Interactive Segmentation and Analysis of Fetal Ultrasound Images

Kalpathi R.Subramanian *Dina M. Lawrence* *M. Taghi Mostafavi*

Department of Computer Science
The University of North Carolina at Charlotte
Charlotte, NC 28223
{krs,dmlawren,taghi}@mail.cs.uncc.edu

Abstract. The ability of ultrasound scanners to image anatomical structures in real time have led to their use in two important applications of medicine, (1) for monitoring the unborn baby (fetal ultrasound), and, (2) coronary treatment of blockages in blood vessels (intravascular ultrasound). The generated images (in the form of a continuous video) are typically noisy and contain numerous artifacts, making it difficult to isolate and measure features of interest. We explore the use of two algorithms, region growing and a variant of split/merge algorithm for segmenting sequences of fetal ultrasound images. We describe an interactive system that can rapidly process and segment an arbitrary number of features. The system exploits frame to frame coherence for accelerating the segmentation process, while at the same time combining the strengths of these algorithms and some post-processing for accurate and robust detection of features.

Introduction

The use of medical imaging scanners for routine clinical diagnosis and treatment planning has gained wide acceptance in medicine in recent years, primarily because the technology is non-invasive (CT, MRI) or minimally invasive (intravascular ultrasound). The most common imaging modalities have been CT (Computer Tomography), MRI (Magnetic Resonance Imaging) and Ultrasound. Ultrasound scanners have a number of advantages over CT and MRI. Perhaps the most important among them are their ability to visualize anatomy easily and at interactive speeds. Ultrasound has found wide application in fetal diagnosis and coronary treatment [6]. Today, almost all pregnant mothers undergo at least one ultrasound scan, during the gestation period, for preventive reasons or early detection of problems relating to the health of the baby or the mother. Coronary (or intravascular) ultrasound is targeted at detection and treatment of atherosclerosis, the buildup of cholesterol induced blockages in the coronary vessels.

Computer assisted processing, analysis and archival of ultrasound images presents new possibilities for improved diagnosis and treatment planning. Critical segments of an ultrasound scan could be processed and archived for review at a later stage, or for training or instructional purposes. In this age of information technology, such data could be made available to physicians at remote sites (via Internet, for instance). Archiving such data using computer technology (as opposed to analog media such as video tape or film) would also make it a more efficient process, since only important segments of a scan need to be archived, and retrieval is considerably faster.

Processing ultrasound images for feature identification and analysis poses difficulties (in contrast to CT or MRI) due to the poor quality of the images[3][4]. Ultrasound images are noisy, possess poor contrast, suffer from variations in illumination and from self shadow problems that result in masking features of interest. The constant motion of the baby is another distinguishing problem faced by computer based processing, unlike CT and MRI scans (with the exception of the beating heart). Motion induced artifacts need to be compensated for before application of any image processing operations to analyze the images. Conventional image processing operators often work very poorly on ultrasound images. Thus, it is important to investigate robust techniques that can reliably enhance and process ultrasound data, for analysis and quantification of anatomical features of interest.

Earlier work on ultrasound image processing and analysis has focused on 3D reconstruction [8, 10, 12, 11]. Unlike CT or MRI, where 3D data can easily be acquired from tomography machines (by stacking a sequence of parallel 2D images), ultrasound scanners produce a video signal that needs to be discretized to obtain 2D images; however, these images are arbitrarily oriented and special 3D sensors will need to be used to obtain spatial information to orient and locate each 2D image [8]. The problems associated with this process have limited the use of such methods to medical research centers or hospitals.

In this article, we describe the development of a new 2D interactive system targeted at processing and segmenting sequences of fetal ultrasound images. The project, in collaboration with Carolinas Medical Center (CMC) [1], is aimed at providing technicians a simple tool that would allow rapid identification and quantification of features of interest over selected segments of an ultrasound scan. The system exploits two ideas for rapidly processing long sequences of 2D ultrasound images, (1) frame to frame coherence, and (2) allow the user to specify regions of interest over which the processing algorithms operate. In this work, we have investigated region growing and a variant of the split/merge algorithms, individually and in combination to segment ultrasound images. We have found out that using the two algorithms in combination with some postprocessing produces more accurate segmentation of the ultrasound images. Preliminary tests of the system indicate that it is possible to segment sequences of up to 50 images (with 2-5 regions of interest) in about 5-6 minutes on standard Unix workstations.

1 Background

1.1 Image Acquisition

Unlike CT and MRI, ultrasound scans are made using high frequency sound waves. In OB/GYN applications, an external probe is used as the source. Reflections of the emitted waves are measured by the sensors in the probe used to perform the scan. Materials with differing impedances cause reflections of varying intensities, which, when combined with their time of arrival at the sensor, helps in generating a 2D image. The strength of the reflection determines the intensity of the image pixels, while their arrival time determines their spatial location in the image. The received signal undergoes internal processing and is then converted into a video format (NTSC, RGB, component, etc) for display on a standard monitor or for recording. The raw images are usually grayscale images, although some scanners provide pseudo coloring for highlighting and labeling various features in the data.

1.2 Processing and Segmentation

For a variety of reasons, ultrasound images contain numerous artifacts [5, 1] that poses significant challenges to computer processing. Poor resolution (axial and lateral), reverberation (multiple

[1] CMC is a statewide hospital authority in Charlotte, NC and includes a dedicated research center.

reflections) and shadowing are some causes of artifacts in ultrasound images. The motion of the baby further complicates the processing algorithms, introducing considerable blurring as well as loss of coherence from frame to frame. With the amount of noise present in ultrasound images, it is essential to perform some amount of smoothing prior to segmentation and feature detection. Both gaussian and median filters can be used to compensate for noise [11]; median filter has been found to be effective in removing speckle noise, which is common in these images. The kernel size to be used is largely based on trial and error, as too large a kernel overly blurs the images with corresponding loss of contrast. Some level of smoothing will need to be applied, else the algorithms that are targeted at feature identification will be highly sensitive to noise. Compensation for motion induced artifacts will be addressed in a companion paper. It is an important issue that needs to be dealt with properly, especially when attempting to segment the heart.

Once the image has been compensated for noise, the next step is to identify objects in the image. The process begins by segmenting the image into disjoint regions. Segmentation is a very difficult problem, and, to date, automating this process has been restricted to the simplest of all images or to organ and disease specific images. Thus, some form of human intervention is needed to assure a high level of confidence in this procedure. An extensive review of segmentation methods can be found in [2].

Two popular techniques to perform segmentation is through 'region growing' and the 'split/merge' algorithms [4]. In the region growing algorithm, a seed pixel is chosen by the user and its intensity is compared to those within a small neighborhood. A similarity measure (intensity tolerance, for instance) decides if any of the neighbors belong to the region of the seed pixel. Accepted neighbors recursively test their neighbors in a similar manner, resulting in a gradual growth of the region towards its boundary. The sensitivity of the algorithm directly depends on the sharpness of the region boundaries. Any weak or missing edge points on the boundary can cause the region to 'bleed' into neighboring regions.

In the split/merge algorithm, a section of the image containing the region of interest is selected by the user. A homogeneity condition is applied to the region's pixel intensities to check if they are sufficiently similar (for instance, the deviation from the mean pixel intensity of the region could be tested to be within a threshold). If this test fails, the region is partitioned into some number of equal sized partitions (a quadtree partitioning is common) and the homogeneity measure is applied to the partitioned regions. This process continues until all regions are classified to be homogeneous. Since the partitioning is performed in a somewhat independent fashion, it is very likely that adjacent regions of two independently partitioned regions could possibly be merged into a single region. Thus, the splitting procedure is followed by a merging step, resulting in merging adjacent regions (which is also a recursive procedure, allowing merged regions to be merged again, if necessary).

The region growing algorithm is a topologically sensitive algorithm and highly sensitive to missing points in the boundary. In addition, it is incapable of detecting disconnected regions belonging to the same object. The split/merge algorithm is a more global approach to determining the regions of an object and uses statistical measures of a region to determine region membership. Such complementary properties of these two algorithms might be used in combination to produce a more robust and accurate segmentation method. At the same time, due to variations in illumination and other artifacts in ultrasound images, both algorithms can leave holes within features, introducing errors in estimating feature size. Morphological operators [4, 3] can be used to compensate for this. The two basic operators, *erosion* and *dilation* can either erode or grow the boundary of a region by an amount determined by the structuring element used. Dilation can be used to fill small holes in images, while erosion can be used to reverse the effects of dilation (for maintaining the size of the region, for instance). Finally, the split/merge algorithm, because of its global nature, can classify disconnected clusters of pixels as belonging to the same object. Where applicable, this can be compensated by the use of a connected component labeling procedure (to be described in Secion 2.2).

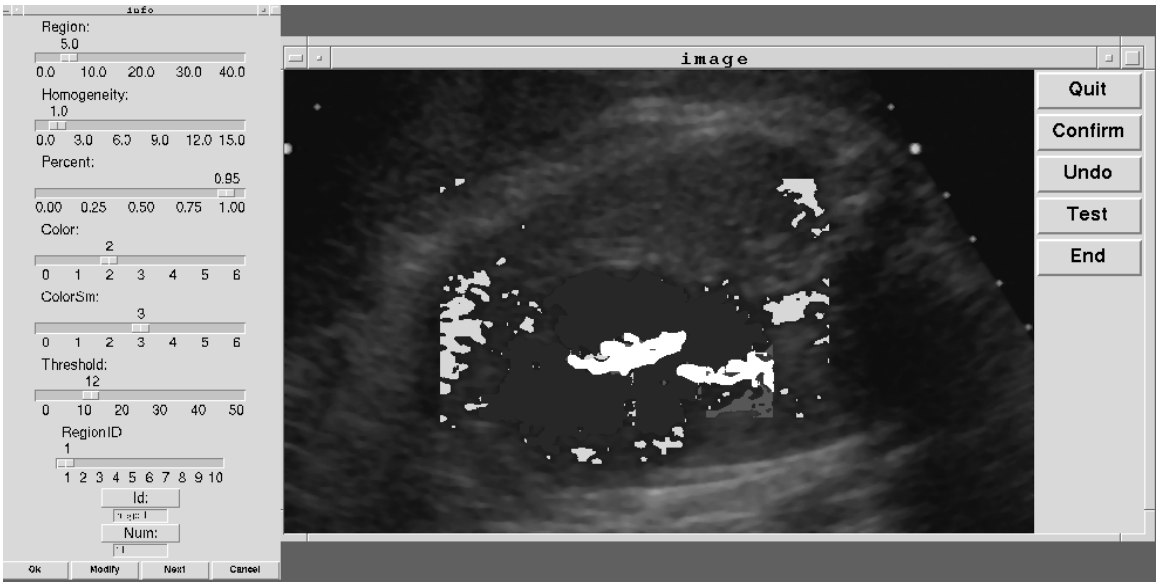

Figure 1: 2D Segmentation System Interface

Segmentation is typically followed by computing descriptors for the identified objects. Descriptors could be based on the boundaries of the objects, region based, topological or textural. For ultrasound images, quantification of the size, location and orientation of objects in the image are useful descriptors.

2 2D Segmentation System Description

The current implementation of the system is on Silicon Graphics workstations; A snapshot of the user interface is illustrated in Figure 1, when using the overlay algorithm, which allows application of both region growing and split/merge in sequence. The user interface is built using the Tcl/Tk toolkit [9], which is publicly available for a variety of architectures, including PCs. All of the processing modules are implemented as Tcl commands and are called from the interface.

2.1 The User View

A user session begins by reading in a sequence of acquired ultrasound images (individual images may also be read separately). The sequence can be smoothed using either a Gaussian or median filter. The user needs to specify the width and deviation of the Gaussian kernel or just the width for the median filter. Once the smoothing is completed, then the sequence can be segmented using (1) Region Growing (2) Split/Merge, or (3) Overlay, where the two algorithms are applied successively, in any order. In each case a rectangular region of interest is first selected to restrict the area of the image over which the segmentation algorithms operate. After a segmentation algorithm is chosen, the region is segmented. Any number of regions can be segmented in this manner. Once the segmentation is completed for the current image, the user can move on to the next image in the sequence, at which point the parameters used to segment the current image are automatically applied to the new image. If this is satisfactory, the user continues onto the following image. If the segmentation is unsatisfactory, the user can enter a 'modify' state, where any of the regions can be segmented again by modification of the parameters controlling the segmentation of the region. The system maintains a list of regions, with associated lists of points making up the region; these are

efficiently manipulated to reflect the changes in segmentation of the region. Since the segmented regions typically contain holes due to variations in illumination, we have also implemented morphological operators (erosion and dilation) as a means to filling them. In particular, dilation has been found to be useful as a postprocess to fill holes left by region growing and split/merge. At any point, a segmented image may be written out to secondary storage for later review and playback. Once the segmentation of a image sequence terminates, the system outputs statistics that reflect the characteristics of the region. In the current implementation, once the processing is completed for each image, the total number of pixels making up each region, the mean intensity and standard deviation is output.

2.2 Segmentation Algorithms

In the region growing algorithm, the user interactively selects a seed pixel from any point within the region of interest and its intensity is determined. A tolerance value (in intensity units) is specified through the interface (threshold parameter in Figure 1). The four orthogonal neighbors of the seed pixel are next evaluated for membership; a pixel is added to the region if its intensity is within the specified intensity tolerance, i.e.

$$I_{seed} - I_{tol} \leq I(x,y) \leq I_{seed} + I_{tol}$$

where I_{seed}, I_{tol} and $I(x,y)$ are the intensities of the seed pixel, specified tolerance and the intensity of the pixel being tested for region membership. A pixel accepted as part of the region will recursively have its neighbors tested. For efficiency, our implementation uses a queue to determine the accepted pixels.

The split/merge algorithm used in our implementation is a modified version of the standard algorithm. The modification makes the algorithm sensitive to the properties of the region of interest. The algorithm begins by determining the mean intensity, μ_r, and standard deviation, σ_r of a small rectangular neighborhood (user defined, 11×11 in our implementation) region r surrounding a chosen point within this region. Next the algorithm performs a quadtree partitioning of the region, creating 4 equal sized sub-regions. Each of these sub-regions, s_i is evaluated for homogeneity. Homogeneity is determined by first determining the mean and standard deviation, μ_s and σ_s of the subregion. A subregion is homogeneous if a given percentage p_s of pixels in the subregion have intensity values which fall within n_s number of standard deviations of the subregion's mean. In our implementation, all of these parameters are controlled through the interface (Figure 1). Typically, $p_s = 0.95$ and $n_s = 1$. If a subregion is determined to be homogeneous, then its statistics are compared to the properties of the region of interest, μ_r and σ_r. If μ_r is sufficiently close to μ_s, all pixels of s_i become part of the segmented region. For acceptance, $|\mu_s - \mu_r|$ must fall within n_r standard deviations of the mean of the region of interest. Thus, there are two tests a subregion must pass before acceptance:

1. $|I(x,y) - \mu_s| \leq n_s\sigma_s$, must be satisfied by at least p_s percent of the pixels of sub-region s_i.
2. $|\mu_s - \mu_r| \leq n_r\sigma_r$.

If both of these conditions are satisfied, then the sub-region s_i is accepted and the region is colored. As our implementation currently keeps an explicit representation of the points making up the segmented region, explicit region merging is unnecessary.

Finally, our segmentation system is also capable of applying both algorithms in sequence on a particular region (which we call an overlay), in an attempt to take advantage of their combined strengths. Our experimental results have indicated some success to using both algorithms in sequence, most notably to fill in holes that are generated by the region growing algorithm. A connected component labeling procedure to discard unrelated parts of the region has been found to

Dataset	Image Size	#frames	#Regions	Time (minutes)			
				Smooth.	Segmentation		
					Reg. Grow (#img. modif.)	Spl. Mrg. (#img. modif.)	Overlay (#img. modif.)
1. Face-1	450×590	30	2	5	3(2)	3(3)	5(3)
2. Heart-1	640×400	30	5	5	6(5)	5(8)	7(8)
3. Face-2	230×160	50	2	1	5(6)	6(11)	5(3)
4. Heart-2	260×260	50	2	1	6(9)	8(8)	9(8)
5. Face-3	210×160	49	2	2	5(5)	5(8)	4(2)
6. Heart-2	325×235	50	3	2	8(14)	7(9)	10(9)

Table 1: Test Results/Runtime Statistics

be useful with the overlay algorithm. This procedure performs region growing with the same seed pixel, with the exception that only pixels within the region that are connected to one another (4-connected neighborhood, starting with the seed pixel) are reported as being part of the segmented region. Any remaining holes in the region can be eliminated by applying a dilation operation on the region. Dilation is currently performed using a 3×3 structuring element of unit height.

3 Experimental Results

We have digitized (from video) six sequences of ultrasound images, obtained from three different ultrasound scans. Datasets 1 and 2 are from a scan of an 8 month old fetus. An Accom video digitizer was used to obtain two segments (Face-1 and Heart-1) of the scan corresponding to the baby's face and heart. The second scan was of a baby that was 17 weeks old. An Abekas Diskus digitizing recorder was used to obtain 3 sequences of images (Face-2, Heart-2 and Face-3), corresponding to the baby's face, heart and a second view of the face. Finally, the third scan is of a 5 month old baby, from which 1 segment was digitized, again using the Abekas Diskus and corresponds to the baby's heart.

Table 1 gives details of the test data and processing times for our experiments. All of the processing was performed on a SGI Indigo-2 Impact workstation with an R10K processor. All datasets were smoothed using a 7×7 median filter except for Heart-2 which used a 5×5 sized kernel. Segmentation times are comparable for all three algorithms. In the overlay algorithm, the region growing algorithm is followed by split/merge. The number of images that needed modification (or to be segmented again) is also shown and ranges anywhere from 4 to 18% of the total number of images in the sequence.

It can be seen that frame to frame coherence helps accelerate the segmentation of the image sequences. For longer sequences, the average time to segment each image should further decrease. Note that in the case of Face-2 and Face-3, the overlay algorithm results in modification of fewer images than either of the two algorithms applied individually.

Table 2 reports on the sizes of the segmented regions. A sample image from 3 of the datasets was used to illustrate the behavior of the various algorithms. Columns 3 and 4 show the region sizes due the region grow and split/merge applied individually. In general, the size reported by split/merge is larger, which is mostly due to unrelated regions, although it does fill in some of the holes left by the region growing algorithm. The last two columns show the region sizes resulting from the overlay algorithm followed by the connected component labeling procedure (column 5), which is followed by dilation (column 6). This sequence of segmentation procedures allows us to be conservative in the region growing and split/merge procedures; while this tends to underestimate the feature size, it is necessary to prevent bleeding; dilation helps fill most of the holes that remain in the regions

Dataset	Region No.	Region Size (pixels)			
		Reg.Gr.	Spl.Mrg.	Conn. Comp.	Dil.
1. Face-1	1	3286	3371	3571	4102
	2	2695	2378	2761	3240
2. Heart-1	1	666	764	865	1059
	2	375	411	397	457
	3	18873	19043	19161	20754
	4	1623	1598	1806	1897
3. Face-2	1	334	364	433	633
	2	571	741	748	286

Table 2: Region Sizes

as well as grow the boundary in a controlled fashion. The use of several procedures does increase the user interaction time, but it permits a more accurate segmentation of the region.

In the top row of Figure 2[2] are shown examples from Face-1. The left image is the original (which has been cropped), the middle image shows the eye sockets segmented by region growing, and the right image also includes dilation. The dilation helps close most of the holes in the segmented region. The second and third rows illustrate the effect of the various algorithms on an example image from the sequence of Heart-1. The left image is the original, middle image shows 4 regions segmented using region growing and right image uses the split/merge algorithm. The white regions (3 of them) are part of the wall that separate the upper and lower chambers of the heart. In the third row, the left image shows the segmentation using the overlay algorithm (region growing followed by split/merge), the middle image shows the same with the disconnected regions eliminated by application of the connected component algorithm. The right image is the result of dilating the segmented regions in the middle image using a 3×3 structuring element (box of unit height). This helps fill the holes in the regions and provides a better estimate of the feature size. In our experimentation, we find that the region growing algorithm by itself tends to be conservative in estimating the size of the feature; increasing the tolerance to include more of the region typically results in including points outside the region, causing it to bleed. The split/merge method helps fill in some of the holes (yellow region in the overlay example) and add more pixels to the outside of the region. But it does generate a number of pixel clusters that are unrelated to the feature, due to the global nature of the algorithm. The connected component procedure eliminates all of these and the dilation is very effective in filling the holes that are still left behind (the white region in the middle and right images of row 3 indicate pixels that are common to both region grow and split/merge). Finally, the left image in row 4 illustrates the segmentation (using region growing only) of the heart into its 4 chambers with the walls in white[3]. Since neither the wall nor the valve is well defined in the original (left image, row 2), it is very difficult to get a clean segmentation of all of these features. The top and bottom images of row 4 are examples from Face-2 and Heart-2 respectively. Here again, the two eye sockets are segmented in Face-2 and two regions corresponding to the heart of Heart-2.

4 Concluding Remarks/Future Work

In this article, we have explored the use of two algorithms, region growing and a variation of split/merge for segmentation of fetal ultrasound images. Because of the variety of artifacts present

[2]Original color images may be found at *http://www.cs.uncc.edu/∼krs/publ.html*.

[3]A Quicktime animation of a sequence of the segmented heart may be found in *http://www.cs.uncc.edu/∼krs/publ.html*

in ultrasound images as well as their poor dynamic range, we have found that it is almost impossible to apply automatic segmentation methods with a high degree of confidence. As a result, we have embarked on an interactive approach, with mechanisms to speed up the segmentation via frame to frame coherence. We are able to segment sequences of 30-50 images within five to six minutes, depending on the degree of coherence between consecutive images. Considering the human interaction involved, the ability to segment each sequence in such a short time is quite promising. In the cases where the boundaries of the regions are fairly well defined, we have been able to use the region growing or split/merge algorithm to successfully segment the images. Our experimentation with both region growing and the split/merge algorithms leads us to conclude that their combined application to ultrasound image sequences, followed by some post-processing to overcome the weaknesses of these algorithms results in a more accurate and robust detection of features. The biggest weakness of the system in its current form is the lack of effective measures that can evaluate the accuracy of the segmentation, and hence, the results tend to be somewhat subjective. One idea to address this is to use ultrasound technicians to interactively mark the features of interest and evaluate the results.

This work highlights the difficulties involved in reliable segmentation of ultrasound images. It also calls for techniques that are more tolerant of the noise and artifacts present in ultrasound images. Algorithms such as region growing are highly sensitive to the local neighborhood, requiring sharp discontinuities (boundary) for region isolation. A better characterization or quantification of the boundary would help in improving the noise tolerance of these algorithms.

A representation that provides a better characterization of the boundary is the *Binary Space Partitioning (BSP) tree* [7]. BSP trees represent functions by encoding all of the discontinuities within a function via partitioning hyperplanes. The representation is simply a binary tree, with internal nodes containing hyperplanes (representing discontinuities) and the leaf cells representing regions that are relatively homogeneous. These regions can be approximated by a continuous function, such as a constant (say, the mean of the pixel intensities) or a linear function.

BSP trees have been used to represent CT and MRI images [13]. They have been shown to be highly tolerant to noise and missing boundary points [14], and hence could find application to ultrasound images. The method used to convert the discrete image into a partitioning tree (refer to [13] or [14] for details) makes it possible to reliably identify the image discontinuities (or boundary), even in the presence of considerable noise or missing edge points. Thus, performing the representation conversion to a BSP tree and then applying algorithms such as region growing on the continuous representation should lead to a more robust segmentation algorithm.

5 Acknowledgements

Our thanks to Carolinas Medical Center, Charlotte, for providing us with clinical data that was used in the experiments. This research was supported, in part, by the National Science Foundation, and by a faculty research grant from UNC Charlotte.

References

[1] R.J. Bartrum and H.C. Crow. *Gray-Scale Ultrasound: A manual for Physicians and Technical Personnel*. W.B. Saunders Company, Harcourt Brace Jovanovich, Inc., 1977.

[2] J.C. Bezdek, L.O. Hall, and L.P. Clark. Review of mr segmentation images using pattern recognition. *Medical Physics*, 20(4):1033–1048, 1993.

[3] C. Busch and M. Eberle. Morphological operations for color-coded images. *Computer Graphics Forum*, 14(3), 1995.

[4] R.C. Gonzalez and R.E. Woods. *Digital Image Processing*. Addison Wesley, 1992.

[5] F.W. Kremkau. *Diagnostic Ultrasound*. W.B. Saunders Company, Harcourt Brace Jovanovich, Inc., 1989.

[6] J. Lengyel, D.P. Greenberg, and R. Popp. Time-dependent three-dimensional intravascular ultrasound. *ACM Computer Graphics Proceedings*, August 1995.

[7] B.F. Naylor. Interactive solid modeling using partitioning trees. In *Proceedings of Graphics Interface '92*, Vancouver, CA, May, 1992.

[8] T.R. Nelson and T. Elvins. Visualization of 3d ultrasound data. *IEEE Computer Graphics and Applications*, 13(6), November 1993.

[9] J. K. Ousterhout. *Tcl and the Tk Toolkit*. Addison Wesley, 1994.

[10] J.R. Roelandt, C. di Mario, N.G. Pandian, L. Wenguang, D. Keane, C.J. Slager, P.J. de Feyter, and P.W. Serruys. Three-dimensional reconstruction of intracoronary ultrasound images. rationale, approaches, problems, and directions. *Circulation*, 90(2), August 1994.

[11] G. Sakas, L. Schreyer, and M. Grimm. Preprocessing, segmenting and volume rendering 3d ultrasonic data. *IEEE Computer Graphics and Applications*, 15(4), July 1995.

[12] G. Sakas and S. Walter. Extracting surfaces from fuzzy 3d-ultrasound data. *ACM Computer Graphics Proceedings*, pages 465–474, August 1995.

[13] K.R. Subramanian and B.F. Naylor. Representing medical images with partitioning trees. In *Proceedings of Visualization '92*, Boston, MA, Oct. 19-23, 1992.

[14] K.R. Subramanian and B.F. Naylor. Converting discrete images to partitioning trees. *IEEE Transactions on Visualization and Computer Graphics*, 1996. Submitted.

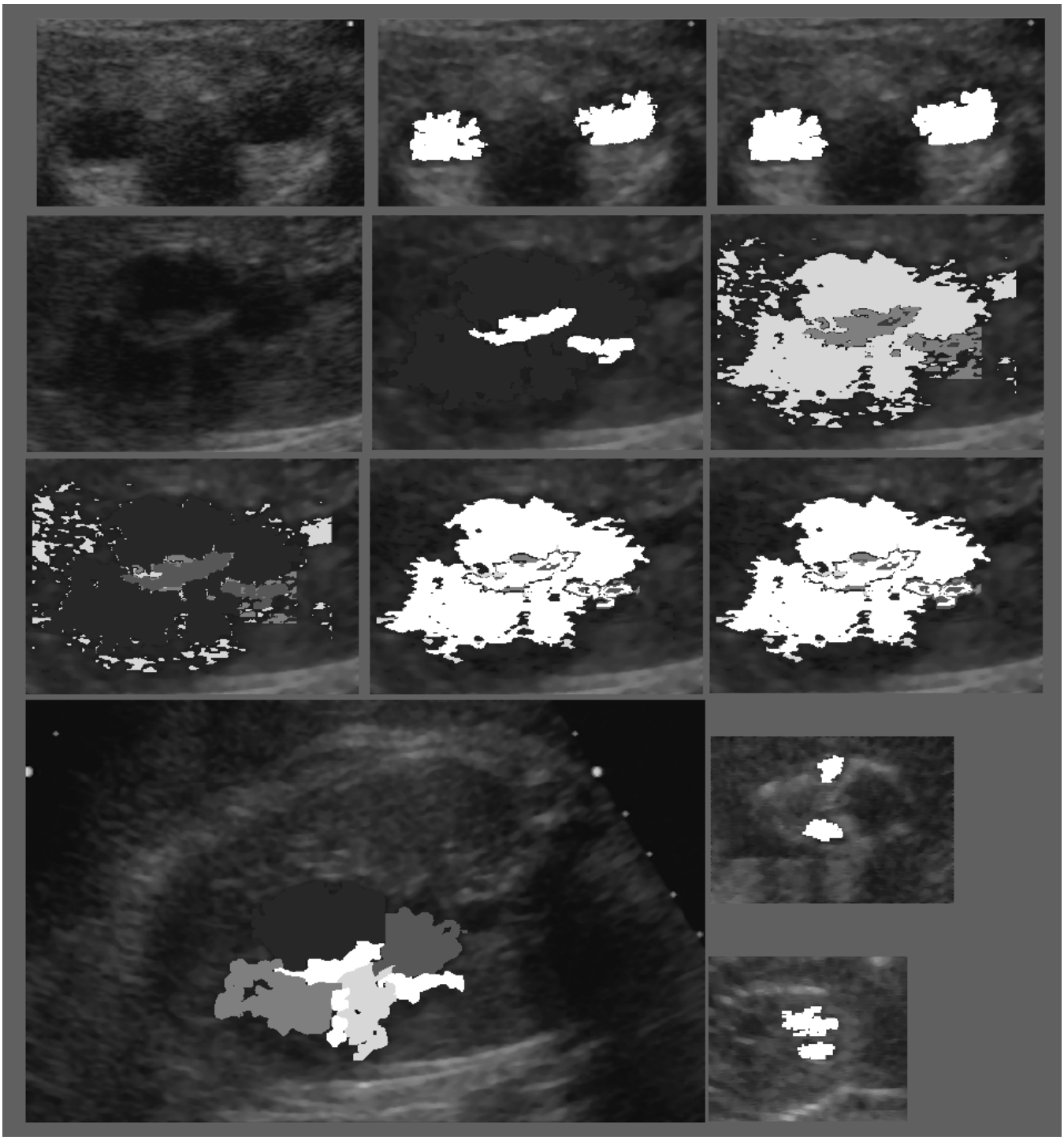

Figure 2: Experimental Results