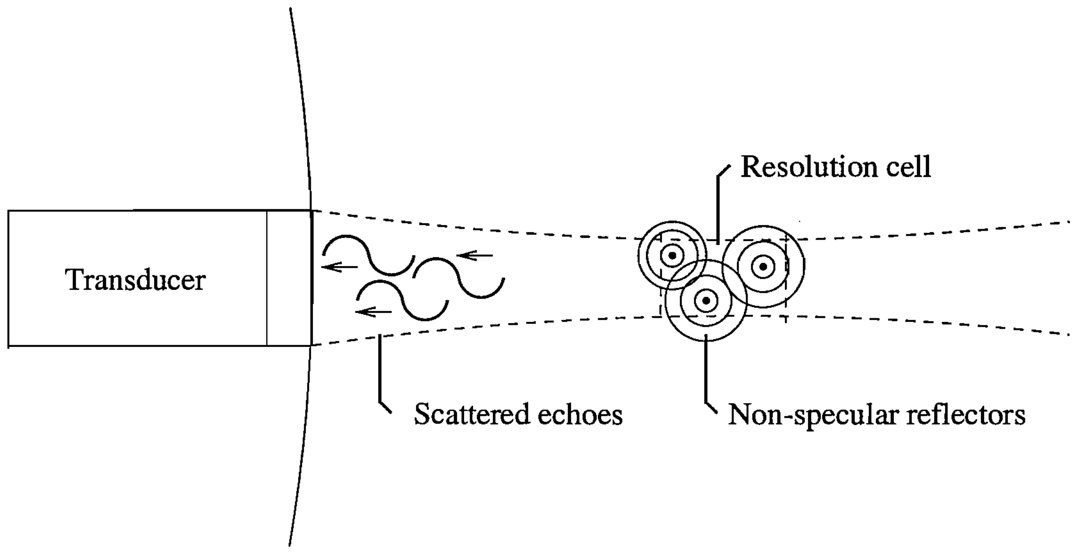


FIGURE 2.2: **Production of a speckle pattern.** Each non-specular reflector, smaller than the resolution cell, receives the ultrasound pulse and produces a scattered wavefront in all directions. The wavefronts from all the reflectors in the cell return to the ultrasound transducer as echoes with various phases, causing interference. This phenomenon occurs at resolution cells that contain a critical number of scatteres, which results in a speckle pattern being formed in the ultrasound B-scan image frame.

In optics, a phenomenon similar to speckle occurs when a beam of coherent light (such as a laser) strikes a rough surface and is reflected back: the roughness of the surface causes multiple reflections to reach the detector and interfere with each other. This adding up of many sinusoidal pulses with statistically independent phases leads to the *random walk* phenomenon. For the purposes of speckle signature prediction in B-scans, the current theory on speckle adopts a model that is analogous to the optical counterpart.

Treating the echo signal amplitudes throughout the B-scan frame as a stochastic variable, the statistics of its distribution can be predicted theoretically. The echo signal is modelled as a sum of components in the complex plane. Each component is the contribution of an individual scatterer in a resolution cell. The complex echo signal returned from that cell is stochastic and can be shown to obey a complex Gaussian probability density function (PDF) if the number of scatterers is large [90]. Processing removes the phase component by *envelope detection*, and the processed signal can be shown to follow a Rayleigh amplitude PDF. If a coherent background is added, a constant strong component arises which transforms this into a Rician PDF, of which the Rayleigh PDF is a special case [258]. The more general K distribution describes the scatterer densities below the critical limit. These PDFs are termed *first-order* statistics because they deal with the intensity of each point in the image.

Which PDF is the most appropriate to describe speckle is still a choice being debated. In practice, ultrasound equipment performs logarithmic compression on the envelope echo signal, resulting in an actual distribution that is more difficult to derive [69]. Instead of trying to predict the PDF of the post-processed signal, the original data can be reconstructed approximately [125] and corrected to compensate for non-linear mappings introduced by the equipment [191].

In the next section, we discuss the consequences of the random walk model, and report on its applications for the purpose of detecting speckle in B-scan image frames.

2.1.1 Speckle Detection

Automatic recognition of speckle patterns in B-scan image frames may employ two types of algorithms: speckle-specific, where the statistics of signal distribution are predicted theoretically from the model of speckle formation and used as a signature for speckle detection and classification; and generalised image processing methods that quantify the distribution of brightness levels and build a texture library, from which classes are derived for image region segmentation.

Speckle detection has important applications in image noise reduction, tissue characterisation for diagnosis, blood flow measurement, automatic image segmentation for volume measurement and, as mentioned before, probe movement detection in 3-D ultrasound when combined with speckle decorrelation. In this section, we survey previous and current research in this area and discuss possible applications. We also present results from our own studies.

Signal-to-Noise Ratio

The fractional n-order signal-to-noise ratio (SNR) of a signal X_i with **N** samples is defined [68] as:

$$SNR^{(n)} = \frac{\overline{A^{(n)}}}{S^{(n)}} \tag{2.1}$$

where

$$\overline{A^{(n)}} = \frac{1}{N}\sum_{i=1}^{N} X_i^n \tag{2.2}$$

and

$$S^{(n)} = \frac{1}{N-1}\sum_{i=1}^{N}(X_i^n - \overline{A_i^{(n)}})^2 \tag{2.3}$$

where (n) denotes the fractional moment being considered. If X_i takes the value of a theoretical ultrasound echo envelope signal that follows a Rayleigh PDF, it has been shown that the integral moment SNR (n=1) should equal 1.91; experimental data from speckle in A-scans yielded SNR values between 1.5 and 2.5 [28], a wide range that suggests SNR may be a non-ideal indicator of the presence of speckle.

Autocorrelation Function

The autocorrelation function (ACF) is a popular second order texture statistic. It is defined [226] as:

$$ACF(p,q) = \frac{MN}{(M-p)(N-q)} \frac{\sum_{i=1}^{M-p} \sum_{j=1}^{N-q} I(i,j)I(i+p,j+q)}{\sum_{i=1}^{M} \sum_{j=1}^{N} I^2(i,j)} \tag{2.4}$$

where p,q is the position difference in the i,j direction, and M,N are the image I's dimensions. The ACF measures the area integral of the internal (dot) product between an image region and its copy, displaced by a varying distance along a chosen direction. The integral is evaluated only where the two patches overlap, and is normalised by the area of overlap [97].

Wagner *et al.* [257] evaluated this integral as distance is increased, using patches of simulated Rician speckle. Beyond a certain distance, their plot resembles a sinusoid-shaped curve whose Fourier transform can be fitted well by a Gaussian. This is to be expected since single frequency structured speckle was used. This type of speckle does not exist in real images, which in our view explains why the paper's strategy of classifying speckle by measuring peak differences in the ACF plot was unsuccessful.

In our own experiments, we produced ACF plots for p,q varying along 0^o, 45^o and 90^o. We found that peaks and valleys could appear at various, unpredictable positions in the sequence, in agreement with [3]. The plots in Figure 2.3 are typical of the problem. This irregularity makes the technique of measuring peak value and position differences meaningless for the purposes of ACF curve classification of *in vivo* US display images. This problem may be alleviated when the unmodified intensity data is available as in Wagner *et al* [257], but that is not the case in our study.

Co-occurrence Matrix

The co-occurrence matrix $COM_{\theta,d}(a,b)$ measures how frequently two image pixels with grey levels a,b appear separated by a distance d along direction θ [97]. For example, the image

$$\begin{matrix} 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 \\ 0 & 2 & 2 & 2 \\ 2 & 2 & 3 & 3 \end{matrix}$$

has the following COM matrices for d=1 at angles 0^o and 135^o, respectively:

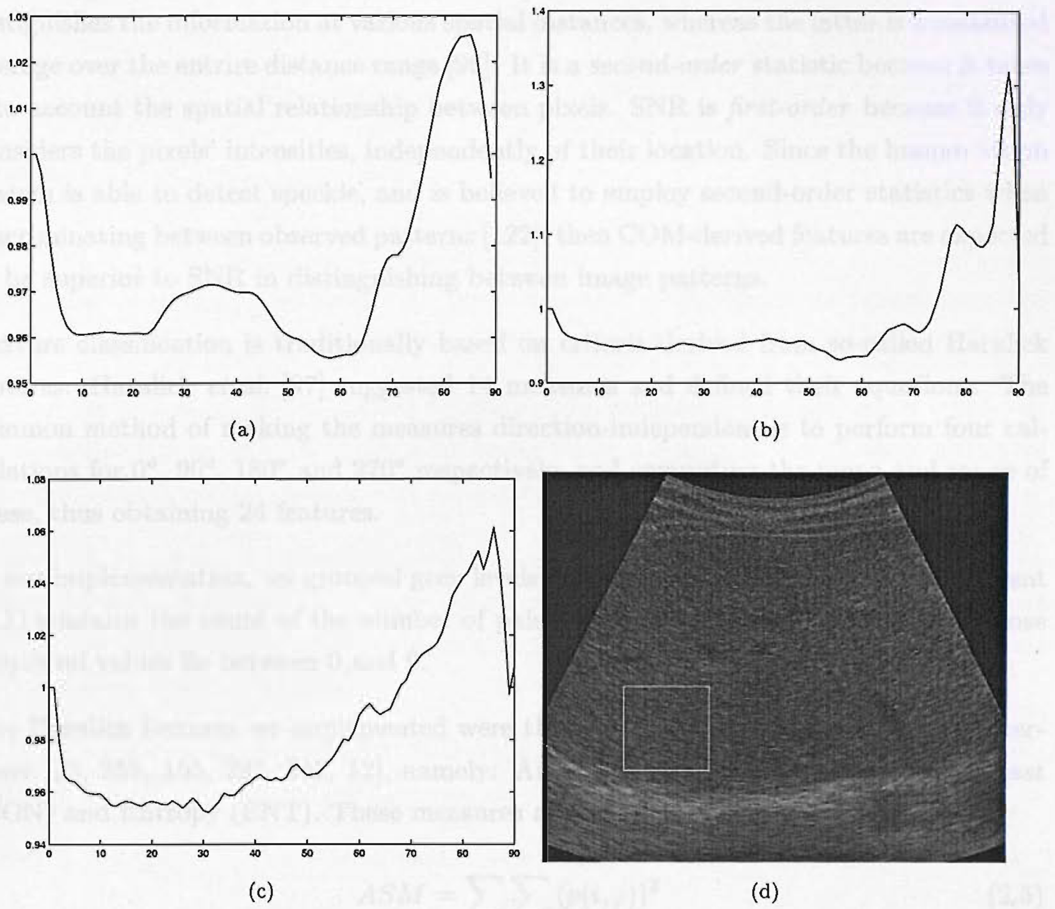


FIGURE 2.3: Plots of autocorrelation function values against distance for different offset orientations: (a) 0^o; (b) 45^o; (c) 90^o; (d) Original image with region of interest highlighted.

$$P_{0^o,1} = \begin{matrix} 4 & 2 & 1 & 0 \\ 2 & 4 & 0 & 0 \\ 1 & 0 & 6 & 1 \\ 0 & 0 & 1 & 2 \end{matrix}$$

$$P_{135^o,1} = \begin{matrix} 2 & 1 & 3 & 0 \\ 1 & 2 & 1 & 0 \\ 3 & 1 & 0 & 2 \\ 0 & 0 & 2 & 0 \end{matrix}$$

Note some pairs count twice as θ denotes the direction between the two, rather than the direction from one to another. With a large image, these matrices are calculated repeatedly over a sliding region of interest (ROI). Two ROIs of the same size produce a fast varying COM for fine textures, and a slow varying COM for coarse textures.

The COM is a better texture descriptor than the autocorrelation function because it distiguishes the information at various spatial distances, whereas the latter is a measured average over the entrire distance range [96]. It is a *second-order* statistic because it takes into account the spatial relationship between pixels. SNR is *first-order* because it only considers the pixels' intensities, independently of their location. Since the human vision system is able to detect speckle, and is believed to employ second-order statistics when discriminating between observed patterns [122], then COM-derived features are expected to be superior to SNR in distinguishing between image patterns.

Texture classification is traditionally based on criteria derived from so-called Haralick *features.* Haralick *et al.* [97] suggested 14 measures and defined their equations. The common method of making the measures direction-independent is to perform four calculations for 0°, 90°, 180° and 270° respectively, and computing the mean and range of these, thus obtaining 24 features.

In our implementation, we grouped grey levels in intervals of 10, so for example element (1,1) contains the count of the number of pairs of pixels separated by d along θ whose grey-level values lie between 0 and 9.

The Haralick features we implemented were the ones most frequently used in the literature [25, 255, 155, 282, 245, 12], namely: Angular Second Moment (ASM), Contrast (CON) and Entropy (ENT). These measures are defined as follows:

$$ASM = \sum_i \sum_j (p(i,j))^2 \tag{2.5}$$

$$CON = \sum_{n=0}^{N_g-1} n^2 \left(\sum_{\substack{i=1 \\ |i-j|=n}}^{N_g} \sum_{j=1}^{N_g} p(i,j) \right) \tag{2.6}$$

$$ENT = -\sum_i \sum_j p(i,j) \log(p(i,j)) \tag{2.7}$$

where $p(i,j)$ is the co-occurrence matrix element, normalised by the number of pixel pairs used in the computation[3] and N_g is the number of distinct grey-level values. For each measure, the direction-independent mean and range are computed, thus yielding 6 separate features.

2.2 Classifier Implementation and Assessment

We implemented a speckle detector using current pattern recognition procedures. In the first phase of development, a pattern recogniser was implemented to assess the

[3]Our implementation does not perform this normalisation for efficiency reasons, since this does not impact the class recognition.

performance of several image feature extraction mechanisms in detecting speckle. In the second phase, results from this assessment were combined to devise a combined speckle detection strategy.

Learning From Data Approach

SNR and COM are examples of image features - they are measures that enable the assignment of image pixels to classes. In the present study, there are two classes: the regions in the ultrasound image that contain speckle, and those that don't.

We assume that each class is defined by decision boundaries, which are upper and lower thresholds of feature values. Pixels with feature values lying inside this range belong to the class, otherwise they belong to a different class.

In practice, classes have feature ranges that overlap, and the choice of decision boundaries is approximate and introduces error. The average probability of classification error *P(error)* is defined as the probability of assigning each pixel to the wrong class, based on the chosen decision boundaries.

Here, we locate decision boundaries by searching in feature space for the thresholds that yield the lowest *P(error)*. This is defined as:

$$P(error) = \frac{\text{no. of missed pixels} + \text{no. of misclassified pixels}}{\text{image size}} \tag{2.8}$$

Missed pixels are speckle pixels classified as non-speckle, and misclassified pixels are non-speckle pixels classified as speckle.

The optimal range of feature values is computed by running this search over a set of images called the training set. The average *P(error)* obtained gives an indication of the best-case error to be expected, but not necessarily achieved, in practice. A realistic estimate of *P(error)* was obtained by applying the optimal range over a different, larger set of images usually called the *testing set* [103].

Speckle Detection Assessmet

We implemented a pattern classifier that computes value ranges on several features from training images, and measures the resulting *P(error)* on test images.

The image sets are frames extracted from Stradx data files, acquired using a Toshiba Powervision with 7 Mhz, 3.5 MHz and small parts probes. The selected datasets consisted of scans through: kidney; 16-week foetus along spine and axial directions; liver; arm; bladder. Approximately 80% of the data was scanned by Laurence Berman, who is a qualified radiologist. Frames were divided in equal proportions between the training dataset (111 images) and the test dataset (222 images), yielding a total of 333 images.

Speckle in these images was marked manually. A separate manual segmentation image was created by this author for each dataset image. The manual segmentation also included pixels in the image that do not correspond to ultrasound information (image canvas) for correct *P(error)* determination. This process relied on the operator's experience and knowledge of ultrasound image formation. The operator did not know which set each image belonged to (training or test), but he was aware of the algorithms under study.

For SNR, we implemented integral as well as fractional order moments taken from decompressed data following the procedure in [191] (features SNR 0.25, SNR 0.5, SNR 0.75 and SNR 1.0 in Figure 2.4). Three estimates of the decompression parameter were computed for each Stradx dataset; this process, repeated for all Stradx datasets used in this study, yielded an estimate for the decompression parameter of 26.3 mean with a 7.1 standard deviation. The mean value was used to decompress all the images. The SNR for the given B-scan was also used (SNR I in Figure 2.4).

Training consisted of a search for a combination of three parameters that resulted in the lowest *P(error)* (computed by comparing the classifier output with the manual segmentation images). The parameters were the lower and upper thresholds of SNR value, and the size of the square sliding region of interest (ROI) used to compute the local SNR value. The search for thresholds was exhaustive over all values in the image; the search for the best ROI size was limited to widths between 24 and 84 pixels over 4-unit increments.

For COM, a search scheme identical to that for SNR was used, over the same range of ROI sizes. Each measure yielded a mean- and range-related feature ("m" and "r" in Figure 2.4).

The mean values of the lower and upper thresholds, and also of the square ROI sizes, obtained from training were used as fixed values to detect speckle using the same features over the test dataset. Manual segmentation images were only employed to measure the final error rates. For COM and SNR features, square ROI widths between 60 and 72 pixels were used.

The speckle detection algorithm (HOM) described in [191] was also assessed; since it requires no training, it was only run on the test dataset.

Figure 2.4 also has algorithms SPK and SPK CON, which will be discussed further ahead in this Chapter.

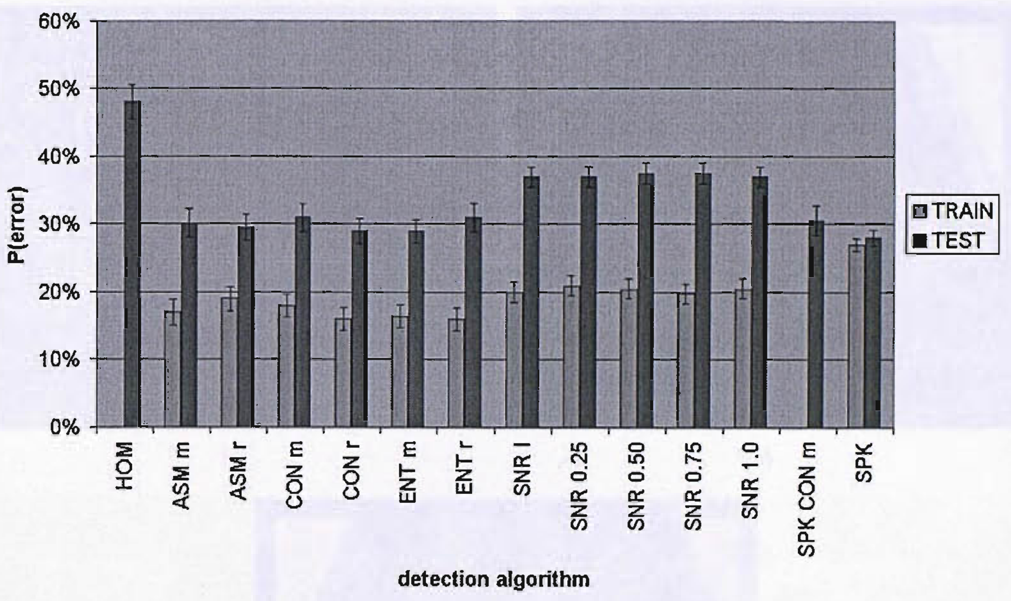


FIGURE 2.4: Train and test results for all detection algorithms. Error bars were computed with the student-t test [161].

Discussion of performance results

Of the algorithms discussed here so far, all had training performances to within 5% of each other. As for testing performance, COM-based algorithms performed best with *P(error)* = 28%-31%, followed by SNR-based with 37%-38%, and HOM with 48%.

All COM features have a tendency to incorrectly mark noise as speckle. We can observe from Figure 2.5 that shadow areas with a purely noisy pattern have feature values that are very close to speckle zones. When setting optimal thresholds with COM features, many noise regions are also marked. This may be acceptable for images with small patches of noise, but may produce large errors when the areas of noise are large. This is particularly serious for e.g. pregnancy scans, with predominance of amniotic fluid regions that generate large shadows.

We also found a noise-related issue when SNR measures are involved. Because of different noise-reduction strategies employed by the US machine post-processor, shadow areas may be displayed either as black regions if the filtering is strong, or as distinct noise if the filtering is weak. This results in the SNR for such regions varying over a range which is unpredictable: black areas have a constant signal that has high SNR, whereas noise areas have steep variations that reduce SNR. Different gain and timing settings on the machine may accentuate this issue.

Also, COM and SNR feature values vary over a large range for speckle areas alone. In fact, when setting the optimal thresholds by training, the standard deviation for ASM

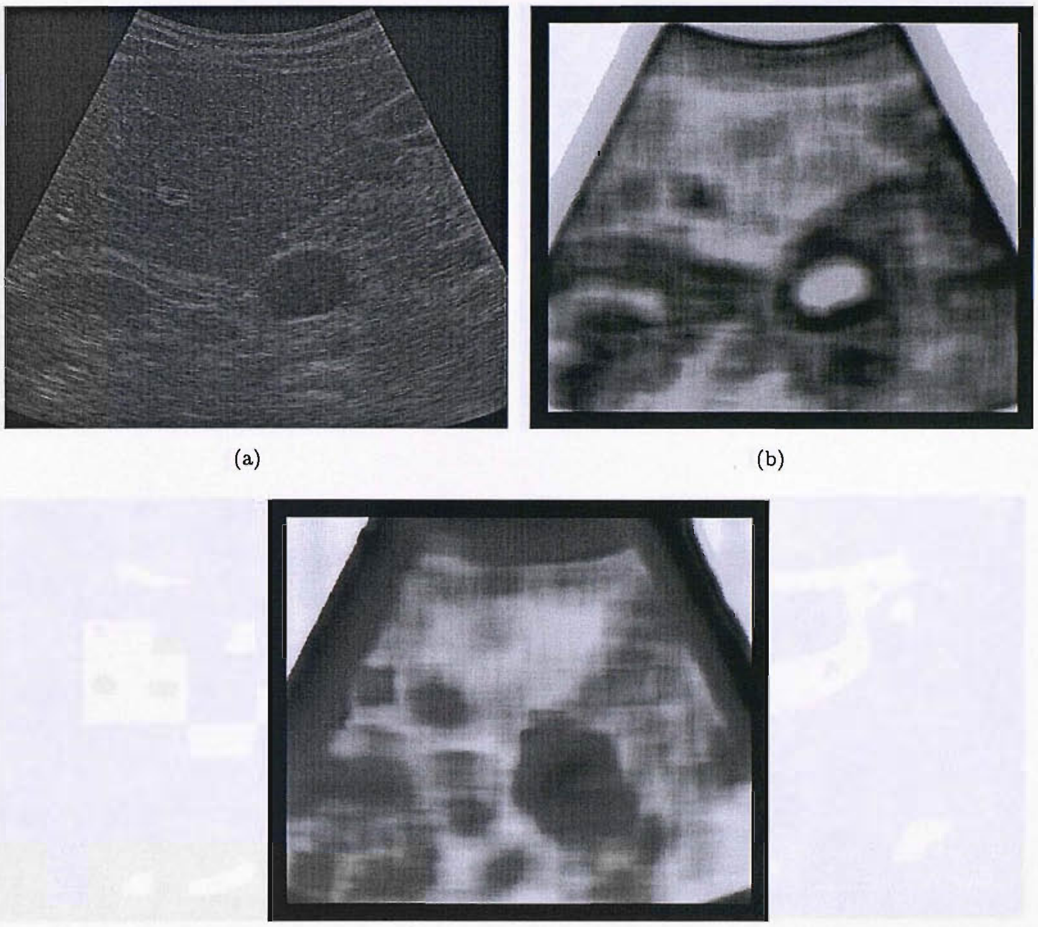


FIGURE 2.5: Feature values plotted at centre of 35×35 ROI window. (a) Original image; (b) COM feature: ASM mean; (c) SNR taken from given B-scan.

and CON range values was over 50% of the mean value. The consequence of this is that performance in testing is poorer than in training. To illustrate this, we trained features **SNR I** and **ASM m** over one US image, and plotted the classification results (Figures 2.6.(a) and (b)); training was done as before, by choosing the range of feature values that yield the minimum error rate from knowledge of the manual segmentation. However, if we test the algorithms on the same image, using the ranges obtained by training over the entire training set, different plots are obtained (Figures 2.6.(c) and (d)) whith observable degradation in classification accuracy.

The HOM algorithm performs poorly because of two factors. First, the need for a large ROI (a circle 64 pixels in diameter) that tends to enclose both speckle and non-speckle regions, generating errors by averaging. Second and most important, we found a high sensitivity of recognition performance to the decompression parameter used (see Figure 2.7), whose value in turn is very sensitive to the position of the areas chosen for the estimation procedure (see [191]). This suggests that the assumption that the US envelope

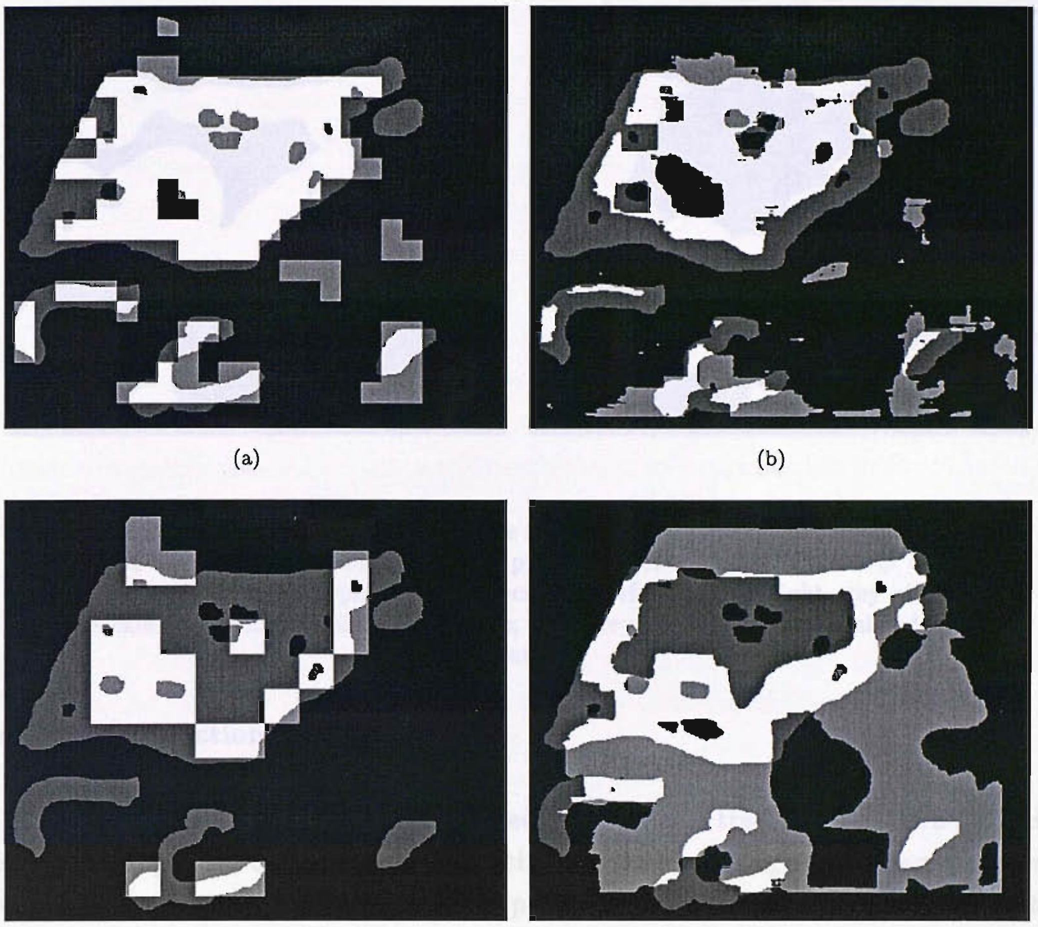


FIGURE 2.6: Speckle detection plots for the same B-scan as in Figure 2.5. a) **ASM m** with optimal feature value range trained from manual segmentation data; b) **SNR I**, also with trained feature value range; c) **ASM m** with range obtained from training over entire training set; d) **SNR I**, also from trained range. White pixels are correctly identified speckle; light grey pixels are non-speckle incorrectly identified as speckle; dark grey pixels are speckle missed by the detector.

data may be reconstructed faithfully by simple exponentiation of the pixel values may need to be revised. In fact, the handling of the RF echo data by the US machine, especially facilities for variable local gains and the rasterisation function, may modify the data to an extent that makes actual image statistics differ highly from predicted ones.

As we describe in the sections to follow, we explored two courses of action in view of these conclusions. First, we attempted to improve the existing features by combining them using traditional pattern recognition techniques. Second, we explored the addition of a novel feature, based on a structural prior.

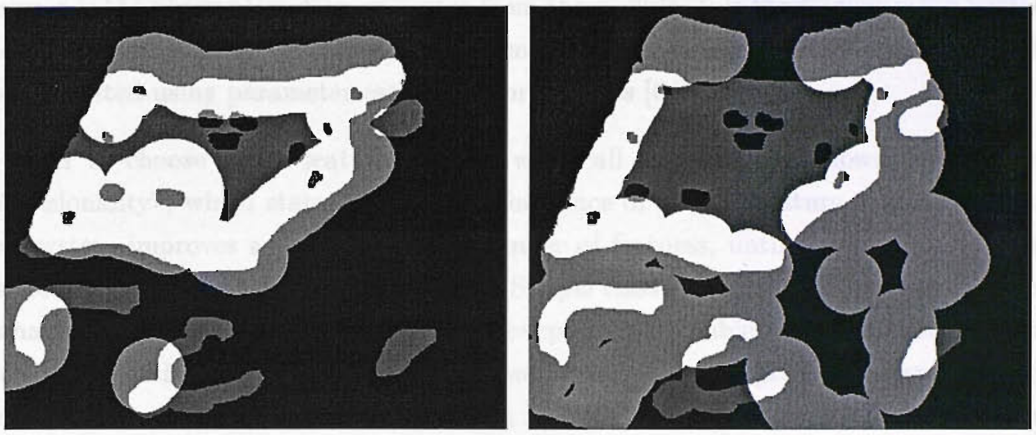


FIGURE 2.7: Speckle detection plots for the same B-scan as in Figure 2.5, produced by the **HOM** algorithm using different decompression levels *decomp*. a) *decomp* = 19.08; b) *decomp* = 23.00. White pixels are correctly identified speckle; light grey pixels are non-speckle incorrectly identified as speckle; dark grey pixels are speckle missed by the detector.

Combined Detection Strategy

In order to combine all the computed features, we followed a traditional pattern recognition approach [83, 19, 65]. One way of identifying samples belonging to different classes is to use hyperplanes to partition feature hyperspace into category zones. Sample points are enclosed by planes so that the probability of their belonging to that category is maximised. However, this implies assuming that the probability densities are normal in form. In our case, we could not make this assumption since the non-speckle class may be composed of many sub-classes in different positions in feature space. Such a case can be modeled by clustering techniques, which consist of automatically identifying regions in feature space where sample density is high, and assuming that those are the centres of the normal probability densities. However, the determination of density thresholds for clustering is diffucult or arbitrary, and the subsequent partitioning is non-optimal.

If we model the classification as a two-class problem (speckle and non-speckle) and assume that the speckle class has a normal density function in feature space, then a search can be conducted for the optimal distance between sample points and the mean vector for which the classification error is minimum. Mahalanobis distance thresholding approaches have been followed previously [170, 139].

The Mahalanobis distance is defined as:

$$r^2 = (\underline{x} - \mu)^T \Sigma^{-1} (\underline{x} - \mu) \tag{2.9}$$

where $\underline{x}$ is the observation feature vector from the sample, μ is the feature mean vector, and Σ is the covariance matrix derived from the observations. The values of μ and Σ are calculated using parameter estimation procedures [64].

In order to choose which features to use, we recall the principle known as "curse of dimensionality", which states that the performance of a multi-feature pattern recognition system improves as we reduce the number of features, until an optimal subset is reached, past which performance degrades. Simple class separability criteria such as the Mahalanobis distance (when used for this purpose) are unable to model this phenomenon [83]. Furthermore, certain combinations of features may provide information not available in any of the features individually. In light of these issues, we decided to perform an exhaustive search over all possible subsets of an appropriately reduced feature set.

In order to perform feature reduction, we calculated the correlation matrix for all feature combinations. The correlation matrix was derived from the covariance matrix using simple matrix manipulations. Parameter estimation was applied over all the images in the training set, using a sampling grid with 4-pixel horizontal and vertical separations. Features which were correlated to others by 50% or more were eliminated, leaving the following: ASM m, CON m, SNR I and SPK var (see section "Specklet Detector", below).

All combinations of two to four features were assessed for their speckle identification performance using the same procedure as before: for each training image, the optimal Mahalanobis distance was computed for each feature subset. Then, the mean threshold for each subset was computed and used as a fixed value over all the images in the traning set. The performance results for the combined approach were all within 5% of those using the features alone, and can again be explained by the large variation of feature values throughout different images. Also, the probability density of the speckle class may be significantly different from the assumed normal distribution. Performance may be improved by implementing Mahalanobis outlayer detection [39]. Also, density functions of separate classes could be catalogued to permit clustering.

Specklet Detector

Czervinski *et al.* [51] applied an optimised version of the Hough transform to detect lines in an ultrasound scan. Their approach produced images that on visual inspection contain enhanced lines and reduced speckle. This suggested that speckle may be differentiated by the absence of lines in its pattern.

In 1995, Bashford and von Ramm [10] reported experimental evidence on the existence of a three-dimensional structure of speckle. In 1996 the same authors describe [11] a manual-based technique to track speckle kernels in order to estimate blood flow velocity.

Two years later, Morsy and Von Ramm [172] presented promising *in vivo* results of a correlation-based tissue motion tracking method. In 1999, the same authors report on positive *in vitro*, in-plane results of a method combining speckle kernel detection with correlation search and suggest its potential application for blood flow measurement. In the same year, Teo [235] obtained a patent for an invention that detects, in an image, areas containing speckle due to blood by using Fourier analysis. A combination of these techniques may in the future enable blood flow and tissue motion estimation in an automatic fashion.

We implemented an unsupervised speckle kernel tracker. The algorithm consists of local peak detection, followed by peak elimination using a flatness criterion and a brightness criterion.

The first stage of the algorithm detects all local peaks in the image in order to roughly estimate the likely location of speckle in the image. We perform peak detection to locate *specklets*, which we postulate to exist and consider as being the smallest textural elements characteristic of speckle [10]. Based on simulated data, we also postulate that speckle areas are those where the greyscale values of specklet peaks change the least within a given region of interest - specklet flatness criterion. Also, we observe that the brightness of noisy regions of the image is lower than in speckled regions - noise differentiation criterion. This second condition is necessary because in certain images, noisy areas also exhibit peaks and can therefore be easily confused with speckle.

We implemented a seed finder algorithm that follows these rules. The algorithm's stages are:

1. Detect all local peaks in the image.
2. Generate an image of the peak differences: slide a square 35×35 window over the original image; for each position, set the value of the pixel at the centre of the window to the variance of greyscale values of the peaks inside the window.
3. From the peak differences image generated in step 2, note which pixel has the lowest value and mark as speckle candidates those pixels whose values are within a set tolerance of this minimum; reject all other pixels. This implements the specklet flatness criterion.
4. From the binary image of step 3, measure the mean brightness over ROIs centered at each candidate pixel; eliminate candidates with a brightness value that lies within a set tolerance of the minimum measured brightness. This implements the noise differentiation criterion.

Given appropriate tolerances and thresholds, the specklet finder outputs a binary image where most marked pixels lie inside regions which correspond to speckle in the original image (Figure 2.8).

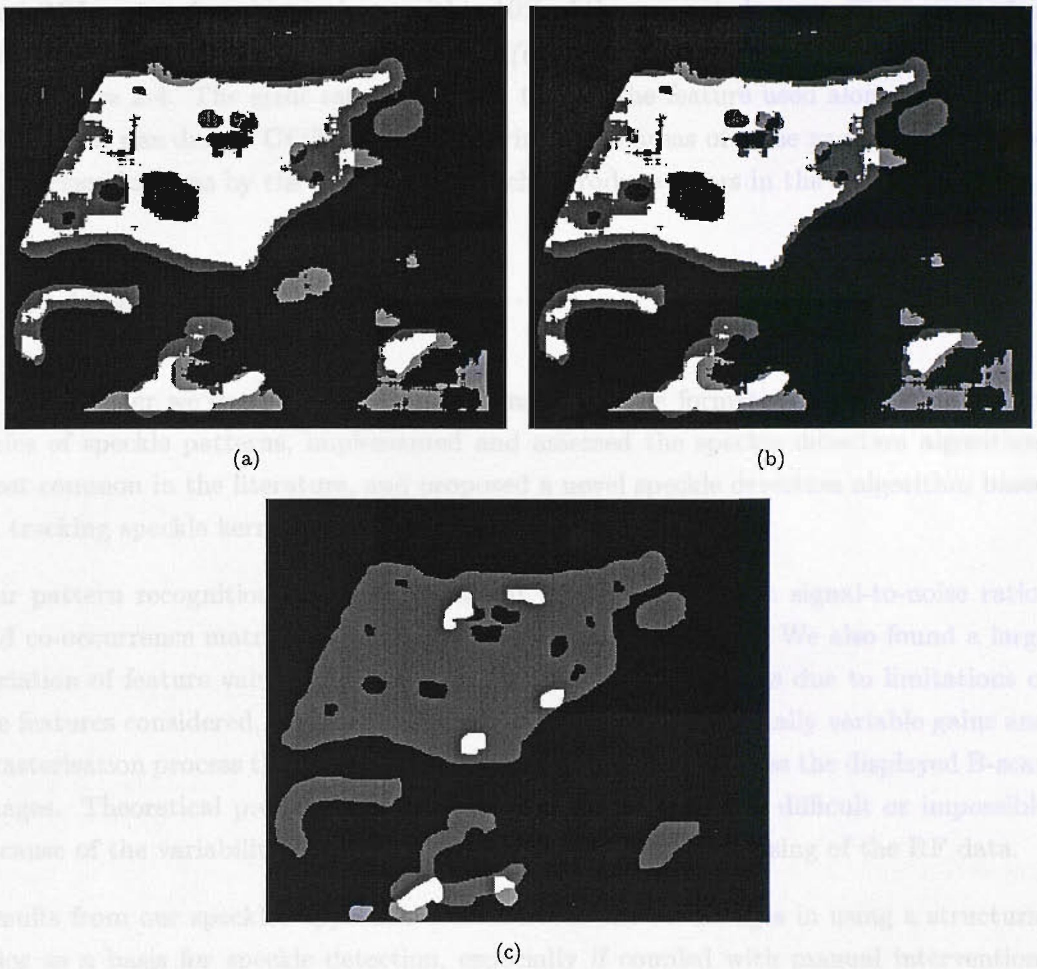


FIGURE 2.8: Specklet detection plots for the same B-scan as in Figure 2.5. a) With specklet flatness criterion only - note shadow region marked as speckle; b) With noise differentiation criterion; c) with narrow tolerances - note misclassification errors still occur. White pixels are correctly identified speckle; light grey pixels are non-speckle incorrectly identified as speckle; dark grey pixels are speckle missed by the detector.

The tolerances for best error rates vary between datasets. For example, images with large shadow regions may require a narrower noise tolerance than others where speckle covers most of the image. This suggests the possibility of user intervention by manually changing of the tolerances until a satisfactory performance is achieved. We explored this possibility by automatically searching for the best tolerances over the test dataset and measuring the error rate - feature **SPK** in Figure 2.4.

We also explored the possibility of using the specklet detector as a seed finder. From this seed, a feature value is computed and used as a threshold for detecting other speckle in the image. We implemented the specklet detector with very narrow tolerances to find small patches of speckle (see Figure 2.8c) for a typical detection). We then measured the average value for feature **CON m** over the seed pixels. Finally, other pixels in the image were marked as speckle depending on whether the feature value inside a 65×65 pixel ROI centered at the pixel was within 10% of the computed mean. This was applied over the testing data set and the average *P(error)* was measured - feature **SPK CON m** in Figure 2.4. The error rate is between that of the feature used alone, and that of **SPK**. This was due to **CON m** misclassifying some areas of noise as speckle, and also to misclassifications by the seed finder, which introduce errors in the computed mean.

2.3 Discussion

In this chapter we described the mechanisms of speckle formation, analysed the properties of speckle patterns, implemented and assessed the speckle detection algorithms most common in the literature, and proposed a novel speckle detection algorithm based on tracking speckle kernels.

Our pattern recognition approach identified weaknesses in both signal-to-noise ratios and co-occurrence matrices in differentiating speckle from noise. We also found a large variation of feature values within the speckle class alone. This is due to limitations of the features considered, coupled with the added complexity of locally variable gains and a rasterisation process that introduces directionality changes across the displayed B-scan images. Theoretical predictions for values of these features are difficult or impossible because of the variability introduced by US machine post-processing of the RF data.

Results from our specklet approach demonstrated the advantages in using a structural prior as a basis for speckle detection, especially if coupled with manual intervention. The relationship between specklet feature values within an image is readily predictable, and user interaction is advantageous for selecting the best US acquisition control settings and specklet feature tolerances. The noise and variability issues identified earlier suggest that such user interaction may not be useful when using SNR or COM discriminants.

This research produced several original contributions, including the systematic assessment of the performance of popular speckle detection algorithms, proposal of a novel algorithm whose advantages are demonstrated by the results of our assessment, and further knowledge about the advantages and weaknesses of one type of fully automatic segmentation strategy, the pattern classifier.

Chapter 3

Cardiac MRI Acquisition and Display

3.1 Cardiac blood flow

Heart Anatomy

As can be seen from Figure 3.1, the heart consists of four chambers, four valves and several vessels carrying blood into and out of the heart. The *superior* and *inferior vena cavae* are the veins that bring blood from the rest of the body to the *right atrium*. The blood then enters through the *tricuspid valve* to the *right ventricle* (RV). From there it is pumped through the *pulmonary valve* entering the *pulmonary artery* and then through to the lungs to be re-oxygenated. Once reoxygenated, the blood is carried back to the heart through the *pulmonary veins* to be circulated to the rest of the body. It enters the *left atrium* and once that is filled, the blood is pushed through the *mitral valve* into the *left ventricle* (LV). The left ventricle does the majority of the work by then pumping the blood through the *aortic valve* to the *aorta* and out to the rest of the body [159].

The structure of the heart can be studied at different levels. The cellular level can be readily analysed and is relatively well known. At the macroscopic level, the muscle structure is still uncertain and results depend on how dissection is performed. At the functional level, the muscle structure must be related to the functional behaviour of the heart. This level of study aims to describe how the normal heart functions and how things go wrong in disease. Non-invasive imaging methods are very useful in achieving this, and in the next section we briefly describe the method related to this thesis, Magnetic Resonance Imaging.

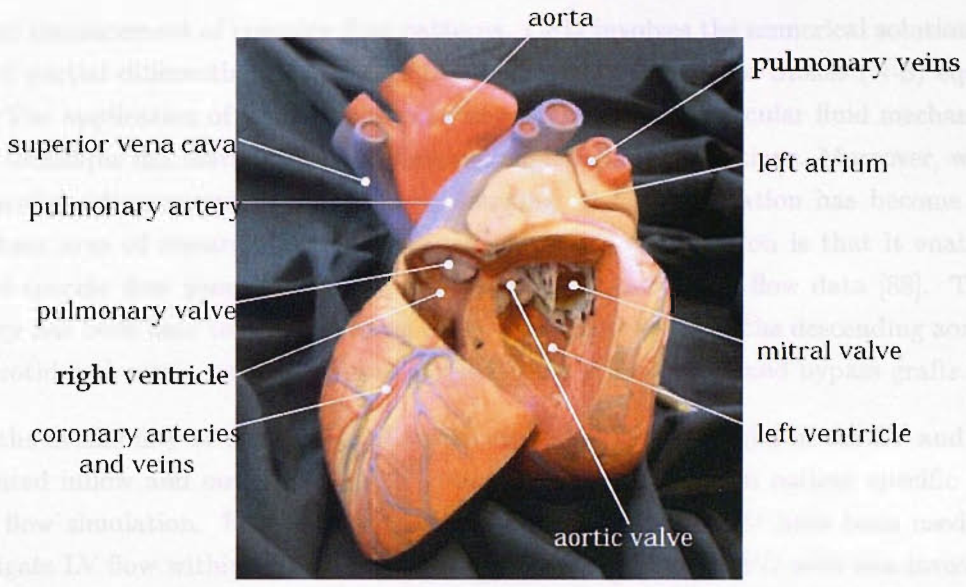


FIGURE 3.1: The anatomy of the heart. Arteries marked in red carry oxygen-rich blood from the lungs to the heart and on to the body, and the ones in blue carry oxygen-poor blood from the body to the heart and on to the lungs. Photo courtesy of Sharmeed Masood, Imperial College, University of London.

Heart Function

The heart acts as a blood pump through the processes of the cardiac cycle. This cycle consists of three basic events, LV contraction, LV relaxation and LV filling. LV contraction, known as systole, occurs when the LV is full and both the aortic and mitral valves are closed, leading to isovolumic contraction. This raises the LV pressure. When that reaches a point where it is greater than aortic pressure, the aortic valve opens resulting in rapid blood ejection. As the LV pressure falls, the aortic valve closes again and the LV enters isovolumic relaxation or diastole. When the LV pressure reaches a point when it is lower than atrial pressure, the mitral valve opens and filling begins.

As we will see in Chapter 6 on Visualisation, large vortical flow structures can be identified during systolic and diastolic phases inside the left atrium and left ventricle. The left atrial rotating flow keeps the blood in motion during the systolic phase when the mitral valve is closed. Vortical flow in the left ventricle is also observed behind the mitral valve, as the entering blood interacts with the existing one forming a vortex ring.

Blood flow patterns are highly complex and vary considerably from subject to subject, even more so in patients with cardiovascular diseases. Despite the importance of studying such flow patterns, this field is relatively immature primarily because of previous limitations in the methodologies involved in acquiring and calculating expected flow details. The parallel advancement of MRI and Computational Fluid Dynamics CFD has now come to a stage that their combined application allows for a more accurate and detailed measurement of complex flow patterns. CFD involves the numerical solution of a set of partial differential equations (PDEs), known as the Navier Stokes (N-S) equations. The application of CFD has become important in cardiovascular fluid mechanics as the technique has matured from its original engineering applications. Moreover, with the parallel advancement of MR velocity imaging, their combination has become an important area of research [184]. The strength of this combination is that it enables subject-specific flow simulation based on in vivo anatomical and flow data [88]. This strategy has been used to examine flows in the left ventricle [205], the descending aorta, the carotid and aortic arterial bifurcation [154], aortic aneurysms and bypass grafts.

With the availability of a detailed 3D model capturing the dynamics of the LV and its associated inflow and outflow tracts, it is now possible to perform patient specific LV blood flow simulation. For many years, techniques based on CFD have been used to investigate LV flow within idealised models. The combination of CFD with non-invasive imaging techniques has proven to be an effective means of studying the complex dynamics of the cardiovascular system as it is able to provide detailed haemodynamic information that is unobtainable by using direct measurement techniques.

In Chapter 6, a visualisation method is implemented and validated in 2D using *in vivo* data, and in 3D using a CFD simulation of the flow of blood inside a model left ventricle.

3.2 Cardiac MRI acquisition

Magnetic excitation and signal detection

Magnetic Resonance Imaging (MRI) depends on the detection of the spin of water and fat protons (hydrogen ^{1}H) in the body. When the protons are placed in a large magnetic field (B_0) generated by a coil, they align with the field's axis and spin around it at a certain frequency. This is given by the Larmor equation:

$$\nu = \gamma B / 2\pi \tag{3.1}$$

where ν is the spin or precession frequency, B is the applied magnetic field and γ is a substance-specific property called the intrinsic gyromagnetic ratio. While the protons rotate around the B_0 axis, another field B_1 is applied. The purpose of this field is to change the orientation of the individual spins. This change in orientation generates a voltage at a receiver coil. The B_1 field is only applied for a short time (milliseconds) and it is perpendicular to B_0. It has a frequency equal to the Larmor frequency of hydrogen ^{1}H at B_0 so that the nuclei will resonate. This resonance has two effects: it enables the weaker B_1 to rotate nuclei under the influence of the much stronger B_0; and it enables selection of ^{1}H nuclei - other nuclei will have different γ and therefore different resonant frequencies and are not excited by B_1. The signal detected with a coil after B_1 excitation has a detectable frequency and decays in amplitude with time. This decay is a result of the nuclei aligning back with B_0. This reversion is a result of two processes:

Spin-lattice relaxation. This involves release of energy to the environment and it takes a time designated as T1. The T1 of the myocardium is about 880 ms at a B_0 of 1.5 Tesla. As a rule of thumb, T1 is short in fat and long in water.

Spin-spin relaxation. This is done through interaction of nuclei in the tissue, and it results in randomisation of the spin phases. It takes a time designated as T2 (75 ms for myocardium). Also time T2* (pronounced "T2 star") includes the spin-spin relaxation plus that which is caused by inhomogeneities in B_0. T2 is also short in fat and long in water.

These different processes are used to generate contrast in MRI. Typically T1 and T2 times are longer with increasing water content, whereas contrast agents affect T1 and T2*. Changes in relaxation properties due to pathology or contrast agents are used to improve contrast by appropriate modification of imaging sequences.

MRI image formation

The goal of an imaging sequence is to determine the X, Y, Z position of spinning nuclei as well as their signal amplitude. The simplest imaging protocol consists of four steps. The following is a highly summarised description of the theory behind each one. Here the Z axis is along B_0, X is along B_1 and Y is the remaining orthogonal axis.

Slice Select. This solves one spatial coordinate by only exciting magnetisation in one slice of the sample. This is done by applying a linear (varying) B_0 gradient, for example from head to toe. The differences in field strength cause spins to have slightly different Larmor frequencies along the gradient. When field B_1 is applied, only a predefined slice of nuclei will be placed in resonance for further modification to create an image.

Phase encode. After the spins have been rotated by slice selection, they are phase encoded by applying a magnetic gradient along the Y axis. This creates another spin rotation which generates a signal that can be read out. However, because of phase effects from the previous step, the position in Y cannot be read from a single phase point. Therefore, repeated measurements must be made to determine the rate of change in phase in these different positions as a function of the Y gradient.

Refocus gradient, frequency encode and readout. This applies a gradient along the X axis so that spin positions along X can be read.

Repeated application of these steps collects data into a matrix in so-called k-space. Each iteration fills phase information into one frequency line in k-space. The final image is generated by applying an inverse Fourier transform to the k-space plot.

Pulse sequences

The *repetition time* TR is the difference between two sequences. It determines the amount of T1 relaxation that is allowed to occur. The *echo time* TE is the difference between application of the B_1 pulse and the peak of the signal induced in the coil (echo). This reflects how much T2 relaxation has occurred. The choice of timings and rotation angles are the determinant factors for image quality, resolution and scan time. There are numerous trade-offs to consider when deciding which combinations to use. For example, TR should be maximised and TE minimised for high signal-to-noise ratios (SNR), but this results in poorer contrast and increased scan time. The latter effect could be important when scanning claustrophobic patients.

The *spin echo* pulse sequence rotates nuclei by 90°. This is followed by a 180° signal to compensate for T2* decay. If short TR and TE are used, fat appears bright and water appears dark because fat nuclei realign with B_0 faster than water. If a second B_1 pulse is applied and TE is now allowed to be long, the opposite occurs because since fat realigns faster, water decay is still taking place even after the long TE.

The *gradient echo* pulse sequence uses a variable, typically short rotation angle. This results in faster scan times but because T2* decay is not eliminated. However, images are more sensitive to inhomogeneities in B_0 and contain artefacts.

Fast spin echo applies several B_1 pulses in one single TR sequence. This fills several lines in k-space per iteration. This reduces scan time but results in lower image quality.

Echo planar imaging (EPI) takes this idea to the limit by filling all the lines in k-space in one single echo train. This demands extremely high switching times from the equipment and requires modified, expensive power supplies in the order of 10,000 kW. EPI is plagued with artefacts, poor SNR and fat signal misregistration. It also raises safety issues because the rapid gradient switching is very close to nerve stimulation thresholds, and strong ear protection is required because of severe noise.

EPI is the current state-of-the-art of pulse sequences targeted by considerable R&D. It has enabled the development of real-time MRI which forms the basis of recent interventional systems such as the General Electric Signa SP. Furthermore, changes in blood oxygenation in response to cortical function result in a rapid alteration of the MR signal which can be measured using EPI. This has enabled the development of functional MRI (fMRI), useful for mapping areas of brain function prior to surgery and task re-learning following brain injury.

There are other pulse sequences, designed to optimise parameters according to factors such as patient characteristics, the anatomy region being scanned, the risk of breathing and other motion artefacts and the type of study being conducted. These are usually programmed into the scanner's control software and can be conveniently chosen by the radiologist, who must however be thoroughly trained in the advantages and drawbacks of each mode.

In cardiac MRI, we are especially concerned with two sources of motion artefact: breathing and heartbeat. Breathing is dealt with by having the patient hold their breath while the scan takes place. Typically while being scanned, the patient is given instructions such as "breathe out, breath in, and hold" followed by "breathe away" after about 15 seconds, hearing these on headphones that he or she is wearing while inside the scanner. The patient also wears ECG electrodes so their heart rate can be monitored by the MRI control computer. This method is called *ECG-gated acquisition* and it enables the scanner to fill in one section of k-space at each cardiac cycle. Because the heart region "looks" the same at the same point of every cycle, the result is a complete k-space plot which looks like a motion snapshot but in effect combines information from several different heartbeats. The disadvantage of this is that the machine is no longer in control of the TR time, which is now called *effective TR* and depends on the variable peak-to-peak difference between heart cycles. For a typical heartbeat, this forces TR to a duration of approximately 600 to 1000 ms, or multiples of that if one or more heart cycles are skipped.

Cine imaging is a modality where scans are taken not only at peaks but also at other points in the cardiac cycle in order to build a whole motion sequence dataset. MRI software can typically produce AVI-formatted files of cine scans. A recently developed technique [272] employed a sequence that determines tissue motion with an initial B_1 pulse. Then it images those sections of the tissue that are static with ungated acquisition, and the moving sections with selective phase encoding. This results in a 35% reduction in scan time.

Because cine MRI captures the entire cardiac cycle, it is possible to make quantitative measurements based on the movement of certain tissue borders, such as the left ventricle. However, finding these borders is generally a tedious manual task requiring the pinpointing of the tissue region and the border in question, with limited automatic support. In Chapter 4 we describe our work on automatic finding of the left ventricle in cine MRI, using a template-based technique.

Velocity MRI

There are two categories of velocity MRI techniques, time-of-flight methods and phase flow imaging methods. Time-of-flight methods rely on the movement of a volume that is magnetically excited. Moving tissue is continuously replaced by the flow, hence velocity can be measured through the rate of decay of the signal intensity inside the same slice [224]. Velocity can also be measured through saturating a band of tissue and then following its progress in different planes [71]. The major limitation of time-of-flight methods is that the contrast of excited blood will reduce with time, eventually making it difficult to measure accurately the distances travelled. Also for cardiac studies where gating is required, only 2D images can be acquired due to the scan timing constraint.

Phase flow imaging methods [77, 129] exploit principles that cause flowing material to acquire a phase shift that is related to its motion. A basic phase velocity sequence is illustrated in Figure 3.2 for a fluid flowing down a tube surrounded by stationary material.

A bipolar gradient pulse is applied, consisting first of a positive magnetic field gradient, followed a certain time later by an equal but opposite negative magnetic field gradient in the direction of the flow. During the period of the positive gradient, flowing and stationary material in a particular location will take up a frequency shift that will depend on their position in the direction of the field gradient. When the gradient is turned off, the phase of the flowing and stationary materials can be considered to be equal. In the period between the positive and negative gradients, the flowing material moves away from its stationary neighbour. During the period of the negative gradient, the stationary material will take up an equal but opposite frequency shift and return to the phase it had prior to the first gradient. However, the flowing fluid will take up a different frequency shift depending on the distance it has moved. Its final phase, therefore, also depends on this distance and hence its velocity.

The images produced by Velocity MRI are inherently noisy and can be potentially misleading due to errors in the flow measurement. Background noise attributed to patient movement during scanning and an uneven magnetic field H_0 is inevitable. A misregistration error may also occur due to the flow of blood between slice selection, the phase encoding and the reading of the signal. The result is that the measured flow on a phase map may be spatially misregistered with respect to an anatomical image acquired of the same plane. This may, for example, result in the blood signal being moved so that it overlays some non-moving anatomy. The resultant phase on the acquired phase map would then be a combination of the flow phase (from blood) and stationary phase (from tissue), thereby reducing the measured flow. Finally, signal loss is another factor that can introduce error into flow measurement. Among other cases, it can occur in zones of rapid flow, when the material moves out of the slice before the pulse sequences are finished. This is manifested by a visible spot of signal blackout in the phase plot.

Some procedures are also added into the sequence to reduce scan time or minimise yet more sources of error. In the Velocity MRI patient data we use in this thesis, several procedures are applied, including k-space segmentation, Maxwell correction and

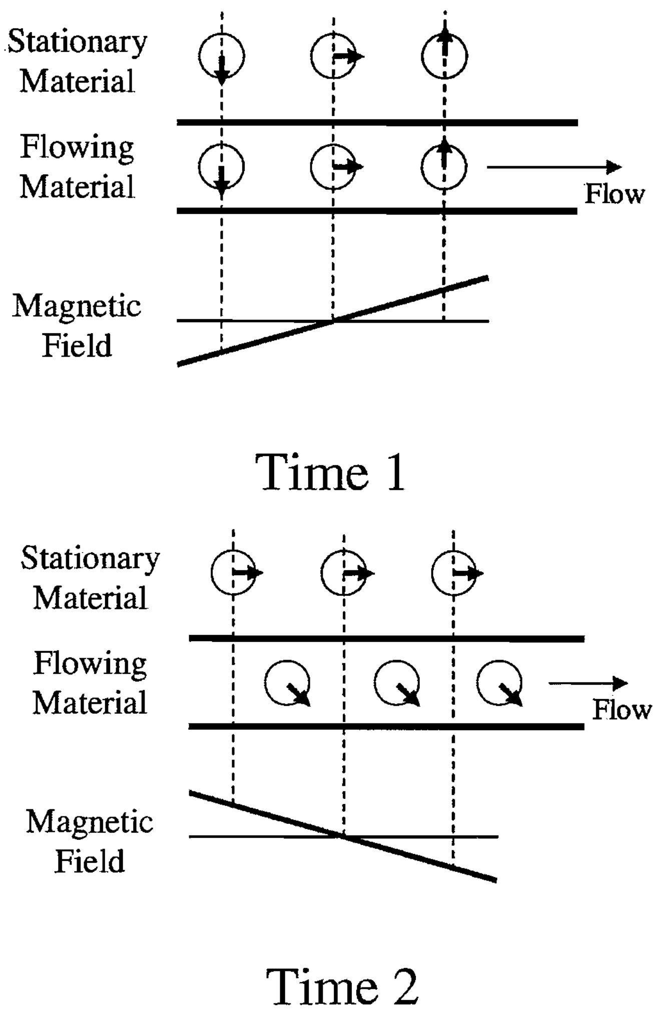


FIGURE 3.2: The principles of phase velocity encoding. At Time 1, a positive magnetic field gradient is applied, which results in an equal frequency and associated phase shift for neighbouring stationary and flowing spins. At Time 2, an equal but opposite magnetic field is applied. By this time, the spins in the flowing blood have moved away from their original neighbours and are now in a different strength magnetic field during gradient application. As a result, the phases of the stationary spins will be returned to zero. The flowing spins will accumulate a phase shift proportional to the distance moved and hence the velocity.

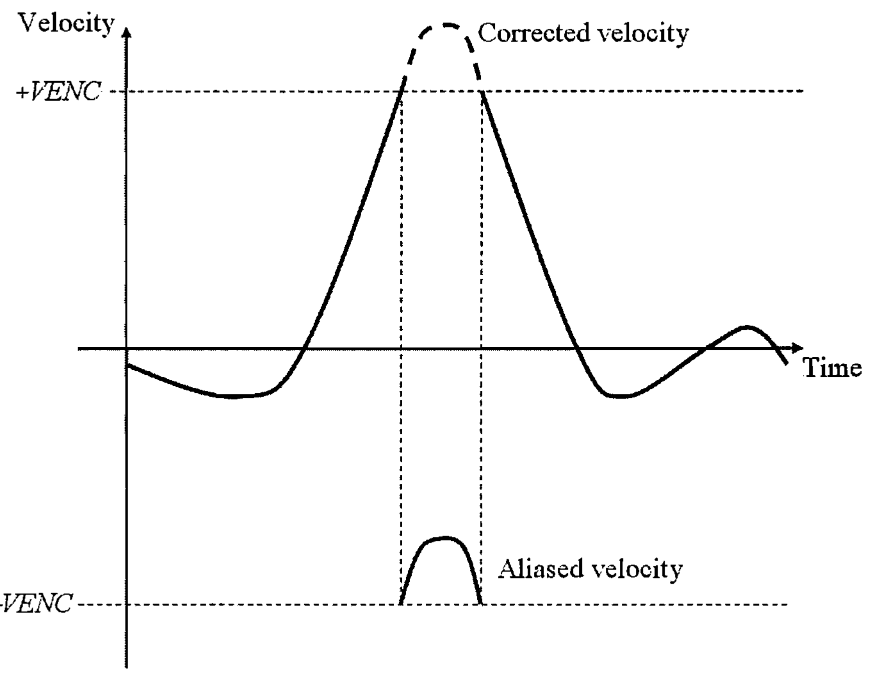


FIGURE 3.3: Phase unwrapping technique: abrupt changes in the velocity over time are detected and corrected for.

temporal phase unrapping. k-space (the frequency domain, see section 3.2) segmentation is a way of sacrificing temporal resolution to get shorter scan times. This is done by interleaving the acquisition of multiple k-space lines within the same heartbeat. Maxwell correction filters out unwanted constant gradient fields. These fields have a nonlinear spatial dependence that can be predicted from the Maxwell equations, which state that the magnetic field must have no divergence or curl. Uncorrected velocity images may display erroneous flow through cardiac walls. Temporal phase unwrapping is linked to the chosen range of velocity encoding (VENC). A small range is desired for resolving the slower flows, however this sometimes cause aliasing problems with faster flow. For example, if the range is from $VENC = -\pi/2$ to $\pi/2$, any flow faster than $\pi/2$ will wrap around to $-\pi/2$. This can be corrected based on the assumption that velocities do not change abruptly between two time frames (see Figure 3.3).

Despite the issues inherent with Velocity MRI, as with any imaging modality, the technique has been validated both *in vivo* and *in vitro* and is now routinely providing useful measurements in clinical and physiological flow studies. For example, in [274] a pulse sequence is used that applied an inverted gradient during the slice select step so that flowing nuclei that are about to enter the selected slice can be discriminated. The data was then processed using flow feature classification. The technique was applied to map cardiac blood flow [130], enabling the authors to postulate that the heart is optimally shaped to maximise output efficiency.

Tagged cardiac MRI

Tagging is a technique that permits tracking of tissue deformation over time. Tagged images are created by a sequence called spatial modulation of magnetisation (SPAMM), which includes a B_1 pulse that induces a sinusoidal variation in nuclei magnetisation. The result is a tagging grid that looks like a black fishnet placed over the scan. The grid has a regular appearance at the beginning, but it then follows the tissue as it deforms. This is useful for clinical studies of heart wall motion, not only by visual inspection but especially by automated analysis, a method that is being researched. A recent review of MRI tagging is given by Reichek [195].

3.3 Other Clinical Assessment Techniques

For assessing heart or cardiovascular disease, a single measure of global cardiac efficiency, including flow and motion components, would be ideal but does not currently exist. The cardiovasuclar system must be considered as a whole, since pressure and flow is dependent on both cardiac function and the condition of the peripheral vessels. A regional description is normally needed in order to localise areas with abnormal function. Currently a set of techniques is used in clinical practice [135]:

Physical examination. A set of procedures conducted directly by the doctor on the patient, including visual inspection, palpation, assessment of arterial pulse, rythm, and auscultation.

Chest X-ray. To assess heart size, calcification and lung vascularity.

Electrocardiography - ECG. A recording of the electrical activity of the heart. The shape of the ECG waveform is similar among healthy subjects. Deviations from the norm can help diagnose specific conditions, when the ECG wave is assessed by a specially trained clinician or analysis software.

Echocardiograpy. Uses conventional 2D ultrasound to assess macroscopic shape, or Doppler ultrasound to assess blood flow.

Nuclear imaging; Positron Emission Tomography (PET) imaging. Images taken immediately after injection of radioisotope markers reflect the flow of blood to the muscular wall, or *myocardium*. PET is also able to estimate blood flow in the coronary arteries, which supply blood to the myocardium.

Cardiac catheterisation. Measures pressure in the heart chambers and related arteries, through the introduction of a thin tube into the circulation.

Computed Tomography - CT. Limited by acquisition speed and the need for ionizing radiation and cardiovascular contrast agents.

Magnetic Resonance Imaging. Synchronisation of MRI scanning with the ECG wave allows for images to be acquired in systole and diastole. This is useful in the assessment of heart shape and contractility. When combined with contrast agents, it enables myocardial perfusion studies. As referred earlier, Velocity MRI obtains images of flow velocity in the heart chambers and some large vessels, but the sampling is insufficient to resolve coronary blood flow.

For Velocity MRI in particular, there is a variety of commercially available software currently in use in hospitals to analyse and quantify the data obtained from a patient scan. For example, the Ducth company Medis supplies the MRI Flow (trademarked) software for delineation of vessels for producing plots of flow velocity and the elastic behaviour of the vessel wall. Their MRI Mass software is used for perfusion and heart wall motion assessments. Other vendors have similar software programs, which have been approved for diagnostic and disease recovery applications employing Velocity MRI data.

3.4 MRI Flow Visualisation

Visualisation of vector fields is generally more complicated than visualising scalar (one-dimensional) fields due to the increased amount of information inherent in vector data. Vector data can be contracted to scalar quantities, for example by computation of vector magnitude, scalar product with a given vector, or magnitude of vorticity. In this case, scalar visualisation techniques such as isosurfaces and volume rendering can be prescribed. Approaches to visualisation of vector fields include iconography, particle tracking, and qualitative global flow visualisation techniques. Section 6.2.2 on page 83 discusses currently available methods of visualising 2D and 3D flows, including feature-based visualisation.

Field topology refers to the analysis and classification of critical (zero-velocity) points and computation of relationships between the critical points of field data. Computation and display of field topology can provide a compact global view of what is otherwise a very large set of data. Techniques such as volume rendering and LIC provide qualitative global views of field topology, as well as critical point detection although the latter technique is too sensitive to noise in acquired flow data.

3.4.1 Streamline Plots

Arrow vectors are the most basic way of displaying a flow field (Figure 3.4(a)). The arrow heads show the direction of the flow, while the arrow colours (or their lengths) are used to depict flow velocity.

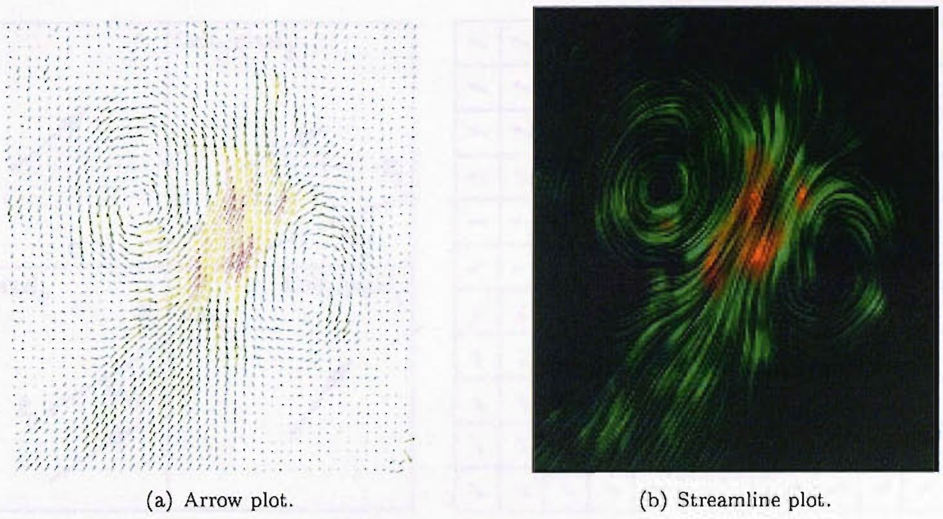

(a) Arrow plot. (b) Streamline plot.

FIGURE 3.4: Depiction of *in vivo* vortical flow inside the human heart from Velocity MRI data.

An alternative to arrows is the streamline plot, popular in fluid dynamics and also applied to MRI [273]. An example of such a plot is shown in Figure 3.4(b) for vortical flow inside the heart. This is sometimes referred to as a *transient streamline* plot, since it depicts one time step of a time-dependent dataset as if it were a steady-state flow. To produce this plot, we implemented Line Integral Convolution - LIC [29]. The method renders an output scalar pixel value for each vector in a vector field. The output pixel value encodes the local vector field by filtering (convolving) a set of input image pixels which lie under the local stream lines. The result is an output image that looks like the input image, but blurred in the direction of the vector field. When the input image is white noise, the resulting output image looks like fine hair running in the direction of the vector field streamlines. The convolution performed by LIC effectively correlates pixels which lie along streamlines and leaves pixels uncorrelated transverse to the streamlines.

The mathematical operation that underlies LIC is the normalised convolution:

$$F'(p) = \frac{\int_{-L}^{L} F(P(p,s))k(s)ds}{\int_{-L}^{L} k(s)ds} \tag{3.2}$$

p is a position vector at which the curve is centered, and s is a scalar distance along P from position p along the flow. A parametric curve, $P(p,s)$, is created in the vector field by locally following the vector field forward and backward for some distance, L. The parametric curve is laid over the corresponding input image pixels, producing a pixel valued function, $F(P(p,s))$. $F(P(p,s))$ is then convolved with a filter kernel, $k(s)$. Finally, the output pixel, $F'(p)$, is normalized by the area under the convolution kernel to produce an even average brightness in the output image.

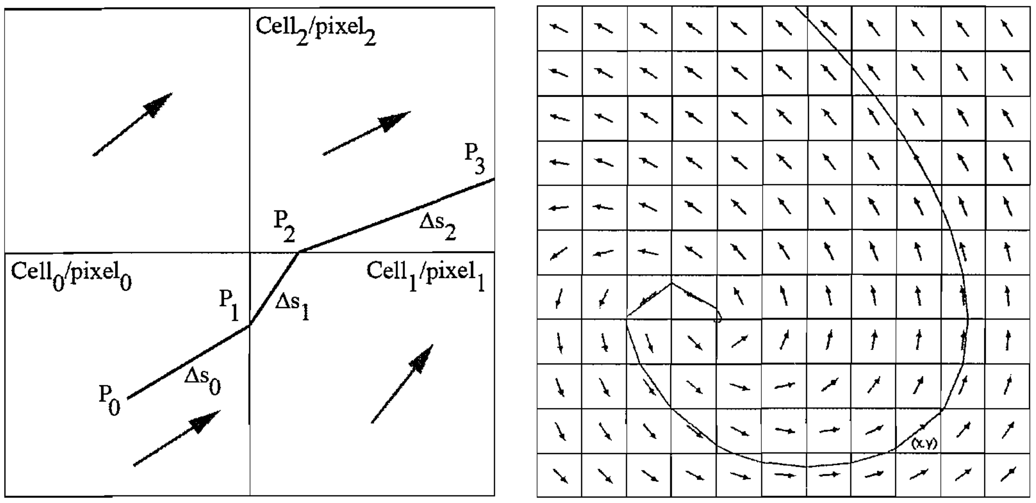


FIGURE 3.5: Variable step Euler's method. *Left,* first three iterations. *Right,* full forwards and backwards iteration for one local stream line starting at pixel (x, y)

To implement LIC and apply it to voxel-based data, a discrete formulation of Equation 3.2 is used,

$$F'(p) = \frac{\sum_{i=-\alpha}^{\alpha} F(P(i))h_i}{\sum_{i=-\alpha}^{\alpha} h_i} \tag{3.3}$$

where P(i) is the i^{th} vector field position along the parametric curve $P(p, s)$, h_i is the analytical area under the portion of the kernel which spans the parametric curve from P(i) to P(i+1), and $2\alpha + 1$ is the number of vector field positions on the parametric curve. The current implementation allows the user to chose whether α should be set to a fixed value, or if it should be a function of the velocity magnitude of the vector at the current $P(i)$. Finally, if we set $h_i = 1$, the formula can be further simplified to:

$$F'(p) = \frac{1}{2\alpha + 1} \sum_{i=-\alpha}^{\alpha} F(P(i)) \tag{3.4}$$

This leaves the problem of finding the streamline path $F(P(i))$. This is achieved here using a variable step Euler's method, as illustrated in Figure 3.5. For the two-dimensional case, the first point centered at pixel (x, y) is defined as:

$$P_0 = (x + 0.5, y + 0.5) \tag{3.5}$$

The method proceeds iteratively by placing each point P_i at the pixel border, taking a ray from position P_{i-1} in the direction of the field vector v_{i-1} at the next pixel:

$$P_i = P_{i-1} + \frac{v_{i-1}}{\|v_{i-1}\|}\delta s_{i-1} \tag{3.6}$$

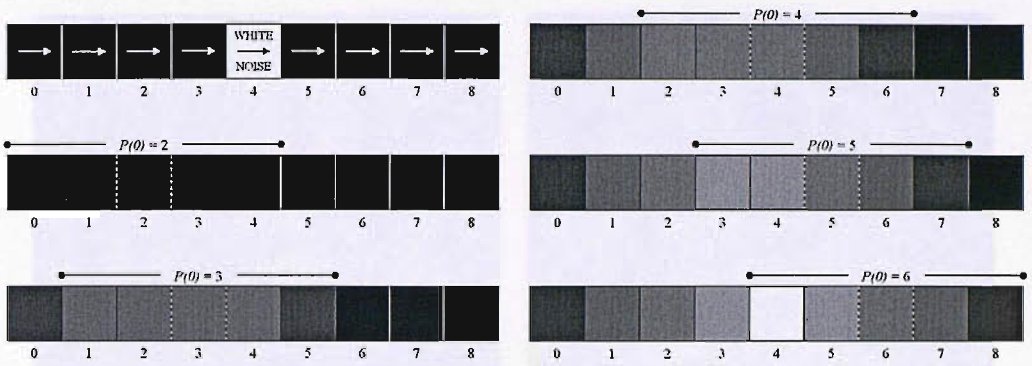


FIGURE 3.6: One iteration of the LIC algorithm. *Top left,* the positions of the velocity vectors and of the source white noise pixel. *Mid left to bottom right,* pixel intensities of the target image as the convolution path progresses over the source noise pixel. The path extent is shown by the bulleted horizontal line, and the path centre $P(0)$ is shown by the broken lines. If a new iteration were started, the path would again be centered at $P(0) = 2$, and the source image would be the final product of the previous iteration, rather than the starting white noise image.

δs_{i-1} is the smallest positive distance along the ray to any of the four sides of the cell:

$$\delta s_{top} = ((y+1) - P_{i-1,y})\frac{\|v_{i-1}\|}{v_{i-1}, y} \tag{3.7}$$

$$\delta s_{bottom} = (y - P_{i-1,y})\frac{\|v_{i-1}\|}{v_{i-1,y}} \tag{3.8}$$

$$\delta s_{right} = ((x+1) - P_{i-1,x})\frac{\|v_{i-1}\|}{v_{i-1,x}} \tag{3.9}$$

$$\delta s_{left} = (x - P_{i-1}, x)\frac{\|v_{i-1}\|}{v_{i-1,x}} \tag{3.10}$$

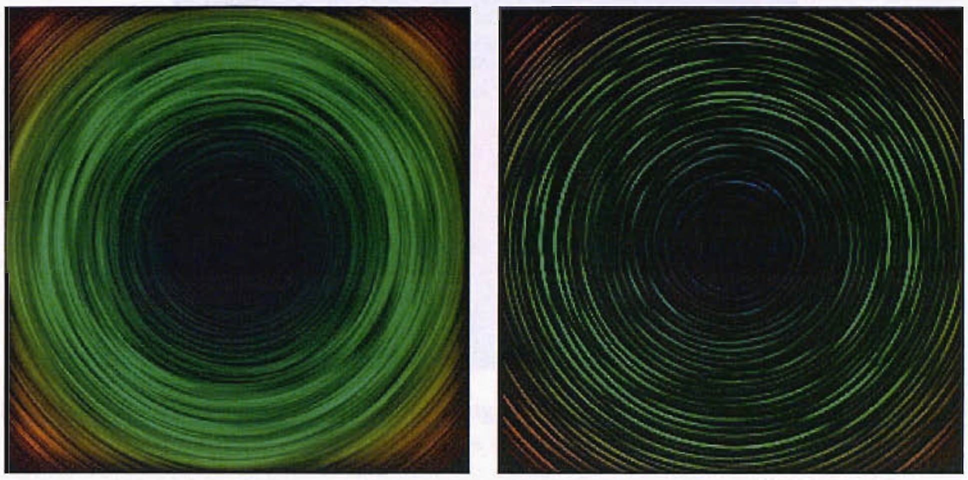
When the divisor becomes zero, $\delta s_{top,bottom,right,left} = \infty$ and in this case δs_i is defined as the positive minimum of $\delta s_{top,bottom,right,left}$.

The local streamline is also iterated backwards by the negative of the vector field. Figure 3.6 illustrates the progress of one iteration as the convolution path passes over a white noise point.

To enhance the linear structure formed by the streamlines, a Lapalacian high pass filter is applied after each iteration [273]:

$$F'_*(p) = F'(p) + \gamma \nabla^2 F'(p) \tag{3.11}$$

where $\nabla^2 = \partial^2/\partial x^2 + \partial^2/\partial y^2 + \partial^2/\partial z^2$. To illustrate this effect, Figure 3.7 shows streamlines through a simulated vortex with no high pass filtering ($\gamma = 0$) and with filtering ($\gamma = 0.1$) over two iterations. In practice, the filter is implemented in 2D with a 5×5 sliding window coefficient array with values:



(a) No high pass filtering. (b) With high pass filtering, γ=0.1.

FIGURE 3.7: Streamline high pass filtering.

$$\begin{array}{ccccc} -1 & -3 & -4 & -3 & -1 \\ -3 & 0 & 6 & 0 & -3 \\ -4 & 6 & 20 & 6 & -4 \\ -3 & 0 & 6 & 0 & -3 \\ -1 & -3 & -4 & -3 & -1 \end{array}$$

Although arrows and streamlines compare well in 2D, for volume data there exists a different rendering situation because the 3D flow elements occlude one another in all directions. This makes it very difficult to render unprocessed flow data.

Our own experience confirms the difficulty of this task. Figure 3.8 shows a volume rendering of 3D streamlines of simulated flow past a circular wire, and as it can be verified the result is not satisfactory. In that dataset, there is vortical flow along the centre of the volume, and this becomes occluded by the linear flow that surrounds it. This justifies the need for higher-level processing of the volumetric data, in order to make visible only the clinically relevant features of the flow and discard the non-relevant ones - a process called *flow abstraction*.

3.4.2 Field Topology

As defined in section 3.4, field topology refers to the analysis of critical points in the vector data. A technique for the visualisation of vector field topology in fluid flows was introduced by Helman and Hesselink [102]. In that work, essential flow information is presented by partitioning the flow field into regions using critical points linked by streamlines. Critical points are defined there as points in the flow where the velocity magnitude is equal to zero. Topology visualisation is effective when applied to flow fields

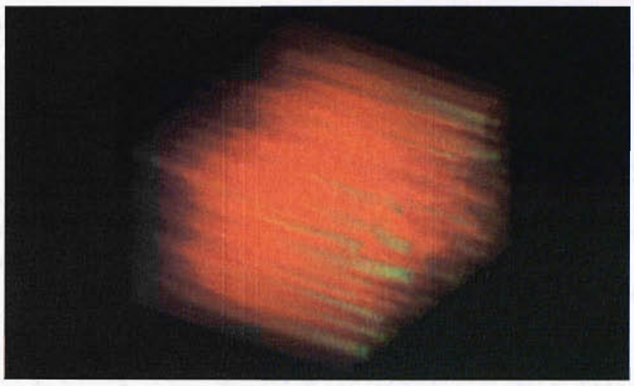


FIGURE 3.8: Transparent volume rendering of three-dimensional streamlines of simulated flow past a curved cylinder.

of limited complexity, however in turbulent flows of some resolution some problems may arise. As fluid flow computations and acquisitions generate more complex and information rich datasets, the set of computed critical points becomes very large. This results in a cluttered image that is difficult to interpret [111]. As it would be expected, we also found in our experiments that a large number of false positive critical points are detected. This has been mitigated in our approach using a clustering technique, as discussed in Chapter 6.

The simplest method of detecting critical points is to look for regions of the flow where the velocity magnitudes are small or zero. Each critical point is classified based on the behaviour of the flow field in the neighbourhood of the point. For this classification, the velocity gradient tensor is used. The velocity gradient tensor, or Jacobian, is defined as

$$\mathbf{J} = \begin{pmatrix} u_x & u_y \\ v_x & v_y \end{pmatrix} \tag{3.12}$$

u and v are the components of the velocity vector (u, v) and subscripts denote partial derivatives. Provided that $\mathbf{J}$ is of full rank, we find that the eigenvalues of $\mathbf{J}$ determine a classification of the solution family into one of six patterns or phase portraits. A representative of each of these classes in which the linear system is canonical is shown in Figure 3.9. The phase portrait of a non-canonical system will be a classification-preserving transformation of one of these. The critical point in question is positioned at the origin of each of these phase portraits.

Thus, the local flow patterns about critical points can be deduced by calculating the eigenvalues of the Jacobian at these points [101, 102, 193]. A vortex core point is expected to have conjugate-pair eigenvalues of the local Jacobian matrix with large imaginary parts compared to their real parts. The phase portrait method thus provides

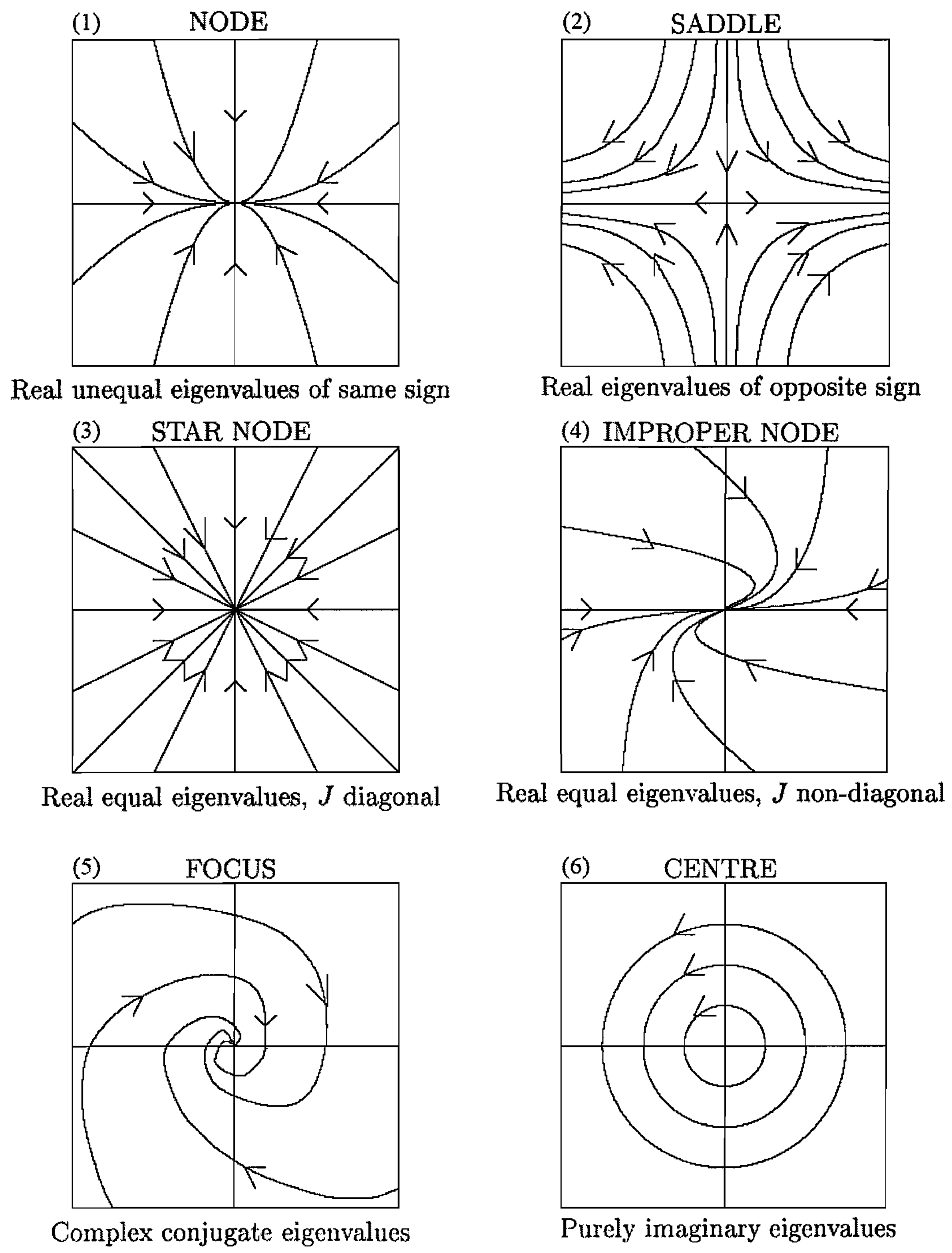

(1) NODE
Real unequal eigenvalues of same sign
(2) SADDLE
Real eigenvalues of opposite sign
(3) STAR NODE
Real equal eigenvalues, J diagonal
(4) IMPROPER NODE
Real equal eigenvalues, J non-diagonal
(5) FOCUS
Complex conjugate eigenvalues
(6) CENTRE
Purely imaginary eigenvalues

FIGURE 3.9: Phase portraits of a simple canonical linear system.

a framework with which to detect vortices. Any implementation of this scheme will be extracting a local feature from a noisy velocity field. It must be able to identify the location of critical points in a velocity field in which noise has corrupted both magnitudes and directions, and to classify these points using the Jacobian even though noise will corrupt the determination of the required velocity derivatives.

Critical point detection

As referred above, searching for points with small or zero velocity magnitude as a means of finding critical points is prone to errors due to noise. The Jacobians can be used to evaluate points in the field to detect critical points, as they enable the extraction of a linear model containing a critical point that describes the local flow. However, many false critical points are detected in this way and must therefore be tested.

A Hough transform-like method has been used to perform this validation [194]. The flow area is first quantised and used as a 2-dimensional histogram that records the probability of a point being a critical point of the field. In fact, 3 such histograms are created, one for each of the possible phase portraits of node, saddle and focus/centre. A window W is then passed over each pixel in the flow image, and a minimisation approach is used to calculate both the Jacobians and the critical points associated with the linear fitting. At each step, the eigenvalues of the Jacobian are used to classify the possible phase portrait of node, saddle or focus centered on the hypothesised critical point. The corresponding two dimensional histogram for the deduced phase portrait is updated at the location that contains the hypothesised critical point according to how closely the deduced linear model fits the window W. Once this has been repeated at all pixels in the image, each of the histograms is thresholded at 50% of the maximum over all three of them. This will leave only areas that have strong support for being a critical point of a particular phase portrait. The centres of the connected components of a particular classification are then taken as the critical points associated with that phase portrait.

The updating scheme incorporated by this approach effectively rejects spurious critical points identified due to non-linearity of the window region, because it is unlikely that these points will be supported by Jacobians at other points in the field. However, it will extract critical points associated with piecewise linear flows if the region of linearity is large which gives the algorithm a robustness to occlusion of the flow image. But because the updating scheme does not incorporate a measure of the relative locations of the piecewise linear region and the hypothesised critical point, this may result in the extraction of spurious critical points.

The above technique is computationally expensive as it requires the solution of a least sum square problem at each point in the field to produce the original estimate of critical point location. To reduce this burden, researchers have investiagted simpler ways of estimating critical point locations that are decoupled from the linearisation. These techniques have been developed for critical point detection in oriented fields which only contain directional information.

One such method is to use the concept of isoclines, line segments in the field along which the orientation angle of the field is constant. It has been shown that in a linear field, a line segment is an isocline if and only if it passes through the single critical point of the field.

Shu [218] use this criterion by classifying points as critical if they lie at the intersection of the isoclines of the field. The difficulty here is the robust identification of the isoclines, especially in the presence of noise or multiple critical points. Ford [80] showed that the constant angle along an isocline of a linear field monotonically increases or decreases as the inclination of the isocline to the vertical increases. This is used to identify potential critical points by calculating, for each point in the image, a monotonicity measure of the field orientation along line segments of a certain length at different angles to the vertical that passes through the point. The monotonicity measure is weighted to favour arguments in which the lines have approximately constant orientations along them, therefore being more likely to be isoclines. This dispenses with the need for prior identification of the isoclines. The measure is thresholded and points that exceed this threshold are labelled as potential critical points. In the study, connected regions identified as the potential critical points tended to cluster near true critical points. Each of the points in the clusters is then checked as a valid critical point. This verification step consists of estimating the Jacobians and critical point positions by least sum square minimusation on circular windows of different sizes up to a predetermined radius around the point. The point is rejected if the Jacobian classifications are inconsistent among the windows of different sizes, or if the estimated critical point position for any window is above a predetermined distance away from the point on which the window is centered.

Winding indices have also been used to identify critical points in 2D [80, 274, 271]. The winding index is a measure of the accumulated directional change in a vector field along a closed path C that encloses the point under consideration. In discrete form:

$$W_{index} = \frac{1}{2\pi} \sum_{i=1}^{N} \phi_{i+1} - \phi_i + \psi(\phi_{i+1} - \phi_i) \tag{3.13}$$

where N is the total number of vectors that lie along C, ϕ is the angle in radians that the ith vector makes with the x axis (with $\phi_{N+1} = \phi_1$ and ψ a phase-wrapping operator defined as

$$\begin{aligned} \psi(\theta) &= 0, & -\pi < \theta < \pi \\ \psi(\theta) &= 2\pi, & \theta < -\pi \\ \psi(\theta) &= -2\pi, & \theta > \pi \end{aligned}$$

confining the angular difference between adjacent vectors within the range $(-\pi, \pi)$. The choice of the closed integration path is generally arbitrary. The value of W_{index} will be ± 1 if the path contains a single critical point and 0 otherwise. In practice, if a circular path is chosen to calculate the winding indices at each point in the field, circular clusters tend to form around the critical points. In our own experiments, many false positives were generated which were eliminated by a winding index thresholding and cluster detection procedure. Critical point locations can then be taken as the cluster centroids, or as an alternative method [274] magnitude information can be incorporated into the detection by linearising the flow over the cluster points by least squares minimisation and computing the critical points from the determined Jacobian.

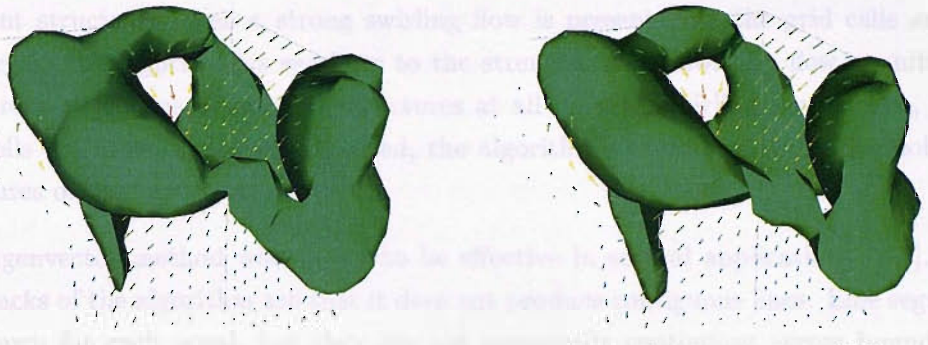
The winding index is indeterminate if a critical point lies on a path, or the region enclosed by the path contains more than one critical point. Thus if winding indices are calculated about each point in a velocity field using a circular path of radius r centered on the point, it is only possible to infer the presence or otherwise of a critical point within the circular region if the critical points in the field are at distances greater than $2r$ from each other. From this it follows that the preferred path radius is the smallest possible, however for very small paths the effect of noise on winding index estimation is considerable. The winding index method can also fail to detect occluded critical points, unlike the isoclines method. However, it does not rely on local linearisations of the field which makes it comparatively quick to compute.

Another method is to find regions with a high *vorticity* magnitude. It should be noted that although a vortex may have a high vorticity magnitude, the reverse is not necessarily true [278]. Vorticity is defined as the curl of the velocity field,

$$\nabla \times \boldsymbol{v} = \left(\frac{\partial v_z}{\partial y} - \frac{\partial v_y}{\partial z}\right)\hat{x} + \left(\frac{\partial v_x}{\partial z} - \frac{\partial v_z}{\partial x}\right)\hat{y} + \left(\frac{\partial v_y}{\partial x} - \frac{\partial v_x}{\partial y}\right)\hat{z} \tag{3.14}$$

and represents the shear and circulation characteristics of the original vector field. An algorithm has been presented for constructing vortex tubes using this concept [254]. The method computes the average length of all vorticity vectors contained in small-radius cylinders, and use the cylinder with the maximum average for constructing the vortex tubes. The *helicity* can also be used instead of vorticity [145, 275]. The helicity of a flow is the projection of the vorticity onto the velocity, that is $(\nabla \times \boldsymbol{v}) \cdot \boldsymbol{v}$. This way, the component of the vorticity perpendicular to the velocity is eliminated.

Another issue involving the phase portrait technique derives from the need to arbitrarily set an imaginary part versus real part threshold ratio, or an imaginary part threshold minimum. A technique has been reported [117] that does not require thresholding to give results superior the Jacobian eigenvalue formulation. The method defines vortices as regions where two of the three (real) eigenvalues of the symmetric matrix $S^2 + \Omega^2$ are negative. S and Ω are the symmetric and antisymmectric part of the Jacobian of the vector field, respectively. The same work also lists some other definitions of a vortex and extensively discusses counterexamples where those methods fail, including vorticity and minima of pressure. The method was used for vortex core rendering in several works, for example [167]. A criterion involving negative second eigenvalues suggests counting hits by setting an upper threshold of zero. In practice, this threshold is set slightly above or below that value, typically $-\frac{1}{2}$.


(a) Imaginary eigenvalue. (b) Local symmetric tensor [117].

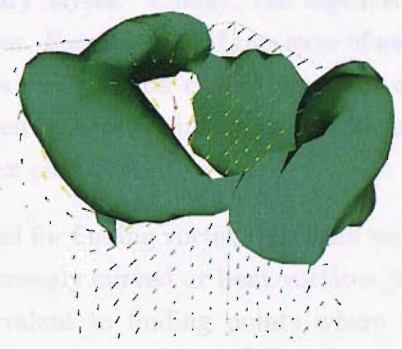

(c) Vorticity.

FIGURE 3.10: Vortex measure isosurfaces, plotted with respect to manually set thresholds. Image local gradients were computed with local linear expansion (section 6.3.2).

In all the 3D methods where it is necessary to set a threshold value to isolate the vortex region, none of the definitions enable the pinpointing of the vortex core (Figure 3.10).

A predictor-corrector algorithm for finding vortex cores has been developed [8]. After initialisation, vortex cores are tracked by predicting in the direction of the vorticity vector and correcting to the pressure minimum in the plane perpendicular to that vorticity vector. Then, vortex tubes are created by computing cross-sections of the vortices, in a plane perpendicular to the vortex core. A threshold of the pressure is used as a selection criterion, in combination with the restriction that the angle between the vorticity vector at any point on the cross section and the vorticity vector at the vortex core is no more than 90°.

An algorithm for finding the centre of swirling flow in 3D vector fields has been implemented [229]. The algorithm is based on critical-point theory and uses the eigenvalues and eigenvectors of the velocity gradient tensor, or rate-of-deformation tensor. The algorithm works on each point in the data set separately, making it suitable for parallel processing. The algorithm searches for points where the velocity gradient tensor has one real and two complex-conjugate eigenvalues and the velocity is in the direction of the eigenvector, corresponding to the real eigenvalue. The algorithm results in large coherent structures when a strong swirling flow is present, and the grid cells are not too large. The algorithm is sensitive to the strength of the swirling flow, resulting in incoherent structures or even no structures at all in weak swirling flows. Also, if the grid cells are large, or irregularly sized, the algorithm has difficulties finding coherent structures or any structures at all.

The eigenvector method was shown to be effective in several applications [127]. The drawbacks of the algorithm are that it does not produce contiguous lines. Line segments are drawn for each voxel, but they are not necessarily continuous across boundaries. Another problem is that flow features are found that are not vortices. Instead, swirling flow is detected, of which vortices are an example. However, swirling flow also occurs in the formation of boundary layers. Finally, the eigenvector method is sensitive to other nonlocal vector features. For example, if two axes of swirl exist, the algorithm will indicate a rotation that is a combination of the two swirl directions. The eigenvector method has successfully been integrated into a finite element solver for guiding mesh refinement around the vortex core [60].

A method has been developed for finding vortex core lines using higher-order derivatives, making it possible to find strongly curved or bent vortices [201]. They observe that the eigenvector method is equivalent to finding points where the acceleration is parallel to the velocity. They improve the algorithm by defining vortex cores as points where the deriative of the acceleration with respect to time is parallel to the velocity vector, i.e. points where the torsion is null. The method involves computing a higher-order derivative, introducing problems with accuracy, but it is found in that work to perform well. In comparison with the eigenvector method, this algorithm finds strongly curved vortices more accurately. The method also introduces two attributes for the core lines: the strength of rotation and the quality of the solution. This makes it possible for the user to impose a threshold on the vortices, to eliminate weak or short vortices.

A different approach for detecting vortex core regions has been presented [118]. The algorithm is based on Sperners lemma in combinatorial topology, which states that it is possible to deduce the properties of a triangulation, based on the information given at the boundary vertices. The algorithm uses this fact to classify points as belonging to a vortex core, based on the vector orientation at the neighbouring points. In 2D the algorithm is simple and has only linear complexity. In 3D, the algorithm is more complex, as it first involves computing the vortex core direction, and then the 2D algorithm is applied to the velocity vectors projected onto the plane perpendicular to the vortex core direction. Still, the 3D algorithm has also linear complexity.

Two geometric methods for extracting vortices in 2D fields have been presented [207]. The first is the *curvature centre* method. For each sample point, the algorithm computes the curvature centre. In the case of vortices, this would result in a high density of centre points near the centre of the vortex. The method works but has the same limitations as traditional local methods, with some false and some missing centres. The second method has been inspired by the *winding-angle* method [190]. The method detects vortices by selecting and clustering looping streamlines. The winding angle α_w of a streamline is defined as the sum of the angles between the different streamline segments. Streamlines are selected that have made at least one complete rotation, that is, $\alpha_w \geq 2\pi$. A second criterion checks that the distance between the starting and ending points is relatively small. The selected streamlines are used for vortex attribute calculation. The geometric mean is computed of all points of all streamlines belonging to the same vortex. An ellipse fitting is computed for each vortex, resulting in an approximate size and orientation for each vortex. Furthermore, the angular velocity and rotational direction can be computed. These attributes have been used for visualising the vortices. (See Figure 6.7 on page 92.)

Field Pre-Processing

In our present application, we are interested in detecting, tracking and displaying vortical flow inside the heart. However, several issues need to be addressed. Hardware setup and patient movement generate noise that corrupts blood flow velocity images. It is necessary to reduce this noise in order to improve the accuracy of subsequent data processing stages. This task is called *restoration*. The volumetric data can then be processed and its relevant features extracted in the already mentioned *abstraction* step. Since cardiac MRI produces a time sequence of volumetric images, the problem also exists of labelling and corresponding the same features over different time frames, a process known as *tracking*.

Our implementation of restoration is detailed in Chapter 5. The final chapter, *Conclusions and Future Work*, discusses possible directions for implementating the abstraction and tracking steps.

Chapter 4

Left Ventricle Detection with Template Matching

When applied to the diagnostic of cardiac disease, MRI enables precise and reproducible quantification of global and local ventricular function. As discussed in section 3.3, ECG-gated MRI magnitude data (without velocity) can be used to assess the overall health status of the heart. One common application involves the assessment of volume, mass, wall motion, wall thickening and dynamic parameters associated with the left ventricle (LV). In such a study, MRI frames are captured at different stages of the cardiac cycle as slices perpendicular to the LV long axis. Figure 4.1 shows an example of a typical image used in those studies, showing the LV and other anatomical features.

FIGURE 4.1: Cardiac MRI magnitude image showing several anatomical features of the heart, including the lung (black region, middle right), the right ventricle (bright central region) and the left ventricle (two concentrical ellipses near the centre).

Clinical heart assessments of this type normally require about 200 images to be acquired and analysed. Quantitative image analysis is routinely supported by established software packages, such as MASS [247], which provide the clinician with useful measures. These are derived given the location of the LV borders in each frame. Unfortunately, this quantitative analysis still relies on manual tracing of contours in the images, which is a specialist, time-consuming and tedious procedure that limits the clinical applicability of cardiovascular MRI [248].

There are several published techniques on automatic border delineation, but these require previous, manual detection of LV location in order to work. A contour detection system using fuzzy logic and dynamic programming [136] required the user to indicate a point near the centre of the LV. An edge detection-based system also required the user to indicate a region of interest in addition to the centre point [79, 78]. A border detector based on motion and deformation tracking required the user to delineate initial contours as two concentric circles [271] and motion and misregistration correction is applied so that the LV is centred at approximately the same image location on subsequent frames.

Those studies that do employ automatic location detection indicate that the methodologies followed could be improved. In a Hough transform-based system [246] (also used in [168]), the LV failed to be detected in 12% of the images in that study. In ref. [67], where the grey-level appearance across four cross-sectional lines is learned and used for discrimination, a detection rate of 98% is reported but at the cost of 10 false alarms per image. In ref. [228], successive Active Appearance Model optimisations are performed at different image locations to search for the LV, but it remains unreported whether or not this scheme is always successful. Fuzzy clustering was also used to segment cardiac MRI frames [24], with the LV being found by a combination of relative position information and grey level thresholding.

In this section we report on our work on automatic detection of the left ventricle. We designed a template for LV detection, which is then used to search for the LV across the image. Computation efficiency improvement is achieved through downscaling of the original image. For comparison, we also implemented the Hough transform for circles [246].

4.1 Previous work on medical image segmentation

Border detection

The LV problem is approached here as an object detection problem. Object detection is defined as the determination of the location and scale of every instance of an object contained in an image, surrounded by an arbitrary background. This is different from object recognition, where several object classes need to be discriminated. The detection problem therefore needs to be able to deal with several types of instances of the same object, without mistaking it for the background.

There is a significant literature in the domain of boundary-driven methods. These methods are based on the generation of a boundary image and the extraction of a boundary structure that best accounts for the boundary information [33, 59]. For the problem of finding organ contours in an image, such an approach is insufficient because there is no knowledge of which edges are part of the border of interest. Also in many cases there are missing borders, or gaps, because of noise, tissue magnetic properties, motion blur and other scanning limitations. When hospital operators were observed manually segmenting the LV in MRI frames as a preliminary part of this study, they often drew borders across homogeneous regions based on their knowledge of where the actual physical border should be. An even more acute problem was observed with ultrasound, as more borders are missing from the image as a result of the angle they make with the ultrasound probe being too high for an echo to be received.

The evolution of boundary-based medical image segmentation techniques have led to a paradigm where an initial structure or set of points is deformed towards the desired image-dependent characteristics. The segmentation inevitably depends on the parametrisation of the initial structure, including positions and number of control points. Also a major issue is the re-parametrisation of the evolving structure, which sometimes allows multiple control points joining to a single point. The segmentation performance is therefore linked to the actual implementation of the method, in addition to the method's principles *per se*.

The snake model [126] is the basis of a large number of boundary-driven image segmentation techniques. The model follows an energy minimisation approach that seeks the lowest potential of a curve-based objective function. This function contains two terms that balance the weight of a chosen measure of the boundary, and the chosen internal properties of the curve. For example, one may want to balance the total length of the curve with its curvature and proximity to image edges. The structure to be recovered then refers to a set of points that is deformed locally towards the desired image characteristics, while being constrained with respect to the desired internal properties. This approach has given promising results, but is very sensitive to noise in the data. Deformable templates [236, 277, 164], parametrised snakes [20], B-spline snakes [50] and active shapes [43] were proposed as variants to the original framework. The use of these methods can improve the segmentation performance if the original model is able to cover most segmentation solutions. However, these models, like the original snake, are very sensitive to the initial conditions and have as a result been referred to by the community as "myopic". Propagation is performed according to the local information and therefore the initial curve has to be in the close vicinity of the optimal solution. During the preliminary study for this work, hospital operators were observed attempting to use snakes using commercially available software based on snakes, and taking just as much time setting up the initial snake and correcting it afterwards as they did with fully manual segmentation. The main problem was that the particular snake formulation used in the software was attracted to strong edges, which sometimes made the snake converge away from the actual border towards a stronger one, having been initialised exactly over that correct border.

There are many published variations of the classical formulation. None of them are entirely successful at solving the basic problems of stability and attraction to local maxima. An extra force term was proposed that makes the snake inflate like a balloon [41, 236]. This principle was applied to contour tracking in ultrasound images of heart ventricles and the carotid artery [87]. The scheme aims to prevent the balloon snake from inflating through missing borders, and from stopping too early due to getting stuck in noise regions. A multiscale optimisation approach was presented to improve the resolution of snake delineation in ultrasound images [166]. Contour tracking in animation sequences, particularly images of the heart walls, has been studied in some detail using snake-related solutions. However, the techniques proposed [166] emphasise robust tracking at the expense of accurate delineation.

There is a large amount of published and ongoing research aiming to improve snakes' performance and address its various issues. For example, a dynamic programming solution was introduced to search for equilibrium [1] and later extended [267]. In snake growing [16], a primary snake later divides into smaller pieces; those with high energy are eliminated, while those with low energy are allowed to grow. Stability is achieved because the low energy sections are used to initialise the snake growing in later iterations, whereas high energy ones become stuck in local minima. Another scheme was proposed [92] by which two contours deform towards the boundary: one that contracts an initial estimate from the outside, and another that expands from the inside. Local energy minima are rejected by comparing the contours' energies at each step. A different dual active contour model [34] was used to track epicardial and endocardial borders. The rough border position is given by user initialisation; a dual contour is applied and optimised there. A systematic trial compared the algorithm's performance with manual border tracking and the authors judged the correlation to be satisfactory.

Snakes have been studied extensively in medical imaging to track heart motion. A B-spline snake optimised with dynamic programming [1] was used to track tagged cardiac MRI images. Snakes have been applied to brain and lung edge-detected MRI without a smoothness assumption. The aorta has also been tracked [204]; a rough estimate of the location and diameter of the aorta is obtained using the medialness and boundariness operators defined by Morse *et al.* [171]. No user interaction is required at any point. In the trial, they found that the algorithm's error rate was lower than inter-observer variability in manual segmentation. They also compared different optimisation techniques: simulated annealing, iterated conditional modes, dynamic programming and the greedy algorithm. They made the intriguing observation that the result of the segmentation varies with the optimisation technique used and was best with their implementation of simulated annealing. Still, the model penalises high-curvature shapes, and the authors suggest using stronger *a priori* knowledge.

Level sets are a common implementation of variational frameworks [181, 216, 180]. An evolving contour is represented using a continuous function of a higher dimension that is defined in the image/volume plane. The use of level sets to evolve interfaces has led to an expansion of boundary-driven methods for image segmentation [238, 276, 157]. Global intensity information has been integrated with boundary-driven flows, resulting in segmentation procedures reportedly independent from initial conditions [220]. They still suffer, however, from noisy or incomplete data. Anatomical information has also been integrated in the form of prior shape knowledge that is available from physiology regarding the medical structures to be segmented [187, 185, 186, 280, 281, 214, 89].

Livewire is a popular contour detection technique, simpler to implement and more reliable than snakes because the user's involvement is distributed more evenly throughout the algorithm's execution [173, 75]. In the basic livewire approach, an image frame is defined as a directed, weighted graph with nodes at pixel corners and arcs along pixel edges. Boundaries are constructed by assigning boundary elements to arcs, forming a chain of these elements. Each element is also assigned a set of feature values that characterise its resemblance with a typical boundary. Costs are then derived from expressions that combine different features. Boundaries are found by minimising the sum of these costs. A typical usage consists of these steps:

1. User picks an initial point on the boundary.
2. All paths with minimum cost between initial point to all other points in the image are computed.
3. User places cursor next to border, some distance away from initial point.
4. A livewire border is displayed along the minimum cost path. This should coincide with the border; if not, the user must position the cursor closer to the initial point or closer to the boundary.

The above process is repeated every time the user clicks the cursor on a position, which causes the current border segment to be accepted and a new segment to be started at the clicked position. The procedure ends when the user closes the border.

To implement livewire, the cost features need to be chosen, a training procedure needs to be designed and a suitable path following algorithm must be chosen. Typical features include the grey level intensities, oriented gradient magnitudes, and distance from boundary in the adjacent slice in the case of 3D data. These features work well for MRI applications, but not on ultrasound data because of noise, speckle and other artefacts.

Training consists loosely of setting the weights and transforms with which features are to be combined together. This can be done previously, in preparation for the segmentation phase, or it can be done dynamically, learning from the user's corrections. Training can also be designed for 3D imaging so that the parameters learned in adjacent slices can be used for the current slice [9]. Finally, the preferred path finding algorithm is typically Dijkstra's algorithm, a variant of dynamic programming, which finds the optimum path between any two vertices of a graph. The optimum path is such that the sum of the costs of all its boundary elements is the lowest of all possible paths.

Livewire has been applied most successfully to 3D brain MRI segmentation. The stronger user control and lack of smoothness constraint makes it a more reliable tool than snakes, but the promised speed-up compared to manual segmentation is lower. A software package exists [241] that provides 3D rendering of surfaces that were previously segmented slice by slice. Another scheme prompts the user to pinpoint landmarks for fast 3D propagation [74]. Finally, progressive livewire [160] used livewire for automatic segmentation of each slice and snakes for contour propagation between slices.

Object detection

Object detection is commonly approached with a sliding window operator, which computes a set of local measures and computes the probability that a pattern can be classified as the object of interest. Such a scheme can be tailored to include the object's appearance (grey scale/colour pattern), shape, and relationship to other objects in the scene. Earlier methods based on specific object/domain features have been applied in medical imaging [162, 66], automatic target recognition [18] and building segmentation [219]. Such object-specific methods depend on the available knowledge about the object and the ability to translate this into a working system. Another common approach is to devise general methods that can be trained to detect an arbitrary object class. These require more limited prior knowledge about the object of interest, and can learn from a set of training examples of the object of interest. Shape-based systems can be object-specific [14, 32, 244] or they can learn the shape characteristics of the object of interest from training examples [143, 2]. These methods are usually sensitive to occlusion, noise and feature point misdetection. Many are also limited to the detection of rigid objects and may not work in case of non-linear variations in the shape of the object of interest, something which does occur in the present problem of LV contraction/distension. Shape learning approaches cope well, however, with large intra-class variations, for example in human faces, although this comes at the cost of an increased false positive count.

Active shape models aim to be fully automatic, and also require training and an initial estimate [46, 47]. The shape model is based on landmark points being designed and their coordinates obtained through training. After alignment of the training shapes, a mean shape is calculated from the trained positions of the landmark points, and a covariance matrix is derived. Eigenvalues and eigenvectors are extracted from that matrix. These capture the modes of variation of the shape dataset. A generative model then incorporates these trained statistics, and produces arbitrary shapes that deform with varying parameters for each eigenvalue. Cootes *et al.* [106, 105] reported successful application of active shapes to brain MRI images. One drawback of this is that it assumes an elliptic distribution of the parameters for the computation of the modes of variation. That assumption may not be valid for real word shapes, where the distribution is often not elliptic. This creates difficulties in dealing with new shapes as the system requires that they must lie inside this elliptic distribution, which is often not the case. Another drawback is the choice of landmark points, which is done manually requiring some care. The team has repeatedly written "we are developing tools to ease the procedure" [47, 46, 45] but so far the only progress has been a very limited automatic method [106] based on shrinking a cylinder with pre-defined landmark points around the volume of interest, and a variant of that [104].

Appearance-based object detectors have also been introduced. An object's appearance depends on its shape, reflectance, pose and illumination conditions and can be identified trough the pattern of grey levels or colour values in the object and immediate neighbourhood [174]. The majority of implementations are based on processing of input grey level information [15, 230, 42, 146, 148] but some more recent approaches [211, 2, 44, 137] combine them with some shape features. Many of these systems have only been applied to face detection. Also, it is difficult to compare the performance of these systems since they were not tested on a common dataset. In general, appearance learning scheme follow the following paradigm. Several windows are placed at different positions and scales in the test image and a set of low-level features is computed from each window and fed into a classifier. Typically, the features used to describe the object of interest are the "normalised grey" level values in the window. This generates a feature vector with a large dimensionality, whose classification is both time consuming and requires a large number of training examples to overcome the "curse of dimensionality". The main difference between these systems is the classification method: probabilistic measures [169, 146], neural networks [148, 202, 264], Markov chains [42, 67], support vector machines [182, 183], trees [2] and prototype distances [230]. One of the main performance indices used to evaluate such systems is the detection time - most detection systems are inherently slow, since for each window position, a feature vector with large dimensionality is extracted and classified.

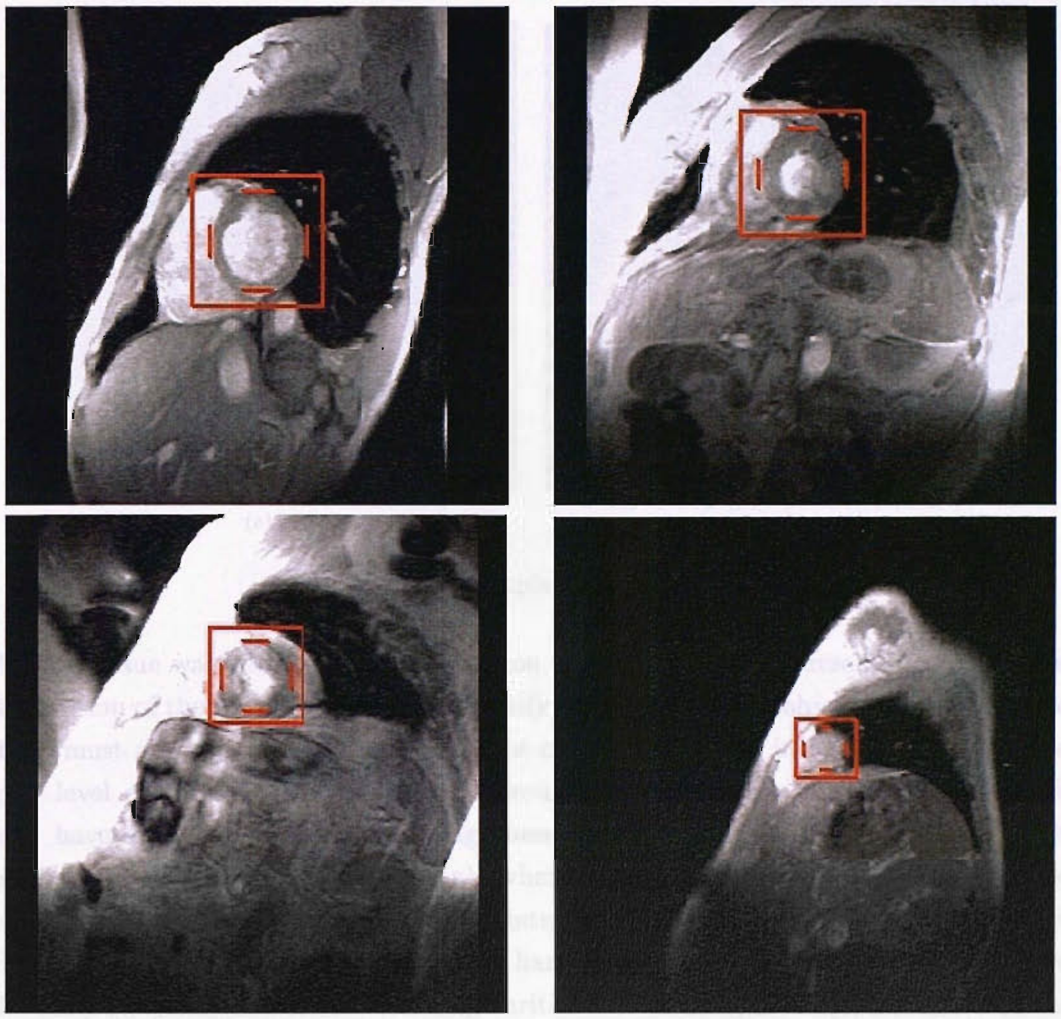


FIGURE 4.2: Variability in the appearance of the left ventricle, in scans from different patients.

4.2 Detection algorithm design

4.2.1 System requirements

The object detection problem is generally considered to be extremely challenging. The present work had to address several general issues, while making a useful contribution towards LV detection. As can be seen from Figure 4.2, there is a large variability in the appearance of the LV between different image instances. There are two main reasons for this: inter-patient variability, because everyone's hearts have different shapes and sizes, and the natural heartbeat that naturally deforms the heart walls. There are also undesirable factors to take into account, including different scan settings that may affect the scaling, contrast or brightness, patient position and breath hold that may affect the LV perspective and position in the image, and inevitable image noise from the electrical equipment and the chosen scan sequence.

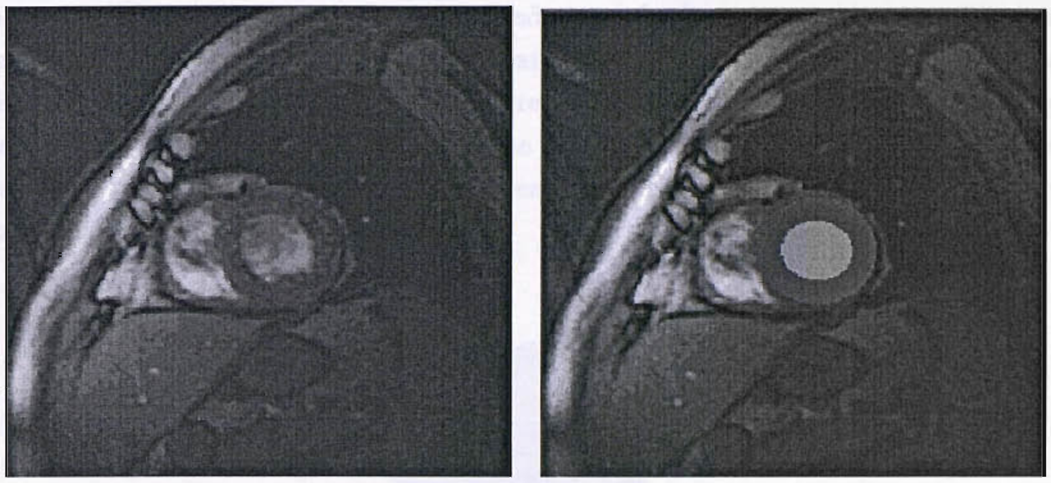

(a) (b)

FIGURE 4.3: Template matching steps

Another issue was what type of information to use for object representation. A useful description of the LV must be able to identify an instance of the object when exposed to it. It must also reject instances of different objects, especially in our application those grey level patterns that occur in the surrounding areas of the image, some of which may have very similar shape and brightness characteristics to the LV pattern. This problem is illustrated in Figure 4.3(a), where there are no boundaries between some cardiac entities and the epicardium. An intensity-driven approach may fail to provide the desired differentiation. On the other hand, a boundary-driven approach will have difficulty in the endocardium. The irregularities shown in Figure 4.3(b) refer to different intensity properties of the papillary muscles found here, and degrade the strength of the edges in that region.

Finally, the accuracy of detection was very important in the design. As referred above, methods already exist to automatically extract LV region and border information, thus this work does not aim to reproduce or reformulate these approaches. However, existing systems usually need to be initialised very close to the target of interest. This is the case, for example, with active contours, which in many formulations tend to follow the nearest strong match; also active shape models, if initialised away from the target, have great difficulty in finding it. Therefore it makes sense to measure the border-tracing accuracy of these methods, but not their object detection performance. The present work aims to eliminate the manual step that is currently required for the initial position, and its performance should therefore be measured as object detection accuracy.

4.2.2 Template matching

A template was designed to match the assumed average appearance of the left ventricle as a double ellipse, with an inner ellipse extending towards the endocardial border superposed over an outer ellipse that extends towards the epicardial border. The long and short axes of each ellipse are parallel, and both ellipses have the same centre. The template has six parameters, namely the grey level intensities of each ellipse (w and g, respectively, see Fig. 4.4), the lengths of the long and short axes of the inner ellipse (lw and lh), the difference in axis length between the two ellipses (wp), and the angle the long axis makes with the horizontal (θ).

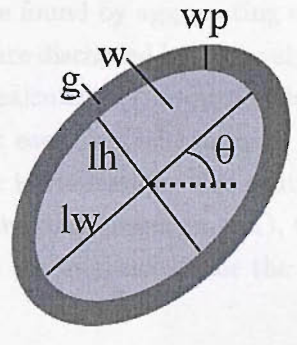


FIGURE 4.4: Template used for left ventricle search and matching

The template was fitted to each image using Powell's method [192]. Powell's is an optimisation method that seeks to change the parameters in order to minimise an energy function. We designed this function with two terms (see Eqn. 1). The first term is the intensity difference between the template pixels $\hat{g}_i$ and the image pixels g_i normalised by the template area T. This term makes the template approximate the intensity appearance of the LV in the image. The second term is the Mahalanobis distance M (see equation 2.9 on page 20) between the current template parameter values and the values obtained by training. This term counteracts the template's natural tendency to shrink to a point. The constants c_1 and c_2 were also determined by training.

$$E = c_1 \frac{1}{T} \sum_{i \in tmpl} (g_i - \hat{g}_i)^2 + c_2 M \tag{4.1}$$

The optimisation was performed with the template centred at each point of the image in turn, except for a small canvas region near the image borders. The initial values of the template were the average of those obtained by training, except for the angle θ which was always 90°. For improved computational efficiency at this stage only, the image was scaled to 25% of its size using nearest-neighbour interpolation, and so were all the size parameter values.

The training values were obtained by manual segmentation of a training dataset, disjoint from the testing dataset. The endo- and epicardial borders were segmented, and the resulting points were approximated using the ellipse fitting facility of the OpenCV library by Intel Corp. The fitted ellipses provided the lengths of the long and short axes of each template ellipse, thus yielding parameters lw, lh and wp, as well as the angle θ. The template was then centred at the fitted ellipses' centre points, and the remaining parameters only were allowed to optimise using Powell's method from manually set initial values. Each set of parameter values obtained in this way constituted a training data point.

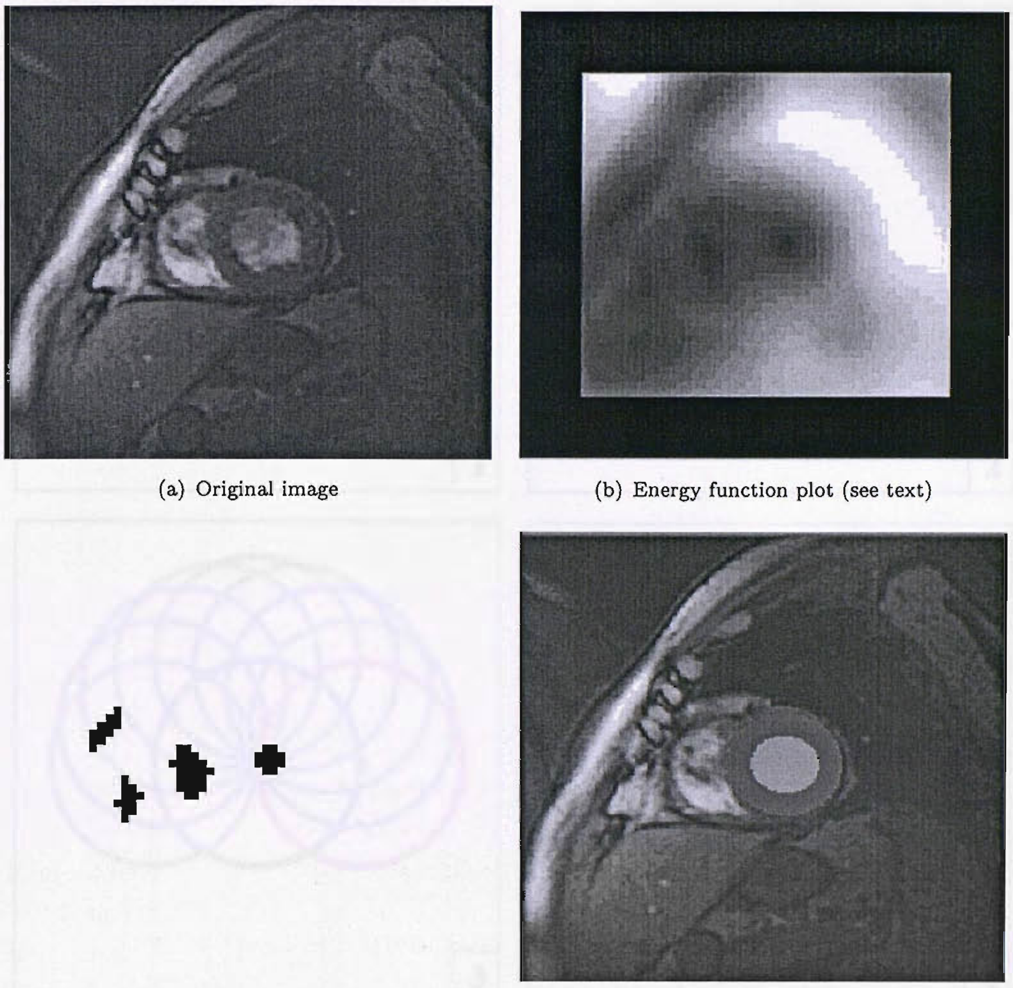
When the template is displaced and optimised over the whole image (Fig. 4.5(a)) as described above, a map of final values of the energy function is obtained (Fig. 4.5(b)). This map is thresholded with a trained value, resulting in a binary image (Fig. 4.5(c)). Clusters in this binary image are found by aggregating eight-connected pixels in groups. Groups with lower pixel counts are discarded leaving only a chosen number of the largest. The centroid of each group is calculated (no outlier detection is performed). Finally, the template is then centred at each centroid in turn, and it is optimised there using the trained parameter values for initialisation. The centroid position yielding the lowest value for the optimised energy function (see Eqn. 4.1), together with the corresponding parameter values, are chosen as the final match for the LV template (Fig. 4.5(d)).

4.2.3 Hough transform

The Hough transform (HT) maps points from a binary (black and white) image into parameter space. The assumption here is that the set of parameters corresponding to the object being looked for will have a high number of hits at the object's image location. In a practical implementation, the hits are recorded in an accumulator array .

The elliptical structure of the LV cross-section is apparent in edge-detected MRI images, and suggests that the LV may be located by a circle detector. In [246] the LV is detected using the Hough transform for circles. That paper does not indicate exactly which implementation of the Hough transform was used, as it only cites a fairly broad survey [109]. In that study, the range of circle radii was chosen so that both the endo- and epicardial borders would contribute to the accumulator array. In the present work, a range suited for the epicardial border only was used as previous testing indicated that considerable false alarms would result otherwise.

In our system, the edges of the original MRI image are detected with Canny edge detection [33]. All pixels output by the edge detector are then plotted in a binary image (see Fig. 4.7(a)). This is supplied to HT for circles, which generates an accumulator array for each given circle radius (Fig. 4.6 illustrates the process and Fig. 4.7(b) shows the result of applying HT for a given circle radius to Fig. 4.7(a)). HT requires the maximum and minimum diameters of the circles being searched for, which we chose as the trained (average) lengths of the longer and shorter axes, respectively, of the epicardial contour. Our implementation of HT performs unit increments of these values, which are in pixels. The plot of the accumulator array is then normalised to lie within the range (0, 255), and then thresholded by a trained value and clustered for determining the centre


(a) Original image (b) Energy function plot (see text)

(c) Thresholded from 4.5(b) (d) Matched template

FIGURE 4.5: Template matching steps.

of the LV. No further action was taken to remove false matches, unlike the procedure followed for the energy function plot in the previous section.

4.3 Test results

The convergence behaviour of Powell's method as applied to the template matching algorithm is plotted in Figure 4.8 for convergence on the MR image in Figure 4.1. The template was initialised manually by setting all the parameter values and centering the template at the correct LV centre.

For testing the system's performance, 31 images were used that were available on the Internet by Nicolae Duta of Michigan State University, US, as the complete database used for those images' corresponding paper [67] is proprietary and not publicly available. These 31 images are those for which that paper's algorithm failed to detect the LV.

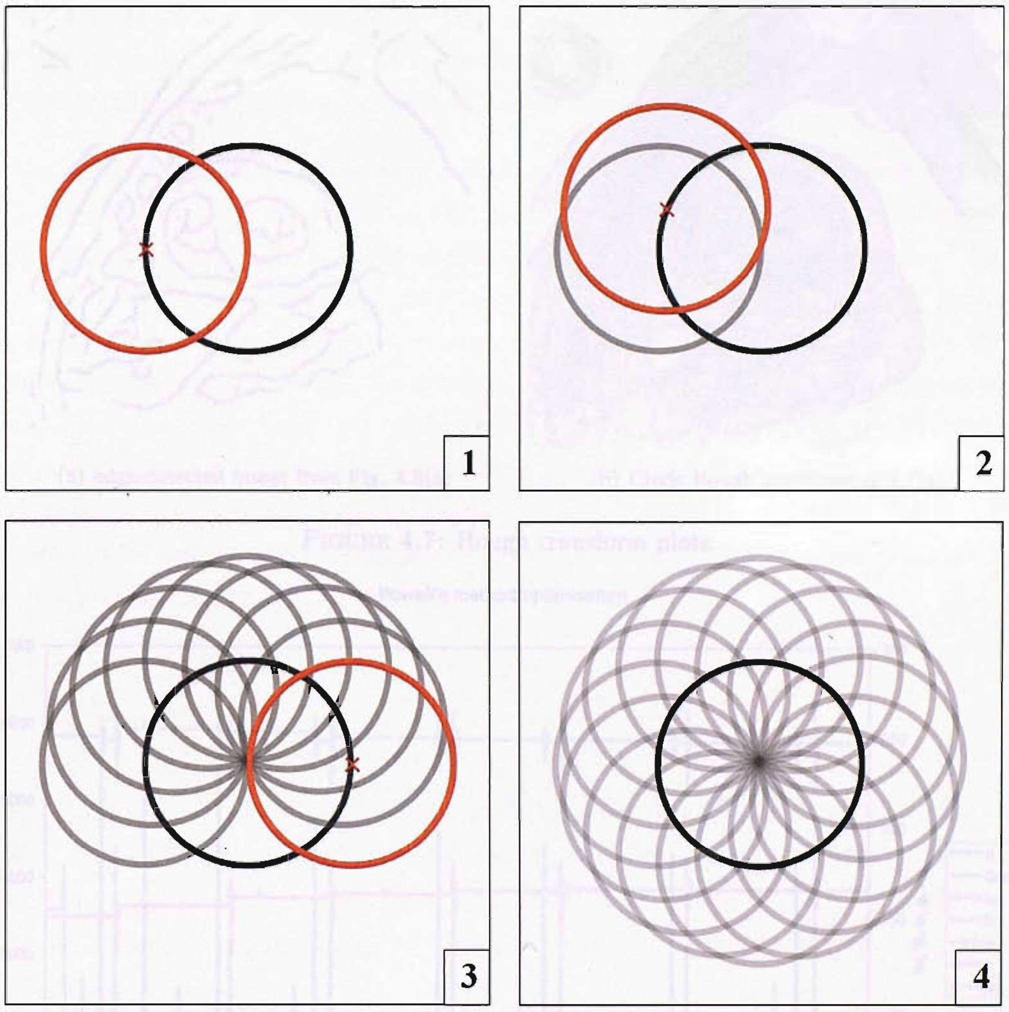
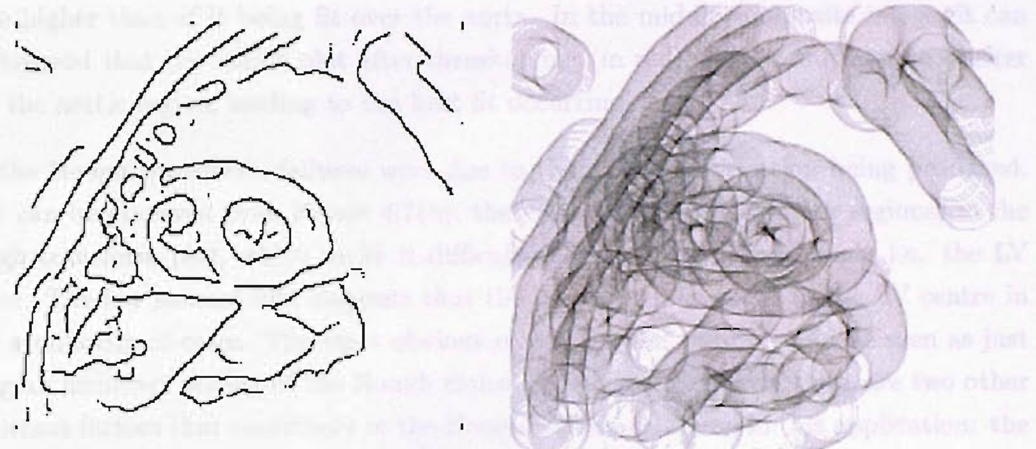


FIGURE 4.6: Circle Hough transform steps for a given circle radius. The location of the original circle edge is shown by the solid black line, with a circle probe (red) being centered at 16 equidistant points of the edge. Each probe position darkens the image (grey), so in the last step the centre of the original circle is the darkest point in the image generated by the probe.

They were acquired using a Siemens Magneton MRI system. In this section, we report performance using the small set of online images only, therefore the figures must not be seen as indicative of actual performance but merely as a tentative assessment. Of the 31 images, 8 were used for training and the remaining 23 were used for testing. The assignment of images to sets was done randomly. Elliptical markings added to the images by post-processing were deleted manually.

The template matching strategy succeeded in correctly finding the LV centre in 16 of the 23 test images, thus yielding a 70% success rate. The Hough transform method found the correct LV centre with no false matches in 6 of the 23 images, a 26% success rate.

To investigate the failure cases in the template matching test, it was noted that poor fits had been achieved during the optimisation which resulted in the energy values being



(a) edge-detected image from Fig. 4.5(a) (b) Circle Hough transform of 4.7(a)

FIGURE 4.7: Hough transform plots.

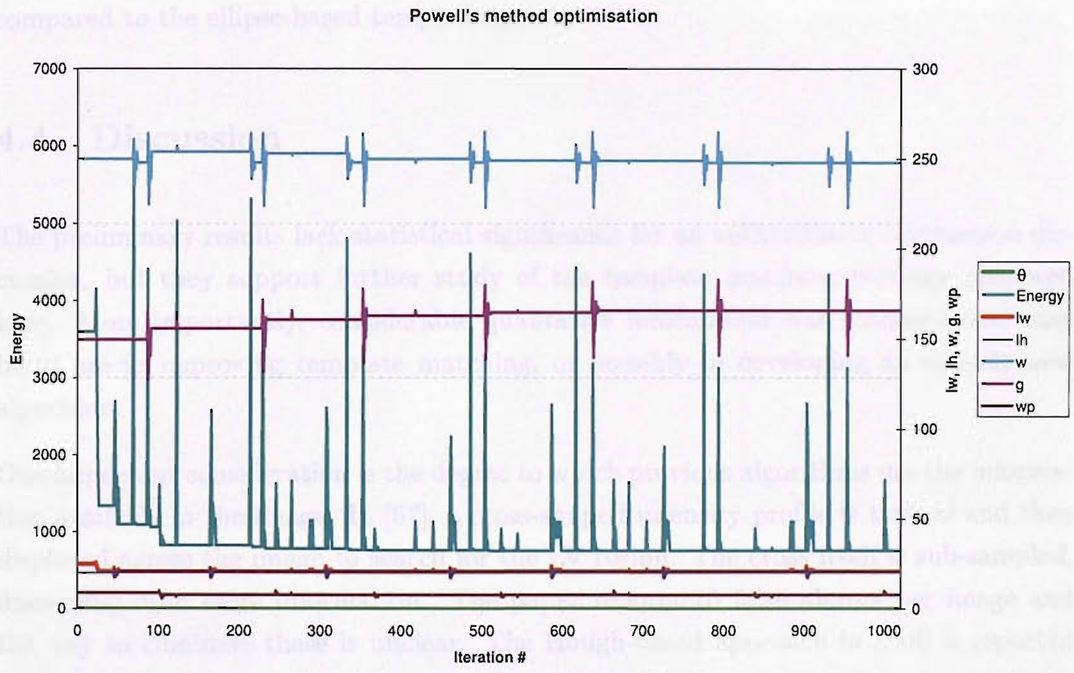


FIGURE 4.8: Powell method convergence.

above the pre-set threshold. Figure 4.9 illustrates the failure modes encountered during this investigation. Solid lines in the template were added for illustration purposes. In the first case illustrated, a successful match is achieved with the error function below the chosen threshold. In the second case, irregular borders can be observed in the LV. The algorithm fits the template to these borders, but because the shapes diverge too much from the template, the inner ellipse includes part of the LV's epicardium and as a result its grey level converges to an average of the lighter endocardium and the darker epicardium. This results in the energy function value for this fit exceeding the chosen threshold. In the third case, the energy function value of the template being fit over the

LV is higher than of it being fit over the aorta. In the middle composite image, it can be observed that the match plot after thresholding (in red) show a much larger cluster over the aortic region, leading to the best fit occurring there.

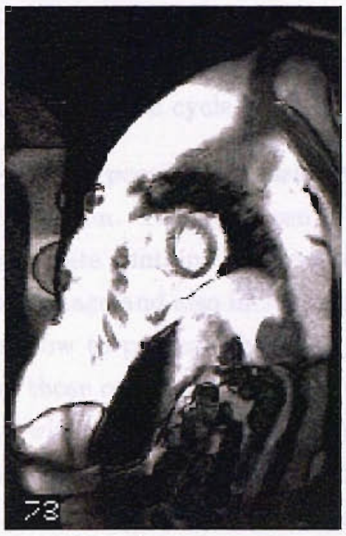
For the Hough transform, failures were due to the wrong LV location being produced. As it can be observed from Figure 4.7(b), there are many peaks (darker regions) in the Hough transform plot, which make it difficult to isolate the desired peak, i.e. the LV centre. The low success rate suggests that the highest peaks occur in the LV centre in only a minority of cases. The most obvious reason for this failure could be seen as just being an incorrect setting of the Hough transform radius. However, there are two other important factors that contribute to the Hough transform failing in this application: the irregularity of the LV boundaries, as can be verified in the instance of Figure 4.7(a); and the already mentioned gaps at the LV borders, which can also be verified in Figure 4.7(a). Because the Hough transform applied here was circle-based, it was also disadvantaged compared to the ellipse-based template matcher.

4.4 Discussion

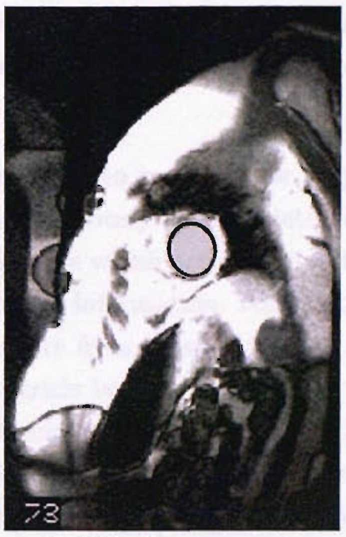
The preliminary results lack statistical significance for an authoritative comparison discussion, but they support further study of the template matching strategy proposed here. More importantly, considerable qualitative information was gathered that may be of use in improving template matching, or possibly in developing an entirely new algorithm.

One important consideration is the degree to which previous algorithms use the information available in the image. In [67], a cross-shaped intensity profile is trained and then displaced across the image to search for the LV region. The cross itself is sub-sampled, discarding even more information. The paper reports 10 false alarms per image and the way to eliminate these is unclear. The Hough-based approach in [246] is reported as performing satisfactorily. This uses edge-detected data and discards region intensity information. Another proposed circle-based vessel detection method, the boundariness operator [204], looks for the dark-to-light transition of the border profile.

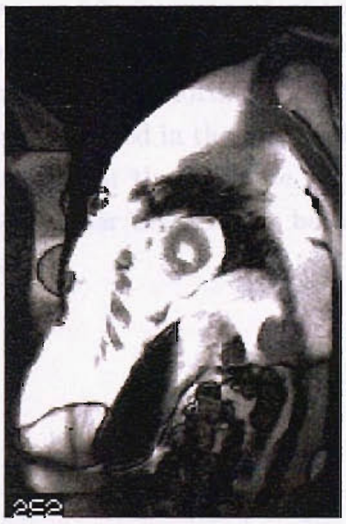
One obvious extension would be to implement the Hough transform for ellipses, as some MRI images of the LV, especially more recent higher-definition ones, have a very clearly defined elliptical rather than circular structure. In practice, this results in a cluster of about 2 to 3 HT peaks around the true LV centre when the circle version is used. Applying the original HT formulation for effective ellipse finding requires a larger number of variables, which becomes computationally intensive and has very large memory requirements. We experimented with the randomised HT [156] and found that it was too sensitive to divergence from the canonical ellipse structure, an issue that LV structures tend to have due to their highly malleable nature. The ellipse HT was



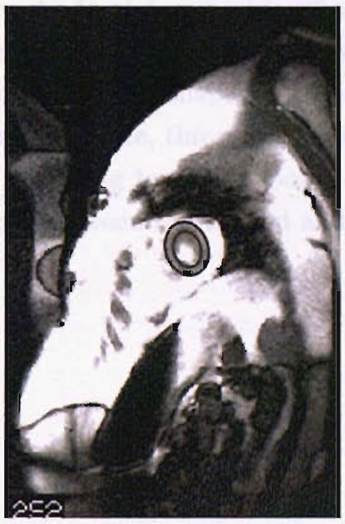
(a) Original MR image.



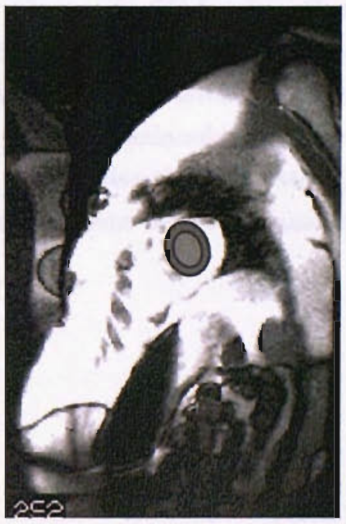
(b) Match at correct location.



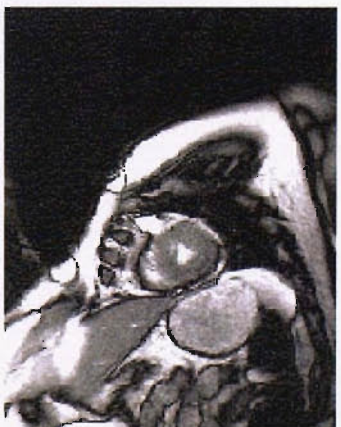
(c) Original MR image.



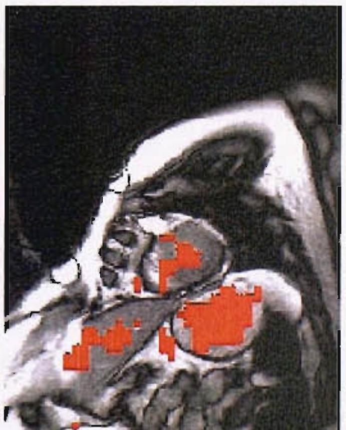
(d) Outline of fit template.



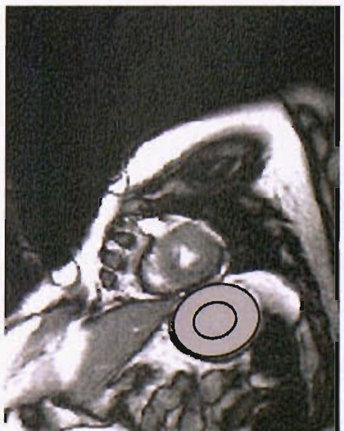
(e) Grey levels of fit template.



(f) Original MR image.



(g) Thresholded energy values in red.



(h) Grey levels of fit template.

FIGURE 4.9: Template fit failure modes.

successfully applied for rendering the LV in SPECT images by using a clustering scheme that merges detected ellipses together to form a global LV description at each time point of the cardiac cycle.

Another possible extension would be to combine the proposed template with edge information. This has been done previously for face detection [283], where a deformable template contains different segments which are attracted to peaks, valleys and edges in the image and also include intensity information. For the LV, however, the issue remains of how to prevent the LV template from being attracted to other, stronger edges such as those of the nearby right ventricle border. This was an important issue we were able to witness when using a commercial implementation of active contours designed for LV tracking. Active contours [126] are a delineation tool that can be designed and trained to be attracted to certain intensity profiles, together with a smoothness constraint. In the LV situation, active contours as implemented by the commercial medical software we used had a strong bias towards the right ventricle, thus missing the LV border. One solution to this could be to use a stronger training dataset by preventing the LV template from distorting towards undesirable shapes or locations. This is already partly implemented in the system described here, through the use of the Mahalanobis distance to prevent the template from shrinking towards a point - the smaller the template is, the higher the distance becomes between the trained and the current set of parameter values.

Chapter 5

Vector Field Restoration for Blood Flow MRI

5.1 Restoration algorithm

The analysis of blood flow patterns and their interaction with cardiovascular structure plays an important role in the study of cardiovascular function. In particular vortical motion, known to exist in the diastolic phase of the cardiac cycle [131], is the most important flow feature and its evolution over different phases of the cardiac cycle can provide important insight into the health status of the heart. Previous research [269] has shown that in order to achieve a comprehensive and integrated description of flow in health and disease, it is necessary to characterise and model both normal and abnormal flows and their effects. This permits the establishment of links between blood flow patterns and the localised genesis and development of cardiovascular disease.

With the ability to acquire multi-dimensional cine flow data, Magnetic Resonance (MR) velocity imaging is increasingly used for acquiring in vivo flow details. Flow velocity images acquired by MR velocity-mapping are generally subject to a certain amount of noise that is intrinsic to system hardware setup and that is specific to patient movement in relation to imaging sequence designs and this poses a problem for automatic quantitative analysis of flow features. To tackle this problem, a vector field restoration method is needed. In recent years, the total variation (TV) based restoration method, first introduced by Rudin [203] for restoration of scalar images, has received a lot of interest. The choice of the TV norm as the function to be minimised is due to the fact that it does not penalise discontinuities, thus edges and other topological features in the image can be preserved. TV-based restoration methods have been demonstrated to be effective and superior to linear filtering methods for scalar images, and several researchers [36, 35] have extended the method for vector-valued images. In this chapter, the TV-based variational method developed in 2D by Ng and Yang [177] is implemented and extended to 3D [175]. This method restores the direction field (the phase of the velocity) and finds the optimal solution of a regularisation parameter, which is known to significantly affect the restoration result. It combines the TV-based formulation by Chan and Chen [36] and employs a novel numerical scheme based on the First Order Lagrangian Method [17]. This restoration scheme has been proven to restore both 2D and 3D flow fields effectively [175].

5.2 Related Work

By letting z denote an image degraded by noise, u the original noise-free image and ϵ the additive noise component present in z, we can write:

$$z = H(u) = u + \epsilon \tag{5.1}$$

where $H(u)$ is a function that maps a point in u to a point in z. The problem of image restoration can be viewed as finding an inverse transformation function H^{-1} for recovering u from z. A possible way of solving this problem is to apply a scalar inverse transformation function to each component of the vector separately, as if they were independent scalar images. Another approach is to derive a function or functions that map vectors to vectors by taking into account the coupling or relationship between the components in the vector. The first approach is usually easier to implement and it gives acceptable results. However, in practice, there is usually some sort of relationship between the different components of the vectors in a field. Thus, including the coupling information into the design of the restoration function will give a better result.

5.2.1 Simple Gaussian Smoothing

The simplest method of image filtering is to compute an output image with weighted sums of the input image. This consists of filling each output location with a weighted sum of the pixel values in the neighbourhood of the same location in the input image. If all pixels in the neighbourhood are weighted by the same constant - uniform local averaging - the result may appear acceptable but suffers from a number of undesirable effects such as ringing, a visual artefact composed of narrow horizontal and vertical bars. If, on the other hands, a set of weights is used that is larger at the centre and falls off sharply as the distance from the centre increases, then the kind of smoothing that occurs in a defocused lens is better approached. A good formal model for this is the symmetric Gaussian convolution kernel:

$$G_\sigma = \frac{1}{2\pi\sigma^2} \exp\left(-\frac{x^2 + y^2}{2\sigma^2}\right) \tag{5.2}$$

where σ is the standard deviation. This smoothing kernel forms a weighted average that weights pixels at its centre more strongly than at its boundaries. If the standard deviation is chosen to be small, then smoothing will have little effect. For a larger standard deviation, the noise will largely disappear at the cost of blurring. A very large standard deviation will cause much of the image detail to disappear along with the noise.

5.2.2 Anisotropic Smoothing

In practical applications, Gaussian smoothing tends to excessively blur out edges. This suggests that smoothing should be done differently at edge points. Non-linear schemes have been proposed in which local structures and statistics are taken into account during the filtering process. Because of their adaptiveness to local image features, they are known as adaptive methods [141, 260, 261]. Edge preserving smoothing makes an estimate of the magnitude and orientation of the gradient. For large gradients, an oriented operator then smoothes strongly perpendicular to the gradient and weakly along the gradient. For small gradients, a symmetric smoothing operator can be used. One of the earliest studies [144] applied iteratively a weighted mask with coefficients that are based on the differences between the value at the centre point and those of its neighbours. Similar approaches can also be found elsewhere [52]. The major drawback of these iterative smoothing methods is that their convergence properties are difficult to establish. Another approach [21] used weak continuity constraints to allow discontinuities in a piecewise continuous reconstruction of a noisy signal. The convergence behaviour was well studied and results provided were promising, but unfortunately it was accomplished with a formidable computational cost.

By casting the problem in terms of a heat equation in an anisotropic medium, an anisotropic diffusion scheme has been presented in a seminal paper [189]. The principal aim of this method was to enhance edges and facilitate region segmentation, however it can also be considered as a useful tool for image noise filtering (for example in [86]). The method allows an image $f(x)$ to evolve over time via the diffusion equation:

$$\frac{\partial f(x,t)}{\partial t} = D(x,t)\Delta f(x) + \nabla D \cdot \nabla f(x) \tag{5.3}$$

where $D > 0$ is a decreasing function with respect to the edge strength and Δ is the Laplacian operator. The result is to diffuse $f(x)$ most where the gradient is smallest and vice-versa. Since the orientation of the edge has not been taken into account, the noise removal is not ideal. When an input image contains a considerable amount of noise, the resulting random fluctuations of $|\nabla f(x)|$ slow down the diffusion process.

An edge orientation-sensitive anisotropic filter was implemented and applied to 2D and 3D MRI, and compared favourably with Gaussian filtering [270]. There, the basic

Gaussian kernel is adopted and modified to define an anisotropic kernel:

$$k(x_0, x) = \rho(\boldsymbol{x} - \boldsymbol{x}_0) \exp\left(- \left(\frac{((\boldsymbol{x} - \boldsymbol{x}_0) \cdot \boldsymbol{n})^2}{\sigma_1^2(\boldsymbol{x}_0)} + \frac{((\boldsymbol{x} - \boldsymbol{x}_0) \cdot \boldsymbol{n}_\perp)^2}{\sigma_2^2(\boldsymbol{x}_0)} \right) \right) \tag{5.4}$$

where $\rho(\boldsymbol{x})$ is a positive rotationally symmetric cut-off function that satisfies $\rho(\boldsymbol{x}) = 1$ when $|\boldsymbol{x}| < r$ (r is the maximum support radius), $\rho(\boldsymbol{x}) = 0$ otherwise. $\boldsymbol{n}$ and $\boldsymbol{n}_\perp$ are mutually normal unit vectors, with $\boldsymbol{n}$ parallel to the principal axis of the resulting ellipse-shaped kernel, and $(\boldsymbol{x} - \boldsymbol{x}_0)$ is the position vector of $\boldsymbol{x}$ with respect to $\boldsymbol{x}_0$, the kernel centre.

The shape of the kernel is determined by the two non-negative functions $\sigma_1(\boldsymbol{x})$ and $\sigma_2(\boldsymbol{x})$. The direction of $\boldsymbol{n}$ is defined as the eigenvector that corresponds to the smallest eigenvalues λ_{min} of the second moment matrix $\mathbf{R}$ of the spatial domain:

$$\mathbf{R}_{ij} = \frac{1}{4\pi^2} \int \int_\Omega \left(\frac{\partial f}{\partial x_i} \right) \left(\frac{\partial f}{\partial x_j} \right) d\boldsymbol{x}_1 d\boldsymbol{x}_2 \tag{5.5}$$

λ_{min} and the largest eigenvalues λ_{max} are used to define a measure of anisotropy:

$$g(\boldsymbol{x}_0) = \left(\frac{\lambda_{max} - \lambda_{min}}{\lambda_{max} + \lambda_{min}} \right)^2$$

In addition to edges, corners and junctions must also be preserved thus a corner strength is defined:

$$c(\boldsymbol{x}_0) = (1 - g(\boldsymbol{x}_0)) |\nabla f(\boldsymbol{x}_0)|^2$$

Finally, these are taken into account in $\sigma_1(\boldsymbol{x})$ and $\sigma_2(\boldsymbol{x})$:

$$\sigma_1(\boldsymbol{x}_0) = \frac{r}{1 + c(\boldsymbol{x}_0)/\alpha}$$

$$\sigma_2(\boldsymbol{x}_0) = \frac{r(1 - g(\boldsymbol{x}_0))}{1 + c(\boldsymbol{x}_0)/\alpha}$$

where r is the kernel support radius and α is a normalisation factor. As the corner strength $c(\boldsymbol{x}_0)$ increases, both $\sigma_1(\boldsymbol{x})$ and $\sigma_2(\boldsymbol{x})$ decrease and the overall kernel is smaller. As the anisotropy $g(\boldsymbol{x}_0)$ increases, $\sigma_2(\boldsymbol{x})$ decreases which makes the kernel shape depart away from circular and more towards elliptical.

This method has been shown to preserve edges better than Gaussian filtering for scalar images. However, when applied to vector-valued images (to each vector component separately) the methods were found to give similar results, a likely consequence of discarding the relationships between components [178].

5.3 Materials and Methods

5.3.1 Flow Field Restoration

Total Variation Based Restoration Method

The restoration method in this study is formulated as a constrained optimisation problem that restores the original images by minimising the total variation energy of the normalised velocity field subject to a constraint that depends on the noise level. The effectiveness of this restoration method greatly depends on the choice of a regularisation parameter, which in practice is often conveniently fixed or determined empirically [49]. The restored image may converge to different results depending on the parameter settings, thus creating problems in practical implementations. To avoid having to fix this parameter, the First Order Lagrangian method was used to derive the optimal value of this parameter while solving the minimisation problem.

Let $\boldsymbol{u}_0$ denote the direction field of the noisy image, $\boldsymbol{u}$ denote the clean direction field that we want to restore and that $\boldsymbol{u}_0 = \boldsymbol{u} + \boldsymbol{n}$ where $\boldsymbol{n}$ denotes the additive noise. Assuming the variance of the noise $\boldsymbol{n}$ is σ^2, the value of which is known, the restoration of the direction of a velocity field can be formulated as a constrained optimisation problem by minimising:

$$E^{TV}(\boldsymbol{u}) = \int\limits_{\Omega} e(\boldsymbol{u}, \alpha) \tag{5.6}$$

subject to the equality constraint:

$$h(\boldsymbol{u}) = \frac{1}{2}\left(\int\limits_{\Omega} |\boldsymbol{u} - \boldsymbol{u}_0|^2 - |\Omega|\sigma^2\right) = 0 \tag{5.7}$$

where E^{TV} denotes the total TV energy for the whole image, $e(\boldsymbol{u}; \alpha)$ is the energy at pixel α, Ω is the image domain and $|\Omega|$ is the size of the image domain.

The optimisation problem can be solved by solving the corresponding Tikhonov regularised unconstrained problem [237], taking $\boldsymbol{u}$ as the desired true solution:

$$\min_{\boldsymbol{u}} \lambda \int_{\Omega} (\boldsymbol{u} - \boldsymbol{u}_0)^2 + \int_{\Omega} e(\boldsymbol{u}; \alpha) \tag{5.8}$$

where λ is a regularisation parameter known as the Lagrange multiplier of the constrained optimisation problem. This approach can be viewed as a penalty approach for the constrained optimisation problem. Several computational algorithms have been proposed [256, 147] to solve this unconstrained problem and typically the value of the regularisation parameter is manually fixed. Instead of solving the Tikhonov regularised unconstrained problem, the constrained problem is tackled by applying the First Order

Lagrangian method, which finds both the minimal solution and the associated Lagrange multiplier for the constrained optimisation problem.

First Order Lagrangian Method

Given a constrained optimisation problem with m constraints:

$$\begin{aligned} &\textit{minimise} \quad f(x) \\ &\textit{subject to} \quad h_i(x) = 0, i = 1, \ldots, m \end{aligned} \tag{5.9}$$

where $f(x)$ is the function to be minimised and $h_i(x)$ are the equality constraints of the problem. The Lagrange Multiplier theorem states the necessary condition for optimality is that for a given minimal solution x^*, there exist scalars $\lambda_1, \ldots, \lambda_m$ such that

$$\nabla f(x^*) + \sum_{i=1}^{m} \lambda_i \nabla h_i(x^*) = 0 \tag{5.10}$$

where λ_i are referred to as the Lagrange multipliers of the constrained problem and are sometimes represented as a vector $\lambda = [\lambda_1, \ldots, \lambda_m]$ called the Lagrange multiplier vector. The necessary condition given in equation 5.10, together with $h_i(x) = 0$ are referred to as the Lagrangian system. If eq. 5.10 is written in terms of the Lagrangian function defined by:

$$L(x, \lambda) = f(x) + \sum_{i=1}^{m} \lambda_i h_i(x) \tag{5.11}$$

then the necessary condition can be written compactly in the form:

$$\begin{aligned} \nabla_x L(x^*, \lambda^*) = 0 \\ \nabla_y L(x^*, \lambda^*) = 0 \end{aligned} \tag{5.12}$$

where the vectors x^* and λ^* are the optimal solutions of the Lagrangian system.

A general class of methods, called the Lagrangian methods, have been derived for solving the Lagrangian system. The simplest of all Lagrange methods, called the First Order Method [17], is given by:

$$x^{n+1} = x^n - \delta_s \nabla_x L(x^n, \lambda^n), \quad \lambda^{n+1} = \lambda^n - \delta_s \nabla_\lambda L(x^n, \lambda^n) \tag{5.13}$$

where δ_s is a positive scalar step-size and $L(\cdot)$ is the associated Lagrangian function.

Numerical Scheme

Based on the First Order method, a numerical scheme is followed for solving the constrained optimisation problem. Let $\boldsymbol{v}$ denote the original velocity vector and $\boldsymbol{u}$ a normalised vector. The velocity vector $\boldsymbol{v}$ in $\mathcal{R}^3$ is mapped to unit vector $\boldsymbol{u}$ in $\mathcal{S}^2$ (the unit sphere) by setting $\boldsymbol{u} = \boldsymbol{v}/|\boldsymbol{v}|$. This mapping $f : \mathcal{R}^3 \longrightarrow \mathcal{S}^2$ is valid for both 2D and 3D vectors, as a 2D vector can be considered a 3D vector with the z-component set to zero.

The energy function $e(\boldsymbol{u};\alpha)$ at voxel α can be defined as:

$$e(\boldsymbol{u};\alpha) = \left[\sum_{\beta\in N_\alpha} d_l^2(\boldsymbol{u}_\beta, \boldsymbol{u}_\alpha)\right]^{\frac{1}{2}} \tag{5.14}$$

where N_α denotes the neighbourhood of pixel α, and d_l denotes the embedded Euclidean distance in $\mathcal{S}^2$:

$$d_l(f,g) = \|f-g\|_{\mathcal{R}^3} \forall f,g \in \mathcal{S}^2 \tag{5.15}$$

The TV-energy of the direction field in domain Ω is then:

$$\begin{aligned} E^{TV} &= \sum_{\alpha\in\Omega} e(\boldsymbol{u};\alpha) \\ &= \sum_{\alpha\in\Omega}\left[\sum_{\beta\in N_\alpha} d_l^2(\boldsymbol{u}_\beta, \boldsymbol{u}_\alpha)\right]^{\frac{1}{2}} \end{aligned}$$

and the constrained optimization can be written as:

$$\min E^{TV} = \sum_{\alpha\in\Omega}\left[\sum_{\beta\in N_\alpha} d_l^2(\boldsymbol{u}_\beta, \boldsymbol{u}_\alpha)\right]^{\frac{1}{2}} \tag{5.16}$$

subject to the constraint of:

$$h(\boldsymbol{u}) = \frac{1}{2}\left[\sum_{\alpha\in\Omega} d_l^2(\boldsymbol{u}_\alpha, \boldsymbol{u}_\alpha^0) - |\Omega|\sigma^2\right] = 0 \tag{5.17}$$

where $\boldsymbol{u}^0$ denotes the original noisy image. The corresponding Lagrange function, also referred to as the unconstrained TV energy, is:

$$\begin{aligned} L(\boldsymbol{u};\lambda) &= E^{TV} + \lambda \cdot h(\boldsymbol{u}) \\ &= \sum_{\alpha\in\Omega}\left[\sum_{\beta\in N_\alpha} d_l^2(\boldsymbol{u}_\beta, \boldsymbol{u}_\alpha)\right]^{\frac{1}{2}} + \frac{\lambda}{2}\left[\sum_{\alpha\in\Omega} d_l^2(\boldsymbol{u}_\alpha, \boldsymbol{u}_\alpha^0) - |\Omega|\sigma^2\right] \end{aligned}$$

At the optimal solution, the constraint is met, i.e. $h(\boldsymbol{u})$ equals zero, and therefore $L(\boldsymbol{u};\lambda)$ equals E^{TV}.

By computing the gradient of $L(\boldsymbol{u},\lambda)$ with respect to $\boldsymbol{u}_\alpha$, we obtain:

$$\frac{\partial L}{\partial \boldsymbol{u}_\alpha} = \sum_{\beta \in N_\alpha} \left[\frac{\partial}{\partial \boldsymbol{u}_\alpha} d_l^2(\boldsymbol{u}_\beta, \boldsymbol{u}_\alpha)\right] \left(\frac{1}{2}\left[\frac{1}{e(\boldsymbol{u};\alpha)} + \frac{1}{e(\boldsymbol{u};\beta)}\right]\right) + \frac{\lambda}{2}\left(\frac{\partial}{\partial \boldsymbol{u}_\alpha} d_l^2(\boldsymbol{u}_\alpha, \boldsymbol{u}_\alpha^0)\right) \tag{5.18}$$

In order to compute the gradient of a function $G(\boldsymbol{u})$ on $\mathcal{S}^2$, the gradient of $G(\boldsymbol{u})$ on $\mathcal{R}^3$ is first calculated and then projected onto the plane that is orthogonal to $\boldsymbol{u}$:

$$\frac{\partial}{\partial \boldsymbol{u}} G(\boldsymbol{u}) = \Pi_{\boldsymbol{u}} \ \mathrm{grad}_{\mathcal{R}^3} \ G(\boldsymbol{u}) \tag{5.19}$$

($\Pi_{\boldsymbol{u}}$ denotes projection onto plane orthogonal to $\boldsymbol{u}$). Hence,

$$\frac{\partial L}{\partial \boldsymbol{u}_\alpha} = -\sum_{\beta \in N_\alpha} \Pi_{\boldsymbol{u}_\alpha}(\boldsymbol{u}_\beta)\left(\frac{1}{e(\boldsymbol{u};\alpha)} + \frac{1}{e(\boldsymbol{u};\beta)}\right) - \lambda \Pi_{\boldsymbol{u}_\alpha}(\boldsymbol{u}_\alpha^0) \tag{5.20}$$

Accordingly, the gradient of $L(\boldsymbol{u};\lambda)$ with respect to λ becomes:

$$\frac{\partial L}{\partial \lambda} = \frac{1}{2}\left[\sum_{\alpha \in \Omega} d_l^2(\boldsymbol{u}_\alpha, \boldsymbol{u}_\alpha^0) - |\Omega|\sigma^2\right] \tag{5.21}$$

The discrete form of the First Order Lagrangian Method can finally be derived and written as a pair of iterative equations as follows:

$$\left\{\begin{array}{l} \boldsymbol{u}_\alpha^n + \delta t \cdot \Pi_{\boldsymbol{u}_\alpha}\left[\sum_{\beta \in N_\alpha} wt_\alpha^\beta \boldsymbol{u}_\beta + \lambda^n \boldsymbol{u}_\alpha^0\right] \\ \lambda^{n+1} = \lambda^n + \delta t \cdot \frac{1}{2}\left[\sum_{\alpha \in \Omega} d_l^2(\boldsymbol{u}_\alpha, \boldsymbol{u}_\alpha^0) - |\Omega|\sigma^2\right] \end{array}\right. \quad \text{where } wt_\alpha^\beta = \left(\frac{1}{e(\boldsymbol{u};\alpha)} + \frac{1}{e(\boldsymbol{u};\beta)}\right) \tag{5.22}$$

and δt denotes the step size. To ensure that $\boldsymbol{u}_\alpha^{n+1}$ remains on $\mathcal{S}^2$, $\boldsymbol{u}_\alpha^{n+1}$ is normalised at the end of each step.

5.4 Results and Discussion

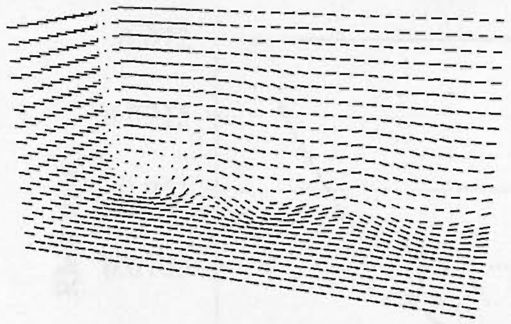
In order to provide a detailed analysis of the performance of the proposed method, a synthetic dataset simulating a 3D vortex with added Gaussian noise was used for examining the restoration process. A sample noise-free image, the corresponding noisy image and the restored image are shown in Figure 5.1. To analyse the sensitivity of the regularization parameter, the existing variational method with various fixed lambda values was used to restore the noisy synthetic data. It is found that when the value of lambda is set to be the optimal value, the restoration result is at its optimum and comparable to that of our newly proposed method. As expected, when the value of lambda is too small, the image is over-smoothed; whereas when the value of lambda is too big, the image is under-smoothed. The results clearly demonstrate the advantage of the proposed method in converging automatically to the optimal solution without explicitly presetting the regularization parameter.

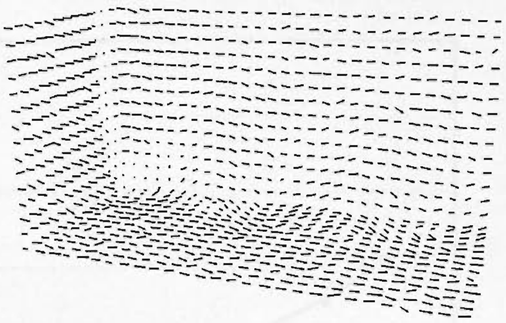
To verify that the value of regularization parameter found by the proposed method corresponds to the optimal value, experiments with a range of lambda values were carried out and the RMS error of the results are compared (Figure 5.2). The optimal value of lambda found empirically coincides with the value obtained by the First Order method, justifying the robustness of the proposed technique. To assess the convergence behaviour of the proposed technique, the constrained and unconstrained energy term and the value of lambda were recorded at each iteration (Figure 5.3). It is found that both energy terms converge to the same value as expected and the value of lambda converges in a similar behaviour. The rate of convergence depends on the choice of step size, which is typically of the order of 0.01. The algorithm converged after approximately 250 iterations.

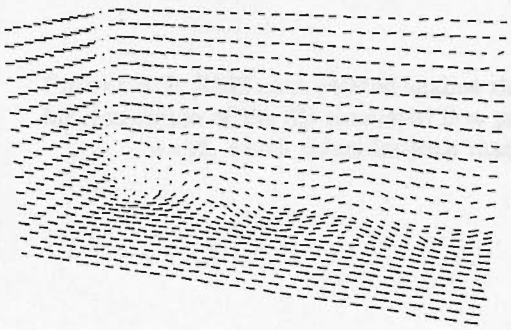
We also applied the algorithm to a 3D Computational Fluid Dynamic (CFD) dataset simulating flow in a normal cardiac left ventricle. The visualised streamline plot of the simulated flow at a selected time frame during diastole is shown in Figure 5.4, where a vortex ring in the left ventricle can be seen. The proposed method and the existing variational method with various fixed lambda values were applied to restore the noisy flow pattern. It is evident from Figure 5.4 that a similar trend to that of Figure 5.1 can be observed. That is, when the value of lambda is set to be smaller than the optimal value, the image is over-smoothed; when the value of lambda is too big, the image is under-smoothed. The corresponding RMS error of the results at various values of lambda were plotted in Figure 5.5 and the optimal value of lambda found empirically also coincides with the value obtained by the First Order method. In order to assess the convergence behaviour of the energy terms and lambda, the values of the constrained and unconstrained energy and lambda were plotted against the number of iterations as depicted in Figure 5.6. The graphs show that all three values converged after about 4000 iterations in a similar behaviour and the two energy terms converged to the same value as expected.

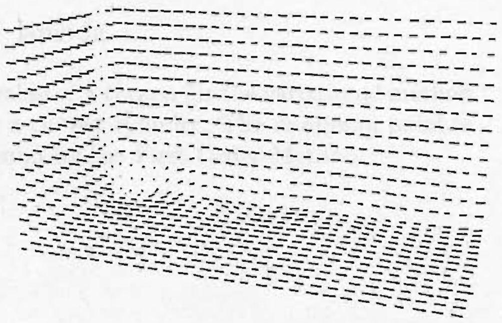
5.5 Conclusions

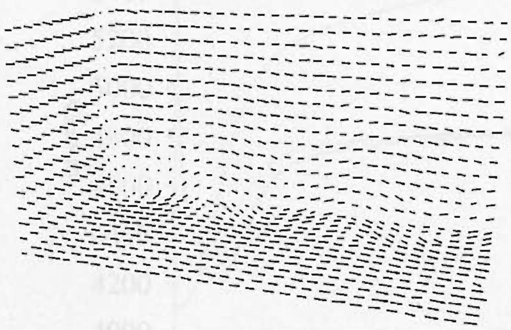
In this study, we have demonstrated that the First Order Lagrangian based restoration method is effective in restoring velocity field and that the choice of the regularization term greatly affects the restoration result. The main advantage of the proposed method is that it converges to the optimal solution without explicit or empirically defined stopping criteria, thus greatly enhancing its practical value.


(a)


(b)


(c)


(d)


(e)

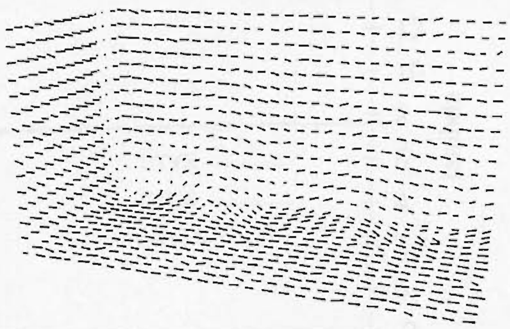

(f)

FIGURE 5.1: Comparison of TV-based variational method with different values of lambda using the 3D dataset of simulated flow past a curved cylinder. Only 3 reslice planes are shown here. (a) Noise-free image. (b) With added noise. (c-f) Restored by TV method: (c) with λ determined by First Order Method. (d) λ fixed at 10. (e) λ fixed at 15 (optimal value). (f) λ fixed at 20.

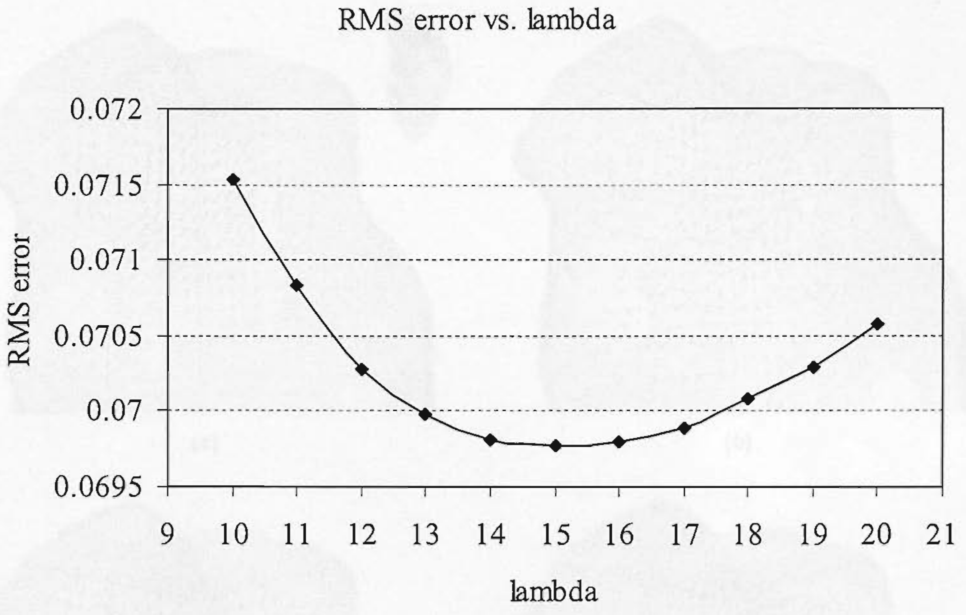


FIGURE 5.2: RMS error plotted against the value of λ chosen for the variational method for restoration of the 3D simulated flow past a curved cylinder. The minimum point is λ=15, which coincides with that found by the First Order Method.

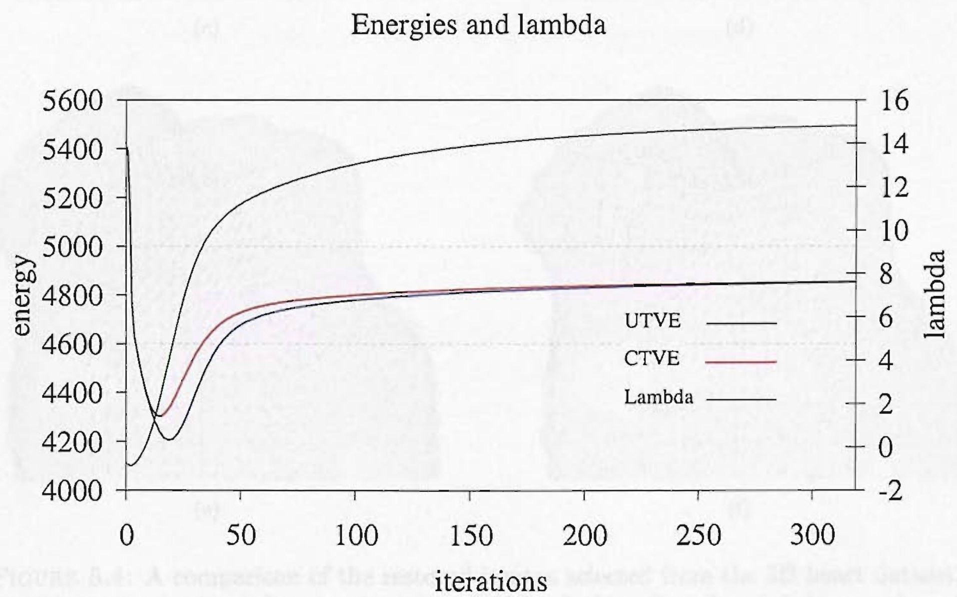


FIGURE 5.3: Constrained and unconstrained energy (left axis scale) and lambda (right axis scale) against the number of iterations for restoration of the 3D simulated flow past a curved cylinder.

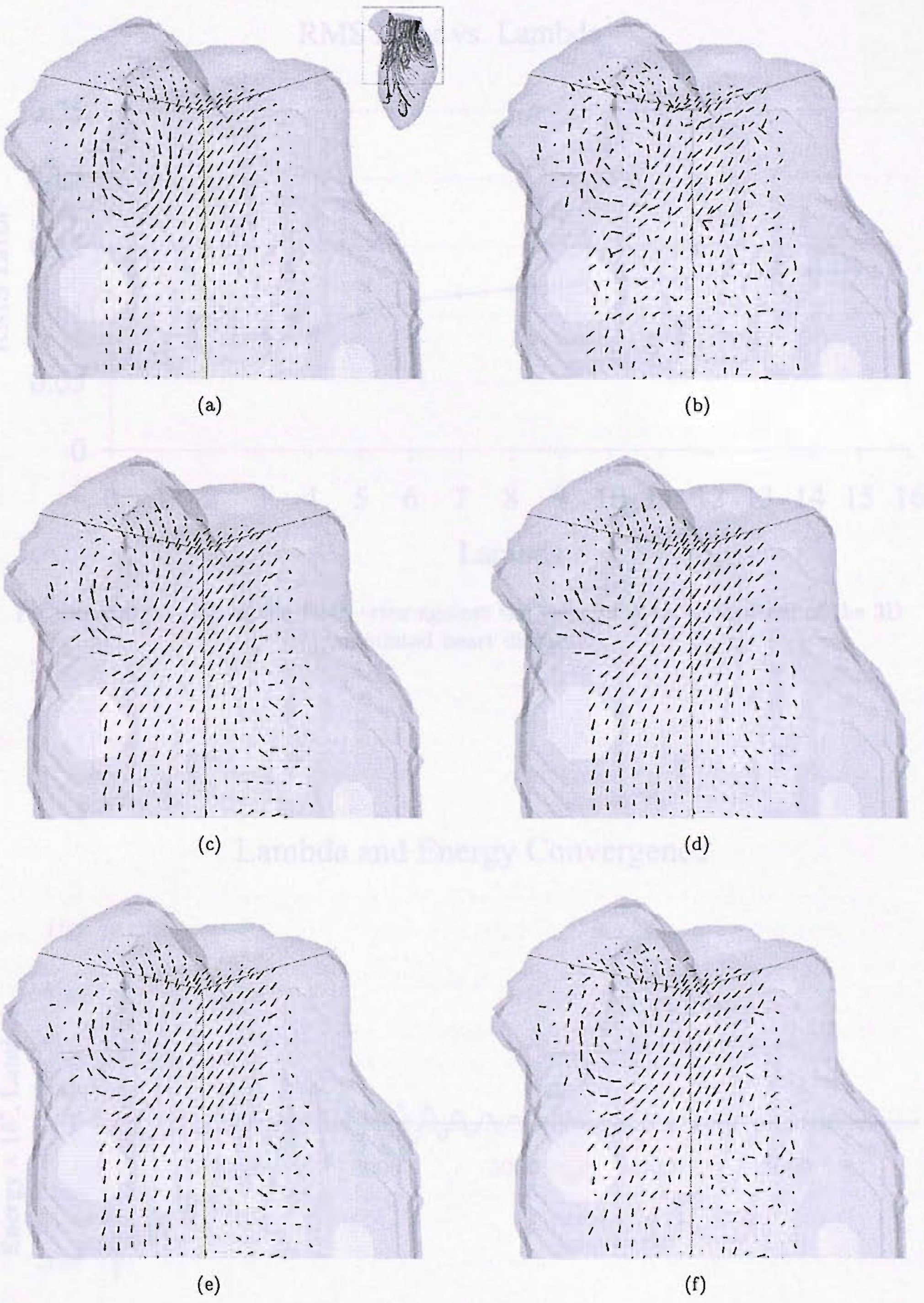


FIGURE 5.4: A comparison of the restored images selected from the 3D heart dataset, which simulates blood flow in the left ventricle during diastole. (a) Arrow plot of the original velocity field, with inset showing zoomed area. (b) With added noise. (c) Restored with First Order method. (d) Restored with lambda fixed at 1.5 (over-smoothed). (e) Restored with lambda fixed at 3.0 (optimal). (f) Restored with lambda fixed at 7.0 (under-smoothed).

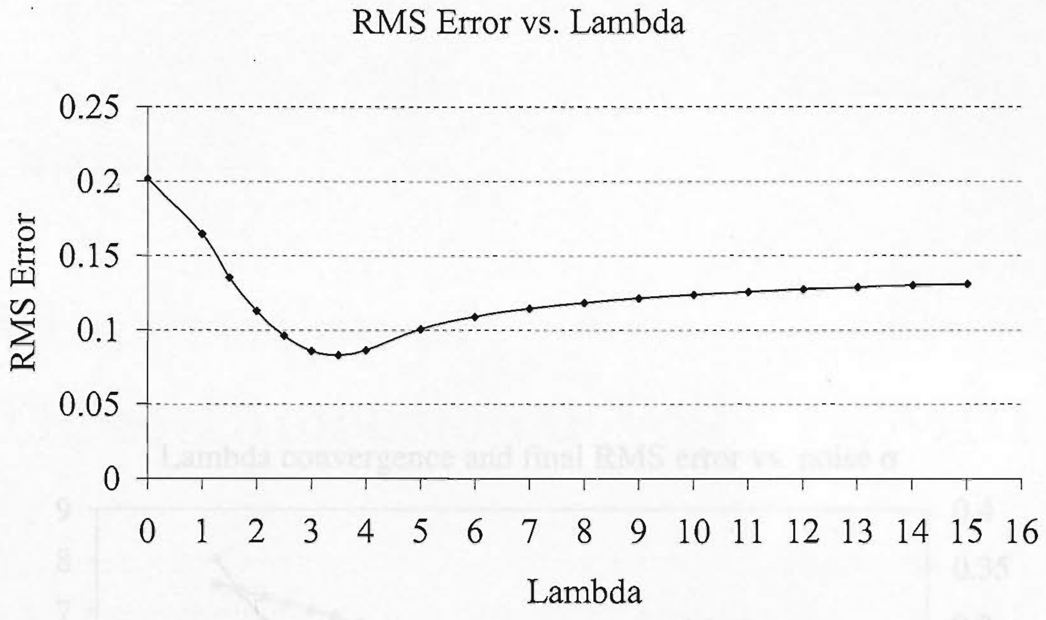


FIGURE 5.5: A plot of the RMS error against the value of λ for restoration of the 3D simulated heart dataset.

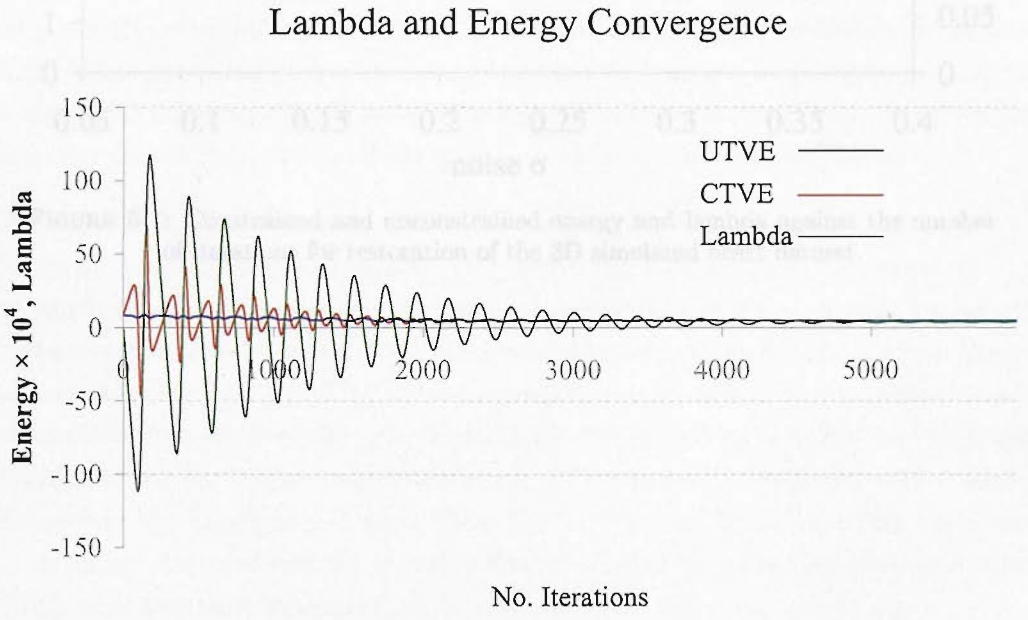


FIGURE 5.6: Constrained and unconstrained energy and lambda against the number of iterations for restoration of the 3D simulated heart dataset.

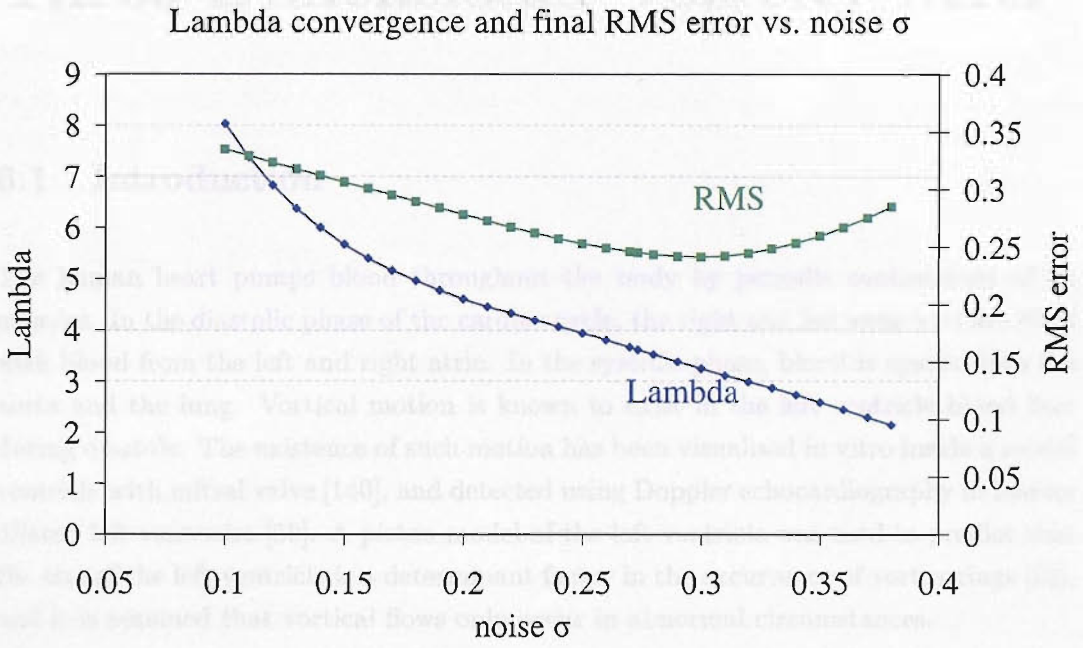


FIGURE 5.7: Constrained and unconstrained energy and lambda against the number of iterations for restoration of the 3D simulated heart dataset.

Chapter 6

Visualisation of Three-Dimensional Velocity MRI

6.1 Introduction

The human heart pumps blood throughout the body by periodic contractions of its muscles. In the diastolic phase of the cardiac cycle, the right and left ventricles are filled with blood from the left and right atria. In the systolic phase, blood is ejected into the aorta and the lung. Vortical motion is known to exist in the left ventricle blood flow during diastole. The existence of such motion has been visualised in vitro inside a model ventricle with mitral valve [140], and detected using Doppler echocardiography in human dilated left ventricles [58]. A piston model of the left ventricle was used to predict that the size of the left ventricle is a determinant factor in the occurrence of vortex rings [23], and it is assumed that vortical flows only occur in abnormal circumstances.

With the recent advances in *in vivo* MR flow imaging techniques, a large amount of velocity data can be acquired rapidly, thus providing a comprehensive measurement of flow information. With the increasing use of combined MRI/CFD approach, the amount of data that needs to be interpreted is becoming significant and challenging to analyse and visualise (Figure 6.2). This is true especially for simulated flow in major cardiac chambers through the cardiac cycle. To examine detailed changes in flow topology, data reduction based on feature extraction needs to be applied. Streamlines give a good indication of the transient pattern of flow [273]. However, for 3D datasets these plots can be highly cluttered and for complex flow they tend to intertwine with each other, limiting their practical value [111].

This chapter presents a new method for velocity MR flow field visualisation based on flow clustering and automatic streamline selection. Flow clustering enables data simplification and compression. The method assumes linearity around critical points (see

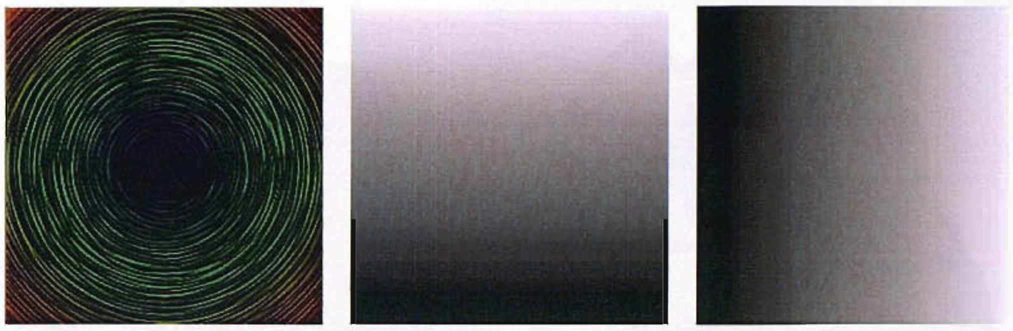

(a) Vortex streamlines. (b) x-velocity component. (c) y-velocity component.

FIGURE 6.1: Synthetic vortex and magnitude of its velocity components showing planar gradients.

Figure 6.1) to ensure that these are preserved by the simplification process. Each cluster therefore contains points sharing a common flow feature. Automatic streamline selection is then applied to determine the salient flow features that are important to the vessel morphology.

6.2 Previous Work

6.2.1 Occurrence of Vortical Flow in the Cardiac Cycle

The human heart pumps blood throughout the body by periodic contractions of its muscles. In the diastolic phase of the cardiac cycle, the right and left ventricles are filled with blood from the left and right atria. In the systolic phase, blood is ejected into the aorta and the lung. Vortical motion is known to exist in the left ventricle blood flow during diastole. The existence of such motion has been visualised in vitro inside a model ventricle with mitral valve [140], and detected using Doppler echocardiography in human dilated left ventricles [58]. A piston model of the left ventricle was used to predict that the size of the left ventricle is a determinant factor in the occurrence of vortex rings [23], and it is assumed that vortical flows only occur in abnormal circumstances.

In contrast with the echography-based technique, colour Doppler velocity mapping, which presently only provides uni-directional velocity information, MR velocity mapping permits multi directional quantification of flow, with a spatiotemporal resolution that is suitable for detailed flow pattern analysis. Vortical flow in the left ventricle has been detected and characterised previously using the winding number method [271]. In that work, false candidates are successfully removed using a combination of candidate cluster analysis and phase theory (see below). However, the winding number method is not readily extendable to 3D data (see Yang *et al.* 1998 [271] for a definition). A 3D template matching strategy has been described in the literature with promising results [26] and reported to detect a vortex pair. A previous rendering technique based on arrow

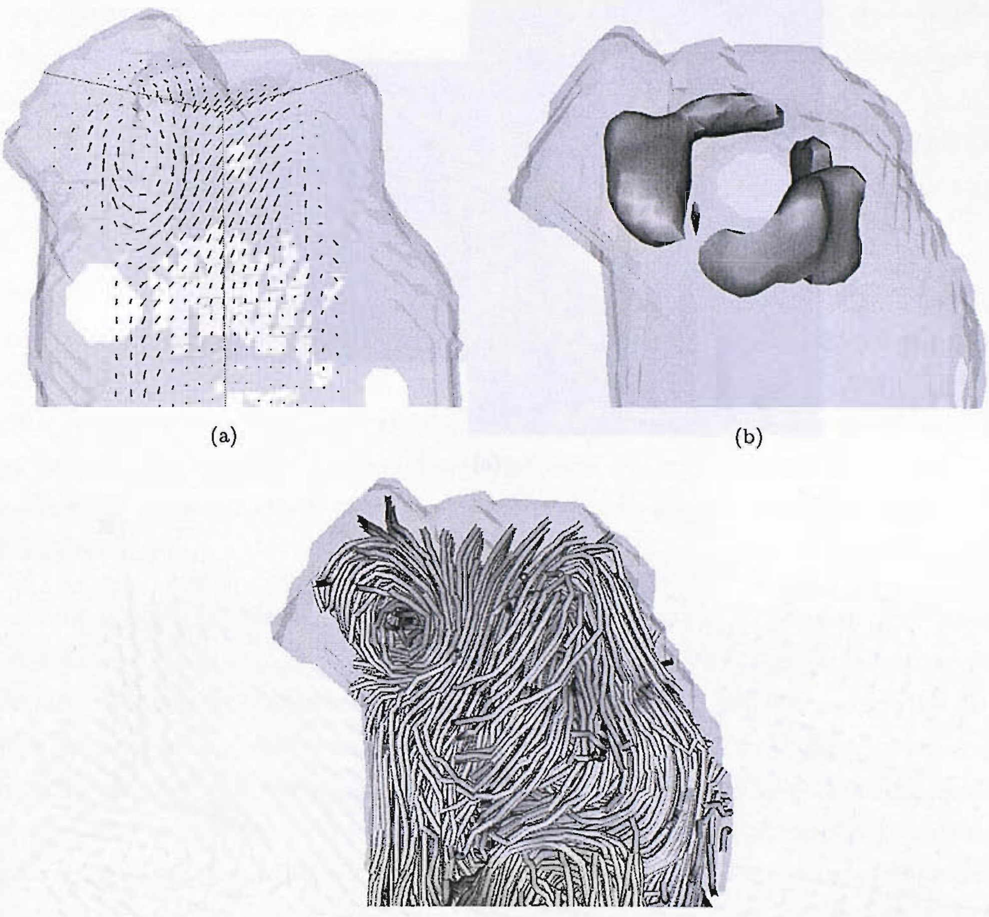
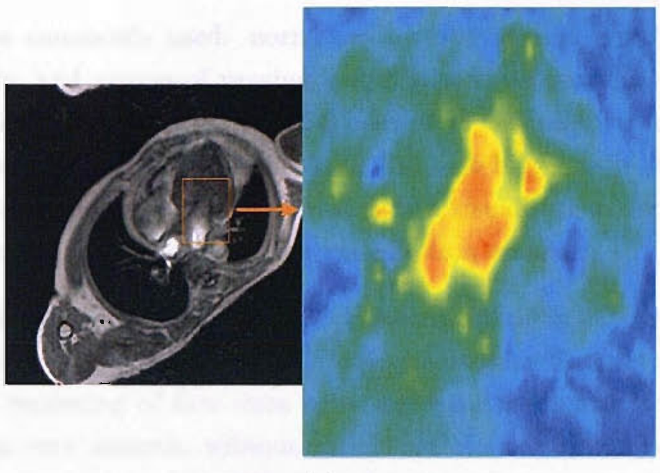


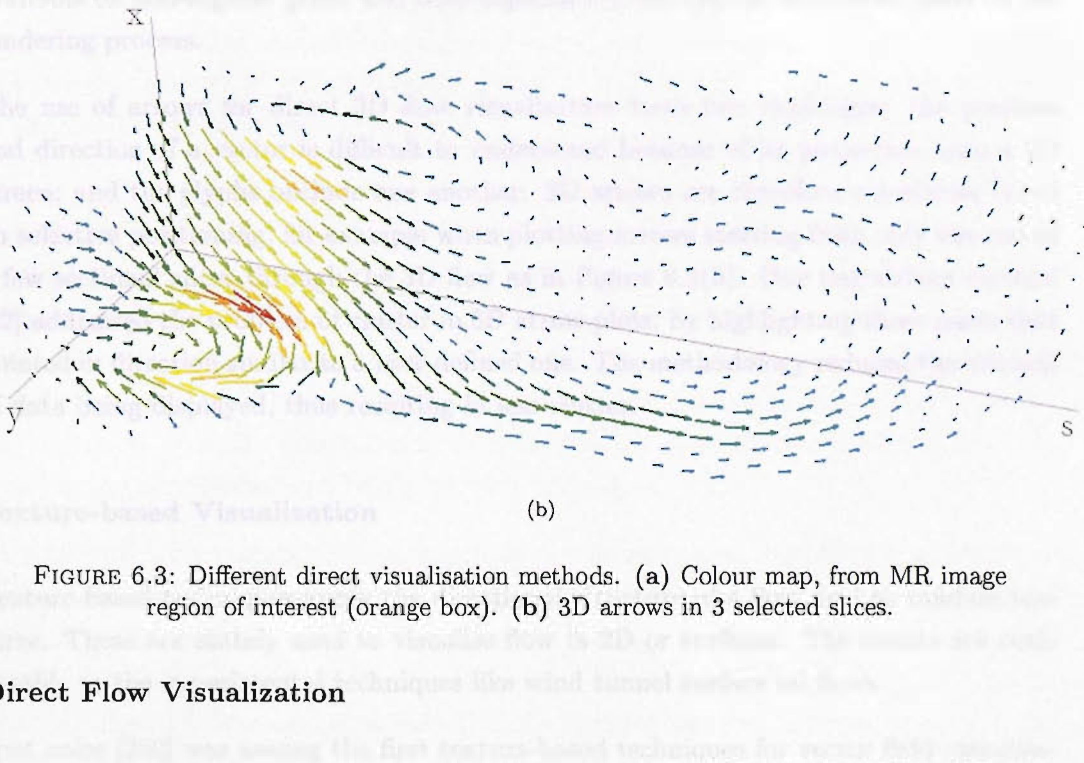
FIGURE 6.2: Different flow field visualisation methods. (a) Arrow plot. (b) Interactive iso-vorticity plot. (c) Streamline plot.

depiction of velocity MR [259] found one vortex line, a result confirmed by their earlier work [131]. A more recent velocity MR study involving 11 healthy patients also reports detection of a single vortex during diastole using streamline visualisation [84]. Given these multiple observations, in this work, we assume the existence of simple vortices and also of vortex rings. *Vortex rings* are those where the centre of vortical motion, i.e. the *vortex core*, forms a circle or ring. With appropriate visualisation, a perfectly circular vortex ring will display a toroidal structure.

6.2.2 Vector Field Visualisation

Currently available techniques can be categorised into four groups: direct flow, texture-based, geometric and feature-based flow visualisation.


(a)



(b)

FIGURE 6.3: Different direct visualisation methods. (a) Colour map, from MR image region of interest (orange box). (b) 3D arrows in 3 selected slices.

Direct Flow Visualization

In this category, the data is visualised directly without much pre-processing. One possibility is to map flow attributes such as velocity, pressure or temperature to colour (Figure 6.3(a)). Since colour plots are widely accepted, this approach results in very intuitive depictions. The colour scale used for mapping must be chosen to respect perceptual differentiation. Colour coding for 2D flows extends well to time-dependent data, resulting in moving colour plots according to changes of flow properties over time. As a middle ground between 2D and the visualisation of truly 3D data is the restriction to subparts of the 3D domain, i.e. sectional slices or boundary surfaces.

Another direct method is to map a line, arrow or glyph to each sample point in the field, oriented according to the flow field, as in Figure 6.3(b). Usually a regular placement of arrows is used in 2D, for example on an evenly spaced Cartesian grid. Two variants of arrow plots are commonly used: normalised arrows of unit length, for visualising flow direction only; and arrows of varying length that is proportional to flow velocity, sometimes called *hedgehog visualisation* [134] because of the bristly result. 2D hedgehog plots can be extended to time-dependent data, although bigger time steps might result in jumping arrows, diminishing the quality of such a visualisation. Arrow plots can be mixed with colour coding to provide richer visualisation results.

The natural extension of colour coding in 2D is colour coding in 3D. However, occlusion and complexity make it difficult or impossible to get an immediate overview of the flow volume. Volume rendering of flow data must address fundamental issues [70]. Flow datasets are often very smooth, without sharp and clear "object" boundaries. This makes mapping to opacities more difficult and less intuitive. Flow data is also often available on non-regular grids, and time-dependency can impose additional loads on the rendering process.

The use of arrows for direct 3D flow visualisation faces two challenges: the position and direction of a vector is difficult to understand because of its projection onto a 2D screen; and the glyphs occlude one another. 3D arrows are therefore sometimes based on selective positioning, for example when plotting arrows starting from only one out of a few sectional slices through the 3D flow as in Figure 6.3(b). One user-driven method [22] addressed the problem of clutter in 3D arrow plots, by highlighting those parts that pointed in direction similar to a user-defined one. The methodology reduced the amount of data being displayed, thus resulting in less clutter.

Texture-based Visualization

Texture-based techniques apply the directional structure of a flow field to random textures. These are mainly used to visualise flow in 2D or surfaces. The results are comparable to the experimental techniques like wind tunnel surface oil flows.

Spot noise [250] was among the first texture-based techniques for vector field visualisation. Spot noise generates a texture by distributing a set of intensity functions, or spots, over the domain. Each spot represents a particle moving over a small step in time and results in a streak in the direction of the local flow from where the particle is seeded. One limitation of the original spot noise algorithm was the lack of velocity magnitude information in the resulting texture. Enhanced spot noise [57] was introduced to address this problem. Spot noise has also been used for the visualisation of simulated turbulent flow [56].

Line integral convolution (LIC) [29], detailed in section 3.4.1 on page 35, takes as its two inputs a vector field and a white noise texture of the same size. The white noise texture is locally filtered, or smoothed, along the path of streamlines to acquire a dense visualisation of the flow field. Directional cues were added [217] by combining animation and introducing dye paths into the computation. A multi-frequency noise texture has been proposed [133]. The spatial frequency of the noise is a function of the magnitude of the local velocity in the field. The result of the method contains streaks with variable resolution and length, depending on the velocity magnitude.

Much research has been dedicated to bringing LIC computation to interactive rates. Coherence along streamlines has been exploited [227, 100], as well as parallel implementations [30, 285]. In 3D, further challenges arise due to demanding memory requirements, occlusion and visual complexity (see 3.8 on page 40). A 3D texture mapping approach combined with an interactive clipping plane [200] has addressed the problems of occlusion and interaction. A combined approach of direct volume rendering and LIC was taken [112] for extending LIC to 3D. Some perceptual difficulties have been identified and addressed [111, 112], including emphasising important regions of interest in the flow, enhancing depth perception, and improving orientation perception of overlapping streamlines. Texture advection has also been applied to 3D and 4D flows [124], resulting in effective steady-state flow animations.

Geometric Visualization

Geometric objects are extracted from the data, and used for the visualisation. The objects can be streamlines, stream surfaces, time surfaces or flow volumes. These geometric objects are directly related to the data, and the results are comparable to experimental dye advection or smoke wind tunnels. This section reviews isosurfaces, streamlines and their numerous variants, and finally particle tracing.

Contours work on the scalar data, similarly to colour coding, but display a boundary between two distinct regions. The user may be interested in transition areas in the vector field. In a colour plot, transitions are shown by a change of colour. With contouring, an explicit line or curve is shown. In 3D, contouring results in the use of isosurfaces for 3D flow visualisation. The isovalues have to be selected with care, because of the smooth nature of the flow data. If the flow characteristic to be plotted lacks sharp transitions, any chosen isovalues may be arbitrary and lack intuitive interpretation. There are, however, useful applications of isosurfaces to flow data, for example in the visualization of shock waves or burning fronts in simulated combustion. Figure 6.2(b) shows a selected isosurface of the flow curl, or vorticity.

Streamlets [150] are generated by integrating flow vectors along a short path. Despite being small, they communicate temporal evolution along the flow. Their main advantage over LIC is their depiction of flow direction. They are difficult to extend to 3D because of rendering and perceptual problems.

Steamlines (as implemented in section 6.3.3, below) result from longer integration, and offer intuitive semantics: users easily understand that flows evolve along natural objects.

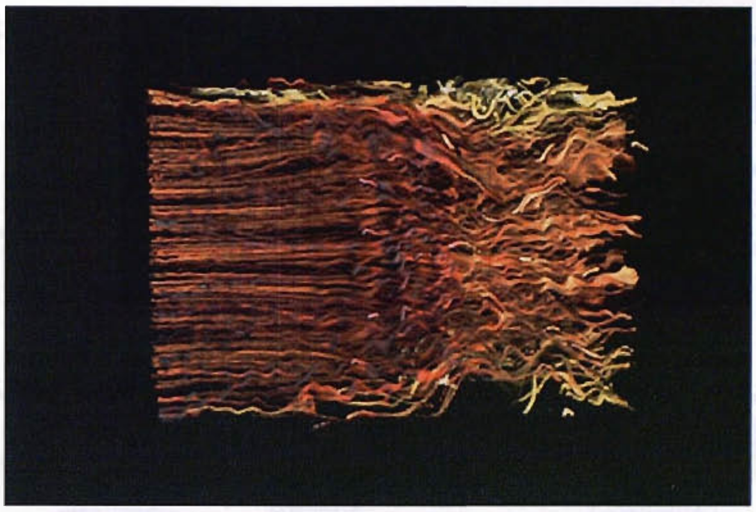
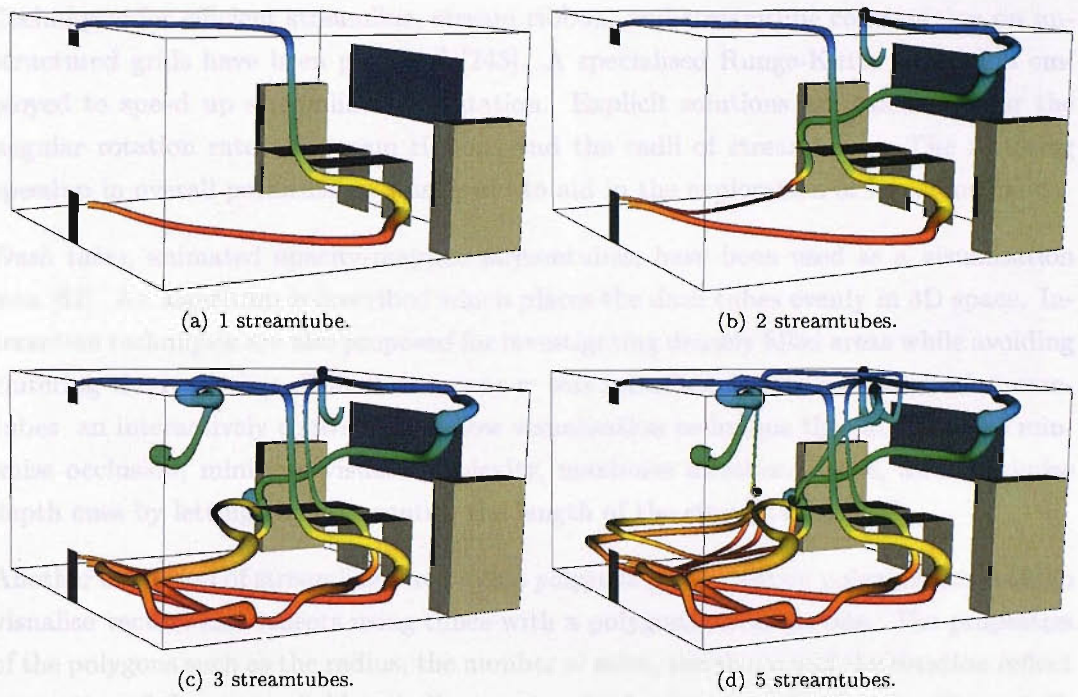


FIGURE 6.4: 3D LIC volume rendering showing a simulated planar and breaking flow. Color varies from red to yellow with increasing temperature, showing the effects of friction across the boundary layers [111].

At NASA, streamlines are used to visualise aerospace CFD data [6]. Careful seeding is necessary to obtain useful results, since visual clutter often becomes a problem (see Figure 6.4). A technique known as illuminated streamlines was aimed at improved perception of streamlines in 3D by taking advantage of the texture mapping capabilities supported by graphics hardware [284] (see Figure 6.5(a)). A shading technique is used to render semi-transparent streamlines which increases depth information. That also addresses the problem of occlusion. For seeding the streamlines, that work proposes a seeding probe that can be moved interactively to start streamlines at specific points of interest.

Other seeding strategies have been proposed. One important aspect of streamlines is the choice of initial conditions. Evenly distributed seed points do not result in evenly spaced streamlines, therefore special algorithms need to be employed. Automatic distance-based seeding techniques have been developed to achieve a uniform distribution of streamlines on a 2D vector field [239, 119]. This has been extended to generate evenly distributed streamlines on boundary surfaces within curvilinear grids [158]. Streamline seeding strategies may also be topology-based. A seed placement strategy has been presented for streamlines based on flow features of the data set [253]. The method depicted flow patterns near critical points in the flow field, but was only trialled in controlled synthetic images. A multiresolution method was presented for visualising large 2D steady-state vector fields [121]. The resulting multiresolution hierarchy supported information enrichment and zooming. The user is able to interactively set the density of streamlines while zooming in and out of the vector field. The density of streamlines is also computed automatically as a function of velocity or vorticity.


(a) 1 streamtube. (b) 2 streamtubes.

(c) 3 streamtubes. (d) 5 streamtubes.

FIGURE 6.5: Streamtubes of a simulated office air conditioning dataset [212].

Seeding becomes a special challenge when dealing with time-dependent data. An unsteady flow algorithm [120] correlates instantaneous streamline visualisations of the vector field. For each frame, a feed forward algorithm computes a set of evenly-spaced streamlines as a function of the streamlines generated for the previous frame. The method also provides full control of the image density so that smooth animations of arbitrary density can be produced.

Streamlines are also used for visualising boundary flows or sectional slices through 3D flow [76]. However, it must be noted that the use of these objects on slices may be misleading, even within steady flow datasets. A streamline on a slice may depict a closed loop, even though no particle would ever traverse the loop. The reason lies in the fact that flow components which are orthogonal to the slice are ommited during flow integration.

A *stream ribbon* is a 3D streamline with a strip added, to also visualise rotational behaviour of the flow, which is not possible with streamlines alone [243]. A streamtube is a thick streamline that can be extended to show the expansion of the flow. Stream ribbons and streamtubes offer advantages over streamlines in that they can encode more properties, such as divergence and convergence of the vector field, in the properties of the rendered geometric objects. Figure 6.5 shows streamtubes of a simulated office air conditioning system. Seeds were set manually and it can be seen that having more streamtubes aid in the understanding of the flow, but also add to the plot's complexity and clutter.

Techniques for efficient streamline, stream ribbon, and streamtube construction on unstructured grids have been proposed [243]. A specialised Runge-Kutta method is employed to speed up streamline computation. Explicit solutions are calculated for the angular rotation rates of stream ribbons and the radii of streamtubes. The resulting speedup in overall performance was found to aid in the exploration of large flow fields.

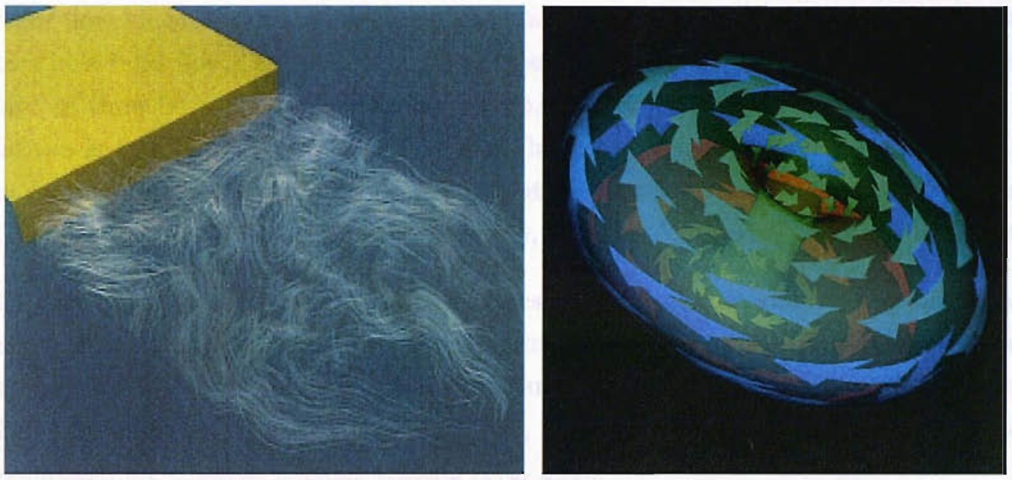
Dash tubes, animated opacity-mapped streamtubes, have been used as a visualisation icon [82]. An algorithm is described which places the dash tubes evenly in 3D space. Interaction techniques are also proposed for investigating densely filled areas while avoiding clutering the rendering. The *stream runner* was introduced as an extension of streamtubes an interactively controlled 3D flow visualisation technique that attempts to minimise occlusion, minimise visual complexity, maximise directional cues, and maximise depth cues by letting the user control the length of the streamtubes [138].

Another extension of streamlines are *stream polygons* [213]. Stream polygons are tools to visualise vectors and tensors using tubes with a polygonal cross section. The properties of the polygons such as the radius, the number of sides, the shape and the rotation reflect properties of the vector field including strain, displacement, and rotation. *Streamballs* are a technique which visualises divergence and acceleration in fluid flow [27]. Streamballs split or merge depending on convergence/divergence and acceleration/deceleration, respectively. *Streakballs* were introduced and compared with other approaches to accelerate particle tracing on sparse grids and curvilinear sparse grids for unsteady flow data [231].

Yet another extension to streamlines are *stream surfaces*, which are surfaces that are everywhere tangent to a vector field. A stream surface can be approximated by connecting a set of streamlines along timelines (and varying the number of streamlines used according to convergence or divergence of the flow) [108]. Stream surfaces are very good for texture-based visualisation techniques such as spot noise and LIC, because there is no cross-flow component normal to the surfaces, i.e. the vector field is not projected like it is for 2D slices through a 3D domain. Stream surfaces present challenges related to occlusion, visual complexity, and interpretation.

An interactive flow visualisation technique was presented using stream surfaces [107]. Two follow-up techniques were proposed for generating implicit stream surfaces [252]. Some issues and solutions associated with the placement and orientation of stream surfaces in 3D have been identified [31].

Stream arrows have been presented as an enhancement of stream surfaces by separating arrow-shaped portions from a stream surface [151, 152] (see Figure 6.6(b)). Stream arrows address the problem of occlusion associated with 3D flow visualisation, but especially with stream surfaces. Stream arrows also provide additional information about the flow, usually not seen with stream surfaces, such as flow direction and convergence/-divergence. Stream surfaces have been simulated by a tecnhique based on large sets of


(a) Illuminated streamlines [284]. (b) Stream arrows [152].

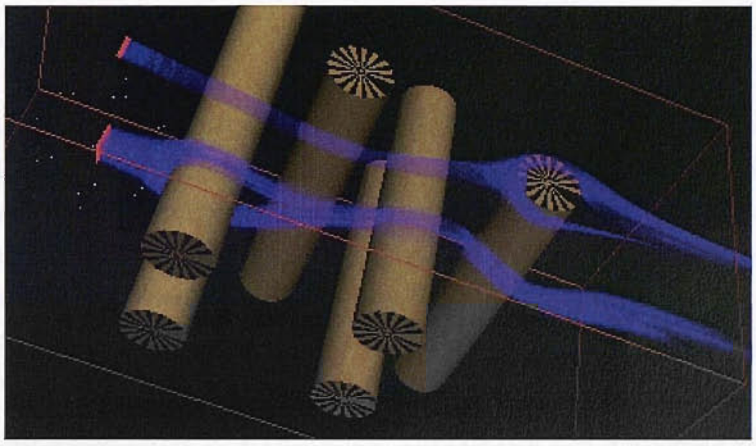

(c) Flow volumes [13].

FIGURE 6.6: Examples of flow visualisation using geometric objects.

so-called *surface particles* [251]. Surface particles exhibit less occlusion when compared to stream surfaces.

Finally, streamlines have been extended to flow volumes (see Figure 6.5(c)). A flow volume is a specific subset of a 3D flow domain, which is traced out by a particular initial 2D patch over time [13]. The resulting volume is divided up into a set of semitransparent tetrahedra, which are volume rendered in hardware. Flow volumes were extended to unsteady flow [13]. The resulting unsteady flow volumes are the 3D analogue of streaklines. Considerations are made when extending the visualisation technique to unsteady flows since particle paths may become convoluted in time. The authors present some solutions to the problems which occur in subdivision, rendering, and system design. The resulting algorithms are applied to a variety of flow types including curvilinear grids.

As referred in section 3.4.1, streamline plots refer to steady-state flow, or if they are applied to one time step of an unsteady flow dataset are referred to as transient streamlines [273]. When unsteady flow data are investigated, several distinct integral objects are used for flow visualisation. A *pathline* or *particle trace* is the trajectory that a particle follows in a fluid flow. A *timeline* joins the positions of particles released at the same instant in time from different insertion points, i.e. joins points at constant time *t*. A *streakline* is traced by a set of particles that have previously passed through a unique point in the domain. Streaklines relate to continuous injection of foreign material into real flow, in that they can be seeded adaptively, e.g. as a function of local vorticity [209].

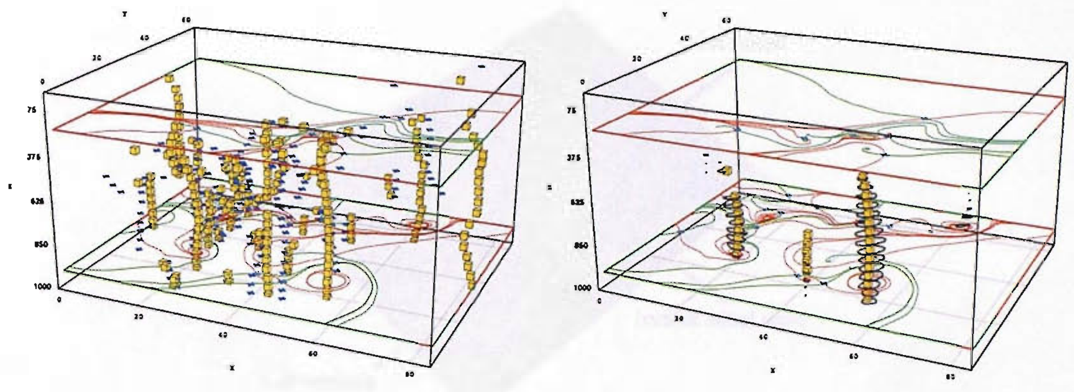
A 3D particle tracing algorithm has been presented for the investigation of large, unsteady aeronautical simulations [128]. Different integration methods have been compared to evaluate the trade-off beween computation time and accuracy, and a sparse grid particle tracing algorithm proposed [232, 233]. Various other methods have been implemented for efficient particle tracing [179, 206].

Feature-Based Visualization

A final approach for visualising flow data is the feature-based approach, in which the flow data is visualised at a high level of abstraction. The flow data is described by features, which represent the interesting objects or structures in the data. The original data set is then no longer needed. Because often, only a small percentage of the data is of interest, and the features can be described very compactly, an enormous data reduction can be achieved. This makes it possible to visualise even very large data sets interactively. This section reviews feature-based visualisation methods, and feature tracking.

The goal of feature extraction is determining, quantifying and describing the features in a data set. A feature can be loosely defined as any object, structure or region that is of relevance to a particular research problem. In each application, in each data set and for each researcher, a different feature definition could be used. Common examples in fluid dynamics are vortices, shock waves, separation and attachment lines, recirculation zones and boundary layers. Although most feature detection techniques are specific for a particular type of feature, in general the techniques can be divided into three approaches: based on image processing, on topological analysis, and on physical characteristics.

Image processing techniques were originally developed for analysis of 2D and 3D image data, usually represented as scalar (greyscale) values on a regular rectangular grid. The problem of analysing a numerical data set, represented on a grid, is similar to analysing an image data set. Therefore, basic image processing techniques can be used for feature extraction from scientific data. A feature may be distinguished by a typical range of data values, just as different tissue types are segmented from medical images. Edges or boundaries of objects are found by detecting sudden changes in the data values, marked by high gradient magnitudes. Thus, basic image segmentation techniques, such as thresholding, region growing, and edge detection can be used for feature detection. Also, objects may be quantitatively described using techniques such as skeletonisation



(a) All critical points selected. (b) Critical points filtered by thresholded vorticity=10.

FIGURE 6.7: Climate simulation vector field simplified with a feature filter [268].

or principal component analysis. However, a problem is that in CFD simulations, grid types are often used such as structured curvilinear grids, or unstructured tetrahedral grids. Many techniques from image processing must be adapted for use with such grids, or some form of data conversion is necessary.

A second approach to feature extraction is the topological analysis of 2D linear vector fields, as introduced in section 3.4.2 on page 39. This is based on detection and classification of critical points. These are the points where the vector magnitude is zero. By computing the eigenvalues and eigenvectors of the velocity gradient tensor, the critical points can be classified and tangent curves can be computed. Using this information, schematic visualisations of the vector field have been generated for 2D, time-dependent and 3D flows.

An algorithm has been presented for visualising nonlinear vector field topology [210], because other known algorithms are based on piecewise or bi-linear interpolation, which destroys the topology in case of nonlinear behaviour. The work makes use of Clifford algebra for computing polynomial approximations in areas with nonlinear local behaviour, especially higher-order singularities.

A climate simulation vector field containing 442 critical points [268] was simplified with a feature filter to display only its 35 most relevant ones (see Figure 6.7). The algorithm finds the vorticity magnitude of the vector field, and thresholds it. It then selects the critical points lying in vorticity magnitude locations above the threshold. The reasoning behind the method is that vorticity measures the strength of shear and circulation, which are high around critical points of interest. The authors were satisfied that manual tuning of the threshold eliminated most irrelevant points, while keeping the important ones.

A threshold-based spot noise method was introduced [55] that interactively changes the scale or "level" of the topology. This is done by a pair distance filter, which selectively eliminates critical points that are more than a user-input distance away from each

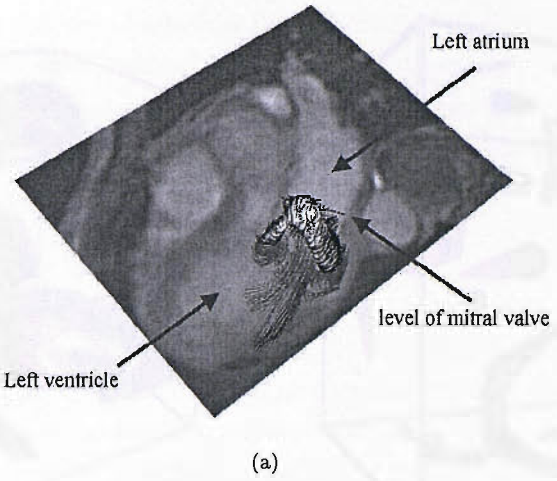
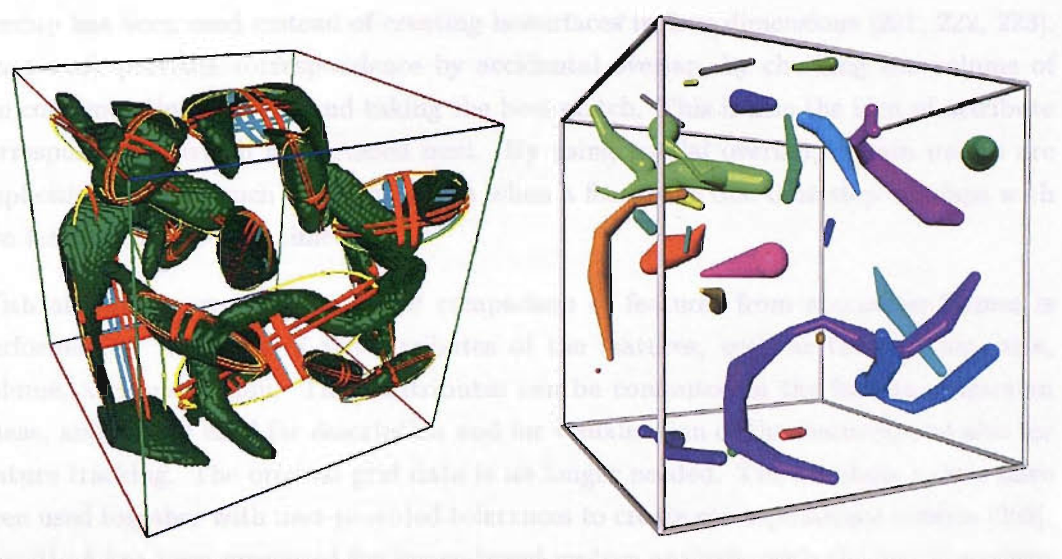


FIGURE 6.8: Vortex ring inside an *in vivo* left ventricle detected using template matching [26]. The vortex core is visualised as a white vorticity isosurface, and streamlines are seeded from within the isosurface.

other. The method was applied to highly turbulent CFD simulated flow, and was found to remove unimportant detail. However, it also introduced some flow inconsistencies when deleting critical points. For example, if three critical points are present with approximately the same distance between each other, deleting the pair with the minimum distance may not be the correct choice. Also deleting a pair could result in a topology with two close foci rotating in the same direction. An improved algorithm has been presented [53], which maintains flow consistency by introducing critical point indices. The indices are related to the number of revolutions the flow field makes by traversing a closed curve. A topology is thus simplified by ensuring index-based consistency. While an improvement, this method was found too slow for interactive use. Further modification [54] achieved 2D, but not 3D, interaction.

A 3D template matching strategy has been applied to 3D Cardiac Velocity MRI [26] and used to detect a vortex ring inside an *in vivo* left ventricle. The method computes a similarity tensor by convolving oriented templates of a 3D vortex core pattern with velocity data. Isosurfaces are then derived by thresholding the similarity data. The detection of the vortex pair by this method is corroborated in that work by streamlines that show swirling flow near the thresholded volume (Figure 6.8).

If points selected by feature extraction are used to seed geometric objects such as streamlines or streamtubes, this can provide useful information about the features and their origin. That has been applied to visualising simulated flow past a step wall [249], resulting in only two streamtubes being needed to depict both vortical and planar flow. Another way to combine feature detection with geometrical visualisation is by computing parametric icons from feature attributes. A CFD simulation of flow past a tapered cylinder [207] visualised features through a combination of vorticity isosurfaces and ellipsoids (Fig. 6.9). The ellipsoid fitting is computed from detected vortex centres using



(a) Ellipsoids fitted to critical points, combined with vorticity isosurfaces [207].

(b) Skeleton graph rendering, with turbulent vortex structures represented by skeleton icons [196].

FIGURE 6.9: A comparison of ellipsoids and skeleton graphs for 3D critical point visualisation.

the winding angle method (section 3.4.2). The ellipsoids do not give a good indication of the shape of the features, but their position and orientation are accurate and can be used for feature tracking. Another type of icon has been proposed to visualise the strongly curved 3D features. Skeleton graph descriptions for features have been proposed [196], with which icons are created that describe the topology of the features, and approximate their shape.

In time-dependent data sets, features are objects that evolve in time. Determining the correspondence between features in successive time steps, that actually represent the same object at different times, is called the correspondence problem. Feature tracking is involved with solving this correspondence problem. The goal of feature tracking is to be able to describe the evolution of features through time. During the evolution, certain events can occur, such as the interaction of two or more features, or significant shape changes of features. Event detection is the process of detecting such events, in order to describe the evolution of the features even more accurately.

There are two popular approaches to solving the correspondence problem, region correspondence and attribute correspondence. Region correspondence involves comparing the regions of interest obtained by feature extraction. The binary images from successive time steps, containing the features found in these time steps, are compared on a cell-to-cell basis. Correspondence can be found using a minimum distance or a maximum cross-correlation criterion [110] or by minimising an affine transformation matrix [123]. It is also possible to extract isosurfaces from the four-dimensional time dependent data set [263], where time is the fourth dimension. The correspondence is then implicitly determined by spatial overlap between successive time steps. The criterion of spatial overlap has been used instead of creating isosurfaces in four dimensions [221, 222, 223]. That work prevents correspondence by accidental overlap, by checking the volume of the corresponding features and taking the best match. This is also the idea of attribute correspondence, which is discussed next. By using spatial overlap, certain events are implicitly detected, such as a *bifurcation* when a feature in one time step overlaps with two features in the next time step.

With attribute correspondence, the comparison of features from successive frames is performed on the basis of the attributes of the features, such as the position, size, volume, and orientation. These attributes can be computed in the feature extraction phase, and can be used for description and for visualisation of the features, and also for feature tracking. The original grid data is no longer needed. The attribute values have been used together with user-provided tolerances to create correspondence criteria [208]. A method has been presented for image-based motion analysis, with the use of markers or tokens [215]. This is based on smoothness of motion of feature point trajectories in property space. Properties or attributes of features are represented by points in property space. These points move through the property space over time, and the algorithm tries to find the smoothest paths or trajectories in this property space. The notion of *property coherence* is used, that is, the properties are supposed to change gradually.

A feature tracking algorithm has been proposed based on prediction and verification [197, 198]. This algorithm is based on the assumption that features evolve predictably. If a part of the evolution of a feature (*path*) has been found, a prediction can be made into the next time step (*frame*). Then, in that next time step, a feature is sought, that corresponds to the prediction. If a feature is found that matches the prediction within certain user-provided tolerances, the feature is added to the evolution and the search is continued to the next time step. When no more features can be added to the path, a new path is started. In this manner, all frames are searched for starting points, both in the forward and backward time directions, until no more paths can be created.

A path is started by trying all possible combinations of features from two consecutive frames and computing the prediction to the next frame. Then, the prediction is compared to the candidate features in that frame. If there is a match between the prediction and the candidate, a path is started. To avoid any erroneous or coincidental paths, there is a parameter for the minimal path length, which is usually set to 4 or 5 frames. A candidate feature can be defined in two ways. All features in the frame can be used as candidates, or only unmatched features can be used, that is, those features that have not yet been assigned to any path. The first definition ensures that all possible combinations are tested and that the best correspondence is chosen. However, it could also result in features being added to more than one path. This has to be resolved afterwards. Using the second definition is much more efficient, because the more paths are found, the less unmatched features have to be tested. However, in this case, the results depend on the order in which the features are tested. This problem can be solved by starting the tracking process with strict tolerances and relaxing the tolerances in subsequent passes.

The prediction of a feature is constructed by linear extrapolation of the attributes of the features from the last two frames. Other prediction schemes could also be used, for example, if *a-priori* knowledge of the flow is available. The prediction is matched against real features using correspondence criteria [208].

After feature tracking has been performed, event detection is the next step. Events are the temporal counterparts of spatial features in the evolution of features. For example, if the path or evolution of a feature ends, it can be interesting to determine why that happens. It could be that the feature shrinks and vanishes, or that the feature moves to the boundary of the data set and disappears, or that the feature merges with another feature and the two continue as one.

A feature tracking system has been presented that is able to detect these and other events [198]. A simulated flow past a tapered cylinder contains vortices with long and thin 3D structures. These have been visualised by performing feature tracking in the *spatial* dimension, in addition to tracking over time [199]. First, feature extraction was performed in the 2D slices. This resulted in two-dimensional vortices, which were represented by a special type of ellipse icon. Next, these 2D features were tracked in the z-direction, forming 3D vortex structures. The 2D icons are used consisting of ellipses with a number of curved spokes. The curvature of the spokes indicates the rotational direction of the vortices, and the number of spokes represents the rotational speed. 3D icons are constructed by connecting the centre points of the 2D ellipses.

In the same work, the tracked features are also plotted along the time dimension. In that work, a software interface presents the user with a graph which gives an abstract overview of the entire data set. The time steps are on the horizontal axis, and the features represented by nodes on the vertical axis. The correspondences between features from consecutive frames are represented by edges in the graph, and therefore the evolution of a feature in time is represented by a path in the graph.

6.2.3 Data Clustering

A key element in data analysis procedures is the grouping, or classification of measurements based on either goodness of fit to a particular model, or natural groupings (clustering) revealed through analysis. Cluster analysis is the organisation of a collection of patterns into clusters based on similarity. These patterns are usually represented as a vector of measurements, or a point in multidimensional space. Intuitively, patterns with a valid cluster are more similar to each other than they are to a pattern belonging to a different cluster. The variety of techniques for representing data, measuring similarity between data elements, and grouping elements has produced a large variety of clustering methods.

Clustering is a form of unsupervised classification, different from discriminant analysis, a form of supervised classification [163]. In supervised classification, there exists a collection of labelled, or preclassified patterns; the problem is to label a newly encountered, yet unlabelled, pattern. Typically, the given labelled training patterns are used to learn the descriptions of classes that in turn are used to label a new pattern. In the case of clustering, the problem is to group a given collection of unlabelled patterns into meaningful clusters. In a sense, labels are associated with clusters also, but these category labels are data driven, as they are obtained from the data alone.

Clustering is useful in several exploratory pattern analysis, grouping, decision-making and machine learning situations, including data mining, document retrieval, image segmentation and pattern classification. However, in many such problems there is little prior information available about the data, and the decision-maker must make as few assumptions about the data as possible. It is under these restrictions that clustering methodology is particularly appropriate for the exploration of interrelationships among the data points to make an assessment of their structure.

Typical clustering activity involves the following steps [114]:

Pattern representation refers to the number of classes, the number of available patterns, and the number, type and scale of the features available to the clustering algorithm. Some of this information may be imposed by the application. Feature selection is the process of identifying the most effective subset of the original features to use in clustering. Feature extraction is the use of one or more transformations of the input features to produce new salient features. Either or both of these techniques can be used to obtain an appropriate set of features to use in clustering [65, 91]. As described below, the implementation approach taken in this Chapter's application has been to use local linear expansion, and the resulting linear representation of the vector field at each data voxel, as the selected feature.

Pattern proximity is usually measured by a distance function defined on pairs of patterns. A simple distance measure like Euclidean distance can often be used to reflect similarity between two patterns. This is defined in 2D as:

$$d_2(\boldsymbol{x}_i, \boldsymbol{x}_j) = \sqrt{\sum_{k=1}^{d}(x_{i,k} - x_{j,k})^2} = ||\boldsymbol{x}_i - \boldsymbol{x}_j|| \tag{6.1}$$

This is the measure used in this Chapter's implementation. Other distance measures can be used to characterise the conceptual similarity between patterns [165]. In multi-feature applications, there is a tendency for the largest-scale feature to dominate the others. This can be solved by normalisation or other weighting schemes. Linear correlation among features can also distort distance measures, which can be alleviated by using the Mahalanobis distance metric. An example of the use of the Mahalanobis distance can be found in section 2.2 on page 20.

The *grouping* step can be performed in a number of ways. One possible distinction is that clustering output can be hard, such as a partition of data into groups, or fuzzy, where each pattern has a variable degree of membership in each of the output clusters [279, 188]. Hierarchical clustering algorithms, such as the present Chapter's application, produce a nested series of partitions based on a criterion for merging or splitting clusters based on similarity. Partitional clustering algorithms identify the partition that optimises, globally or locally, a clustering criterion [61]. The k-means is the simplest and most commonly used algorithm, employing a squared error criterion [19]. Another possible distinction relevant to the present work is between agglomerative and divisive clustering. This aspect relates to algorithmic structure and operation. An agglomerative approach begins with each pattern in a distinct cluster, and successively merges clusters together until a stopping criterion is satisfied. A divisive method begins with all patterns in a single cluster and performs splitting until a stopping criterion is met. The grouping algorithm implemented in the present work is therefore classified as hierarchical agglomerative.

Data abstraction is the process of extracting a simple and compact representation of a data set. Here, simplicity serves the purpose of either automatic analysis, so that a machine can perform further processing efficiently, or it is human-oriented, so that the representation obtained is intuitive and easy to interpret. In the clustering context, a typical data abstraction is a compact description of each cluster, usually in terms of cluster prototypes or representative patterns such as the centroid. The centroid is the most popular scheme, however it fails when the clusters are elongated or non-isotropic, such as the ones found in our application. In such cases, the use of a collection of boundary points in a cluster generally captures its shape well in 2D. The number of points used to represent a cluster would increase as the complexity of its shape increased. In 3D however, a complete plot of all cluster borders would generate a highly cluttered plot with multiple borders occluding each other. In this Chapter's implementation, the clusters are stored as a tree of member voxels. The clustering result on the flow field, however, is expressed by the output streamlines as described in section 6.3.3 below.

Finally, *cluster validity analysis* is the assessment of a clustering procedure's output [93, 132, 94, 95]. Often this analysis uses a specific criterion of optimality; however, these criteria are usually arrived at subjectively. Hence, little in the way of "gold standards" exist in clustering except in well-prescribed subdomains. Validity assessments are objective [62, 114] and are performed to determine whether the output is meaningful. A clustering structure is valid if it cannot reasonably have occurred by chance or as an artifact of a clustering algorithm. There are three types of validation studies. An *external* assessment of validity compares the recovered structure to an *a priori* structure.

An *internal* examination of validity tries to determine if the structure is intrinsically appropriate for the data. A *relative* test compares two structures and measures their relative merit. Further in this Chapter, results are shown of slow streamlines selected using the clustering output as a way of visual internal validation. A quantitative internal examination has been performed for the 2D version of this Chapter's application, by measuring the error rate of automatically detecting vortices from clustered data [178]. That assessment, however, does not isolate out the error component generated by the limitations of the vortex detection algorithm, separately from the clustering step.

There is no clustering technique that is universally applicable in uncovering the variety of structures present in multidimensional data sets. Not all clustering techniques can uncover all the clusters present in any given data set with equal facility, because clustering algorithms often contain assumptions about cluster shape or multiple-cluster configurations based on the similarity measures and grouping criteria used. Humans perform competitively with automatic clustering procedures in 2D, but most real problems involve clustering in higher dimensions. It is difficult for humans to obtain an intuitive interpretation of data embedded in a high-dimensional space. In addition, data hardly follows "ideal" structures such as hyperspherical or linear. This explains the large number of clustering algorithms that continue to appear in the literature.

Even though there is an increasing interest in the use of clustering methods in pattern recognition, image processing and information retrieval, clustering has been popular in other disciplines such as biology, psychiatry, psychology, archaeology, geology, geography and marketing. A number of books on clustering have been published [114, 62, 72, 4, 116] in addition to useful review papers [63, 142, 113, 115].

6.3 Materials and Methods

6.3.1 Clustering Implementation

In recent years, several methods based on hierarchical clustering have been proposed for vector field abstraction. They can generally be divided into two categories: a top down strategy and a bottom up strategy. In both cases, a hierarchical tree is generated to represent the vector field. Heckel et al. [98] used a top down approach where a discrete vector field is segmented into a disjoint set of clusters, whereas Telea et al. [234] used a bottom up clustering approach by recursively merging similar clusters to form larger cluster. Lodha et al. [149] adopted the same clustering approach as Telea et al., but with a different error metric to preserve topology of the vector field. More recently, Garcke et al. [85] have proposed a continuous clustering method for simplifying vector fields. This method is based on the Cahn-Hilliard model that describes phase separation and coarsening in binary alloys. The essence of the technique is similar to the other bottom-up methods in that it builds a multi-scale vector field representation by merging neighbouring cells, but it was done in a continuous manner. Generally speaking, all these methods use a single vector to represent a cluster and are well suited for visualisation purposes, however, given a set of such clusters, it is difficult to recover the original vector field.

For the purpose of providing a better ground for vortices detection, we used local linear expansion for representing a cluster of 2D vectors, with each plane representing a velocity component. By using this representation, the neighbourhood of a critical point can be compactly described. The advantage of using this representation is that given a cluster, the approximation of the original vectors can be easily deduced from the planes. We show that this approach greatly reduces the amount of information needed to describe complex flow fields while preserving all the important critical points intrinsically. Comparison to the traditional approach of vortex detection with winding index is also provided [80, 274], and we demonstrate the much-improved sensitivity provided by the proposed new method.

A hierarchical clustering algorithm merges flow vectors in an iterative process. The fitting error, $E(C)$, for each cluster C is defined as the total square distance between the original data points and the those fitted by the gradients. The cost M_C of merging two clusters, C_1 and C_2, to form a new cluster C_{new} is:

$$M_C(C_1, C_2) = E(C_{new}) - [E(C_1) + E(C_2)] \quad . \tag{6.2}$$

Initially, each vector forms its own cluster and the neighbouring vectors are used to approximate the local linear expansion of this cluster. Subsequently, the associated cost of merging a pair of clusters is calculated and stored in a pool for each pair of neighbouring clusters. The following steps are then repeated until all clusters are merged to form one single cluster enclosing the entire flow field. First, the pair of clusters with the smallest merging cost in the pool is removed and merged to form a new cluster. Then, the cost of merging the latter with its neighbours is calculated and inserted into the pool. By repeatedly merging clusters, a hierarchical binary tree is constructed in the process, with each node representing a cluster and its children representing its sub-clusters as shown in Figure 6.10. Once the hierarchical tree is constructed, abstract flow fields at various clustering levels can then be obtained from this tree efficiently.

6.3.2 Local Linear Expansion

The flow field near critical points is assumed here to be linear. In order to enclose regions around critical points, each cluster joins points with approximately the same velocity gradients. Local linear expansion is carried out to determine these gradients inside each cluster.

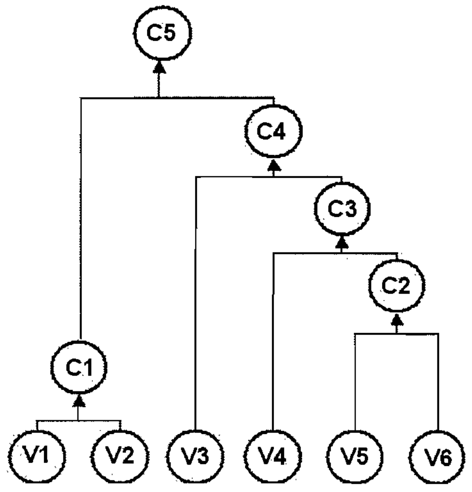


FIGURE 6.10: Example of a hierarchical cluster tree with nodes Cn built from six initial vectors Vn.

We assume that the velocity in a region, $\mathcal{R}$, is given by

$$\begin{aligned} \boldsymbol{v} &= \mathbf{A}(\boldsymbol{x} - \boldsymbol{x}_0) \\ &= \mathbf{A}\,\boldsymbol{x} - \boldsymbol{v}_0 \end{aligned} \tag{6.3}$$

where $\boldsymbol{v}_0 = \mathbf{A}\,\boldsymbol{x}_0$.

We assume that we known $\boldsymbol{v}$ at a set of lattice points. We therefore perform a least squares fitting over the lattice points in the region $\mathcal{R} = \left\{\boldsymbol{x}(i)\right\}_{i=1}^{N}$. The size of region depends on the scale on which you believe linearity is a reasonable approximation (e.g. two vortex cores should not be in the same region) and on the amount of noise in the data (the noisier the data, the larger the region should be). Least squares is equivalent to minimising the energy

$$E = \frac{1}{N}\sum_{i=1} \left\| \boldsymbol{v}(i) - \mathbf{A}\boldsymbol{x}(i) + \boldsymbol{v}_0 \right\|^2 .$$

where $\boldsymbol{v}(i) = \boldsymbol{v}(\boldsymbol{x}(i))$ (i.e. the value of the velocity at the lattice point $\boldsymbol{x}(i)$). Optimising with respect to $\boldsymbol{v}_0$

$$\frac{\partial E}{\partial \boldsymbol{v}_0} = \frac{1}{N}\sum_{i=0}^{N} 2(\boldsymbol{v}(i) - \mathbf{A}\,\boldsymbol{x}(i)) + 2\boldsymbol{v}_0 = 0$$

or

$$\boldsymbol{v}_0 = \mathbf{A}\left\langle \boldsymbol{x} \right\rangle - \left\langle \boldsymbol{v} \right\rangle$$

where $\left\langle \cdots \right\rangle$ denotes the average of the lattice points in the region.

Substituting the optimal value of $\boldsymbol{v}_0$ back into the energy we get

$$\begin{aligned} E &= \frac{1}{N}\sum_{i=1} \left\| \boldsymbol{v}(i) - \left\langle \boldsymbol{v} \right\rangle - \mathbf{A}\left(\boldsymbol{x}(i) - \left\langle \boldsymbol{x} \right\rangle\right) \right\|^2 \\ &= \frac{1}{N}\sum_{i=1} \Big(\left\| \boldsymbol{v}(i) - \left\langle \boldsymbol{v} \right\rangle \right\|^2 - 2\left(\boldsymbol{v}(i) - \left\langle \boldsymbol{v} \right\rangle\right)^{\mathsf{T}} \mathbf{A}\left(\boldsymbol{x}(i) - \left\langle \boldsymbol{x} \right\rangle\right) \\ &\quad + \left(\boldsymbol{x}(i) - \left\langle \boldsymbol{x} \right\rangle\right)^{\mathsf{T}} \mathbf{A}^{\mathsf{T}}\mathbf{A}\left(\boldsymbol{x}(i) - \left\langle \boldsymbol{x} \right\rangle\right) \Big) \end{aligned}$$

The matrix $\mathbf{A}$ that minimises E is given by:

$$\mathbf{A} = \mathbf{W}^{\mathsf{T}}\mathbf{V}^{-1} \tag{6.4}$$

where

$$\begin{aligned} \mathbf{W} &= \frac{1}{N}\sum_{i=1} \left(\boldsymbol{x}(i) - \left\langle \boldsymbol{x} \right\rangle\right)\left(\boldsymbol{v}(i) - \left\langle \boldsymbol{v} \right\rangle\right)^{\mathsf{T}} \\ \mathbf{V} &= \frac{1}{N}\sum_{i=1} \left(\boldsymbol{x}(i) - \left\langle \boldsymbol{x} \right\rangle\right)\left(\boldsymbol{x}(i) - \left\langle \boldsymbol{x} \right\rangle\right)^{\mathsf{T}} \end{aligned}$$

To be able to reconstruct the flow field, we store $\boldsymbol{v}_0$ and the contents of $\mathbf{A}$ for each cluster. Retrieving the field's velocities then simply involves applying equation (6.3).

6.3.3 Topology Display

In this work, streamlines were chosen to display the flow. These are generated in the same way as steady flow streamlines (see section 3.4.1 on page 35 for implementation details), but they must be interpreted as transient in time as they do not result from steady flow. Used in conjunction with clustering, however, they can be used to convey the overall topology of the field.

After clusters have been formed from the flow field, streamlines are grown from equally spaced points throughout the image and stored in a streamline array. Each streamline passes through one or several clusters, and this is recorded in a list. A streamline "correlation" matrix $\mathbf{C}_{cluster}$ is then built by computing the ratio between common clusters occupied by streamlines and the total number of clusters spanned by each streamline:

$$\mathbf{C}_{cluster}(e, v) = \frac{\tau_{cluster}(e, v)}{\gamma_{cluster}(e)} \tag{6.5}$$

where e, v are streamline array indices, $\tau_{cluster}(e, v)$ is the number of clusters occupied by both e and v and $\gamma_{cluster}(e)$ is the total number of clusters occupied by streamline e.

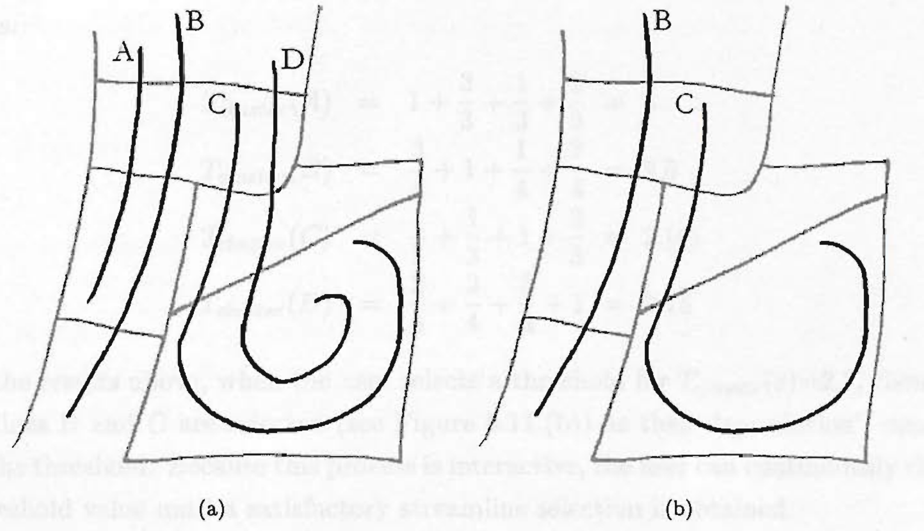


FIGURE 6.11: Streamlines (black lines) and cluster borders (grey lines) for a fictitious flow, with (a) all streamlines shown and (b) simplified flow with only the most topologically significant streamlines selected.

The most representative streamlines can then be selected by setting a maximum streamline correlation $T_{cluster}(e)$, the value of which is interactively chosen by the user. This is defined as:

$$T_{cluster}(e) = \sum_{\substack{v=1 \\ v\neq}}^{N} \mathrm{C}_{cluster}(e, v) \tag{6.6}$$

where N is the total number of streamlines in the streamline array.

To illustrate these measures, Figure 6.11.(a) shows ficticious streamlines passing through clusters. The streamlines are labelled A, B, C and D. Streamline A passes through 3 clusters, B through 4 clusters, C through 3 clusters and D through 4 clusters. Streamline A and B share 3 clusters, C shares 1 cluster with A and B and 3 clusters with D. D shares 2 clusters with A and B. The "correlation" matrix for this set of streamlines and clusters then becomes:

$$\mathrm{C}_{cluster}(e, v) = \begin{array}{c c} & \begin{array}{cccc} \mathrm{A} & \mathrm{B} & \mathrm{C} & \mathrm{D} \end{array} \\ \begin{array}{c} \mathrm{A} \\ \mathrm{B} \\ \mathrm{C} \\ \mathrm{D} \end{array} & \begin{pmatrix} 1 & \frac{3}{4} & \frac{1}{3} & \frac{2}{4} \\ \frac{3}{3} & 1 & \frac{1}{3} & \frac{2}{4} \\ \frac{1}{3} & \frac{1}{4} & 1 & \frac{3}{4} \\ \frac{2}{3} & \frac{2}{4} & \frac{3}{3} & 1 \end{pmatrix} \end{array}$$

For each streamline, the "correlation" sum $T_{cluster}(e)$ is then computed by summing the columns:

$$\begin{aligned}
T_{cluster}(A) &= 1+\frac{3}{3}+\frac{1}{3}+\frac{2}{3} = 3 \\
T_{cluster}(B) &= \frac{3}{4}+1+\frac{1}{4}+\frac{2}{4} = 2.5 \\
T_{cluster}(C) &= \frac{1}{3}+\frac{1}{3}+1+\frac{3}{3} = 2.(6) \\
T_{cluster}(D) &= \frac{2}{4}+\frac{2}{4}+\frac{3}{4}+1 = 2.75
\end{aligned}$$

Given the results above, when the user selects a threshold for $T_{cluster}(e)$=2.7, then only streamlines B and C are selected (see Figure 6.11.(b)) as their "correlation" sums are below the threshold. Because this process is interactive, the user can continuously change the threshold value until a satisfactory streamline selection is obtained.

6.3.4 Flow Field Abstraction and Vortical Feature Detection

Our proposed clustering algorithm generates an abstract flow field by iteratively merging similar clusters of flow vectors. The crucial factor to a good clustering algorithm is the choice of the representation for a cluster of vectors. In our study, locally linear expansion was used and two planes are used to represent a cluster of 2D vectors, one for each velocity component. The planes are fitted automatically using the least square fitting method and the cost of merging two clusters is simply defined as the second moment of the points in the merged cluster.

The choice of using planes to represent a flow field is particularly suitable for regions near critical points, thus, this abstraction technique intrinsically preserves critical points and provides a good foundation for identification of critical points. The first step of the clustering algorithm is to divide the image domain into clusters of 2×2 vectors. Each cluster is represented by two planes which can be generated using the least square fitting method. Then, for each neighbouring cluster pair, the associated cost of merging the pair is calculated and stored in a pool. After the initialisation, the following two steps process is repeated until the pool is empty: i) the pair of clusters with smallest merging cost in the pool is selected and merged to form a new cluster , and ii) the cost of merging the newly merged cluster with its neighbours is calculated and inserted into the pool.

The detection of critical points in vector fields require the evaluation of the Jacobian matrix at positions of zero velocity. The traditionally adopted method of winding index is sensitive to image noise. With the proposed flow field abstraction method, this problem is naturally avoided as the Jacobian matrix can be readily retrieved from the velocity planes. After the critical points are extracted, they can be classified by using the phase portrait theory [15] to separate them into attracting/repelling focus, attracting/repelling node, planar vortex, or saddle by solving for the eigenvalues of the Jacobian matrix. For vortices, the associated eigenvalues will have large imaginary values.

6.4 Results

The proposed method was applied to 3D flow through the human heart simulated from a CFD model [153]. This is briefly described here for completeness. The model after mesh processing contained 54,230 nodes and 41,000 cells. A total of 16 meshes representing the LV across the complete cardiac cycle were generated from the original image data. In order to permit CFD simulation, it was necessary to increase the temporal resolution of the model. This ensured that none of the constituent cells underwent excessive deformation or displacement between adjacent time steps. To this end, cubic spline interpolations were performed to generate a total of 49 meshes across the cardiac cycle. The Navier-Stokes equations for 3D time-dependent laminar flow with prescribed wall motion was solved using a finite-volume based CFD solver - CFX4 (CFX international, AEA technology, Harwell). The blood was treated as an incompressible Newtonian fluid with a constant viscosity of 0.004 Kg/(ms). The simulation was started from the beginning of systole with the pressure of the aortic valve plane set to zero and with the mitral valve plane treated as a non-slip wall. At the onset of diastole, the aortic valve was closed by treating it as a solid wall, whilst the mitral valve was opened by using a combination of pressure and flow boundaries.

As can be seen in Figure 6.12, the choice of the maximum total correlation is important for the rendering result. If that value is too low, this results in too few streamlines being selected and flow features being missed (Figures 6.12(c,d)); if the value is too high, the display is cluttered by excessive streamlines (Figure 6.12(a)). This visualisation method was also applied along the time series data of the same dataset. The previous results depict the flow during diastole showing flow through the mitral valve, as shown in Figure 6.13. The proposed method was also applied to 2D *in vivo* MR velocity data acquired with sequential examination following myocardial infarction using a Marconi whole body MR scanner operating at 1.5T. Cine phase contrast velocity mapping was performed using a FEER sequence with a TE of 14ms. The slice thickness was 10mm and the field of view was 30-40cm. The dataset has a temporal resolution of 45ms and the diastolic phase is covered in about 5-10 frames, five of which are shown here. The dataset was denoised using the restoration scheme described in [177]. Figure 6.14 shows a comparison between our automatically selected streamlines and arrow plots overlayed on the corresponding conventional MR images. In order to quantify the error introduced by the clustering process we measured the root-mean-square difference between the original velocity field and that reconstructed by the cluster gradients. The results are plotted in Figure 6.15 for clustering of the 12th frame (540ms) of the 2D *in vivo* dataset and time sample 12 of the 3D simulated dataset. The number of clusters used to produce all

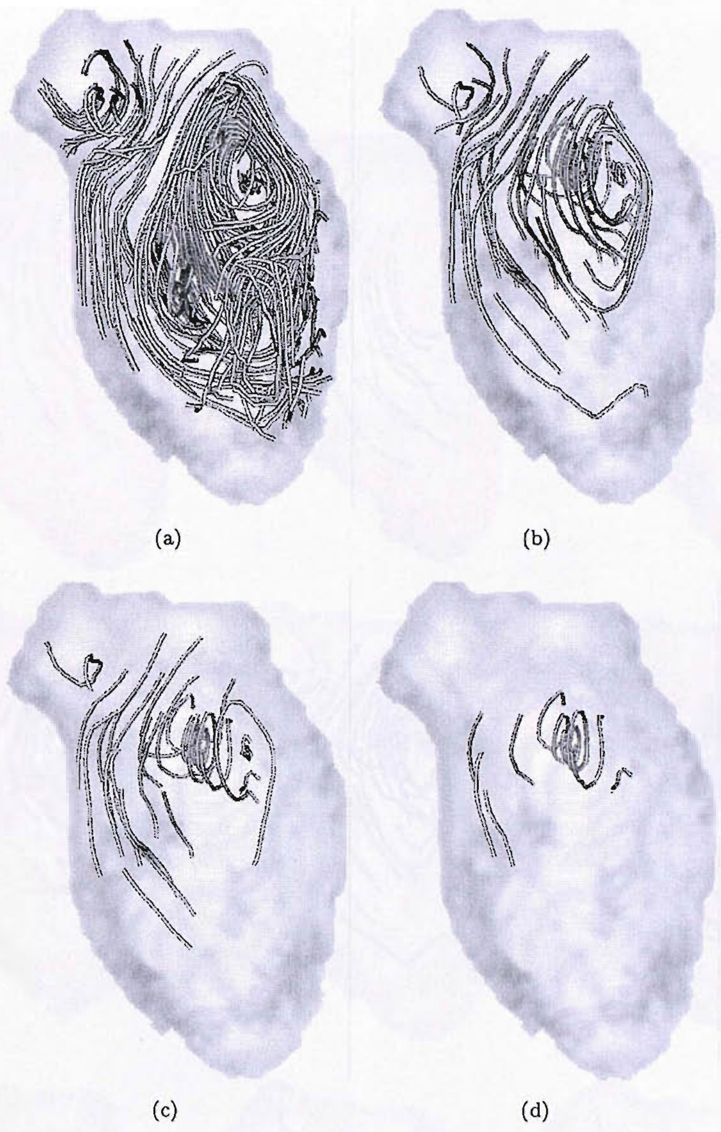


FIGURE 6.12: Effect of interactively changing the maximum total correlation threshold. (a) No threshold - all streamlines selected. (b) $T_{cluster}(e)_{MAX} = 20$; (c) $T_{cluster}(e)_{MAX} = 10$; (d) $T_{cluster}(e)_{MAX} = 5$, the smaller vortex is now invisible.

streamline plots in this work was 10 for both datasets. The size of the region of interest was 3536 pixels and 47250 voxels for the 2D and 3D datasets, respectively.

6.5 Discussion

A novel method is presented for velocity MR flow simplification and display. Flow simplification is achieved by clustering, where each cluster contains points with similar local linear approximation gradients. This ensures that each cluster encloses points sharing relevant flow features. Streamlines are then generated over the flow field and

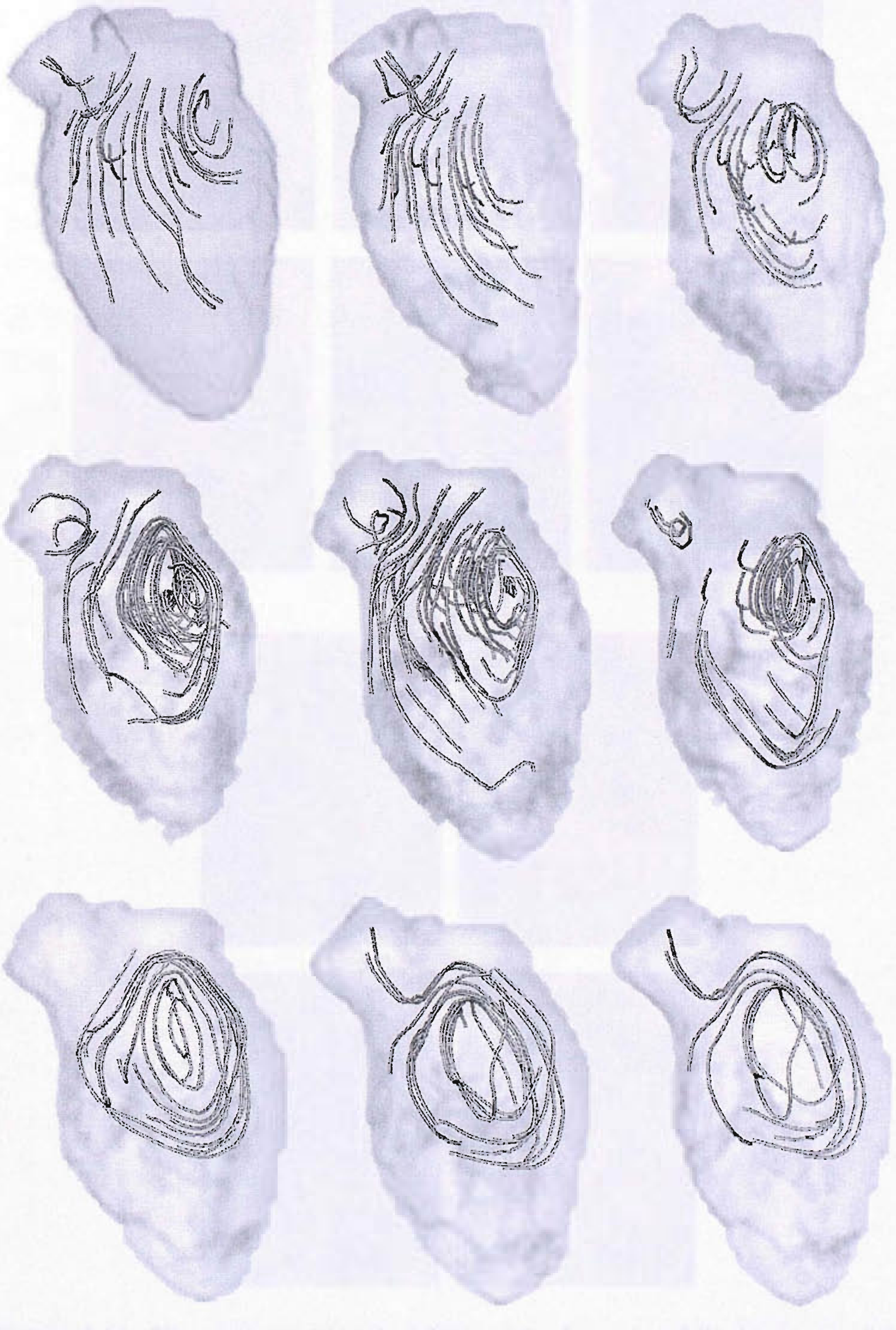


FIGURE 6.13: Simplified streamline rendering of 3D simulated flow inside the human cardiac left ventricle. Time samples 8 to 16 of a set of 33, ordered left to right, row by row.

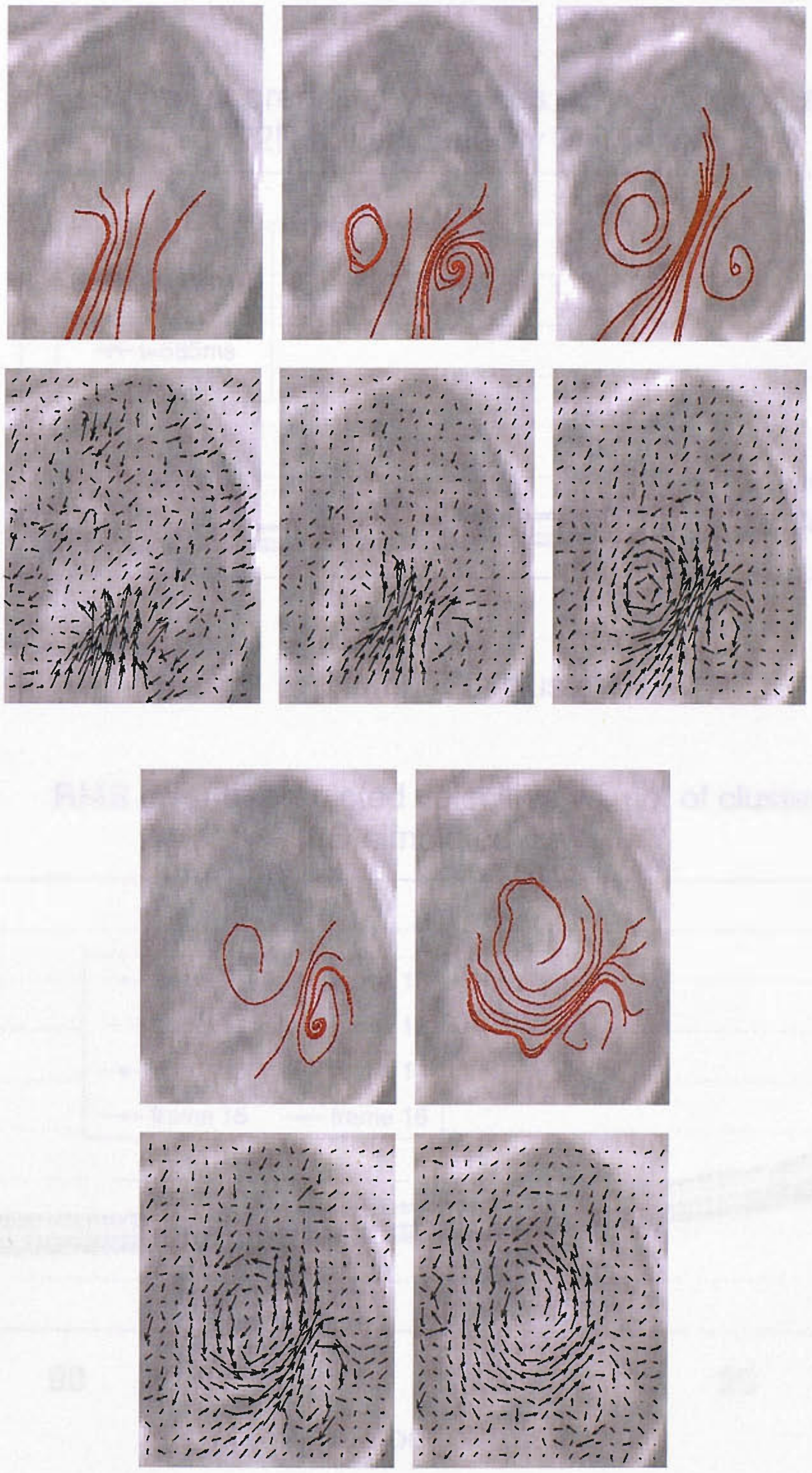
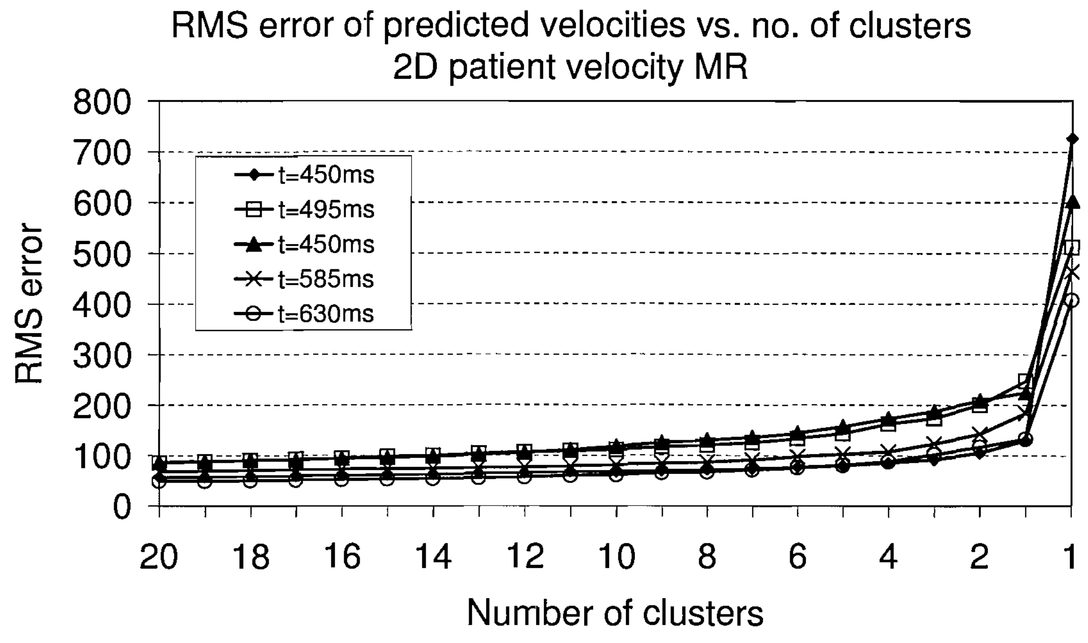


FIGURE 6.14: Flow pattern within the left ventricle of a patient following myocardial infarction, from 450ms to 630ms after onset of ECG R wave. Vortical flow rendered by streamlines selected by clustering are shown along with the corresponding 2D arrow plots, where arrow length is proportional to velocity magnitude. A 3-spaced grid was used to select the arrow data to be displayed.



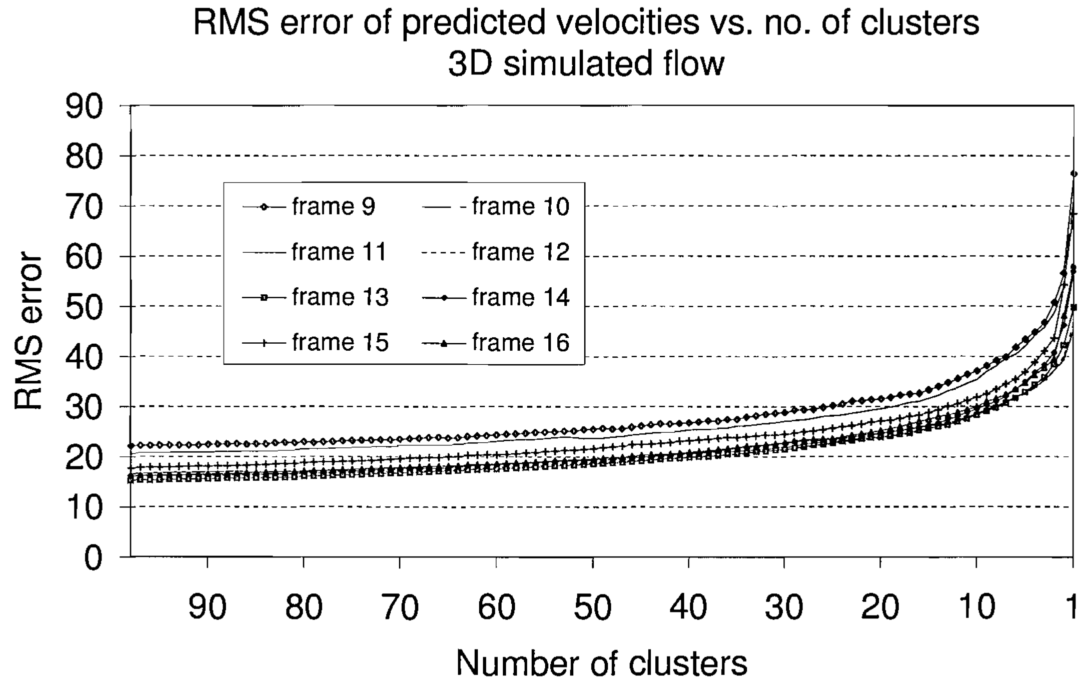


FIGURE 6.15: Root mean square error between original and cluster velocity predicted from gradients (see text) for the 2D frames of Figure 6.14 (left) and 3D simulated time samples of Figure 6.13 (right).

selected using a measure of inter-streamline cluster overlap, interactively thresholded. The rendered streamlines depict the important features of the flow and present the observer with an overview of the general flow topology of the field.

Chapter 7

Conclusions and Future Work

This work has focused on the pre-processing step of medical imaging data. As described in Chapter 1, this is the first module in most image processing and analysis systems, and is followed by as well as facilitates the execution of visualisation, manipulation and analysis. In previous Chapters, issues were addressed in the areas of 3D ultrasound segmentation, cardiac MRI segmentation, and 3D cardiac MRI restoration and flow feature analysis. The following sections synthesise the work carried out, including experiments and their findings. The final section suggests possible extensions to this work.

7.1 Segmentation for 3D Ultrasound

A current problem in 3D ultrasound imaging, the detection of speckle in the data, was addressed as a segmentation problem. The approach taken was to provide a comparison of traditional image processing approaches, analyse the results and propose improvements. Appropriate image feature extraction algorithms were selected and compared using a learning from data framework for training and testing the algorithms. The features used were the co-occurrence matrix, signal-to-noise ratio (SNR) and the autocorrelation function. These were selected as they are traditionally applied in the literature for texture recognition. A fourth algorithm, the homodyned-k distribution for speckle detection, was also implemented for comparison.

In the comparison experiment, it was found that the feature values of co-occurrence and SNR of speckle overlapped significantly with those computed over purely noise regions. Feature values inside speckle-only areas also varied over a large range. Second-order features (that explore the relationship between two pixels) performed better than first-order (single pixel-based) ones. The homodyned-k algorithm performed worst due to the large region of interest necessary, and its sensitivity to a decompression parameter that had to be pre-set. Combining several features using the Mahalanobis distance did not significantly improve the success rate compared to the features individually.

A novel method was proposed, called specklet detection. This measured the occurrence and brightness of peaks in the image, and was based on previously published studies of the speckle structure in images. This algorithm produced a small but significant improvement in detection rate compared to using second order features.

The original contributions of Chapter 2 were thus the first comparison study on speckle detection techniques, and a novel method for detecting speckle.

7.2 Segmentation for Cardiac MRI

As described in Chapter 1, magnetic resonance imaging (MRI) data are acquired by exposing the subject to sequences of harmless magentic pulses. If cardiac cine (movie) images are desired, the pulses can be synchronised with the heartbeat, a technique called electrocardiogram (ECG) gating. Several cross-sectional slices of the subject can be acquired into a 3D volume of data. In order for clinical diagnostic measures such as elasticity, extensibility and others to be taken into consideration, the left ventricle (LV) must be found and its borders and other features of interest delineated. Several techniques exist for border delineation, such as edge detection, active contours and livewire.

Chapter 4 focused on the less well developed area of finding the LV, a problem in image object detection. Previous approaches have included a pixel intensity profile matching method, which was limited to a cross-shape matching template, and a Hough-based method that was limited to the detected edges in the image. In this thesis, a novel template is proposed that covers the whole LV. The template consists of two concentric elliptic grey level regions, an inner ellipse extending towards the endocardial border superposed over an outer ellipse that extends towards the epicardial border. Powell's method of optimisation is applied to fit the template to MRI images, placing the template at each image location and optimising it locally to obtain a fitness error measure. The location with the least fitness error is then chosen as the correct match. The algorithm was entirely successful on some ideal cases, but failed in 7 out of 23 test images. The 2 main reasons for failure were: the presence of irregular borders in the image, that did not follow the canonical elliptic shape; and the presence of the aorta, having a very regular elliptical shape that attracted the template towards it. The original contributions of this section were thus the proposal of a novel method for LV detection and the analysis of its performance.

7.3 Noise Reduction of 3D Velocity MRI

Hardware setup and patient movement generate noise that corrupts blood flow velocity images. It is necessary to reduce this noise in order to improve the accuracy of subsequent data processing stages.

A restoration algorithm was developed in Chapter 5 that combines Total Variation denoising with a First Order optimisation, with automatic setting of a restoration parameter that has been shown to significantly affect the result when fixed manually. Application of the algorithm resulted in a clear improvement of noisy images. Synthetic images were used to measure the root-mean-square error and it was found that the proposed method converged successfully to the minimum error. Improvement in in-vivo images could not be measured objectively, but there was a significant visual improvement.

All images in the subsequent Chapters were denoised using this restoration method.

7.4 Feature Abstraction for 3D Velocity MRI

3D Velocity MRI, as well as 3D CFD heart simulations, produce large amounts of data that are difficult to analyse and interpret. It is known that vortical flow exists in the human left ventricle; from the limited amount of studies, evidence suggests there exists a vortex ring that is larger and lasts for longer in post-infarction patients than healthy subjects. In order to enable further study of this phenomenon, Chapter 6 proposed a novel method to visualise LV flow, especially indicated for visualising vortices.

The method is part of a larger framework for providing Cardiac Velocity MRI analysis. The framework consists of three steps: restoration, abstraction and tracking. Restoration, already referred to above, is essential for clarity of visualisation and accuracy of measurements. Abstraction detects and identifies important features of the flow. Tracking is necessary for time series data, to ensure continuity of the same detected features from one time frame to the next, and is outside the scope of this work.

The novel method developed in this thesis implements data compression and simplification. This is achieved by applying clustering to the data. Linearity is assumed around critical points. Local linear expansion is thus used to compute local gradients at points inside the clusters, and these gradients are used as the homogeneity criterion for cluster growth. An iterative cluster merging algorithm is used, resulting in a hierarchical tree with a single cluster at the top. The velocity data can then be reconstructed from the cluster parameters and the linear fitting for each cluster. This can be done at any level of the tree, resulting in velocity reconstructions with varying degrees of compression. Higher compression eliminates more flow features but generates a simpler reconstruction, and vice versa.

The second main contribution of Chapter 6 is automatic streamline selection to display the most relevant flow features in uncluttered plots. This is done using a streamline "cross-correlation" measure, computed over the clusters each streamline occupies. The streamlines are rendered in 3D and their coordinates can be exported for further processing. The resulting visualisation system was tested successfully on restored 3D Heart Velocity CFD.

7.5 Preliminary Approaches for Further Work

Watersheds are a popular segmentation technique [73], used when there exist regular patterns of similar shapes. For example, in cell microscopy image problems, watersheds are often used to detect and delineate each cell in the image. This makes possible the computation of statistical measures about cell size, symmetry, orientation, and others. A preliminary study was conducted to apply watersheds to speckle detection. As can be observed from Figure 7.1, the watersheds were successful in isolating each speckle structural unit, referred to as "specklet" in Chapter 2. Peak and trough detection was then used to find the peaks in each specklet. If appropriate statistical measures and an analysis algorithm are designed, it is conceivable that enough features could be computed for successful identification of speckle watersheds, and their differentiation from the non-speckle ones.

Figure 7.2 shows frames of a dye injection video, forming a vortex ring before reaching a wall. These images were captured from real flow in a fluid laboratory setting. It would be useful to achieve a similar-looking image, computed from the flow vector data acquired inside the LV. Such a synthetic image would require determining the shear planes in the flow, however such a computation for flow that is not planar is non-trivial.

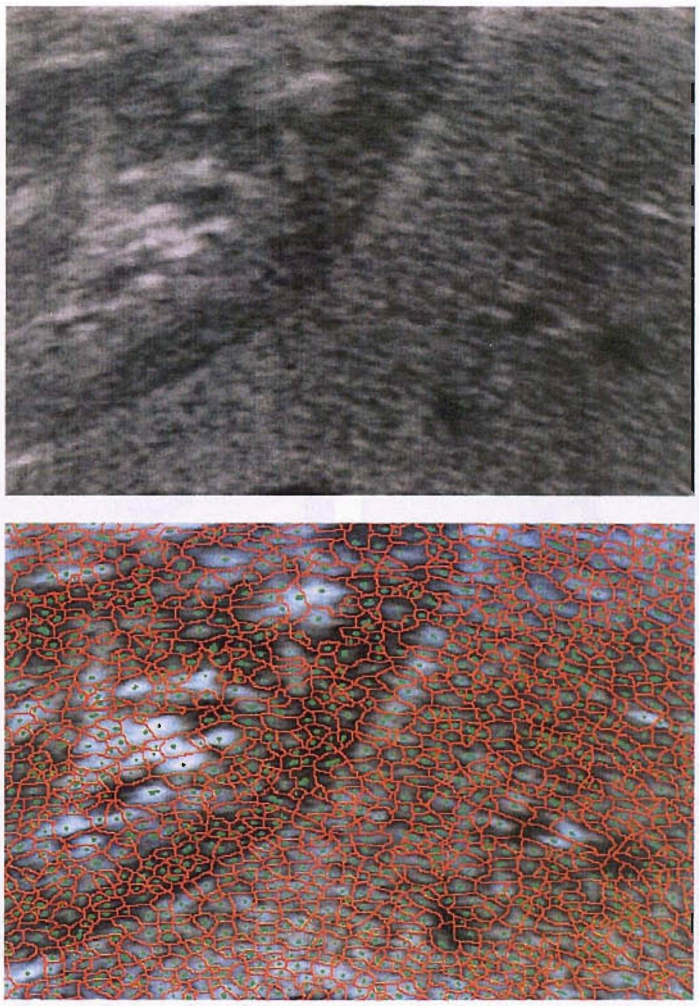


FIGURE 7.1: Top image showing an ultrasound image of the kidney and liver. Below, the same image to which watershed (red borders) and peak detection (green dots) were applied.

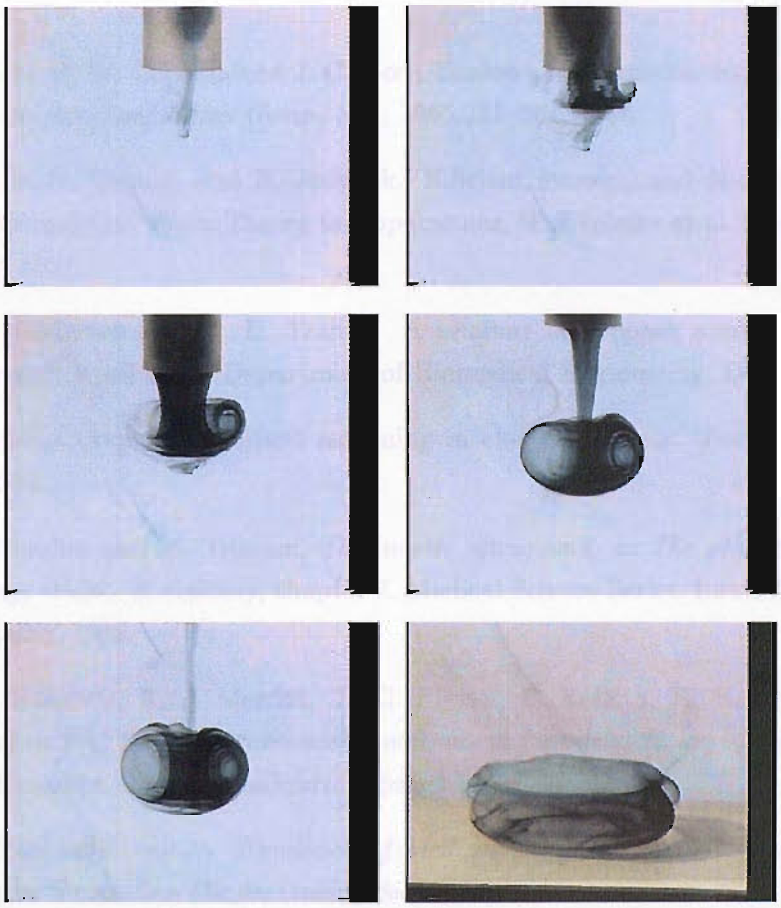


FIGURE 7.2: Frames from a fluid laboratory movie, showing the path and evolution of an injected dye through a clear fluid. The dye forms a vortex ring, and progresses until reaching a wall. Courtesy G. Z. Yang, Department of Computing, Imperial College.

Bibliography

[1] A. Amini, R. W. Corwen, and J. C. Gore. Snakes and splines for tracking non-rigid heart motion. *Lec. Notes Comp. Sci.*, 1065:251–261, 1996.

[2] Y. Amit, D. Geman, and B. Jedynak. Efficient focusing and face detection. *in Face Recognition: From Theory to Applications*, H. Wechsler et al. (eds.), Springer Verlag, 1997.

[3] M. E. Anderson and G. E. Trahey. A seminar on k-space applied to medical ultrasound, April 2000. Department of Biomedical Engineering, Duke University.

[4] E. Backer. Computer-assisted reasoning in cluster analysis. *Prentice-Hall Int., UK*, 1995.

[5] J. C. Bamber and M. Tristam. *Diagnostic ultrasound, in The physics of medical imaging, Webb., S. (editor)*, chapter 7. Medical Science Series. Institute of Physics Publishing, 1988.

[6] G. V. Bancroft, F. J. Merritt, T. C. Plessel, P. Kelaita, R. K. McKabe, and A. Globus. FAST: a multiprocessed environment for visualization of computational fluid dynamics. *Proc. Visualization*, pages 14–27, 1990.

[7] I. N. Bankman, editor. *Handbook of Medical Imaging Processing and Analysis.* Academic Press, San Diego, California, 2000.

[8] D. C. Banks and B. A. Singer. A predictor-corrector technique for visualizing unsteady flow. *IEEE T. Vis. Comp. Graph.*, 1(2):151–163, 1995.

[9] W. A. Barrett and E. N. Mortensen. Interactive live-wire boundary extraction. *Medical Image Analysis*, 1(4):331–341, 1996.

[10] G. R. Bashford and O. T. von Ramm. Speckle structure in three dimensions. *Journal of the Acoustical Society of America*, 98(1):35–42, 1995.

[11] G. R. Bashford and O. T. von Ramm. Ultrasound three-dimensional velocity measurements by feature tracking. *IEEE Transactions on Ultrasonics, Ferroelectrics, and Frequency Control*, 43(3):376–385, 1996.

[12] O. Basset, Z. Sun, J. L. Mestas, and G Gimenez. Texture analysis of ultrasonic images of the prostrate by means of co-occurrence matrices. *Ultrasonic Imaging*, 15:218–237, 1993.

[13] B. G Becker, D. A. Lane, and N. L. Max. Unsteady flow volumes. *Proc. Visualizations.*, pages 19–24, 1995.

[14] J. Beis and D. Lowe. Indexing without invariants in 3D object recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 21(10):1000–1015, 1999.

[15] S. Ben-Yacoob. Fast object detection using MLP and FFT. *IDIAP Research Report 97-11, Switzerland*, 1997.

[16] M. O. Berger. Snake growing. *ECCV'90, in Lecture Notes in Computer Science*, pages 570–572, 1990.

[17] D. P. Bertsekas. *Nonlinear Programming.* Athena Scientific, Belmont, Massachussets, 1995.

[18] B. Bhanu and T. Jones. Image understanding research for automatic target recognition. *IEEE AES Sys Mag*, pages 15–22, 1993.

[19] C. M. Bishop. *Neural Networks for Pattern Recognition.* Clarendon Press, 1995.

[20] A. Blake and M. Isard. Active Contours. Springer-Verlag, 1997.

[21] A. Blake and A. Zisserman. Visual reconstruction. *MIT Press, Cambridge, Mass.*, 1987.

[22] E. Boring and A. Pang. Directional flow visualization of vector fields. *Proc. Visualization*, pages 389–392, 1996.

[23] H. Bot, J. Verburg, B. J. Delemarre, and J. Strackee. Determinants of the occurrence of vortex rings in the left ventricle during diastole. *J. Biomechanics*, 23:607–615, 1990.

[24] A. E. O. Boudraa. Automated detection of the left ventricular region in magnetic resonance images by fuzzy c-means model. *Int. J. Card. Imag.*, 13(4):347–355, 1997.

[25] D. Boukerroui, Basset O., Baskurt A., and G. Gimenez. A multiparametric and multiresolution segmentation algorithm of 3-D ultrasonic data. *IEEE Transactions on Ultrasonics, Ferroelectrics, and Frequency Control*, 48(1):64–77, 2001.

[26] E. Brandt, T. Ebbers, L. Wigström, J. Engvall, and M. Karlsson. Automatic detection of vortical flow patterns from three-dimensional phase contrast mri. *Proc. Intl. Soc. Mag. Reson. Med.*, page 1838, 2001.

[27] M. Brill, H. Hagen, H.-C. Rodrian, W. Djatschin, and S. V. Klimenko. Streamball techniques for flow visualization. *Proc. Visualization.*, pages 225–231, 1994.

[28] C. B. Burckhardt. Speckle in ultrasound B-mode scans. *IEEE Transactions on Sonics and Ultrasonics*, SU-25(1):1–6, January 1978.

[29] B. Cabral and C. Leedom. Imaging vector fields using line integral convolution. *Proc. SIGGRAPH '93*, pages 263–270, 1993.

[30] B. Cabral and C. Leedom. Highly parallel vector visual environments using line integral convolution. *Proc. 7th SIAM Conf. Parallel Proc. Sci. Comp.*, pages 802–807, 1995.

[31] W. Cai and P. A. Heng. Principal stream surfaces. *Proc. Visualization.*, pages 75–80, 1997.

[32] A. Califano and R. Mohan. Multidimensional indexing for recognizing visual shapes. *IEEE Trans. Patt. Anal. Mach. Intel.*, 16(4):373–392, 1994.

[33] J. Canny. A computational approach to edge detection. *IEEE Trans. Patt. Anal. Mach. Intel.*, 8(6):679–698, 1986.

[34] V. Chalana, D. T. Linker, D. R. Haynor, and Y. Kim. A multiple active contour model for cardiac boundary detection on echocardiographic sequences. *IEEE Transactions on Medical Imaging*, 15(3):290–298, 1996.

[35] T. Chan, S. H. Kang, and J. Shen. Total variation denoising and enhancement of color images based on the cb and hsv color representation. *J. Visual Comm. Image Rep.*, 12(4), 2001.

[36] T. Chan and J. Shen. Variational restoration of non-flat image features: models and algorithms. *IAM Journal on Applied Mathematics*, 61(4):1338–1361, 2000.

[37] J-F. Chen, J. B. Fowlkes, P. L. Carson, and J. M. Rubin. Determination of scan-plane motion using speckle decorrelation: theoretical considerations and initial test. *International Journal of Imaging Systems Technology*, 8:38–44, 1997.

[38] J-F. Chen and Weng L. System and method for 3-D ultrasound imaging and motion estimation., March 1999. United States patent 5,876,342. Application number 884,708.

[39] J. Cho and P. J. Gemperline. Pattern recognition analysis of near-infrared spectra by robust distance method. *Journal of Chemometrics*, 9:169–178, 1995.

[40] Z. Cho, J. P. Jones, and M. Singh. *Foundations of Medical Imaging.* John Wiley, New York, 3rd edition, 1993.

[41] L. Cohen. On active contour models and balloons. *CVGIP: Image Understanding*, 53(2):211–218, 1991.

[42] A. Colmenarez and T. Huang. face detection with information-based maximum discrimination. *Proc. CVPR*, pages 782–787, 1997.

[43] T. C. Cootes, C. Taylor, C. Cooper, and J. Graham. A trainable method of parametric shape description. *British Mach. Vis. Conf.*, pages 54-61, 1991.

[44] T. F. Cootes, G. J. Edwards, and C. J. Taylor. Active appearance models. *Proc. ECCV*, pages 484–498, 1998.

[45] T. F. Cootes and C. J. Taylor. Combining elastic and statistical models of appearance variation. *Proc. Eur. Conf. Comp. Vis.*, 1:149–163, 2000.

[46] T. F. Cootes, C. J. taylor, D. H. Cooper, and J. Graham. Training models of shape from sets of examples. *Proc. British Mach. Vis. Conf.*, pages 9–18, 1992.

[47] T. F. Cootes, C. J. Taylor, D. H. Cooper, and J. Graham. Active shape models - their training and application. *Computer Vision and Image Understanding*, 61(1):38–59, 1995.

[48] D. Cosgrove, H. Meire, and K. Dewbury, editors. *Clinical ultrasound: a comprehensive text*, volume 1 of *Abdominal and General Ultrasound.* Churchill Livingstone, 1993.

[49] O. Coulon, D. C. Alexander, and S. R. Arridge. Tensor field regularisation for DT-MR images. *Proc. MIUA*, 2001.

[50] R. Curwen and A. Blake. Dynamic contours: real-time active splines. *In Blake, A. and Yuille, A. (eds.) Active Vision*, pages 39–57, MIT Press, 1993.

[51] R. N. Czerwinski, D. L. Jones, and W. D. O'Brien. Line and boundary detection in speckle images. *IEEE Transactions on Image Processing*, 7(12):1700–1713, 1998.

[52] L. S. Davis and A. Rosenfeld. Noise cleaning by iterated local averaging. *IEEE Trans. Sys. Man Cyb.*, 8:705–710, 1978.

[53] W. de Leeuw and R. van Liere. Collapsing flow topology using area metrics. *Proc. IEEE Visualization, D. Ebert, M. Gross, and B. Hamann (eds.)*, pages 349–354, 1999.

[54] W. de Leeuw and R. van Liere. Methods for interactive visualization of large flow datasets. *Visual Data Exploration and Analysis VII, Proc. of SPIE*, 3960:260–267, 2000.

[55] W. de Leeuw and YHP. Visualization of global flow structures using multiple levels of topology. *Groller, E. et al. (eds.), Proc. Joint Eurographics, IEEE TCGV*, 1999.

[56] W. C. de Leeuw, F. H. Post, and R. W. Vaatstra. Visualization of turbulent flow by spot noise. *Proc. Virtual Env. Sci. Vis.*, pages 286–295, 1996.

[57] W. C. de Leeuw and R. van Liere. Enhanced spot noise for vector field visualization. *Proc. Visualization*, pages 233–239, 1995.

[58] B. J. Delemarre, H. Bot, and A. S. Pearlman. Diastolic flow characteristics of severely impaired left ventricles: a pulsed doppler ultrasound study. *J. Clin. Ultrasound*, 15:115–119, 1987.

[59] R. Deriche. Using Canny's criteria to derive a recursively implemented optimal edge detector. *Int. J. Comp. Vis.*, 1:167–187, 1987.

[60] M. Dindar, A. Lemnios, M. S. Shephard, K Jansen, and D. Kenwright. Effect of tip vortex resolution on UH-60A rotor-blade hover performance calculations. *54th Ann. Forum Tech. Displ., Amer. Helicopter Soc.*, 1998.

[61] R. C. Dubes. How many clusters are best? - an experiment. *Pattern Recogn.*, 20(6):645–663, 1987.

[62] R. C. Dubes. Cluster analysis and related issues. *Handbook of Pattern Recognition and Computer Vision, Chen, C. H. et al. eds., World Sci. Pub., NJ*, 1993.

[63] R. C. Dubes and A. K. Jain. Clustering methodology in exploratory data analysis. *Advances in Computers, Yovits. M. C. (ed.), Academic Press, NY*, 1980.

[64] R. O Duda. Pattern recognition for HCI lecture notes, 2001.

[65] R. O. Duda, P. E. Hart, and D. G. Stork. *Pattern Classification.* Wiley Interscience, 2000.

[66] J. Duncan and N. Ayache. Medical image analysis: progress over two decades and the challenges ahead. *IEEE Trans. Pattern Anal. Mach. Intel.*, 22(1):85–106, 2000.

[67] N. Duta, A. K. Jain, and M. P. Jolly. Learning-based object detection in cardiac MR images. *Proc. ICCV'99, Corfu, Greece*, pages 1210–1216, 1999.

[68] V. Dutt and J. F. Greenleaf. Speckle analysis using signal to noise ratios based on fractional order moments. *Ultrasonic Imaging*, 17:251–268, 1995.

[69] V. Dutt and J. F. Greenleaf. Statistics of the log-compressed echo envelope. *Journal of the Acoustical Society of America*, 99(6):3817–3825, June 1996.

[70] D. S. Ebert and P. Rheingans. Volume illustration: non-photorealistic rendering of volume models. *IEEE Trans. Vis. Comp. Graphics*, 7(3):253–264, 2001.

[71] R. R. Edelman, H. P. Mattle, J. Kleefield, and M. S. Silver. Quantification of blood flow with dynamic MR imaging and presaturation bolus tracking. *Radiology*, 171(2):551–556, 1989.

[72] B. S.. Everitt. Cluster analysis. *Edward Arnold, London.*, 1993.

[73] A. X. Falcao, J. Stolfi, and R. de Alencar Lotufo. The image foresting transform: theory, algorithms, and applications. *IEEE T. Patt. Anal. Mach. Intell.*, 26(1):19–29, 2004.

[74] A. X. Falcao and J. K. Udupa. A 3D generalization of user-steered live-wire segmentation. *Medical Image Analysis*, 4:389–402, 2000.

[75] A. X. Falcao, J. K. Udupa, S. Samarasekera, and S. Sharma. User-Steered image segmentation paradigms: Live wire and live lane. *Graphical Models and Image Processing*, 60:233–260, 1998.

[76] A. J. Fenlon and T. David. An integrated visualization and design toolkit for flexible prosthetic valves. *Proc. Visualization.*, pages 453–456, 2000.

[77] D. N. Firmin, G. L. Nayler, P. J. Kilner, and D. B. Longmore. The application of phase shifts in NMR for flow measurement. *Magn. Reson. Med.*, 14(2):230–241, 1990.

[78] S. R. Fleagle, D. R. Thedens, W. Stanford, R. I. Pettigrew, N. Reichek, and D. J. Skorton. Multicenter trial of automated border detection in cardiac MR imaging. *J. Mag. Res. Imag.*, 3(2):409–415, 1993.

[79] S. R. Fleagle, D. R. Thedens, W. Stanford, B. H. Thompson, J. M. Weston, P. P. Patel, and D. J. Skorton. Automated myocardial edge-detection on MR-images - accuracy in consecutive subjects. *J. Mag. Res. Imag.*, 3(5):738–741, 1993.

[80] R. M. Ford. Critical point detection in fluid flow images using dynamical system properties. *Pattern Recognition*, 30(12):1991–2000, 1997.

[81] B. H. Friemel, L. Weng, T. Teo, and K. G. Brown. 3-Dimensional volume by aggregating ultrasound fields of view, May 1999. United States patent 5,899,861. Application number 08/728,280.

[82] A. L. Fuhrmann and E. Gröller. Real-time techniques for 3D flow visualization. *Proc. Visualization.*, pages 305–312, 1998.

[83] K. Fukunaga. *Introduction to Statistical Pattern Recognition.* Academic Press, 2nd edition edition, 1990.

[84] A. Fyrenius, L. Wigström, T. Ebbers, M. Karlsson, J. Engvall, and A. F. Bolger. Three dimensional flow in the human left atrium. *Heart*, 86:448–155, 2001.

[85] H. Garcke, T. Preuer, M. Rumpf, A. Telea, U. Weikard, and J.J. van Wijk. A phase field model for continuous clustering on vector fields. *IEEE Transactions on Visualization and Computer Graphics*, 7(3):230–241, 2001.

[86] G. Gerig, O. Kübler, R. Kikinis, and F. Jolesz. Nonlinear anisotropic filtering of MRI data. *IEEE Trans. Med. Imag.*, 11:221–232, 1992.

[87] A. Giachetti. On-line analysis of echocardiographic image sequences. *Medical Image Analysis*, 2(3):261–284, 1998.

[88] J.D. Gill, H.M. Ladak, D.A. Steinman, and A. Fenster. Accuracy and variability assessment of a semi-automatic technique for segmentation of the carotid arteries from 3D ultrasound images. *Med. Phys.*, 27:1333–1342, 2000.

[89] R. Goldenberg, R. Kimmel, E. Rivlin, and M. Rudzsky. Cortex segmentation: A fast variational geometric approach. *IEEE Trans. Med. Imag.*, 21(12):1544–1551, 2002.

[90] J. W. Goodman. Some fundamental properties of speckle. *Journal of the Optical Society of America*, 66(11):1145–1150, 1976.

[91] K. C. Gowda and E. Diday. Symbolic clustering using a new dissimilarity measure. *IEEE Trans. Sys. Man Cybern.*, pages 368–378, 1992.

[92] S. R Gunn and M. S. Nixon. A robust snake implementation: A dual active contour. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(1):63–68, 1997.

[93] M. Halkidi, Y. Batistakis, and M. Vazirgiannis. On clustering validation techniques. *J Intelligent Info. Sys.*, 17(2-3):107–145, 2001.

[94] M. Halkidi, Y. Batistakis, and M. Vazirgiannis. Cluster validity methods: Part i. *Sigmod Record.*, 31(2):40–45, 2002.

[95] M. Halkidi, Y. Batistakis, and M. Vazirgiannis. Cluster validity methods: Part i. *Sigmod Record.*, 31(3):19–27, 2002.

[96] R. M. Haralick. Statistical and structural approaches to texture. *Proceedings of the IEEE*, 67(5):786–804, 1979.

[97] R. M. Haralick, K. Shanmugam, and I. Dinstein. Textural features for image classification. *IEEE Transactions on Systems, Man, and Cybernetics*, SMC-3(6):610–621, 1973.

[98] B. Heckel, G. Weber, B. Hamann, and K.I. Joy. Construction of vector field hierarchies. *Proc. IEEE Visualization'99*, pages 19–25, 1999.

[99] W. R. Hedrick, D. L. Hykes, and D. E. Starchman. *Ultrasound physics and instrumentation.* Mosby, 3rd edition, 1995.

[100] H. Hege and D. Stalling. Fast LIC with piecewise polynomial filter kernels. *Math. Vis.*, pages 295–314, 1998.

[101] J. Helman and L. Hesselink. Representation and display of vector field topology in fluid-flow data sets. *Computer*, 22(8):27–36, 1989.

[102] J. L. Helman and L. Hesselink. Visualizing vector field topology in fluid flows. *IEEE Computer Graphics and Applications*, 11(3):36–46, 1991.

[103] W. H. Highleyman. The design and analysis of pattern recognition experiments. *The Bell System Technical Journal*, March 1962.

[104] A. Hill, A. D. Brett, and C. J. Taylor. Automatic landmark identification using a new method of non-rigid correspondence. *Lec. Notes Comp. Sci.*, 1230:483–488, 1997.

[105] A. Hill, T. F. Cootes, C. J. Taylor, and K. Lindley. Medical image interpretation: a generic approach using deformable templates. *J. Medical Informatics*, 19(1):47–59, 1994.

[106] A. Hill, A. Thornham, and C. J. Taylor. Model-based interpretation of 3D medical images. *Proc. British Mach. Vis. Conf.*, pages 339–348, 1993.

[107] J. P. M. Hultquist. Interactive numerical flow visualization using stream surfaces. *Comp. Sys. Eng.*, 1(2-4):349–353, 1990.

[108] J. P. M. Hultquist. Constructing stream surfaces in steady 3D vector fields. *Proc. Visualization*, pages 171–178, 1992.

[109] I. J. Illingworth and J. Kittler. A survey of the hough transform. *Comp. Vis. Graph. Image Proc*, 44(1):87–116, 1988.

[110] M. in den Haak, H. J. W. Spoelder, and F. C. A. Groen. Matching of images by using automatically selected regions of interest. *Comp. Sci. Netherlands*, pages 27–40, 1992.

[111] V. Interrante and Grosch C. Strategies for effectively visualizing 3D flow with volume LIC. *Proc. Visualization*, pages 421–424, 1997.

[112] V. L. Interrante. Illustrating surface shape in volume data via principal direction-driven 3D line integral convolution. *Proc. SIGGRAPH.*, pages 109–116, 1997.

[113] A. K. Jain. Data clustering: a review. *ACM Computing Surveys*, 31(3):264–323, 1993.

[114] A. K. Jain and R. C. Dubes. Algorithms for clustering data. *Prentice-Hall advanced reference series, NJ*, 1988.

[115] A. K. Jain and P. J. Flynn. Image segmentation using clustering. *Advances in Image Understanding, Ahuja, N. and Bowyer, K. (eds.), IEEE Press, NJ*, 1996.

[116] K. Jajuga, A. Sokolowski, and H. H. (eds.) Bock. Classification, clustering and data analysis : recent advances and applications. *Springer, Berlin*, 2002.

[117] J. Jeong and F. Hussain. On the identification of a vortex. *J. Fluid Mech.*, 285:69–64, 1995.

[118] M. Jiang, R. Machiraju, and D. Thompson. A novel approach to vortex core region detection. *Eurographics - IEEE TCVG Symp. Visualization.*, pages 217–225, 2002.

[119] B. Jobard and W. Lefer. Creating evenly-spaced streamlines of arbitrary density. *Proc. 8th Worksh. Vis. Sci. Comp.*, 7, 1997.

[120] B. Jobard and W. Lefer. Unsteady flow visualization by animating evenly-spaced streamlines. *Comp. Graph. Forum.*, 19(3):C31–C39, 2000.

[121] B. Jobard and W. Lefer. Multiresolution flow visualization. *Proc. WSCG.*, pages 163–170, 2001.

[122] B. Julesz. Experiments in the visual perception of texture. *Scientific American*, 232(4):34–43, 1975.

[123] D. S. Kalivas and A. A. Sawchuk. A region matching motion estimation algorithm. *Comp. Vision, Graph. Im. Proc.: Im. Underst.*, 54(2):275–288, 1991.

[124] D. Kao, B. Zhang, K. Kim, and A. Pang. 3D flow visualization using texture advection. *Int. Conf. Comp. Graph. Imag.*, 2001.

[125] D. Kaplan and Q. Ma. On the statistical characteristics of log-compressed Rayleigh signals: theoretical formulation and experimental results. *Journal of the Acoustical Society of America*, 95(3):1396–1400, March 1994.

[126] M. Kass, A. Witkin, and D. Terzopoulos. Snakes: active contour models. *International Journal of Computer Vision*, 1(4):321–331, 1987.

[127] D. N. Kenwright and R. Haimes. Automatic vortex core detection. *IEEE Comp. Graph. App.*, 18(4):70–74, 1998.

[128] D. N. Kenwright and D. A. Lane. Interactive time-dependent particle tracing using tetrahedral decomposition. *IEEE TVCG.*, 2(2):120–129, 1996.

[129] P. J. Kilner, G. Z. Yang, R. H. Mohiaddin, D. N. Firmin, and D. B. Longmore. Helical and retrograde secondary flow patterns in the aortic arch studied by three-directional magnetic resonance velocity mapping. *Circulation*, 88(5):2235–2247, 1993.

[130] P. J. Kilner, G. Z. Yang, R. H. Mohiaddin, D. N. Firmin, and M. Yacoub. Asymmetric redirection of flow through the heart. *Nature*, 404:759–61, 2000.

[131] W. Y. Kim, P. G. Walker, E. M. Pedersen, J. K. Poulsen, S. Oyre, K. Houlind, and A.P. Yoganathan. Left ventricular blood flow patterns in normal subjects: a quantitative analysis by three-dimensional magnetic resonance velocity mapping. *J. Amer. Coll. Cardiol.*, 26(1):224–238, 1995.

[132] Y. Kim and S. Lee. A clustering validity assessment index. *Lec. Notes Art. Intell.*, 2637:602–608, 2003.

[133] M. Kiu and D. C. Banks. Multi-frequency noise for LIC. *Proc. Visualization*, pages 121–126, 1996.

[134] R. V. Klassen and S. J. Harrington. Shadowed hedgehogs: a technique for visualizing 2D slices of 3D vector fields. *Proc. Visualization*, pages 148–153, 1991.

[135] P. Kumar and M. Clark. *Clinical Medicine.* W. B. Saunders, Edinburgh, 2002.

[136] A. Lalande, L. Legrand, P. M. Walker, M. C. Jaulent, F. Guy, Y. Cottin, and F. Brunotte. Automatic detection of cardiac contours on MR images using fuzzy logic and dynamic programming. *J. Amer. Med. Inf. Assoc.*, Suppl. S:474–478, 1997.

[137] A. Lanitis, C. J. Taylor, and T. F. Cootes. Automatic interpretation and coding of face images using flexible models. *IEEE Trans. Patt. Anal. Mach. Intel.*, 19(7):743–756, 1997.

[138] R. S. Laramee. Interactive 3D fow visualization using a streamrunner. *Proc. Conf. Human Factors in Comp. Sys.*, pages 804–805, 2002.

[139] A. Latif-Amet, A. Ertuzun, and A. Ercil. An efficient method for texture defect detection: sub-band domain co-occurrence matrices. *Image and Vision Computing*, 18:543–553, 2000.

[140] C. S. F. Lee and L. Talbot. A fluid mechanical study on the closure of heart valves. *J. Fluid Mech.*, 91:41–63, 1979.

[141] J. S. Lee. Digitalimage enhancement and noise filtering by use of local statistics. *IEE. Trans. Patt. Anal. Mach. Intel.*, 2(2):165–168, 1980.

[142] R. C. T. Lee. Cluster analysis and its applications. *Advances in Information Series Science, Tou, J. T. (ed.), Plenum Press, NY*, 1981.

[143] T. K. Leung, M. C. Burl, and P. Perona. Finding faces in cluttered scenes using random labelled graph matching. *Proc. ICCV*, 1995.

[144] A. Lev, S. W. Zucker, and A. Rosenfeld. Iterative enhancement of noisy images. *IEEE Trans. Sys. Man Cyb.*, 7(2):482–484, 1977.

[145] Y. Levy, D. Degani, and A. Seginer. Graphical visualization of vortical flows by means of helicity. *AIAA Journal.*, 28(8):1347–1352, 1990.

[146] M. Lew and N. Huijsmans. Information theory and face detection. *Proc. ICPR*, pages 601–610, 1996.

[147] Y. Li and F. Santosa. A computational algorithm for minimising total variation in image restoration. *IEEE T Image Proc.*, 5:987–995, 1996.

[148] S. H. Lin, S. Y. King, and L. J. Lin. face recognition/detection by probabilistic decision-based neural network. *IEEE Trans. Neural Net.*, 8(1):114–131, 1997.

[149] S. K. Lodha, J. C. Renteria, and K. M. Roskin. Topology preserving compression of 2d vector fields. *Proc. IEEE Visualization*, 2000.

[150] H. Löffelmann and E. Gröller. Enhancing the visualization of characteristic structures in dynamical systems. *Proc. Eurographics*, pages 59–68, 1998.

[151] H. Löffelmann, L. Mroz, and E. Gröller. Hierarchical streamarrows for the visualization of dynamical systems. *Proc. Vis. Sci. Comp., Eurographics*, pages 155–164, 1997.

[152] H. Löffelmann, L. Mroz, E. Gröller, and W. Purgathofer. Stream arrows: enhancing the use of streamsurfaces for the visualization of dynamical systems. *The Visual Computer*, 13:359–369, 1997.

[153] Q. Long, R. Merrifield, G.Z. Yang, P.J. Kilner, and D.N. Firmin. The influence of inflow boundary conditions on intra left ventricle flow predictions. *J. Biomech. Eng.*, 125(6):922–927, 2003.

[154] Q. Long, X.Y. Xu, B. Ariff, S.A. Thom, A.D. Hughes, and A.V. Stanton. Reconstruction of blood flow patterns in a human carotid bifurcation: A combined CFD and MRI study. *JMRI*, 11:299–311, 2000.

[155] Wu C. M., Y. C. Chen, and K. S. Hsieh. Texture features for classification of ultrasonic liver images. *IEEE Transactions on Medical Imaging*, 11(2):141–152, 1992.

[156] R. A. MacLaughlin. Randomized hough transform: improved ellipse detection with comparison. *Patt. Rec. Lett.*, 19(3-4):299–305, 1998.

[157] R. Malladi, J. A. Sethian, and B. C. Vemuri. Shape modelling with front propagation - a level set approach. *IEEE Trans. Patt. Anal. Mach. Intel.*, 17(2):158–175, 1995.

[158] X. Mao, Y. Hatanaka, H. Higashida, and A. Imamiya. Image-guided streamline placement on curvilinear grid surfaces. *Proc. Visualization.*, 1998.

[159] F. H. Martini. *Fundamentals of Anatomy and Physiology.* Prentice Hall, New Jersey, 1995.

[160] S. Marx and C. Brown. Progressive livewire for automatic contour extraction. *Western New York Image Processing Workshop*, 2000.

[161] R. B. McCall. *Fundamental statistics for psychology.* Harcourt Brace Jovanovich, London, 3 edition, 1980.

[162] T. McInerney and D. Terzopoulos. Deformable models in medical image analysis: a survey. *Medical Image Analysis*, 1(2):91–108, 1996.

[163] G. J. McLachlan. Discriminant analysis and statistical pattern recognition. *Wiley, NY.*, 2004.

[164] D. Metaxas. Physics-based deformable models. Kluwer, 1996.

[165] R. S. Michalski and R. E. Stepp. Automated construction of classifications - conceptual clustering versus numerical taxonomy. *IEEE Trans. Patt. Anal. Mach. Intel.*, 5(4):396–410, 1983.

[166] M. Mignotte and J. Meunier. A multiscale optimization approach for the dynamic contour-based boundary detection issue. *Computerized Medical Imaging and Graphics*, 25(3):265–275, 2001.

[167] A. Miliou, S. J. Sherwin, and J. M. R. Graham. Wake topology of curved cylinders at low reynolds numbers. *(under review).*

[168] S. C. Mitchell, B. P. F. Lelieveldt, R. J. van der Geest, H. G. Bosch, J. H. C. Reiber, and M. Sonka. Multistage hybrid active appearance model matching: segmentation of left and right ventricles in cardiac MR images. *IEEE T. Med. Imag.*, 20(5):415–423, 2001.

[169] B. Moghaddam and A. Pentland. Probabilistic visual learning for object representation. *IEEE Trans. Patt. Anal. Mach. Intel.*, 19(7):696–710, 1997.

[170] R. Momenan, R. F. Wagner, B. S. Garra, M. H. Loew, and M. F. Insana. Image staining and differential diagnosis of ultrasound scans based on the Mahalanobis distance. *IEEE Transactions on Medical Imaging*, 13(1):37–47, 1994.

[171] B. S. Morse, S. M. Pizer, and A. Liu. Multiscale medial analysis of medical images. *Lecture Notes in Computer Science*, 687:112–131, 1993.

[172] A. A. Morsy and O. T von Ramm. 3D ultrasound tissue motion tracking using correlation search. *Utrasonic Imaging*, 20:151–159, 1998.

[173] E. N. Mortensen and W. A. Barrett. Interactive segmentation with intelligent scissors. *Graphical Models and Image Processing*, 60:349–384, 1998.

[174] S. K. Nayar, H. Murase., and S. Nene. Parametric appearance representation. *Early Visual Learning*, S. K. Nayar, T. Poggio (eds.), Oxford University Press, 1996.

[175] Y. H. P. Ng, B. S. Carmo, A. Prügel-Bennett, and G. Z. Yang. A first order lagrangian based variational approach for MR flow vector field restoration. *Proc. CARS 2003*, 2003.

[176] Y. H. P. Ng, B. S. Carmo, and G. Z. Yang. Flow field restoration and feature extraction for mr velocity mapping. *IEEE TMI, under review.*, 2003.

[177] Y. H. P. Ng and G. Z. Yang. Vector-valued image restoration with application to magnetic resonance velocity imaging. *Journal of WSCG*, 11(2), 2003.

[178] YHP Ng. Flow feature restoration, abstraction and tracking for velocity mri. *PhD thesis, Imperial College, London*, 2005.

[179] G. M. Nielson. Tools for computing tangent curves for linearly varying vector fields over tetrahedral domains. *IEEE TVCG.*, 5(4):360–372, 1999.

[180] S. Osher and R. Fedwick. Level Set Methods. *UCLA Tech. Rep. CAM-00-08.*, 2000.

[181] S. Osher and J. Sethian. Fronts propagating with curvature-dependent speed: algorithms based on the hamilton-jacobi formulation. *J. Computational Physics*, 79:12–49, 1988.

[182] E. Osuna, R. Freund, and F. Girosi. Training support vector machines: and application to face detection. *Proc. CVPR*, pages 130–136, 1997.

[183] C. P. Papageorgiou, M. Oren, and T. Poggio. A general framework for object detection. *Proc. ICCV*, pages 555–562, 1998.

[184] Y. Papaharilaou, D.J. Doorly, S.J. Sherwin, J. Peiro, J. Anderson, B. Sanghera, N. Watkins, and Caro C.G. Combined MRI and computational fluid dynamics detailed investigation of flow in a realistic coronary artery bypass graft model. *Proc. Int. Soc. Mag. Res. Med.*, page 379, 2001.

[185] N. Paragios. A variational approach for the segmentation of the left ventricle in cardiac image analysis. *Int. J. Comp. Vis.*, 50(3):345–362, 2002.

[186] N. Paragios. A level set approach for shape-driven segmentation and tracking of the left ventricle. *IEEE Trans. Med. Imag.*, 22(6):773–776, 2003.

[187] N. Paragios and R. Deriche. Geodesic active regions and level set methods for supervised texture segmentation. *Int. J. Comp. Vis.*, 46(3):223–247, 2002.

[188] W. Pedrycz. Fuzzy clustering with a knowledge-based guidance. *Pattern Recogn. Lett.*, 25(4):469–480, 2004.

[189] P. Perona and J. Malik. Scale-space and edge detection using anisotropic diffusion. *IEEE Trans. Patt. Anal. Mach. Intel*, 12:629–639, 1990.

[190] R. M. Portela. Identification and characterization of vortices in the turbulent boundary layer. *PhD thesis, Stanford University*, 1997.

[191] R. W Prager, A. H. Gee, G. M. Treece, and L. H. Berman. Decompression and speckle detection for ultrasound images using the homodyned k-distribution. Preprint submitted to Elsevier Preprint, November 2000.

[192] W. H. Press, S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery. *Numerical Recipes in C: the art of scientific computing.* Cam. Univ. Press, 1992.

[193] R. Rao and R. Jain. Analysing oriented textures through phase portraits. *10th Int. Conf. Patt. Rec.*, 1:336340, 1990.

[194] R. Rao and R. Jain. Computerized flow field analysis: Oriented texture fields. *IEEE Trans. Patt. Anal. Mach, Intel.*, 14(7):693–708, 1992.

[195] N. Reichek. MRI myocardial tagging. *J Magnetic Resonance Imaging*, 10(5):609–616, 1999.

[196] F. Reinders, J. E. D. Jacobson, and F. H. Post. Skeleton graph generation for feature shape description. *Data Vis., Proc. VizSym*, pages 73–82, 2000.

[197] F. Reinders, F. H. Post, and H. J. W. Spoelder. Attribute-based feature tracking. *Proc. VisSym, Data Vis.*, pages 63–72, 1999.

[198] F. Reinders, F. H. Post, and H. J. W. Spoelder. Visualization of time-dependent data using feature tracking and event detection. *The Vis. Comp.*, 17(1):55–71, 2001.

[199] F. Reinders, I. A. Sadarjoen, B. Vrolijk, and F. H. Post. Vortex tracking and visualisation in a flow past a tapered cylinder. *Computer Graphics Forum*, 21(4):675–682, 2001.

[200] C. Rezk-Salama, P. Hastreiter, C. Teitzel, and T. Ertl. Interactive exploration of volume line integral convolution based on 3D texture mapping. *Proc. Visualization.*, pages 233–240, 1999.

[201] M. Roth and R. Peikert. A higher-order method for finding vortex core lines. *Proc. Visualization '98*, pages 143–150, 1998.

[202] H. Rowley, S. Baluja, and T. Kanade. Neural network-based face detection. *IEEE Trans. Patt. Anal. Mach. Intel.*, 20(1):23–38, 1998.

[203] L. Rudin and S. Osher. Total variation based image restoration with free local constraints. *Proc. 1st IEEE ICIP*, pages 131–135, 1994.

[204] D. Rueckert, P. Burger, S. M. Forbat, R. D. Mohiaddin, and G. Z. Yang. Automatic tracking of the aorta in cardiovascular MR images using deformable models. *IEEE Transactions on Medical Imaging*, 16(5):581–590, 1997.

[205] N.R. Saber, A.D. Gosman, N.B. Wood, P.J. Kilner, C. L. Charrier, and D. N. Firmin. omputational flow modelling of the left ventricle based on in vivo MRI data - initial experience. *Annals Biomed. Engineering*, 29:275–283, 2001.

[206] I. A. Sadarjoen, A. J. de Boer, F. H. Post, and A. E. Mynett. Particle tracing in σ-transformed grids using tetrahedral 6-decomposition. *Proc. Eurographics.*, pages 71–80, 1998.

[207] I. A. Sadarjoen and F. H. Post. Geometric methods for vortex detection. *Proc. Data Vis.*, pages 53–63, 1999.

[208] R. Samtaney, D. Silver, N. Zabusky, and J. Cao. Visualizing features and tracking their evolution. *IEEE Computer.*, 27(7):20–27, 1994.

[209] A. Sanna, B. Montrucchio, and R. Arinaz. Visualizing unsteady flows by adaptive streaklines. *Proc. WSCG.*, 2000.

[210] G. Scheuermann, H. Kruger, M. Menzel, and A. P. Rockwood. Visualizing non-linear vector field topology. *IEEE TVCG.*, 4(2):109–116, 1998.

[211] H. Schneiderman and T. Kanade. A statistical method for 3D object detection applied to faces and cars. *Proc. CVPR*, 2000.

[212] W. Schroeder, K. Martin, and B. Lorensen. The visualization toolkit an object-oriented approach to 3d graphics. *Kitware*, 2003.

[213] W. Schroeder, C. R. Volpe, and W. E. Lorensen. The stream polygon: a technique for 3D vector field visualization. *Proc. Visualization.*, pages 126–132, 1991.

[214] T.B. Sebastian, H. Tek, J.J. Crisco, S.W. Wolfe, and B.B. Kimia. Segmentation of carpal bones from 3d ct images using skeletally coupled deformable models. *Lec. Notes Comp. Sci.*, 1496:1184–1194, 1998.

[215] I. K. Sethi, N. V. Patel, and J. H. Yoo. A general approach for token correspondence. *Patt. Recog.*, 27(12):1775–1786, 1994.

[216] J. A. Sethian. *Level Set Methods and Fast Marching Methods.* Cambridge Monographs on Applied and Computational Mathematics. Cambridge University Press, Cambridge, UK, 2 edition, 1999.

[217] H. W. Shen, C. R. Johnson, and K. L. Ma. Visualizing vector fields using line integral convolution and dye advection. *Proc. Symp. Vol. Vis.*, pages 63–70, 1996.

[218] C. F. Shu, R. Jain, and F. Quek. A linear algortithm for computing the phase portraits of oriented textures. *Proc. IEEE Comp. Vis. Patt. Rec.*, pages 352–357, 1991.

[219] J. Shufelt. Performance evaluation and analysis of monocular building extraction from artificial imagery. *IEEE Trans. Patt. Anal. Mach. Intel.*, 21(4):311–326, 1999.

[220] K. Siddiqui, Y. B. Lauziere, A. Tannenbaum, and S. Zucker. Area and length minimizing flows for shape segmentation. *IEEE Conf. Comp. Vis. Patt. Recog.*, pages 621–627, 1997.

[221] D. Silver and X. Wang. Volume tracking. *Proc. Vis.*, pages 157–164, 1996.

[222] D. Silver and X. Wang. Tracking and visualizing turbulent 3D features. *IEEE T. Vis. Comp. Graphics*, 3(2):129–141, 1997.

[223] D. Silver and X. Wang. Visualizing evolving scalar phenomena. *Future Gen. Comp. Sys.*, 15(1):99–108, 1999.

[224] J. R. Singer. NMR diffusion and flow measurement and an introduction to spin phase graphing. *J Phys E: Sci Intrum*, 11:281–291, 1978.

[225] W. L. Smith and A. Fenster. Optimization of 3D ultrasound scan spacing using speckle statistics. In K. K. Shung and M. F. Insana, editors, *Ultrasonic Imaging and Signal Processing, Proc. SPIE Vol. 3982*, pages 56–67, San Diego, California, February 2000.

[226] M. Sonka, V. Hlavac, and R. Boyle. *Image Processing, Analysis and Machine Vision*. PWS Publishing, Pacific Grove, CA, 1998.

[227] D. Stalling and H. Hege. Fast and resolution independent line integral convolution. *Proc. SIGGRAPH*, pages 249–256, 1995.

[228] M. B. Stegmann, R. Fisker, and B. K. Ersbll. Extending and applying active appearance models for automated, high precision segmentation in different image modalities. *Proc. 12th Scandinavian Conference on Image Analysis - SCIA 2001, Bergen, Norway*, 2001.

[229] D. Sujudi and R. Haimes. Identification of swirling flow in 3-D vector fields. *AIAA 12th Comp. Fluid Dynamics Conf.*, pages Paper 95–1715, 1995.

[230] K. Sung and T. Poggio. Example-based learning for view-based human face detection. *IEEE Trans. Patt. Anal. Mach. Intel.*, 20(1):39–52, 1998.

[231] C. Teitzel and T. Ertl. New approaches for particle tracing on sparse grids. *Proc. Data Vis., Eurographics*, pages 73–84, 1999.

[232] C. Teitzel, R. Grosso, and T. Ertl. Efficient and reliable integration methods for particle tracing in unsteady flows on discrete meshes. *Proc. Eurographics.*, pages 31–42, 1997.

[233] C. Teitzel, R. Grosso, and T. Ertl. Particle tracing on sparse grids. *Proc. Eurographics.*, pages 81–90, 1998.

[234] A. Telea and J. J. van Wijk. Simplified representation of vector fields. *Proc. IEEE Visualization99*, pages 35–42, 1999.

[235] T. Teo. Method and apparatus for blood speckle detection in an intravascular ultrasound imaging system., March 1999. United States patent 5,876,343.

[236] D. Terzopoulos, J. Platt, A. Barr, and K. Fleischer. Elastically deformable model. *Computer Graphics.*, 21:205–214, 1987.

[237] A. N. Tikhonov and V. Y. Arsenin, editors. *Solutions of ill-posed problems.* John Wiley, London, 1977.

[238] A. Tsai, A. Yezzi and A. Willsky. A curve evolution approach to smoothing and segmentation using the Mumford-Shah functional. *IEEE Conf. Comp. Vis. Patt. Recog.*, pages 119–124, 2000.

[239] G. Turk and D. Banks. Image-guided streamline placement. *Proc. SIGGRAPH*, pages 453–460, 1996.

[240] T. A. Tuthill, J. F. Krücker, J. B. Fowlkes, and P. L. Carson. Automated three-dimensional US frame positioning computed from elevational speckle decorrelation. *Radiology*, 209:575–582, 1998.

[241] J. K. Udupa. 3dviewnix: a data-, machine-, and application-independent software system for the visualization and analysis of multidimensional images.

[242] J. K. Udupa and G. T. Herman. *3D Imaging in Medicine.* CRC Press, New York, 2nd edition, 2000.

[243] S. K. Ueng, C. Sikorski, and K. L. Ma. Efficient streamline, streamribbon and streamtube constructions on unstructured grids. *IEEE TVCG.*, 2(2):100–110, 1996.

[244] S. Ullman. Aligning pictorial descriptions: an approach to object recognition. *Cognition*, 32(3):193–254, 1989.

[245] F. M. J. Valckx and J. M. Thijssen. Characterization of echographic image texture by coocurrence matrix parameters. *Ultrasound in Medicine and Biology*, 23(4):559–571, 1997.

[246] R. J. van der Geest, V. G. M. Buller, E. Jansen, H. J. Lamb, L. H. B. Baur, E. E. van der Wall, A. de Roos, and J. H. C. Reiber. Comparison between manual and semiautomated analysis of left ventricular volume parameters from short-axis MR images. *J. Comp. Assisted Tomography*, 21(5):756–765, 1997.

[247] R. J. van der Geest, E. Jansen, V. G. M. Buller, and J. H. C. Reiber. Automated detection of left ventricular epi- and endocardial contours in short-axis MR images. *Proc. Comp. Cardiol.*, pages 33–36, 1994.

[248] R. J. van der Geest and J. H. C. Reiber. Quantification in cardiac MRI. *J. Magn Res. Imag.*, 10:602–608, 1999.

[249] T. van Walsum, F. H. Post, D. Silver, and F. J. Post. Feature extraction and iconic visualization. *IEEE TVCG.*, 2(2):111–119, 1996.

[250] J. J. van Wijk. Spot noise-texture synthesis for data visualization. *Comp. Graphics, Proc. SIGGRAPH*, pages 309–318, 1991.

[251] J. J. van Wijk. Flow visualization with surface particles. *IEEE Comp. Graph. Appl.*, 13(4):18–24, 1993.

[252] J. J. van Wijk. Implicit stream surfaces. *Proc. Visualization.*, pages 245–252, 1993.

[253] V. Verma, D. Kao, and A. Pang. A flow-guided streamline seeding strategy. *Proc. Visualization.*, pages 163–170, 2000.

[254] J. Villasenor and A. Vincent. An algorithm for space recognition and time tracking of vorticity tubes in turbulence. *Comp. Vision, Graph. Im. Proc.: Im. Underst.*, 55(1):27–35, 1992.

[255] D. G. Vince, K. J. Dixon, R. M. Cothren, and J. F Cornhill. Comparison of texture analysis methods for the characterization of coronary plaques in intravascular ultrasound images. *Computerized Medical Imaging and Graphics*, 24:220–229, 2000.

[256] C. R. Vogel and M. E. Oman. Iterative methods for total variation denoising. *SIAM Journal on Scientific Computing*, 17(1):227–238, 1996.

[257] R.F. Wagner, M. F. Insana, and D. G. Brown. Unified approach to the detection and classification of speckle texture in diagnostic imaging. *Optical Engineering*, 25(6):738–742, 1986.

[258] R.F. Wagner, Smith S. W., J. M. Sandrick, and H. Lopez. Statistics of speckle in ultrasound B-scans. *IEEE Transactions on Sonics and Ultrasonics*, 30(3):156–163, 1983.

[259] P. G. Walker, G. B. Cranney, R. Y. Grimes, J. Delatore, J. Rectenwald, G. M. Pohost, and A. P. Yoganathan. Three-dimensional reconstruction of the flow in a human left heart by using magnetic resonance phase velocity encoding. *J. Biomedical Eng.*, 24:139–147, 1996.

[260] D. C. Wang, A. H. Vagucci, and C. C. Li. Gradient inverse weighted scheme and the evaluation of its performance. *Computer Graphics and Image Processing.*, 15:167–181, 1981.

[261] X. Wang. On the gradient inverse weighted filter. *IEEE Trans. Signal Proc.*, 40(2):482–484, 1992.

[262] S. Webb, editor. *The Physics of Medical Imaging.* Institute of Physics Publishing, Bristol, 1988.

[263] C. Weigle and D. C. Banks. Extracting iso-valued features in 4-dimensional scalar fields. *Proc. Symp. Vol. Vis.*, pages 103–110, 1998.

[264] J. Weng, N. Ahuja, and T. S. Huang. Learning, recognition and segmentation using the cresceptron. *Int. J. Comp. Vision*, 25:105–139, 1997.

[265] L. Weng and A. P. Tirumalai. Method and apparatus for generating large compound utrasound image. United States Patent 5,575,286 Application Number 414,978, November 1996.

[266] L. Weng, A. P. Tirumalai, and L. Nock. Method and apparatus for generating and displaying panoramic ultrasound images. United States Patent 5,782,766 Application Number 747,429, July 1998.

[267] D. J. Williams and M. Shah. A fast algorithm for active contours and curvature estimation. *CVGIP: Image Understanding*, 55(1):14–26, 1992.

[268] P. C. Wong, H. Foote, R. Leung, E. Jurrus, D. Adams, and J. Thomas. Vector fields simplification - a case study of visualizing climate modelling and simulation data sets. *IEEE Conf. Visualization*, pages 485–488, 2000.

[269] G. Z. Yang. The role of quantitative MR velocity imaging in exploring the dynamics of in vivo blood flow. *IEEE Engineering in Medicine and Biology*, 17:64–72, 1998.

[270] G. Z. Yang, P. Burger, D. N. Firmin, and S. R. Underwood. Structure adaptive anisotropic image filtering. *Image and Vision Computing*, 14:135–145, 1996.

[271] G. Z. Yang, P. Burger, J. Panting, P. D. Gatehouse, D. Rueckert, D. J. Pennell, and D. N. Firmin. Motion and deformation tracking for short-axis echo-planar myocardial perfusion imaging. *Med. Image Anal.*, 2(3):285–302, 1998.

[272] G. Z. Yang, J. Keegan, and D. N. Firmin. Motion-selctive encoding for fast cine imaging. *Mag. Res. Med.*, 42:430–435, 1999.

[273] G. Z. Yang, P. J. Kilner, R. H. Mohiaddin, and D. N. Firmin. Transient streamlines: texture synthesis for in vivo flow visualisation. *Int. J. Card. Imag.*, 16(3):175–184, 2000.

[274] G. Z. Yang, R. H. Mohiaddin, P. J. Kilner, and D. N. Firmin. Vortical flow feature recognition: a topological study of in vivo flow patterns using MR velocity mapping. *J. Comp. Assist. Tomography*, 22(4):577586, 1998.

[275] L. A. Yates and G. T. Chapman. Streamlines, vorticity lines, and vortices. *AIAA Tech. Rep. 91-0731.*, 1991.

[276] A. Yezzi, S. Kichenassamy, A. Kumar, P. Olver, and A. Tannenbaum. A geometric snake model for segmentation of medical imagery. *IEEE Transactions on Medical Imaging*, 16(2):199–209, 1997.

[277] A. Yuille, D. Cohen, and P. Hallinan. Feature extraction from faces using deformable templates. *Proc. IEEE Conf. Comp. Vis. Patt. Recog.*, pages 104–109, 1989.

[278] N. J. Zabusky, O. N. Boratav, R. B. Pelz, M. Gao, D. Silver, and S. P. Cooper. Emergence of coherent patterns of vortex stretching during reconnection: a scattering paradigm. *Phys. Rev. Lett.*, 67(18):2469–2472, 1991.

[279] L. A. Zadeh. Fuzzy-logic. *Computer.*, 21(4):83–93, 1988.

[280] X. L. Zeng, L. H. Staib, R. T. Schultz, and J. S. Duncan. Segmentation and measurement of the cortex from 3-d mr images using coupled-surfaces propagation. *IEEE Trans. Med. Imag.*, 18(10):927–937, 1999.

[281] X. L. Zeng, L. H. Staib, R. T. Schultz, H. Tagare, L. Win, and J. S. Duncan. A new approach to 3d sulcal ribbon finding from mr images. *Lec. Notes Comp. Sci.*, 1679:148–157, 1999.

[282] X. Zhang, C. R. McKay, and M. Sonka. Tissue characterization in intravascular ultrasound images. *IEEE Transactions on Medical Imaging*, 17(6):889–899, 1998.

[283] Y. Zimmer, R. Tepper, and S. Akselrod. A two-dimensional extension of minimum cross entropy thresholding for the segmentation of ultrasound images. *Ultrasound in Medicine and Biology*, 22(9):1189–1190, 1996.

[284] M. Zöckler, D. Stalling, and H. Hege. Interactive visualization of 3D-vector fields using illuminated streamlines. *Proc. Visualization*, pages 107–113, 1996.

[285] M Zöckler, D. Stalling, and H. Hege. Parallel line integral convolution. *Par. Comp.*, 23(7):975–989, 1997.