Database-Guided Simultaneous Multi-slice 3D Segmentation for Volumetric Data

Wei Hong[2], Bogdan Georgescu[1], Xiang Sean Zhou[3], Sriram Krishnan[3], Yi Ma[2], and Dorin Comaniciu[1]

[1] Integrated Data Systems Department, Siemens Corporate Research, Princeton NJ 08540, USA
[2] Department of Electrical and Computer Engineering, University of Illinois at Urbana-Champaign, Urbana IL 61801, USA
[3] Siemens Medical Solutions, Malvern PA 19355, USA

Abstract. Automatic delineation of anatomical structures in 3-D volumetric data is a challenging task due to the complexity of the object appearance as well as the quantity of information to be processed. This makes it increasingly difficult to encode prior knowledge about the object segmentation in a traditional formulation as a perceptual grouping task. We introduce a fast shape segmentation method for 3-D volumetric data by extending the 2-D database-guided segmentation paradigm which directly exploits expert annotations of the interest object in large medical databases. Rather than dealing with 3-D data directly, we take advantage of the observation that the information about position and appearance of a 3-D shape can be characterized by a set of 2-D slices. Cutting these multiple slices simultaneously from the 3-D shape allows us to represent and process 3-D data as efficiently as 2-D images while keeping most of the information about the 3-D shape. To cut slices consistently for all shapes, an iterative 3-D non-rigid shape alignment method is also proposed for building local coordinates for each shape. Features from all the slices are jointly used to learn to discriminate between the object appearance and background and to learn the association between appearance and shape. The resulting procedure is able to perform shape segmentation in only a few seconds. Extensive experiments on cardiac ultrasound images demonstrate the algorithm's accuracy and robustness in the presence of large amounts of noise.

1 Introduction

Three dimensional imaging technologies such as ultrasound, MRI and X-ray are developing rapidly. While 3-D volumetric data contain much richer information than 2-D images, 3-D volumetric data is still not widely used in clinical diagnosis mainly because quantitative analysis by human is much more time-consuming than analyzing 2-D images. Thus, automatic segmentation of anatomical structures in 3-D volumetric data is extremely important to have a fast quantitative analysis and to increase the use of volumetric data in clinical practice.

Segmentation of structures in 2-D images or 2-D video sequences has been extensive studied [1, 2, 3, 4]. However, automatically processing 3-D volumetric data is much more challenging, due to the enormous amount of data and the resulting computational complexity. In the traditional formulation, segmentation is defined as a perceptual grouping task and solved through clustering or variational methods. However, as the difficulty of the desired segmentation increases, it becomes harder to incorporate prior knowledge into the grouping task. The 3-D active appearance model (3-D AMM) [5, 6] extends the 2-D AAM into 3-D. However, matching 3-D AAM to volumetric data is a non-linear optimization problem which requires heavy computation and good initialization to avoid local minima. Recently, segmentation methods based on prior knowledge learnt from large annotated databases through boosting [7, 8, 9] show promising performance on segmentation tasks with complex object appearance and noisy data. The advantage of boosting is that it can implicitly encode the large amount of prior knowledge relevant to the segmentation task and yield algorithms capable of running in real-time for 2-D images. However, there is no trivial way to implement it for 3-D volumetric data because the increase in dimension from two to three will dramatically increase the complexity of the algorithm.

Contributions. The main contribution of this paper is to propose a fast 3-D database-guided segmentation method that directly exploits expert annotation of the interest object in large databases. The key is to transform the 3-D learning problem into several 2-D learning problems solved simultaneously. By cutting multiple 2-D slices to represent a 3-D shape, the segmentation in 3-D is extremely accelerated. Haar-like rectangle features are used for appearance representation because they can be evaluated rapidly in 2-D by using the "integral images"[8]. It is difficult to directly use 3-D features and an "integral volume" due to the increased computational complexity. Also, the number of all possible 3-D features is much higher than the number of 2-D features, making the feature selection through boosting very difficult. Our method converts the 3-D problem into a 2-D problem while keeping most of the 3-D information. The computational complexity for evaluating features in our method is similar to the complexity for 2-D images. The 2-D features simultaneously obtained from all 2-D slices are used to solve two 3-D learning problems: 1. *Shape detection*, where a classifier is trained to distinguish between object appearance and non-object appearances (Section 3) and 2. *Shape inference*, where the association between an object appearance and its 3-D shape is solved by selecting the relevant features (Section 4).

The multiple slices of all 3-D shapes must be cut consistently according to their local coordinate systems. The local coordinates of each shape will be put in correspondence through shape alignment. Alignment of two 3-D shapes is in general a very hard problem because the meshes annotated by experts do not have *pairwise* correspondence. The task in our application is easier because some landmarks such as the principal axis and a representative plane (denoted by the A4C plane) are already known about the object of interest. The focus of this paper is segmentation of the left ventricle in 3-D ultrasound heart images however the method is general and can be applied to a wide range of anatomical object segmentation in volumetric data. In Section 2 we introduce an iterative algorithm to efficiently solve the alignment problem. Section 3 presents the method for shape detection followed by shape inference in Section 4 and experimental results in Section 5.

2 Non-rigid Linear 3-D Shape Alignment for Training Data

For all the shapes in the training database, the location, orientation, size, aspect ratio and non-linear deformation vary a lot (See Figure 1). The variation among these shapes must be eliminated to acquire their essential common characteristics and build their local coordinates.

Suppose that we have a mean shape which is the average of all the training shapes after alignment, all the training shapes must be aligned to this mean shape by transformations which will minimize the shapes variations. Ideally, a non-linear transformation can reduce the variation to zero. However, this transformation has to be searched at detection time. Thus, using an ideal non-linear transformation will considerably increase the search space. In our method, we only consider linear transformations, which provide a computationally feasible way to reduce the variation. The shape after the linear transformation will be very close to the mean shape and we denote it by the prototype of the original shape. The mean shape will be the average of all the prototypes.

Each training shape is represented by a 3-D triangle mesh annotated by experts. The mesh can be represented by a set of points (vertices), denoted as $P^0 \doteq \{\mathbf{X}_i = [X_i, Y_i, Z_i]^T \in \Re^3\}_{i=1}^N$ in world coordinates. N is the number of vertices of each mesh. For each point $\mathbf{X}$ on a shape, the corresponding point on the prototype shape is $\mathbf{x} \in \Re^3$ in its local coordinates. The prototype shape is denoted as $P \doteq \{\mathbf{x}_i = [x_i, y_i, z_i]^T \in \Re^3\}_{i=1}^N$. The mean shape is denoted as $\bar{P} \doteq \{\bar{\mathbf{x}}_i = [\bar{x}_i, \bar{y}_i, \bar{z}_i]^T \in \Re^3\}_{i=1}^N = \frac{1}{M}\sum P_j$. Among all the linear transformations, we assume that each shape is transformed to a prototype shape by rotating, translating, scaling and changing of aspect ratio. The linear transformation

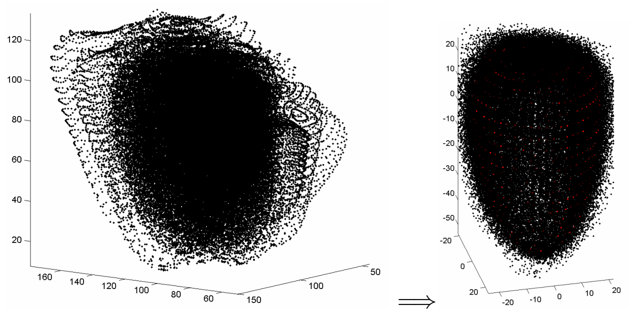

Fig. 1. Left: The location, orientation, size, aspect ratio and non-linear deformation of the shapes in the training set vary a lot. Right: The shapes aligned by proposed method.

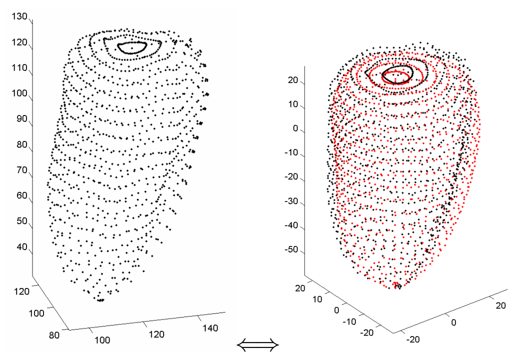

Fig. 2. Left: A shape used for training in world coordinates. Right: The prototype shape (black) and the mean shape (red) in their local coordinates.

between the original shape and its prototype (also between world coordinates and local coordinates) can be expressed as,

$$\mathbf{X} = RS\mathbf{x} + T, \tag{1}$$

$$\mathbf{x} = R^{-1}S^{-1}(\mathbf{X} - T), \tag{2}$$

where $R \in SO(3)$ is a rotation matrix, $S = \text{diag}[w, d, h] \in \Re^{3\times 3}$ is a scaling matrix and $T = [t_x, t_y, t_z]^T \in \Re^3$ is a translation vector. Figure 2 shows an example of a shape, its prototype shape and the mean shape. For each of the samples in the training set, the parameters of the transformation have to be estimated. A total of 9 parameters are needed to represent such a linear transformation, i.e., 3 parameters for R, 3 parameters for S and 3 parameters for T. In our data set of left ventricles, w and d are roughly equal. So we can simply set $w = d$ to reduce the total number of parameters to 8. h/w is defined to be the aspect ratio of the shape.

If the vertices of two shapes have pairwise correspondence, the distance of two shapes is defined as

$$\text{dist}(P_1, P_2) \doteq \sum_{i=1}^{N} \|\mathbf{x}_i^1 - \mathbf{x}_i^2\| \tag{3}$$

The problem of aligning all the shapes can be written as the following optimization problem:

$$\{\bar{P}, R_j, S_j, T_j\}_{j=1}^{M} \doteq \text{argmin} \sum_{j=1}^{M} \text{dist}(P_j, \bar{P}) = \sum_{j=1}^{M}\sum_{i=1}^{N} \|\mathbf{x}_i^j - \bar{\mathbf{x}}_i\| \tag{4}$$

Most existing methods for aligning two sets of 3-D data such as the popular Iterative Closest Point (ICP) [10] do not use any landmarks on the data. They also usually only consider rigid motion and ignore the changing of aspect ratio. The non-linear optimization in those methods is prone to local minima. For our problem domain, we know the extrema of the principal axis which pass through the center of the left ventricle and a plane named apical-four-chamber plane (A4C plane) which passes through all 4 chambers of the heart. However, our 3-D data do not have pairwise correspondences. So the optimized mean shape and transformation parameters $\{\bar{P}, R_j, S_j, T_j\}_{j=1}^{M}$ still cannot have a closed-form solution. We introduce an iterative linear method to solve the optimization problem.

- Step 1: In the first step, the principal axis which links the two apexes of the shape is aligned to the z-axis. For each shape, the rotation matrix R_a needed for this transformation is,

$$R_a \doteq R_2 R_1^T, \quad R_1 = [\mathbf{v}, \mathbf{w}, \mathbf{v} \times \mathbf{w}], \quad R_2 = [\mathbf{u}, \mathbf{w}, \mathbf{u} \times \mathbf{w}], \quad \mathbf{w} = \mathbf{v} \times \mathbf{u}, \tag{5}$$

  where $\mathbf{u}$ is the normalized principal axis vector and $\mathbf{v} = [0, 0, 1]^T$.
- Step 2: After the first step, all the shapes still have different angles of rotation along the z-axis. For each shape of a left ventricle, the A4C plane can determine the rotation along the z-axis. So in the second step, we rotate each mesh along its principal axis by certain degree so that the A4C plane of the left ventricle matches the x-z plane. The rotation is denoted as R_z,

$$R_z \doteq \begin{bmatrix} \cos\theta & \sin\theta & 0 \\ -\sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix}, \tag{6}$$

  where θ is the angle between the A4C plane and the x-z plane. The estimated rotation matrix of each shape should be $R = R_a * R_z$.
- Step 3: For all the meshes annotated by the experts, the vertices do not have one-to-one correspondence between two meshes except the two apexes. The points of the shape should correspond to the same physical points of the left ventricle. However it is impossible to determine automatically which points correspond the same physical points. So after we align the orientation of each mesh, we just move the centroid of each mesh to the origin of the coordinates and roughly evenly re-sample each mesh using polar coordinates. The re-sampled points will approximately have pairwise correspondences.
- Step 4: The mean shape $\bar{P}$ is calculated by $\bar{P} \doteq \frac{1}{M} \sum P_j$.
- Step 5: Finally, we need to align the position, scale and aspect ratio of all the shapes. For each shape, the parameters of S and T can be determined by solving the following equation,

$$\begin{bmatrix} \sum_{i=1}^{N}(x_i^2 + y_i^2) & 0 & \sum_{i=1}^{N}(x_i + y_i) & 0 & 0 \\ 0 & \sum_{i=1}^{N} z_i^2 & 0 & 0 & \sum_{i=1}^{N} z_i \\ \sum_{i=1}^{N} x_i & 0 & N & 0 & 0 \\ \sum_{i=1}^{N} y_i & 0 & 0 & N & 0 \\ 0 & \sum_{i=1}^{N} z_i & 0 & 0 & N \end{bmatrix} \begin{bmatrix} w \\ h \\ t_x \\ t_y \\ t_z \end{bmatrix} = \begin{bmatrix} \sum_{i=1}^{N}(x_i \bar{x}_i + y_i \bar{y}_i) \\ \sum_{i=1}^{N} z_i \bar{z}_i \\ \sum_{i=1}^{N} \bar{x}_i \\ \sum_{i=1}^{N} \bar{y}_i \\ \sum_{i=1}^{N} \bar{z}_i \end{bmatrix}, \tag{7}$$

  where $\mathbf{x} = [x, y, z]^T$, $\bar{\mathbf{x}} = [\bar{x}, \bar{y}, \bar{z}]^T$ and the estimated $S = \mathrm{diag}[w, w, h]$, $T = [t_x, t_y, t_z]$.

Steps 2 and 3 only need to be performed only once. Steps 1, 4 and 5 will be iterated until the change of the parameters is below a threshold determined by the working resolution.

After the convergence of the algorithm, the prototypes $\{\bar{P}\}_{j=1}^{M}$ of all shapes and their local coordinate transformations $\{R_j, S_j, T_j\}_{j=1}^{M}$ will be used to cut the multiple slices.

3 3-D Object Detection

The detection method we are using is based on boosted cascade of simple features, which has been widely used for real-time object detection[11, 8, 9]. However, most of such detection methods are only applied on 2-D images. Extending those methods to 3-D is not trivial and several problems must be addressed.

3.1 Multi-slice Representation of 3-D Data

One of the most popular sets of features for object detection is the Haar-like rectangular features, which can be computed very efficiently through the "integral images" [8]. However, in our problem domain, the enormous amount of 3-D data makes computing these features difficult. For example, for an 24×24-pixel 2-D image, there are already more than 180 thousand possible rectangle features. If we extend the rectangular features to cubical features, the number of possible features will be several million. Also, computing an "integral volume" is much slower than computing an integral image.

To avoid the difficulty in dealing with 3-D data directly, we first represent the 3-D data by 2-D slices simultaneously cut from the volume. These slices should be perpendicular to the surface of the shape in order to make the slices sensitive to the changing of the shape. So in the local coordinates we built in section 2 for each shape, we cut its prototype into vertical slices through the z-axis at different angles from the $x - z$ plane and cut horizontal slices at different z. For example, in Figure 3, there are two vertical slices at angle 0 and 90 degree from $x - z$ plane and one horizontal slice at $z = 0$. These three slices are already sufficient for detection purposes because any changes in the location, orientation or aspect ratio will cause large changes in at least one slice, since the slices coincide with the three coordinate axes. However, for the shape inference in Section 4, more slices are preferred to achieve better accuracy.

Rectangle features are computed simultaneously for each of the slice as shown in Figure 4. Features from all the 2-D slices consist the feature vector for the 3-D volume. The "integral image" is adopted to accelerate the computation of the features. The "integral mask" proposed in [9] is also needed to ensure correct computation for invalid image regions.

Training a detection classifier requires positive and negative samples. For the positive samples, the slices are obtained in local coordinates from the correct transformation we obtained in section 2. Several positive examples are shown in Figure 5 to demonstrate that the multiple slices capture the variations of the 3-D shapes. Some perturbation will be added to the correct transformation to

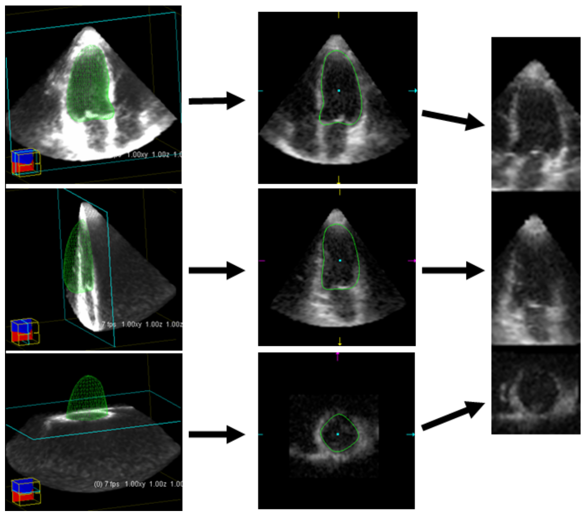

Fig. 3. An example of the slices that represent the 3-D volume. Two vertical slices are cut at 0 and 90 degree from the $x - z$ plane. One horizontal slice is cut at $z = 0$.

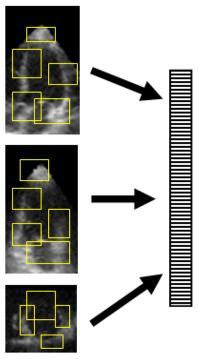

Fig. 4. Rectangle features are computed independently for each of the slice. Features from all the 2-D slices define the feature vector for the 3-D volume.

generate negative samples. Figure 6 shows positive samples and negative samples generated from one volume.

3.2 3-D Shape Detection

The detection of a 3-D shape is equivalent to finding the correct transformation between world coordinates and local coordinates of this object. From Equation 1, the transformation is determined by R, S and T, which contain 8 transformation parameters $[\omega_x, \omega_y, \omega_z, w, h, t_x, t_y, t_z]$, where ω_x, ω_y and ω_z are three Euler angles. Exhaustive searching in an 8 dimensional space would be very time-consuming. In the left ventricle detection problem, the A4C plane (i.e., the $x - z$ plane of local coordinates) is usually easy annotated by human or other

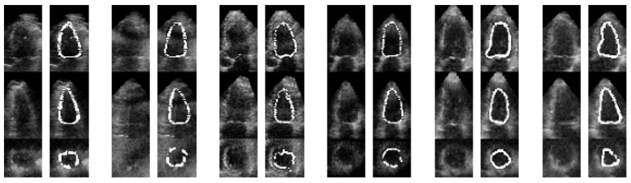

Fig. 5. Several positive training samples. The multiple slices capture the variations of the 3-D shapes. There are two columns of slices for each sample. The left columns show the slices and the right columns show the slices with the meshes annotated by experts.

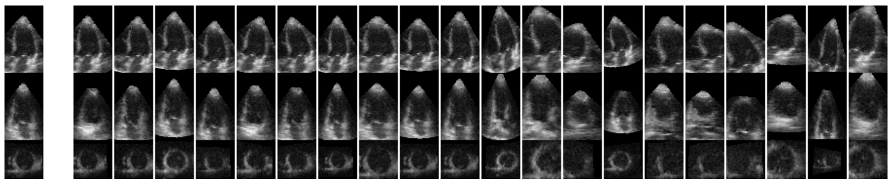

Fig. 6. Positive samples and negative samples generated from one volume. Only the leftmost one is a positive sample generated by correct local coordinates. All others are negative samples generated by incorrect local coordinates.

automatic detection. So we will assume the A4C is known so that we only need to search inside the A4C plane. Suppose that we know the normal vector $\mathbf{n}$ and a point $\mathbf{b}$ on the A4C plane. The A4C plane will be,

$$\mathbf{n}^T(\mathbf{x} - \mathbf{b}) = 0. \tag{8}$$

Suppose the initial transform R, S, T are R_0, S_0 and T_0. The initial rotation R_0 must satisfy that the y-axis of local coordinates is the normal of the A4C plane. It can be determined as,

$$R_0 = [\mathbf{u}_1, \mathbf{u}_2, \mathbf{u}_3], \quad \mathbf{u}_2 = \mathbf{n},\ \mathbf{v} = [0, 0, 1]^T,\ \mathbf{u}_1 = \mathbf{u}_2 \times \mathbf{v},\ \mathbf{u}_3 = \mathbf{u}_1 \times \mathbf{u}_2. \tag{9}$$

The initial scale matrix S_0 can be set to be the mean scale S_m for the training set. But the initial translation vector T_0 cannot be the mean translation vector T_m because it may not be on the A4C plane. So we will use the projection of T_m on the A4C plane as T_0, i.e.,

$$T_0 = T_m + \mathbf{n}\mathbf{n}^T(\mathbf{b} - T_m). \tag{10}$$

During the search, we will change the initial transformation R_0, S_0, T_0 by another relative transformation R_r, S_r, T_r. Since we need to fix our transformation inside the A4C plane, the only possible relative rotation is along y-axis.

$$R_r = \begin{bmatrix} \cos\omega_{ry} & 0 & \sin\omega_{ry} \\ 0 & 1 & 0 \\ -\sin\omega_{ry} & 0 & \cos\omega_{ry} \end{bmatrix}. \tag{11}$$

The relative translation $T_r = [t_{rx}, 0, t_{ry}]^T$ is a translation on the A4C plane. The relative scale matrix is $S_r = \begin{bmatrix} w_r & 0 & 0 \\ 0 & w_r & 0 \\ 0 & 0 & h_r \end{bmatrix}$, where w_r is the changing of the width and h_r is the changing of the height.

So the overall transform from the prototype to the shape is,

$$\begin{aligned} \mathbf{x} &= RS\mathbf{x}_0 + T = R_0R_r((S_0 + S_r)\mathbf{x}_0 + T_r) + T_0 \\ &= R_0R_r(S_0 + S_r)\mathbf{x}_0 + R_0R_rT_r + T_0. \end{aligned} \tag{12}$$

$$R = R_0R_r,\ S = S_0 + S_r,\ T = R_0R_rT_r + T_0 = RT_r + T_0. \tag{13}$$

The searching for $[\omega_{ry}, w_r, h_r, t_{rx}, t_{rz}]$ will be in a 5-dimensional space and will be performed on a coarse to fine fashion. In the coarse stage, the search range is determined by the statistics of the training data. After finding the maximum response of the detection classifier, a new iteration of searching will be performed. The initial point will be located at the maximum response found in the coarse stage and the search range will be reduced by half.

4 3-D Non-rigid Shape Inference

The problem is now to determine the shape associated with the detected object. This task is solved by finding the relevant features which best describe the non-rigid variations of the shapes around the mean. All the prototype shapes $\{P_j\}_{j=1}^M$ are clustered into K classes $\{C_i\}_{i=1}^K$ as shown in Figure 7 by the K-means algorithm. The rectangle feature vectors of each shape are acquired by the multi-slice presentation in Section 3.1. The best features vectors for each class $\{\mathbf{f}_i\}_{i=1}^K$ that discriminate these classes of shapes are selected by the forward sequential feature selection [9]. For each input volume, we first find the linear non-rigid transformation of the shape by the detection method in Section 3.2. In local coordinates

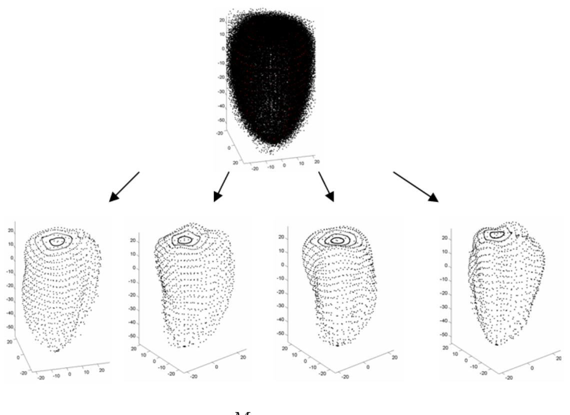

Fig. 7. All the prototype shapes $\{P_j\}_{j=1}^M$ are clustered into K classes $\{C_i\}_{i=1}^K$ by the K-means method

of the shape, the multiple slices are cut from its prototype to generate a query feature vector $\mathbf{f}_q$ for the shape in this volume. The distance of the query and a reference is,

$$d(\mathbf{f}_q, \mathbf{f}_r) = (\mathbf{f}_q - \mathbf{f}_r)^T \Sigma (\mathbf{f}_q - \mathbf{f}_r), \tag{14}$$

where $\mathbf{f}_q$ and $\mathbf{f}_r$ are the feature vectors of the query and the reference respectively. Σ is the Fisher linear discriminating matrix [12] learnt from the training samples.

The inferred shape $\hat{P}$ is computed by Nadaraya-Watson kernel-weighted average [13, 14] of the K prototype classes,

$$\hat{P} = \frac{\Sigma_{i=1}^{K} K_h(\mathbf{f}, \mathbf{f}_i) C_i}{\Sigma_{i=1}^{K} K_h(\mathbf{f}, \mathbf{f}_i)}, \tag{15}$$

where K_h is the Epanechnikov kernel [15] defined as,

$$K_h(\mathbf{f}, \mathbf{f}_i) = \begin{cases} \frac{3}{4}\left(1 - \frac{d(\mathbf{f},\mathbf{f}_i)}{d(\mathbf{f},\mathbf{f}_{[h]})}\right), \text{ for } \frac{d(\mathbf{f},\mathbf{f}_i)}{d(\mathbf{f},\mathbf{f}_{[h]})} \leq 1; \\ 0, \quad \text{otherwise}, \end{cases} \tag{16}$$

where $\mathbf{f}_{[h]}$ is the feature vector which has h^{th} smallest distance to $\mathbf{f}$.

5 Experiments

The proposed segmentation method was tested on two sets of 3-D ultrasound cardiac volumes of size $160 \times 144 \times 208 = 4,792,320$ voxels. The End-Diastolic(ED) set consists of 44 volumes and the End-Systolic(ES) set consists of 40 volumes. For each volume in the training sets, the A4C plane and a mesh of left ventricle with 1,139 vertices are annotated by experts.

We first demonstrate results of the 3-D shape alignment method introduced in Section 2. Figure 8 shows the distances between each aligned shape and the mean shape. The distance between two shapes is defined in Equation 3. The variation among the shapes is very small after the shape alignment.

Figure 9 illustrates the effectiveness of the 3-D multi-slice detection method described in Section 3.2 by leave one out method. The four curves indicate

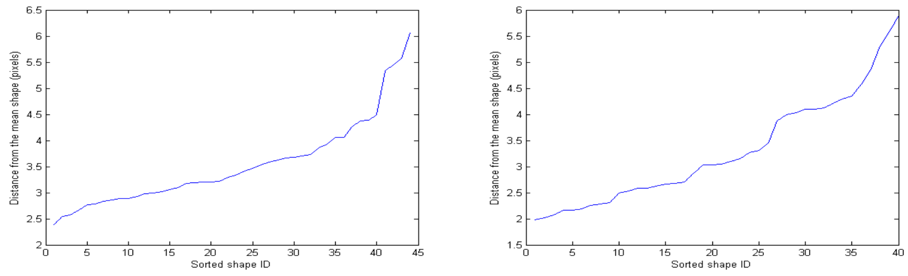

Fig. 8. The distances between each aligned shape and the mean shape. The shapes are sorted by their distances to the mean shape. Left: Results for ED volumes. Right Results for ES volumes.

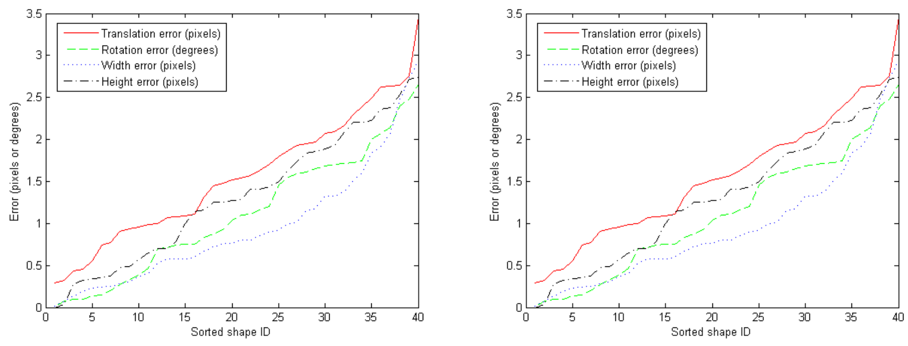

Fig. 9. The error of the translation, rotation, width and height for the left ventricle detection. Left: Results for ED volumes. Right Results for ES volumes.

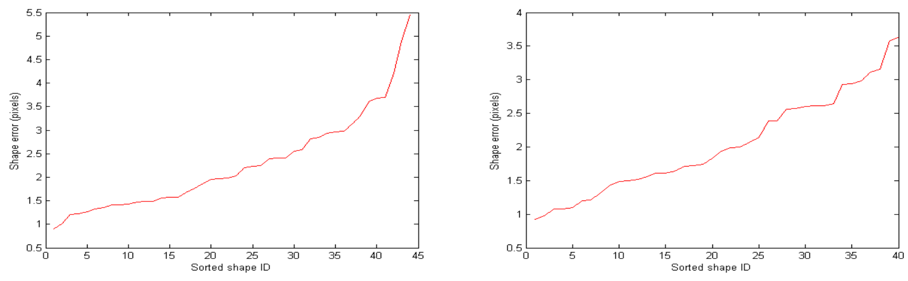

Fig. 10. The error of the entire segmentation procedure. Left: Results for ED volumes. Right Results for ES volumes.

errors of the translation $\|(\Delta t_x, \Delta t_z)\|$, rotation $|\Delta\omega_y|$, width $|\Delta w|$ and height $|\Delta h|$ respectively. The ground truth of these parameters are obtained by the shape alignment in Section 2.

The error of the entire segmentation procedure is shown in Figure 10. The automatic segmentation result of each volume is compared with the mesh annotated by experts. The error of the segmentation is the distance of the inferred shape and the ground truth in world coordinates. This error contains error both from the detection and shape inference.

The results in Figure 8 can be thought of as the error of another segmentation method which uses the ground truth of detection but does not use any shape inference. All the shapes are assumed to be the same as the mean shape. Comparing Figure 10 and Figure 8, we can conclude that the shape inference largely reduces the shape error even when the detection is not perfect.

Figure 11 shows additional segmentation results from volumes which do not have the meshes drawn by experts. The meshes found by our segmentation method visually fit the borders of the left ventricles very well.

The whole segmentation procedure takes about 3 seconds for each volume with $4,792,320$ voxels on a Xeon 2.8GHz machine. Our algorithm is faster than

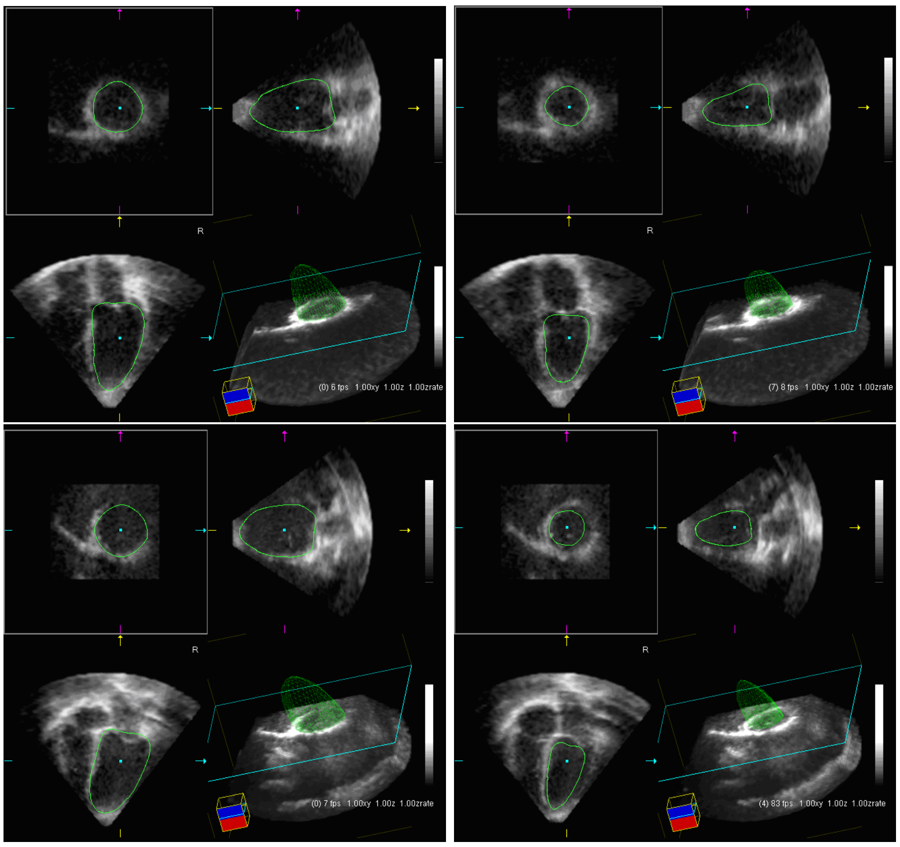

Fig. 11. Results from volumes which do not have the meshes drawn by experts. The meshes found by our segmentation method visually fit the boarders of the left ventricles very well.

3-D AAM models proposed in [5, 6]. Mitchell's work [5] requires 2-3 minutes for each frame on MRI data. Stegmann's 3-D AAM [6] takes 3.4 seconds on MRI data with 22,000 voxels.

6 Limitations and Future Work

We have proposed a fast method for segmenting anatomical objects from 3-D volumetric data. It overcomes the difficulty of working directly with 3-D data by simultaneously solving several 2-D problems. The method is learning-based and directly exploits expert annotations in medical databases.

Due to the difficulty of annotating 3-D shapes by hand, our training and testing sets are not very large or all-inclusive. In the future, more data will be collected and used to validate our method. In fact our method should only benefit from more training data. For the shape detection, it is also possible to utilize other local features or detection methods to generate a better initialization and reduce the searching range. For the non-rigid shape alignment, so far only linear transformations are considered. Integrating non-linear transformations might capture more complex variation of shapes.

References

1. Kass, M., Witkin, A., Terzopoulos, D.: Snakes: Active contour models. International Journal of Computer Vision **2** (1988) 321–331
2. T.F.Cootes, Edwards, G., C.J.Taylor: Active appearance models. In: Proc. 5th European Conference on Computer Vision. Volume 2. (1998) 484–498
3. Shi, J., Malik, J.: Normalized cuts and image segmentation. IEEE Transactions on Pattern Analysis and Machine Intelligence **22** (2000) 888–905
4. Comanicu, D., Meer, P.: Mean shift: A robust approach toward feature space analysis. IEEE Trans. Pattern Anal. Machine Intell. **24** (2002) 603–619
5. Mitchell, S., Bosch, J., Lelieveldt, B., van der Geest, R., Reiber, J., Sonka, M.: 3-d active appearance models: segmentation of cardiac mr and ultrasound images. IEEE Transactions on Medical Imaging **21** (2002) 1167– 1178
6. Stegmann, M.B.: Generative Interpretation of Medical Images. PhD thesis, Technical University of Denmark (2004)
7. Freund, Y., Schapire, R.E.: A decision-theoretic generalization of on-line learning and an application to boosting. In: European Conference on Computational Learning Theory. (1995) 23–37
8. Viola, P., Jones, M.: Rapid object detection using a boosted cascade of simple features. In: Proceedings IEEE Conf. on Computer Vision and Pattern Recognition. (2001)
9. Georgescu, B., Zhou, X., Comaniciu, D., Gupta, A.: Database-guided segmentation of anatomical structures with complex appearance. In: Proceedings IEEE Conf. on Computer Vision and Pattern Recognition. (2005)
10. Besl, P.J., McKay, N.D.: A method for registration of 3-D shapes. IEEE Trans. Pattern Anal. Mach. Intell. **14** (1992) 239–256
11. Papageorgiou, C., Poggio, T.: A trainable system for object detection. Int. J. Comput. Vision **38** (2000) 15–33
12. Duda, R.O., Hart, P.E.: Pattern Classification and Scene Analysis. Wiley-Interscience (1973)
13. Nadaraya, E.A.: On estimating regression. Theory Prob. Appl. **10** (1964) 186–190
14. Watson, G.S.: Smooth regression analysis. Sankhy a, Series A **26** (1964) 359–372
15. Epanechnikov, V.: Nonparametric estimates of a multivariate probability density. Theory of Probability and its Applications **14** (1969) 153–158