3D Ultrasound Tracking of The Left Ventricle Using One-Step Forward Prediction and Data Fusion of Collaborative Trackers

Lin Yang[†§*] Bogdan Georgescu[§] Yefeng Zheng[§] Peter Meer[†] Dorin Comaniciu[§]
[†]Electrical and Computer Engineering
Rutgers University
Piscataway, NJ 08854
[§]Integrated Data Systems
Siemens Corporate Research
Princeton, NJ 08540

Abstract

Tracking the left ventricle (LV) in 3D ultrasound data is a challenging task because of the poor image quality and speed requirements. Many previous algorithms applied standard 2D tracking methods to tackle the 3D problem. However, the performance is limited due to increased data size, landmarks ambiguity, signal drop-out or non-rigid deformation. In this paper we present a robust, fast and accurate 3D LV tracking algorithm. We propose a novel one-step forward prediction to generate the motion prior using motion manifold learning, and introduce two collaborative trackers to achieve both temporal consistency and failure recovery. Compared with tracking by detection and 3D optical flow, our algorithm provides the best results and subvoxel accuracy. The new tracking algorithm is completely automatic and computationally efficient. It requires less than 1.5 seconds to process a 3D volume which contains 4,925,440 voxels.

1. Introduction

The 3D Echocardiography is one of the emerging diagnostic tools among modern imaging modalities for visualizing cardiac structure and diagnosing cardiovascular diseases. Ultrasound imaging is real-time, noninvasive and less expensive than CT and MR. However, ultrasound normally produces noisy images with poor object boundaries.

Recently, the problem of automatic detection, segmentation and tracking of heart chambers have received considerable attentions [5, 10, 11, 13, 27]. Among these applications, segmentation and tracking of the left ventricle (LV) have attracted particular interests. It provides clinical significance for radiologists to evaluate disease, such as acute myocardial infarction. Several difficulties exist compared with traditional 2D tracking algorithms:

- The computation demand is much higher for 3D volumetric data.
- Feature point based tracking has difficulty for ultrasound images because they lack reliable landmarks.
- In order to provide a practical diagnosis tool for radiologists, a tracking algorithm should be able to recover from failures.

The widely used 2D tracking algorithms [25] won't provide good results if directly applied on 3D ultrasound tracking applications. In order to achieve robust tracking in 3D ultrasound which is characterized by low image quality, learning based detector and boundary classifiers are used to track the LV boundary in each frame. This tracking by detection strategy can avoid accumulating errors and is proven to be quite effective in recent literature [1, 15, 26]. However, it still has several problems:

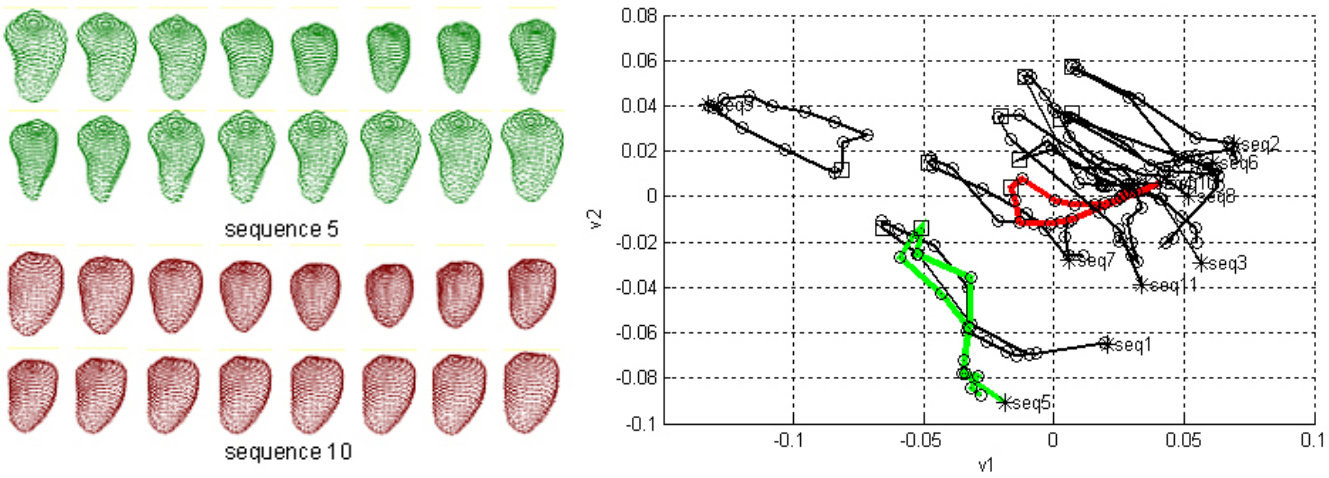
- The boundary classifiers are sensitive to initial positions [4] and good initializations have to be provided because we can not exhaustively search all the possible configurations in the whole 3D volume. Considering the speed, the current search range is constrained on the normal directions within ± 12 mm.
- Tracking by detection applies an universal description of the objects without considering any temporal relationship, which leads to temporal inconsistence between adjacent frames.

In this paper, we propose a fast and novel automatic 3D ultrasound tracking algorithm which address these difficulties. The contributions of this paper are:

- We propose a novel one-step forward prediction using motion manifold learning, which respects the geodesic distances of both the shape and motion of the heart. The motion priors can provide good initial positions for the boundary classifiers.
- A collaborative 3D template tracker is applied to erase the temporal inconsistence introduced by 3D detection tracker.
- A rectangle filter is used to reject outliers and achieve robust data fusion. Smooth boundary tracking is obtained by projecting the tracking points in each frame to the constrained shape boundary.
- The algorithm can process a 3D volume, e.g., $160 \times 144 \times 208$ voxels, in less than 1.5 seconds and we obtain subvoxel tracking accuracy.

*This research was completed when the author was in the Integrated Data Systems Department of Siemens Corporate Research.

978-1-4244-2243-2/08/$25.00 ©2008 IEEE



(a) (b)

Figure 1. Manifold embedding for LV motion patterns. (a) Two LV boundary mesh sequences. (b) 11 sequences embedded in a 2D subspace. Note: The end diastolic (ED) phase has larger volumes and represented as stars in (b), while the end systolic (ES) phase has smaller volumes and represented as squares in (b).

Section 2 introduces the Bayesian tracking framework. The learning methods and tracking algorithm are described in Section 3 and Section 4. Section 5 provides the experimental results and Section 6 concludes the paper.

2. Bayesian Tracking Framework

Define $\mathbf{x}_t$ as the true position of each 3D boundary point of the left ventricle (LV) at time t and let $\mathbf{z}_t$ to represent the measurement. The tracking problem can be formulated as an estimation of A posterior probability $p(\mathbf{x}_t|\mathbf{z}_{1:t})$, where $\mathbf{z}_{1:t} = \{\mathbf{z}_1, ...\mathbf{z}_t\}$ represents the past t measurements. Sequential Bayesian tracking based on Markovian assumption is performed recursively in a *prediction*

$$p(\mathbf{x}_t|\mathbf{z}_{1:t-1}) = \int p(\mathbf{x}_t|\mathbf{x}_{t-1})p(\mathbf{x}_{t-1}|\mathbf{z}_{1:t-1})d\mathbf{x}_{t-1} \quad (1)$$

and *updating* step

$$p(\mathbf{x}_t|\mathbf{z}_{1:t}) \propto p(\mathbf{z}_t|\mathbf{x}_t)p(\mathbf{x}_t|\mathbf{z}_{1:t-1}). \quad (2)$$

Bayesian tracking assumes that the following densities are known. The $p(\mathbf{x}_0)$ denotes the distribution of the 3D LV surface points in the first frame. In our algorithm $p(\mathbf{x}_0)$ is automatically calculated using the trained detector and boundary classifiers. The $p(\mathbf{x}_t|\mathbf{x}_{t-1})$ represents the motion prior (state model) and is predicted for the next frame. The $p(\mathbf{z}_t|\mathbf{x}_t)$ represents the measurement density.

If all the densities in (1) and (2) are Gaussian, Bayesian tracking can be formulated as the Kalman filter, otherwise a particle filter can be applied. The CONDENSATION algorithm [12] can be used to sample the posterior probability for single object, and Markov Chain Monte Carlo (MCMC) [14] sampling can be used for multiple objects tracking. Both Kalman filtering and particle filter assume a Markov model, which only considers the previous state $\mathbf{x}_{t-1}$ to estimate the density of current state $\mathbf{x}_t$. The $p(\mathbf{x}_t|\mathbf{x}_{t-1}, \mathbf{z}_{1:t-1})$ is therefore equal to $p(\mathbf{x}_t|\mathbf{x}_{t-1})$ in (1) and is often modeled using a predetermined distribution.

In real tracking problems, the motion prior (state model) $p(\mathbf{x}_t|\mathbf{x}_{t-1}, \mathbf{z}_{1:t-1})$ may not follow a Markovian assumption and it could be of any form. In our algorithm, we model the $p(\mathbf{x}_t|\mathbf{x}_{t-1}, \mathbf{z}_{1:t-1})$ to be dependent on both $\mathbf{x}_{t-1}$ and $\mathbf{z}_{1:t-1}$. One-step forward prediction using motion manifold learning is applied to estimate this motion prior.

For the measurement densities $p(\mathbf{z}_t|\mathbf{x}_t)$, we select two collaborative trackers: the detection tracker and the template tracker, which can mutually benefit each other. The detection tracker can discriminate the 3D target from background in low image quality and noisy environment. The template tracker respects the local image information and preserves the temporal consistence between adjacent frames. The trackers are modeled as r_k, $k = 1$ for detection tracker and $k = 2$ for template tracker, then

$$p(\mathbf{z}_t|\mathbf{x}_t) = p(\mathbf{z}_t|\mathbf{x}_t, r_1)p(r_1) + p(\mathbf{z}_t|\mathbf{x}_t, r_2)p(r_2). \quad (3)$$

Substituting (3) in (2) and replacing $p(\mathbf{x}_t|\mathbf{x}_{t-1})$ with $p(\mathbf{x}_t|\mathbf{x}_{t-1}, \mathbf{z}_{1:t-1})$ in (1), the final posterior probability $p(\mathbf{x}_t|\mathbf{z}_{1:t})$ is obtained from the robust data fusion and the $\mathbf{x}_t = \arg\max_{\mathbf{x}_t} p(\mathbf{x}_t|\mathbf{z}_{1:t})$.

3. Motion Learning and Classifiers Training

Each training sequence contains a heart motion cycle which starts from the end-diastolic (ED) phase, passes through the end-systolic (ES) phase and comes back to ED. The training procedure is in a batch mode where all the annotated sequences are used at the same time. It contains

three steps. First the motion modes are learned using manifold learning and hierarchical K-means. Next, an ED detector is trained to locate the position of the object in the first frame. Finally, two boundary classifiers (one for ED and one for ES) are trained using the annotated sequences to delineate the boundary.

3.1. Learning the Motion Modes

Motion Alignment Using 4D Generalized Procrustes Analysis. Our training sequences contain 11-25 time frames and 16 frames are chosen to resample each motion sequence. In this way we generate 4D motion vectors containing the same dimensionality d, where $d = N_f \times 3 \times 16$, $N_f = 289$ is the number of boundary points and three represents x, y and z dimensions.

Generalized procrustes analysis (GPA) is used to align all resampled motion vectors to remove the translation, rotation and scaling [7, ch. 5]. However, the shape differences and motion patterns are still preserved. After the 4D GPA, these aligned motion vectors are decomposed into separated 3D shapes. All the following learning steps are performed on the aligned 3D shape vectors.

Motion Manifold Learning. Because the actual number of constraints that control the LV motion are much less than its original dimensionality, the aligned 3D shape vectors lie on a low-dimensional manifold. Given the whole set of 3D training shape vectors, $M = \{\mathbf{m}_0, ..., \mathbf{m}_i, ..., \mathbf{m}_n\}$ where $\mathbf{m}_i \in R^d$, there exists a mapping F which represents $\mathbf{m}_i$ in the low-dimensions as

$$\mathbf{m}_i = F(\mathbf{v}_i) + \mathbf{u}_i \quad i = 1, 2, ..., n \tag{4}$$

where $\mathbf{u}_i \in R^d$ is the sampling noise and $\mathbf{v}_i \in R^q$ denotes the original i-th shape $\mathbf{m}_i$ in the low-dimensional subspace. We set $q = 2$ because the higher dimensions did not provide additional accuracy in our experiments. The nonlinear mapping F is the transformation from the low-dimensional subspace to the original space.

We apply ISOMAP [21] to embed the nonlinear manifold into a low-dimensional subspace. We start by finding the seven closest neighbors (seven was found to be the best) of each point $\mathbf{m}_i$ in the original space R^d and connect the neighbors to form a weighted graph G. The weights are calculated based on the Euclidean distance between each connected pair of vectors. We then calculate the shortest distance $d_G(i, j)$ between any pair of points $\mathbf{m}_i$ and $\mathbf{m}_j$ in the graph G. The final step is to apply the standard multiple dimensional scaling (MDS) to the matrix of graph distance $\{d_G(i, j)\}$. In this way, the ISOMAP applies a linear MDS on the local patch but preserve the geometric distance globally using the shortest path in the weighted graph G.

Figure 1a shows two annotated LV motion sequences. Figure 1b shows several LV motion representations in a low-dimensional subspace. An interesting but expected observation is illustrated in Figure 1b. The LV motion is almost periodic because one cycle of heart beat starts from ED and returns to ED. In total we applied manifold learning on 36 annotated LV motion sequences. In order to make the figure readable, we only show 11 sequences in Figure 1b.

Hierarchical K-means Clustering. Given all the motion cycles shown on the embedded subspace, we applied a hierarchical K-means to learn the motion modes. First, we apply K-means on all the training ED shapes and K is taken as four. (Four was chosen using a pilot experiment.) After this step, we align all motion sequences in one group by moving their ED vectors to their cluster center. In this way we cancel the translation among the shapes in one subgroup, but focus on the difference in motion patterns. Each aligned sequence is transformed to a 2×16 dimensional vector, where two represents the reduced dimensionality and 16 represents the number of frames. K-means is applied again within each group and the cluster center in the 32 dimensional space corresponds to a motion mode. In this step the K is decided by evaluating the difference between the cluster center and each vector within the group. Each motion mode is a weighted sum of all sequences that are clustered into the same group. The weights are proportional to their Euclidean distance from the cluster center. The geodesic distance in the original manifold is modeled by Euclidean distance in the embedded low-dimensional subspace.

3.2. Learning the ED Detector

In this step we train a 3D detector to locate the pose of LV in the motion sequence. We first calculate a mean shape by averaging all the LV in the ED frames of the annotated training sequences. A principal component analysis (PCA) shape space is calculated for all the ED shapes at the same time. In order to automatically initialize the tracker, we need to find the similarity transformation from the mean shape to the LV in the ED frame for each sequence. Discriminative learning based approaches have proven to be efficient and robust for 2D object detection [24]. The object is found by scanning the classifier over an exhaustive range of possible locations, orientations and scales in an image. However, it is challenging to extend them to 3D objection detection problems since the number of hypotheses increases exponentially with respect to the dimensionality of the parameter space. As the posterior distribution is often clustered in a small region, it is not necessary to search the parameter space exhaustively.

Marginal space learning (MSL) [2, 27] is used to learn an ED detector to locate LV in the first frame efficiently. The idea for MSL is to incrementally learn classifiers on projected marginal spaces. We split the estimation into position detection, position-orientation detection and full similarity transformation detection. The MSL reduces the number of

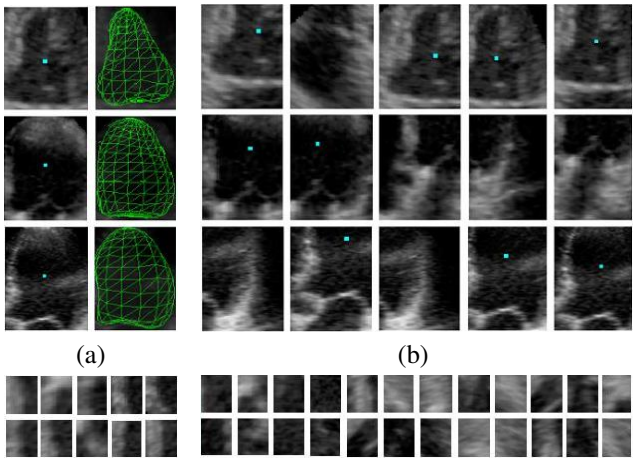


Figure 2. The positive (a), (c) and negative samples (b), (d) used for training. (a) and (b) Training samples for the detector. (c) and (d) Training samples for the the boundary classifiers.

testing hypotheses by six orders of magnitude in our applications, which makes the directly training of the detector on 3D volumetric data a feasible procedure.

3.3. Learning the Boundary Classifiers

After we obtain the pose of the LV in the ED frame, we need to segment its boundary to automatically start the trackers. Active shape models (ASM) [4] is used to deform an initial estimate of a nonrigid shape. The non-learning boundary classifier using gradients in the original ASM does not work in our application due to the complex background and weak edges in 3D ultrasound. Learning based methods can exploit image evidences to achieve robust boundary detection. Boundaries with different orientation are usually detected on prerotated images [6]. Since 3D volume rotation is very time consuming, we use steerable features for boundary detection and avoid rotating the 3D volume. For each boundary point (289 in total), we sample several points from the volume around it under a special pattern, which embed the orientation information into the distribution of the sampling. A few local features for each sampling point, such as voxel intensity and gradients, etc., are calculated. The advantages of steerable features are that they combine the advantages of both local and global features.

Two boundary classifiers, one for LV motion close to the ED phase and the other for LV motion close to ES, are trained using probabilistic boosting tree (PBT) [22]. The PBT ensembles many strong classifiers into a tree structure. The widely used cascade boosting [24] can be treated as a special case in PBT. The learned boundary classifiers are used to automatically segment the boundary of LV, to initialize the trackers, and also used as the detection tracker for the following frame. In Figure 2 we show some positive and negative training samples used for training both the detector and the boundary classifiers.

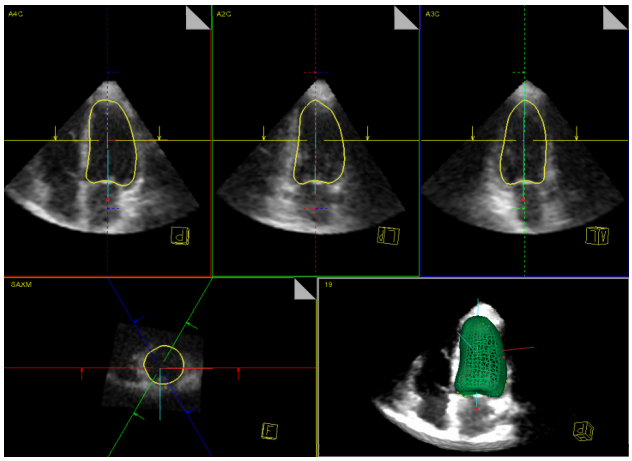


Figure 3. The four canonical view and 3D representations of the segmentation result (automatic tracking initialization) of LV.

4. Tracking Procedure

The tracking procedure on a testing sequence contains four steps. The $p(\mathbf{x}_0)$ is initialized using the learned ED detector and the ED boundary classifier. At time $t-1$, registration based reverse mapping and one-step forward prediction are used to estimate the next state $p(\mathbf{x}_t|\mathbf{x}_{t-1}, \mathbf{z}_{1:t-1})$. We then apply two collaborative trackers and robust data fusion to estimate the measurement density $p(\mathbf{z}_t|\mathbf{x}_t)$. In order to obtain smooth tracking of LV, each boundary point is mapped to a shape constrained 3D boundary. The final results are obtained by maximizing the posterior probability in (2). These *prediction* (1) and *updating* (2) steps are performed recursively for each frame in one sequence.

4.1. Initialization of Tracking

In order to initialize the boundary tracking of LV in an automatic manner, we need to automatically detect and segment LV in the ED frame of a testing sequence. Given the ED frame, all positions are scanned by the trained position detector. The top 100 candidates (x_i, y_i, z_i), $i = 1, 2, ..., 100$ are kept. Each candidate is then expanded using 1000 hypothesis on orientations $(\psi_{ij}, \phi_{ij}, \theta_{ij})$, $j = 1, 2, ..., 1000$. The trained position-orientation detector is applied for each candidate and the best 50 are kept. These 50 candidates are scaled using 1000 hypothesis $(sx_{kl}, sy_{kl}, sz_{kl})$, $k = 1, 2, ..., 50$, $l = 1, 2, ..., 1000$. Evaluated by the position-orientation-scale detector, the best 100 candidates are averaged to produce the final similarity transformation estimation.

After the similarity transformation is calculated, the LV mean shape is registered and superimposed on the ED frame

of the testing sequence as the initial position. For each boundary point we search ± 12 mm range on the normal directions of the boundary. (The value ± 12 mm was set considering both speed and accuracy.) The learned boundary classifier is used to move each boundary point to its optimal position where the estimated boundary probability is maximized. Figure 3 shows the tracking initialization result.

4.2. One-Step Forward Prediction

In this step we calculate the motion prior (state model) $p(\mathbf{x}_t|\mathbf{x}_{t-1}, \mathbf{z}_{1:t-1})$ using the learned motion modes. At time $t-1$, we first transform the current 3D shape $p(\mathbf{x}_{t-1}|\mathbf{z}_{1:t-1})$ to the corresponding frame of *each* motion mode in the 4D GPA coordinate system. Thin plate spline (TPS) transformation [16] is applied to perform this mapping. The TPS is a nonrigid transformation between two 3D point sets. The transformation T contains an affine mapping plus a wrapping coefficient matrix. We used 72 uniformly sampled points from 289 boundary points to estimate the TPS transformation T by minimizing

$$E_{TPS}(T) = \sum_{i=1}^{72} \|\mathbf{w}_i - T(\mathbf{b}_i)\|^2 + \lambda f(T) \qquad (5)$$

where $\mathbf{w}_i$ denote the 3D mesh point on the learned motion modes and $\mathbf{b}_i$ denote the points on the testing object's boundary. The $f(T)$ is a function containing a kernel which represents the internal structure relationship of the point set. The regularization parameter λ is chosen as 1.5. We refer the readers to [16] for more details.

The prediction is applied on the motion mode which minimizes the previous 1 to $t-1$ *accumulated* TPS registration errors

$$I = \arg\min_i \sum_{j=0}^{t-1} \min_T E_{TPS}(\mathbf{x}_j, \mathbf{q}_{i,j}, T) \qquad (6)$$

where $i = 1, ..., Q$ represents the number of motion modes. The $\mathbf{q}_{i,j}$ is the i-th motion mode in the j-th frame and the $E_{TPS}(\mathbf{x}_j, \mathbf{q}_{i,j}, T)$ is the registration error.

After we found the correct motion mode $\mathbf{q}_{I,t}$ using (6), the final prediction result in the real world coordinate system of the $p(\mathbf{x}_t|\mathbf{x}_{t-1}, \mathbf{z}_{1:t-1})$ is obtained using the reverse mapping T^{-1}. A motion mode generated by reverse mapping using TPS is shown in Figure 4.

There could be motion mode changes during the prediction when the prediction starts from one motion mode and jumps to another mode during tracking. This corresponds to the LV motion which starts from an ED shape in one learned motion mode, but has a motion trajectory close to another mode. This is the reason we apply the *accumulated* TPS registration error based one-step forward prediction. The algorithm provides accurate motion prior for boundary tracking.

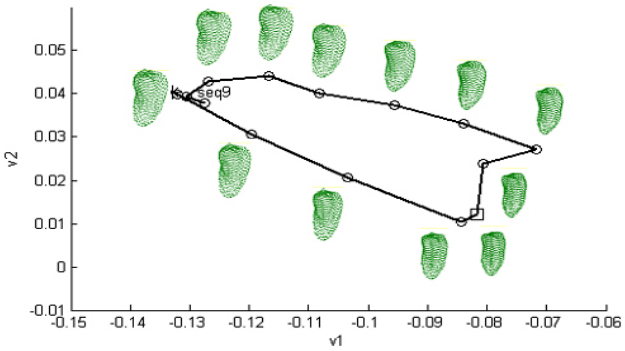


Figure 4. The LV boundaries in 3D world coordinates of a motion mode. The results are calculated using TPS reverse mapping and superimposed in the two-dimensional reduced subspace.

4.3. Collaborative Trackers

Given the motion prior $p(\mathbf{x}_t|\mathbf{x}_{t-1}, \mathbf{z}_{1:t-1})$ learned using one-step forward prediction on the motion manifold, for each boundary point the learned boundary classifiers are used to search in its ± 12 mm range on the normal direction. The optimal position is found by maximizing the boundary probability. The ED boundary classifier is used when the frame index is close to ED and the ES boundary classifier is used when it is close to ES. The final position using *detection* tracker is obtained by maximizing $p(\mathbf{z}_t|\mathbf{x}_t, r_1)$ in (3).

In order to compensate the disadvantages of detection tracking mentioned in the introduction, a 3D *template* tracker is also applied. Given $\mathbf{x}_t$ a 3D boundary point and its neighborhood $N(\mathbf{x}_t)$, let $G(\mathbf{x}_t, \mu)$ denotes the transformation of the template. (The neighborhood was chosen to be a $13 \times 13 \times 13$ cube based on experiments.) The goal is to search the best transformation parameters which minimize the error between $N(\mathbf{x}_t)$ and $G(\mathbf{x}_t, \mu)$.

$$\mu = \arg\min_{\mu} \sum_{\mathbf{x}_t \in N(\mathbf{x}_t)} [G(\mathbf{x}_t, \mu) - N(\mathbf{x}_t)]^2 . \qquad (7)$$

Because there is only a small change of parameter μ between adjacent frame, the minimization of (7) can be solved by linearizing the expression

$$G(\mathbf{x}_t, \mu) = G(\mathbf{x}_t) + \frac{\partial G(\mathbf{x}_t, \mu)}{\partial \mu} d\mu. \qquad (8)$$

At the end the result $p(\mathbf{z}_t|\mathbf{x}_t, r_2)$ is obtained.

Although the template matching algorithm is not robust and only works under the assumption of small inter-frame motions, it preserves temporal consistence and its disadvantages can be compensated by the one-step forward prediction and the detection tracker. Template updating is a key issue in template tracking. If we update the template in each frame only based on the previous template tracking result, the error will be inevitably accumulated and finally

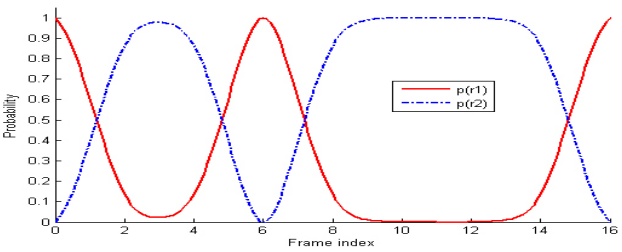


Figure 5. The prior $p(r1)$ for detection tracker (red solid line) and $p(r2)$ for template tracker (blue dotted line). The ED phase has frame index zero and ES phase is around frame six.

results in the template drifting mentioned in [3, 17]. Generally it is difficult for template tracking to recover from drifting without special processing. In our method we update the template using the previous collaborative tracking result. Because the learned motion prior is enforced and detection is used, this updating scheme can help the template tracker to recover from the template drifting. As shown in [20, 23, 28], learned motion prior is quite effective to help tracking to recover from failures. Collaborative kernel trackers are also successfully used in [9].

4.4. Data Fusion

Fusion of the collaborative tracking is obtained by defining prior distribution $p(r_1)$, $p(r_2)$ in (3). Based on domain expert's knowledge, both priors were designed as the exponential functions of t, which is illustrated in Figure 5. We show only one heart beat cycle which contains 17 frames. In order to reject outliers and achieve robustness, we apply a rectangle filter of $[-12\ 12]^3$ mm on the final data fusion results to erase the motion replacements which are larger than this size between adjacent frames. The corresponding position of the eliminated boundary point is recalculated based on the bicubic interpolation of its neighbors. After this step we obtained the $p(\mathbf{z}_t|\mathbf{x}_t)$.

4.5. Postprocessing and Projection

Due to speed considerations, the detection tracker is designed to search on the normal direction of the boundary, and the template tracker is searching along the direction of the gradients. Both of them can not provide smooth tracking results. Let B_f denotes the 3D boundary point-set after the data fusion step. We project B_f onto the PCA shape space calculated from the training stage, and obtain a smooth boundary point set B_s. A surface of the smooth boundary B_s is constructed using the 3D triangulation. Each boundary point P on B_f is projected onto the smooth surface by finding the triangle $S \in T = \{\text{all triangles of } B_s\}$ which minimize the square distance

$$dist(s,t) = [S(s,t) - P]^2 \tag{9}$$

Table 1. The *point-to-mesh* (PTM) errors measured in millimeters using three tracking algorithms.

	Mean	Variance	Median
3D optical flow	2.68	1.28	2.39
Tracking by detection	1.61	1.24	1.31
Collaborative trackers	**1.28**	**1.11**	**1.03**
	Min	Max	80%
3D optical flow	0.94	10.38	3.23
Tracking by detection	0.59	9.89	1.89
Collaborative trackers	**0.38**	**9.80**	**1.47**

where $S(s,t) = \mathbf{b} + s\mathbf{e}_0 + t\mathbf{e}_1$ with $(s,t) \in D = \{(s,t) : s \in [0\ 1], t \in [0\ 1], s+t \leq 1\}$. The $\mathbf{b}$ is one vertex in the triangle and $\mathbf{e}_0$ and $\mathbf{e}_1$ represent two edges. In this way, we keep the tracking results and achieve smooth motion tracking of the LV.

5. Experimental Results

We collected 67 annotated 3D ultrasound LV motion sequences. The 4D (x, y, z, t) motion sequences contain from 11 to 25 3D frames. In total we have 1143 ultrasound volumetric data. Our dataset is much larger than those reported in the literature, e.g., 29 cases with 482 3D frames in [13], 21 cases with about 400 3D frames in [19] and 22 cases with 328 3D frames in [29].

The imaging protocols are heterogeneous with different capture ranges and resolutions along each dimension. The dimensionality of 27 sequences is $160 \times 144 \times 208$ and the other 40 sequences is $160 \times 144 \times 128$. The x, y and z resolution ranges are $[1.24\ 1.42]$, $[1.34\ 1.42]$ and $[0.85\ 0.90]$ mm. In our experiments, we randomly select 36 sequences for training and the rest are used for testing.

The accuracy is measured by the *point-to-mesh* (PTM) error. All 3D points on each frame of the testing sequence are projected onto the corresponding annotated boundary of the test set. The projection distance from the point to boundary is recorded as the PTM error, e_{ptm}. For a perfect tracking, the e_{ptm} should be equal to zero for each 3D frame. In Table 1, we compared the quantitative e_{ptm} using our proposed algorithm, with tracking by 3D optical flow [8] and tracking by detection [26].

The 80% column in Table 1 represents the sorted 80% smallest error of all e_{ptm}, and is commonly used by doctors to evaluate the usability of the system. For example, if the doctor can tolerate an error of 1.5 mm, they normally expect 80% of the errors to be smaller than this number. The mean e_{ptm} we obtained is 1.28 mm with a 80% error below 1.47 mm. These are the best results in the literature as far as we know (mean error 1.45 mm in [29], 2.5 mm in [18] and 2.7 mm in [19]). Considering the range of resolution in the test set, we actually obtained subvoxel tracking accuracy on the

mean e_{ptm}.

The systolic-diastolic function is the volume-time curve which represents continuous LV volume change over the time t. It is an important diagnosis term to evaluate the health condition of the heart. In Figure 6, we illustrate one heart cycle of two systolic-diastolic functions. The curves for all three tracking algorithms and the ground-truth annotation are shown. Our algorithm (Tr. collab.) provides the most similar functions to the ground truth curves.

Two types of errors are frequent in tracking LV in 3D ultrasound. The first type is the leakage error, $e_{leakage}$, which happens on the mitral valve region. This is introduced by the similar appearance of the mitral valve to the LV boundary. A good LV boundary tracking algorithm should not follow the motion of the leaflets of the mitral valve. Tracking by detection failed on frame 8 (row 3, columns 3 and 4 in Figure 7b) because it always searches for what it learned in the training stage, but ignores all the local image information and temporal consistency.

The second type is the shrinkage error, e_{shrink}, which happens on the apex region. The 3D optical flow failed on the frame 6 (row 2, columns 5 and 6 in Figure 7c) because of the low image quality around the apex region. This proves that motion prior is necessary to obtain enough shrinkage for LV tracking in the 3D echocardiography because of the low quality of ultrasound imaging.

Using our algorithm, shown in Figure 7a, none of the errors are observed. We also provided two additional comparative sequences and seven tracking demos using the proposed algorithm in the supplementary materials of this paper.

The most important practical consideration for 3D tracking is computational complexity. One of the major reasons to propose marginal space learning, steerable features, registration based reverse mapping and one-step forward prediction is the speed. Our currently C++ implementation requires $1 - 1.5$ seconds per frame containing $160 \times 148 \times 208 = 4,925,440$ voxels and about 20 seconds for the whole sequence. Our implementation is at least two times faster than the slice-cut algorithm presented in [11], even if they are not working on 3D volumetric data directly, and about hundreds of times faster than [18] which reported using a MATLAB implementation.

6. Conclusions

In this paper, we presented a robust, fast and accurate LV tracking algorithm for 3D echocardiography. Our algorithm can process each 3D volume in less than 1.5 seconds and provides subvoxel accuracy. According to our knowledge, this is the first study reporting fast and reliable 3D ultrasound tracking of the left ventricle on a very large dataset. We demonstrated that collaborative trackers increase the tracking accuracy dramatically. The final accurate results

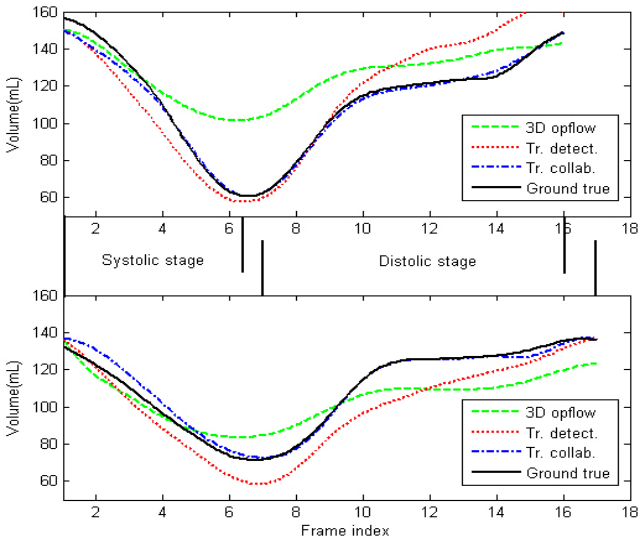


Figure 6. Two volume-time curves demonstrate the whole cardiac cycle, which includes the systole stage and the diastole stage.

are also achieved by applying the motion priors using one-step forward prediction. The robustness to complex background and weak edges is obtained from the learned discriminative detector and boundary classifiers. The temporal consistency is preserved by the template tracker. Instead of building specific models for the heart, all the major steps in our algorithm are based on learning. Our proposed algorithm is therefore general enough to be extended to other 3D medical tracking problems.

References

[1] S. Avidan. Ensemble tracking. *PAMI*, 29(2):261–271, 2007.

[2] A. Barbu, V. Athitsos, B. Georgescu, S. Boehm, P. Durlak, and D. Comaniciu. Hierarchical learning of curves application to guidewire localization in fluoroscopy. *CVPR*, 2007.

[3] J. Chen and Q. Ji. Online spatial-temporal data fusion for robust adaptive tracking. *CVPR*, 2007.

[4] T. F. Cootes, C. J. Taylor, D. H. Cooper, and J. Graham. Active shape models: Their training and application. *CVIU*, 61(1):38–59, 1995.

[5] M. Dewan, C. H. Lorenz, and G. D. Hager. Deformable motion tracking of cardiac structures (DEMOTRACS) for improved MR imaging. *CVPR*, 2007.

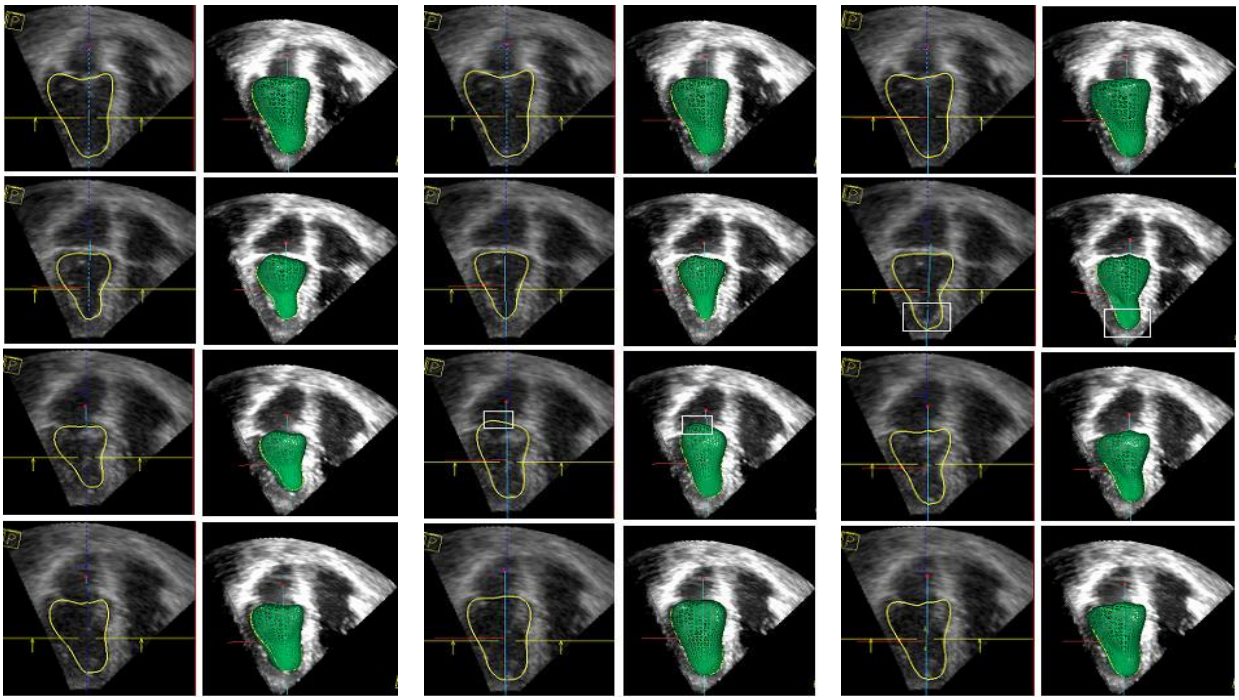
[6] P. Dollar, Z. Tu, and S. Belongie. Supervised learning of edges and object boundaries. *CVPR*, 2:1964–1971, 2006.

[7] I. L. Dryden and K. V. Mardia. *Statistical Shape Analysis*. John Wiley and Sons, 1998.

[8] Q. Duan, E. Angelini, S. Homma, and A. Laine. Validation of optical-flow for quantification of myocardial deformations on simulated RT3D ultrasound. *ISBI*, pages 944–947, 2007.

[9] Z. Fan, Y. Wu, and M. Yang. Multiple collaborative kernel tracking. *CVPR*, 2:502–509, 2005.

[10] P. F. U. Gotardo, K. L. Boyer, J. Saltz, and S. V. Raman. A new deformable model for boundary tracking in cardiac



(a) (b) (c)

Figure 7. Comparative tracking results on a testing sequence with 12 frames. (a) Tracking results using the proposed method. (b) Tracking by detection. (c) Tracking using 3D optical flow. The rows correspond to frame index 1, 6, 8 and 10.

MRI and its application to the detection of intra-ventricular dyssynchrony. *CVPR*, 1:736–743, 2006.

[11] W. Hong, B. Georgescu, X. S. Zhou, S. Krishnan, Y. Ma, and D. Comaniciu. Database-guided simultaneous multi-slice 3D segmentation for volumeric data. *ECCV*, 4:397–409, 2006.

[12] M. Isard and A. Blake. CONDENSATION — Conditional density propagation for visual tracking. *IJCV*, 28(1):5–28, 1998.

[13] M. P. Jolly. Automatic segmentation of the left ventricles in cardiac MR and CT images. *IJCV*, 70(2):151–163, 2006.

[14] Z. Khan, T. Balch, and F. Dellaert. An MCMC-based particle filter for tracking multiple interacting targets. *ECCV*, 4:279–290, 2004.

[15] Y. Li, H. Ai, T. Yamashita, S. Lao, and M. Kawade. Tracking in low frame rate video: A cascade particle filter with discriminative observers of different lifespans. *CVPR*, 2007.

[16] J. Lim and M. Yang. A direct method for modeling non-rigid motion with thin plate spline. *CVPR*, 1:1196–1202, 2005.

[17] I. Matthews, T. Ishikawa, and S. Baker. The template update problem. *PAMI*, 26(6):810–815, 2004.

[18] A. Myronenko, X. B. Song, and D. J. Sahn. LV motion tracking from 3D echocardiography using textural and structural information. *MICCAI*, 4792:428–435, 2007.

[19] F. Orderud, J. Hansgård, and S. I. Rabben. Real-time tracking of the left ventricle in 3D echocardiography using a state estimation approach. *MICCAI*, 4791:858–865, 2007.

[20] D. Ormoneit, M. J. Black, T. Hastie, and H. Kjellström. Representing cyclic human motion using functional analysis. *Image and Vision Computing*, 23(14):1264–1276, 2005.

[21] J. Tenebaum, V. de Silva, and J. Langford. A global geometric framework for nonlinear dimensionality reduction. *Science*, 290(5500):2319–2323, 2000.

[22] Z. Tu. Probabilistic boosting-tree: Learning discriminative models for classification, recognition, and clustering. *ICCV*, 2:1589–1596, 2005.

[23] R. Urtasun, D. J. Fleet, and P. Fua. Temporal motion models for monocular and multiview 3D human body tracking. *CVIU*, 104:157–177, 2006.

[24] P. Viola and M. J. Jones. Rapid object detection using a boosted cascade of simple features. *CVPR*, 1:511–518, 2001.

[25] A. Yilmaz, O. Javed, and M. Shah. Object tracking: A survey. *ACM Computer Survey*, 38(4):13.1–13.45, 2006.

[26] T. Zhao and R. Nevatia. 3D tracking of human locomotion: A tracking as recognition approach. *ICPR*, pages 1054–1060, 2002.

[27] Y. Zheng, A. Barbu, B. Georgescu, M. Scheuering, and D. Comaniciu. Fast automatic heart chamber segmentation from 3D CT data using marginal space learning and steerable features. *ICCV*, 2007.

[28] S. Zhou, J. Shao, B. Georgescu, and D. Comaniciu. Pairwise active appearance model and its application to echocardiography tracking. *MICCAI*, 4190:736–743, 2006.

[29] Y. Zhu, X. Papademetris, A. Sinusas, and J. S. Duncan. Segmentation of myocardial volumes from real-time 3D echocardiography using an incompressibility constraint. *MICCAI*, 4791:44–51, 2007.