Pattern Analysis & Applications (2000)3:39–60

Greedy Algorithm for Error Correction in Automatically Produced Boundaries from Low Contrast Ventriculograms

J. S. Suri[1], **R. M. Haralick**[1] **and F. H. Sheehan**[2]

[1]*Intelligent Systems Laboratory, Department of Electrical Engineering, University of Washington, Seattle, WA;*
[2]*Cardiovascular Research & Training Center, University of Washington Medical Center, University of Washington, Seattle, WA, USA*

Abstract: Non-homogeneous mixing of the dye with the blood in the left ventricle chamber of the heart causes poor contrast in the ventriculograms. The pixel-based classifiers [1] operating on these ventriculograms yield boundaries which are not close to ground truth boundaries as delineated by the cardiologist. They have a mean boundary error of 6.4 mm and an error of 12.5 mm in the apex zone. These errors have a systematic positional and orientational bias, the boundary being under-estimated in the apex zone. This paper discusses two calibration methods: the *identical coefficient* and the *independent coefficient* to remove these systematic biases. From these methods, we constitute a fused algorithm which reduces the boundary error compared to either of the calibration methods. The algorithm, in a greedy way, computes which and how many vertices of the left ventricle boundary can be taken from the computed boundary of each method in order to best improve the performance. The corrected boundaries have a mean error of less than 3.5 mm with a standard deviation of 3.4 mm over the approximately 6×10^4 vertices in the data set of 291 studies. Our method reduces the mean boundary error by 2.9 mm over the boundary produced by the classifier. We also show that the calibration algorithm performs better in the apex zone where the dye is unable to propagate. For end diastole, the algorithm reduces the error in the apex zone by 8.5 mm over the pixel-based classifier boundaries.

Keywords: Boundaries; Calibration; Left ventricle; Low contrast; Motion; Optimisation; Polyline

1. INTRODUCTION

Boundary estimation of the Left Ventricle (LV) is needed for quantification of the left ventricle motion. This helps cardiologists in studying different kinds of cardiomyopathies. The quantification can be done using manual ways, but it is very tedious to trace these boundaries for each time frame of the cardiac cycle and for large voluminous data sets. As a result, cardiologists are very interested in its automatisation. This research solves this problem.

One of the popular methods for studying cardiac disorders is by catheterisation. In this procedure, the contrast medium or dye (radio opaque material) is injected into the left ventricle where it mixes with the blood during the heart cycle. Because this mixing takes place in a non-homogeneous manner, therefore very little dye reaches the apex zone (see Figs 1-(left), 2 and 3). Consequently, the grey scale contrast of the apex zone is poor. In addition, the ventriculograms (LVgrams) have a high level of noise due to the scattering by tissue volumes which are not related to the left ventricle. Due to the above problems, accurate and automatic boundary estimation process of the left ventricle becomes a difficult process.

Left ventricle extraction is not new to pattern recognition researchers. In fact segmentation of left ventricles for medical diagnosis began when the imaging systems became available for imaging left ventricle (see Moore et al [2], Moodi et al [3] and Mancini et al [4]). In heart imaging, the shape of the left ventricle varies considerably between patient studies and through time. The foundation of most of the boundary segmentation techniques have been histogram-based or traditional edge detection based (see Chow et al [5], Griffith et al [6], Tananka et al [7], Jong et al [8], Reiber et al [9], Han et al [10] and Wollschlaeger et al

Received: 3 September 1998
Received in revised form: 16 April 1999
Accepted: 28 April 1999

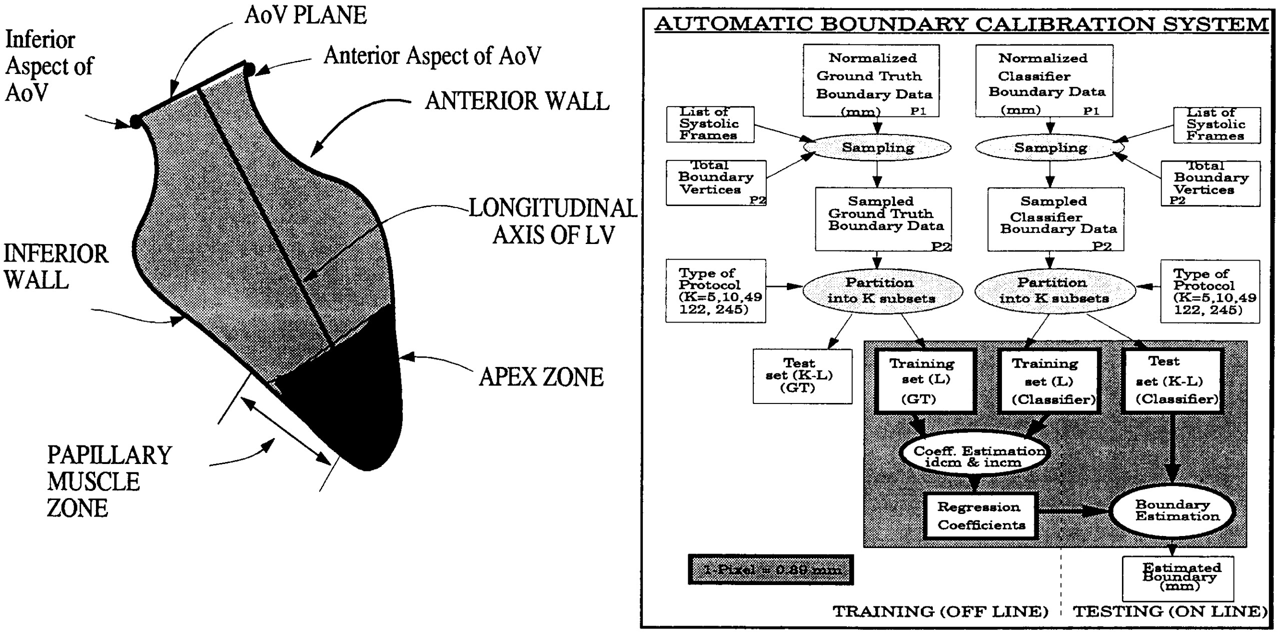

AoV PLANE
Inferior Aspect of AoV
Anterior Aspect of AoV
ANTERIOR WALL
LONGITUDINAL AXIS OF LV
INFERIOR WALL
APEX ZONE
PAPILLARY MUSCLE ZONE
AUTOMATIC BOUNDARY CALIBRATION SYSTEM
Normalized Ground Truth Boundary Data (mm) P1
Normalized Classifier Boundary Data (mm) P1
List of Systolic Frames
Total Boundary Vertices P2
Sampling
Sampled Ground Truth Boundary Data P2
Sampled Classifier Boundary Data P2
Type of Protocol (K=5,10,49 122, 245)
Partition into K subsets
Test set (K-L) (GT)
Training set (L) (GT)
Training set (L) (Classifier)
Test set (K-L) (Classifier)
Coeff. Estimation idcm & incm
Regression Coefficients
Boundary Estimation
Estimated Boundary (mm)
1 Pixel = 0.39 mm
TRAINING (OFF LINE)
TESTING (ON LINE)

Fig. 1. Left: Labeling of the left ventricle. Right: object process diagram for the IdCM and InCM calibrations for any frame of the heart cycle. We sample and interpolate the input normalised data, followed by partitioning the data into K subsets, L used for training and $K-L$ used for the testing set. Magnification conversion factors are used to convert pixels to millimeters where 1 pixel = 0.39 mm.

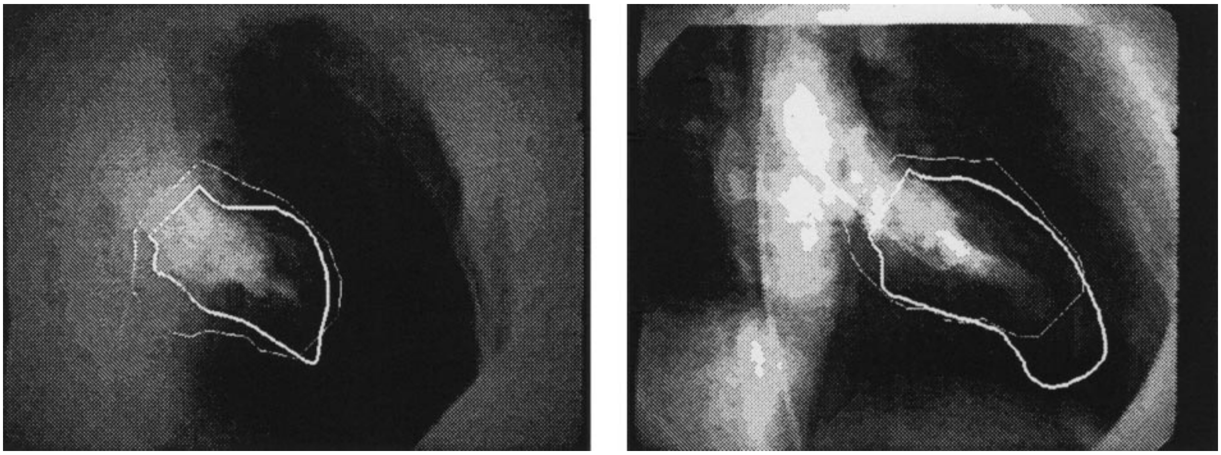


Fig. 2. Left: end-systole (ES) frame showing very little dye in the apex zone of the LV. ES frame is that frame of the heart cycle when the LV has fully contracted. It is named so because it is the last frame of the systole (contradiction) cycle. Right: end-systole frame showing with little dye and interference by ribs with the LV. Thick lines represent the border drawn by the cardiologist. Thin lines are the border computed by the pixel-based classifier or boundaries produced by image processing algorithms. Background consists of grey scale ventriculograms (LVgrams) of size 384 × 512.

[11]). All these methods are either semi-automatic or are not robust enough to automatically produce reliable boundaries. They lack model-based processing. We will concentrate here on model-based left ventricle estimation techniques geared towards learning.

The goals of this research are to introduce two model-based left ventricle calibration algorithms to correct biases in raw left ventricle boundaries, and to introduce a fusion algorithm to fuse two sets of boundaries using a greedy technique. Besides the above goals, this research introduces the following new items: (1) a methodology by which the global shape information is extracted from the ground truth in the form of coefficients which is then utilised to remove the bias error in raw boundaries. This knowledge of the left ventricle global shape is the human-traced left ventricle boundaries taken from a given population. This calibration involves an *off-line* training system which can train two sets of boundaries: the boundaries produced by an image processing algorithm (classifier boundaries) and the boundaries traced by the physicians. The trained system can then be

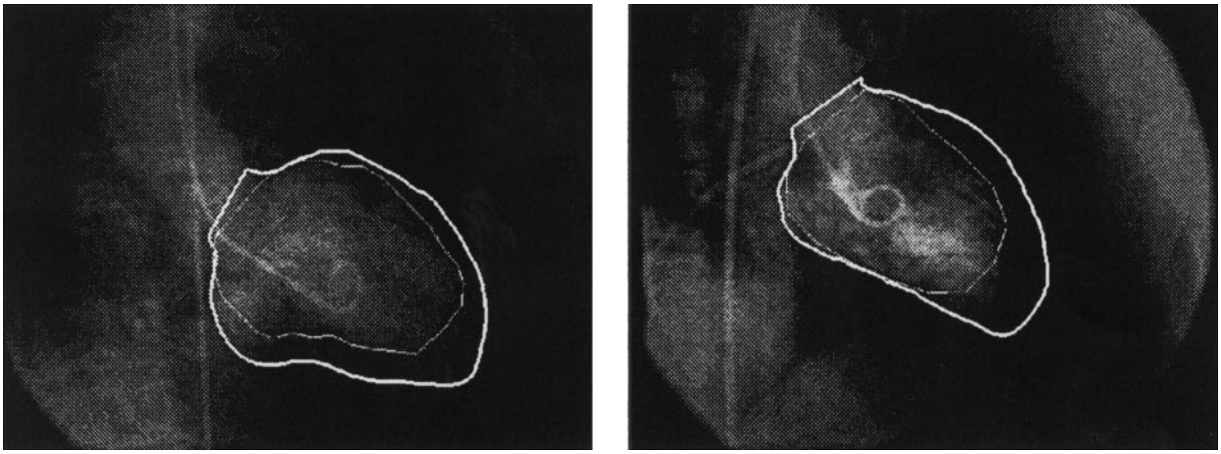


Fig. 3. Left: end-diastole (ED) frame showing very little dye in the apex zone of the LV. ES frame is that frame of the heart cycle when the LV has fully expanded or dilated. It is named so because it is the last frame of the diastole (expansion) cycle. Right: end-diastol frame showing the pixel-based classifier boundaries are under-estimated in anterior, inferior walls and apex zone. Thick lines represent the border drawn by the cardiologist. Background consists of grey scale LVgrams of size 384 × 512.

applied *on-line* to new patient boundaries generated by an image processing algorithm; (2) introduce the ball-basket model to reduce the error in the fused boundary; (3) a polyline measurement tool to measure the error between two left ventricle boundaries; (4) finally, develop an image processing based validation scheme to spot the boundaries which are not clinically useful. The remaining part of this section concentrates on some directly relevant model-based semi-automatic and automatic boundary estimation techniques.

Semi-automatic model based techniques: Van Bree et al [12] attempted to find the left ventricle borders using a combination of probability surfaces and dynamic programming. Van Bree's algorithm consisted of three major steps. The first step was building the probability surfaces from a database of hand-drawn boundaries after the left ventricle boundaries had been corrected for translation and rotation. The second step consisted of the generation of extraction lines (also called search lines) for the design of the search matrix. These search lines are the lines which originated from the interior of the left ventricle and extend radically outward to the outer region (background) of the left ventricle. The third step consisted of smoothing the search matrix, followed by dynamic programming to search for the optimal path of the left ventricle border.

Active shape modelling has been in existence in medical imaging, since Kass et al [13] developed the Snake model. This paper does not compare snakes and our model. The main drawback of the snakes model is the requirement of the user-interaction and being unable to place the initial contour on the LVgrams. The application of active shape modelling has been applied by several groups. Cootes et al [14] attempted using an active shape model based on Least Squares to infer the position of boundary parts where there was missing data (top of the ventricle). There are a lot of similarities between Cootes method and our technique, discussed in Section 3.

In learning algorithms, Hwang et al [15] addressed the issue of contour finding using a combination of neural networks, active contour and Gibbs sampler, called NNS-SNAKE, named after a neural net stochastic-snake. Chiou et al used a neural network classifier for building the interior contour, inside which the initial snake is placed. The active contour model is used to alter this snake towards the target contour. A Gibbs sampler basically helps to avoid trapping of the intermediate contours during the slithering operation of the snake. Chiou's algorithm for boundary estimation consists of three major steps. First, interior contour generation using the neural network classifier. In this step, Chiou et al first train the neural net using contour pixels and non-contour pixels. Now the neural net is fed by the test data set which outputs the energy profile image. This image is first binarised and then ANDed with the edge map of the original image. The ANDed image is then smoothed using a Gaussian kernel, and inverted, which yields the interior contour. The second step is generation of the starting contour which is inside the interior contour, and then running the active contour model to inflate towards the target contour. In step three, Gibbs sampling is used to avoid trapping of the intermediate contour which could happen by the local minimum of the energy function. This is done using the Bayes approach by taking the prior probability of the shape and likelihood function of the output of the neural network. Chiou et al used MRI brain images for training and testing of the neural network to obtain the final contours of the brain inside the brain cavity slice-by-slice.

Lee [1] was the first to actually automatically estimate the left ventricle boundaries in LVgrams. Lee used a pixel-based Bayesian approach for the left ventricle boundary extraction, looking at the grey scale value of the location throughout the cardiac cycle as a vector. Lee assumed that the distribution was bi-variate normal. For each vector observed, one class is assigned according to the ground truth, which is available by filling the left ventricle region surrounded by the ground truth left ventricle boundary. The ground truth of a pixel through F frames generates 2^F

possible classes, but far less than 2^F classes are actually required to represent all existing classes. Due to the different heart rates (i.e. variable number of frames in the cardiac cycle for different studies), Lee did interpolation to normalise the number of frames to 12, which corresponded to a systole cycle. Lee simultaneously segmented all the frames from the end diastole through to the end systole to the next end diastole, instead of segmenting frame-by-frame separately, assuming that each pixel was statistically independent in the same frame. Lee later modified his assumption by considering the nearest neighbor. For performance and error computation of the detected left ventricle boundary, Lee compared the classifier boundary with the manually traced boundary using the Hausdorf distance measure. These classification algorithms, however, did not yield boundaries close to the hand-drawn boundaries as delineated by the cardiologist. Of the many reasons for the failure of pixel-based image processing algorithms, foremost is the poor contrast in the apex zone that causes the image processing boundaries to fall short of ground truth boundaries or to be underestimated (see Figs 2 and 3). Furthermore, irregularities of the inferior walls, variations in the left ventricle shape, size, orientation and heart rate cause the boundaries from the pixel-based classifier output to push outwards or to be overestimated (see Fig. 2). Due to all of the above complications, a reliable left ventricle boundary detection algorithm must make use of as much knowledge of the left ventricle shape, size, position and orientation as possible.

In this paper we show that the aforementioned training procedure is a very promising technique for measuring the accuracy of left ventricle borders of the left ventricle, particularly in the apex zone and walls where the dye has not mixed well with the blood. We show that when the trained system is applied to new left ventricle test patient boundaries generated from the pixel-based classifier, the system reduces the systematic boundary biases and provides automatic left ventricle boundary delineation that are within 3.5 mm of the physician traced boundaries. Our procedure can be thought of as a calibration procedure: calibrating the initial pixel-based boundaries closer to the ground truth boundaries by removing any systematic bias in shape, position and orientation.

The layout of the paper is as follows. In Section 2, we formally state the two calibration algorithms. The fusion algorithm using the greedy approach is presented in Section 3. Section 4 states the mathematical derivation for the polyline method, a tool used for the measurement of boundary errors. The results of the three calibration algorithms are presented in Section 5, along with data analysis. Section 6 presents the reliability algorithm and validation scheme for spotting the weak points of the system for its feedback. Finally, we conclude the paper with discussions and conclusion sections.

2. TWO COEFFICIENT METHODS: IdCM AND InCM

This section presents the mathematical statements of the two calibration methods for bias correction of the classifier boundaries produced by Bayesian classifier [1]. We will not discuss the classification scheme here, but interested readers can look at the dissertation by Lee [1].

In the *identical coefficient* method, each vertex is associated with a set of coefficients. The calibrated x-coordinate for that vertex is computed as the linear combination of raw x-coordinates of the left ventricle boundary using the training coefficients associated with that vertex. The calibrated y-coordinate of that vertex is similarly computed as the *same* linear combination of raw y-coordinates of the left ventricle boundary. In the *independent coefficient* method, the calibrated x-coordinate is computed as the linear combination of raw x- and raw y-coordinates of the left ventricle boundary, using the training coefficients associated with that vertex. The calibrated y-coordinate of that vertex is computed with a *different* linear combination of raw x- and y-coordinates. The problem of calibration then reduces to a problem of determining the coefficients of the linear combination which can be accomplished by solving a regression problem. The initial (x,y) coordinates of the left ventricle are converted from pixels to millimeters using a magnification correction factor. This factor is computed by keeping a grid of lead wires of a known millimeter size or a kugel of a known diameter (in mm) over the ventriculograms. The kugel is approximately the same size as the left ventricle (approximately 70 millimeters). These input raw and ground truth boundaries are initially in an irregularly spaced vertex polygon format with 100 vertices and unit dimensions in millimeters. Thus there is a need for changing the polygons dfvxfcto equally spaced vertices, as discussed in the next section.

2.1. Data Correspondence: Interpolation & Resampling

Due to the variation in left ventricle sizes, we therefore resample and interpolate each of these polygons into polygons with equally spaced vertices. Note that here, since we have no prior information about the motion such as uniform expansion or contraction, we thus take the simplest case by using equal-sampling-normalisation, similar to approach by Duncan et al [16,17]. The interpolation is done with respect to the arc length. The arc length for a vertex ‘v’ is defined as the distance traveled along the left ventricle contour to that vertex ‘v’, starting from the anterior aspect of the aortic valve (i.e. clockwise direction). Thus, the arc length for the last vertex is the perimeter of the left ventricle contour. Since the original contour is sampled into P_2 vertices, the interval length between the vertices of the sampled contour is given as: $\mathcal{P}/(P_2 - 1)$, where $\mathcal{P}$ is the perimeter of the left ventricular contour, given as

$$\mathcal{P} = \sum_{i=2}^{P_1} \sqrt{(x_i - x_{i-1})^2 + (y_i - y_{i-1})^2}$$

As a result of this resampling and interpolation process, every vertex number of ground truth boundary corresponds to the same vertex of the classifier boundary. This resampling and linear interpolation is done in an automatic way, where the user needs only to specify the total number of boundary vertices to be sampled, and a list of frames of the cardiac cycle which needs to be sampled. In our case, we choose two frames for sampling: the end diastole (first frame) and the end systole (last frame), and the number of sampled boundaries vertices are P_2. Note that the aortic valve (AoV) plane is known. This means the (x,y)-coordinates for the anterior aspect of the aortic valve and the inferior aspect of the aortic valve are known; this is called two feature correspondence.

Note that the above data correspondence scheme takes the prior information of the AoV plane (similar to Van Bree et al [12]) between the classifier and the ground truth boundary, while the remaining vertices are equally arc sampled and interpolated. This correspondence uses two features: the first and last points of the left ventricle contour (see Fig. 1, left). There are other features on the left ventricle contour which can also be used for data correspondence, e.g. the apex point (furthest point from the mid AoV plane to the bottom one-third of the left ventricle). There are two ways to incorporate the apex information in the data correspondence. We must either find the apex from the grey scale images, or use the ground truth apex as delineated by the cardiologist. Estimating the apex from the grey scale images is a very difficult process, since the apex zone is the most uncertain zone in the LVgrams (as we know, that there is no grey scale information present). The ES grey scale apex is even harder to estimate than the ED grey scale apex. This is because, during the ES frame, the dye is emptying the left ventricle chamber and the contrast level further decreases in the apex zone. In contrast to estimating the apex using pattern recognition techniques, Suri et al [18] developed schemes to estimate an ES apex using the dependence approach (not part of this paper). Using prior information of the ground truth, the ES apex as delineated by the cardiologist, and the observed ED apex, Suri et al estimated the ES apex using a traning-based system. This estimated ES apex now can be used for the data correspondence for ES frame boundaries. Suri et al [19] also developed a robust apex estimation scheme where they estimated the apex using a weighted iterative least squares algorithm. This apex can be used as the third feature in data correspondence. Suri et al [21] recently developed a forced calibration algorithm, where the left ventricle contour is forced to pass through the apex point. This can be thought of as putting a penalty on the apex point and forcing the left ventricle contour to pass through the apex. Thus, this method is very similar to using three feature points (or three point correspondence): two points from the AoV plane, and the third point as the apex. Using this concept, the two wall curves can be independently equally arc sampled on both the sides of the apex point. We are currently developing a robust three feature data correspondence. The three point correspondence is an alternative method, and can be considered as an option if the apex information is available. In this paper, we take the data correspondence by taking only the two feature points, which is a reasonable assumption, given that there is no prior information about the apex position or the source of the classifier boundaries. Thus, we can consider the above correspondence to be a simple case of a non-rigid correspondence. Using this relaxed assumption, we will see that we loose hardly any information around the left ventricle contour, except little near the apex. Using the two feature correspondence, we learn the global left ventricle shapes based on the first layer of the neural networks, yielding results up to the expectations of the cardiologists. This involves the IdCM and InCM calibrations, which are mathematically stated in the next section.

2.2. Identical Coefficient Method (IdCM) for any Frame

Let $g'_n = [x_1,x_2,\ldots,x_P]_n$ and $h'_n = [y_1,y_2,\ldots,y_P]_n$ be the row vectors of x-coordinates and y-coordinates, respectively, for the ground truth boundaries for patient n. Let $r'_n = [x_1,x_2,\ldots,x_P]_n$ and $s'_n = [y_1,y_2,\ldots,y_P]_n$ be the row vectors of x-coordinates and y-coordinates, respectively, for the classifier boundary for any patient n, where $n = 1,\ldots,N$. For the calibrated boundary estimation in left ventriculograms using the *identical coefficient* method, we are:

- **Given**: corresponding pairs of ground truth boundaries $\mathbf{R}[2N \times P]$, and the classifier boundaries $\mathbf{Q}$ $[2N \times (P+3)]$, respectively:

$$\mathbf{R} = \begin{pmatrix} g'_1 \\ h_1 \\ \cdots \\ \cdots \\ g'_N \\ h'_N \end{pmatrix} \quad \mathbf{Q} = \begin{pmatrix} r'_1\ \underbrace{1u_{11}\ u_{21}} \\ s'_1\ \underbrace{1v_{11}\ v_{21}} \\ \cdots \\ \cdots \\ r'_N\ \underbrace{1u_{1N}\ u_{2N}} \\ s'_N\ \underbrace{1v_{1N}\ v_{2N}} \end{pmatrix}$$

  where, (u_{11}, v_{11}), (u_{1N}, v_{1N}) and (u_{21}, v_{21}), (u_{2N}, v_{2N}) are the coordinates for the anterior aspect and inferior aspect of the AoV plane of the left ventricle (see Fig. 1, left) from the ground truth boundary, the known information of the starting and ending points of the contour. The last three columns constitute the translation offset effect (unity padding), and the pair u_{1n}, u_{2n} are the x-coordinates for the starting and ending vertex. Similarly, the pair v_{1n}, v_{2n} are the y-coordinates for the starting and ending vertex.
- Let $\mathbf{A}$ $[(P+3) \times P]$ be the unknown regression coefficients matrix.
- The problem is to estimate the coefficient matrix $\mathbf{A}$, to minimise $\|\mathbf{R} - \mathbf{QA}\|^2$. Then for any classifier boundary matrix $\mathbf{Q}$, the calibrated vertices of the boundary are given by $\mathbf{Q}\hat{\mathbf{A}}$, where $\hat{\mathbf{A}}$ is the estimated coefficient matrix.

Note the coefficients that multiply g'_n also multiply h'_n, hence the name *identical coefficient* method. Also, note that the new x-coordinates for the nth boundary depend only upon the old x-coordinates from the nth boundary, and the new y-coordinates from the nth boundary depend only upon the old y-coordinates from the nth boundary.

2.3. Independent Coefficient Method (InCM) for any Frame

As before, let g'_n and h'_n be the row vectors of the x- and y-coordinates for any patient n. Let r'_n and s'_n be the row vectors of the x- and y-coordinates of the classifier boundary. For the calibrated boundary estimation in ventriculograms using the *independent coefficient* method, we are:

- **Given**: corresponding ground truth boundaries $\mathbf{R}$ $[N \times 2P]$, classifier boundaries $\mathbf{Q}$ $[N \times (2P+5)]$, respectively:

$$\mathbf{R} = \begin{pmatrix} g'_1 \; h'_1 \\ \dots \\ g'_N \; h'_N \end{pmatrix} \mathbf{Q} = \begin{pmatrix} \underbrace{r'_1 \; s'_1 \; 1 \; u_{11} \; v_{11} \; u_{21} \; v_{21}}_{} \\ \dots \\ \underbrace{r'_N \; s'_N \; 1 \; u \; _{u1N} \; v_{1N} \; u_{2N} \; v_{2N}}_{} \end{pmatrix}$$

  where (u_{11}, v_{11}), (u_{1N}, v_{1N}) and (u_{21}, v_{21}), (u_{2N}, v_{2N}) are the coordinates of the anterior aspect and inferior aspect of the AoV plane of the left ventricle (see Fig. 1) from the ground truth boundary. The padding explanation is the same as in the previous section, except that the pair (x,y) for the starting and ending points are on the same row; this makes the total number of padding columns five.
- *Let $\mathbf{A}$ $[(2P+5) \times 2P]$ be unknown regression coefficient matrix.
- The problem is to estimate the coefficient matrix $\mathbf{A}$, to minimise $\|\mathbf{R} - \mathbf{QA}\|^2$. Then for any classifier boundary matrix $\mathbf{Q}$, the calibrated vertices of the boundary are given by $\mathbf{Q\hat{A}}$, where $\mathbf{\hat{A}}$ is the estimated coefficient matrix.

Note that the new (x,y)-coordinates of the vertices of each boundary is a *different* linear combination of the old (x,y)-coordinates for the polygon, hence the name *independent coefficient* method. The above two methods are different in the way in which the calibration model is set up. The classifier boundary matrix $\mathbf{Q}$ in IdCM is of the size $\underbrace{2N \times (P+3)}_{}$, while in InCM it is of the size $\underbrace{N \times (2P+5)}_{}$. For IdCM, the number of coefficients estimated in the $\mathbf{\hat{A}}$ matrix is $\underbrace{(P+3) \times P}_{}$. For InCM, the number of coefficients estimated is $\underbrace{(2P+5) \times 2P}_{}$. Thus, the *independent coefficient* method requires around four times the number of coefficients of the *identical coefficient* method to be estimated, and this difference could represent a significant factor for our data size in the ability of the technique to generalise rather than memorise. For this reason, we first optimise both calibration techniques before they undergo the greedy fusion.

2.4. Identical and Independent Optimisation Calibrations: Training & Estimation

Once the data has been interpolated and equally arc sampled, we apply the regression model [20] to find the offline training coefficient matrix $\mathbf{A}(t)$ to minimise

$$\epsilon^2_{ifr} = \|\mathbf{R}(t) - \mathbf{Q}(t)\, \mathbf{\hat{A}}(t)\|^2 \tag{1}$$

Generalising for any frame t of the systolic cycle, minimising $\mathbf{\hat{A}}$ is given by the normal equation

$$\mathbf{\hat{A}}_{tr} = \underbrace{(\mathbf{Q}^T\, \mathbf{Q})^{-1}\, \mathbf{Q}^T \mathbf{R}}_{} \tag{2}$$

The above equation is solved using the Singular Value Decomposition (SVD) (see Press et al [25] and Haralick et al [26]). Given the test set $(\mathbf{Q}_{te})$ or training set $(\mathbf{Q}_{tr})$, we can estimate the calibrated boundary as

$$\underbrace{\mathbf{\hat{R}}_{te} = \mathbf{Q}_{te}\, \mathbf{\hat{A}}_{tr}}_{} \quad \text{and} \quad \underbrace{\mathbf{\hat{R}}_{tr} = \mathbf{Q}_{tr}\, \mathbf{\hat{A}}_{tr}}_{} \tag{3}$$

Thus, the estimated matrices for the IdCM and InCM test sets are

$$\underbrace{\mathbf{\hat{R}}_{id} = \mathbf{Q}_{te}\, \mathbf{\hat{A}}_{id}}_{} \quad \text{and} \quad \underbrace{\mathbf{\hat{R}}_{in} = \mathbf{Q}_{te}\, \mathbf{\hat{A}}_{in}}_{} \tag{4}$$

Note that, if P_2 are the sampled vertices, then $\mathbf{\hat{R}}_{id}$ is of dimension $2N \times P_2$ and $\mathbf{\hat{R}}_{in}$ is of dimension $N \times 2P_2$.

Figure 1 (right) shows the overall boundary calibration system, where the heart of the system is the IdCM or InCM calibrator. The input to the calibrator is the left ventricle boundary data which is represented by polygons of N studies, F frames and $P_1 = 100$ vertices. We used a *cross-validation* procedure for estimating the error of the calibration system. The procedure took a database of N patient studies and partitioned this database into K equal sized subsets. For all the K *choose* L combinations, we trained the system using L subsets, and applied the estimated transformation on the remaining $(K - L)$ subsets. The mean error of the transformed boundary was then computed from these $(K - L)$ subsets coming from all K *choose* L combinations. Our experiments consisted of varying the calibration parameters: N, K, L, P. We chose six different sets of K values (corresponding to each protocol) for training the system. Because of the small number of available patient studies in our database $(N = 291)$, and the large number of parameters (about 200 times N) in the transformation, there was a danger of memorisation rather than generalisation in the estimation of the transformation parameters. Therefore, it was essential that the number of vertices (P in the left ventricle polygon) be carefully chosen. As P decreased, the generalisation improved, but the representation of the true left ventricle shape became worse, thereby causing higher error with respect to the ground truth. As P increased, generalisation was lost but representation of the true left ventricle shape improved. With the other parameters K, L and N fixed, there was an optimal number of boundary vertices P^* balancing the representation error with the memorisation error. Our protocol finds the *optimal number*. The estimated boundary using the two calibration algorithms undergo performance measure using the polyline metric method, as discussed in Section 4.

3. GREEDY ALGORITHM: LV CALIBRATION BY VERTEX

This section first discusses what the greedy algorithm is, its need, followed by a ball-basket method to illustrate its idea. We also discuss the approaches of Cootes et al and ours.

3.1. What is Greedy Algorithm and What is its need?

The greedy algorithm is an iterative algorithm for fusing two given boundaries in such a way that the fused boundary is closer to the ideal boundary compared to the two given boundaries. The idea is to obtain a fused boundary whose error with respect to the ideal boundary is lower than the errors of the two fusing boundaries when compared to the ideal boundary. This fused boundary in each iteration is compared to the ideal boundary, and the process is repeated until no further improvement can be done. The need for the gready algorithm was felt when we observed in Section 1 that there is not enough information in the apical zone and LV wall zones in the grey scale ventriculograms. The above two calibrations help in stretching the initial classifier boundaries closer to the ground truth boundaries [20], thereby removing the bias errors in shape, position and orientation. These two calibration algorithms are sensitive to the number of data vectors (N) and dimensions of the data vectors (P). As a result, one calibration technique performs better than the other for the same frame of the cardiac cycle. This is particularly seen in the apical zone and the papillary muscle zone, where dye is unable to propagate and mix well with the blood. Since we used the left ventricle ground truth boundaries for training, the same database of the left ventricle contours can be used in greedy technique. The two boundary calibration estimates (vertex-by-vertex) from the above calibration algorithms can be fused to produce a boundary closer to the boundary traced by the cardiologist. We select a fixed subset of estimated vertex positions from the IdCM and InCM techniques which, when fused together, minimise the resulting error between the final estimated polygon boundary and the physician-traced left ventricle boundary. We illustrate this idea using a ball-basket method.

3.2. Ball-Basket Method

Consider two baskets (say b_1 and b_2) each containing the same number of balls P_2. Let the colour of the balls of b_1 basket be white, and that of the b_2 basket be black. These balls can be imagined to represent the vertices of the left ventricle boundary. The goal is to fuse these two baskets in such a way that the fused basket (representing the fused boundary) has the greatest resemblance to the ideal basket (ideal boundary). In the first cycle, the algorithm consists of searching for that white ball from basket b_1 which, when combined with the remaining $P_2 - 1$ black balls in basket b_2, will yield a lower error when no ball was transferred. One such greedy cycle is shown in Fig. 4 (left). Finally, one such ball is transferred from basket b_1 to basket b_2 (see Fig. 4, left). Now the greedy cycle is repeated until no more balls are found, which improves the performance of the system (see Fig. 4, right, row #3 (cycle 2), row #4 (cycle 3)). The implementation of the greedy algorithm can be seen in the appendix.

3.3. Comparision Between Cootes and Our Technique

As we have discussed, steps one (two calibration algorithms) and two (iterative fusion algorithm) of our system, we now compare our methodology with Cootes et al's [14] technique, which is as follows: (1) Cootes et al used knowledge of the expected shape combined with information from the areas of the image where good wall evidence could be found to infer missing parts of the left ventricle. In our approach, we also use the Least Squares (independent of each frame) technique to infer the position of the parts of the boundary where there is less contrast or left ventricle information (apex zone of the left ventricle or bottom third region of the left ventricle) by using the knowledge of the top two-thirds region of the left ventricle, where there is good evidence of the left ventricle data points; (2) for the final shape estimation, Cootes et al used a weighted iterative algorithm where weights were proportional to the standard deviation of the shape parameter over the training set. This is more like Weighted Least Squares, where the initial guess was the first stage: the least squares. The termination process of the iteration depended upon the Mahalanobis distance D_m when compared to the constant, say D_{max}. The idea behind the iterative algorithm was to improve the accuracy of the border detection. In our method, the second stage is the greedy algorithm, which fused the two sets of boundaries to yield boundaries closer to the ideal boundary (as traced by the cardiologist). First, we optimised for that P number of vertices which yields the best left ventricle shapes from the *identical* and *independent coefficient* methods. These methods are based on the least squares models. Thereafter, the greedy algorithm can be considered as an iterative process, where we select that vertex of the left ventricle boundary from one estimation technique which, when fused with the other boundary (other estimation technique), will yield an error lower than the error of either of the two methods. The greedy algorithm does not give up unless it has checked all the vertices on the boundary. (3) In our approach, the initial error is not a guess, but the error which is the best of the two existing errors, unlike in Cootes' method, where the initial guess is taken as the mean shape with the addition of a weighted principal axis of the ellipsoid. Cootes et al first find the mean shape, then the eigenvectors of the covariance matrix of the deviation. Thus, any shape is approximated as

$$\mathbf{x} = \bar{\mathbf{x}} + \mathbf{Pb} \tag{5}$$

where $\bar{\mathbf{x}}$ is the mean shape, $\mathbf{P}$ is the matrix of the eigenvectors and $\mathbf{b}$ is the vector of the weight matrix. In the greedy method, if the fusing boundary vertices have a large error compared to the the ideal boundary (traced by the cardiologist), then the greedy algorithm rapidly picks the vertices. The nice thing about the greedy algorithm is that we do not have to repeat the *identical coefficient* method and the *independent coefficient* method calibrations again. Once these runs are over, we then just choose the vertices (switching columns between two boundary data matrices) in such a way that in each greedy cycle, we are heading

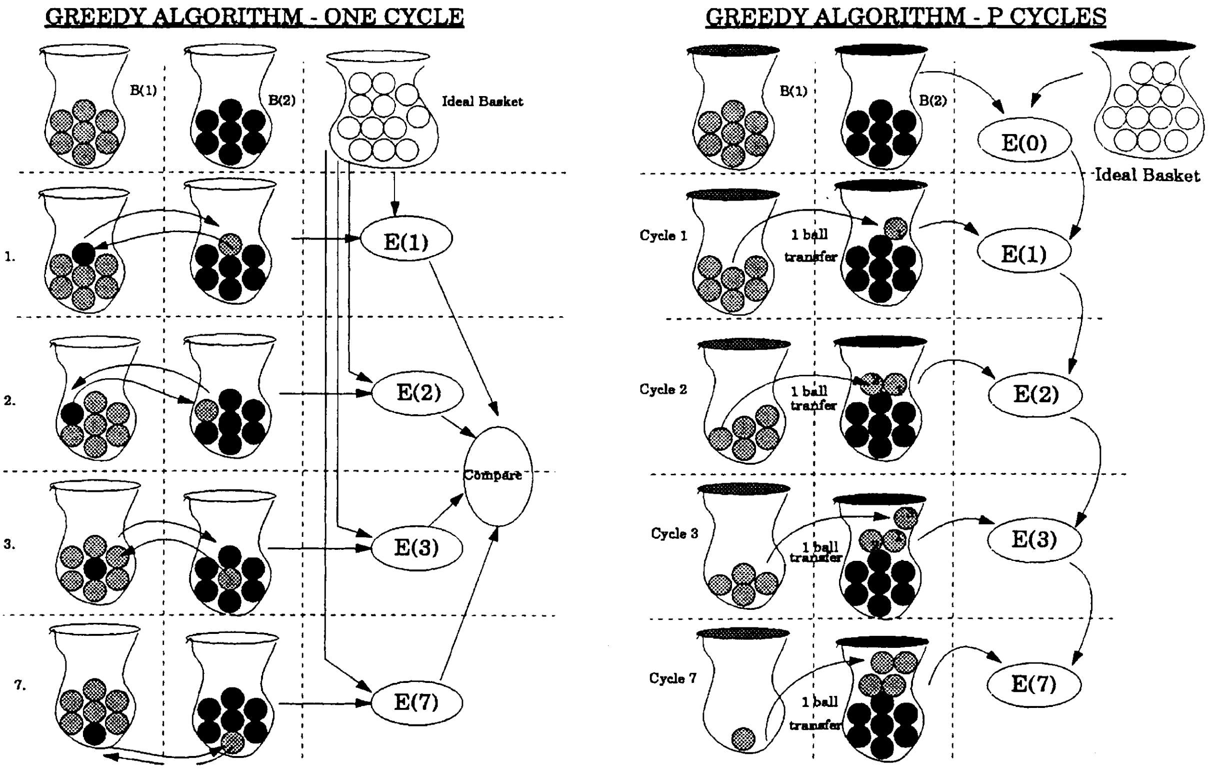


Fig. 4. Left: one greedy cycle where best vertex (ball) is selected out of basket b_1. E's denote the errors when compared with ideal basket. Right: the figure shows basket number b_2 filling up after each greedy cycle.

closer to the ground truth boundaries until all the vertices are checked. (4) Another similarity between Cootes' and our method is that both methods use the point model, which means that the starting analysis of the boundaries are the vertices or points. The only difference is in the dimension of the matrices. Finally, Cootes et al use the Mahalanobis distance for performance, while we use the polyline distance metric, as discussed in the next section.

4. POLYLINE DISTANCE MEASURE AND PERFORMANCE TERMS

Several methods have been developed for measuring the left ventricle wall motion (see details in Klausner et al [22]). Sheehan et al [24] developed a centreline method for quantitative assessment of the ventricular boundary. Sheehan measured the motion along 100 chords constructed perpendicular to a centre line drawn midway between the end diastole and end systole left ventricle contours. The centre line method was developed to find the extent of local left ventricle wall motion. The algorithm consists of the following principle. If the two left ventricle polygons are end diastole and end systole boundaries, the end diastole having a larger number of points on it, then first linearly interpolate the larger contour to get 200 points, and then for each tuple of three points on this contour, draw a perpendicular to the tangent of the circle passing through these three points. The centres of this *perpendicular* constitutes the centre line. This distance is a function of the area swiped between two left ventricle boundaries. The performance of the training algorithm is measured by the error of closeness between the estimated boundary and the ideal boundary, as traced the cardiologist. The error of closeness is measured using a stable method based on vector calculus called the *polyline metric*, as derived below.

The polyline metric is actually based on the ratio of the average area between two polygons *to* the average perimeter of the two polygons. Our assumption prior to the polyline distance error computation is that the polygons have equal perimeters, and the vertices are equally spaced along the perimeter, i.e. the two arc intervals on the two contours are the same. Using this assumption, we mathematically show that the ratio of the average area between polygons to their perimeters is actually the average polyline distance error between the two polygons.

Let the two polygons be B_1 and B_2, consisting of total points P_1 and P_2 having interval lengths of l_1 and l_2, respectively. This is shown in Fig. 5. Let the perpendicular distances from each of the vertices (of polygon B_1) to the opposite interval sides of polygon B_2 be d_n, where, $1 \leq n \leq P_1$. Similarly, let d'_m be the perpendicular distances from each of the vertices (of polygon B_2) to the opposite interval sides of polygon B_1, where, $1 \leq m \leq P_2$. Thus, the area of the triangle with height d_n and base l_2 is $\frac{1}{2}l_2d_n$. Similarly, the area of the triangle with height d'_m and base

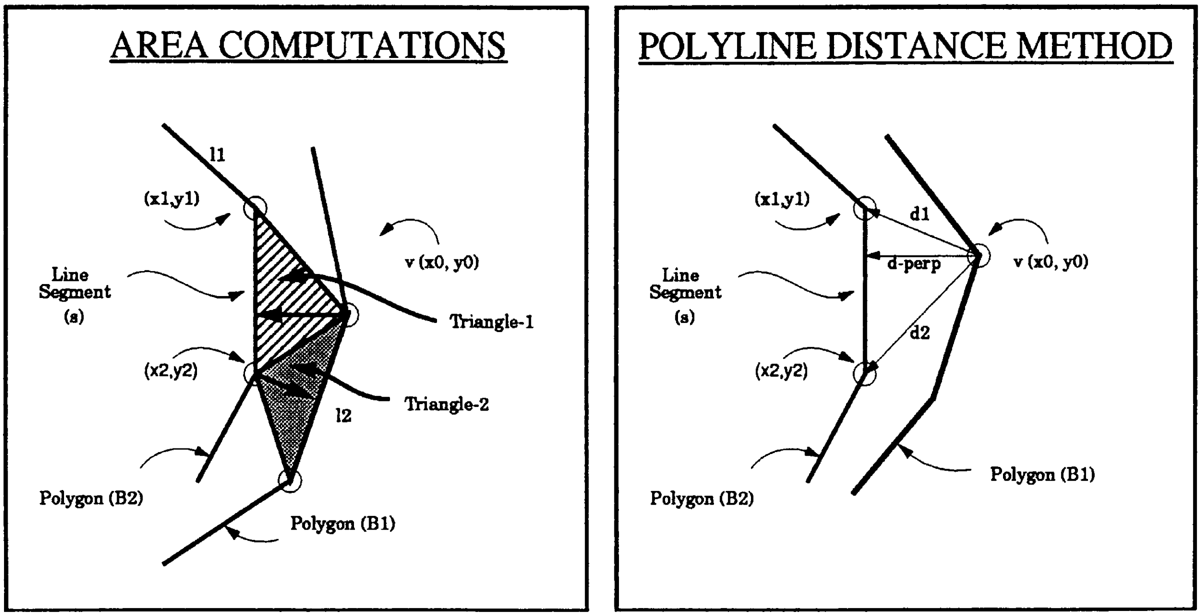


Fig. 5. Left: figure showing the area of triangles between the two polygons. Right: geometry of the polyline distance computation.

l_1 is $\frac{1}{2}l_1d'_m$. The total area $\bar{A}$ between two polygons is derived by computing the sum of the area of all the triangles whose base lies on the polygon B_1 and the sum of the area of all the triangles whose bases lies on the polygon B_2, and is given as

$$\bar{A} = \sum_{n=1}^{P_1} \tfrac{1}{2} l_2 d_n + \sum_{m=1}^{P_2} \tfrac{1}{2} l_1 d'_m \tag{6}$$

Similarly, we can compute the average perimeter of the two polygons as

$$\bar{P} = \frac{l_2P_1 + l_1P_2}{2} \tag{7}$$

Taking the ratio of Eq. (6) to Eq. (7), we have

$$\frac{\bar{A}}{\bar{P}} = \frac{\sum_{n=1}^{P_1} \frac{1}{2}l_2d_n + \sum_{m=1}^{P_2} \frac{1}{2}l_1d'_m}{\frac{l_2P_1 + l_1P_2}{2}} \tag{8}$$

Using the above assumption, $l_1 \approx l_2$ Eq. (8) reduces to

$$\frac{\bar{A}}{\bar{P}} = \frac{\sum_{n=1}^{P_1} d_n + \sum_{m=1}^{P_2} d'_m}{P_1 + P_2} \tag{9}$$

We show that this is what the polyline distance method computes, which will be shown to be $D_s(B_1{:}B_2)$. The polyline distance $D_s(B_1{:}B_2)$ between two polygons representing boundary B_1 and B_2 is symmetrically defined as the average distance between a vertex of one polygon and the boundary of the other polygon. To define this measure precisely, we first need to define a distance $d(v,s)$ between a point v and a line segment s. The distance $d(v,s)$ between a point v having coordinates (x_o,y_o), and a line segment having end points (x_1,y_1) and (x_2,y_2) is

$$d(v,s) = \begin{cases} \min\{d_1,d_2\}, & \text{if } \lambda < 0,\ \lambda > 1 \\ |d^{\perp}|; & \text{if } 0 \leq \lambda \leq 1 \end{cases} \tag{10}$$

where

$$d_1 = \sqrt{(x_0 - x_1)^2 + (y_0 - y_1)^2}$$

$$d_2 = \sqrt{(x_0 - x_2)^2 + (y_0 - y_2)^2}$$

$$\lambda = \frac{(y_2 - y_1)(y_0 - y_1) + (x_2 - x_1)\,(x_0 - x_1)}{(x_2 - x_1)^2 + (y_2 - y_1)^2}$$

$$d^{\perp} = \frac{(y_2 - y_1)(x_1 - x_0) + (x_2 - x_1)(y_0 - y_1)}{\sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}} \tag{11}$$

Note that d_1 and d_2 are the distances from vertex v to the end points of the segment s. λ is the distance along the vector of the segment s, while $d^{\perp}$ is the perpendicular distance along the vector orthogonal to the segment s. The polyline distance $d_b(v,B_2)$ from vertex v to the boundary B_2 is defined by

$$d_b(v,B_2) = \min_{s \in sides\ B_2} d(v,s) \tag{12}$$

This step is a confirmation that we are choosing the closest segment on B_2 from the vertex v. The distance $d_{vb}(B_1,B_2)$ between the vertices of polygon B_1 and the sides of polygon B_2 is defined as the sum of the distances from the vertices of the polygon B_1 to the closest side of B_2:

$$d_{vb}(B_1,B_2) = \sum_{v \in vertices\ B_1} d(v,B_2) \tag{13}$$

Reversing the computation from B_2 to B_1, we can similarly compute $d_{vb}(B_2,B_1)$. Using Eq. (13), the polyline distance between polygons, $D_s(B_1{:}B_2)$ is defined by

$$D_s(B_1{:}B_2) = \frac{d_{vb}(B_1,B_2) + d_{vb}(B_2,B_1)}{(\#vertices \in B_1 + \#vertices \in B_2)} \tag{14}$$

This equation is basically the same as Eq. (9), where the numerator is the sum of all the perpendicular distances for both the polygons, while the denominator is the sum of the total number of vertices on the two polygons. This mathematical expression $D_s(B_1{:}B_2)$ signifies how much one left ventricle wall is away from the other, and is very helpful in wall motion estimation, for example, wall motion measurement between the end diastole and end systole left ventricle boundaries. We use this expression in computing how much the estimated left ventricle boundaries are away from the ideal boundaries, and it represents an average error measure.

The IdCM and InCM optimisation algorithms are based on the vertex-to-vertex distance error, while the performance measure of the estimated boundaries is based on the slightly closer polyline distance measure. The polyline distance measure basically finds the average distance between the vertex to a polygon. Computationally, they are similar measures, but from the cardiologist's point of view (as tested in our Cardiovascular research laboratory), the polyline measurement tool does not consider the algorithm or technique from which the estimated boundaries have come, e.g. the estimated boundaries could come from either the pixel classifier scheme, the regression scheme or a neural network scheme, or from any segmentation technique. Since the earlier method developed by Sheehan [24] computed the minimum distance as the *perpendicular* distance between the two contours (left ventricle polygon), we use similar grounds to estimate the short distance from a vertex to the opposite polygon. We saw that the polyline distances were superior to current methods. These advantages are:

1. The method is very stable and computes the error based on the geometry of the triangles and vector calculus, which is the ratio of the average area to the average perimeter, as shown in the mathematical derivation.
2. The polyline distance error has an added advantage if there are cups or cusps in the polyline, as the algorithm can spot them easily. Since the algorithm looks for those perpendiculars whose λ lies between 0 and 1 to satisfy the condition of closeness, there is no possibility of any error.
3. Since, prior to polyline error computation, the curves (left ventricle polygons) undergo equal arc sampling and interpolation, there exists one-to-one correspondence between the estimated and ground truth vertices of the curves.
4. This is a non-iterative computation and thus saves time, unlike Bolson and Sheehan's method [23,24], where one has to do the computations again to smooth the curves and improve the accuracy.
5. In Bolson and Sheehan's [24] method, one has to compute the tangents by fitting the circle through three points. The circle fitting algorithm is very sensitive to the number of points taken on either side of the vertex (see Haralick et al [26]). In the polyline method there is no fitting involved. Also, Suri et al [18] showed that the polyline method is more accurate than the center line method to the third decimal accuracy.
6. Another advantage of the polyline algorithm is that it can easily find out whether the distance computed for every vertex is positive or negative, positive when going from one vertex to the opposite polygon, and *vice versa*. This is very useful in finding out if part of the curve is inside or outside.
7. The polyline algorithm is in conjunction with the swapping mechanism, which make the process symmetric. So the average statistics gives a better estimate of the mean error when computed from one polygon to another, and vice versa.
8. The method has an advantage in that the correct bias errors at every vertex are computable. Since not only is the closest perpendicular computed, but also the coordinate position and its bias relative to the vertex on the opposite polygon, we can thus use this distance to correct the bias errors in calibration algorithms by making the optimisation algorithm a weighted least squares algorithm, where the weights are inversely proportional to the polyline bias errors. We are developing a technique by which we can use these errors and build a constrained optimisation problem.
9. One of the main advantages of the polyline distance measure is the coherence with the ejection fraction computation. For example, if the ES left ventricle shape is totally convex and inside the ED left ventricle shape, then the ejection fraction number truly reflects the polyline distance error. As 95% of the ES left ventricle lies inside the ED left ventricle, the polyline distance error will reflect very good approximations for sweeping the area between left ventricle shapes for volume computations. Thus, the primary advantage of contour extraction and quantification can be better studied, and the cardiologist can establish better clinical relevance. The polyline disance measure and optimisation algorithms are computing a measure which is computationally similar, and any difference in terms of error can be accepted by cardiologists, bearing in mind the fact that the polyline distance method not only computes a similar mean error measure to the center line [18] technique to the third decimal, but also has the advantages of superiority and stability compared to the other methods.

4.1. Mean Error ($e^{poly}{}_{NFP}$) or Measure of Agreement

The performance of the calibration algorithms is evaluated by computing the boundary error on the test data set ($\mathbf{Q}_{te}$). Using the definition of the polyline distance between two polygons, we can now compute the mean error of the overall calibration system. It is denoted by e^{poly}_{NFP}, and defined by

$$e^{poly}_{NFP} = \frac{\sum_{t=1}^{F} \sum_{n=1}^{N} D_s(G_{nt},C_{nt})}{F \times N} \tag{15}$$

where $D_s(\mathrm{G}_{nt},\mathrm{C}_{nt})$ is the polyline distance between the ground truth G_{nt} and calibrated polygons C_{nt} for patient study n and frame number t. This term is very significant, as it represents how far the estimated boundary and the ideal boundary are to each other on average over the entire population of patient studies, frames and vertices. We will use this term to analyse the estimated data and the training performance of the cross-validation procedure. Equation (15) is very relevant in the optimisation technique. For each set of $P = P_2$ number of points, e^{poly}_{NFP} is computed and the operating point is estimated, which corresponds to the best fitted shape. Using the definition of the polyline distance between two polygons, the standard deviation can be computed as

$$\sigma^{poly}_{NFP} = \sqrt{\frac{\sum_{t=1}^{F} \sum_{n=1}^{N} \{\sum_{v \in \ vertices\ \mathrm{G}_{nt}} (d_b(v,\mathrm{C}_{nt}) - e^{poly}_{NFP})^2 + \sum_{v \in \ vertices\ \mathrm{C}_{nt}} (d_b(v,\mathrm{G}_{nt}) - e^{poly}_{NFP})^2\}}{N \times F \times (\#vertices \in B_1 + \#vertices \in B_2)}} \tag{16}$$

4.2. Error per Vertex and Error per Arc Length

Using the polyline distance formulae, we can compute the Error Per Vertex (EPV) from one polygon (ground truth) to another polygon (calibrated). This is defined as the mean error for a vertex v over all the patients and all the frames. The *error per vertex* for a fixed vertex v when computed between ground truth and calibrated boundary is defined by

$$e_v^{\mathrm{GC}} = \frac{\sum_{t=1}^{F} \sum_{n=1}^{N} d_b(v,\mathrm{G}_{nt})}{F \times N} \tag{17}$$

Similarly, we can compute the *error per vertex* between calibrated and ground truth using Eq. (12). Error Per Arc Length (EPAL) is computed in the following way: for the values e_v^{GC} where $v = 1, 2, 3,\ldots P_1$, we construct a curve f^{GC} defined on the interval [0,1] which takes the value e_v^{GC} at point x, which is the *normalised arc length* to vertex v, and whose in-between values are defined by linear interpolation. We compute the curve f^{CG} between calibrated boundary and ground truth boundary in a similar way. We then add these two curves algebraically to yield the final *error per arc length*, given as $f = \frac{f^{\mathrm{GC}} + f^{\mathrm{CG}}}{2}$. Note that the EPV is a very useful term, as it gives the error between the estimated vertex and the ideal vertex over the entire population. This is particularly useful in the apical zone of the left ventricle. EPAL is a better representation of EPV, since it is normalised over the entire contour. The EPAL is equally significant in the apical zone of the left ventricle, as it tells us how much of an improvement the training algorithms gave when removing the bias errors of the classifier boundaries around the entire contour.

5. DATA ANALYSIS USING IdCM, InCM AND GREEDY METHOD

Our database population consisted of 291 patient studies, out of which 135 studies had acute myocardial infarction, with the top 50% in quality on a scale of 0 to 10, where 0 is rejected and 10 is considered excellent. The number of subjects with acute infarction who then underwent follow-up studies over the course of one year was 35. The number of subjects from Japan who were normal and had a diagnostic cardiac catheterisation was 27. The Japanese patient's left ventricle images represented the top 50% in quality; the remainder of the 94 studies were from the Catheterisation Laboratory at the University of Washington Medical Center, Seattle, Washington. These studies represented a top 30% in quality. We use this knowledge to build our training system. The performance of the system can be judged by evaluating the error measures on the test data $\mathbf{Q}_{te}$ set. This section discusses the performance of IdCM, InCM and the greedy methods. All our performance is with respect to the original ground truth boundaries having $P_1 = 100$ vertices.

5.1. Data Analysis 1: Vertex Optimisation for Cross-Validation

We find the mean error as a function of the number of polygon boundary vertices on the left ventricle contour. The optimisation curve and operating point is shown in Fig. 6. The database consists of $N = 291$ patient studies and the selected number of partitions $K = 145$. We now vary the number of vertices P_2 on the left ventricle polygon, varying it from 10 vertices to 90 vertices with five vertex increments. If $L = 144$ are the training sets, then for each combination there are $K - L$ test set boundaries on which the error is computed. We choose the number of vertices P_2 to minimise the error on the test set. Since there are ${}^{K}C_L = 145$ trials, each trial has $(K - L)$ subsets, each subset consists of N/K patients (in a protocol, if N/K is not a perfect division, then for the last trial in ${}^{K}C_L$ combinations, we have $(K - L + r_p)$ patients as the testing set, where r_p is a remainder number of N/K), and each patient consists of P_2 vertices and $F = 2$ frames. We thus get the total number of points as: $F \times {}^{K}C_L \times (K - L) \times N/K \times P_2$, resulting in: $N \times F \times P_2 \times (K - 1)!/((K - L - 1)!L!)$, points for each (N,K,L,P_2) tuple. Since we are computing the polyline

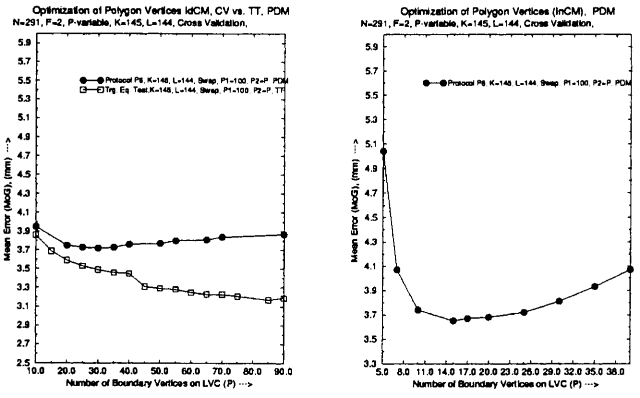


Fig. 6. Vertex optimisation using the polyline distance metric. Left: IdCM (identical coefficient method), cross validation vs. training equals testing (TT). Right: InCM (Independent Coefficient method). Note the IdCM operating point is 30 vertices and InCM operating point is 15 vertices.

distances, the number of operations is $N \times F \times P_1^2 \times P_2 \times (K-1)!/((K-L-1)!\ L!)$. We see from the plot Fig. 6 that the *optimal* number of vertices in InCM is about *half* the optimal number of vertices in IdCM, the reason being that the number of coefficients that have to be estimated in InCM is about *four* times the number of coefficients that have to be estimated in IdCM.

5.2. Data Analysis 2, 3: Cumulative Distribution of $\left(\frac{ED+ES}{2}\right)$ Errors and Error Per Arc Length along Left Ventricle Contour

We show here the cumulative distribution of end frame errors (*ED*+*ES*/2) from both the calibration methods using IdCM and InCM, as shown in Fig. 7. Figure 8 demonstrates the mean EPAL along the LVC. The abscissa shows the length of the arc starting from AAV. The ordinate shows the error at each vertex in millimeters. As seen in the plot, the mean EPV is largest near the middle of the normalised arc length, which is close to the apex of the left ventricle. Thus the error is maximum in the apex region.

We see that the greedy algorithm does *best* in the apex zone compared to the IdCM and InCM methods. The EPV in Fig. 8 shows that in the end diastole frame, the apex zone error is reduced by 8.5 mm (from 12.5 mm to about 4 mm), while in the end systol frame, the apex zone error is reduced by 3 mm (from 9 mm to 6 mm). The corresponding mean error over the ED and ES frames of the pixel-based boundaries was 6.4 mm, which is reduced to 3.8 mm in IdCM and 3.5 mm in the greedy scheme. As per our assumption, the error is least at the end points of the LVC, since the AoV plane is known, thus the EPV curve drops at both ends. Our results show that 81% of the patient boundaries had a mean (*ED*+*ES*/2) less than 4.0 mm using the greedy calibration technique.

5.3. Data Analysis 4: ED and ES Errors vs. InCM Pool Vertices

Here we show the effect of the greedy calibration scheme. Figure 9 shows the drop in end diastole and end systole frame errors when the IdCM pool vertices are transferred to the InCM pool. This is implemented using the greedy *do-while loop*, where some columns (or vertices) of the IdCM matrix $\hat{\mathbf{R}}_{id}$ are replaced by corresponding columns (or vertices) of the InCM matrix, $\hat{\mathbf{R}}_{in}$. Figure 9 (right) shows that the greedy algorithm reduces the error by 0.3 mm over IdCM. We also observe that the best number of vertices for IdCM is 30, while for InCM it is 15. The best performance over all the three techniques is by the greedy algorithm with the number of vertices being 30. In the greedy calibration technique, the error does not rise very sharply after 30 vertices, but rises gradually by 100th of a millimeter from 30 vertices to 40 vertices. The input and output of the boundary estimation system for the IdCM, InCM and greedy techniques are shown Figs 10–12.

6. RELIABILITY ALGORITHM AND VALIDATION: REJECTION FOR LARGE BOUNDARY DEVIATIONS

In this section we develop a general and automatic validation technique that can detect the left ventricle boundaries whose mean end frame boundary errors (*ED*+*ES*/2) are above a given threshold, R_{th}. This validation scheme has the following features and advantages:

1. It determines those left ventricle boundaries from the database (which could be coming from any source) whose (*ED*+*ES*/2) error is above a given threshold error.
2. The scheme provides feedback to the boundary cali-

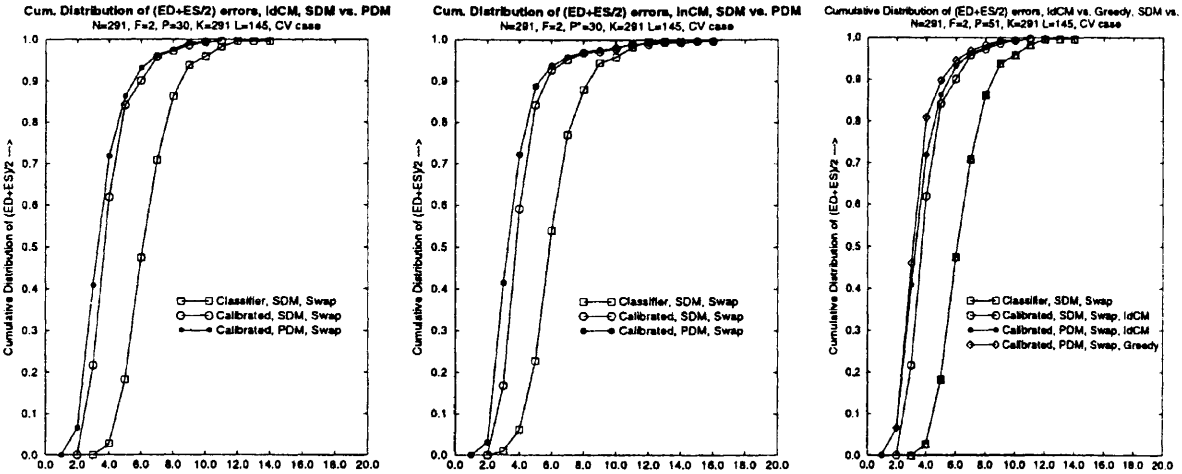


Fig. 7. Cumulative distribution vs. mean error of (*ED* + *ES*/2) errors. Left: identical coefficient method. Middle: independent coefficient method. Right: greedy vs. IdCM. The curves show that 80% of patient estimated boundaries have an error ≤4 mm in IdCM, while 72% of the patients have an error ≤4 mm in InCM, and 81% of patients have an error ≤4 mm in the greedy method. Partition protocol parameters: $N = 291$, $F = 2$, $K = 145$, $L = 144$, $P_1 = 100$, $P_2 = 30$.

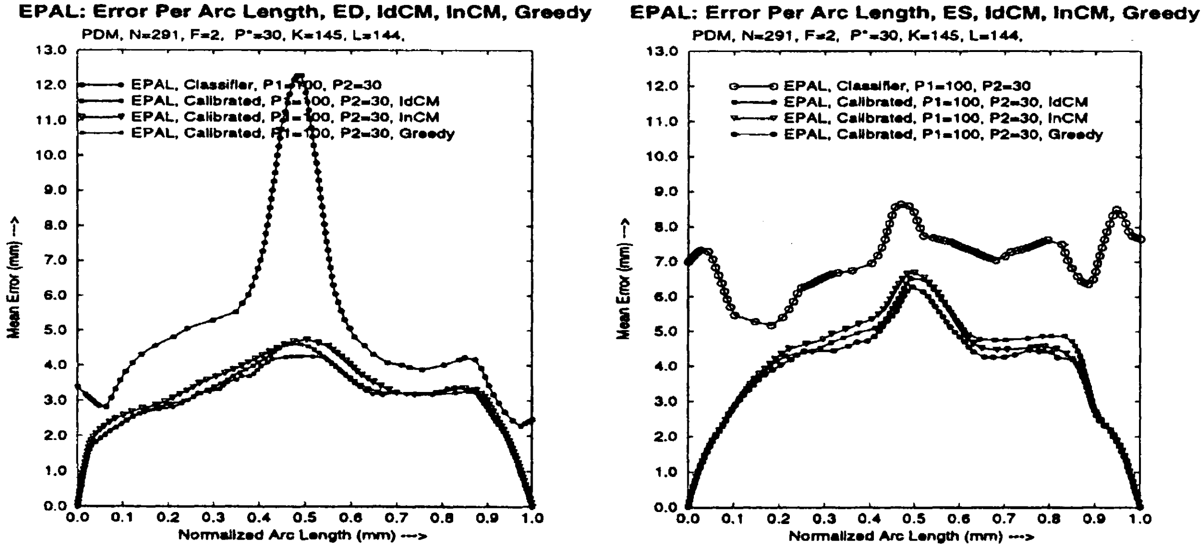


Fig. 8. Mean Error Per Arc Length (EPAL) using the polyline distance method, where we superimpose four curves, the initial raw (perturbed), identical coefficient method, independent coefficient method and greedy calibration method. Left: ED frame. Right: ES frame. The greedy method does the best out of the three calibration techniques in both the ED and ES frames. The greedy method also does better in the apex zone (where the dye was unable to reach) compared to all the other vertices. In the ED frame, the apex error reduces by 8.5 mm and in the ES frame, the apex error reduces by 3 mm.

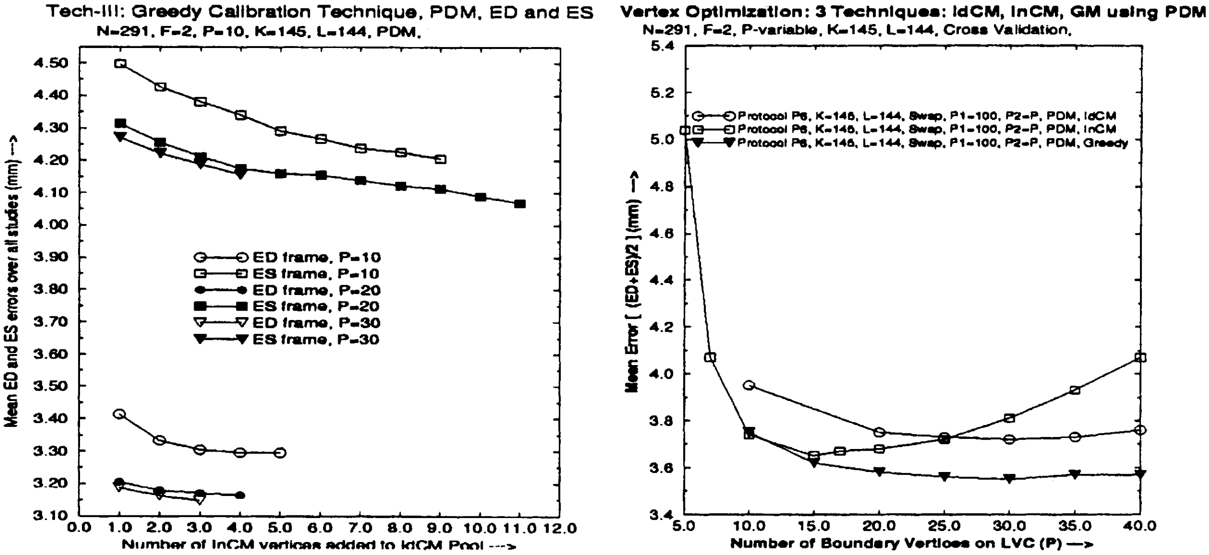
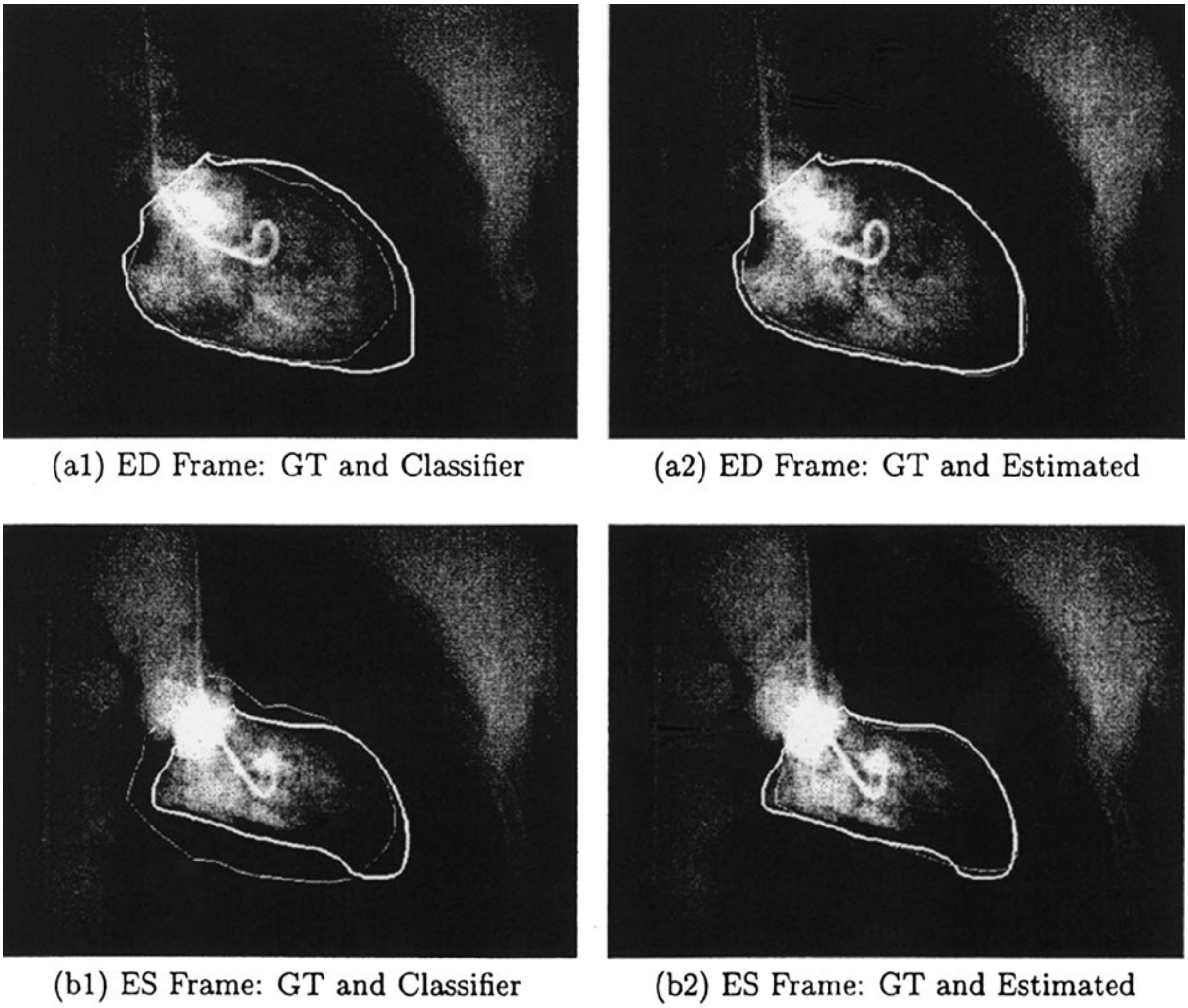


Fig. 9. Greedy performance. Left: plot showing the reduction in the error for ED and ES frames when some vertices are calibrated using IdCM and others using InCM. With the increase InCM pool, the error drops. Right: comparison of three calibration techniques, the greedy method does the best. Partition protocol parameters: $N = 291$, $F = 2$, $K = 145$, $L = 144$, $P_1 = 100$, $P_2 = 30$. Mean error for IdCM = 3.8 mm, InCM = 3.9 mm and Greedy = 3.5 mm. So the greedy method improves by 0.3 mm over the IdCM method.

bration system so that the system knows which left ventricle boundaries can be rejected.

3. It estimates the overall performance of the boundary calibration system without taking the rejected boundaries into consideration.
4. It provides a check for consistency and reliability of the output estimation algorithms (for example: Classification algorithm, IdCM, InCM or Greedy).

For this test we need three inputs: the left ventricle boundary coordinates (*x*,*y*); the grey scale left ventriculograms; and the binary indicator for the left ventricle region: 1 for inside the left ventricle region and 0 for outside the left



(a1) ED Frame: GT and Classifier

(a2) ED Frame: GT and Estimated

(b1) ES Frame: GT and Classifier

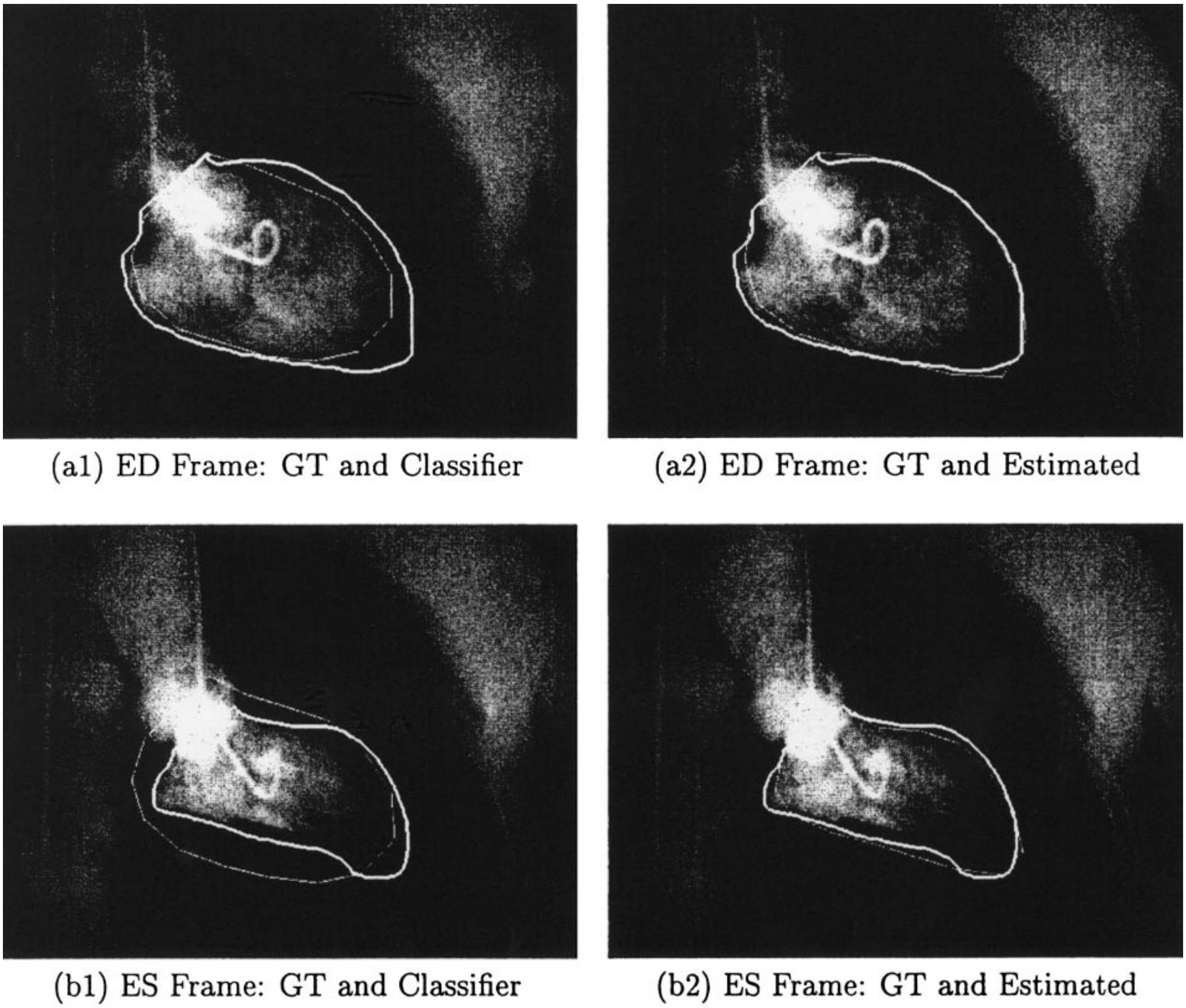
(b2) ES Frame: GT and Estimated

Fig. 10. Classifier vs. estimated boundaries with grey scale in the background using the Identical Calibration Method (IdCM). Quartile 1, Thick LV contour – Ground Truth, Thin LV contour-Classifier, and Estimated, Upper: (a1) Uncalibrated ED frame boundary with ground truth. (a2) Calibrated ED frame boundary with ground truth. Bottom: (b1) Uncalibrated ES frame boundary with ground truth. (b2) Calibrated ES frame boundary with ground truth. Calibration Parameters: $N = 291$, $K = 145$, $L = 144$, $F = 2$, $P_1 = 100$, $P_2 = 30$, Mean end frame error $(\frac{ED + ES}{2}) = 1.30$ mm, Mean error $(e^{poly}_{NFP}) = 3.7$ mm.

ventricle region. This boundary rejection scheme is based on the grey scale information near the boundary of the left ventricle. Using a mutually exclusive window of a fixed size centred on the left ventricle boundary vertex and along the left ventricle contour, we compute the mean grey level value for part of the window which is inside the left ventricle region and partof the window which is outside the left ventricle region (see Fig. 13, left). We then associate this difference in the mean grey scale intensities (also called contrast values) to the corresponding vertex of the observed left ventricle boundary. Since we know the observed boundary errors (*ED*+*ES*/2) for patient study *n* estimated from any technique, we can regress these contrast values against the observed errors to compute the rejection training coefficients. These training coefficients can then be used to find the predicted errors on the test Contrast Boundary Data (CBD). The predicted errors which are above the threshold correspond to boundary delineation, which are to be rejected. This rejection scheme is a reliability test, because it helps to determine how reliable the estimated boundaries are. In Section 6.1, we give the mathematical statement for computing the predicted errors. Section 6.2 discusses the reliability test algorithm and training system, and Section 6.3 gives the mathematical formulae for computing the probability of a false alarm, a mis-detection, and the mean predicted errors when *n* patients are rejected from the database. Finally, we discuss the resulting reliability curves from our experiments.

6.1. Problem Statement: Reliability Equation

Boundary contrast data refers to the grey scale data generated along the left ventricle boundary by superimposing the left ventricle boundary over the grey scale left ventriculograms.



(a1) ED Frame: GT and Classifier

(a2) ED Frame: GT and Estimated

(b1) ES Frame: GT and Classifier

(b2) ES Frame: GT and Estimated

Fig. 11. Classifier vs. estimated boundaries with grey scale in the background using the Independent Coefficient Method (InCM). Quartile 1, Thick VL contour – Ground Truth, Thin LV contour-Classifier, and Estimated, Upper: (a1) Uncalibrated ED frame boundary with ground truth. (a2) Calibrated ED frame boundary with ground truth. Bottom: (b1) Uncalibrated ES frame boundary with ground truth. (b2) Calibrated ES frame boundary with ground truth. Calibation Parameters: $N = 291$, $K = 145$, $L = 144$, $F = 2$, $P = 100$, $P_2^* = 15$, Mean end frame error $(\frac{ED + ES}{2})$ = 1.57 mm, Mean error $(e_{NFP}^{poly}) = 3.65$ mm.

At each chosen vertex of the LVC, there is a corresponding contrast value. The contrast value for a given window is the difference between the mean grey scale intensities inside and outside the window. This superimposed boundary can be from any given boundary estimation algorithm. *Observed errors* are the mean end frame boundary errors $(ED + ES/2)$ for each patient study n.

We now present the mathematical statement for the estimation of predicted errors for the left ventricle boundaries, given the boundary contrast data matrix $\mathbf{C}$ and observed end frame error $(ED + ES/2)$ vector $\mathbf{e}$. The predicted errors are used for spotting the left ventricle boundaries that are above the given error threshold.

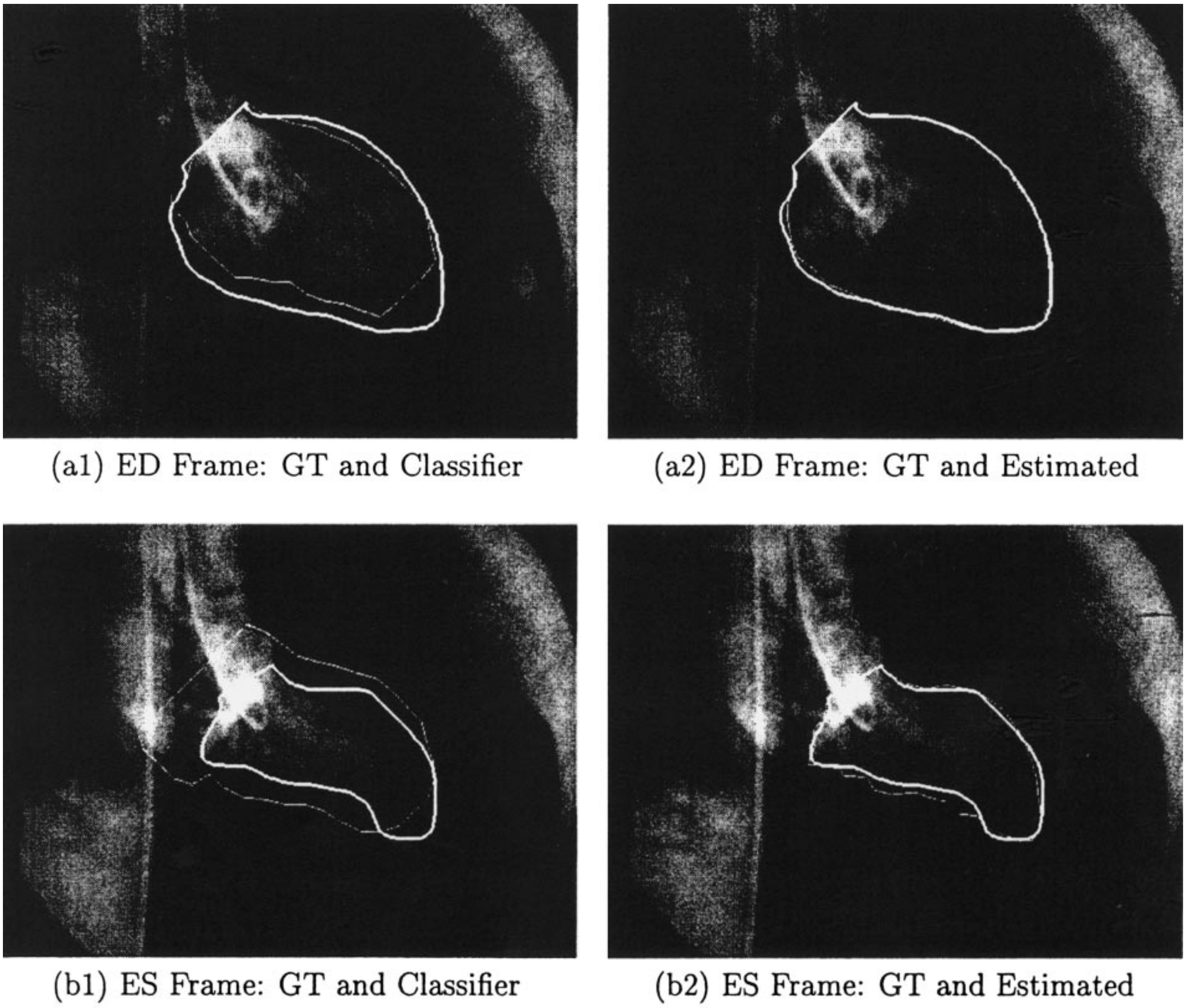
Let $c'_n = [c_1, \ldots, c_P]$ be the row vector of contrast values of dimension P for patient study n, where, $n = 1, \ldots, N$. Let $\mathbf{e} = [e_1, \ldots, e_N]$ be the vector of the observed end frame errors, $(ED + ES/2)$ for N patient studies. For the predicted errors of the patient study n, we are:

- **Given**: corresponding pairs of contrast data matrix $\mathbf{C}[N \times (P + 3)]$, and the observed errors $\mathbf{e}[N \times 1]$, respectively, as:

$$\mathbf{C}^{N\times(P+3)} = \begin{pmatrix} c'_1 \underbrace{1\ s_1\ m_1} \\ \cdots \\ c'_N \underbrace{1\ s_N\ m_N} \end{pmatrix} \qquad \mathbf{e}^{N\times 1} = \begin{pmatrix} e_1 \\ \cdots \\ e_N \end{pmatrix}$$

  where s_n, m_n is the standard deviation and mean of the P contrast values for the patient study n along the LVC.

- Let $\mathbf{a}[N \times 1]$ be the vector of unknown regression coefficients.



(a1) ED Frame: GT and Classifier

(a2) ED Frame: GT and Estimated

(b1) ES Frame: GT and Classifier

(b2) ES Frame: GT and Estimated

Fig. 12. Classifier vs. estimated boundaries with grey scale in the background using the Greedy Calibration Method. Quartile 1, Thick LV contour – Ground Truth, Thin LV contour-Classifier, and Estimated, Upper: (a1) Uncalibrated ED frame boundary with ground truth. (a2) Calibrated ED frame boundary with ground truth. Bottom: (b1) Uncalibrated ES frame boundary with ground truth. (b2) Calibrated ES frame boundary with ground truth. Calibation Parameters: $N = 291$, $K = 145$, $L = 144$, $F = 2$, $P_1 = 100$, $P_2 = 35$, Mean end frame error $(\frac{ED + ES}{2})$ = 1.16 mm, Mean error $(e^{poly}_{NFP}) = 3.5$ mm.

- The problem is to estimate the coefficient vector **a**, to minimise $\|\mathbf{e} - \mathbf{C}\mathbf{a}\|^2$. Then for any boundary contrast data matrix **C**, the predicted error for the boundary is given as: $\mathbf{C}\hat{\mathbf{a}}$, where $\hat{\mathbf{a}}$ is the estimated coefficients.

6.2. Reliability Algorithm: C-e Relation

The following are algorithm steps for estimating the predicted errors which form the basis for the reliability of the boundaries estimated from any boundary estimation algorithm:

1. Boundary contrast matrix generation (**C**): the contrast matrix can be generated by first superimposing the left ventricle boundaries over the grey scale images (LVG). We use two methods for generating the contrast data. The mathematical statements for expressing the contrast value at a vertex i are given as follows: let g_p be the grey scale value for pixel p. Let I and O be the sets which contain the pixels inside and outside the mutually exclusive moving window. Let G_I and G_O be the sum of all the grey scale intensities for pixels which are inside and outside the window, given by:

$$G_I = \sum_{p \in I} g_p \quad \text{and} \quad G_O = \sum_{p \in O} g_p \tag{18}$$

Let f_I and f_O be the cardinality of the sets I and O. Using the above notation, we give the expression for the contrast value at vertex i using the two methods:

$$c_i = \frac{(G_I - G_O)_i}{(f_I + f_O)} \quad \text{and} \quad c_i = \left(\frac{G_I}{f_I}\right)_i - \left(\frac{G_O}{f_O}\right)_i \tag{19}$$

Note that the difference between the above equations lies in the way in which the sums of the grey scale intensities G_I and G_O are subtracted in the two cases. In method I,

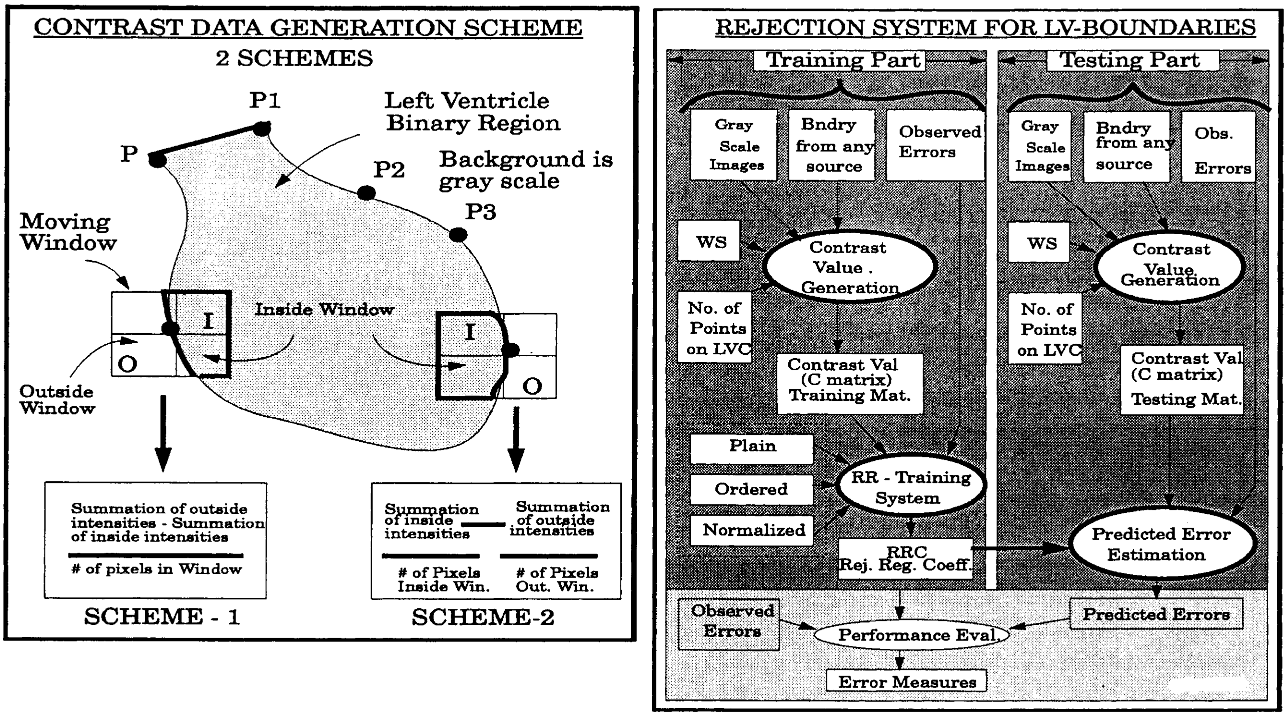


Fig. 13. Right: Overall system for reliability and validation test. It has two parts: the first is the generation of the training rejection coefficients from the contrast data set; the second consists of applying the training coefficients on the test contrast data to generate the predicted errors. These predicted errors show the performance evaluation of the system. WS is the window size which moves along the LVC. We use three statistical techniques for estimating the rejection regression coefficients (RRC), depending upon the layout of the contrast data matrix (**C**). Left: Contrast data generation process showing two schemes. (i) Non-separate contrast data; and (ii) separate contrast data.

we compute the grey scale difference between the total grey scale values inside and outside the window, and then divide the difference by the total number of pixels in the window. In the second case, we find the mean values for inside and outside separately, and then subtract it. The former is called *point-wise contrast* data, as the contrast value for a vertex i is computed by taking all of the pixels for that point into account, while the latter is called *region-wise contrast* data, as the contrast value at a vertex i is computed by subtracting two different regions of the window (see Fig. 13, left).

We also compute the mean and standard deviation of the contrast values for each patient study n. Now, using Eqs (18) and (19), we find the contrast values for P vertices on LVC, which yields the contrast vector c'_n for patient study n. Repeating this process for all the studies N, we get the contrast data matrix $\mathbf{C}$ $[N \times (P+3)]$.

2. Estimation of the training coefficients ($\hat{\mathbf{a}}$): we compute the training coefficients using the standard Least Squares to minimise the error function ϵ^2_{tr}, given as:

$$\epsilon^2_{tr} = \|\mathbf{C}_{tr}\mathbf{a} - \mathbf{e}\|^2 \quad \text{and} \quad \hat{\mathbf{a}} = \underbrace{(\mathbf{C}^T_{tr}\mathbf{C}_{tr})^{-1}\,\mathbf{C}^T_{tr}\,\mathbf{e}} \tag{20}$$

Note that $\mathbf{C}_{tr}$ has a dimension of $\underbrace{N_{tr} \times (P+3)}$, $\hat{\mathbf{a}}$ has a dimension of $\underbrace{(P+3) \times 1}$, and $\mathbf{e}$ has a dimension of $\underbrace{N_{tr} \times 1}$. N_{tr} is the number of training studies. Note here that the residue of the regression tells us about the placement of a vertex with respect to the contrast value (grey scale) at the edges in a ventriculogram. If the residue is high, then that vertex is definitely far away from the ideal boundary, which means that it can be far outside (over-estimated) or far inside (under-estimated) the left ventricle boundary. On the contrary, if the residual error is small, the vertex is definitely close to the ideal boundary. The residual error of the validation model largely depends upon the training data size (L-partitions out of N, as discussed in step 5).

3. Estimating the predicted errors $\hat{\mathbf{e}}$ on test boundaries (N_{te}): $\hat{\mathbf{e}} = \mathbf{C}_{te}\ \hat{\mathbf{a}}$ where, $\mathbf{C}_{te}$ has dimension $\underbrace{N_{te} \times (P+3)}$.
4. Statistical techniques and window sizes: here we repeat the rejection regression coefficient estimation process for three different sets of statistical techniques: Plain; Ordered; and Normalised and Ordered. Plain, because we arrange the boundary contrast values as per the vertex number of the LVC; Ordered, because we arrange the contrast values in increasing order, and then use it for calibration; Normalised, because we normalise the contrast values by its standard deviation. For each of the above techniques, we also change the window size for generation of the contrast data. Two sets of window sizes were taken into consideration, namely, 11×11 and 22×22.
5. Our validation approach is based on a *cross-validation* procedure for estimating the predicted errors. This pro-

cedure takes a database of N patient studies and partitions this database into K equal-sized subsets. Then for all the $K *choose* L combinations, we train the system using L subsets, and apply the estimated transformation on the remaining $(K-L)$ subsets. The mean predicted error of the contrast boundary data is then computed from these $(K-L)$ subsets coming from all K *choose* L combinations. The predicted errors which are above the threshold correspond to boundary delineation, which are rejected.

6.3. Performance: Mathematical Formulae for Probability of Misdetection, False Alarm and Mean Predicted Errors

Let $(s_{(i)},e_{(i)})$ be the name and the error for a patient from the list of observed (ideal) errors and corresponding names. This list is the sorted list of errors, and its corresponding names from the output of the boundary estimation algorithm. Similarly, let $(q_{(j)},p_{(j)})$ be the name and the error for a patient from the list of the predicted errors and corresponding names. This list of errors results from the output of the boundary rejection scheme. Note that the errors in both lists are sorted in increasing order, and hence the names of the patient boundaries are not in same order in both lists. These two lists will be used for finding the probabilities of misdetection vs. probabilities of false alarm, and mean errors for rejected and non-rejected patient boundaries. This is a very efficient scheme, because we can easily compute the elements of the contingency table and spot the patient boundaries given a threshold, R_{th}.

Let us define mathematically the terms of the contingency table shown in Table 1. m_{rr} is the number of patient studies truly rejected, and which was assigned to be rejected. m_{rs} is the number of patient studies truly rejected, but which was assigned to be non-rejected (selected). m_{sr} are the number of patient studies truly non-rejected (selected), but assigned to be rejected. m_{ss} is the number of patient studies truly non-rejected (selected) and which were assigned to be non-rejected (selected). These four terms can be mathematically expressed using our two input lists as:

$$m_{rr}(n) = \#\{i \mid \exists\, j,\ (s_{(i)} = q_{(j)}),\quad j > (N-n),\ i > (N-n) \tag{21}$$

$$m_{rs}(n) = \#\{i \mid \exists\, j,\ (s_{(i)} = q_{(j)}),\quad j \le (N-n),\ i > (N-n) \tag{22}$$

$$m_{sr}(n) = \#\{i \mid \exists\, j,\ (s_{(i)} = q_{(j)}),\quad j > (N-n),\ i \le (N-n) \tag{23}$$

$$m_{ss}(n) = \#\{i \mid \exists\, j,\ (s_{(i)} = q_{(j)}),\quad j \le (N-n),\ i \le (N-n) \tag{24}$$

Note that the elements of the contingency table are a function of each set of patient studies rejected n. Using these definitions, we can express the probability of misdetection $\mathbf{P}_{md}(n)$ and the probability of false alarm $P_{fa}(n)$ as a function of the total number of studies rejected n:

$$\mathbf{P}_{md}(n) = \frac{m_{rr}(n)}{m_{rr}(n) + m_{rs}(n)} \quad \text{and} \quad \mathbf{P}_{fa}(n) = \frac{m_{sr}(n)}{m_{sr}(n) + m_{ss}(n)} \tag{25}$$

Table 1. Contingency table

	Assigned	
True	m_{rr}	m_{rs}
True	m_{sr}	m_{ss}

6.3.1. Mean Errors for Rejected $\bar{p}_r(n)$ and Non-Rejected Studies $\bar{p}_{non}(n)$. Given the above lists, we can express the mean error of the rejected and non-rejected patient boundaries for the ideal (observed) and cross-validation cases as follows: let $\bar{e}_r(n)$ and $\bar{e}_{non}(n)$ be the errors for rejected and non-rejected (selected) patient boundaries for the ideal case. Let $\bar{p}_r(n)$ and $\bar{p}_{non}(n)$ be the errors for rejected and non-rejected (selected) patient boundaries for the cross-validation case.

For the ideal case:

$$\bar{e}_r(n) = \frac{1}{n} \sum_{i=N-n+1}^{N} e_{(i)} \quad \text{and} \quad \bar{e}_{non}(n) = \frac{1}{N-n} \sum_{i=1}^{N-n} e_{(i)} \tag{26}$$

For the cross-validation (CV) case:

$$\bar{p}_r(n) = \frac{1}{n} \sum_{i=N-n+1}^{N} p_{(j)} \quad \text{and} \quad \bar{p}_{non}(n) = \frac{1}{N-n} \sum_{i=1}^{N-n} p_{(j)} \tag{27}$$

The rejection threshold for n patient studies is computed using the list of predicted errors, and is given as: $R_{th}(\mathrm{n}) = p_{N-n}$.

6.4. Relationships Developed: Reliability Curves

1. $\bar{e}_{non}(n)$ and $\bar{p}_{non}(n)$ vs. n: this is the relation between the mean error for the non-rejected patients for both the ideal and cross-validation cases, and the total number of patients rejected n. $\bar{e}_{non}(n)$ and $\bar{p}_{non}(n)$ can be computed using the Eqs (26) and (27). We took the ED and ES frames from the same data set of 245 studies using IdCM errors for these experiments. In the first case, n is made to increase from 1 to 50. These two plots are shown in Fig. (14). We see that, as we reject 20% of the patient boundaries, the mean error for $\bar{e}_{non}(n)$ and $\bar{p}_{non}(n)$ drops from 4.4 mm to 4 mm (for the CV case) and 3.85 mm (for the Ideal case), respectively.
2. $\bar{e}_{non}(n)$ and $\bar{p}_{non}(n)$ vs. R_{th}: the is the relation between the mean error for the non-rejected patients for both the ideal and cross-validation cases, and the rejection threshold R_{th}. $\bar{e}_{non}(n)$ and $\bar{p}_{non}(n)$ can be computed using Eqs (26) and (27). This is shown in Fig. 11 (right). With the reduction in the rejection threshold R_{th}, the mean error drops down gradually.
3. $\mathbf{P}_{md}(n)$ vs. $\mathbf{P}_{fa}(n)$: this is the relation for the probability of misdetection versus the probability of false alarms for the set of n studies rejected. The curve is shown in Fig. 15. With an increase in $\mathbf{P}_{md}(n)$, $\mathbf{P}_{fa}(n)$ decreases. Note that both scales are from 0 to 1 (see Eq. (25)).

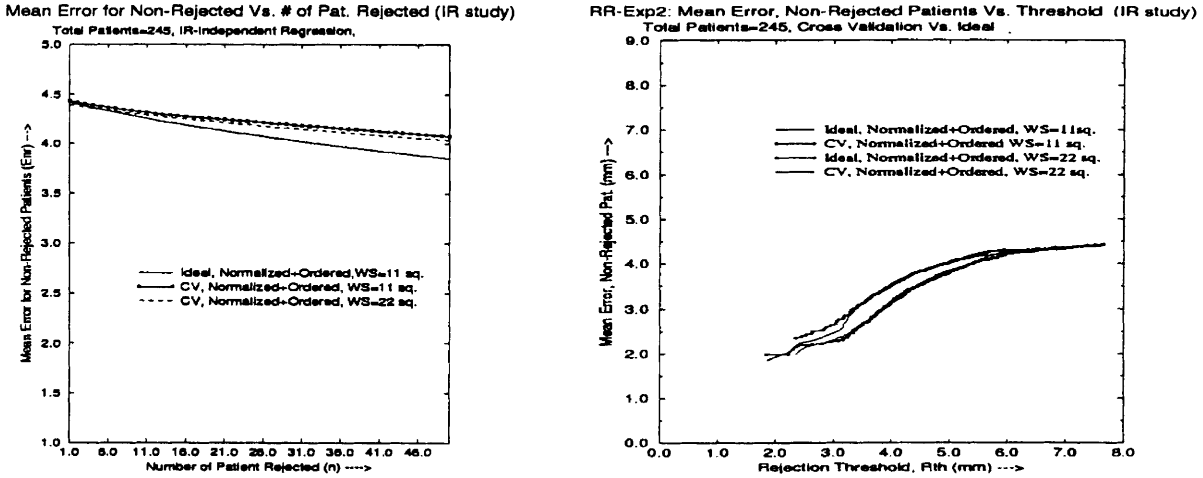


Fig. 14. Left: plot of mean of non-rejected patient studies vs. number of patients rejected (n). Right: plot of mean of non-rejected patient vs. rejection threshold.

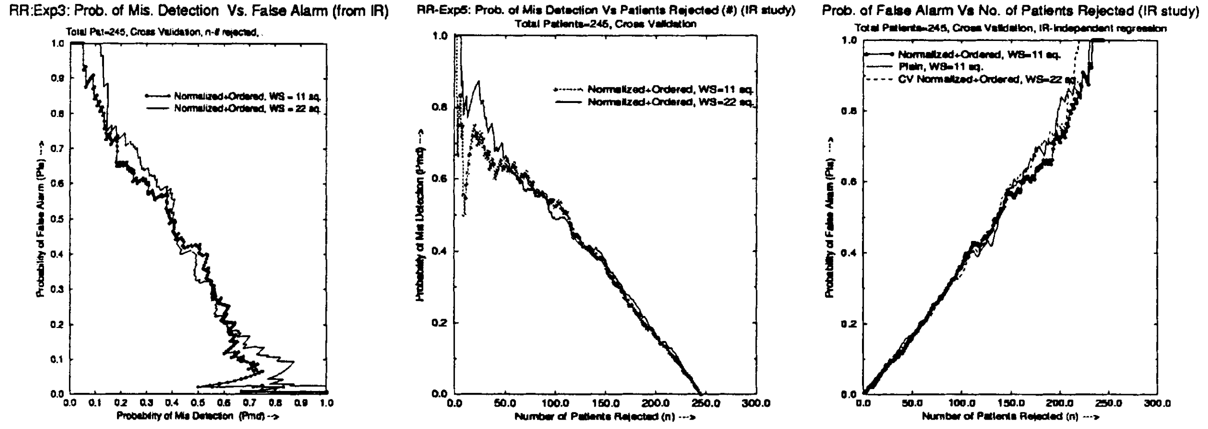


Fig. 15. Left: with the increase in probability of misdetection $\mathbf{P}_{md}(n)$ the probability of false-alarm $\mathbf{P}_{fa}(n)$ decreases. Note that each point on the curve corresponds to the total number of patients rejected (n). There are two cases shown in this plot. First, when the window size (WS) is 11 × 11, and second, when the WS is 22 × 22. Middle: plot of $P_{md}(n)$ vs. n. Right: plot of $\mathbf{P}_{fa}(n)$ vs. n.

4. $\mathbf{P}_{md}(n)$ vs. n: this is the relation for the probability of misdetection versus the number of patients rejected n (see Eq. (25)). This plot demonstrates that when more patients are rejected, then the probability of misdetection falls (see Fig. 12).
5. $\mathbf{P}_{fa}(n)$ vs. n: this is the relation between the probability of false alarms and the number of patients rejected n, computed using the relations expressed in Eq. (25). We see from the plot, with an increase in n, the probability of false alarm $(\mathbf{P}_{md}(n))$ increases. We have shown two cases: first, when the window size is 11 × 11; and second, when the WS is 22 × 22. When n is small, that is when a small number of patient boundaries are rejected, the $\mathbf{P}_{fa}(n)$ is more or less the same for both window sizes, but when a large number of patients is rejected, then the $\mathbf{P}_{fa}(n)$ is higher for a window size of 22 × 22.

The following are the advantages of our validation system. The general technique: (1) requires only grey scale cardioangiograms, observed errors and boundaries which needs to be validated; (2) detects automatically the patient boundaries which need to be rejected or that have large errors with respect to the ground truth boundaries, thus providing feedback to the boundary estimation system; (3) can be applied for observed errors (taken as ideal) coming from any source, say (segmentation algorithm, calibration algorithm, active contour fitting algorithm, dynamic programming-based algorithm, etc.). Finally, the validation system is easily implementable and portable on any any Unix-based system.

6.5. Discussion

The greedy algorithm for error correction fuses two sets of estimated boundaries: boundaries produced by the *identical coefficient* method, and boundaries produced by the *independent coefficient* method. If these two estimated boundaries have large errors with respect to the ground truth (ideal or hand-drawn boundary), then the greedy algorithm rapidly identifies those vertices which are too far from the ideal vertices, significantly reducing the boundary error. Nevertheless, the true shape of the left ventricle depends upon the number of optimised boundary vertices selected on the left ventricle contour. If the number of vertices is less than the optimum number of vertices, the true left ventricle shape may not be represented. On the other hand, if the number of vertices is larger than the optimised boundary vertices, the shape becomes more accurate, but generalisation is lost. Therefore, we first optimise the IdCM and InCM techniques, and then fuse the best results. Consequently, the error will always improve, but the drop in error will depend upon the following factors:

1. The number of vertices on LVC.
2. The initial errors of the IdCM and InCM boundary data before the fusion starts, which in turn depends upon the number of data vectors N for training the calibration model.
3. The starting error value (ϵ) before the greedy loop starts. (In our case, the starting value is the best error for the IdCM boundary data.)

These preliminary results indicate that the three sets of calibration algorithms significantly reduce the boundary error over the image processing algorithms. In other words, given any method for finding a digitised contour (computer-based estimates) in the plane for a certain class of images and the corresponding set of expert (or ground truth) contours, these three calibration algorithms will refine the computer-based estimates to be in better agreement with the expert (here the cardiologist). However, we can make the left ventricle boundary calibration system more robust by padding information or features like the apex information to the $\mathbf{Q}$ matrix (classifier data) to improve the accuracy [18]. Our algorithm requires no operator assistance; further, the algorithm is relatively simple and can be implemented on any commercial imaging system. One could say that the training algorithm alone used in this research is not a sophisticated method compared to neural networks. If you look at the entire system of boundary estimation, it fits very well: a Bayesian classifier as a raw boundary estimator followed by a calibrator for bias correction. Such a complex system is justified for left ventriculograms, because the Bayesian classifier uses temporal information and the calibration uses spatial information. We are also working on developing a constrained calibration system along with an automatic apex estimation technique for robust design of the left ventricle boundary estimation.

6.6. Conclusions

We presented three sets of calibration algorithms, the *identical coefficient* method, the *independent coefficient* method and the greedy calibration method. The *greedy calibration algorithm* for calibrating the initial pixel-based classifier boundaries takes the best of the other two calibration methods: the *identical coefficient* method and the *independent coefficient* method. The mean error over ED and ES frames using a *cross-validation* protocol and polyline distance metric is 3.5 mm over the database of 291 patient studies. The greedy algorithm is a considerable improvement over the *identical coefficient* method by 0.3 mm, which is significant for accuracy of the overall calibration system. The greedy algorithm performs best in the apex zone of the left ventricle, where the dye is unable to propagate, reducing the error by approximately 8.5 mm. Thus, we see that the calibration constitutes a significant last step for boundary estimation.

Acknowledgements

The authors would like to thank Linda G. Shapiro, Dean D. Lytle, D. D. Meldurm, Arun K. Somani and Werner Stuetzle (all from University of Washington) and Drs Poggio (MIT) and Ajit Singh (Siemens) for input and motivation. We also thank Ms Nan and others from CVRTC, University of Washington Medical Center, Seattle, for tracing the left ventricle borders in ventriculograms. Thanks are also due to Dr Kathy S. Ewing, University of Washington, for editing the manuscript.

References

1. Lee CK. Automated Boundary Tracing Using Temporal Information. PhD Thesis, Department of Electrical Engineering, University of Washington, Seattle, 1994
2. Moore RJ. Imaging Principles of Cardiac Angiography. Aspen, Rockville, MD, pp 1–258
3. Moodi DS, Yiannikas J. Digital Subtraction Angiography of the heart and lungs. Grune and Stratton (Harcourt Brace Jovanovich), New York, NY, 1986
4. Mancini GBJ. Clinical Applications of Cardiac Digital Angiography. Raven Press, NY, 1988, pp 1–324.
5. Chow CK, Kaneko T. Border detection based on local characteristics. Computers in Biomedicine Research 1972; 5:388–410
6. Griffith RL, Grant C, Kaufman H. An algorithm for locating the aortic valve and the apex in left ventricular angiocardiograms. IEEE Transactions on Biomedical Engineering 1974; 21(5): 345–349
7. Tanaka et al. U.S. Patent #: 4,952,805, Method of judging the presence or absence of a limited irradiation field, method of selecting a correct irradiation field, and method of judging correctness or incorrectness of an irradiation field, 1990
8. De Jong LP, Slager CJ. Detection of the left ventricle outline in angiographs using TV signal processing technique. IEEE Transactions in Biomedical Engineering 1975; 22(3):230–237
9. Reiber JHC. US Patent #: 4,101,961, Contour Detector and Data Acquisition System for the left ventricular outline, July 18 1978
10. Han CY, Porembka DT, Kwun-Nan. US Patent #: 5,457,754, Method for Automatic contour extraction of a cardiac image, October 10 1995
11. Wollschleger H, Tenspiel RW, Solzbach U, Zeiher AM, Just H. Reliable automatic frame-by-frame contour detection of digitised LV cine-angiograms. IEEE Computers in Cardiology 1988; 353–356
12. Van Bree. Improving left ventricular border recognition using probability surfaces. IEEE Computers in Cardiology 1989; 121–124
13. Kass, Wikins, Terzapolous. Snakes: active contour models. Int J Computer Vision 1988; 1(4):321–331

14. Cootes TF, Taylor CJ, Cooper DH, Graham J. Active shape models: their training and applications. Computer Vision and Image Understanding 1995; 61(1):38–59
15. Chiou GI, Hwang J-N. A neural network based stochastic active contour model (NNS-SNAKE) for contour finding of distinct features. IEEE Trans Image Proc 1985; 4(10):1407
16. Duncan J, Owen R, Anandan P, Staib L, et al. Shape-based tracking of left ventricle wall motion. Information Processing in Medical Imaging 1991; 41–44
17. Duncan JS, Owen RL, Staib LH, Anandan P. Measurement of non-rigid motion using contour shape descriptors. IEEE Trans. Pattern Recognition and Machine Intelligence (PAMI) 1991; 5:318–330
18. Suri JS, Haralick RM, Sheehan FH. Accurate left ventricle apex position and boundary estimation from noisy ventriculograms. Proc. IEEE Computers in Cardiology, Indianapolis, 1996
19. Suri JS, Haralick RM, Sheehan FH. Left ventricle longitudinal axis fitting and LV apex estimation using a robust algorithm and its performance: a parametric apex model. Proc International Conference in Image Processing, Santa Barbara, CA, 1997
20. Suri JS, Haralick RM, Sheehan FH. Two automatic calibration algorithms for left ventricle boundary estimation in X-ray images. Proc IEEE Int Conf of Engineering in Medicine and Biology (EMBS), Amsterdam, The Netherlands, October 31–November 3 1996
21. Suri JS, Haralick RM, Sheehan FM. Two automatic training-based forced calibration algorithms for left ventricle boundary estimation in cardiac images. 19th International Conference of IEEE Engineering in Medicine and Biology (EMBS), Chicago, IL, October 30–November 2 1997, pp 528–532
22. Klausner SC, Blair TJ, Bulawa WF, Jeppson GM, Jensen RL, Clayton PD. Quantitative analysis of segmental wall motion throughout systole and diastole in the normal human left ventricle. Circulation 1982; 65(3):580–590
23. Bolson EL, Kilman S, Sheehan FH, Dodge HT. Left ventricular segmental wall motion – a new method using local direction information. IEEE Computers in Cardiology 1980; 245
24. Sheehan FH, Stewart DK, Dodge HT et al. Variability in the measurement of regional left ventricular wall motion from contrast angiograms. Circulation 1983; 68(3):550–559
25. Press WH, Flannery BP, Teukolsky SA, Vetterling WT. Numerical Recipes in C. Cambridge University Press, Cambridge, 1988
26. Haralick RM, Shapiro LG. Computer and Robot Vision. Cambridge, Addison-Wesley, Reading, MA, 1991
27. Chen T, Lin WC, Chen CT. Artificial Neural Networks for 3-d nonrigid motion analysis. Artificial Neural Networks in Engineering, St. Louis, MI, November 1992
28. Beier J, Joerke T, Lempert S, Wellnhofer E, Oswald H, Fleck E. A comparison of 7 different volumetry methods of left and right ventricle using post-mortem phantoms. IEEE Computers in Cardiology 1993; 33–36
29. Geman D, Geman S, Graffigne C, Dong P. Boundary detection by constrained optimisation. IEEE Pattern Recognition and Machine Intelligence 1990; 12:609–628
30. Anthony HM, Cox MG. An automatic algorithm for immunoassay curve calibration using controlled quadratic splines. IMA Journal of Mathematics Applied in Medicine and Biology 1989; 6(2):91–110
31. Chen CW, Huang TS. Modeling, analysis, and visualisation of left ventricle shape and motion by hierarchical decomposition. IEEE Trans on Pattern Analysis and Machine Intelligence 1994; 16(4):342–356

Jasjit Suri was born in India in 1964, and received his Bachelors degree in Electronics and Computer Engineering from Maulana Azad College of Technology, Bhopal, his Master's degree in Computer Science from the University of Illinois, Chicago, and his Doctorate in Electrical Engineering from the University of Washington, Seattle, with specialisation in medical imaging. Dr Suri has more than 30 publications in the area of cardiac X-rays, brain MR, abdominal CT, dental radiography, ultrasound and pathology imaging. Dr Suri has presented his research work at several international conferences in the United States and abroad. He has worked with the IBM Scientific Center and Siemens Research Divisions in the United States. He has also spent time as a scientist at the University of Wisconsin, Madison, and was a faculty member at Maulana Azad College of Technology, Bhopal, India. He is a member of the IEEE, EMBS, ACM and SPIE societies. He was honored with the Indian President's Gold Medal in 1980, and holds US patents.

Robert M. Haralick was born in Brooklyn, New York, in 1943. He received a BA degree in Mathematics from the University of Kansas in 1964, a BS degree in Electrical Engineering in 1966, and a MS degree in Electrical Engineering in 1967. In 1969, after completing his PhD at the University of Kansas, he joined the faculty of the Electrical Engineering Department there, where he last served as Professor from 1975 to 1978. In 1979 Dr Haralick joined the Electrical Engineering Department at Virginia Polytechnic Institute and State University, where he was a Professor and Director of the Spatial Data Analysis Laboratory. From 1984 to 1986 Dr Haralick served as Vice President of Research at Machine Vision International, Ann Arbor, MI. Dr Haralick now occupies the Boeing Clairmont Egtvedt Professorship in the Department of Electrical Engineering at the University of Washington. Professor Haralick is a Fellow of IEEE for his contributions in computer vision and image processing, and a Fellow of IAPR for his contributions in pattern recognition, image processing, and for service to IAPR. He has served on the Editorial Board of *IEEE Transactions on Pattern Analysis and Machine Intelligence* and has been the computer vision area editor for *Communications of the ACM* and as an associate editor for *Computer Vision, Graphics, and Image Processing* and *Pattern Recognition*. He has published over 450 papers.

Florence H. Sheehan is a professor in the Department of Medicine, Division of Cardiology, University of Washington, Seattle. Dr Sheehan has vastly contributed in the area of cardiac imaging, and is a very active researcher. She has also made several inventions in the area of cardiac image processing.

Correspondence and offprint requests to: J. S. Suri, Division of Image Guided Surgery, Picker International, World HQ, 595 Miner Road, Highland Heights, Cleveland, OH 4414, USA. Email: jsuri@enya.picker.com

APPENDIX: GREEDY ALGORITHM IMPLEMENTATION

The greedy algorithm consists of three basic steps. First, fusion of IdCM and InCM boundary data; secondly, polyline performance to compute the errors; and thirdly, the vertex selection. The input to the fusion process are two sets of boundaries $\hat{\mathbf{R}}_{id}$ and $\hat{\mathbf{R}}_{in}$, which need to be fused. The fusion is done by selection of that vertex (or column) of the InCM boundary which contributes to a reduction in error. The Polyperformance() function takes two sets of boundaries: the fused boundary $\hat{\mathbf{R}}_{com}$ and the original ground truth $\mathbf{R}_{gt}$ consisting of $P_1 = 100$ vertices, and computes the mean error (as discussed in Section 4). The third function is the argmin() function or vertex selection function which takes the error associated with P_2 vertices and finds that vertex number from the InCM LV boundary that yielded the least error in each greedy cycle.

Greedy Algorithm

Let S, S_{id} and S_{in} be three sets consisting of all vertices, IdCM pool vertices and InCM pool vertices, respectively. Let $\hat{\mathbf{R}}_{id}$ $(2N \times P_2)$ and $\hat{\mathbf{R}}_{in}$ $(N \times 2P_2)$ be the estimated boundary matrices from the IdCM and InCM techniques with P_2 samples vertices. Let $\mathbf{R}_{gt}$ $(N \times 2P_1)$ and R $(N \times 2P_2)$ be the matrices consisting of (x,y)-coordinates from the original ground truth with $P_1 = 100$ and sampled P_2 vertices, respectively. Initially, all the vertices are considered in the IdCM pool and the error is computed. Denote its error by ϵ_{id}. Now we select that vertex from the IdCM pool which, when fused with the InCM pool vertices, yields an estimated boundary error lower than ϵ_{id}. This procedure is repeated until there is no further improvement. If ϵ is the error at any time in the greedy *do-while loop*, and $\mathcal{F}$ is the set consisting of all the frames, then the greedy *do-while loop* for any frame t in the set $\mathcal{F}$, consists of the following steps:

```
Greedy Boundary Calibration()
For each $t \in \mathcal{F}$
  $S_{id} = S$; $S_{in} = \phi$, $\epsilon = 0$ greedyCounter = 0
  While $(\epsilon \leq \epsilon_{id})$ do
    greedyCounter++
      For each $i \in S_{id}$, /* total vertices are $P_{id}$ */
        $S_{id} = S_{id} - \{i\}$ ; $S_{in} = S_{in} \cup \{i\}$
        Combine IdCM ($\hat{\mathbf{R}}_{id}$) and InCM ($\hat{\mathbf{R}}_{in}$) Using $S_{id}$ and $S_{in}$
        $\hat{\mathbf{R}}_{com}$ = Combine($\mathbf{R}_{id}$,$\mathbf{R}_{in}$,$N$,$P_2$,$S_{id}$,$S_{in}$, greedyCounter)
        Performance Evaluation using Original GT: Error for index $i$
        $\epsilon_i$ = PolyPerformance($\hat{\mathbf{R}}_{com}$,$\mathbf{R}_{gt}$,$N$,$P_1$,$P_2$)
      End /* end of the for loop */
    ArgMin Computation: Minimum error and best vertex $j$ selection
    $(\epsilon_{min}, j)$ = ArgMin($\epsilon[i]$, $P_{id} - greedyCounter$)
    if($\epsilon_{min} < \epsilon$) then $S_{id} = S_{id} - \{j\}$ ; $S_{in} = S_{in} \cup \{j\}$ else break;
    End
  END /* end of the while loop */
End /* end of all the frames of systolic heart cycle */
```

The advantage of the above algorithm is that our initial error is decided from either of the above coefficient methods. Note that this is implemented independently for each frame of the cardiac cycle.