Accurate Left Ventricle Apex Position and Boundary Estimation from Noisy Ventriculograms

JS Suri, RM Haralick, FH Sheehan[1]

Intelligent Systems Laboratory, [1]Cardiovascular Research & Training Center,
University of Washington, Seattle, USA

Abstract

This paper describes a methodology by which more accurate end-diastole and end-systole left ventricle boundaries can be automatically computed from the initial boundaries generated by a left ventricle pixel-based classifier. First, the end-diastole boundary is estimated from the boundary produced by a pixed-based classifier. Then the end-systole apex position is estimated from the end-diastole boundary. The end-systole boundary is then estimated using the end-systole boundaries produced by classifier and the estimated end-systole apex position. The methodology also permits a fusing of left ventricle boundary estimates produced by different techniques using a greedy algorithm.

We used two different boundary estimation techniques having mean errors of 3.631 mm and 3.539 mm with standard deviation of 3.418 mm and 3.376 mm. The corresponding fused boundary has a mean error of 3.471 mm with standard deviation of 3.277 mm. The initial boundary produced by classifier had a mean error of 6.4 mm with an error of 8.5 mm in apex zone.

1 Introduction

A left ventricle (LV) pixel-based classification procedure was developed in our laboratory [1], to estimate the borders of the LV in left ventriculograms (LVG). Because of the low contrast in the apex zone, high level of noise due to scattering of X-rays by tissue volume not related to LV, interference of ribs and diaphram with LV, and breathing artifacts from the catheterization procedure, the classifier failed to yield boundaries close to as delineated by the cardiologist. The error was 6.4 mm. The segmented LV boundaries have a systematic bias errors in shape, position and orientation.

The overall system of boundary estimation employs 3 sets of algorithms: identical coefficient, independent coefficient and fusion. Since the borders of the same LV is traced at different times in heart cycle, we can take advantage of the knowledge of the wall motion of one frame to help estimate the apex position in another frame. The greedy algorithm for end-systole (ES) apex estimation is discussed in section 3. The boundaries estimated from the two boundary techniques utilizing the apex information, is then, fused to improve the accuracy.

For estimation of boundary and apex error, the partition protocol uses a database of N patients studies and partitions it into K subsets each containing $\frac{N}{K}$ studies. Estimates are then obtained using L of the K subsets. Rotating through all L choose K combinations, we measure the accuracy of the results on the remaining K–L subsets. Because of the small number of patient studies N, and large number of parameters (about 200 times N) in the boundary transformation, there is a danger of *memorization* rather than *generalization* in the estimation of the transformation parameters. With the other parameters N, K, and L fixed, there will be an *optimal* number of boundary vertices balancing the *representation error* with the *memorization error.* Our protocol finds this *optimal number.*

2 Two Boundary Estimators

Ground truth boundaries refer to the hand delineated boundaries traced by the cardiologist. Raw or classifier boundaries are the the boundaries produced by the pixel based classification algorithm [1].

In the identical coefficient method (IdCM), the estimated x and estimated y coordinates are computed using the *same* linear combination of raw x and raw y coordinates associated with that vertex of the LV boundary. In the independent coefficient method (InCM), the estimated x and estimated y coordinates are computed as a *different* linear combination of the raw x and raw y coordinates associated with that vertex of the LV boundary. The problem of boundary estimation then reduces to a problem of determining the coefficients of the linear combination. This can be accomplished by solving a regression problem as discussed below.

2.1 Identical Coefficient Method (IdCM)

Let g'_n=[$x_{1n} \ldots x_{Pn}$] and h'_n=[$y_{1n} \ldots y_{Pn}$] be the P-dimensional row vectors of x-coordinates and y-coordinates respectively, corresponding to the *ground truth boundaries* for any patient n, where $n = 1, \ldots, N$. Let r'_n and s'_n be the P-dimensional row vectors of x-coordinates and y-coordinates respectively for the classifier boundary for any patient n, where $n = 1, \ldots, N$. For the boundary estimation in IdCM we are:

- **Given**: Corresponding pairs of ground truth boundary matrix $\mathbf{R}$ [$2N \times P$], and the classifier boundary matrix, $\mathbf{Q}$ [$2N \times (P+4)$], respectively:

$$\mathbf{R} = \begin{pmatrix} g'_1 \\ h'_1 \\ \cdots \\ g'_N \\ h'_N \end{pmatrix} \qquad \mathbf{Q} = \begin{pmatrix} r'_1\ \underbrace{1\ u'_1}\ \underbrace{p_1} \\ s'_1\ \underbrace{1\ v'_1}\ \underbrace{q_1} \\ \cdots \\ r'_N\ \underbrace{1\ u'_N}\ \underbrace{p_N} \\ s'_N\ \underbrace{1\ v'_N}\ \underbrace{q_N} \end{pmatrix}$$

where, u'_n, v'_n, are the vectors for x and y coordinates of the ground truth AoV plane for patient n and (p_n, q_n) is the estimated apex coordinates of LV.

- We find $\mathbf{A}$ [$(P+4) \times P$], the unknown coefficient matrix that minimizes $\| \mathbf{R} - \mathbf{Q}\,\mathbf{A} \|^2$.

2.2 Independent Coefficient Method

Using the same notations as above, g'_n, h'_n, r'_n and s'_n, for boundary estimation of the LV in InCM we are:

- **Given**: Corresponding ground truth boundaries $\mathbf{R}$ [$N \times 2P$], classifier boundaries $\mathbf{Q}$ [$N \times (2P+7)$] respectively:

$$\mathbf{R} = \begin{pmatrix} g'_1\, h'_1 \\ \cdots \\ g'_N\, h'_N \end{pmatrix} \qquad \mathbf{Q} = \begin{pmatrix} r'_1\ s'_1\ \underbrace{1\ u'_1\ v'_1}\ \underbrace{p_1\ q_1} \\ \cdots \\ r'_N\ s'_N\ \underbrace{1\ u'_N\ v'_N}\ \underbrace{p_N\ q_N} \end{pmatrix}$$

- We find $\mathbf{A}$ [$(2P+7) \times 2P$], the unknown coefficient matrix that minimizes $\| \mathbf{R} - \mathbf{Q}\,\mathbf{A} \|^2$.

Thus for any classifier boundary matrix $\mathbf{Q}$, the estimated vertices of the boundary are given by $\mathbf{Q}\hat{\mathbf{A}}$, where $\hat{\mathbf{A}}$ is the estimated coefficient matrix. The above two methods are different in the way that the estimation model is set up. The classifier boundary matrix $\mathbf{Q}$ in IdCM is of size $2N \times (P+4)$ while in InCM is of size $N \times (2P+7)$. For IdCM, the number of coefficients estimated in the $\hat{\mathbf{A}}$ matrix is $(P+4) \times P$. For InCM the number of coefficients estimated is $(2P+7) \times 2P$. Thus InCM requires around *4 times* the number of coefficients of IdCM to be estimated, and this difference represents a significant factor in the ability of the technique to *generalize* rather than *memorize* for our data set size which is N=312. Using the partition protocol and generalizing for any frame t, the minimizing $\hat{\mathbf{A}}_{tr}$ and estimated boundaries $\hat{\mathbf{R}}_{te}$ on the test set $\mathbf{Q}_{te}$, are:

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

$\hat{\mathbf{A}}_{tr}$ is calculated using a singular value decomposition.

3 ES LV Apex (p, q) and LV Boundary Estimation $\mathbf{R}_{te}$: Greedy Approaches

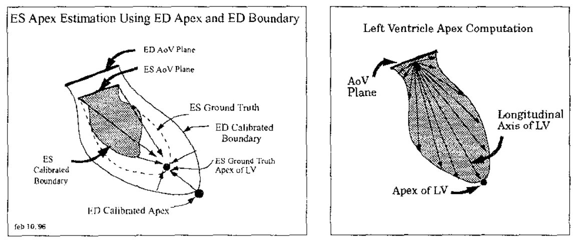


Figure 1: **Left**: Computation of the ES LV apex position using ED boundary information (dependence approach), **Right**: Computation of the apex position using the farthest vertex algorithm.

(**a**) *ED Boundary Estimation*: We first estimate the ED boundaries using IdCM and InCM models without apex padding (fig. 2), that is (p, q) are not taken into consideration. (**b**) *ES Apex Estimation*: Now we use the InCM model, $\| \mathcal{R}_{tr} - \mathcal{Q}_{tr}\,\mathcal{A}_{tr} \|^2$, for estimating the transformation parameters $\mathcal{A}_{tr}$ required in the ES apex estimation. Here, $\mathcal{R}_{tr}$ is the training ground truth ES apex coordinates, and $\mathcal{Q}_{tr}$ is the training ED apex coordinates along with the coordinates of the selected ED boundary vertices. We use a greedy approach for ED boundary vertex selection, subject to minimization of error between the estimated ES apex position and the ground truth ES apex position (fig 1, left). $\mathcal{A}_{tr}$ is calculated the same way as in Eq. 1 but this time, the $\mathcal{R}_{tr}$ and $\mathcal{Q}_{tr}$ matrices are determined by first computing the ED apex and ES apex using the estimated ED and ground truth ES boundaries. From these boundaries, the apex position is given as that vertex on the bottom $(\frac{1}{3})^{rd}$ LV boundary which is farthest from the midpoint of the AoV plane (see fig.

1,right). The greedy algorithm for ES apex estimation is as follows: Let S be the set of all the vertices, S_{ed} be the set of vertices in the ED pool, S_{com} be the set of vertices in the combination pool which contains the combination of ED LV apex, ground truth ES AoV plane, and selected vertices of the estimated ED LV boundary. Initially the ED apex without any ED boundary vertex is considered and the error is computed. Denote its error by ϵ_{ed}. Now we select that vertex from ED pool which when linearly combined with current combination pool vertices yields an estimated ES apex having an error lower than ϵ_{ed}. This procedure is repeated until there is no further improvement. Note each time a new ED LV vertex is transferred from ED pool to the combination pool, we design a new $\mathcal{Q}_{tr}$, perform estimation and performance. Using the same partition protocol, we then run the algorithm ${}^{K}C_{L}$ times so that each partition becomes a test set.

Greedy Apex Estimation Algorithm

```
cn=0   /* combination number */
For all Combinations, $({}^{K}C_{L})$
cn++; $S_{ed}=S$; $S_{com}=\phi$, $\epsilon$=0, gc=0
$\epsilon_{ed}$ = InitialErrorNoED$\epsilon_{ed}$( $\mathcal{Q}_{tr}$, $\mathcal{R}_{tr}$, $\mathbf{G}_{tr}$, $N_{tr}$, $N_{te}$ $\mathbf{D}_{tr}$, $\hat{\mathcal{R}}_{te}$, $\mathcal{Q}_{te}$, $\mathcal{R}_{te}$, $\mathbf{G}_{te}$, $N$, $P$)
    While ($\epsilon \leq \epsilon_{ed}$)  do
        gc++ /* greedy counter */
        For each $i \in S_{ed}$
            $S_{ed}=S_{ed}-\{i\}$; $S_{com}=S_{com} \bigcup \{i\}$
            $\mathcal{R}_{tr}$= GenerateR( $\mathbf{G}_{tr}$, $N_{tr}$, $P$, gc)
            $\mathcal{Q}_{tr}$= GenerateQ( $\mathbf{D}_{tr}$, $N_{tr}$, $P$, $S_{ed}$, $S_{com}$, gc)
            $\mathcal{A}_{tr}$= LinearCalibrator( $\mathcal{Q}_{tr}$, $\mathcal{R}_{tr}$, $N_{tr}$, $P$, gc)
            $\hat{\mathcal{R}}_{te}$= Estimator($\mathcal{A}_{tr}$, $\mathcal{Q}_{te}$, $N_{te}$, $P$)
            $\epsilon_i$=  Performance($\hat{\mathcal{R}}_{te}$, $N_{te}$, $P$, $\mathcal{R}_{te}$ )
            CurrentApexLocation(i)=$\hat{\mathcal{R}}_{te}$
        end   /* end of the for loop */
        ArgMin Comp: Min. error & best vertex $j$ selection
        $(\epsilon_{min}, j)$ = ArgMin($\epsilon[i]$, ($P$-gc) )
        if ( $\epsilon_{min} < \epsilon$)  then
            $S_{ed}=S_{ed}$ - $\{j\}$ ; $S_{com}=S_{com} \bigcup \{j\}$
            LatestApexPosition=CurrentApexPosition(j)
        else break; endif
    end /* end of the while loop */
    OutputESapexCoordinates(LatestApexPosition,$\hat{\mathcal{R}}_{te}$)
end /* end of all combinations */
```

(c) *ES Boundary Estimation Utilizing Apex*: Utilizing the above estimated ES apex position, we estimate ES boundaries using boundary algorithms, IdCM and InCM, as shown in fig.2. **(d)** *Boundary Fusion*: The final ES boundary is now estimated by fusing the above estimated ES boundaries using the greedy boundary estimator [3], where we select a fixed subset of estimated vertex positions from each technique (IdCM and InCM) which when fused together, minimizes the resulting error between the fused estimated polygon boundary and the physician traced LV boundary. For the final ED LV boundary, we simply fuse the ED boundaries estimated from IdCM and InCM algorithms (step a) using the same greedy boundary estimator [3]. The fused boundary for any frame (ED or ES) is now B-spline fitted and interpolated back from P_2 to 100 vertices. The last stage of the system (fig. 2) undergoes performance evaluation using the polyline distance metric discussed below.

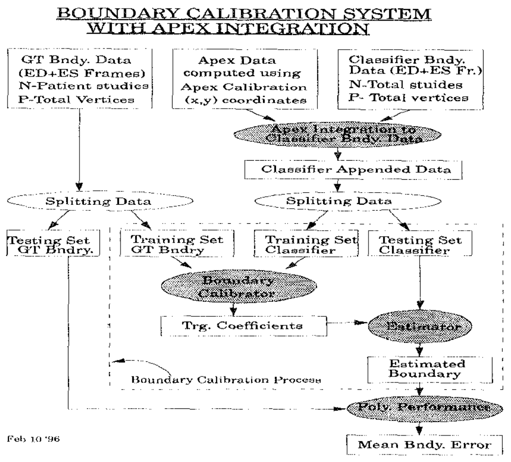

BOUNDARY CALIBRATION SYSTEM WITH APEX INTEGRATION
GT Bndy. Data (ED+ES Frames) N-Patient studies P-Total Vertices
Apex Data computed using Apex Calibration (x,y) coordinates
Classifier Bndy. Data (ED+ES Fr.) N-Total stuides P- Total vertices
Apex Integration to Classifier Bndy. Data
Classifier Appended Data
Splitting Data
Splitting Data
Testing Set GT Bndry.
Training Set GT Bndry
Training Set Classifier
Testing Set Classifier
Boundary Calibrator
Trg. Coefficients
Estimator
Estimated Boundary
Boundary Calibration Process
Poly. Performance
Mean Bndy. Error
Feb 10 '96

Figure 2: Boundary estimation system utilizing apex position. Boundary data is first sampled from 100 vertices to P_2, then partitioned into training (L) and testing (K-L) sets. $\hat{\mathbf{A}}_{tr}$ is estimated off-line and $\mathbf{R}_{te}$ is estimated on-line. 1 pixel= 0.39 mm.

4 Performance: Results & Discussions

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 to the boundary of the other polygon. To define this measure precisely, first requires having defined 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_0,y_0), and a line segment having end points (x_1,y_1) and (x_2,y_2) is:

$$d(v,s) \;=\; \begin{cases} \min\{d_1,d_2\}; & if \;\; \lambda<0, \lambda>1 \\ |d^{\perp}|; & if \;\; 0 \leq \lambda \leq 1, \end{cases} \tag{2}$$

where d_1, d_2 are the distance between coordinates pairs (x_0,y_0), (x_1,y_1) and (x_0,y_0), (x_2,y_2),

$$\begin{aligned} \lambda &= \frac{(y_2-y_1)(y_0-y_1)+(x_2-x_1)(x_0-x_1)}{(x_2-x_1)^2+(y_2-y_1)^2} \\ d^{\perp} &= \frac{(y_2-y_1)(x_1-x_0)+(x_2-x_1)(y_0-y_1)}{\sqrt{(x_2-x_1)^2+(y_2-y_1)^2}} \end{aligned} \tag{3}$$

The distance $d_b(v, B_2)$ measuring the polyline distance from vertex v to the boundary B_2 is defined:

$$d_b(v, B_2) \quad = \quad \min_{s \in \; sides \; B_2} \quad d(v, s) \qquad (4)$$

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.

$$d_{vb}(B_1, B_2) \quad = \quad \sum_{v \in \; vertices \; B_1} \quad d(v, B_2) \qquad (5)$$

On reversing from B_2 to B_1, we can similarly compute $d_{vb}(B_2, B_1)$. Using Eq. 5, the polyline distance between polygons, $D_s(B_1{:}B_2)$ is defined by:

$$D_s(B_1 : B_2) \quad = \quad \frac{d_{vb}(B_1, B_2) + d_{vb}(B_2, B_1)}{(\#vertices \in B_1 + \#vertices \in B_2)} \qquad (6)$$

Using the definition of the polyline distance between 2 polygons, we can now compute the overall mean error of the system. It is denoted by e^{poly}_{NFP} and defined by:

$$e^{poly}_{NFP} = \frac{\sum_{t=1}^{F} \sum_{n=1}^{N} D_s(G_{nt}, C_{nt})}{F \times N} \qquad (7)$$

where, $D_s(G_{nt}, C_{nt})$ is the polyline distance between the ground truth G_{nt} and estimated polygons C_{nt} for patient study n and frame number t. Using Eq. 7, the over all mean is plotted in Fig. 3. We compare our ruler of polyline distance metric with centerline method [2] developed by Sheehan, and found that we are better at the 3rd decimal place. As the number of boundary vertices P_2 increase, we observe that resulting boundary error first drops for all three algorithms and then starts rising again. The greedy algorithm that fuses IdCM and InCM boundaries performs better. The optimal number of vertices was 25. The mean error was 3.471 mm with a std. deviation 3.277 mm.

References

[1] C. K. Lee, *Automated Boundary Tracing Using Temporal Information*, PhD Thesis, Dept. of Electrical Engineering, Universtiy of Washington, Seattle, 1994.

[2] Florence H. Sheehan al., *Advantages and applications of the centerline method for characterizing region ventricular function* , CIRCULATION, vol. 74, No. 3, p293-p305, 1986.

[3] Jasjit S. Suri, Robert M. Haralick and Florence H. Sheehan, *Greedy Algorithm for Error Correction in Automatically Produced Boundaries in Low Contrast Ventriculograms*, Submitted to IEEE Transactions in Medical Imaging.

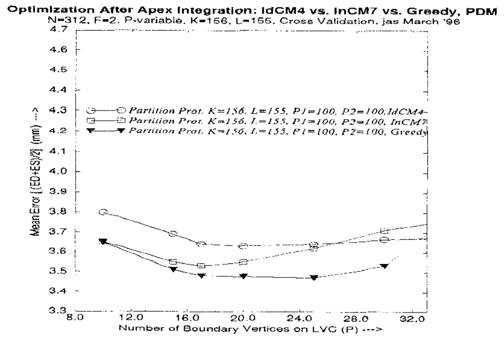
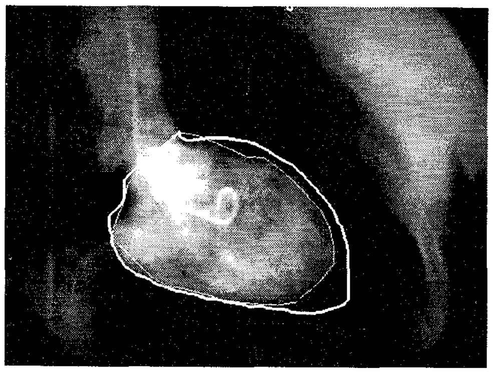
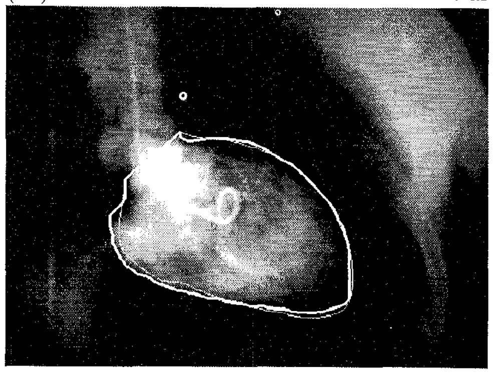


Figure 3: Optimization curves for 3 algorithms, IdCM, InCM and Greedy utilizing the apex position information. *Calibration Parameters*: N=312, K=156, L=155, F=2, P_1=100, $5 \leq P_2 \leq 40$.



(a1) ED Frame: GT and Classifier or Initial



(a2) ED Frame: GT and Calibrated (Estimated)

Figure 4: Results of greedy algorithm after apex integration. **Upper**: (a1) Classifier (thin) vs. GT (thick). **Bottom** :(a2) Calibrated (thin) vs. GT (thick). Background: gray scale X-ray image. *Estimation Parameters*: N=312, K=156, L=155, F=2, P_1=100, P_2=30, Mean end frame error $(\frac{ED+ES}{2})$ =1.30 mm, Mean error e^{poly}_{NFP}=3.471, Std. Deviation=3.277 mm.