Fall 2012

Automated and interactive approaches for optimal surface finding based segmentation of medical image data

Shanhui Sun
University of Iowa

AUTOMATED AND INTERACTIVE APPROACHES FOR OPTIMAL SURFACE FINDING BASED SEGMENTATION OF MEDICAL IMAGE DATA

by

Shanhui Sun

An Abstract

Of a thesis submitted in partial fulfillment of the
requirements for the Doctor of Philosophy
degree in Electrical and Computer Engineering
in the Graduate College of
The University of Iowa

December 2012

Thesis Supervisor: Assistant Professor Reinhard R. Beichel

ABSTRACT

Optimal surface finding (OSF), a graph-based optimization approach to image segmentation, represents a powerful framework for medical image segmentation and analysis. In many applications, a pre-segmentation is required to enable OSF graph construction. Also, the cost function design is critical for the success of OSF. In this thesis, two issues in the context of OSF segmentation are addressed. First, a robust model-based segmentation method suitable for OSF initialization is introduced. Second, an OSF-based segmentation refinement approach is presented.

For segmenting complex anatomical structures (e.g., lungs), a rough initial segmentation is required to apply an OSF-based approach. For this purpose, a novel robust active shape model (RASM) is presented. The RASM matching in combination with OSF is investigated in the context of segmenting lungs with large lung cancer masses in 3D CT scans. The robustness and effectiveness of this approach is demonstrated on 30 lung scans containing 20 normal lungs and 40 diseased lungs where conventional segmentation methods frequently fail to deliver usable results. The developed RASM approach is generally applicable and suitable for large organs/structures.

While providing high levels of performance in most cases, OSF-based approaches may fail in a local region in the presence of pathology or other local challenges. A new (generic) interactive refinement approach for correcting local segmentation errors based on the OSF segmentation framework is proposed. Following the

automated segmentation, the user can inspect the result and correct local or regional segmentation inaccuracies by (iteratively) providing clues regarding the location of the correct surface. This expert information is utilized to modify the previously calculated cost function, locally re-optimizing the underlying modified graph without a need to start the new optimization from scratch. For refinement, a hybrid desktop/virtual reality user interface based on stereoscopic visualization technology and advanced interaction techniques is utilized for efficient interaction with the segmentations (surfaces). The proposed generic interactive refinement method is adapted to three applications. First, two refinement tools for 3D lung segmentation are proposed, and the performance is assessed on 30 test cases from 18 CT lung scans. Second, in a feasibility study, the approach is expanded to 4D OSF-based lung segmentation refinement and an assessment of performance is provided. Finally, a dual-surface OSF-based intravascular ultrasound (IVUS) image segmentation framework is introduced, application specific segmentation refinement methods are developed, and an evaluation on 41 test cases is presented. As demonstrated by experiments, OSF-based segmentation refinement is a promising approach to address challenges in medical image segmentation.

Abstract Approved: _____

Thesis Supervisor

_____

Title and Department

_____

Date

AUTOMATED AND INTERACTIVE APPROACHES FOR OPTIMAL SURFACE FINDING BASED SEGMENTATION OF MEDICAL IMAGE DATA

by

Shanhui Sun

A thesis submitted in partial fulfillment of the
requirements for the Doctor of Philosophy
degree in Electrical and Computer Engineering
in the Graduate College of
The University of Iowa

December 2012

Thesis Supervisor: Assistant Professor Reinhard R. Beichel

Graduate College
The University of Iowa
Iowa City, Iowa

CERTIFICATE OF APPROVAL

PH.D. THESIS

This is to certify that the Ph.D. thesis of

Shanhui Sun

has been approved by the Examining Committee for the thesis requirement for the Doctor of Philosophy degree in Electrical and Computer Engineering at the December 2012 graduation.

Thesis Committee: _____
Reinhard R. Beichel, Thesis Supervisor

_____
Milan Sonka

_____
Joseph M. Reinhardt

_____
Xiaodong Wu

_____
John E. Bayouth

ACKNOWLEDGEMENTS

I would like to thank my advisor Dr. Reinhard R. Beichel for all the patient guidance and support. Throughout my Ph.D study, I learnt a great deal from him including critical thinking and research attitude.

I would like to thank Dr. Milan Sonka for providing IVUS data sets and valuable suggestions for my research. Also, I would like to thank Dr. John E. Bayouth for providing access the commercial radiation treatment planning system Pinnacle. Many thanks are due to Dr. Junyi Xia who taught me to use the Pinnacle system. I would like to thank Dr. Eric A. Hoffman, Dr. Joseph M. Reinhardt and Dr. John E. Bayouth for providing lung data sets. The contribution of Tomas Kovarnik, M.D. whose manual tracing of IVUS image data formed the independent standard is gratefully acknowledged.

Many thanks are due to my colleagues Dr. Christian Bauer, Dr. Honghai Zhang, Dr. Qi Song, Dr. Xin Dou, Dr. Mingqing Chen, Richard Downe, and Yao Wang. I fully appreciate the great discussion with them.

In addition, I would like to thank Dr. Milan Sonka, Dr. Joseph Reinhardt, Dr. Xiaodong Wu, and Dr. John E. Bayouth for serving on my committee.

This work was supported in part by the Biological Sciences Funding Program of The University of Iowa, and by NIH grants R01–EB004640 and R01–HL063373.

Finally, last but not least, I would like to especially thank my wife, Zhiyun Gao, for her love and support throughout this process.

ABSTRACT

Optimal surface finding (OSF), a graph-based optimization approach to image segmentation, represents a powerful framework for medical image segmentation and analysis. In many applications, a pre-segmentation is required to enable OSF graph construction. Also, the cost function design is critical for the success of OSF. In this thesis, two issues in the context of OSF segmentation are addressed. First, a robust model-based segmentation method suitable for OSF initialization is introduced. Second, an OSF-based segmentation refinement approach is presented.

For segmenting complex anatomical structures (e.g., lungs), a rough initial segmentation is required to apply an OSF-based approach. For this purpose, a novel robust active shape model (RASM) is presented. The RASM matching in combination with OSF is investigated in the context of segmenting lungs with large lung cancer masses in 3D CT scans. The robustness and effectiveness of this approach is demonstrated on 30 lung scans containing 20 normal lungs and 40 diseased lungs where conventional segmentation methods frequently fail to deliver usable results. The developed RASM approach is generally applicable and suitable for large organs/structures.

While providing high levels of performance in most cases, OSF-based approaches may fail in a local region in the presence of pathology or other local challenges. A new (generic) interactive refinement approach for correcting local segmentation errors based on the OSF segmentation framework is proposed. Following the

automated segmentation, the user can inspect the result and correct local or regional segmentation inaccuracies by (iteratively) providing clues regarding the location of the correct surface. This expert information is utilized to modify the previously calculated cost function, locally re-optimizing the underlying modified graph without a need to start the new optimization from scratch. For refinement, a hybrid desktop/virtual reality user interface based on stereoscopic visualization technology and advanced interaction techniques is utilized for efficient interaction with the segmentations (surfaces). The proposed generic interactive refinement method is adapted to three applications. First, two refinement tools for 3D lung segmentation are proposed, and the performance is assessed on 30 test cases from 18 CT lung scans. Second, in a feasibility study, the approach is expanded to 4D OSF-based lung segmentation refinement and an assessment of performance is provided. Finally, a dual-surface OSF-based intravascular ultrasound (IVUS) image segmentation framework is introduced, application specific segmentation refinement methods are developed, and an evaluation on 41 test cases is presented. As demonstrated by experiments, OSF-based segmentation refinement is a promising approach to address challenges in medical image segmentation.

TABLE OF CONTENTS

LIST OF TABLES

Table

LIST OF FIGURES

Figure

LIST OF ALGORITHMS

Algorithm

CHAPTER 1
INTRODUCTION

1.1 Background and Motivation

Medical imaging modalities, such as magnetic resonance imaging (MRI), computed tomography (CT) and ultrasound (US), are used to noninvasively create images of organs inside the human body for the purpose of diagnosis, screening, treatment planning or study of anatomy and pathology. Since its introduction, medical imaging technology has become increasingly important in medicine. Quantitative and highly automated analysis of medical images is essential for efficient utilization of medical image data. In a routine clinical task, physicians look at images to screen for diseases like cancers. However looking at images is often not sufficient for diagnosis and treatment, especially for tasks like planning of surgical resection or radiation treatment planning. Thus, segmentation and subsequent quantitative analysis of anatomical and pathological structures is needed. In addition, state-of-the-art imaging technology enables physicians to create high-resolution volumetric images showing anatomy and pathology in great detail, but the increased amount of image data to be (quantitatively) analyzed represents a burden for physicians. Manual analysis of such image data is impracticable. To address this problem, highly automated quantitative image analysis tools are required.

Medical image segmentation is often the first step in quantitative image analysis. The aim of segmentation is to identify the target anatomy or pathology and

delineate the boundary of these structures of interest. Image segmentation plays a vital role in numerous medical applications [87, 106].

As the development of medical imaging technology is progressing, increased amount of image data need to be processed. For example, the segmentation of 3D and 4D (3D + time) image data is quite challenging and an active research topic. Recently, an optimal surface finding (OSF) [124, 75] and layered optimal graph image segmentation of multiple objects and surfaces (LOGISMOS) [128, 75] approaches have been reported with applications to medical image segmentation tasks. The approach guarantees global optimality of the resulting segmentation according to a given cost function. It is based on a node-weighted graph in d-D space ($d \geq 2$) and enables the detection of a surface under geometric constraints in a low-order polynomial time [124, 75]. In the OSF-based segmentation, the surface (or multiple surfaces) segmentation task is transformed into finding a minimum-cost closed-set which is solved by means of a maximum-flow algorithm [17].

Many OSF-based approaches have been published and demonstrate the performance of globally optimal segmentation. For example, Zhao *et al.* [134] proposed an automated segmentation method for the aorta in 4D cardiovascular magnetic resonance (MR) image data based on an OSF-based approach [75]. The pre-segmentation was generated from 4D level-sets and the tubular structure was transformed into terrain-like structure on which a multisurface graph was constructed. This segmentation method utilized the edge information of the transformed image [134]. Haeker *et al.* [48] combined region information with boundary information to define

a cost function for OSF-based intraretinal layer segmentation of optical coherence tomography (OCT) images. Wang and Beichel [122] presented a lymph node segmentation framework using a mesh-based OSF approach. The graph was built based on a sphere around a lymph node, and graph column profiles were constructed along the direction of surface points to the sphere center. They took gray value homogeneity and image edge information into account in their cost function design. Song *et al.* [105] developed a framework based on the OSF approach for simultaneous bladder and prostate segmentation. Their work incorporated a smoothness penalty function using weighted arcs between neighboring columns. In this way, a soft smoothness shape compliance is introduced to incorporate prior shape information. As mentioned in [127], the mesh surface based OSF approach may suffer from the problem of mesh folding when normal directions are employed in the column profile construction especially in the case of sharp shape changes and concave regions. To address this issue, Yin *et al.* proposed an electric force based method to construct the column profiles. However the algorithm [127] has $O(n^2)$ computation complexity, because the computation of the force vector at a certain point in space requires considering all vertex points of the initial surface. To avoid mesh folding, Bauer *et al.* [7] proposed a gradient vector flow (GVF) [125] based approach to build column profiles. First, gradient vectors are calculated with the method presented in [125] from the initial surface. Second, starting from a surface point of the initial surface, vectors are followed in either direction to construct non-overlapped column profiles [7].

Graph-cuts [16, 17, 18] and a more recent strategy of detecting a maximum

weight region decomposable into two star-shaped regions [47] represent alternative options for generating globally optimal segmentation. Although these optimization problems are ultimately transformed to solving maximum-flow problem, OSF-based methods [124, 75] are fundamentally different from graph-cuts [16, 17, 18] and the "star-shape" region approach presented in [47]. One advantage of OSF-based approach is that it can be easily extended to a simultaneous multi-surface detection incorporating smoothness constraint.

In contrast to OSF, graph-cuts [16, 17, 18] and the "star-shape" region approach [47] are grid graph based and do not require a pre-segmentation. However, seeds may be required for graph-cuts, and star shape region "center" is needed in case of the work presented in [47]. For OSF-based segmentation, if the target surfaces have relatively "simple geometry" such as terrain-like, spherical, cylindrical and tubular structure, a pre-segmentation may not be needed for constructing the graph. For example, the intraretinal layer segmentation of OCT images [48, 44] does not require a pre-segmentation since the intraretinal layers are intrinsically a terrain-like structure and the graph is constructed as a grid weighted graph where nodes on the graph columns correspond to the voxels in the 3D OCT image data. The lymph node segmentation presented in [122] directly utilized a sphere structure to build the graph. Although the aorta segmentation in [134] utilized a level-set based pre-segmentation, the graph was not directly built from the pre-segmentation. Instead a tubular structure constructed from the centerline of the pre-segmentation was utilized. However, medical objects are not always "simple geometric" structures, and

often an approximate pre-segmentation is required to be able to utilize OSF-based segmentation approaches. For example, in the case of bladder and prostate segmentation in [105], cartilage segmentation of the knee joint in [127] and liver segmentation in [52, 133], initial segmentations were first performed to generate a shape prior close to the target object. The more accurate the initialization is, the larger the likelihood that the subsequent OSF-based segmentation succeeds. Hence, developing a robust and automated initial segmentation approach is essential for many OSF-based segmentation applications.

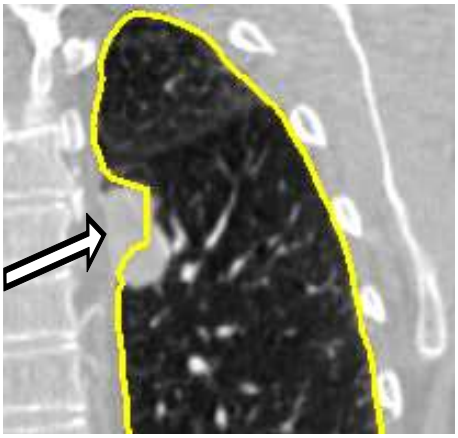


Figure 1.1: Example of a local segmentation error (arrow) due to lung pathology (cancer). The defined cost function is not well suited for this specific location of the lung boundary.

It is a nontrivial problem to find a suitable cost for a specific segmentation problem. For example, a cost function may work for the majority of cases, but some

anatomical and pathological variation might cause the automated OSF segmentation to fail to accurately delineate the correct boundary in some local region. For example, in the case of 3D lung segmentation in CT image data, the high density pathology (cancer masses) can occur near to the boundary. Under such circumstances, edge and region based cost may not be sufficient to produce a correct segmentation in this region (Fig. 1.1). To correct the local failure of the automated OSF-based segmentation, effective and efficient interactive refinement methods are required to produce a segmentation suitable for quantitative analysis.

1.2 Specific Aims

This work deals with developing a novel robust active shape model (ASM) for initialization of OSF-based segmentation approaches and the interactive refinement of OSF based segmentation results. Specifically, the aims of this work are as follows:

Aim 1: Develop an automated robust ASM based segmentation method for the initialization of OSF segmentation approaches.

Aim 2: Apply and validate the robust ASM segmentation in the context of OSF to automatically segment lungs with large masses in 3D CT images.

Aim 3: Develop a generic interactive OSF-based refinement approach utilizing a virtual reality environment for display and user interaction.

Aim 4: Adapt the generic segmentation refinement approach to several medical image segmentation problems. Specifically, the following applications are investigated:

(a) lung segmentation in 3D CT images (3D single surface graph-based segmentation

approach),

(b) lung segmentation in 4D CT images (4D single surface graph-based segmentation approach), and

(c) segmentation of 3D intravascular ultrasound (IVUS) images (3D multiple surface graph-based segmentation approach).

1.3 Thesis Overview

The thesis is organized in eight chapters. In Chapter 2, a robust ASM algorithm is presented that is suitable to serve as initial segmentation required for OSF-based segmentation. In Chapter 3, the robust ASM is utilized and validated in combination with OSF segmentation to automatically segment lungs with large cancer masses in 3D CT images. In Chapter 4, a generic interactive OSF-based segmentation refinement framework using a hybrid VR/desktop user interface is introduced. The generic refinement approach is adapted to 3D OSF-based single surface lung segmentation refinement and validated in Chapter 5. In a feasibility study presented in Chapter 6, this approach is extended to 4D lung segmentation and refinement. In addition, a performance assessment is presented. Segmentation refinement in the context of 3D IVUS segmentation is described and validated in Chapter 7. Finally, conclusions of this work are discussed in Chapter 8.

CHAPTER 2
ROBUST ACTIVE SHAPE MODEL BASED SEGMENTATION FOR INITIALIZATION OF OPTIMAL SURFACE FINDING

2.1 Introduction

In many medical image segmentation problems, it is advantageous to utilize a priori shape information, which is likely to increase the robustness of the segmentation approach. OSF-based segmentation intrinsically incorporates shape information via the graph construction process and allows to restrict the solution by means of smoothness constraints [124, 75]. Thus, for OSF segmentation applications that require a pre-segmentation, the quality of this initial segmentation is critical to fully utilize the potential of OSF. However, generating a good pre-segmentation is often a nontrivial problem. There is no dictated universal applicable algorithm to get the pre-segmentation. The requirement for the pre-segmentation is to produce an approximate solution close to the target boundary. In addition, in order to be applicable in routine quantitative analysis, the algorithm should be reasonably fast. Even though only a coarse result is required, it is challenging to balance robustness and computing time for 3D object segmentation because of the potentially large size of the target structure (e.g., lungs) and the amount of image data to be processed. The developed approach addresses this issue. The robust ASM matching algorithm is specifically designed to take advantage of general-purpose computation on graphics processing units (GPGPU), which reduces the execution time considerably. In this chapter we described our robust ASM segmentation approach in details.

2.2 Related Work

Parametric (e.g., snake models [62]) and geometric (e.g., level-set [22]) deformable models are widely used in medical image segmentation. However, it is challenging for these algorithms that do not use a prior (shape) information to handle cases with weak edge information, similar shapes next to each other (e.g., prostate and bladder), or pathology (e.g., cancers adjacent to the object boarder). Incorporating object shape or appearance information can help to improve the robustness of the segmentation.

A parametric deformable template model presented in [61] is one example to overcome these limitations by starting the segmentation from a shape template learned from a set of training data. The deformable model minimizes the objective function by iteratively update the transformation parameters in order to match the shape of the template to the target. A segmentation application based on a Bézier deformable template model for coarse lung segmentation can be found in [63]. The construction of the template is a nontrivial procedure especially for large objects in volumetric medical image data.

Atlas or registration-based segmentation method share a similar idea with a deformable template model. The atlas is constructed from the learning data. During the segmentation, the atlas is deformed to match the target object by means of registration, utilizing image features to change the transformation parameters. Zhang *et al.* [131] utilized an atlas registration framework to initialize a pulmonary fissure segmentation approach, which is followed by a two-step graph search procedure to refine the

fissure segmentation. However, atlas-based segmentation is typically computationally expensive.

The active shape model (ASM) is a model based segmentation method which was introduced by Cootes *et al.* in [30, 29, 26, 32]. ASMs allow to model the mean shape as well as shape variations. The shape model is learned from a set of shapes, which are aligned in a common coordinate system. Information about the learned shape variability is utilized for segmentation by iteratively fitting the model to the target object.

ASMs have been used for several medical image segmentation tasks. In [26], an ASM has been utilized to detect the heart chamber in 2D echocardiogram images. Duta and Sonka [85] performed neuroanatomic structure segmentation in 2D MR brain images by an improved ASM algorithm. Ginneken *et al.* [115] proposed an extended ASM, by utilizing "optimal features" and used a kNN classifier to locate update points instead of the frequently utilized Mahalanobis distance. The authors utilized their algorithm for lung segmentation in 2D chest radiographs as well as cerebellum and corpus callosum segmentation in 2D slices of MRI brain image data [115]. 3D ASM model were also developed by the medical image analysis community. Heimann *et al.* [54, 53] described a ASM based liver segmentation in 3D CT images. Heimann *et al.* [52] utilized a ASM in combined with OSF for the 3D segmentation of the liver in CT data. Zhang *et al.* [129] proposed a hybrid segmentation framework of ASM and OSF for 3D echocardiography images of the left ventricle. The authors replaced the standard ASM landmark position searching method by a OSF method.

Thus, a globally optimal shape points are used to update the ASM.

Active appearance models (AAMs) utilize shape and appearance information learned from samples for segmentation [27, 41, 32, 28]. For the matching, AAMs try to minimize the intensity difference between the texture patch warped from the sampled original image and the appearance model patch reconstructed by the model. To match an AAM, shape, intensity and model parameters need to be optimized. Note that AAMs represent appearance and shape for the whole target object while ASMs model shape and appearance along profiles utilized to update the ASM.

Medical image segmentation applications based on AAMs can be found in several publications. Cootes *et al.* [25] reported an approach to segment ventricles, the caudate nucleus and the lentiform nucleus in 2D MR brain cross-sections utilizing the AAM framework. Mitchell *et al.* [83] proposed a left and right ventricle segmentation method based on AAMs for 2D MR cardiac images. Mitchell *et al.* [82] improved the left ventricle segmentation by extending the conventional AAM framework to an Active Appearance Motion Model, for 2D + time cardiac MR images. A left and right ventricle segmentation method using 3D AAMs was proposed by the same group in [80]. Beichel *et al.* [12] developed an automated AAM method to segment diaphragm surface in 3D CT images. Zhang *et al.* [130] extended a hybrid ASM/AAM algorithm [81] to allow for left and right ventricle segmentation in 4D MR cardiac images. A comprehensive review about ASMs and AAMs can be found in [51].

Statistics shape models (SSMs) like ASMs and AAMs provide a generic segmentation framework for volumetric medical image data. For OSF-based segmenta-

tion targeting a complex surface, these two SSMs are potential candidates for generating an initial segmentation. We have chosen the ASM for the following reasons.

(1) In volumetric medical segmentation, the potentially large volume of organs such as lungs leads to long computation time for AAM based approaches because the iterative matching process needs to deal with image volumes. In addition, the steadily increased resolution of medical images amplifies this issue.

(2) The appearance of target organs can change significantly due to artifacts or pathology e.g., tumors. Such disturbances are frequent. A standard AAMs will fail to overcome this issue, because the learned model may not cover all variability in appearance. The robust extensions of AAMs like the algorithm proposed by Beichel *et al.* [10] provide options to solve this issue, but will require significantly more computation time.

Because of the disadvantages of AAMs when utilized to segment potentially large structure, this work will use a ASM-based initial segmentation for OSF.

2.3 Methods

In the following sections, a robust ASM (RASM) segmentation approach will be presented. It is utilized to produce a pre-segmentation of (large) structures as required for subsequent OSF-based segmentation. In Section 2.3.1, an approach for generating a shape model is described. Section 2.3.2 outlines the standard ASM matching approach. In Section 2.3.3, a novel RASM is introduced, and in Section 2.3.4, we outline how this algorithm can be implemented in parallel on graphics processing units

to speedup computing time.

2.3.1 Model Generation

The ASM-based segmentation approach requires learning shapes to build the shape model. In this context it is desirable to have a learning set (n shapes) which is representative for the targeted population. To build the shape model, a set of corresponding points (landmarks) $\{\mathbf{s}_1, \mathbf{s}_2, \ldots, \mathbf{s}_m\}$, where m is the number of the landmarks, need be identified on all the learning set. The identification can be done manually for a 2D data set, but it becomes impossible task for 3D. Therefore, an automated identification method is required to identify corresponding landmarks. There are several methods available to generate landmarks automatically. For example, Frangi *et al.* [42] presented an atlas-based registration framework. First, the learning shapes are aligned and registrated to an atlas template. Second, the landmarks are identified on the template, and landmarks in template space are then propagated to each of the original learning shapes. For genus-0 closed-surface shapes (e.g., in medical context, lungs, liver, kidneys, spleen, prostate etc.), minimum description length (MDL) [54] and spherical harmonics descriptors method (SPHARM) [46] can be utilized to find point correspondence in 3D. To find landmark correspondence for open-surface shapes, Dalal *et al.* [35] proposed an iterative landmark-sliding method.

Once all m landmarks are defined for all n shapes, n landmark sets are aligned in a common coordinate frame by using Procrustes analysis [40], resulting in a mean shape vector $\bar{\mathbf{x}}$. For each learning shape, a shape vector

$\mathbf{x}_i$ with $i = 1, 2, \ldots, n$ is generated by concatenating the coordinates: $\mathbf{x}_i = [x_{i,1}, y_{i,1}, z_{i,1}, x_{i,2}, y_{i,2}, z_{i,2}, \ldots, x_{i,m}, y_{i,m}, z_{i,m}]^T$. A principal component analysis (PCA) was applied to the covariance matrix $S = \frac{1}{n-1}\sum_{i=1}^{n}(\mathbf{x}_i - \bar{\mathbf{x}})(\mathbf{x}_i - \bar{\mathbf{x}})^T$ to generate a point distribution model (PDM) [26]. An instance of a object shape can be generated from the corresponding PDM by the linear model

$$\mathbf{x} = \bar{\mathbf{x}} + \mathbf{P}\mathbf{b}\,, \tag{2.1}$$

where $\mathbf{P}$ denotes the shape eigenvector matrix derived from S, and $\mathbf{b}$ represents the shape coefficients. The statistic model (Eq. (2.1)) describes object shapes in terms of a mean shape and variation about the mean.

2.3.2 Standard ASM Matching

The PDM (Section 2.3.1) can be used for image segmentation by matching the model to the target structure. This can be achieved by utilizing a standard ASM matching framework [26]. This matching procedure consists of four steps:

i) An instance of the shape model (e.g., mean shape) is generated and placed in proximity to the target structure.

ii) To match the model to the target, shape points are updated by searching from the current landmark location along a profile normal to the model surface with length l_{ASM}. To find update points $\mathbf{y}$, several approaches can be used. For

example, one could search for the strongest edge. Alternatively, a local appearance profile can be modeled for each landmark point from learning images [31]. Update points are then detected based on the similarity to the learned appearance profile.

iii) Once all shape points are updated, pose parameters are adjusted to map the updated shape points in the target image coordinate frame to the mean shape in the model coordinate frame. For this purpose, a Procrustes alignment step is used to estimate transformation matrix $\mathbf{T}$, which consists of scaling, rotation, and translation parameters, by minimizing $(\mathbf{T}[\mathbf{y}] - \bar{\mathbf{x}})^T(\mathbf{T}[\mathbf{y}] - \bar{\mathbf{x}})$. Model parameters $\mathbf{b}$ are updated using

$$\mathbf{b} = \mathbf{P}^T(\mathbf{T}[\mathbf{y}] - \bar{\mathbf{x}}) \tag{2.2}$$

and a new instance of the model is calculated utilizing Eq. (2.1) and transformed to the image space by $\mathbf{T}^{-1}$.

iv) Steps i) to iii) are repeated until the model converges.

2.3.3 Robust ASM Matching

We are interested in the segmentation of medical image data. The tissue surrounding the target structure may show the similar appearance pattern. Also, the appearance of the object to be segmented can be altered by disease e.g., tumors. Thus, it is very likely that some update points are found during the model matching

procedure that do not belong to the target surface (outliers). Consequently, the standard matching approach will fail, because it is a least squares optimization procedure that is not suitable to handle outliers. Therefore, a robust shape model matching approach is required.

The basic idea behind robust ASM matching is to only use inlier components of $\mathbf{y}$ to update model parameters. In this context, Rogers *et al.* investigated M-estimators and random sampling-based robust parameter estimation techniques for 2D ASM matching [94]. It is well known that the effectiveness of M-estimators strongly depends on the selection of the weighting function and its parameters. Usually, this selection is not trivial, and the optimal selection might change from case to case. Random sampling techniques try to find a subset of inliers by evaluating a number of randomly sampled subsets of update points. Such approaches work well, if the required subset of inliers is quite small. In the case of large ASM models, this strategy can lead to suboptimal results, because a small set of inliers might not be representative enough to describe a complex shape (many landmark points), and thus can negatively impact the matching result. For our application, it is desirable to use as many inliers as possible for the model update. Lekadir *et al.* proposed a robust 3D ASM matching method based on local shape dissimilarity defined by point triplet ratios [73]. In the case of large ASMs, a large number of possible triplets exist and selecting an optimal subset is not trivial.

Our approach utilizes a robust PCA coefficient estimation scheme that builds on the work of Storer et al. [110]. Storer's method was designed for robust image

reconstruction and targets a pre-defined number of inliers. Here, we propose a novel voting scheme that does not require to specify a targeted number of inliers. Our method consists of two processing steps. First, normal shape patterns of landmark subsets are learned. Second, these patterns are then utilized during ASM matching to identify and reject outliers.

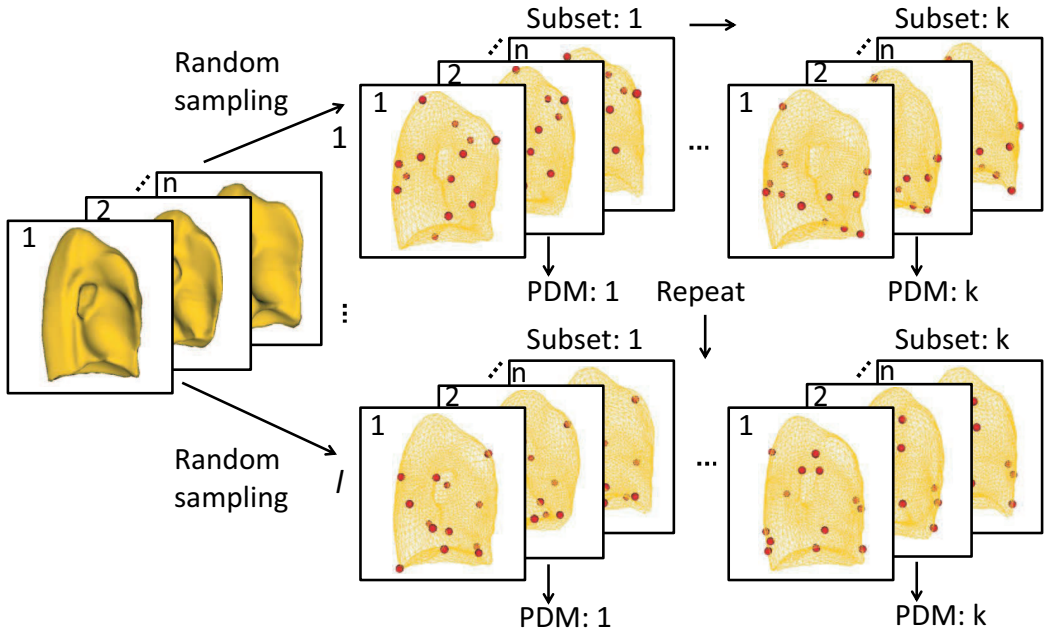


Figure 2.1: Robust shape pattern learning. A random sampling process is repeated l-times (rows). In each random sampling process, k shape subsets are derived from all n training shapes and utilized to generated point subset distribution models.

The overview of the robust shape pattern learning process is shown in Fig. 2.1 using left lung shape model as an example. Corresponding landmark points of

all learning shapes are partitioned randomly into k shape subsets of approximately equal size. This process is repeated l-times, resulting in a set of subsets: $\Omega = \{\omega_{i,j} | i \in 1, 2, \ldots, l;\ j \in 1, 2, \ldots, k\}$. Note that corresponding landmark points of all n learning data sets are always assigned to the same subset. Consequently, each subset $\omega_{i,j}$ consists of n subset samples. For each subset $\omega_{i,j}$, a mean shape $\bar{\mathbf{x}}_{\omega_{i,j}}$ is calculated by using Procrustes analysis, and all shapes of the subset are aligned. The subset shapes are then converted to shape vectors by concatenating their x-, y-, and z-components. By means of PCA, the corresponding eigenvector matrices $\mathbf{P}_{\omega_{i,j}}$ and eigenvalue vectors $\lambda_{\omega_{i,j}}$ are calculated. $\bar{\mathbf{x}}_{\omega_{i,j}}$, $\mathbf{P}_{\omega_{i,j}}$ and $\lambda_{\omega_{i,j}}$ are stored and utilized for robust ASM matching.

In order to match the model to the target image, outlier components of the update position $\mathbf{y}$ in each iteration need to be identified. This is accomplished by analyzing the subset combinations of $\mathbf{y}$ and utilizing a voting scheme. Let $\mathbf{y}_{\omega_{i,j}}$ represent the components of $\mathbf{y}$ that are corresponding to the landmark points that constitute subset $\omega_{i,j}$. Let $m_{\omega_{i,j}}$ represent the number of update points in each subset. A subset reconstruction error $e_{\omega_{i,j}}$ is defined:

$$e_{\omega_{i,j}} = \left\| \mathbf{T}_{\omega_{i,j}}[\mathbf{y}_{\omega_{i,j}}] - \left[\bar{\mathbf{x}}_{\omega_{i,j}} + \mathbf{P}_{\omega_{i,j}} \tilde{\mathbf{b}}_{\omega_{i,j}} \right] \right\| \tag{2.3}$$

where $\mathbf{T}_{\omega_{i,j}}$ is a transformation matrix that aligns $\mathbf{y}_{\omega_{i,j}}$ to the corresponding mean $\bar{\mathbf{x}}_{\omega_{i,j}}$. The vector $\tilde{\mathbf{b}}_{\omega_{i,j}}$ is derived from:

$$\mathbf{b}_{\omega_{i,j}} = \mathbf{P}^T_{\omega_{i,j}} \left[\mathbf{T}_{\omega_{i,j}}[\mathbf{y}_{\omega_{i,j}}] - \bar{\mathbf{x}}_{\omega_{i,j}} \right] \tag{2.4}$$

by constraining $\mathbf{b}_{\omega_{i,j}}(v)$ in $[-\xi\sqrt{\lambda_{\omega_{i,j}}(v)}, \xi\sqrt{\lambda_{\omega_{i,j}}(v)}]$, $v \in \{1, 2, \ldots, m_{\omega_{i,j}}\}$. A large

reconstruction error $e_{\omega_{i,j}}$ is an indication that subset $\omega_{i,j}$ is very likely contaminated by one or more outliers. The l-times repeated subdivision increases the possibility of outlier free point combinations. To identify the outliers, the reconstruction error $e_{\omega_{i,j}}$ is interpreted as a vote, which is casted for all update points that are included in the subset $\mathbf{y}_{\omega_{i,j}}$. This voting process is carried out for all subsets $\omega_{i,j} \in \Omega$. Outliers frequently get large vote values. The casted votes are collected in a matrix $\mathbf{V}_{err}$ of size $m \times l$, in which rows correspond to shape points in $\mathbf{y}$. After all votes are casted, $\mathbf{V}_{err}$ is analyzed to detect outliers. First, to increase robustness, a rank order statistics filter is applied to each row; the values are sorted, and the g-lowest value is selected to represent the filter result. This filtering step reduces $\mathbf{V}_{err}$ to a vector $\mathbf{v}_{err} = [v_1, v_2, \ldots, v_m]^T$ and helps to reject accidentally occurring point constellations that contain outliers, which are similar to constellations of inlier points. A typical histogram of vector component values is shown in Fig. 2.2. Second, a threshold δ is derived from $\mathbf{v}_{err}$ by analyzing the distribution of vector components v_i: $\delta = \mu + \beta\sigma$ with $\mu = median_{i \in 1,2,\ldots,m}\{v_i\}$ and $\sigma = \sqrt{1/m \sum_{i=1}^{m} (v_i - \mu)^2}$, where β represents a constant. Third, the threshold is applied to $\mathbf{v}_{err}$ to yield a selection vector: $\mathbf{p}_{sel} = [p_1, p_2, \ldots, p_m]^T$ with

$$p_i = \begin{cases} 1 & : \quad v_i < \delta \\ 0 & : \quad v_i \geq \delta \end{cases} \tag{2.5}$$

to discriminate between inliers ($p_i = 1$) and outliers ($p_i = 0$). Once the inliers are identified, the transformation matrix $\mathbf{T}$ is obtained by aligning selected update points $\mathbf{y}_{\{p_i=1\}}$ to the selected mean shape points $\bar{\mathbf{x}}_{\{p_i=1\}}$ and then shape parameters $\mathbf{b}$ are

calculated using

$$\mathbf{b} = \mathbf{P}_{\{p_i=1\}}^{T}(\mathbf{T}[\mathbf{y}_{\{p_i=1\}}] - \bar{\mathbf{x}}_{\{p_i=1\}}) \ . \tag{2.6}$$

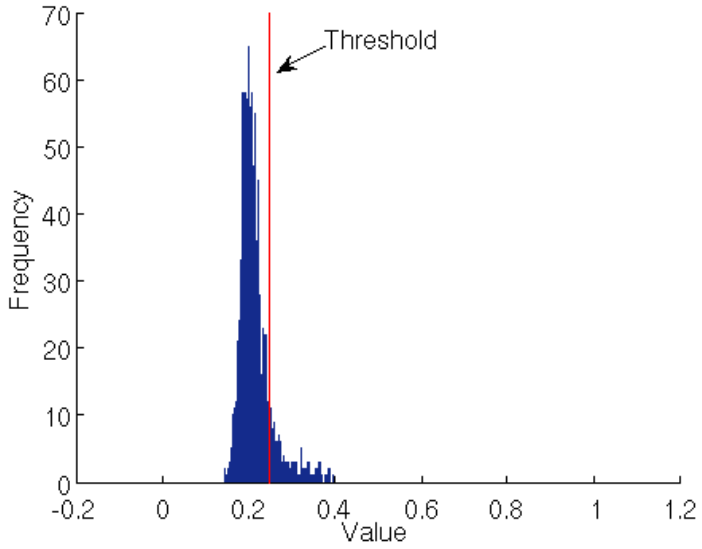


Figure 2.2: Histogram of component values of $\mathbf{v}_{err}$ and automatically derived threshold utilized to detect outlier components.

The outlier rejection scheme is illustrated in Fig. 2.3.

To achieve a robust behavior of the ASM during matching, all outliers present must be rejected. In addition, we would like to utilize as many inliers as possible to achieve a good match between image and model. This has several implications for the selection of parameters for the RASM matching algorithm. For example, when selecting the number of shape subsets k, a trade-off must be made. On the one hand, the value for k must be small enough such that the shape points within each shape set can form distinctive point patterns. On the other hand, a larger number of shape

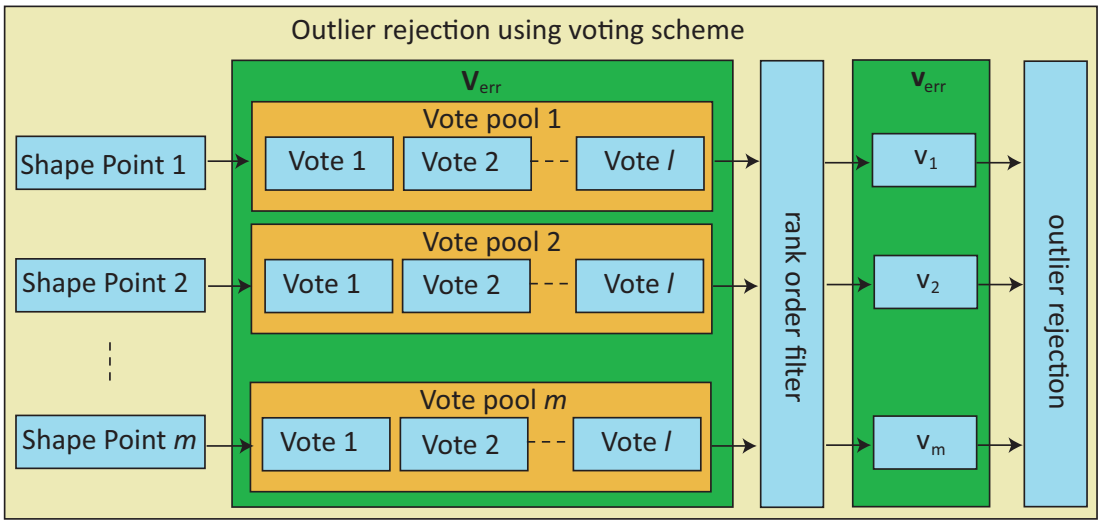


Figure 2.3: Outlier rejection scheme.

subsets is preferable, because it becomes more likely that outlier-free shape subsets can be found, which reduces the number of required subset evaluation iterations (parameter l). The parameter β should be selected conservatively, to make sure that all outliers are rejected, even if this implies that a small number of inliers is not utilized for matching the ASM.

2.3.4 Algorithm Speed-up

An advantage of our outlier detection algorithm presented in the Section 2.3.3 is that it is well suited for parallel processing, because the analysis of the k shape subsets which is repeated l-times is independent from each other, and thus, can be done in parallel. We utilize a GPGPU-based implementation to speed up the outlier detection algorithm. $\bar{\mathbf{x}}_{\omega_{i,j}}$, $\mathbf{P}_{\omega_{i,j}}$ and $\lambda_{\omega_{i,j}}$ were stored in the GPU memory. After point

positions $\mathbf{y}$ updated, they are subdivided into $\mathbf{y}_{\omega_{i,j}}$ by a $k \times l$ allocated CUDA[1] threads and aligned to corresponding subset means shape $\bar{\mathbf{x}}_{\omega_{i,j}}$ based on $\mathbf{T}_{\omega_{i,j}}[\mathbf{y}_{\omega_{i,j}}]$. Each thread then calculates corresponding reconstruction error $e_{\omega_{i,j}}$ in parallel according to the Formula 2.3. The subset reconstruction errors are finally synchronized to the CPU side for subsequent outlier rejection.

[1]http://www.nvidia.com/object/cuda_home_new.html

CHAPTER 3
VALIDATION OF ROBUST ACTIVE SHAPE MODEL BASED SEGMENTATION: AUTOMATED 3D SEGMENTATION OF LUNGS WITH LUNG CANCER IN CT DATA

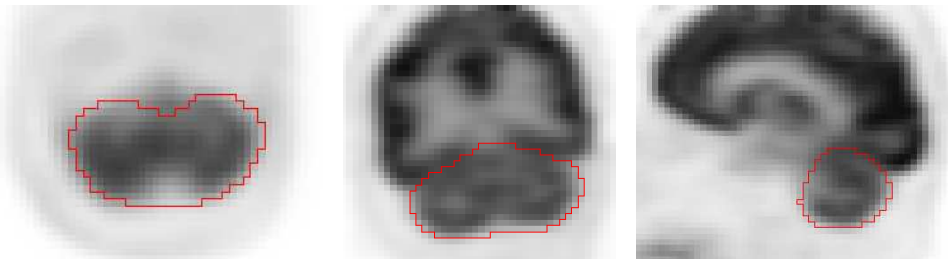
3.1 Introduction

Application and validation of the RASM described in the Chapter 2 in the context of fully automated 3D segmentation of lungs with large lung cancer is described in this chapter. Note that after the publication of RASM in [112], Bauer *et al.* [8] has utilized proposed novel RASM method for segmentation of cerebella in volumetric PET images (Fig. 3.1) to deal with partially imaged cerebella. For a detailed validation of the RASM in this context, the reader is referred to [8].

3.2 Background

Lung cancer represents a major health problem. Worldwide, lung cancer is responsible for 1.3 million deaths annually, according to the WHO[1]. Tomographic imaging modalities like multidetector X-ray computed tomography (CT) play an important role in diagnosis, treatment, and research of lung cancer. State-of-the-art CT imaging technology enables physicians to create high-resolution volumetric scans describing lung anatomy and pathology, but manual or semiautomated lung segmentation is becoming impossible for the increased amount of image data to be processed. To address this problem, automated lung image analysis methods are required.

[1]http://www.who.int/mediacentre/factsheets/fs297/en/; accessed January 2011



(a) (b) (c)

Figure 3.1: An example of cerebellum segmentation in a Fluorodeoxyglucose PET scan utilizing RASM approach. (a) Axial image slice. (b) Coronal image slice. (c) Sagittal image slice. All PET images are shown with inverted gray-scales.

3.3 Related Work

Many approaches to automated quantification of lung disease require the segmentation of lung parenchyma in an initial processing step. In the case of normal lungs imaged with CT, a large density difference between air-filled lung parenchyma and surrounding tissues can be observed. A number of algorithms can be found in the literature (e.g., [56, 103, 57, 4, 72]) that rely on this observation for the segmentation of lungs. We will denote such methods as conventional lung segmentation approaches.

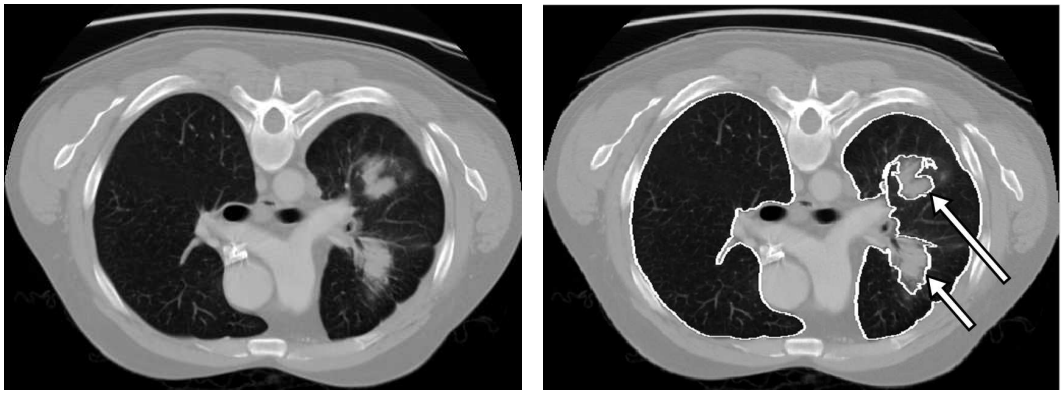
Many conventional lung segmentation approaches are based on some form of gray-level thresholding. Typically, such approaches start with thresholding and followed by a sequence of morphological operations, airway exclusion, and left and right lung separation. For example, Hu *et al.* in [57] utilized an optimal threshold method to separate the body from background followed by 3D connected component labeling to identify lung voxels. Interior cavities were eliminated by morphological

operations. The trachea and main bronchi were detected slice by slice. The left and right lung regions were separated by identifying the anterior and posterior junctions. Methods with similar major processing steps can be found in [103], [72], [4] and Lo's PhD thesis [76]. Shojaii *et al.* [102] presented a lung segmentation method based on a form of watershed transform avoiding the processes of finding optimal threshold and separating left/right lung. Ali *et al.* in [2, 3] proposed a lung segmentation framework combining Markov Gibbs random field and a graph-cuts method. The initial labeling was generated by thresholding.

The above mentioned conventional lung segmentation methods rely gray-value thresholding. However, in the case of lungs with lung cancer (Fig. 3.2(a)) or other high density pathologies (e.g., pneumonia), lung segmentation becomes a non-trivial task, and frequently, conventional algorithms fail to deliver suitable segmentation results (Fig. 3.2(b)). Thus, to enable computer-aided cancer treatment planning (e.g., surgery or radiation treatment) and to facilitate the quantitative assessment of lung cancer masses (e.g., treatment response assessment), robust lung segmentation methods are needed.

Apart from thresholding based methods, Li *et al.* in [74] presented a statistical approach. First, a coarse lung segmentation was performed using a standard ASM based approach. The appearance along profiles learned from training samples was utilized to update the ASM. Second, an Active Contour Model based approach was used to refine the pre-segmentation. The prior shape information was incorporated in this approach, which can be more robust against pathology than thresholding based

methods, but it was not tested on pathological cases. Potentially, outliers introduced by large lung cancer masses can reduce the robustness of the standard ASM based segmentation.



(a) (b)

Figure 3.2: Segmentation of a lung with cancer using a conventional approach. (a) Axial CT image showing normal right and cancerous left lung tissue. (b) Corresponding segmentation result generated with a conventional lung segmentation method. Segmentation errors are indicated by arrows.

Only a few papers have been published that deal with segmentation of diseased lungs. None of the existing methods directly targets the segmentation of lungs with large cancer regions at arbitrary locations. For example, a Bézier surface-based method was proposed in [63] to deal with lesions adjacent to the chest wall and mediastinum. First a shape model of lung side walls is fitted to the target image by

using an affine transformation. Second, an active contour model was utilized to refine the initial segmentation. Since the lung apex and base part are not included in the model, the resulting segmentation was combined with a conventional segmentation. Thus, lesions adjacent to the lung apex or diaphragm can result in segmentation errors. Pu *et al.* [91] proposed an automated lung segmentation approach based on a 2D adaptive border marching algorithm to deal with juxtapleural lung nodules. Larger areas of under-segmentation were reported in hilar and pulmonary consolidation regions. In recent work, Pu *et al.* describe a Shape "Break-and-Repair" strategy which was utilized to segment lungs with juxtapleural lung nodules [90]. A method for the robust segmentation of lung parenchyma based on the curvature of ribs was presented in [89]. The method is based on an adaptive thresholding scheme and utilizes a comparison of the curvature of the lung boundary to the curvature of the ribs to select thresholds. Because lung pathologies like cancer can have density values similar to other tissues surrounding the lung, the method will likely show errors in such cases. Recently Wang *et al.* proposed a method for the segmentation of lungs with interstitial disease [121]. First, an initial segmentation was generated by utilizing a threshold-based conventional lung segmentation method. Second, interstitial lung tissue regions were identified based on texture features. The resulting segmentations were then combined to form the final segmentation result. Sluimer *et al.* proposed a segmentation by registration approach for the segmentation of pathological lungs [104]. While delivering promising results, not all pathological cases could be handled successfully [104]. In addition, the authors also identified the need to reduce pro-

cessing time from three hours to a clinically more acceptable processing time [104]. To solve this problem, a hybrid lung segmentation method was presented in a recent publication [116] of the same group. The basic idea is to first use a conventional lung segmentation method, assess the correctness of the segmentation based on volume and shape features, and utilize the segmentation by registration approach similar to [104] only if the conventional method failed. Korfiatis *et al.* proposed a segmentation by classification approach based on texture analysis for the segmentation of interstitial pneumonia in high-resolution CT [69]. The approach utilized k-means cluster method following morphological closing operation to obtain an initial segmentation. The lung region segmentation was refined by an iterative support vector machine based neighborhood labeling of border pixels utilizing gray-level and wavelet coefficient features. In addition, the authors also investigated gray level averaging and gradient features as alternatives. Hua proposed two methods to segment pathological lungs [58]. The first is a Geodesic Active Contour approach and the second is based on a constraint graph search method. The constraint graph search based lung segmentation method was also reported in [59]. Both methods from a presegmentation generated by Hu's lung segmentation method described in [57]. In order to create an initial shape, morphological dilation was applied to this presegmentation. For the case of healthy lungs or lungs with minor lung disease (e.g., small lung nodule or small areas of interstitial disease), the initial shape was in proximity to the target. However, in the case of severe lung disease (e.g., large lung cancer mass or large areas of interstitial disease) the initial shape may be far away from the lung boundary. In

addition, the presegmentation can leak to air-filled region in the CT image and both subsequently applied methods may fail to deliver expected result.

In this chapter, we present a novel approach for the fully automated segmentation of lungs with lung cancer regions which addresses many of the limitations of existing methods like robustness or processing speed. Our approach is based on a RASM method presented in Chapter 2. To automatically initialize the RASM, we propose a model initialization method which is based on a novel rib detection approach that is suitable for normal and contrast enhanced CT scans. The performance of our fully automated lung segmentation system is assessed on 30 lung CT scans with 40 abnormal (lung cancer) and 20 normal (no signs of lung disease) left/right lungs. In addition, we provide a performance comparison with two commercially available methods on the same image data. Both methods are utilized in the context of lung radiation treatment planning. The first method is based on a region growing algorithm and the second method utilizes a deformable template-based segmentation approach. In terms of computing time, the model-based 3D segmentation of lungs is particularly challenging, because of the size of lungs and the amount of image data to be processed. As outlined in Chapter 2, parallel GPGPU computing will be used to address this issue which reduces the execution time considerably.

3.4 Methods

An overview of our segmentation approach is shown in Fig. 3.3. First, ribs are detected and utilized to initialize (place) the ASM [26] in the lung CT scan. Second,

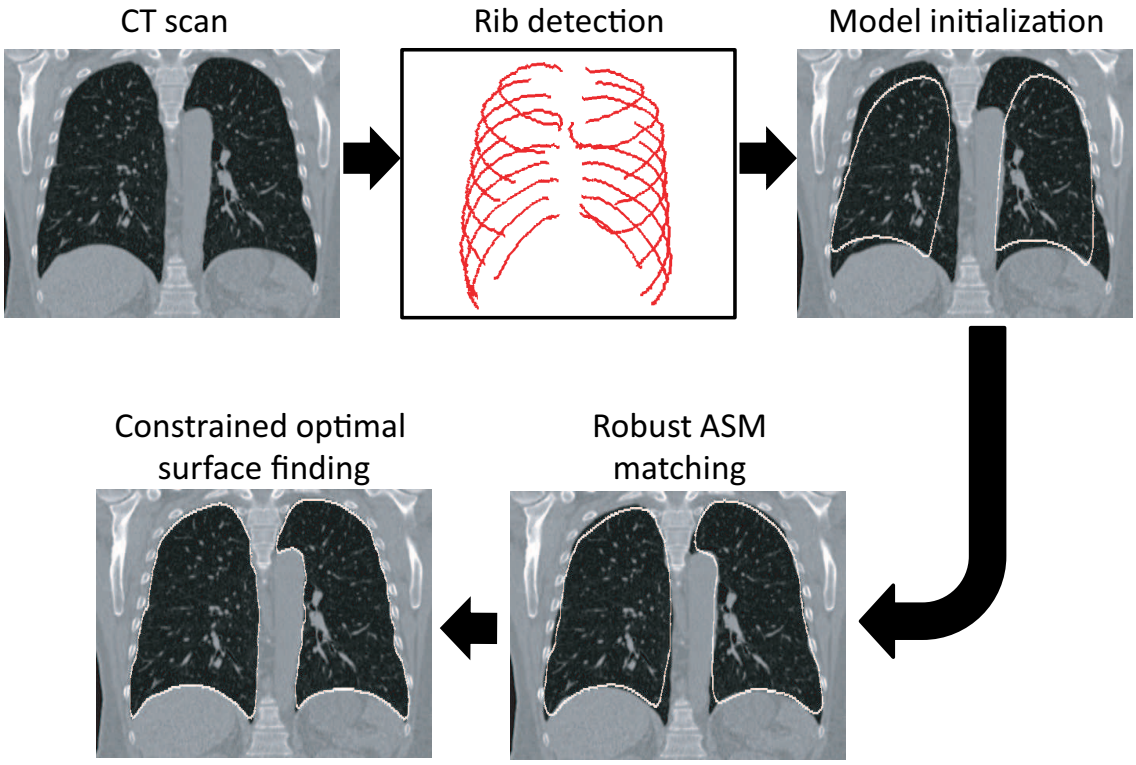


Figure 3.3: Overview of our model-based segmentation approach.

the robust ASM (RASM) matching algorithm, which was introduced in Chapter 2, is applied to generate a coarse segmentation of the (diseased) lungs. Third, the segmentation result is adapted by means of a constraint optimal surface finding approach. In the following sections, we describe our approach in detail.

3.4.1 Lung Model Generation

Our lung segmentation approach requires learning shapes to build a lung model. In this context it is desirable to have a learning set which is representative for the targeted population. For our experiments, a set of $n = 41$ different total lung capacity (TLC) lung CT scans without contrast enhancement, which showed no signs of lung disease or other pathology, were available. Clearly, the utilized learning set size is limited. However, additional learning samples can be easily added, if needed. The details of the model generation process are described in the following paragraph.

Learning data sets were segmented using the commercial lung image analysis software Pulmonary Workstation 2.0 (PW2) from VIDA Diagnostics, Inc., Coralville IA. The selected segmentation algorithm generates smooth surfaces which cut across main bronchi and pulmonary arteries/veins in the area near the mediastinum. In addition, all segmentation results were manually inspected and corrected, if needed. To produce left and right lung models, the below outlined process was applied to segmented left and right lungs, respectively. From the segmentations, triangle meshes were generated by utilizing a marching cube algorithm [78]. A set of corresponding

points (landmarks) $\{\mathbf{s}_1, \mathbf{s}_2, \ldots, \mathbf{s}_m\}$ with $m = 2562$ were automatically identified on all meshes by means of a minimum description length (MDL) approach [54] based on shape index and curvedness [67]. The result of this processing step is a set of n meshes with m corresponding vertices. In this context, the selected number of landmarks represents a good trade-off between computing time and surface point density.

An example of the shape variation of the left lung associated with two largest PCA modes is shown in Fig. 3.4.

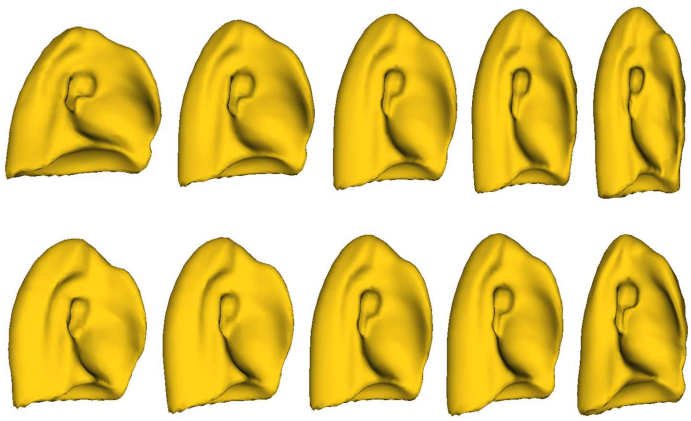


Figure 3.4: Shape variations associated with the two largest PCA modes: The first row shows $\bar{\mathbf{x}} - 3\sqrt{\lambda_1}$, $\bar{\mathbf{x}} - 1.5\sqrt{\lambda_1}$, $\bar{\mathbf{x}}$, $\bar{\mathbf{x}} + 1.5\sqrt{\lambda_1}$ and $\bar{\mathbf{x}} + 3\sqrt{\lambda_1}$. The second row shows $\bar{\mathbf{x}} - 3\sqrt{\lambda_2}$, $\bar{\mathbf{x}} - 1.5\sqrt{\lambda_2}$, $\bar{\mathbf{x}}$, $\bar{\mathbf{x}} + 1.5\sqrt{\lambda_2}$ and $\bar{\mathbf{x}} + 3\sqrt{\lambda_2}$. $\lambda_i \in \{1, 2\}$ are two largest eigenvalues of covariance matrix S.

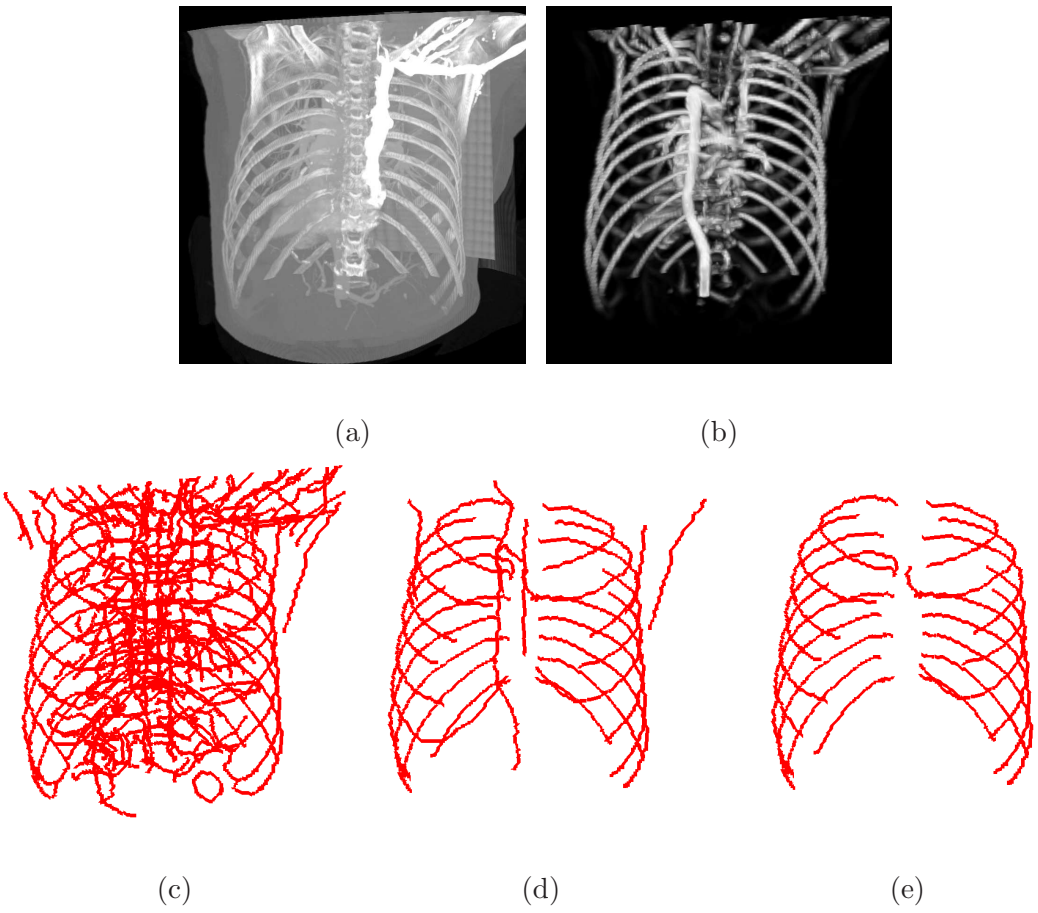


Figure 3.5: Outline of main rib detection processing steps. (a) Volume rendering of the input thorax CT data truncated to a gray-value range between 0 and 500 HU. (b) Volume rendering of Frangi's "vesselness measure" [43] computed at a scale of $\sigma = 5$ mm and (c) corresponding centerlines of rib candidates. Note that many responses from vessels (e.g., aorta) can be found in (b) and (c), because the CT image is contrast enhanced. Centerlines from vessels and other non-rib structures are removed in subsequent rib detection steps. Output of first (d) and second (e) rib clustering/detection stage.

3.4.2 Automated Model Initialization

It is well-known that active shape models (ASMs) have typically a limited capture range. Consequently, they need to be initialized in proximity to the target structure. Therefore, initial shape ($\mathbf{b}$) and pose (size, rotation, and location) parameters of the ASM need to be determined. For initialization, we set the lung model to its mean shape ($\mathbf{b} = \mathbf{0}$) and utilize an automated method which detects rib centerlines in the CT volume to determine isotropic scale and location (pose) parameters for a given CT data set (Fig. 3.5). Note that instead of segmenting ribs, we utilize a shape-based rib centerline detection approach, which is more robust against rib density variations due to age, etc. A detailed description is given in the next paragraphs.

In a first step, tubular structures that are comparable in size, density, and scale to rib structures are detected and a centerline-based representation is generated as follows:

i) The density values of the volume data set are truncated to a gray-value range of interest between 0 and 500 HU (Fig. 3.5(a)). Subsequently, Frangi's "vesselness measure" [43] at a scale of $\sigma = 5$ mm is computed to identify tubular structures of appropriate size (Fig. 3.5(b)). Note that at this scale, the cross-section of ribs appears as a bright "blob", thus the darker (less dense) bone marrow is no longer visible. For the other parameters of Frangi's approach, values of $\alpha = 0.5$, $\beta = 0.5$, and $c = 5$ are used. Note that the cross-section of ribs varies significantly, but the choice of parameters allows us to overcome this problem. To speed up the computation of Frangi's vesselness measure, the data set is

first downsampled by a factor of 4 in each dimension. In addition, a volume of interest is generated by means of region growing to exclude the volume (e.g., air) outside of the human body from calculation of the vesselness measure. For this purpose, the border of the CT image is set as seed region and a threshold of -500 HU is utilized.

ii) From the vesselness response image (Fig. 3.5(b)), a centerline description is extracted for each tubular structure by utilizing a height ridge traversal with hysteresis thresholding, similarly as described in [6] (Fig. 3.5(c)). In a post-processing step, centerlines with less than ten voxels are discarded, because they typically are caused by image noise or imaging artifacts. In addition, centerlines are cut at furcations of multiple centerlines to avoid problems in areas where other structures like bones (e.g., shoulder blade) or contrast enhanced vessels are in close proximity to ribs, and thus, appear connected at the scale of $\sigma = 5$ mm. The centerlines found (Fig. 3.5(c)) are then utilized in the subsequent analysis step to detect centerlines representing ribs.

Our approach is based on the observation that ribs show similar centerline patterns among each other, compared to other structures like (contrast enhanced) vessels, for example. Thus, we utilize a clustering-based approach outlined in Fig. 3.5 to find rib centerlines. The algorithm consists of two stages. First, potential candidates are selected (Fig. 3.5(d)). Second, a more fine-grain pattern analysis is performed (Fig. 3.5(e)). This two step approach has been chosen to save computing time, because the second step is computationally more expensive. The two steps are described

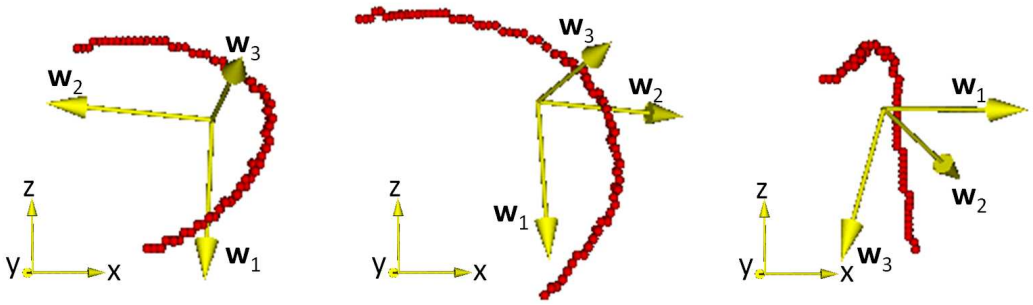
in detail below.

In both stages, a mean shift [24] clustering approach is utilized to detect repeating patterns of ribs. For analysis of feature vectors $\mathbf{q} = \{q_1, q_2, \ldots, q_d\}$, an exponential kernel with profile $k(a) = \exp(-\frac{a}{2})$ and kernel size h_j is used. h_j corresponds to the feature dimension j and is utilized to set the scale of the mode detection. Using the above defined kernel, the mean shift algorithm is utilized to find the mode points for all centerlines. After this analysis, a quantification step is performed to group close-by mode points. If two mode points are closer than the smallest kernel size, they are combined to a new mode point represented by the average of both modes. This process is repeated until convergence.

The features used for clustering are based on geometric properties of the centerlines. For this purpose, a PCA analysis is applied to all points of each centerline, and the resulting eigenvalues $|e_1| \leq |e_2| \leq |e_3|$ and corresponding eigenvectors $\mathbf{w}_1$, $\mathbf{w}_2$ and $\mathbf{w}_3$ are further analyzed.

As outlined above, the actual rib detection is performed in two stages:

i) The goal of this stage is to reject centerline objects that do not show a typical spatial extent of ribs (long curved structures). For all detected centerlines, a feature vector $\mathbf{q} = (e_1, e_2, e_3)^T$ is generated. For analysis, the mean-shift kernel sizes are set as follows: $h_1 = 5.0e\text{-}1$, $h_2 = 5.0e\text{-}4$, and $h_3 = 3.0e\text{-}5$. The above described mean shift clustering typically results in one large and a number of smaller clusters. The feature points of the large cluster represent irregular and small-size centerlines corresponding to structures like vessels, vertebrae,



(a) (b) (c)

Figure 3.6: Visualization of eigenvector patterns utilized for rib detection. Eigenvectors are shown for ribs (a, b) and the aortic arch (c).

etc. After removing this cluster, the remaining feature points relate to ribs and large-size structures like spine, aorta, or parts of the shoulder blade (Fig. 3.6(b), which are further analyzed in the next stage.

ii) For rib centerlines, the eigenvectors $\mathbf{w}_2$ and $\mathbf{w}_3$ span a slanted plane in which the rib centerline is located (Fig. 3.6(a) and (b)). In addition, the eigenvector $\mathbf{w}_1$ is approximately oriented along the z-axis. Because the eigenvectors of similar rib centerlines can point in opposite directions (Fig. 3.6(a) and (b)), we utilize a tensor-based representation ($\mathbf{T}_1$, $\mathbf{T}_2$, and $\mathbf{T}_3$) of eigenvectors with $\mathbf{T}_i = [t_{i_{k,l}}]_{k=1,2,3;\, l=1,2,3}$, where $\mathbf{T}_i$ corresponds to $\mathbf{w}_i$. Note that each tensor matrix $\mathbf{T}_i$ is symmetric. Consequently, only the upper triangular part of the matrix is utilized for further analysis: $\mathbf{t}_i = \{t_{i_{1,1}}, t_{i_{1,2}}, t_{i_{1,3}}, t_{i_{2,2}}, t_{i_{2,3}}, t_{i_{3,3}}\}$. To describe the orientation of centerlines, the term $\tau = |\mathbf{w}_1 \cdot (0, 0, 1)^T|$ is calculated. A feature vector is generated for each centerline by concatenation: $\mathbf{q} = \{\mathbf{t}_1, \mathbf{t}_2, \mathbf{t}_3, \tau\}$.

For cluster analysis, the kernel sizes are set as follows: $h_{\mathbf{t}_1} = 0.85$, $h_{\mathbf{t}_2} = 0.9$, $h_{\mathbf{t}_3} = 1.0$, and $h_\tau = 0.07$. After applying mean shift clustering, the majority of ribs are located in the largest cluster, because of the similarity of their feature vectors, and all other clusters are discarded.

The kernel sizes for both stages were determined on the learning data set. We observed that occasionally false positives (e.g., included clavicles or costal cartilage) and false negatives cases (missing ribs) occur, but we found that such minor errors still allow deriving model pose parameter which are suitable for a rough initialization of the shape model. As demonstrated in Fig. 3.5, our approach is also suitable to deal with contrast enhanced lung CT scans.

Based on the detected rib centerlines, isotropic scale and center location for the ASM are derived. For this purpose, a bounding box for the ribcage is calculated. First, the smallest and largest x- and y-coordinate as well as the median z-coordinate for each centerline in the detection result are calculated. Second, the bounding box $\mathbf{B} = (x_{min}, x_{max}, y_{min}, y_{max}, z_{min}, z_{max})$ is determined. To robustly estimate x_{min}, x_{max}, y_{min}, and y_{max}, the third largest (max) or third smallest (min) value of all centerline extremes in x- and y-direction is selected, respectively. This allows us to deal with occasionally occurring false positive centerline parts (see discussion in previous paragraph). z_{min} and z_{max} are directly calculated from the median z-coordinates of all centerlines. Finally, the scale and position are estimated for the left and right lung separately. For this purpose the rib bounding box is split in the middle, perpendicular to the x-axis. The initial position for the left and right lung model is

found by calculating the center of the left and right bounding box, respectively. An isotropic scale factor is calculated for each lung by averaging the two x- and y-size ratios between left/right mean shape and corresponding bounding box. If needed, the algorithm can be extended to estimate rotation parameters, but based on the utilized CT scan protocols this was not necessary.

3.4.3 RASM Matching

For ASM-based segmentation, we utilize the novel RASM matching approach described in Section 2.3.3.

To find update points $\mathbf{y}$, we use the following cost function:

$$c_i = \begin{cases} \text{ignore point} & if\ \mathbf{n}_i \cdot \mathbf{g}_{dir_i} < 0 \\ g_{mag_i} & otherwise \end{cases} . \tag{3.1}$$

c_i represents the cost of the i-th column element, and the associated sampled gradient magnitude, gradient direction, and surface normal vector are denoted as g_{mag_i}, $\mathbf{g}_{dir_i}$, and $\mathbf{n}_i$, respectively. The gradient calculation is based on Gaussian derivatives with a standard deviation of σ_{ASM}, and the calculation of $\mathbf{g}_{dir}$ and g_{mag} is done for each voxel of the volume before the model matching is started. These pre-calculated gradient values are then used to interpolate gradient vectors during model matching. If a gradient value outside the volume is required during model matching, the position closest to the boundary is utilized. In the case that no new update point can be found, the old position is used.

For our application, we used the following parameters: $l_{ASM} = \pm 40$ mm, $k = 200$, $l = 60$, $\xi = 2$, $g = 10$, and $\beta = 1.3$. To update the robust ASM, a gradient

image was calculated based on Gaussian derivatives with a standard deviation of $\sigma_{ASM} = 4$. The maximum gradient position along the search profile was used to calculate updates for shape points. The model matching was iterated until the average of the shape point movement was below 0.04 mm or at most 100 iterations.

3.4.4 OSF-based Segmentation

Depending on the training data utilized for model building, the model might not be able do describe smaller local shape variations. To capture this information, we generate the final lung segmentation by applying OSF-based segmentation. The graph construction of a single surface OSF-based segmentation is described as below.

The main idea of the OSF-based segmentation algorithm is to transform the segmentation problem into a graph optimization problem. To utilize OSF-based segmentation framework proposed by Li *et al.* [75], a weighted graph $G(N, A)$ is built from an initial mesh surface $M(V, F)$ (shape and topology prior) close to the target surface, where N represents a graph node set, A a graph arc set, V a triangle vertex set, and F a triangle face set respectively. For each vertex $v \in V$, a graph column with l_p is generated along the search profile. The direction of the search profile goes from inside to the outside of the segmented object. The node density on the profile is d_n, which is adjusted to the given image resolution. Intra-column arcs are built to connect nodes $n(v, k)$ to $n(v, k-1)$ on a column $col(v)$ with infinity weights, where k is the column node index. Column $col(v_i)$ and $col(v_j)$ are adjacent columns, if vertices v_i and v_j are on the same triangle edge. For adjacent columns, inter-column arcs are

built to connect the node $n(v_i, k)$ to the node $n(v_j, k - \Delta)$ with infinity weights. Here Δ is the hard smoothness constraint, which is the largest allowed difference in nodes between two adjacent vertices. An example of such graph representation is shown in Fig. 3.7. The graph node weights C (cost function) are derived from volumetric image data to describe local image characteristics.

The segmentation task is transformed to find a minimum-cost closed set by means of a maximum-flow algorithm [17]. To define a minimum-cost closed set problem, node costs are transformed into s-t arc capacities. A node weighted graph eventually becomes an arc weighted graph. This process is depicted in Fig. 3.8. To achieve arc weighted graph, the weight of each graph node is assigned based on the cost of the corresponding node minus the cost of the node below it (Eq. 3.2).

$$w(p,i) = \begin{cases} c(p,i) & if\ i = 0 \\ c(p,i) - c(p,i-1) & otherwise \end{cases}, \tag{3.2}$$

where $c(p, i)$ represents the cost function for the node i on the column p. In addition, to avoid an empty closed set solution, the cost of one of the bottommost nodes of the graph is modified so that the summation of all the bottommost nodes is negative [75]. Left and middle sub-figures in Fig. 3.8 describes cost transformation. Then, arc weighted graph (right sub-figure in Fig. 3.8) is defined as follows. Two additional terminal nodes are added in $G(N, A)$, source node s and sink node t. Two new arcs (terminal arc) are added for each column node. Source arcs connect from node s to each node i of each column p. Sink arcs connect from each node i of each column p to the node t. A capacity $cap_s(p, i)$ is assigned for the arc connecting node s to node i on the column p and a capacity $cap_t(p, i)$ is assigned for the arc connecting node i on

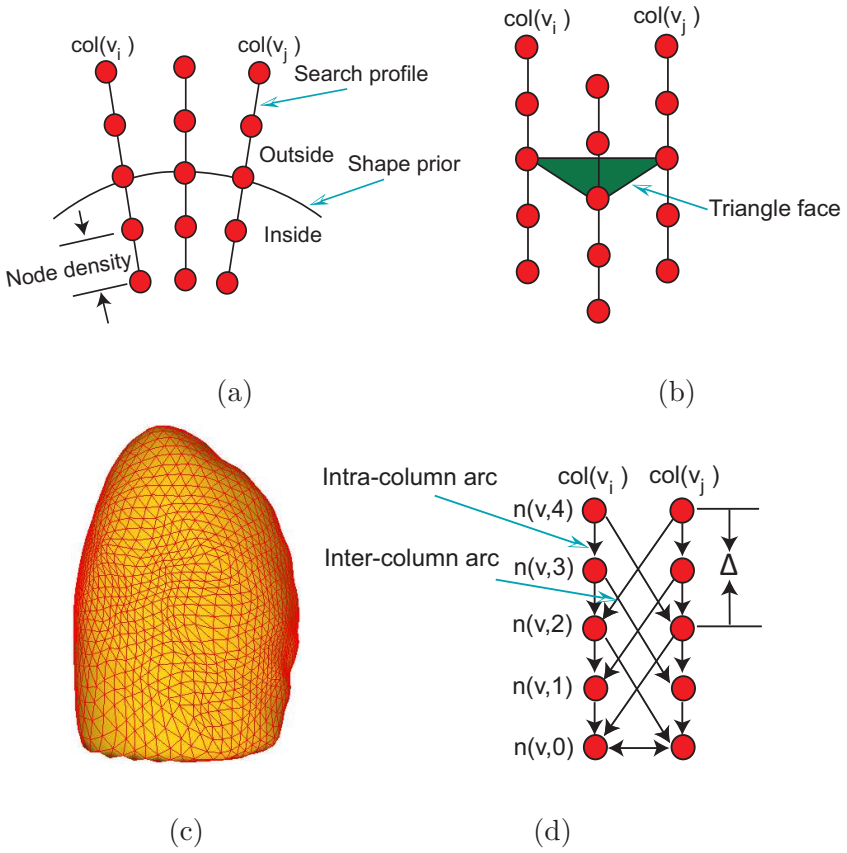


Figure 3.7: Graph construction of a single-surface segmentation problem in the OSF framework. (a) Search profiles are constructed starting from the shape prior. (b) Relation between search profiles and triangle face of the shape prior. (c) Example of the shape prior (pre-segmentation) used for OSF-based lung segmentation. (d) OSF graph structure with arcs enforcing the surface smoothness constraints.

the column p to the node t. If $w(p,i) < 0$, the $cap_s(p,i) = -w(p,i)$ and $cap_t(p,i) = 0$ and if $w(p,i) \geq 0$, the $cap_s(p,i) = 0$ and $cap_t(p,i) = w(p,i)$. Eventually, node weights are not used in the arc weighted graph any more. Note that zero capacity arcs do not appear in the arc weighted graph in original work [75], however we used them in our work to avoid computing maximum-flow from scratch during refinement (Section 4.3.2). After calculating maximum-flow, the feasible surface is found as the envelope of the minimum-cost closed set.

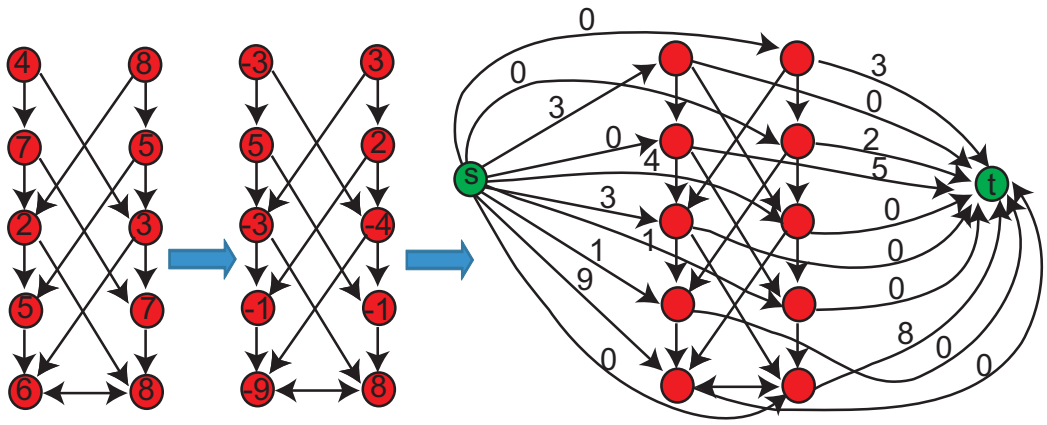


Figure 3.8: Simple example of 2D s-t arc weighted graph (right) transformed from node weighted graph with original costs (left) and intermediate cost-transformed node weighted graph (middle).

3.4.5 Multi-scale OSF-based Lung Segmentation

For lung segmentation we use the following cost function:

$$c_i = \begin{cases} g_{max} & if\ \mathbf{n}_i \cdot \mathbf{g}_{dir_i} < 0 \\ g_{max} - g_{mag_i} & otherwise \end{cases}, \tag{3.3}$$

where c_i represents the cost of the i-th column element and g_{max} the maximum gradient magnitude of the volume, similar to the cost function utilized in Section 3.4.3. The gradient calculation is based on Gaussian derivatives with a standard deviation of σ_g. The OSF is utilized in an iterative coarse to fine fashion using following sequence of values of σ_g and Δ: $\{6.0, 3.0, 1.0, 0.5\}$ and $\{10, 8, 5, 2\}$, respectively. For the search profile $l_p = \pm 10$ voxel is used.

3.5 Validation Methodology

For validation, 30 multidetector computed tomography (MDCT) thorax scans of cancer patients with lung tumors were available. The images where acquired with several different scanners and imaging protocols. In 26 MDCT scans, the vasculature was contrast enhanced. In each data set, either the left and/or right lung contained one or more lung cancer regions with significant higher density, compared to normal lung tissue. To roughly quantify the size of high density lung pathology, the longest diameter in the axial image plane was measured, similar to the Response Evaluation Criteria In Solid Tumors (RECIST) [113]. In this context, note that necrotic lung masses were measured in the same way as solid tumors. The average diameter was 46.28 mm, and the average diameter for right and left lungs was 53.0 mm and 38.8 mm, respectively. The image size varied from $512 \times 512 \times 424$ to $512 \times 512 \times 642$ voxel.

The in-plane resolution of the images ranged from 0.58×0.58 to 0.82×0.82 mm and the slice thickness from 0.6 to 0.7 mm.

With our approach, 60 segmentations of 21 diseased right, 9 normal right, 19 diseased left, and 11 normal left lungs were performed, respectively. All computations were performed on a workstation equipped with a Nvidia Tesla C1060 Computing Processor which offers 240 thread processors and 4 GB of memory.

In addition to the proposed combination of robust ASM and optimal surface finding (RASM+OSF), segmentations were performed with a standard ASM and robust ASM (RASM) without the subsequent surface finding step. The same automatically generated initialization (Section 3.4.2) was utilized for all three segmentation variants. Further, we applied two different methods for lung segmentation provided by a commercial radiation treatment planning system (Pinnacle^3, Philips, The Netherlands). The first approach which will be denoted as "P1" and utilizes region growing and morphological post-processing steps, similar to many commercially available and clinically used lung segmentation methods. The second will be denoted as "P2" and is based on a deformable template approach. In order to utilize method P2, the user is required to manually place a lung shape template in the volume data and to adapt its scale, before iterative matching is performed. All test data sets were processed with both methods. In the case of method P2, the process was repeated three times to allow assessing the impact of model initialization on segmentation performance. Performance measures reported for method P2 represent the average over all three repetitions. For both methods, clinically utilized standard parameter settings were

used. Method P2 allows the user to manually refine a lung segmentation result. For a fair comparison between methods, the operator was not allowed to use this feature.

3.5.1 Independent Reference Standard

For quantitative evaluation of our segmentation method, an independent reference standard was generated by utilizing a commercial lung image analysis software package PW2 from VIDA Diagnostics, Inc., Coralville IA. First, an automated (conventional) lung segmentation was performed. Second, since the software was not designed to deal with lungs containing large lung cancer regions, two experts inspected all the segmentations slice-by-slice and corrected all segmentation errors manually. In the case of diseased lungs, this process took several hours per lung. Because of this, each case was processed only by one expert.

3.5.2 Quantitative Indices

The following quantitative error indices are utilized: Dice coefficient D [107], Hausdorff distance H [107], mean signed border positioning errors (d_s) [107], and mean absolute surface distance (d_a) [45]. In the case of d_s, a negative value indicates that the segmentation boundary is inside and a positive value indicates that the border is outside the reference.

The Dice coefficient is defined:

$$D = \frac{2|S \cap R|}{|S| + |R|} \tag{3.4}$$

where S represents the segmentation and R the reference. The $D = 0$ represents no overlap and $D = 1$ represents a complete overlap.

The Hausdorff distance is defined:

$$H = \max\{h(S,R), h(R,S)\} \tag{3.5}$$

where $h(X,Y) = \max\{d(x,Y) \mid x \in X\}$ and $d(x,Y)$ represents the unsigned Euclidean distance from the point x on the surface X to the surface Y.

The mean signed border positioning error is defined:

$$d_s = mean(\hat{d}_s(S,R)) \tag{3.6}$$

where $\hat{d}_s(S,R)$ represents all the signed Euclidean distances of the points on the surface S to the surface R.

The mean absolute surface distance is defined:

$$d_a = \frac{mean(d(S,R)) + mean(d(R,S))}{2.0} \tag{3.7}$$

where $d(X,Y)$ represents all the unsigned Euclidean distances of the points on the surface X to the surface Y.

3.6 Results

Segmentation performance measures averaged over all left lungs and right lungs are summarized in Table 3.1 for the proposed method as well as the ASM and RASM approaches. Table 3.2 summarizes the results achieved on the same data with methods P1 and P2 in combination with the results of a statistical comparison with the proposed approach. Corresponding boxplots of the Dice coefficient are depicted in Fig. 3.9.

Table 3.1: Comparison of overall performance between standard ASM, robust ASM (RASM), and proposed (RASM+OSF) lung segmentation approaches averaged over all left and right lungs processed.†

		ASM	RASM	RASM+OSF
D (-)	mean	0.848	0.936	0.975
	std	0.046	0.015	0.006
H (mm)	mean	35.06	22.77	20.13
	std	7.87	5.75	6.17
d_a (mm)	mean	5.47	2.24	0.84
	std	1.37	0.50	0.23
d_s (mm)	mean	-3.02	0.51	0.59
	std	1.14	0.41	0.39

† The mean and standard deviation (std) is given for each index.

Table 3.2: Overall performance measures for methods P1 and P2 averaged over all left and right test lungs.†

		P1	P2
D	mean (-)	0.844	0.949
	std (-)	0.106	0.012
	P-value (-)	2.47e-14	5.92e-20
H	mean (mm)	98.49	33.07
	std (mm)	50.88	7.69
	P-value (-)	1.11e-19	2.05e-15
d_a	mean (mm)	8.56	1.89
	std (mm)	6.14	0.45
	P-value (-)	6.64e-17	4.21e-20
d_s	mean (mm)	15.21	1.25
	std (mm)	12.76	0.72
	P-value (-)	2.31e-05	1.77e-09

† The mean, standard deviation (std) is given for each index. In addition, the P-value of a paired Wilcoxon rank sum test of the hypothesis that method P1 or P2 and our approach come from distributions with equal medians.

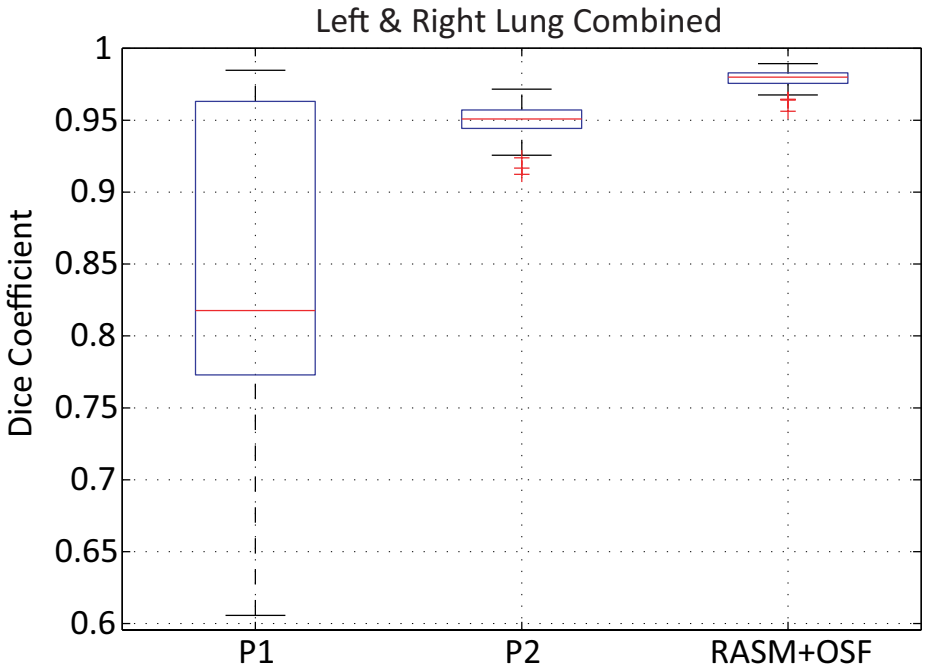
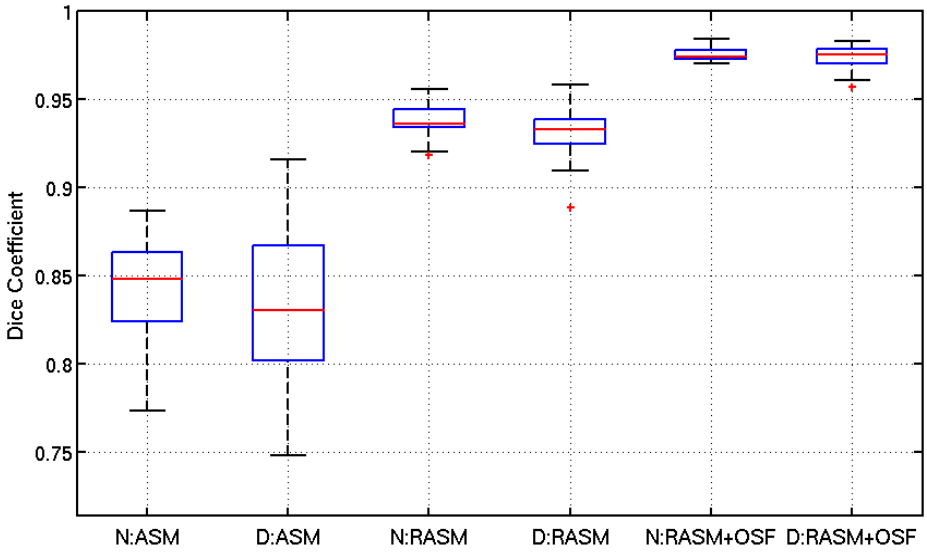


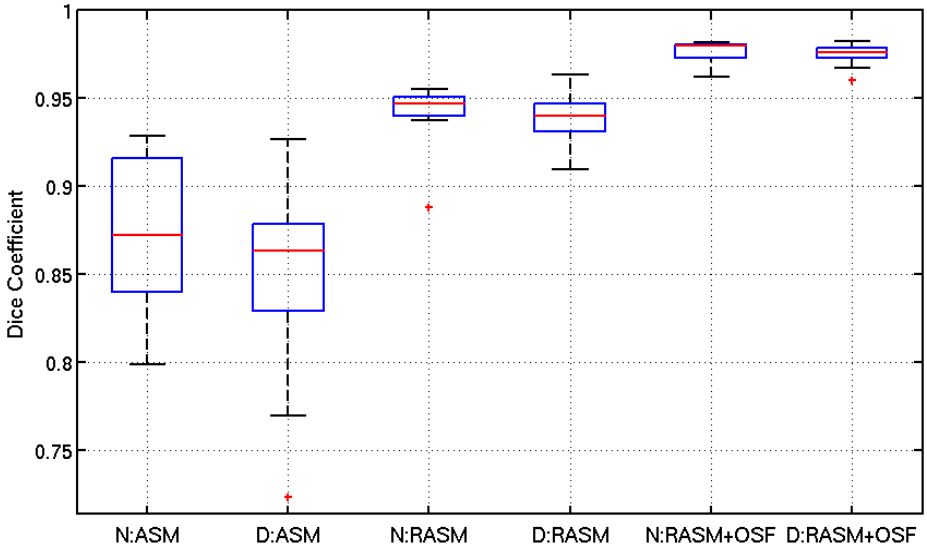
Figure 3.9: Boxplots of the Dice coefficient for P1, P2, and our approach.

Based on the Dice coefficient, the boxplots in Fig. 3.10 show a comparison among those three methods for left (Fig. 3.10(a)) and right (Fig. 3.10(b)) normal and diseased lungs, respectively. The mean and standard deviation of performance measures of our method for left and right lungs with and without disease is presented in Table 3.3.

Fig. 3.11(b) depicts a segmentation result which was generated by our method. For comparison, the reference segmentation is shown in Fig. 3.11(a). A segmentation of the same data set with a conventional approach was previously shown in Fig. 3.2(b). Additional segmentation examples are depicted in Fig. 3.12, which also shows corresponding results generated with methods P1 and P2.



(a) Left Lung



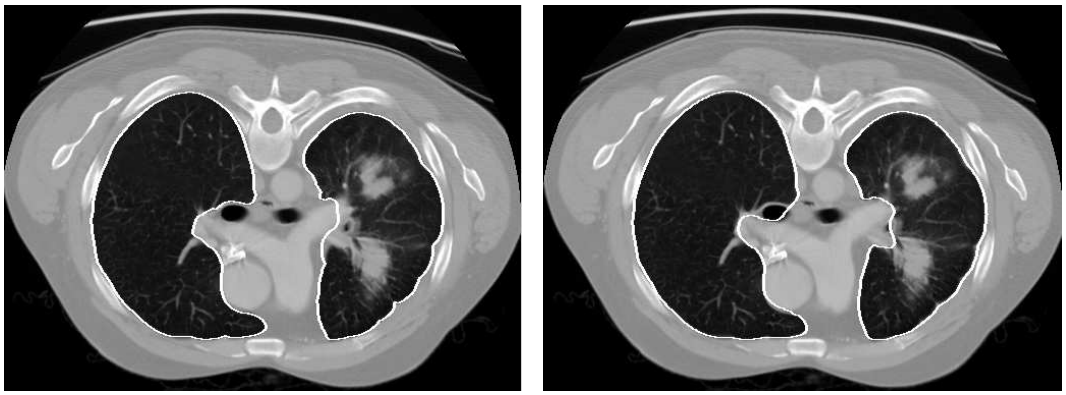
(b) Right Lung

Figure 3.10: Comparison of the Dice coefficient of standard ASM, robust ASM (RASM), and proposed (RASM+OSF) lung segmentation approaches for (a) left and (b) right lungs. Note that boxplots for normal (N) and diseased (D) lungs are shown separately.

Table 3.3: Segmentation results of the proposed method on normal left (NL), diseased left (DL), normal right (NR), and diseased right (DR) lungs.†

		Lung			
		NL	DL	NR	DR
D (-)	mean	0.975	0.974	0.976	0.976
	std	0.004	0.007	0.006	0.005
H (mm)	mean	18.63	19.32	21.64	21.00
	std	5.51	5.82	9.02	5.52
d_a (mm)	mean	0.75	0.84	0.94	0.85
	std	0.10	0.2	0.43	0.18
d_s (mm)	mean	0.44	0.50	0.92	0.60
	std	0.09	0.24	0.87	0.19

† The mean and standard deviation (std) is given for each index.



(a) (b)

Figure 3.11: Segmentation result for the example shown in Fig. 3.2(a). (a) Reference segmentation and (b) proposed segmentation approach.

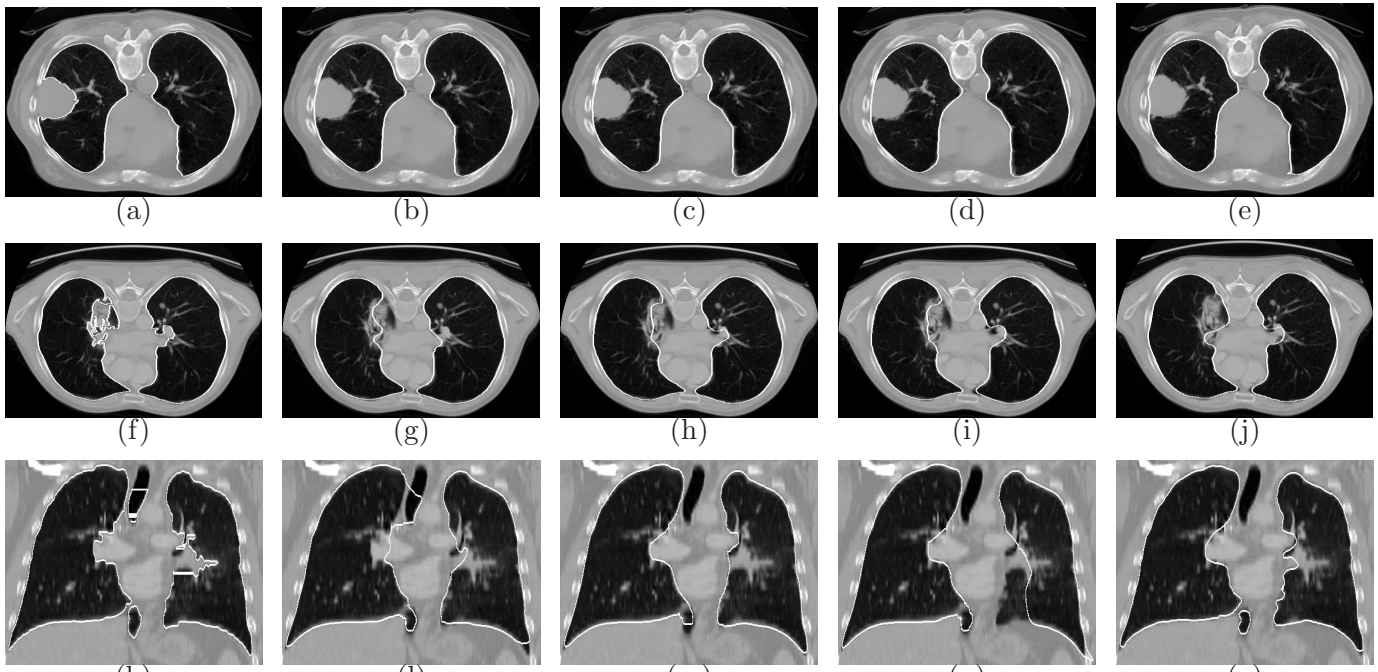


Figure 3.12: Examples of segmentation results on three different data sets (rows). (a,f,k) results for method P1. (b-d, g-i, l-n) Results generated with method P2 based on three different template initializations. (e,j,o) Results of the proposed automated approach (RASM+OSF).

On average, 2 minutes were required for calculating left and right model initialization parameters and another 2 minutes for segmenting a right or left lung, which resulted in an overall average processing time of 6 minutes per data set. In this context, the mean time required for the intrinsic robust ASM matching was only 24 seconds due to code optimization and utilization of GPGPU techniques.

3.7 Discussion

The robust ASM is an important component of the presented fully automated lung segmentation method. Results presented in Table 3.1 and Figs. 3.10 and 3.13 demonstrate that our robust ASM matching approach outperforms the standard ASM approach. Even on normal lungs, our robust ASM delivers superior performance. These results are not surprising, because standard ASM matching is a least squares optimization, which is sensitive to outliers. Since the model is only roughly initialized in proximity to the lung, all obstacles between model and target structure like aorta and vessels can cause problems. Our robust matching method can even handle missing data, as shown in Fig. 3.14.

Since our shape model was built from 41 data sets, smaller local shape variations cannot be explained by the model. The optimal surface finding step allows us to overcome this problem. A good initial match between model and image data is required for this processing step, and the performance of a standard ASM would not be sufficient for this task (Fig. 3.13). The boxplots in Fig. 3.10 show that the Dice coefficient is increased significantly by the optimal surface finding step for all

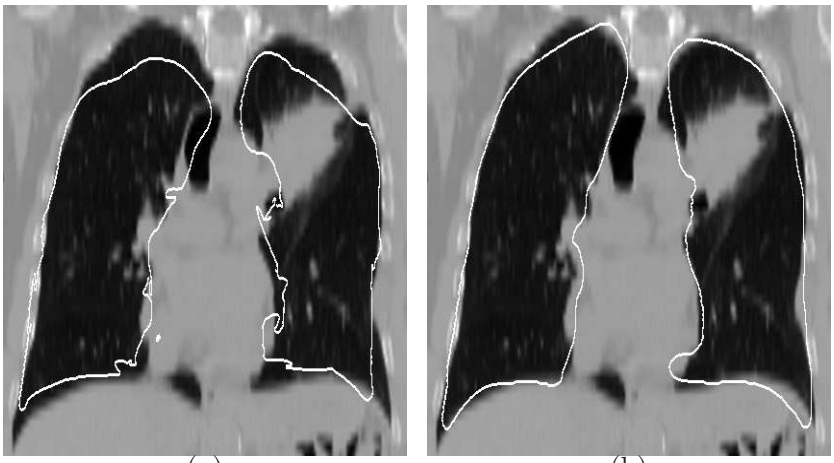
constellations of left/right and normal/diseased lungs. This is also clearly reflected in the averaged results for the Dice coefficient D, Hausdorff distance H, and mean absolute surface distance error d_a (Table 3.1). In case of the mean signed border positioning error (d_s) shown in Table 3.1, the value increases slightly after optimal surface finding, but is within the average dimension of a voxel. Table 3.3 shows performance measures for normal left/right and diseased left/right lungs, respectively. The results for the Dice coefficient are in a close range for all possible constellations. Distance-based error metrics for normal left lungs are somewhat lower compared to diseased left lungs. In the case of the right lungs, distance error metrics are higher for normal lungs compared to diseased lungs. The reason for this is that the size of one out of the nine normal right lung is extremely small compared to the corresponding left lung (Fig. 3.15(a)). Thus, the shape is significantly deviating from learned shapes. Consequently, the model does not initialize the optimal surface finding in close proximity to the target surface in this region, which results in a larger local distance error. If this case is excluded from the calculation, the following performance measures are obtained for normal right lungs: $D = 0.978 \pm 0.004$, $H = 18.93 \pm 4.19$ mm, $d_a = 0.80 \pm 0.15$ mm, $d_s = 0.63 \pm 0.17$ mm. These results show a similar pattern to the results for the left lung. The described problem can be addressed by expanding the learning shape set utilized for model generation (Section 3.4.1) such that similar variations are included. Another option would be to allow anisotropic scaling of the model during matching which would give it more flexibility.

In our segmentation results, we observed frequently major deviations from the

reference in hilar regions where airways and pulmonary vessels enter/leave the lung. Even for experts, it is hard to segment this area consistently.

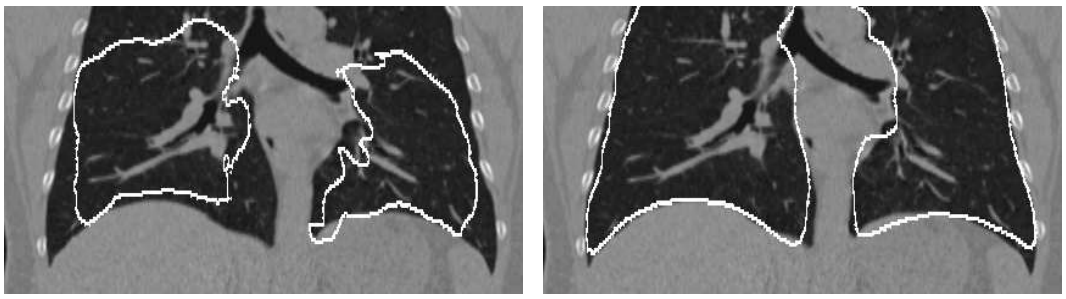
Currently, our method requires on average 6 minutes for the processing of normal or diseased lungs, consisting of initialization and sequential segmentation of the left and right lung. The core component of our approach is a novel robust 3D ASM matching algorithm which is suitable for large models and can run in parallel. For example, our GPGPU-based implementation required approximately 24 seconds for matching the RASM to a left or right lung. This demonstrates the feasibility of fast and robust model-based segmentation of large structures. In the current implementation, many processing steps are not optimized nor do they utilize GPGPU computing. For example, the segmentation of left and right lungs can be done in parallel and parts of the model initialization method can be optimized. Thus, the processing time can be significantly reduced, which is an important issue for routine utilization.

Segmentation of lungs with large lung cancer regions in chest CT scans is a nontrivial problem. Many of the currently utilized methods are prone to produce incorrect results, as shown in Fig. 3.2(b). Such methods typically rely on simple strategies (e.g., region growing), that do not incorporate knowledge about the shape of the target object. Problems with standard methods can even occur in case of normal lungs, as depicted in Fig. 3.16. This is also clearly demonstrated by our assessment of method P1 (Table 3.2) and corresponding examples depicted in Fig. 3.12. As a consequence, extensive manual post-processing of segmentations is necessary.



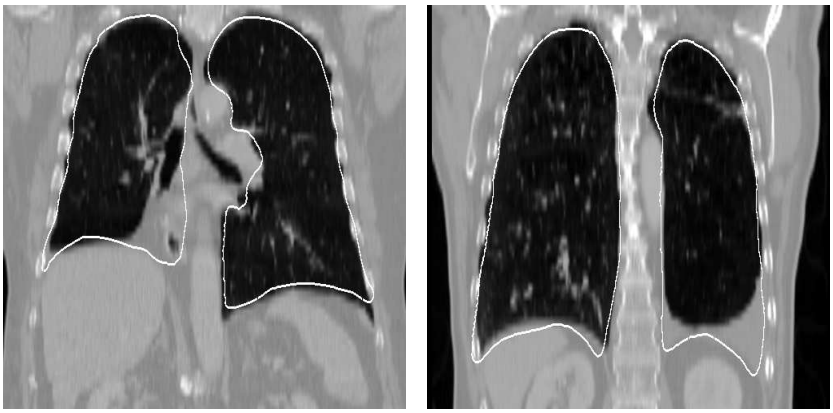
(a) (b)

Figure 3.13: Performance comparison between (a) standard ASM and (b) RASM matching (without optimal surface finding step). The RASM delivers a better match for normal and diseased lungs.



(a) (b)

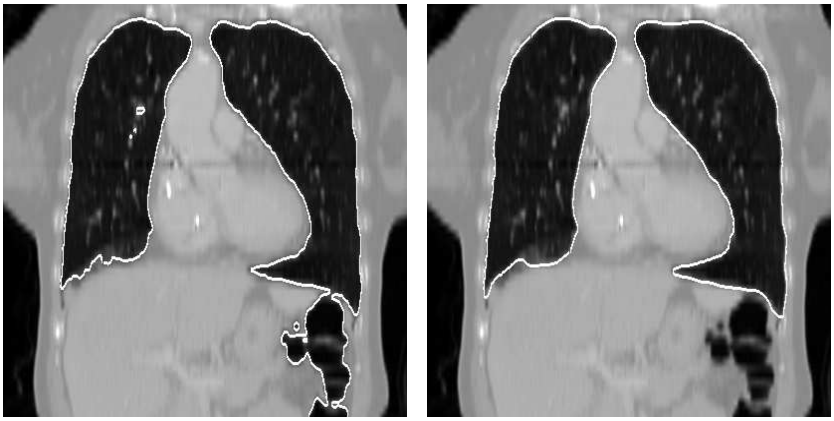
Figure 3.14: Segmentation of an incomplete lung CT data set; the top portion was not scanned. (a) Standard ASM. (b) Robust ASM. Note that the standard and robust ASM are not aware of the spatial extent of the data set, because of the clamping of gradient values to the boundary (Section 2.3.2). Surfaces outside of the data set were clipped after the segmentation process was completed.



(a) (b)

Figure 3.15: Examples of RASM segmentation results (without optimal surface finding step). (a) Case with small right lung and (b) pleural effusion in left lung. See text for details.

The deformable template approach (method P2) represents an improvement compared to method P1 (Table 3.2 and Fig. 3.9), but still shows local segmentation errors (Fig. 3.12). The result of method P2 can vary significantly with the initialization (Figs. 3.12(l-n)), which limits reproducibility. In some cases, method P2 provided correct segmentations of lungs with masses (Figs. 3.12(b-d)), while in other cases it consistently produced segmentation errors (Figs. 3.12(g-i)). Both methods are outperformed by the prosed approach (RASM+OSF) which shows statistically significant better results for all performance metrics (Table 3.2) and does not require manual initialization. In this context, it also interesting to note that method P2 and our RASM (without OSF) show similar values for the average Dice coefficient (Tables 3.2 and 3.1). However, when the Hausdorff distance and the mean signed boarder



(a) (b)

Figure 3.16: Comparison between conventional and proposed lung segmentation methods. (a) The conventional method leaks into the gas filled colon. (b) Our method provides a correct segmentation.

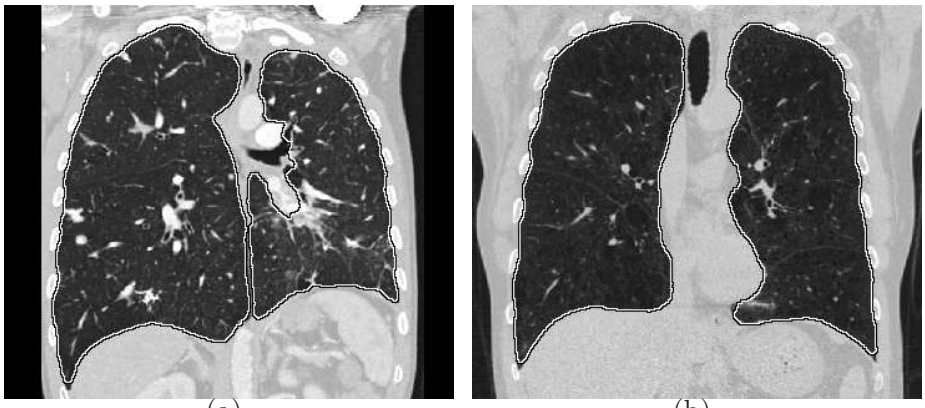
positioning error are considered, which are more relevant for refining a segmentation in a subsequent processing step (e.g., OSF), our RASM is the better choice, since it is on average closer to the true lung boundary.

We applied our method to the LOLA11 test set, consisting of 55 chest CT scans of normal lungs and lungs with a variety of different lung diseases and imaged with different scan protocols. For performance assessment, all lung meshes generated with the proposed approach were voxelized and sent to the LOLA11 organizers, which in return provided the volumetric segmentation overlap measures with respect to a ground truth[2]. In Table 3.4, the results for left and right lungs are shown which consist

[2]Details regarding the validation procedure can be found at http://www.lola11.com

Table 3.4: Results of lung segmentation for the 55 scans on LOLA11.

	mean	SD	min	Q1	median	Q3	max
left lung	0.939	0.173	0.0392	0.979	0.990	0.994	0.997
right lung	0.959	0.122	0.167	0.985	0.990	0.994	0.998
score	0.949						



(a) (b)

Figure 3.17: Examples of lung segmentations in CT images with (a) and without (b) contrast agent.

of the mean, standard deviation, minimum, first quartile, median, third quartile, and maximum overlap of the 55 test cases, as well as an overall score. In this context, the overlap between two binary segmentation volumes is defined as the volume of their intersection divided by the volume of their union.

Left and right lung segmentations show the same median overlap value of 0.990 (Table 3.4), which is an indication that in the majority of results generated with the proposed approach closely match the gold standard produced by the organizers of LOLA11. The examples depicted in Figs. 3.17, 3.18, and 3.19 confirm

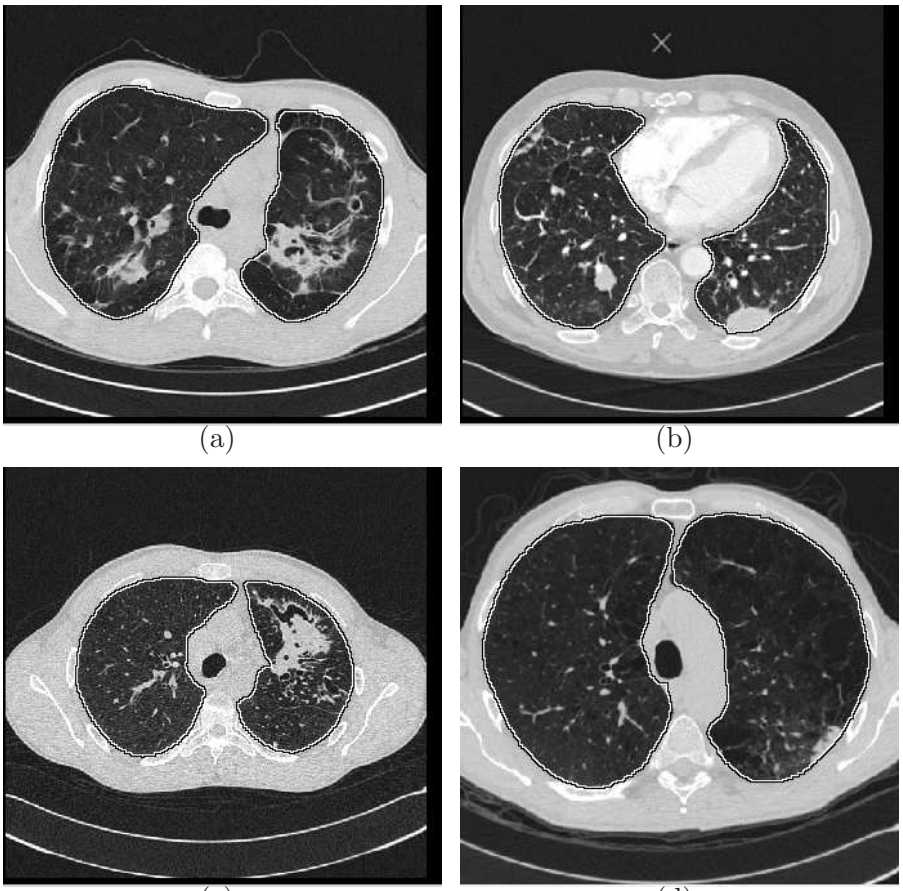


Figure 3.18: Examples of lung segmentations. The axial images depict lungs with different types of high density pathology.

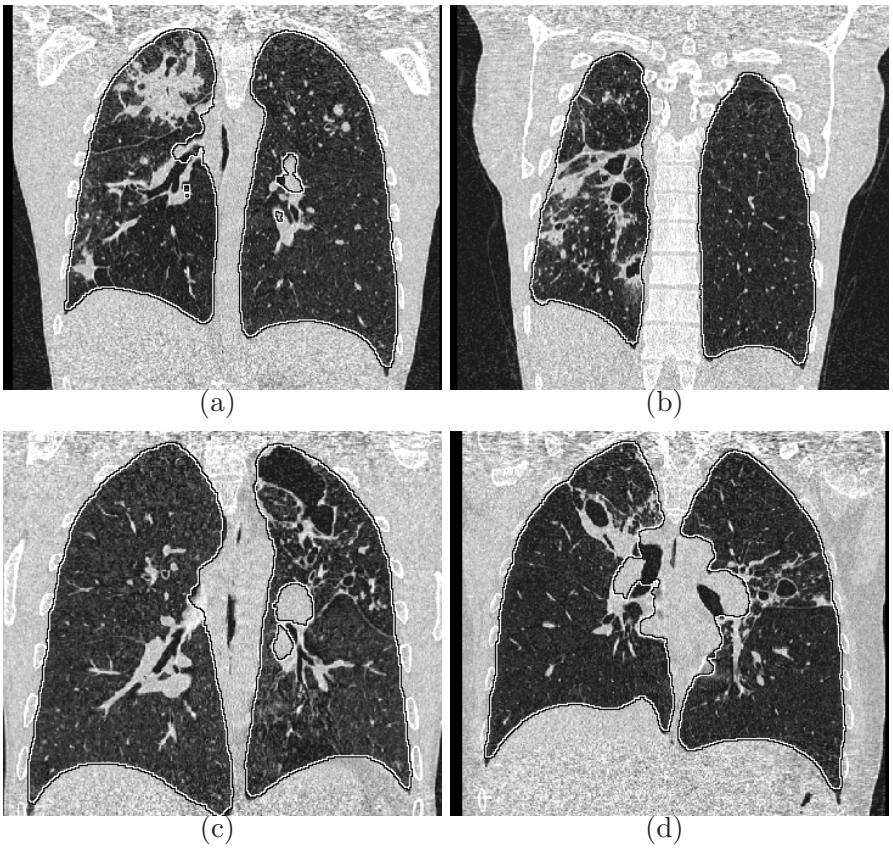


Figure 3.19: Examples of lung segmentations showing coronal views of lungs with high density pathology.

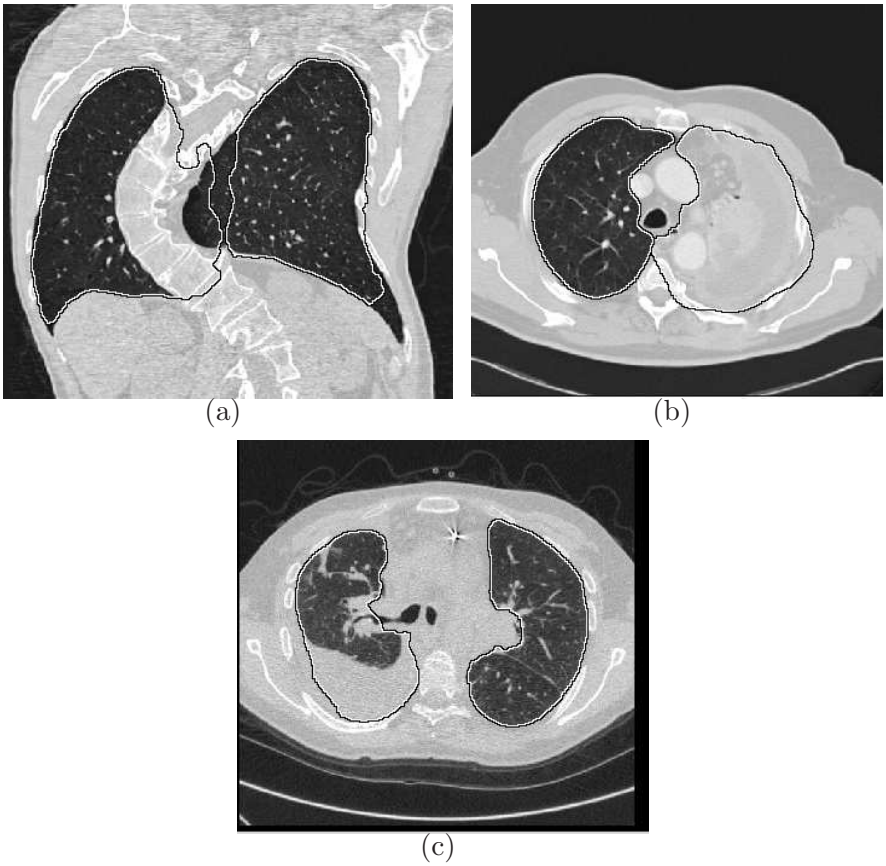


Figure 3.20: Examples of lung shape variation in the LOLA11 test set. For each CT scan, the corresponding segmentation results is depicted.

Table 3.5: Scores for all submissions to the LOLA11 challenge.

Method	Score
Method A [71]	0.973
Method B [123]	0.970
Method C [77]	0.963
Method D [117]	0.962
Method E [84]	0.952
Proposed method ([111])	0.949
Method F [70]	0.949
Method G [88]	0.948

this—segmentations of normal lungs and lungs with high-density pathology (e.g., lung cancer) show only small errors.

As can be seen from Table 3.4, the mean overlap value for segmented left and right lungs is below the median and first quartile (Q1). This indicates that our approach failed in a few cases. This is also reflected by the minimum overlap values shown in Table 3.4. Fig. 3.20 depicts some examples of segmentation errors. In cases where the lung shape widely deviates from the learned lung shapes (Fig. 3.20)—e.g. collapsed or partly removed lungs—model-based segmentation is challenging.

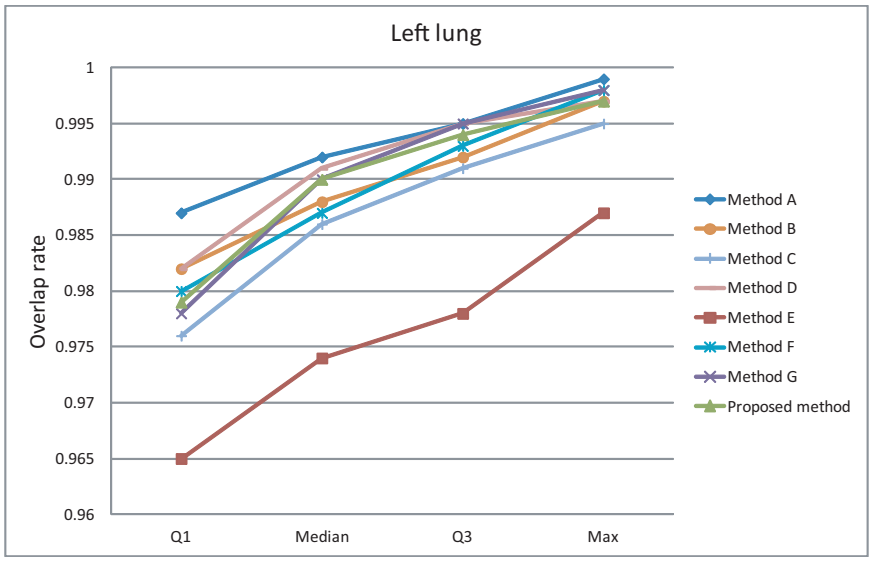
A comparison of the proposed approach, which was developed for lungs containing large lung masses, with other methods is shown in Fig. 3.21. The majority of the competing methods is based on a form of region growing with different pre- and/or post-processing steps. The final score (average of left and right mean results) of all eight methods is given in Table 3.5. In this context, note that region growing

based methods have a clear advantage in cases like the ones depicted in Figs. 3.20(a) and 3.20(c).

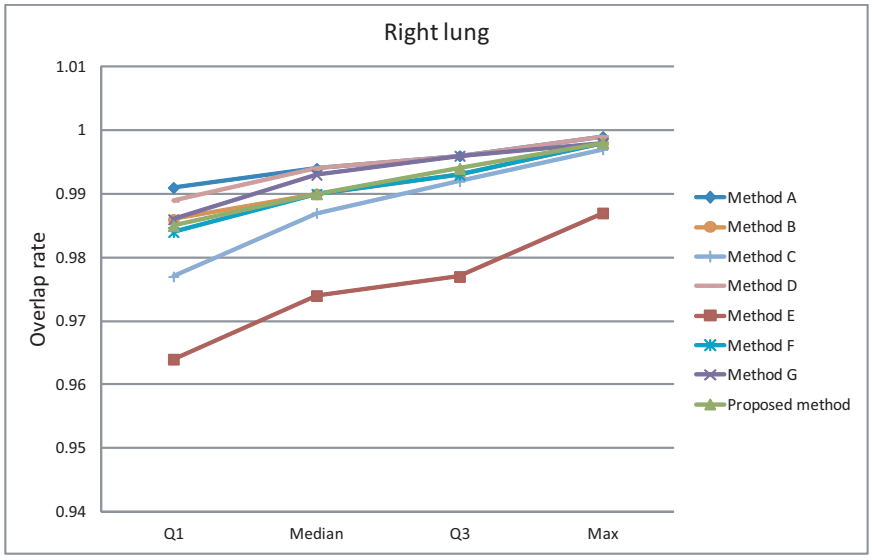
While the robust model matching method successfully deals with outliers and other disturbances (e.g., missing data) (Figs. 3.13 and 3.14), it requires learning data which is representative for the target population. If a lung shape is encountered that cannot be explained by the model, the shape needs to be added to the learning set. Such an approach allows to build a more complete model over time and will reduce the likelihood that similar problems are encountered in the future. Also, note that the optimal surface finding step after robust ASM segmentation reduces the need to add new lung shapes to the learning set.

In general, cases of pneumothorax or pleural effusion are difficult to segment automatically, and our model-based approach might require some additional processing steps. For example, Fig. 3.15(b) depicts a robust ASM matching result of a thorax CT scan with a pleural effusion. The left lung model approximates the "normal" lung location and matches with the diaphragm at the bottom. To segment the left lung, some additional steps are needed and might also allow to quantify the pleural effusion volume, which is of interest to physicians.

For our experiments, we have utilized a simple cost function based on gradient magnitude and direction. Thus, performance can be further improved by utilizing more complex cost functions for model matching and optimal surface finding, which could be based on the relative location of shape points as well as density/gray-value properties and shape features. For example, currently the detected ribs are only



(a)



(b)

Figure 3.21: Comparison among all methods (Table 3.5) submitted to the LoLA11 challenge with first quartile (Q1), median, third quartile (Q3), and maximum (Max) overlap values. (a) Left lung. (b) Right lung.

utilized for model initialization, but can provide valuable information for cost function design. Also, the proposed work targets larger cancer masses and is not optimized for handling juxtapleural nodules. Again, this problem can be addressed by adapting the cost function as well as the formulation of the smoothness constraint utilized for optimal surface finding.

In the current version of our algorithm, left and right lungs are segmented separately, which can lead to inconsistencies (e.g., overlap). This problem can be solved by utilizing a multiple surface graph search approach as described in [75].

In some cases the optimal surface segmentation has problems in segmenting areas with sharp angles like the area where the diaphragm meet the ribs (costophrenic angles). To solve this problem, locally more dense mesh vertices in combination with an adaption of search profiles will be required.

(a) (b) (c)

Figure 3.22: Example segmentation results of lung with idiopathic pulmonary fibrosis. (a) A manual reference segmentation. (b) Result of a conventional segmentation method. (c) Preliminary segmentation result of our approach (RASM+OSF).

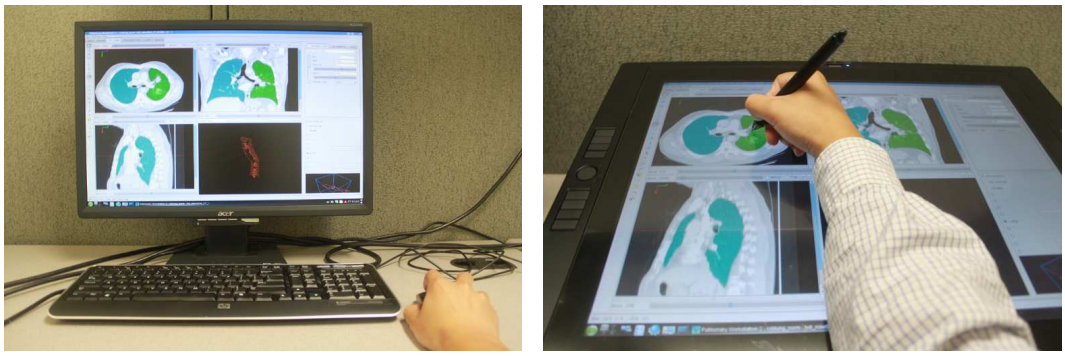
Preliminary work investigating the applicability of the lung segmentation method to lungs with other kinds of diseases like idiopathic pulmonary fibrosis is promising, but needs to be addressed in a separate research effort (Fig. 3.22).

CHAPTER 4
GENERIC INTERACTIVE OPTIMAL SURFACE FINDING BASED SEGMENTATION REFINEMENT

4.1 Introduction

For OSF-based segmentation, the cost function plays a critical role in achieving good segmentation results. Designing a cost function that delivers correct results for all possible anatomical and pathological variations is difficult. When automated OSF-based segmentation approaches fail to deliver a correct result, user intervention is required to overcome this problem. The main idea is to manipulate the cost of the graph nodes in the failure region by means of user interaction to guide the segmentation to the correct object border.

In the case of conventional (2D) segmentation refinement, contours are manipulated in a slice-by-slice manner, which is tedious, time consuming and error prone. Fig. 4.1(a) shows one example when an expert used mouse and keyboard to correct the segmentation errors. Alternatively, a tablet and stylus setup as shown in Fig. 4.1(b) which may improve efficiency. In this chapter, an OSF-based approach for the interactive manipulation of surfaces is presented to speed up the refinement process. In addition, to facilitate interaction with surface, an interactive VR system will be utilized.



(a) (b)

Figure 4.1: Conventional 2D viewer based segmentation refinement. (a) 2D monitor set up with 2D input device (mouse and keyboard). (b) Tablet setup with a stylus.

4.2 Related Work

4.2.1 Interactive Segmentation and Segmentation Refinement

Interactive segmentation refinement techniques are required when automated segmentation algorithms are not successful. Segmentation refinement is used to correct the errors of the automatically generated result. In this section we review related interactive segmentation and segmentation refinement techniques.

The simplest form segmentation is pure manual segmentation. In 2D, the user directly draws a contour of the object. In 3D, the user performs the contour drawing slice by slice to define an object's surface. Pure manual segmentation is often used to generate a ground truth for validation. This is a very time consuming process.

Computer-aided (semi-automated) methods require manual interaction for subsequent automated processing steps. The interaction often includes placing con-

trol points indicating the target border or seed points for indicating the location of "object" or "background" etc. One of the semi-automated approaches is contour or surface interpolation from control points. For interpolation, BSpline or thin plate spline (TPS) interpolation algorithm can be used. For example, Ross *et al.* [96] proposed an interactive lung lobe segmentation approach by constructing pulmonary fissure surface from user placed seeds by means of TPS interpolation.

Another example of an interactive segmentation approach based on control points is the live wire approach [5]. The user interactively places control points at the object border, which are automatically connected by a graph search based method. The contour is the minimal cost path according to utilize image features. Live wire based approaches were also extended to 3D (volumetric) data sets [98]. For example, Schenk *et al.* [98] applied a conventional live wire algorithm to some slices and then utilized shape interpolation techniques to generate contours for the remaining slices of volumetric data set from segmented contours. To reduce user interaction, Salah *et al.* [97] developed a method for propagating the control points of the key slices to the successive unprocessed slices. Because organ contours on the successive slices are usually not deviating too much from each other, the adaptive propagation is performed by searching for local optimal position in terms of gradient magnitude. The propagated control points are then used as input for a conventional live wire algorithm.

Interactive graph cuts [16] requires the user to mark object and background before calculating a globally optimal segmentation based on image features.

Schwarz *et al.* [100] proposed an interactive surface editing framework to refine the result of ASM segmentation. This tool allows user to move one surface point to expected boundary in the case of local error. New locations of surface points near to this surface point were determined automatically by the algorithm. Finally, the shape model is updated by taking all new locations into account.

To facilitate interactive segmentation, desktop based graphics user interfaces (GUIs) were developed. For example, 3D Slicer is the general purpose medical imaging computing and visualization framework of the National Alliance for Medical Image Computing (NAMIC). The framework provides an interactive editor for interactive segmentation algorithms like interactive "grow cut" segmentation [118]. Maleike *et al.* [34] proposed the Medical Imaging Interaction Toolkit (MITK). The toolkit [34] includes semiautomatic segmentation methods, segmentation result editing, shape based interpolation for unsegmented slices, multi-level modifications to binary images etc.

A sparse number of interactive segmentation tools involving true 3D interaction and virtual reality techniques have been proposed. Senger [101] presented an immersive virtual reality environment for medical image visualization and segmentation based on region growing. The stereoscopic visualization was supported by an immersive workbench which projected images onto the underside of a translucent table surface and the seed points were selected by a magnetic tracked probe [101].

Harders *et al.* [49] presented a virtual reality based system for interactive segmentation of tubular structures. The stereoscopic visualization assisted the user in

navigating through the volumetric data set. The 3D user interaction was enabled by utilizing a haptic feedback (force feedback) device. The force map was first precomputed before the actual interaction. The calculated force was used to guide user in drawing the centerline of the tube structure in order to generate an initial shape description of the tubular model. When image features are not clear, the user can utilize his/her knowledge to draw the centerline by exerting force on the haptic device. The final segmentation was realized by fitting a deformable model. The effectiveness of the proposed haptic feedback based interactive segmentation tool was later demonstrated in an application of the intestinal tract segmentation [50].

Bornik *et al.* [15] proposed different mesh editing tools as part of a virtual liver surgery planning system [14], using both 2D and 3D interaction techniques in a VR environment. In this framework [15], a presegmentation result was converted into a simplex mesh [36], a deformable model utilizing Newtonian law of motion involving internal and external forces. The direct and indirect segmentation refinement operations were applied on the simplex meshes by controlling internal and external forces and minimizing the energy functional. In the case of small segmentation error, the dragging operation allowed the user to directly move the surface vertices to the expected position. This direct operation did not involve any form of energy minimization. In the case of larger segmentation errors, the force framework was utilized. Three external force based tools were developed: sphere, plane, and TPS template tools. When utilizing the sphere deformation tool, the user can move the sphere in the 3D space by a 3D cursor and can utilize this form element to manipulate the sim-

plex mesh surface. The plane deformation tool is much like the sphere deformation tool, except that a different form element is utilized. The TPS template deformation tool was proposed to fix major segmentation errors. The user could draw some contours in the 3D space. Through the contours, the TPS plane interpolation was used to calculate a surface, and the object surface automatically deformed towards the calculated surface.

As demonstrated in a recent study on liver segmentation in [9] and [11], segmentation refinement performed with VR-based tools was found less time consuming compared to standard 2D refinement. In [15], [9] and [11], refinement is solely driven by the user without utilizing image segmentation algorithms during refinement. Thus, the user needs to manually "drive" the surface to match object boundaries visible in the image data.

In this work, we presented a novel interactive 3D segmentation refinement method based on the OSF segmentation framework. The basic idea behind this method is that the user interacts directly with the segmentation algorithm to effectively correct errors in occurring automatically generated OSF segmentation results. Our approach utilizes a hybrid desktop/VR user interface. Before describing our hybrid desktop/VR user interface, VR technology will be discussed in the next subsection.

4.2.2 Virtual Reality

Virtual reality (VR) utilizes computer graphics to create a virtual 3 dimensional world, where the user can interact with virtual objects. In our application, we will use VR to interactively manipulate segmentation results. The utilized VR system consists of two major components: (1) a stereoscopic display system, and (2) an optical tracking system for head and interaction tool tracking.

In this section, we are going to review VR components. A more comprehensive review about the VR technology can be found in the book "Virtual Reality Technology" of Burdea and Coiffet [19].

Stereoscopic visualization is a fundamental component of virtual reality and allows to create depth illusion. In reality, we see an object from a slightly different perspective (binocular disparity) with each eye, because our eyes are separated horizontally by a distance. The binocular images are then processed by the brain to extract the depth information. In the virtual world, we need to render a stereo image pair — one image for each eye — with a binocular disparity in such a way that they will enable realistic perception of 3D objects.

Stereoscopic display devices are required to visualize stereo pairs. The following type of devices are frequently utilized: head-mounted displays, auto-stereoscopic displays and spatial displays. The displays include: cathode ray tube (CRT), liquid crystal display (LCD), digital light processing (DLP) displays. LCD and DLP active stereo displays offer good display quality at a low cost. These displays are able to generate at least 50-60 stereo frames per second, which is required to create a realistic

user experience. Such displays require the user to wear shutter glasses to selectively filter images for each eye.

For stereo displays, several 3D stereo mode formats are available: quad buffer mode, packed frame, side by side, top-bottom and checkerboard etc (see Fig. 4.2 for some examples). The quad buffer mode requires a professional graphics card like the Nvidia Quadro Fx series[1] with compatible displays. The quad buffer mode (Fig. 4.2(b)) uses four rendering buffers - left back, right back, left front and right front buffers (images are swapped from back buffers to front buffers for display). The stereo pairs are sent to left/right buffers respectively. It is a full resolution mode, and this allows to produce high quality 3D visualization. The packed frame mode (full resolution top-bottom for example is supported in HDMI 1.4[2]), shown in Fig. 4.2(a), packs two full resolution 1920×1080 frames into a single 1920×2205 frame (there is an 45 pixel divider between top and bottom). Although it is full resolution, it has to sacrifice refresh rate (more than half refresh rate of the quad buffer mode for each image). Side by side, shown in Fig. 4.2(d), and top-bottom mode, shown in Fig. 4.2(e), are similar. In these two modes, the full resolution stereo pairs are first shrunk to half resolution frames respectively and then organized into one full resolution frame where each occupies one half frame (left/right or top/bottom). The checkerboard mode, shown in Fig. 4.2(c), alternates between pixels from a stereo-pair and then forms a checkerboard pattern in one frame. In this mode, half of the vertical

[1]http://www.nvidia.com/object/quadro_pro_graphics_boards; accessed September 2011

[2]http://www.hdmi.org/manufacturer/hdmi_1_4; accessed September 2011

and horizontal resolution are lost. The stereo pairs are transmitted to a stereoscopic display device. It displays the original size frames (image interpolation is needed for the none full resolution modes) in a time sequential way. The active shutter glasses are synchronized by a signal sent from the graphics cards in quad buffer mode or stereoscopic display devices in the other mentioned modes.

To track the user or input devices of a VR system, tracking devices are required. The devices communicate with VR system about the position and the orientation of real objects in physical space, which enable spatial consistency between real and virtual objects. Typical VR tracking system can track objects in six degrees of freedom (6-DOF), which are the object's position within the (x-, y- and z-) coordinate system and the object's orientation about three perpendicular axes (pitch, yaw and roll).

There are different kinds of tracking systems used in VR systems. For example, sensor-based, vision-based and hybrid tracking techniques are quite common. The sensor-based tracking system utilizes magnetic, acoustic, inertial, optical and mechanical sensors based techniques [135]. A detailed survey about sensor based tracking system can be found in [95]. Vision-based tracking techniques utilized image processing techniques to calculate the camera pose relative to real objects [135]. In hybrid tracking system of vision- and sensor-based tracking techniques are combined [135]. In this work, an optical sensor based tracking system is utilized.

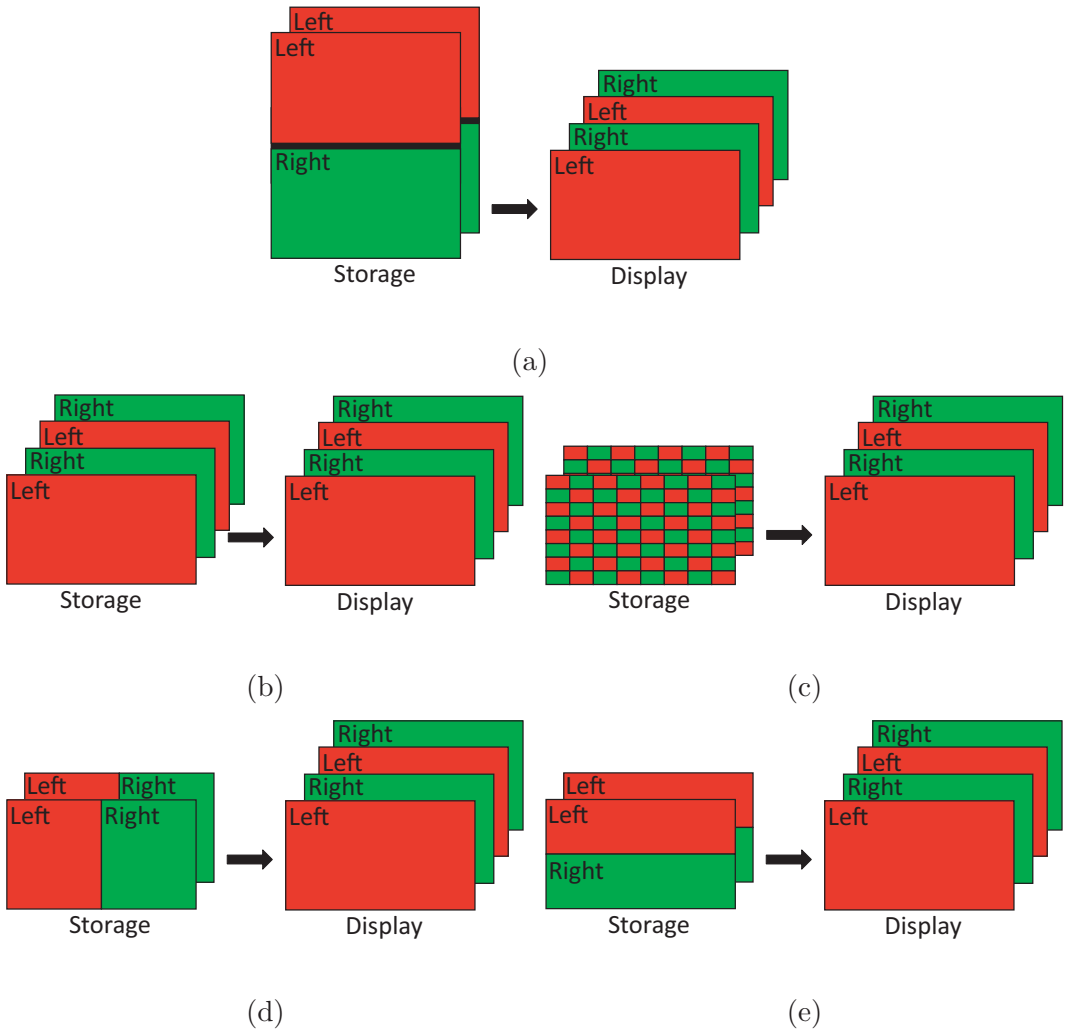


Figure 4.2: Illustration of stereo rendering mode formats. Red represents left eye image and green represents right eye image. The left side of the arrow indicates storage mode (graphics render mode) of the stereo pair, which are transmitted to stereo devices, and the right side of the arrow indicates sequential frame output of the stereo devices. By wearing shutter glasses, the user can only see the left frame with the left eye and the right frame with the right eye. (a) Full resolution top-bottom mode. (b) Quad buffer mode. (c) Checkerboard mode. (d) Side by side mode. (e) Top-bottom mode.

4.3 Generic OSF-based Segmentation Refinement Approach

For segmentation refinement, we assume that an initial segmentation was generated with an OSF segmentation approach that contains local segmentation errors, and thus, requires refinement. The proposed refinement approach builds on the initially utilized OSF framework (Section 3.4.4), but allows the user to locally guide the segmentation result. This section is organized as follows. First, we describe the hybrid desktop/VR user interface utilized for refinement. Second, we present the generic framework for interactive OSF-based segmentation refinement approach.

4.3.1 Hybrid Desktop/VR User Interface

We utilize a combination of desktop (2D) and VR (3D) user interfaces similar to the work reported in [15]. Refinement can be accomplished by using a stereoscopic display with a tracked (6 DOF) input device or a standard 2D interface (e.g., monitor and mouse) for more accurate control. The hybrid user interface reported in [15] utilized a distributed architecture with two computers. Thus, all operations, data, and displays needed to be synchronized over the network between 2D and 3D user interface computers. The drawback of such an approach is that large data sets might slow down communication and lead to low frame rates, besides potential network latency issues. Since a responsive system with high frame rates is essential for an interactive VR system, we have developed a hybrid user interface where 2D and 3D interfaces are implemented on the same machine.

The hardware setup consists of several components and includes an active

stereo display, an optical tracking system, a Wacom interactive pen display (Wacom Co., Ltd., Japan), and a graphics workstation (Fig. 4.3 and 4.4).

A Mitsubishi 3D DLP HDTV with 73 inch diagonal, 1920 × 1080 pixels, and refresh rate of 120 Hz was utilized as active stereo display (Fig. 4.3). It was operated in the side by side stereo mode in combination with Nvidia 3D Vision (Nvidia Corp., Santa Clara, CA) stereo shutter glasses (Fig. 4.5(c)). The TV generates an infrared synchronization signal for the shutter glasses. The infrared emitter is shown in Fig. 4.3 and circled in blue. In addition, because the tracking cameras use IR flash lights too, to avoid interference they are also synchronized with the shutter glasses. To avoid tedious calibration process, the 3D TV is tracked by the tracking system. Thus, the display needs to be calibrated only once the system is installed. The stereo display and Wacom display are driven by a Linux workstation with four 6-core 2.93 GHz Xeon CPUs and a Nvidia Quadro Fx 5800 graphics card (Nvidia Corp., Santa Clara, CA).

We use the optical tracking system "OptiTrack" (NaturalPoint Inc., Corallis, OR), which delivers 6-DOF tracking data. Five OptiTrack V100:R2 optical tracking cameras are mounted on a triangle shaped frame (2.22 × 2.22 × 3.6 m) which is mounted to the ceiling. The tracking cameras and our designed camera framework are shown in Fig. 4.3. The location of the cameras and the frame were optimized by means of simulation (Fig. 4.6). Each camera has a horizontal field of view (FOV) 57.5° and capture distance from 0.5 m to 7 m. In our VR setup, we track the user's head, the display and the input device. Only one user (active user) is tracked,

therefore the scene is rendered for this user. In Fig. 4.3, the active user wears the tracked shutter glasses. Other (passive) users can view the scene via shutter glasses but without any interaction. The input device for the hybrid environment is named "Hawkeye" (Fig. 4.5(b)). The Hawkeye consists of a Bluetooth mouse, a Wacom panel stylus and the tracking target. It offers three interaction modes: 2D interaction with Wacom panel via the stylus, 3D interaction via tracking target and mouse button events via Bluethooth mouse. The Bluetooth mouse button events are handled by OpenTracker [93] tracking library, supporting system mouse events. An example for the 3D interaction mode is shown in Fig. 4.3 and 2D interaction mode is shown in Fig. 4.4.

The independent tracking server and the rendering server are connected via TCP/IP LAN. Alternatively, if the tracking server and rendering server are runing on the same machine, the loopback interface can be used to set up the communication between the two servers. The tracking software is running on the tracking server, and the rest of the software is running on the rendering server. The Virtual Reality Peripheral Network (VRPN) protocol [60] is utilized over TCP/IP and carries the tracking data from the tracking machine (VRPN server) to the rendering machine (VRPN client). The tracking data is received by the rendering machine via OpenTracker [93] tracking library supporting VRPN client protocol.

In the following paragraphs, we will introduce our software framework and modules used for VR setup. The software involving the interactive refinement will be explained in the subsequent sections.

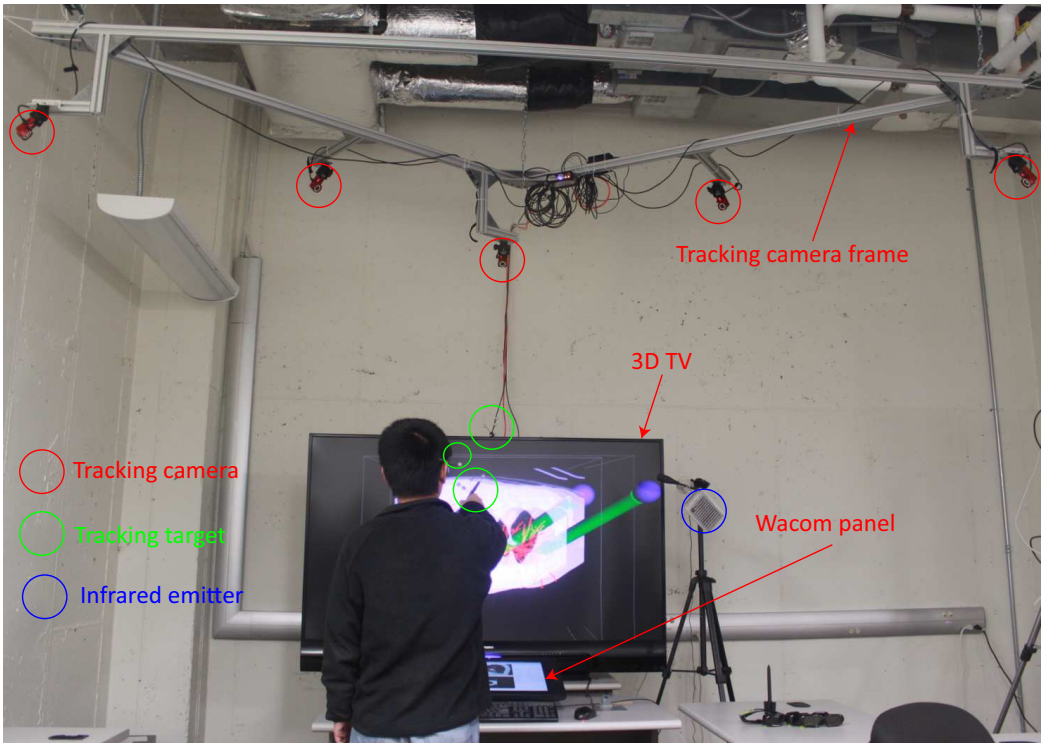


Figure 4.3: Overview of the hybrid VR system–a user explores the object in a true 3D environment. Note that the tracking and rendering server (a workstation) is not shown in this figure.

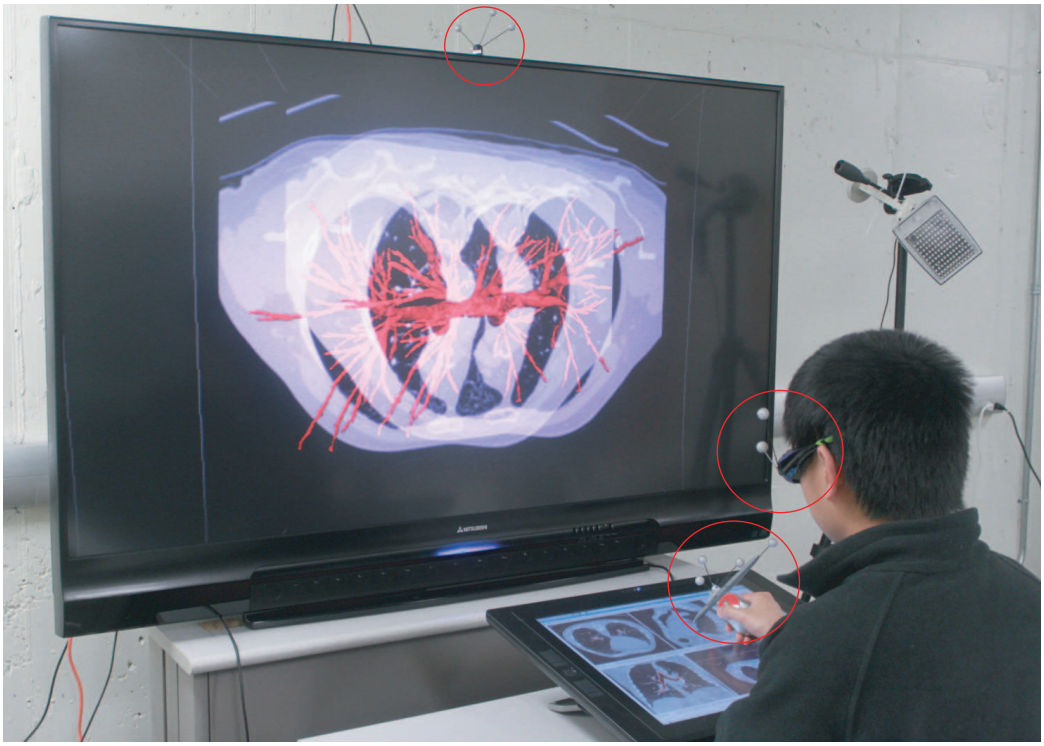
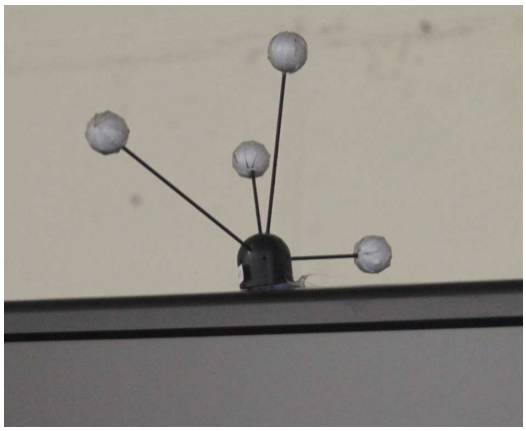
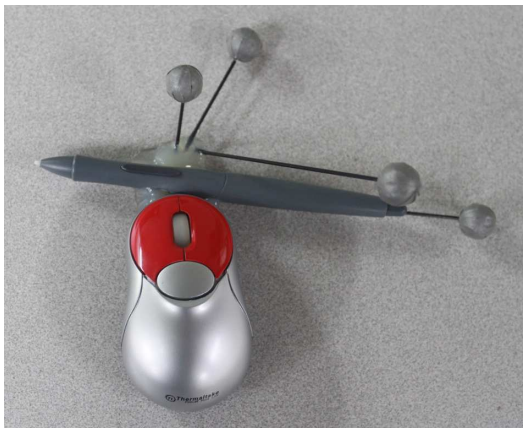


Figure 4.4: Hybrid VR system utilizing 2D interactive mode–the user utilizes the 2D user interface consisting of Wacom panel and Hawkeye. The tracked targets (TV, Hawkeye and 3D glasses) are circled in red.



(a)



(b)

(c)

Figure 4.5: Tracking targets for interactive segmentation in VR environment. (a) The tracking target is mounted on the top of the 3D TV. (b) The tracked Hawkeye consists of the tracking target, Bluetooth mouse and Wacom panel compatible stylus. (c) Tracked 3D glasses for the active user.

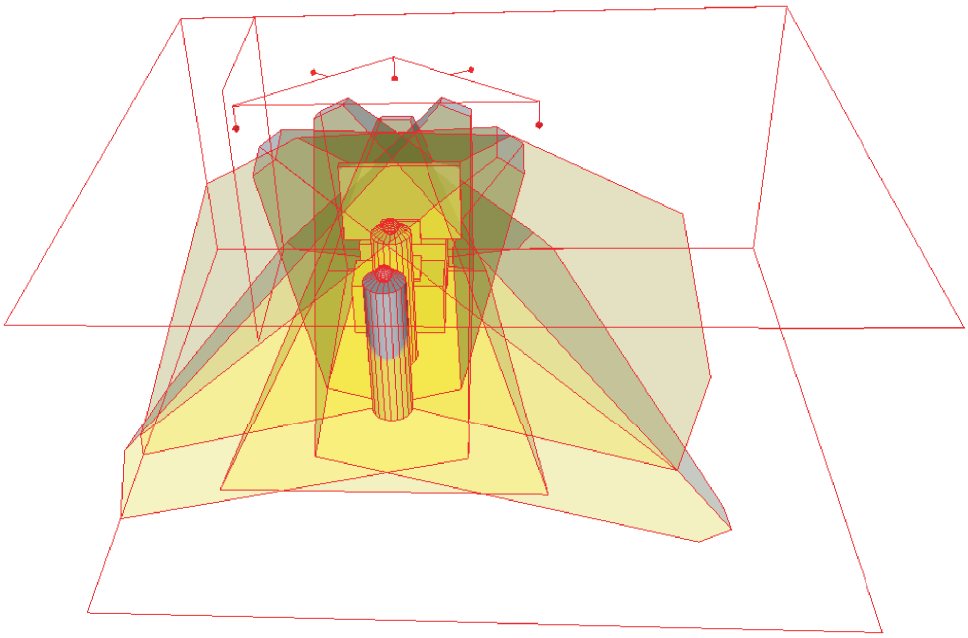


Figure 4.6: Simulation of the by the tracking system covered volume. Five cameras (cubes) are mounted on a triangle. Users are simulated by a sphere and cylinder. Displays and tables are simulated by cubes. The yellow frustums show the tracking volumes. Ceiling, walls and ground are shown as rectangles. The implementation of this simulation is shown in Fig. 4.3

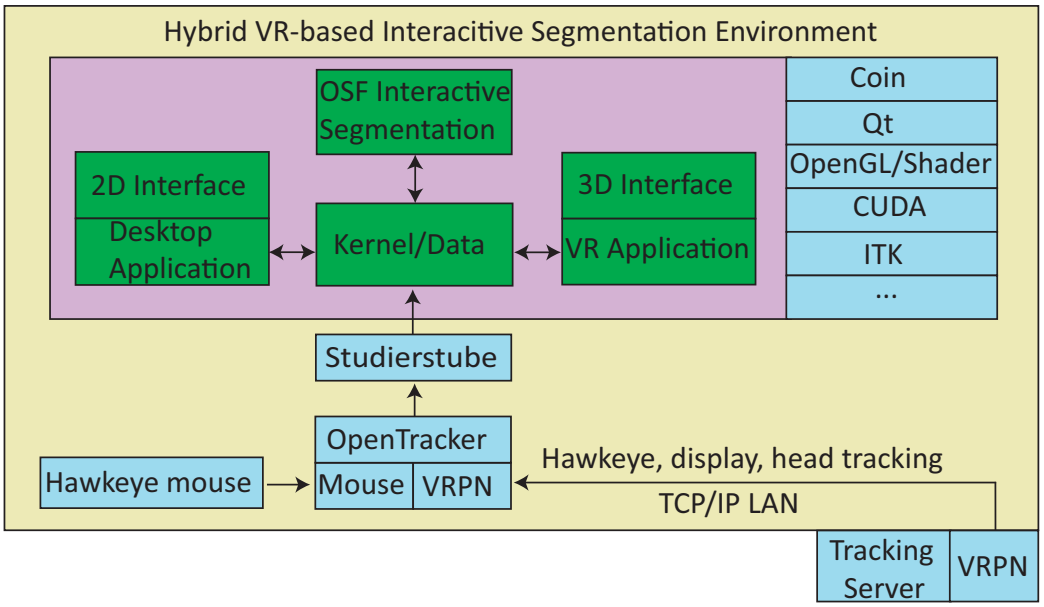


Figure 4.7: Software structure of hybrid VR-based interactive segmentation.

The main software structure of our hybrid VR-based interactive segmentation environment is shown in Fig. 4.7. The main external libraries for the hybrid system are Studierstube [99] and OpenTracker [93]. The Studierstube is a group of C++ classes developed for Augmented Reality (AR) and VR applications. The classes are based on Coin using the Open Inventor scene graph concept. In our application, the visualization algorithms are developed in Coin based on scene graph nodes. We extended the original Studierstube library concerning stereo rendering as described below. Studierstube provides an interface to Opentracker for receiving the tracking data and allows forwarding it to our applications. Our application consists of four main blocks: the kernel and data module, 2D desktop application module, 3D VR-based application module and OSF-based interactive segmentation module.

In order to acquire accurate tracking data, we need to calibrate the tracking system. The first step is to set up a physical (world) coordinate system W. The 6-DOF tracking data refer to W. In addition, to achieve a correct stereo view volume, the z-axis of the physical coordinate system needs be perpendicular to the stereoscopic display (in the word of computer graphics, the display is a projection plane). The position relation between W and the display should be constant. However once the relation changes, we have to recalibrate the VR coordinate system. This is a time consuming task. To avoid the repeated re-calibration, we transform W to the display by utilizing tracking data, thus the relation stays constant. The reference matrix M_{ref} is calculated with Eq. 4.1.

$$M_{ref} = {C_{cal}}^{-1} M_{dis}, \tag{4.1}$$

where M_{dis} is a tracking data matrix consisting of the display location and orientation. C_{cal} is a constant matrix consisting of the calibrated display location and orientation. The transformation of tracking inputs is shown in Fig. 4.8.

The off-axis perspective projection [55] and head tracking are combined to create a stereo pair for display. To make the stereo rendering portable to various display devices and stereo modes, a two pass rendering algorithm is utilized as shown in Fig. 4.9. In the first pass, the stereo pair is rendered to two independent frame buffer objects (FBOs) at full screen resolution. In the second pass, each image from the respective FBO is used as a texture source. The texture images are attached to corresponding viewer (quad buffer mode) or viewers (side by side etc). For the checkerboard mode, the stencile buffer may be used to combine two textures into

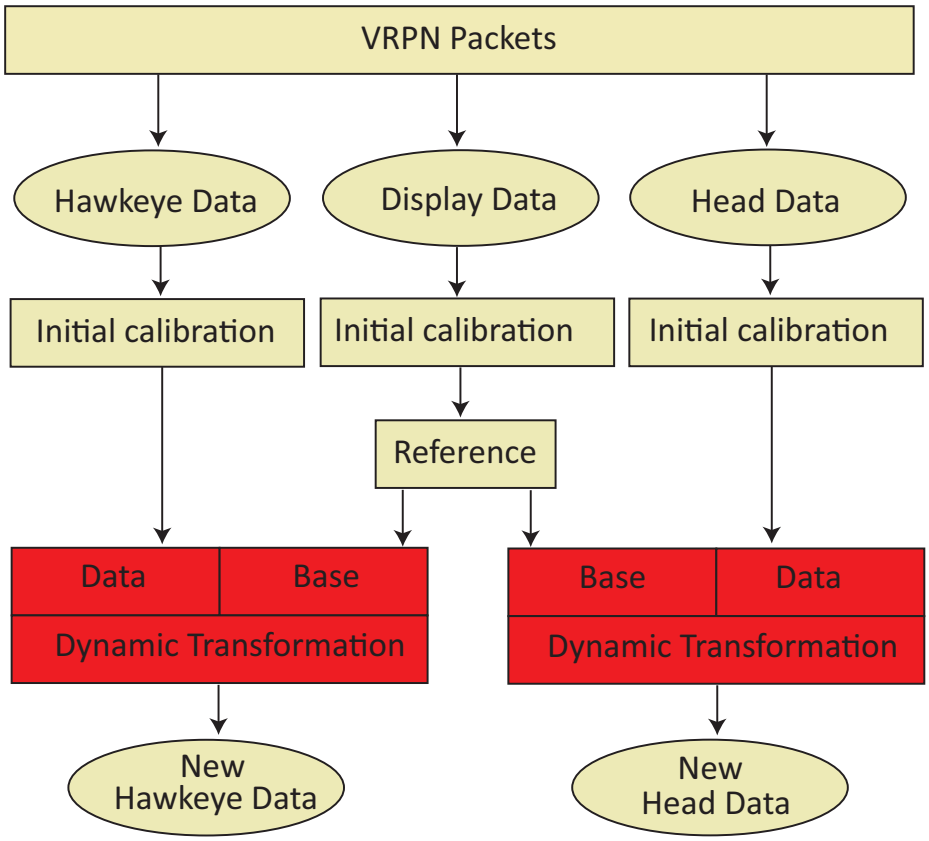


Figure 4.8: Graph outlining the transformation of Hawkeye and head tracking data relative to the display location/orientation. The red blocks show a dynamic transformation utilizing $M_{ref}(\cdot)$, where M_{ref} is indicated by "Base" in the graph. Hawkeye and head tracking data are transformed by static offsets based on calibration before they are feed to reference transformation block.

one texture. In the first rendering pass, FBOs are also used for other purposes, for example real-time clipping contour rendering, thus a multi-level FBO technique is used. The advantage of the two pass algorithm is that the rendering routine (first path) is not aware of the existence of the stereo rendering pass. Thus, the second rendering pass can easily be adapted to any stereo/none-stereo mode. Fig. 4.10 shows two examples of stereo pair transmitted to the 3D TV using our FBO-based rendering method.

Input device: In the hybrid user interface, the Hawkeye supports all the interactive operations. We render a virtual pen and use it for interactive segmentation refinement in the VR environment. The position and orientation of the virtual pen is derived from the tracking data of the Hawkeye. In addition, mouse button events are used to trigger interactive segmentation events.

The Hawkeye is also used as navigation tool which allows to translate, and rotate objects and to zoom in/out. Zoom operations are realized by scrolling the mouse wheel. The 6-DOF user interactions are realized by utilizing tracking information and with mouse button events. When the user press the left mouse button, the new position and orientation of the scene objects are calculated:

$$\begin{aligned} T_{new} &= T'_{new}(T'_{old})^{-1}T_{old} \\ R_{new} &= R'_{new}(R'_{old})^{-1}R_{old} \end{aligned}, \tag{4.2}$$

where T and R represent position and orientation of the scene objects, T' and R' represent position and orientation of the input device, X_{old} represents position/orientation at start moment of pressing input device button, X_{new} represents position/orientation after the button pressed.

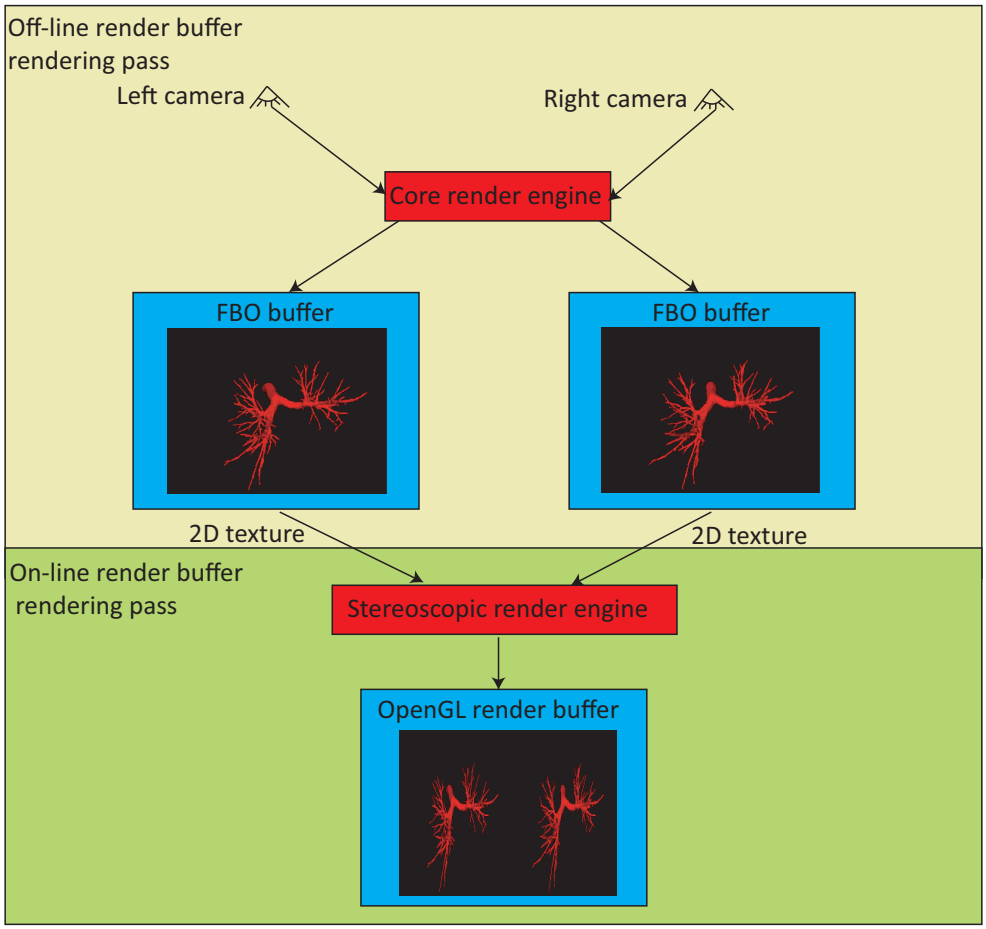
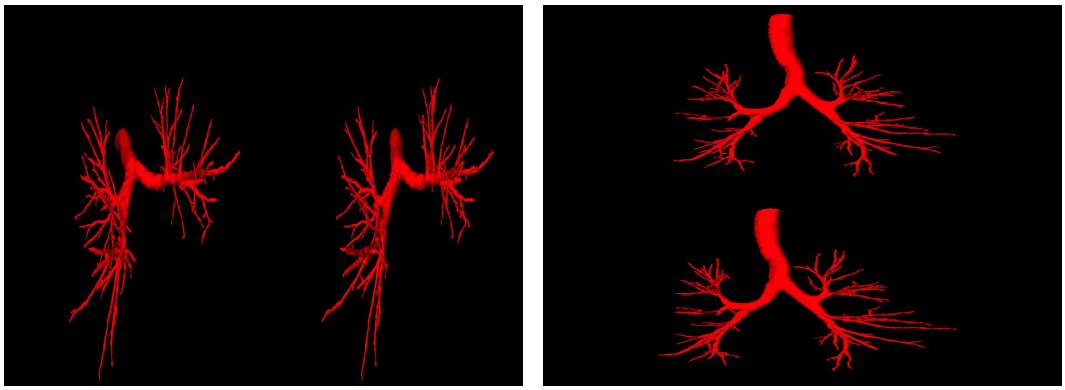


Figure 4.9: Overview of 2-pass stereoscopic rendering algorithm. The first pass (top) is a fixed pipeline and the second pass (bottom) can be adapted to the utilized stereo modes.



(a) (b)

Figure 4.10: Example of binocular images transmitted to the 3D TV. (a) Side by side mode. (b) Top bottom mode.

Context data visualization: For VR-based medical image visualization, iso-surface and volume rendering are frequently utilized. However, for the VR-based segmentation refinement system, accurate display of context information is need to allow the user to find out if a segmentation is correct or not. For instance, 2D multi-planar reconstruction (MPR) technique which allows the extraction of a slice in any position and orientation of a 3D volume. In addition, the segmentation represented by the iso-surfaces should not block the observation of the MPR when the segmentation refinement is done. One way is to remove the portion of the iso-surface in front of the MPR and showing the 2D segmentation contour on the MPR. Since this is done in an interactive environment, real-time rendering (high frame rate) is required. Our system supports visualization of 3D and 4D medical image data. 4D data visualization is realized by displaying 3D context for a selected time phase. The user can change

time phases manually.

The 2D textured plane gives detailed context information, and physician are used to 2D viewers. Thus, we also provide a 2D viewer in our 2D display. 2D views in three directions (axial, sagittal and coronal) are implemented by means of OpenGL 2D texture and Qt.

The loaded data is converted to a 8-bit format scaled by maximum and minimum gray values. The 3D 8-bit data is then copied to the 3D texture memory of the GPU. Our GPU memory is sufficient to hold most of the 3D medical image data. The MPR is realized by a textured polygon. An "infinite plane" is virtually (not visible) set up in 3D space and the user can translate and rotate it by manipulating its normal vector and the distance to the origin of the 3D space. The volumetric data is represented by a bounding "box". In the navigation mode, the user operates the Hawkeye to move the bounding "box", which represents movement of the volume. The vertex positions of the polygon are calculated by intersecting the six faces of the "box" against the "infinite plane". The vertex positions are normalized to the 3D texture coordinates which are utilized in 3D texture interpolation. The standard OpenGL rendering pipeline for the textured plane rendering is replaced by OpenGL Shading Language (GLSL) shader for the following reasons. First, it is easy to combine the segmentation contour visualization on the MPR. Second, it is a real time implementation and allows to change MPR properties such as transparency and gray-value transfer function. Examples of the MPR are shown in Fig. 4.15.

The segmentation result is visualized by rendering the triangle mesh. The

mesh can be generated in two ways. For navigation, the mesh is generated from the binary segmentation volumes by employing marching cube algorithm [78] in an off-line pre-processing step. In this step, the mesh data structure is described in Open Inventor file format. If the interactive segmentation refinement is involved, the mesh vertex positions are derived from the OSF segmentation result. The face sets of the triangle mesh is kept the same during the segmentation refinement. The OSF graph structure and the mesh data structure are shown in Fig. 4.11. Note that multiple surfaces are supported by this data structure. For a given pre-segmentation, the OSF graph data structure is stored on the hard disk. If loaded into our VR software, a copy of the data structure is maintained in the CPU memory. Each time refinement is performed, the costs and mesh vertex positions are updated. There are two copies of the mesh structure (vertex position, normal, color and triangle face sets) for each surface. One is on the CPU side and the other is on the GPU side. The copy on the CPU side is corresponding to the OSF segmentation and input/output operation. The copy on the GPU side is used for rendering. After OSF iteration, the mesh vertices are updated and the normal vectors are recalculated. The corresponding contexts on the GPU side are also updated. CUDA programing is used to modify the vertex properties, for example change the transparency and color, in parallel. The copy on the GPU side is also used for our novel contour rendering algorithm described in the following sub-section. The wireframe rendering mode can be enabled, if the user prefers to see detailed triangle information. In addition, the OpenGL clipping plane is used for the contour rendering on the MPR. All these features can be switched on

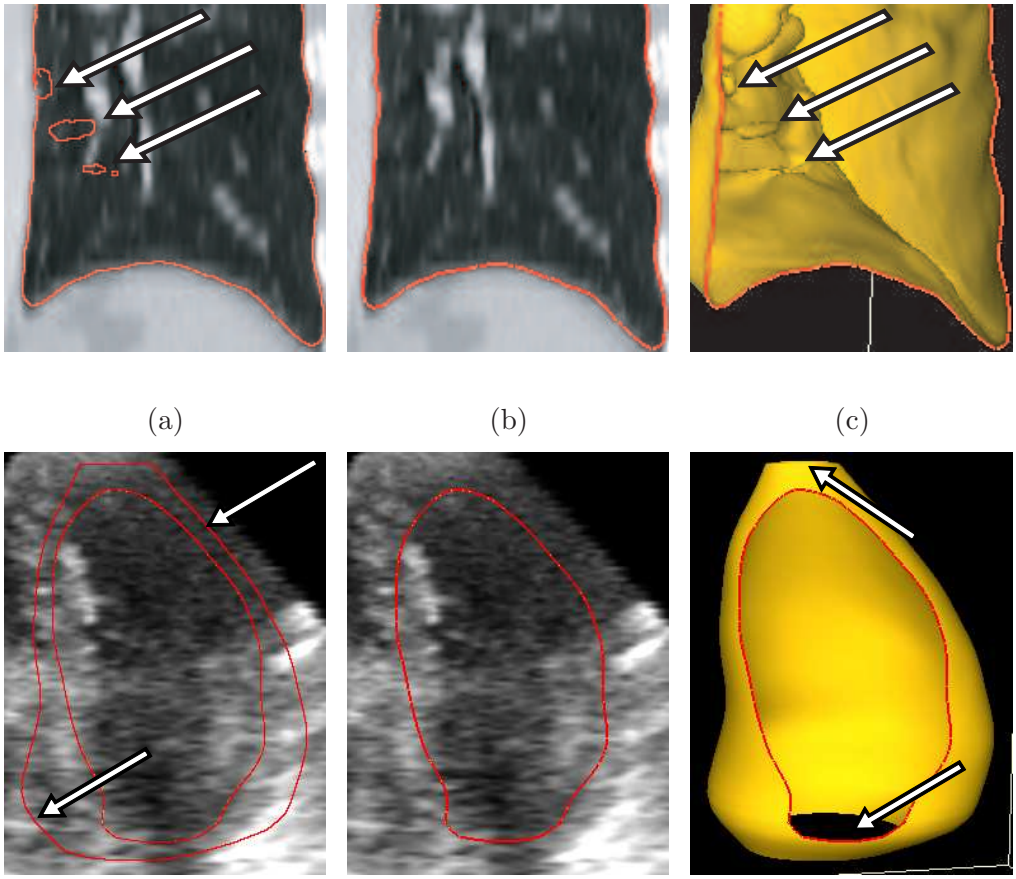
or off on the fly by the user at any time.

Real-time clipping contour rendering: The contour rendering is essential for verifying the segmentation result. However, the mesh surface can consist of many triangles, and MPR location/orientation and the surface are subject to frequent changes. Thus, a fast contour rendering method is required. For this problem, Bornik *et al.* [15] proposed a two-pass image based contour rendering algorithm. First, the algorithm renders the clipped surface in two-sided light mode to the OpenGL Frame Buffer Object (FBO) buffer, which is used to generate an image with a white cross section of the clipped surface on a black background. Second, edge detection is used to extract the boundary. This approach has some disadvantages. If there are holes in the surface (e.g., open surface) or the orientation of some polygons is flipped (e.g., due to mesh folding), artifacts in form of false silhouettes can appear (Fig. 4.12(a) and 4.12(d)). Also, this algorithm is not able to highlight a portion of the contour with a different color, which is required by our refinement method.

To address these problems, a Nvidia CUDA based contour visualization algorithm was developed. In this approach, we assume that each mesh vertex has a label corresponding to a color. In the first pass, the intersection points of the mesh surface and MPR are calculated in parallel. In the CUDA implementation, a thread is generated for a triangle, which delivers zero or two intersection points of the triangle with the MPR plane. The number of threads is equal to the number (N) of triangles. Note that without loss of generality, the case with only one intersection point can be represented with two intersection points. Each intersection point is labeled

```
<OSFGRAPH>

 <surface id="0">
  <vertex id="0" initialPositionId="1" currentPositionId="2">
   <position id="0" pos="0.00 0.00 0.00" cost="0.8">
   <position id="1" pos="0.01 0.01 0.01" cost="0.1">
   <position id="2" pos="0.02 0.02 0.02" cost="0.7">
  </vertex>
  <vertex id="1" initialPositionId="1" currentPositionId="2">
   <position id="0" pos="0.00 0.00 0.00">
   <position id="1" pos="0.01 0.01 0.01">
   <position id="2" pos="0.02 0.02 0.02">
  </vertex>
  <vertex id="2" initialPositionId="1" currentPositionId="2">
   <position id="0" pos="0.00 0.00 0.00">
   <position id="1" pos="0.01 0.01 0.01">
   <position id="2" pos="0.02 0.02 0.02">
  </vertex>
  <face id="0" index="1 2 0"/>
  <face id="1" index="1 0 2"/>
  <face id="2" index="2 1 0"/>
 <surface/>

 <surface id="1">
  ...
 </surface>

 <graph nodeNum="3" edgeNum="3">
  <node id="0" surfaceId="0" vertexId="1" positionId="1" cap_source="1" cap_sink="0"/>
  <node id="1" surfaceId="0" vertexId="1" positionId="1" cap_source="0.01" cap_sink="0"/>
  <node id="2" surfaceId="0" vertexId="1" positionId="1" cap_source="0" cap_sink="0.02"/>
  <edge id="0" node_id="0 1" cap="inf" rev_cap="0"/>
  <edge id="1" node_id="1 2" cap="inf" rev_cap="0"/>
  <edge id="2" node_id="0 2" cap="inf" rev_cap="0"/>
 </graph>

</OSFGRAPH>
```

Figure 4.11: OSF graph structure in XML format used for VR-based segmentation refinement. The graph structure supports multiple surface OSF [75].



(a) (b) (c)

(d) (e) (f)

Figure 4.12: Comparison of the visualization approach presented in [15] (a), (d) and our proposed method (b), (e) . (a), (d) False contours are clearly visible on the cutting plane. (b), (e) Our algorithm produces a correct contour. (c) The contour shown in (b) is combined with the clipped mesh. (f) The contour shown in (e) is combined with the clipped mesh. Arrows in (c) show locations of mesh folding. Arrows in (a) and (d) show locations of false silhouette. Arrows in (f) show locations where the open surface is.

according to the set label of the nearest triangle vertex. Thus, each intersection point is assigned a color according to its set label. The parallel intersection algorithm is described in Algorithm 4.1, where *inter_pts* holds intersection points and *inter_flag* holds the binary values for N polygons indicating whether the polygon intersected (1) or not (0).

Algorithm 4.1 A parallel algorithm of calculating intersection points of cutting plane and the model surface.

1. **input:** plane function, surface vertex points, polygon indices
2. **output:** *inter_pts* and *inter_flag*
3. **for each thread i (each polygon):**
4. *inter_count* := 0 // inter_count is used to record the number of plane and polygon intersection points, in computer graphics it is 0 or 2
5. $\mathbf{v_0}$:= first vertex in the polygon // vertex coordinate (x,y,z)
6. **for** ($\mathbf{v_1}$:= from the second vertex to the last vertex then to the first vertex in i^{th} polygon) **then**
7. d_0 := distance from $\mathbf{v_0}$ to the plane
8. d_1 := distance from $\mathbf{v_1}$ to the plane
9. **if** ($d_0 \neq d_1$ and $d_0 * d_1 \leq 0.0$) **then**
10. $\mathbf{dir} = normalize(\mathbf{v_1} - \mathbf{v_0})$
11. $dotp = \mathbf{dir} \cdot \mathbf{n_{plane}}$ // $\mathbf{n_{plane}}$ is the normal vector of the plane
12. $inter\_pts[i * 2 + inter\_count] := \mathbf{v_0} - \mathbf{dir} * (d_0/dotp)$
13. $inter\_count := inter\_count + 1$
14. **end if**
15. $\mathbf{v_0} := \mathbf{v_1}$
16. **end for**
17. $inter\_flag[i] := (inter\_count == 0\ ?\ 0\ :\ 1)$

Since most of polygons have no intersection and the total number of intersected polygons is not known either, the resulting data structure *inter_pts* on the

GPU cannot be used as rendering buffer. To avoid copying the whole data structure back to CPU, a parallel cuda algorithm is utilized to move intersection points into a continuous buffer new_inter_pts and determine the total number M of intersected polygons. First, a parallel inclusive prefix summation algorithm [86] is applied to the binary array $inter\_flag$, and the resulting array $inter\_scan$ holds N prefix summation results. The inclusive prefix summation is presented in Eq. 4.3.

$$inter\_scan[i] = \begin{cases} inter\_flag[i] & if\ i = 0 \\ inter\_scan[i-1] + inter\_flag[i] & otherwise \end{cases}, \tag{4.3}$$

An example of inclusive prefix summation is shown in Fig. 4.13. The last element of $inter\_scan$ is equal to M. None-zero values in $inter\_scan$ in combination with none-zero elements in $inter\_flag$ are used to indicate the destination locations of intersection points in new_inter_pts. This data moving process is implemented with Algorithm 4.2, which runs in parallel. Finally, pairs of intersection points are drawn as lines using new_inter_pts as a render buffer and rendered into the FBO buffer as 2D texture. The continuous color buffer for rendering contour color is achieved in a similar process as described in Algorithm 4.2.

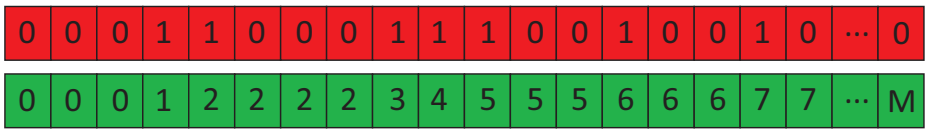


Figure 4.13: An example of inclusive prefix summation from $inter\_flag$ array (the first row in red) to $inter\_scan$ array (the second row in green).

Algorithm 4.2 A parallel algorithm for moving the intersection points to a continuous array.

1. **input:** $inter\_scan$, $inter\_flag$ and $inter\_pts$ arrays
2. **output:** new_inter_pts array
3. **in each thread i:**
4. **if** ($inter\_scan[i] \neq 0$ and $inter\_flag[i] \neq 0$) **then**
5. $index := inter\_scan[i] - 1$
6. **for** ($j = 0; j < 2; j++$) **then**
7. $new\_inter\_pts[2 * index + j] := inter\_pts[2 * i + j]$
8. **end for**
9. **end if**

In the second pass, the 2D texture is superimposed onto the textured plane representing the image context utilizing an OpenGL Shading Language (GLSL) fragment shader program. Thus, the contour is always shown on the textured plane without any occlusion. Note that if the two intersection points were assigned different colors, the color of the line in between will be interpolated accordingly. The overview of context data visualization is depicted in Fig. 4.14.

A comparison of our approach and method in [15] is shown in Fig. 4.12. More visualization examples using our approach are shown in 4.15.

4.3.2 Interactive Generic OSF Refinement

Our segmentation refinement method utilizes the same OSF-based graph structure $G(N, A)$ as utilized for initial segmentation (Section 3.4.4). Note that our refinement method does not change the topology of the underlying graph structure.

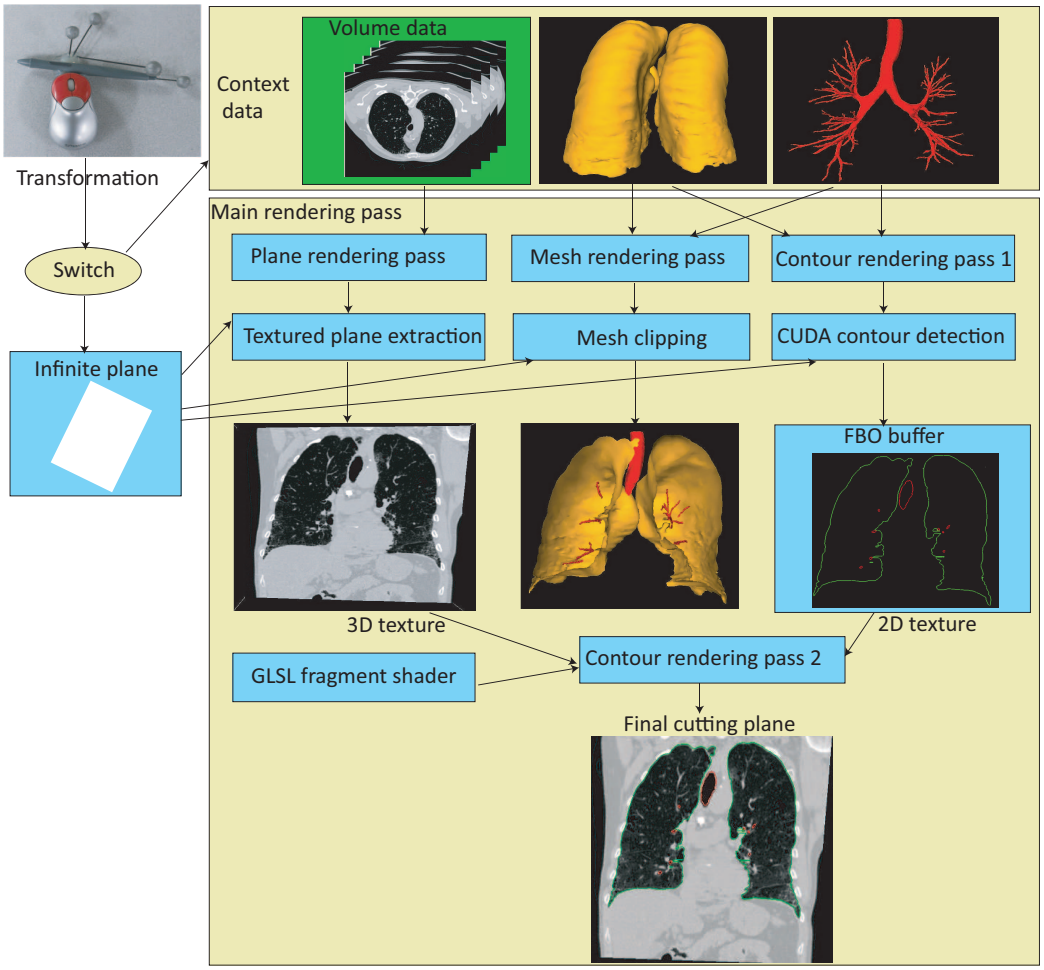
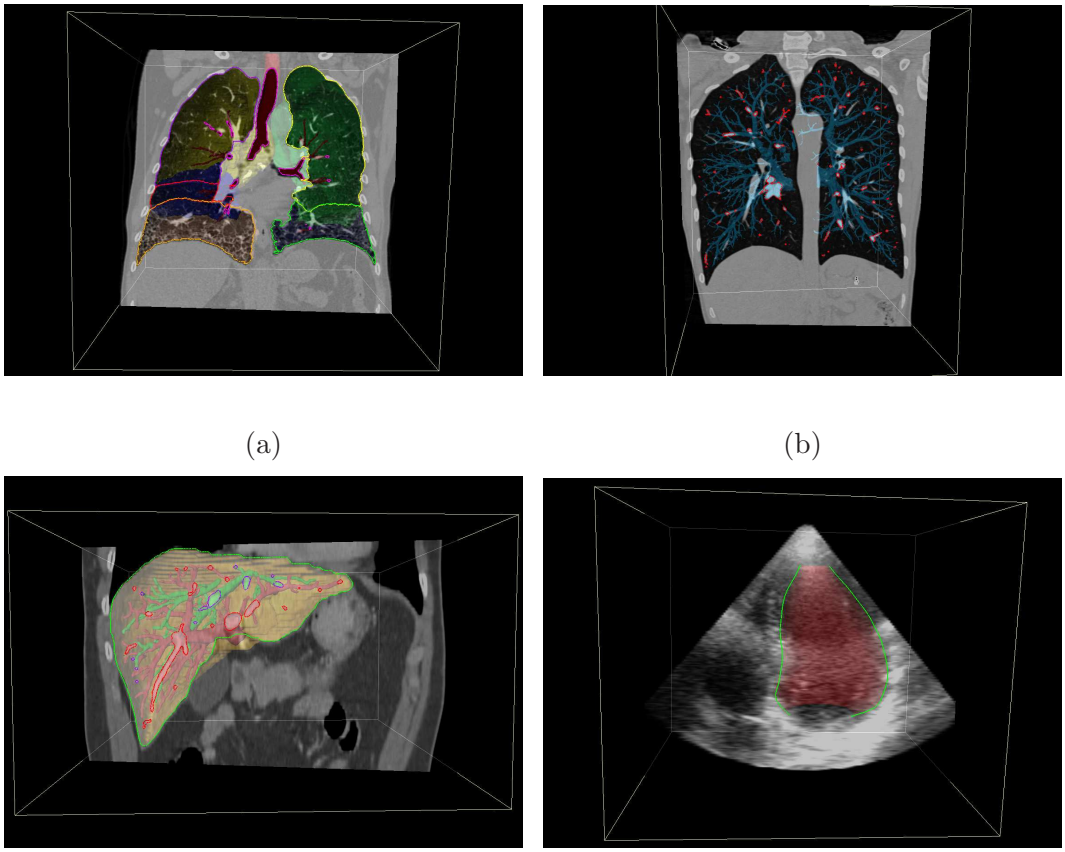


Figure 4.14: Summary of the context data visualization pipeline. Tracking data comes from the Hawkeye. The user selects the transformation for either the cutting plane or the context data. The plane rendering pass extracts an arbitrary texture plane from the volume data based on the "plane" position and orientation. The Mesh rendering pass removes the triangles in front of the plane. The clipped meshes behind the cutting plane can be seen by enabling the plane transparency (Fig. 4.15) or changing the viewpoint. Contours are detected utilizing a CUDA program and rendered to the FBO buffer. The GLSL shader then combines the arbitrary texture plane (3D texture) and the contour from FBO (2D texture).



(a) (b)

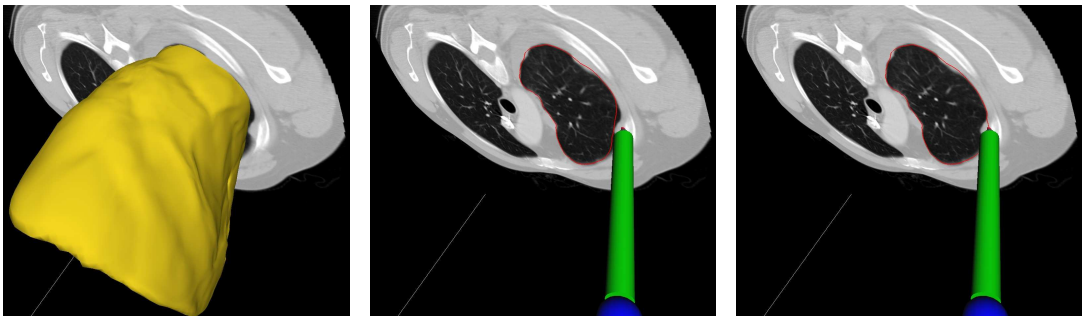
(c) (d)

Figure 4.15: Examples of the contour rendering using the cutting plane algorithm. Alpha blending is enabled for cutting plane rendering. (a) Lung lobes and airway rendering. (b) Lung vessels rendering. (c) Liver and vessels rendering. The transfer function window is adapt to a liver window. (d) Left ventricle in an ultrasound image. In this case, the surface is open.

Simple Example: To reach an intuitive understanding of the underlying interactive OSF refinement framework, a very simple generic refinement tool (named point refinement) is presented (Fig. 4.16). User identifies the segmentation error (Fig. 4.16(a)) and location of the correct boundary (Fig. 4.16(b)) using the hybrid user interface. The nearest graph column $Col(p)$ and node i are identified by the algorithm. The node cost $c(p, i)$ is set to 0 and the rest of cost c on the column $Col(p)$ is set to a large constant value. Thus, it is likely that the updated segmentation result goes through the node i. The maximum-flow algorithm is recalculated to generate an updated segmentation result (Fig. 4.16(c)). Using such an approach can fix segmentation errors but this approach requires lots of user efforts to fix large segmentation errors. More efficient application specific interactive refinement tool will be presented in Chapters 5, 6 and 7.

The generic individual processing steps of the OSF-based refinement algorithm are summarized in detail below.

1. The user inspects the segmentation result and locates a segmentation inaccuracy using our hybrid user interface (Fig. 4.16(a)).
2. The user provides rough clues for the desired locations of incorrectly positioned boundaries/surfaces using refinement tools. This task is supported by our hybrid user interface (Fig. 4.16(b)).
3. Utilizing the information provided by the user, the algorithm locally updates costs in the graph structure G.
4. The maximum-flow is recalculated for the updated graph G. To speed up the



(a) (b) (c)

Figure 4.16: Simple example of interactive OSF segmentation refinement using the hybrid user interface. (a) The user inspects segmentation result using the visualization tool. (b) The user specifies a point indicating the true boundary location. (c) Segmentation result after recalculating maximum-flow.

computation, recomputing the maximum-flow from scratch is avoided by utilizing the previously calculated residual graph in a similar way as described in [16].

5. The display of the new boundaries/surfaces is updated (Fig. 4.16(c)).

The above described refinement method can be utilized iteratively, if required. The program's GUI allows the user to "undo" a refinement operation, if needed. In Chapter 5 (single surface based 3D lung segmentation), 6 (4D lung segmentation) and 7 (dual-surface based IVUS segmentation), details describing step 2, 3 of the refinement algorithm in specific applications will be provided.

Step 4 is explained as below. Suppose the cost of nodes on the column p is changed due to user interaction. The updated node weight $w_{new}(p, i)$ for node i

Table 4.1: Negative new weight assignment for terminal edges of node i on column p.

terminal edge	initial capacity	add	new capacity
$\{s,(p,i)\}$	cap_s	$cap_t - w_{new}$	$cap_s + cap_t - w_{new}$
$\{(p,i),t\}$	cap_t	cap_s	$cap_s + cap_t$

Table 4.2: None-negative new weight assignment for terminal edges of node i on column p.

terminal edge	initial capacity	add	new capacity
$\{s,(p,i)\}$	cap_s	cap_t	$cap_s + cap_t$
$\{(p,i),t\}$	cap_t	$cap_s + w_{new}$	$cap_s + cap_t + w_{new}$

is calculated according to the Eq. 3.2. If $w_{new}(p,i) < 0$, the new $cap_s(p,i)$ should have been $-w_{new}(p,i)$ and new $cap_t(p,i)$ should have been 0. To utilize a similar approach described in [16], the updated capacity assignment is performed according to Table 4.1. Similarly, if $w_{new}(p,i) \geq 0$ (the new $cap_s(p,i)$ should have been 0 and new $cap_t(p,i)$ should have been $w_{new}(p,i)$), the corresponding capacity assignment is done according to Table 4.2. Thus, the new terminal arc capacity assignments are equivalent to the original capacity assignment because the extra constant cap_s+cap_t at both terminal arcs of a column node does not change the minimum s/t cut (minimum-closed set) [16].

CHAPTER 5
INTERACTIVE SEGMENTATION REFINEMENT FOR 3D OSF-BASED LUNG SEGMENTATION

5.1 Introduction

In this chapter, the proposed interactive refinement approach is adapted to 3D lung segmentation and validated. For this purpose, an improved version of automated lung segmentation approach described in Chapter 3 is utilized to produce pre-segmentation, and two specific refinement tools are presented. One is a more generic refinement tool and the other is a specific tool to correct leakage to trachea and main bronchus.

5.2 Methods

5.2.1 Initial Lung Segmentation

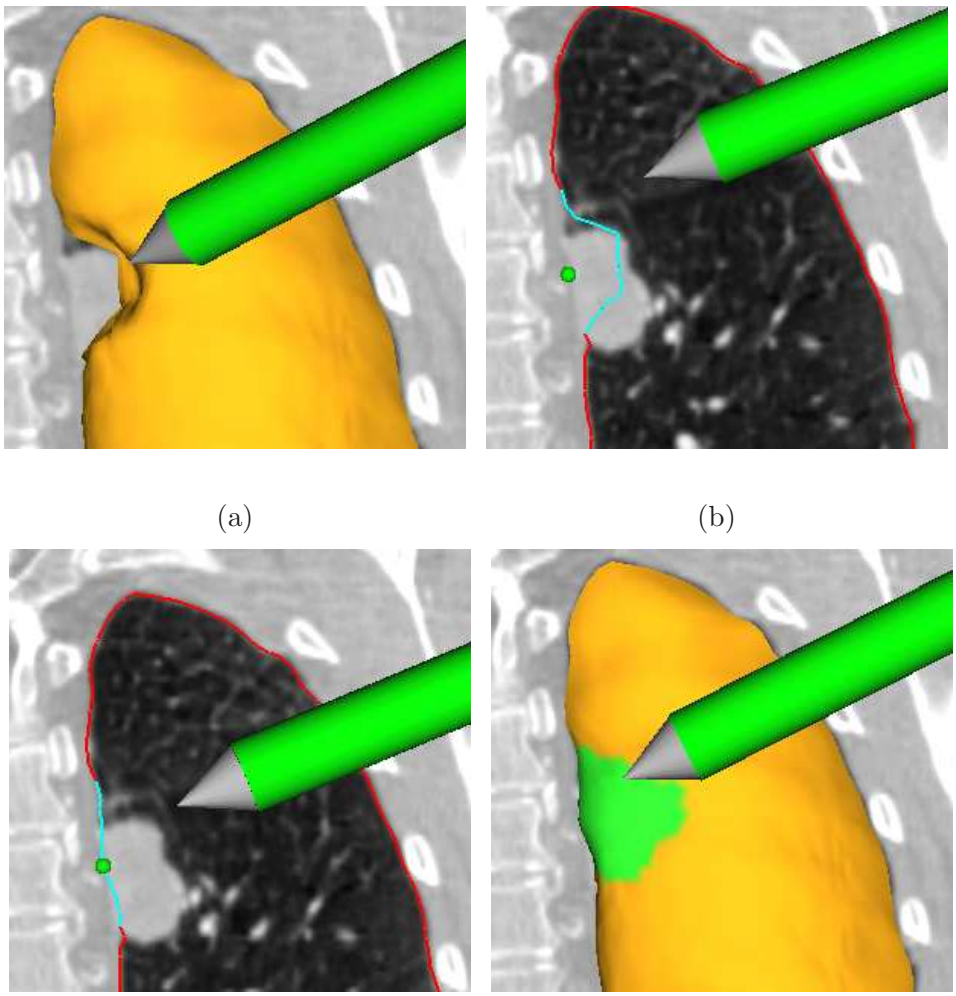
For automated lung segmentation, the approach presented in Chapter 3 was adapted. The lung segmentation uses a robust ASM based segmentation followed by an OSF-based segmentation approach. Chapter 3 utilized a multi-scale OSF approach with a hard smoothness constraint [75] and straight line search profiles normal to the mesh surface. For the method used in lung segmentation refinement, we utilized an improved version of this algorithm, which included a linear soft smoothness constraint (a constant weight α was used to penalize the shift on two adjacent vertices) proposed in [105] and a gradient vector flow based approach to build column profiles [7]. Also, in contrast to Chapter 3, we used a single-scale approach. The graph structure

utilized in this chapter was described in Section 3.4.4. The number of mesh vertices used for the OSF-based segmentation was $10,242$. For the soft and hard smoothness constraints, $\alpha = 0.001$ and $\Delta = 12$ were used, respectively. The search profile length was $l_p = 117$ nodes. Points on the search profile were obtained at discrete sampling positions with a distance of 0.35 mm between them. A Gaussian gradient filter kernel with variance $\sigma = 2.0$ mm was utilized to calculate the cost function.

5.2.2 Generic OSF-based Segmentation Refinement

Basically, the task of 3D lung segmentation refinement can be split into two sub-tasks: (a) identify (label) the local error on the surface and (b) change the cost of columns associated with errors such that the error is corrected or at least reduced when the new optimal surface is calculated for the updated graph. Based on the generic OSF-based interactive refinement framework described in Section 4.3.2, the individual processing steps of the developed segmentation refinement algorithm are as follows.

1. The user inspects the segmentation result and detects an error on the surface by comparing CT data visualized on the cutting plane to the boundary of the segmentation result (Fig. 5.1(a)).
2. The incorrect part of the surface is labeled. For this purpose, the user identifies a point on the true surface location near the error region (Fig. 5.1(b)). During this process, the algorithm displays the estimated incorrect (labeled) portion of the surface interactively, which allows the user to pick a good location for the



(a) (b)

(c) (d)

Figure 5.1: Visualization of generic interactive OSF-based segmentation refinement for a lung with a lung mass adjacent to the lung boundary. (a) The user inspects the lung segmentation and locates a segmentation error. (b) In a cross-section, the user select a point on the correct boundary location with a virtual pen. Note that the incorrect portion of the contour is highlighted in light blue, which was automatically generated based on the selected point. (c) and (d) Refinement result after calculating maximum-flow. (d) The corrected surface region is highlighted in green.

input point.

3. Costs in $G(N, A)$ are locally updated for affected columns.

4. The maximum-flow is recalculated for the graph $G(N, A)$.

5. The new solution (surface) is displayed (Fig. 5.1(c) and 5.1(d)).

In the following, we describe step 2 and step 3 of our refinement algorithm in detail.

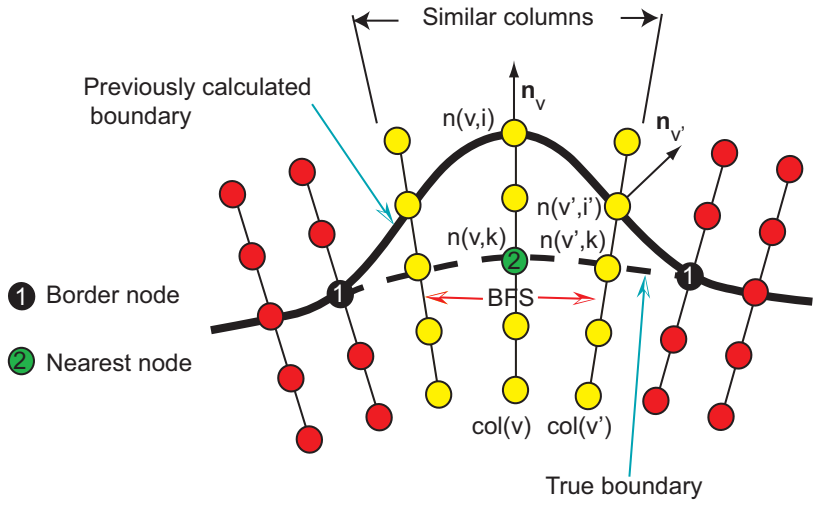


Figure 5.2: Search for similar neighboring columns in the OSF graph structure.

Error region labeling (step 2) — The user specifies a point on the true boundary in an erroneous segmentation region using the hybrid user interface (Fig. 5.1(b)). The algorithm searches for the nearest node on all graph columns based on the specified point. The nearest node $n(v, k)$ is found. Corresponding column v is labeled as the center column, and the center column node $n(v, i)$ on the previously calculated

surface is found (Fig. 5.2). A breadth-first-search (BFS) algorithm [33] is applied to find similar neighboring columns. The neighborhood relation is defined based on mesh topology. The BFS starts from the center column v and examines neighboring columns by utilizing predefined similarity criteria, which are based on three components that are combined by means of a logical *AND* operation:

1. *Surface normal vector* — We assume the angle between the surface normal vector of the center column v and a neighboring column v' to be less than 90°. Thus, we require $\mathbf{n}_v \cdot \mathbf{n}_{v'} > 0$, where $\mathbf{n}$ denotes the surface normal (Fig. 5.2).

2. *Incorrect surface appearance* — This criterion is based on the observation that the incorrect surface passing through the center and the neighboring columns should have similar local image characteristics. The gray-value profile around a node $n(v,i)$ will be denoted as set $P(v,i) = \{g(n(v,i+j))|j \in \{-7,-6,\ldots,7\}\}$, where $g(n)$ represents the gray-value of a node n. Let v' represent a column in the proximity to the center column with node $n(v',i')$ on the previously calculated surface (Fig. 5.2). Then $D(P(v,i),P(v',i')) = \max_{j\in\{-7,-6,\ldots,7\}}\{|g(n(v,i-l_r+j)) - g(n(v',i'+j))|\}$ denotes a gray-value profile similarity function. Columns v and v' are similar if the following criterion is fulfilled:

$$\min_{m\in\{-5,-4,\ldots,5\}}\{D(P(v,i),P(v',i'+m))\} < t_1, \tag{5.1}$$

where t_1 is a threshold. In our lung CT segmentation task, we use $t_1 = 180$ HU (Hounsfield Units).

3. *True boundary appearance* — The basic idea behind this criterion is that the correct surface point(s) on the center column and neighboring columns have similar gray-value appearance. The correct surface point $n(v,k)$ on the center column was selected by the user (Fig. 5.2). However, we need to search for the correct surface points on neighboring columns. For the neighboring column v', we start the search from $n(v',k)$ (Fig. 5.2) with a variable search length

$$l_t = round(5 + 7(1.0 - e^{\frac{-d_{v,v'}^2}{2\sigma^2}})) \tag{5.2}$$

to express increasing uncertainty regarding location of true boundary points with increasing distance from column v. Here, $d_{v,v'}$ denotes the Euclidean distance between previously calculated surface nodes on column v and v'; σ was set to 5 mm. Columns v and v' are similar if

$$\min_{m \in \{-l_t, -l_t+1, \ldots, l_t\}} \{D(P(v,k), P(v', k+m))\} < t_2. \tag{5.3}$$

Threshold t_2 was set to 90 HU. The node with the most similar gray-value profile on column v' is stored in a set V_{sim}.

After the BFS algorithm stops, holes are filled in the labeled surface. Nodes corresponding to the surface patch are stored in the set V_{err}. Fig. 5.1(b) shows an example of a labeled erroneous surface patch visualized on the cutting plane. Parameters for the similarity criteria were selected conservatively to avoid leakage. Thus, the BFS step might in some cases not completely label the incorrect surface patch. To address this issue, the patch can be dilated by the user.

Updating OSF costs (step 3) — Cost function of the center column is updated by

$$c(v,i) = \begin{cases} 10 & if\ i \neq k \\ 0 & i = k \end{cases} . \tag{5.4}$$

Because all costs are initially normalized between 0 and 1, the final segmentation is very likely to pass through the user determined location k on column v. All remaining columns related to the set V_{err} are updated

$$c(v',i) = \begin{cases} c_t(v',i) & if\ v' \notin \Omega \\ c(v',i) & otherwise \end{cases} , \tag{5.5}$$

where Ω is a set containing all center columns selected by the user during the refinement process. Thus, the refinement never changes the cost function of columns already updated using Eq. 5.4 in the previous refinement steps. The term

$$c_t(v',i) = \left(1.0 - 0.5e^{\frac{-(|i-k'|)^2}{2\sigma(v,v')^2}}\right) c(v',i) \tag{5.6}$$

is used to locally adapt the previously utilized costs, where node k' represents an estimate for the true boundary location on column v'. The estimation of k' will be described in the next paragraph. $\sigma(v,v')$ is calculated with $\sigma(v,v') = 5 + e^{\frac{d_{vv'}}{10}}$. The idea behind Eq. 5.6 is that the costs on column v' near to the node $n(v',k')$ have to be low while the impact of the weighting function becomes weaker for nodes on columns that are further away from $n(v',k')$.

The estimate for the true boundary node $n(v',k')$ is calculated as follows. Let V_{border} denote a set of outer border nodes of V_{err} (Fig. 5.2). A Thin Plate Spline (TPS) surface [13] is fit through user selected node $n(v,k)$, nodes in V_{border}, and nodes in Ω_{sim}. To speed-up computation, Ω_{sim} is a randomly selected subset of V_{sim}

with 20 % of the size of V_{sim}. We utilize a TPS interpolation because we assume that the corrected surface has low shape complexity. For TPS interpolation, the kernel function $U(r) = r^2 \log r$ is utilized. The interpolation works in 2.5D while the surface patch consists of 3D points. Thus, an appropriate 2D plane has to be found to apply TPS interpolation. By means of singular value decomposition, a plane is fitted in a least square fashion to nodes in V_{border}. The intersection of the interpolated surface and columns related to set $V_{err} \backslash \{n(v, i)\}$ form estimates for the true boundary.

Note that $n(v, k)$ and nodes in V_{border} are believed to be on the true surface, but nodes corresponding to Ω_{sim} were estimated. Therefore, we enforce the fitted surface to pass through nodes $n(v, k)$ and V_{border} with a regularization parameter value of 0, but do not enforce it through nodes related to Ω_{sim} with a regularization parameter value of 1.0 [38].

Note that parameters used in the described method were determined experimentally on five cases, which were not included in test data sets. An example of generic OSF-based interactive segmentation refinement is given in Fig. 5.1.

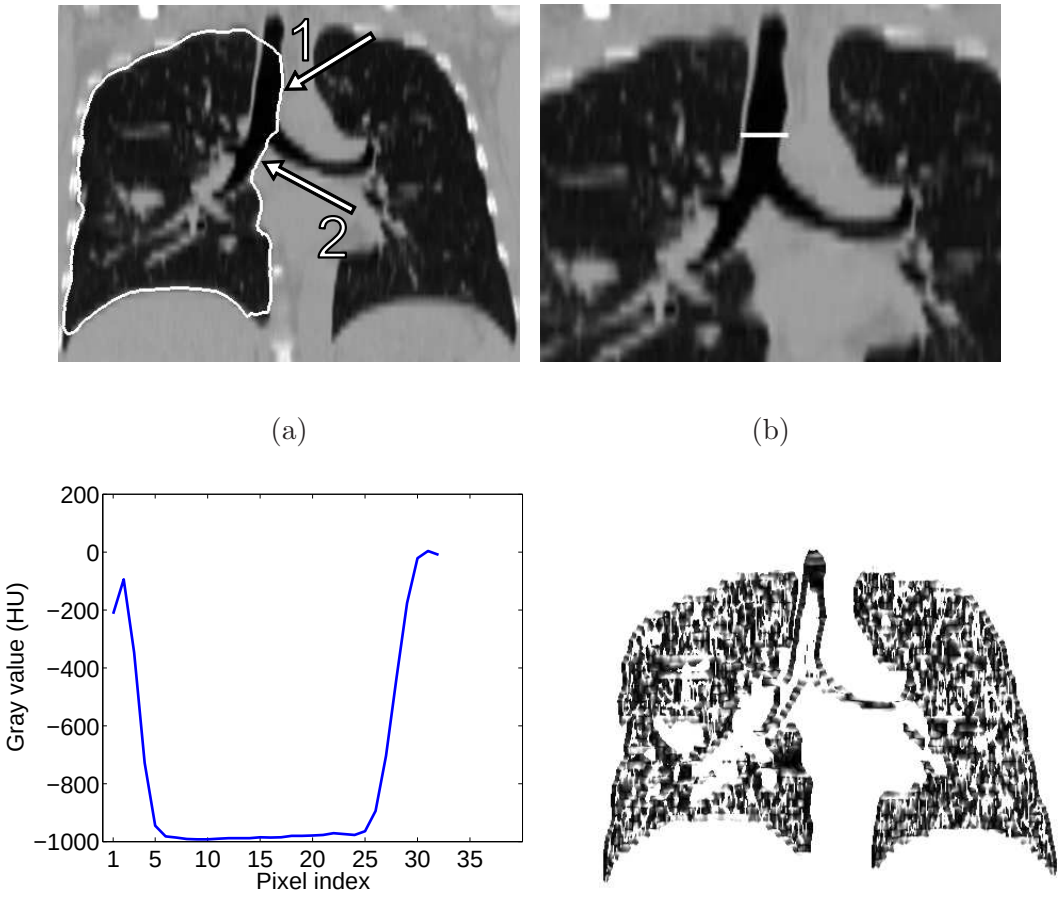
5.2.3 A Specific OSF-based Refinement Method

The tool described in Section 5.2.2 is well suitable to correct lung segmentation errors. However, we observed that when the OSF based lung segmentation leaks to the trachea and main bronchus (Fig. 5.3(a)), the method is not efficient, requiring the user to specify too many refining contour points. To address this issue, we designed a specific tool for this segmentation problem. This specific method also consists of

5 steps as described in Section 5.2.2, but step 2 (error region labeling) and step 3 (updating OSF costs) are different from the generic approach. In the next sections, these steps are explained in detail.

Error region labeling (step 2) — The basic idea is to utilize the gray-value and gradient characteristics of CT image data to identify/label surface points corresponding to the leak to trachea and the main bronchus. The center column node $n(v,k)$ (nearest node in Fig. 5.4) is found based on the manually selected point on the true lung boundary in the area of the leak. The BFS algorithm is utilized to identify the incorrect nodes and is based on two properties:

1. *Gray-value properties* — A large density difference between air-filled airway lumen and surrounding tissue can be observed. Fig. 5.3(c) shows a typical gray-value profile passing through the airway lumen and surrounding tissue corresponding to the profile shown in Fig. 5.3(b). Columns involved in the leakage to trachea and main bronchus pass through the airway lumen as illustrated in Fig. 5.4. Thus, for the neighboring column v', nodes starting from $n(v',i)$ to $n(v',0)$ are searched (Fig. 5.4) and their average gray-values $g_a(v,i) = 1/7 \sum_{j=-3}^{3} g(n(v,(i+j))$ are examined. The averaging is used to reduce the influence of noise. At first, the search looks for a node $n(v',j)$ with $g_a(v',j) < -900$ HU (air). Once such a node is found, the search continues on the column until the first node $n(v',k)$ with $g_a(v',k) > -600$ HU is found. The node $n(v',k)$ will be close to the inner airway boundary (Fig. 5.4). The search is refined by identifying the maximal gradient magnitude location about node



(a) (b)

(c) (d)

Figure 5.3: Example of incorrect (leakage to trachea and main bronchus) lung segmentation and corresponding image features utilized for error identification. (a) Lung segmentation leaking to trachea (arrow 1) and main bronchus (arrow 2). (b) Profile location and (c) corresponding gray-value profile passing through airway lumen and surrounding tissues. (d) Corresponding $\mathbf{z} \cdot \mathbf{g}_{dir}$ volume.

$n(v', k)$ in a search range of ± 3 nodes, resulting in node $n(v', m)$. Gradient magnitude g_{mag} and direction $\mathbf{g}_{dir}$ are pre-calculated for each voxel in the volume. Prior to gradient calculation, the gray-value range is truncated to -1000 and -700 HU, to reduce the effect of unrelated structures on the gradient. The gradient is calculated by utilizing Gaussian derivatives with a standard deviation $\sigma_g = 0.5\ mm$. In addition, to reduce the influence of noise, gradient magnitudes less than 10 are ignored. The same search is performed for the center column, but it takes place between nodes $n(v, i)$ and $n(v, k)$.

2. *Gradient properties* — Trachea and main bronchus are elongated tubular structures along the z-axis of the volume. Thus, the z-axis is approximately perpendicular to the (normalized) gradient direction of each voxel on the airway boundary. This constellation can be described by the dot product $\mathbf{z} \cdot \mathbf{g}_{dir}$, where $\mathbf{z} = (0, 0, 1)^T$ represents the z-axis direction and $\mathbf{g}_{dir} = (g_x, g_y, g_z)^T$ is the normalized gradient direction of each voxel. Fig. 5.3(d) shows a typical example of a $\mathbf{z} \cdot \mathbf{g}_{dir}$ volume.

The criterion is fulfilled if a node $n(v', m)$ can be found for a neighboring column with $\mathbf{z} \cdot \mathbf{g}_{dir}(n(v', m)) < t_3$ with the angle threshold $t_3 = 0.4$. After the BFS algorithm stops, nodes corresponding to the surface patch are stored in the set V_{err}, and holes in the patch are closed. In addition, nodes on the airway wall $(n(v', m))$ are stored in the set V_{sim}. For columns related to holes in the surface patch, the location of the node $n(v', m)$ is estimated by interpolation, and the result is added to V_{sim}.

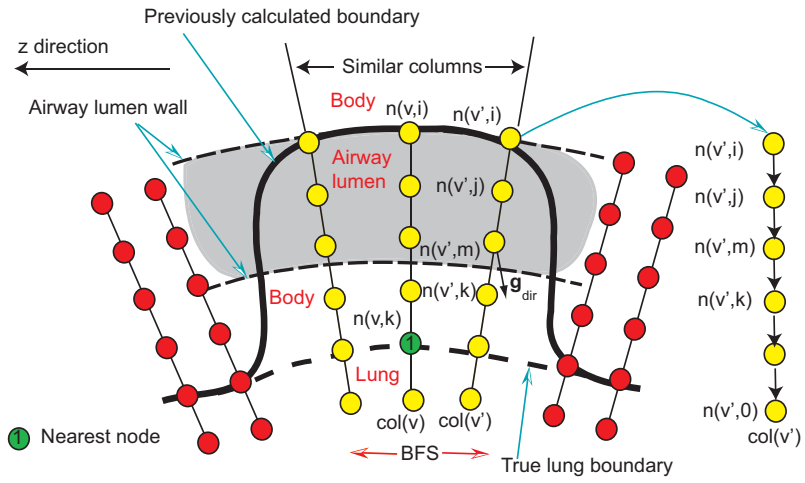


Figure 5.4: Labeling surface points corresponding to the leakage to trachea and main bronchus based on BFS.

Updating OSF costs (step 3) — The cost function of the center column is updated according to Eq. 5.4. All other columns corresponding to nodes in V_{err} are updated by Eq. (5.5) as

$$c_t(v', i) = \begin{cases} 1 & if\ i > m \\ c_0(v', i) & otherwise \end{cases}, \tag{5.7}$$

with $n(v', m) \in V_{sim}$. $c_0(v', i)$ is the initial cost function utilized before refinement. Thus, nodes inside the airway lumen receive a cost of one.

5.3 Evaluation Methodology

5.3.1 Image Data

For this study, 18 multidetector computed tomography (MDCT) thorax scans of patients with lung tumors were selected from a larger pool of data sets such that at least the left or right lung required segmentation refinement after automated segmen-

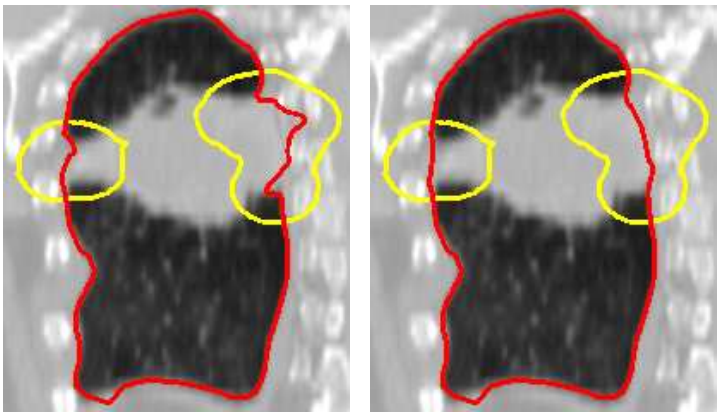
tation. In the 18 MDCT scans, 21 left/right lungs were found to require segmentation refinement. MDCT images where acquired with different scanners and imaging protocols. The image sizes varied from $512 \times 512 \times 415$ to $512 \times 512 \times 642$ voxels. The slice thickness of images ranged from 0.6 to 0.7 mm and the in-plane resolution from 0.60×0.60 to 0.79×0.79 mm. None of the test data sets has been used for the development of algorithms.

5.3.2 Independent Reference Standard

For all tested data sets, an independent reference standard was generated by utilizing a commercial lung image analysis software package PW2 (VIDA Diagnostics Inc., Coralville, IA). First, an automated lung segmentation was performed. Second, since the software was not designed to deal with lungs containing large lung cancer regions, an expert inspected all the segmentations slice-by-slice and corrected all segmentation errors manually. In the case of diseased lungs, this process took several hours per lung.

5.3.3 Identification of ROIs for Performance Analysis

To assess segmentation refinement performance, 30 volumetric region of interests (ROI) were identified in the image data that contain major local segmentation errors. These ROIs were also utilized to indicate which region should be refined by the user. Fig. 5.5 shows an example of such an ROI. Table 5.1 summarizes segmentation error types included in the defined ROIs. Note that there can be one or more types of segmentation errors in a single ROI. An expert was asked to refine the segmentation



(a) (b)

Figure 5.5: A comparison of segmentation before (a) and after (b) refinement. The segmentation is highlighted in red and the ROI region is shown in yellow. (a) Initial automated OSF segmentation. (b) Corresponding 3D segmentation refinement result.

errors inside the ROIs.

5.3.4 Quantitative Indices

The utilized ROIs were defined such that the segmentation errors can be corrected by manipulating the surface portion inside the ROIs. However, the user might unintentionally affect a portion of the surface in close proximity to the ROI boundary or the global optimal OSF calculation might cause changes outside of the ROI. Therefore, we evaluate refinement performance inside and outside the ROIs.

1. *Validation inside ROIs* — The following quantitative error indices were utilized: mean absolute surface distance (d_a) [45] and mean signed border positioning errors (d_s) [107]. A negative value for d_s indicates that the segmentation boundary

Table 5.1: Summary of segmentation error types included in the predefined ROIs.[†]

Error location	Frequency
cancer region	12
leak to high contrast region (e.g., contrast agent)	12
leak to airway branch & hilar region	5
leak to trachea and main bronchi	2

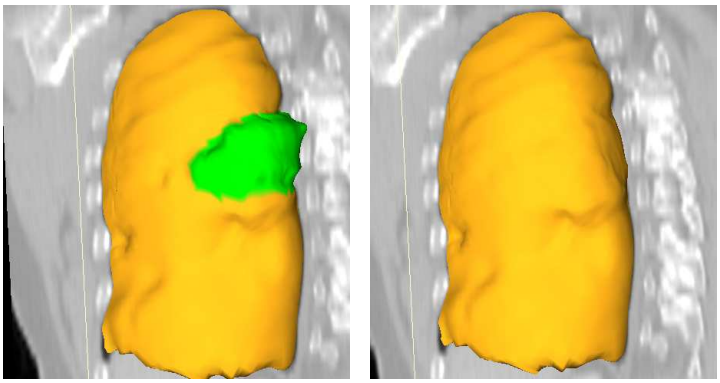
[†] Note that only major segmentation problems in ROIs were taken into account.

is inside the reference object and a positive value indicates that the boundary is outside the reference object.

2. *Validation outside ROIs* — The Euclidean distance (d_e) of two corresponding surface vertices before and after refinement are utilized to measure vertex displacement. In addition, for each refinement-modified vertex outside the ROI, a shortest geodesic distance (d_g) to the ROI boundary is utilized to measure proximity to the ROI. To calculate d_g, a weighted undirected graph G_{d_g} is constructed from the triangle mesh. The arc weight is based on the Euclidean distance between two triangle vertices. d_g is calculated based on Dijkstra shortest path algorithm [33].

5.4 Results

The mean and standard deviation of user interaction times needed for segmentation refinement per ROI was 1.9 ± 1.2 min with a median of 1.6 min. User interaction times ranged between 0.4 and 4.5 min. On average 5 ± 3.4 (median: 4) manually defined surface-correcting points were required, and the minimum and



(a) (b)

Figure 5.6: A comparison of automated segmentation and segmentation refinement in mesh representation for the case shown in Fig. 5.5. (a) Initial automated OSF segmentation. The region marked green indicates segmentation error due to a large lung cancer region. (b) Corresponding 3D segmentation refinement result.

maximum were 1 and 13 points, respectively.

A plot of required interaction in dependence of surface area inside the ROI is depicted in Fig. 5.7. The actual computing time required by the algorithm was 150 ± 152 ms (median: 101 ms) with minimum and maximum computing time of 73 and 1220 ms, respectively.

5.4.1 Results inside ROIs

The mean absolute surface distance and mean signed border positioning errors before and after refinement measured inside ROIs for all thirty test cases are shown in Figs. 5.8 and 5.9, respectively. A Student's t-test at a significance level of 0.05 was performed to determine whether the average error indices after refinement were

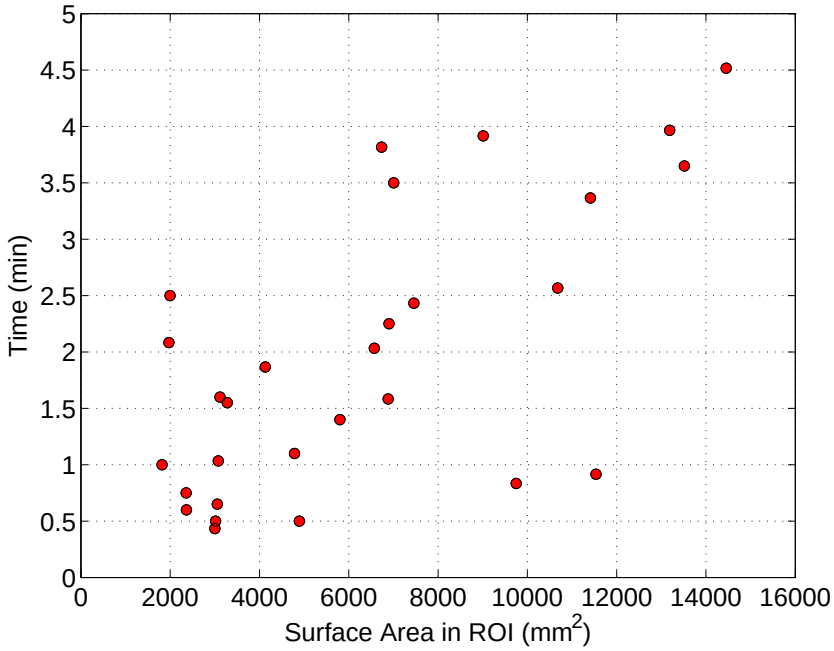
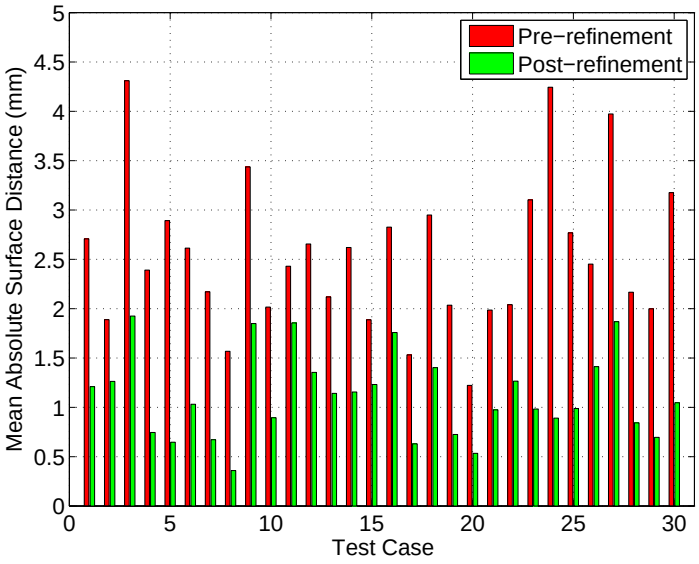
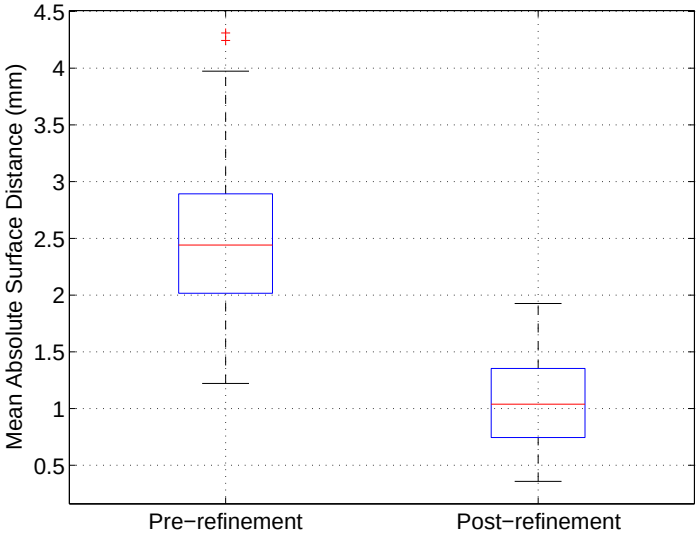


Figure 5.7: User interaction times required for each refinement task in dependence of surface size inside the ROI. Note that conventional approach to correcting severe segmentation failures using slice-by-slice contour editing tools typically require tens of minutes and up to several hours per image dataset.

significantly different than prior to the refinement. Both indices, the mean absolute surface distance ($p << 0.001$) and mean signed border positioning error ($p = 0.02$), were significantly improved. The mean absolute surface distance errors prior to the refinement were 2.54 ± 0.75 mm (median: 2.44 mm) and the same errors decreased to 1.11 ± 0.43 mm (median: 1.04 mm) after the refinement. Fig. 5.8b shows boxplots graphically demonstrating improvements in the surface distance errors after refinement.



(a)



(b)

Figure 5.8: Mean absolute surface distance error before and after segmentation refinement measured inside the ROIs. (a) A case by case comparison. (b) A boxplot comparison summarizing performance in all tested cases.

Figure 5.9: Mean signed border positioning errors before and after segmentation refinement measured inside the ROIs.

Examples of segmentations before and after refinement are depicted in Figs. 5.5, 5.6, and 5.10. In the case of Fig. 5.10, the independent reference standard is also shown for comparison.

5.4.2 Results outside ROIs

The impact of segmentation refinement on the segmentation outside the ROIs is summarized in Fig. 5.11, which shows a plot of the number of altered vertices as a function of the mean number of triangle edges on the ROI boundary. For each test case, boxplots for the displacement of nodes outside the ROIs after refinement are shown in Fig. 5.12. Combined over all 30 cases, the average node displacement was 0.56 ± 0.38 mm (median: 0.57 mm) with a range of 0 to 1.34 mm.

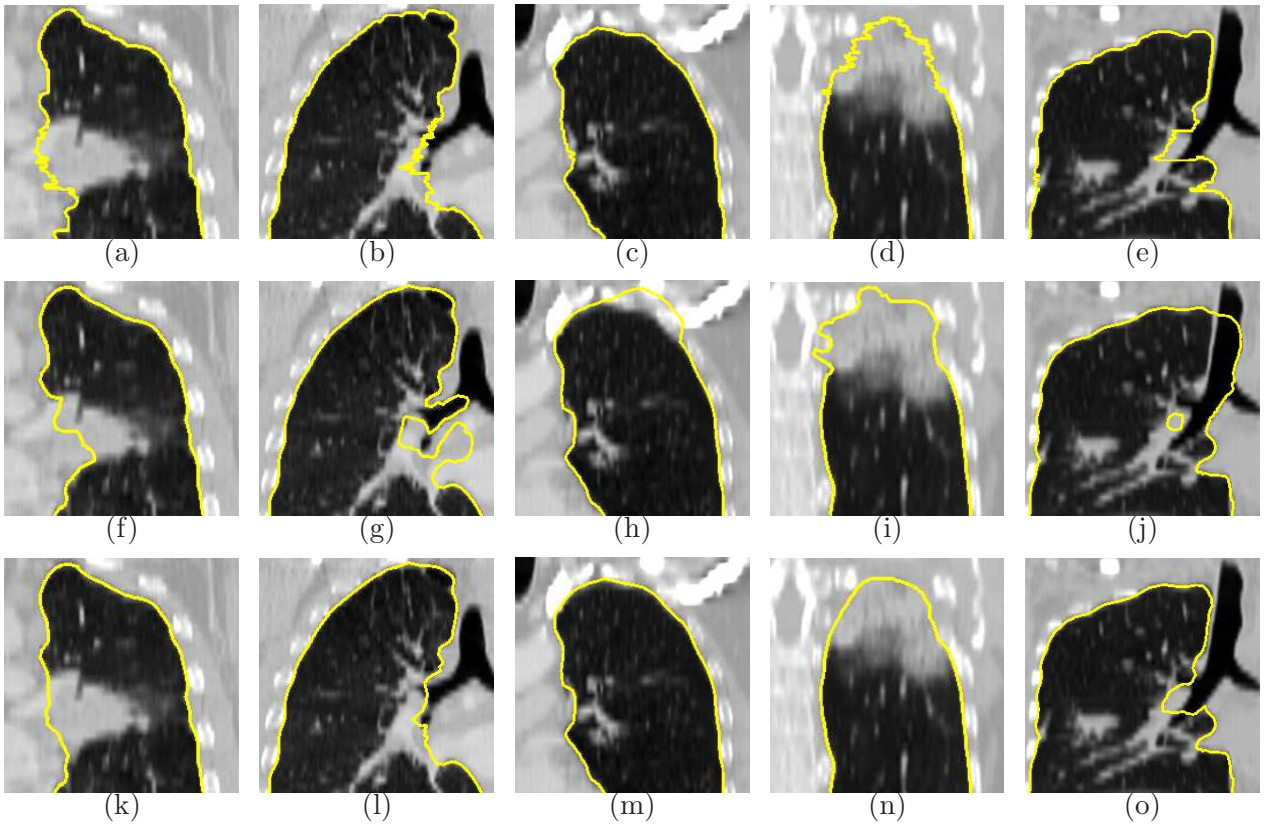


Figure 5.10: Examples of segmentation results on five different data sets (columns). (a)-(e) Independent reference standard. Note that a zigzag pattern of the reference boundary can be observed on both the sagittal or coronal views, because the manual expert segmentation was performed in a slice-by-slice fashion, which typically leads to slice-by-slice inconsistencies. (f)-(j) Initial automated OSF segmentation results. (k)-(o) Segmentation after 3D refinement.

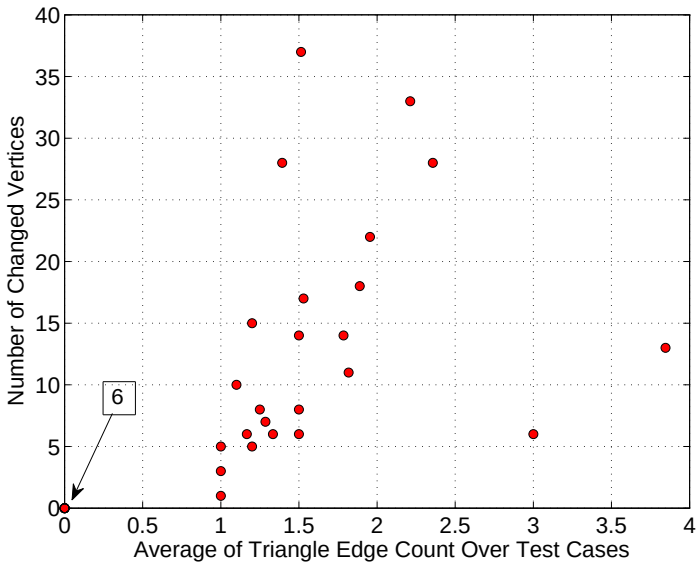


Figure 5.11: Dependence between number of altered vertices before and after refinement outside the ROI and average triangle edge count on the geodesic shortest path from altered vertices to the ROI boundary. Note that six test cases are located at the origin of the coordinate system, as indicated by the arrow.

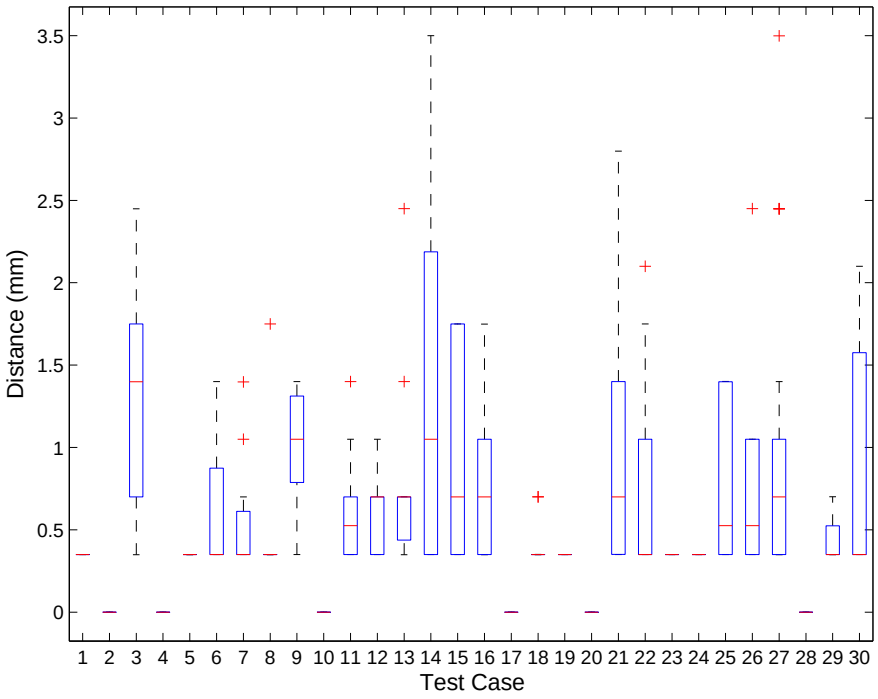


Figure 5.12: Boxplot of the displacement distance of vertices outside the ROI that were altered during refinement. Note that for the six cases without displacement a red line at zero is shown.

5.5 Discussion

The validation showed that the developed method allowed the user to successfully reduce/correct segmentation inaccuracies in all cases of our testing data set, which consisted of 18 datasets for which the automated segmentation approach locally failed (in 30 ROIs) and interactive surface refinement was necessary (Fig.5.8 and 5.9). Quantitative assessment of the achieved segmentation improvements demonstrated that the improvements are statistically significant.

Our 3D refinement results (Fig. 5.10) show that the refined surfaces exhibit smooth boundaries, an additional benefit of our approach compared to slice-by-slice manual editing of 2-D contours. For example, Fig. 5.10 shows zig-zag line/surface inconsistencies across slices. Additionally, it is difficult to consistently define the lung boundary in the area near the hilum where vessels and airways enter/leave the lungs and no generally accepted standard for segmentation exist. In this context, it is interesting to note that in four out of the five test cases, for which the mean absolute surface distance after refinement exceeded 1.5 mm, required refinement in the hilum region, further stressing the difficulty of expert-determination of proper surfaces in this location of pulmonary anatomy.

Because the OSF approach delivers a globally optimal solution, a local manipulation of a cost function could potentially lead to an alteration of the solution (surface) outside of the local area within which the cost function was purposely modified (i.e., surface detection changes may theoretically appear outside of the refinement ROI). The performed assessment of such change outside the predefined ROIs indi-

cates that in our 18 datasets, only minor changes occurred in close proximity to the ROI region and no changes were detected in the remaining parts of the lung surface. This result suggests that the influence of the modifications remains local in practice and the global modifications possibility does not form a problem in the image segmentation refinement application.

The required user interaction time was in the low single-minute range. The plot shown in Fig. 5.7 suggests that the refinement times of more than 2.5 mins. were only required for inaccuracies affecting relatively large portions of the lung surface. Manual editing of the segmentation error in a slice-by-slice fashion would take much longer, because manipulating a surface is more efficient than editing 2D contours in a cross-sectional images.

The average computing time of 150 ms per refinement iteration demonstrates that our algorithm is well suited for real-time interactive use. The maximum computing time of $1,220$ ms or just little over one second was recorded for a case involving a large surface region and was mainly required for surface interpolation. To further increase the responsiveness of the interactive refinement environment, the interpolation steps could be implemented using CUDA, which would provide the user with a truly real-time interactive feeling when using the surface refinement environment.

In our CT lung surface segmentation refinement application, the two refinement tools—one generic and one specifically designed for leaks to trachea/main bronchus—were sufficient to handle the full range of frequently occurring lung segmentation inaccuracies. However, depending on the application (e.g., type of the lung

disease, the utilized imaging protocol, etc.), adaptation of the existing tools and/or development of new tools may provide further benefits.

CHAPTER 6
4D OSF-BASED LUNG SEGMENTATION WITH INTERACTIVE SEGMENTATION REFINEMENT

6.1 Introduction

In this chapter, the proposed interactive refinement framework using the hybrid user interface is investigated in the context of 4D lung segmentation. First, a novel 4D lung segmentation framework is utilized to produce a simultaneous OSF-based segmentation of lungs imaged at inspiration and expiration. Second, the two refinement tools described in Chapter 5 are integrated in a 4D segmentation refinement framework. Finally, assessment of performance is provided. Note that the main goal of the work described in this Chapter is to assess the feasibility of 4D segmentation refinement.

6.2 Method

In this work, we assume that the 4D lung CT data consists of volumetric scans at inspiration and expiration. If needed, the presented approach can be easily adapted to other combinations of respiratory states. The approach consists of two main stages: a) initial automated segmentation and b) interactive 4D refinement, if needed. Fig. 6.1 provides an overview of the approach to 4D lung segmentation. Core components of the automated method are: RASM-based segmentation (Chapters 2 and 3), image registration, and 4D OSF-based segmentation. For inspection and OSF-based refinement, an adapted version of the VR environment introduced in Chapter 4

will be utilized, which allows the user to visualize either the expiration or inspiration CT scan.

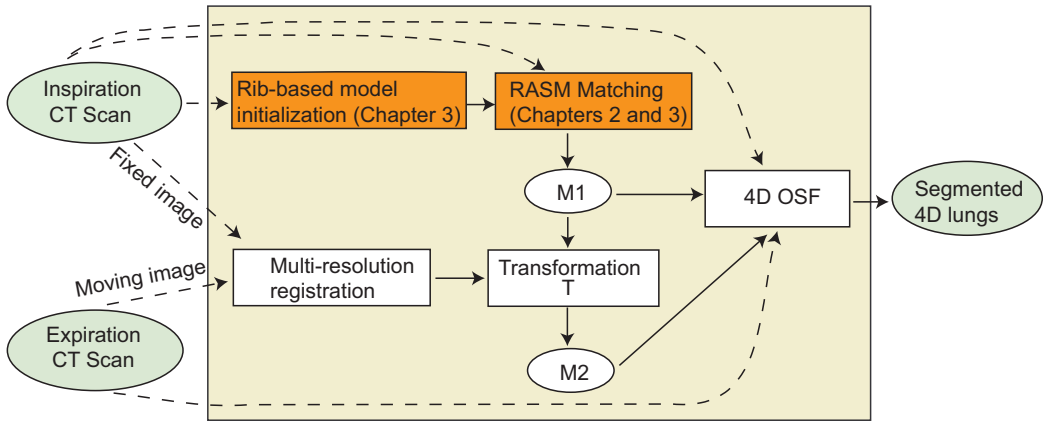
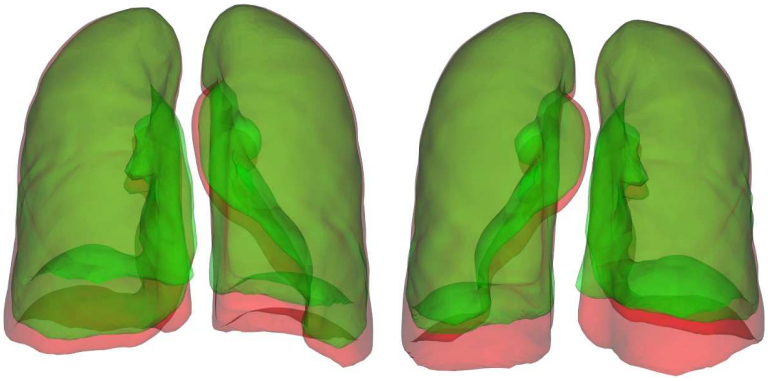


Figure 6.1: Overview of 4D lung segmentation approach. Methods that are shown in orange were discussed in previous chapters.

6.2.1 Automated Segmentation

In this work, a 4D OSF-based approach is utilized to produce left and right lung segmentations, respectively. The RASM approach with a rib cage detection approach is used to initially segment left and right lung in the inspiration CT scan as described in Chapter 3. The resulting mesh is denoted as "M1". To construct a 4D graph for OSF-based segmentation, an initial segmentation of the expiration scan of the 4D data set is needed. As depicted in Fig. 6.1, the expiration scan is registered to the inspiration scan using a multi-resolution registration approach which produces a deformation (transformation) field T (from fixed image to moving image) as a result.

Afterwards, M1 is deformed to $M2 = T(M1)$, where M2 serves as an initial lung segmentation result of the expiration scan. Finally, 4D OSF graph is constructed to segment the 4D target structure (left or right lung) simultaneously using mesh "M1" and "M2" as initial shapes. The approach to registration and 4D OSF-based segmentation is described below in detail.



(a) (b)

Figure 6.2: Examples of inspiration (M1) and expiration (M2) meshes. The inspiration mesh is shown in red and the expiration mesh in green. (a) Anterior-posterior view. (b) Posterior-anterior view.

Image Registration: In our work, 3D image registration was implemented using the registration package **elastix** developed by Klein *et al.* [64]. Specifically, the lung registration approach proposed by Staring *et al.* [109] was utilized for registering the two lung scans within **elastix** framework. More details about the lung registration

algorithm can be found in [109] and on the elastix webpage[1]. Note that the lung mask was not used as proposed in [109]. Fig. 6.2 shows one example for inspiration mesh M1 and transformed mesh M2.

4D Graph-based Segmentation: For 4D OSF-based segmentation, a weighted graph G containing multiple sub-graphs $G = \{G_i = (V_i, A_i) : i = 1, 2, ...N_p\}$ is constructed, where V_i represents a graph node set, A_i a graph arc set, i time index related to individual volumetric data set of the 4D image data set. In this case N_p equals two, and S_i represents a surface corresponding to sub-graph G_i.

Sub-graph $G_i(V, A)$ is built from an initial mesh surface M_i (shape and topology prior), which is assumed to be close to the target surface. A graph column with l_p is generated along the search profile for each surface. Each node corresponds to a point in the volume with intensity $I_i(x, y, z)$ in the volumetric image I_i. The node density d_n on the profile is adjusted to the given image resolution. An example of sub-graphs is shown in Fig. 6.3(a). Intra-column arcs are built to connect nodes $n(v, k)$ to $n(v, k-1)$ on a column $col(v)$ with infinity weights, where k is the column node index. Column $col(v_i)$ and $col(v_j)$ are adjacent columns, if vertices v_i and v_j are on the same triangle edge. For adjacent columns, inter-column arcs are built to connect the node $n(v_i, k)$ to the node $n(v_j, k - \Delta)$ with infinity weights (Fig. 6.3(b)). To achieve a 4D graph-based segmentation, sub-graphs G_1 and G_2 are interconnected by inter-phase arcs which connect nodes $n(v_i, k)$ of G_2 and $n(v_i, k - \delta^u)$ of G_1 with infinity weights, and nodes $n(v_i, k)$ of G_1 and $n(v_i, k - \delta^l)$ of G_2 with infinity weights

[1]http://elastix.bigr.nl/wiki/index.php/Par0011

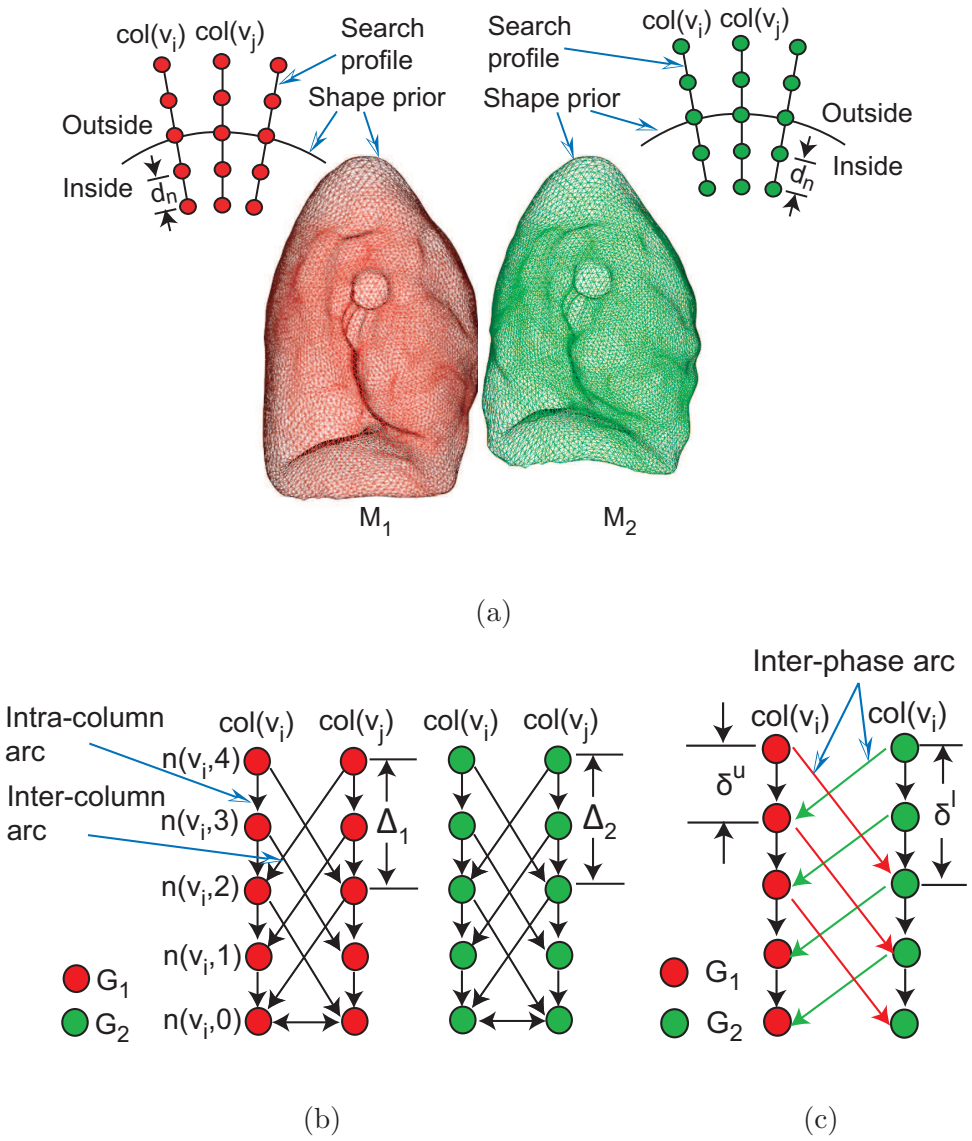


Figure 6.3: Graph construction for 4D OSF. (a) Search profiles are constructed starting from shape priors (meshes M1 and M2). (b) OSF graph structures (arcs) enforcing the surface smoothness constraints. (c) OSF graph structure (arcs) enforcing the inter-phase constraints.

(Fig. 6.3(c)). δ^l and δ^u are inter-phase motion constraints representing the allowed maximum distances between surfaces S_1 and S_2.

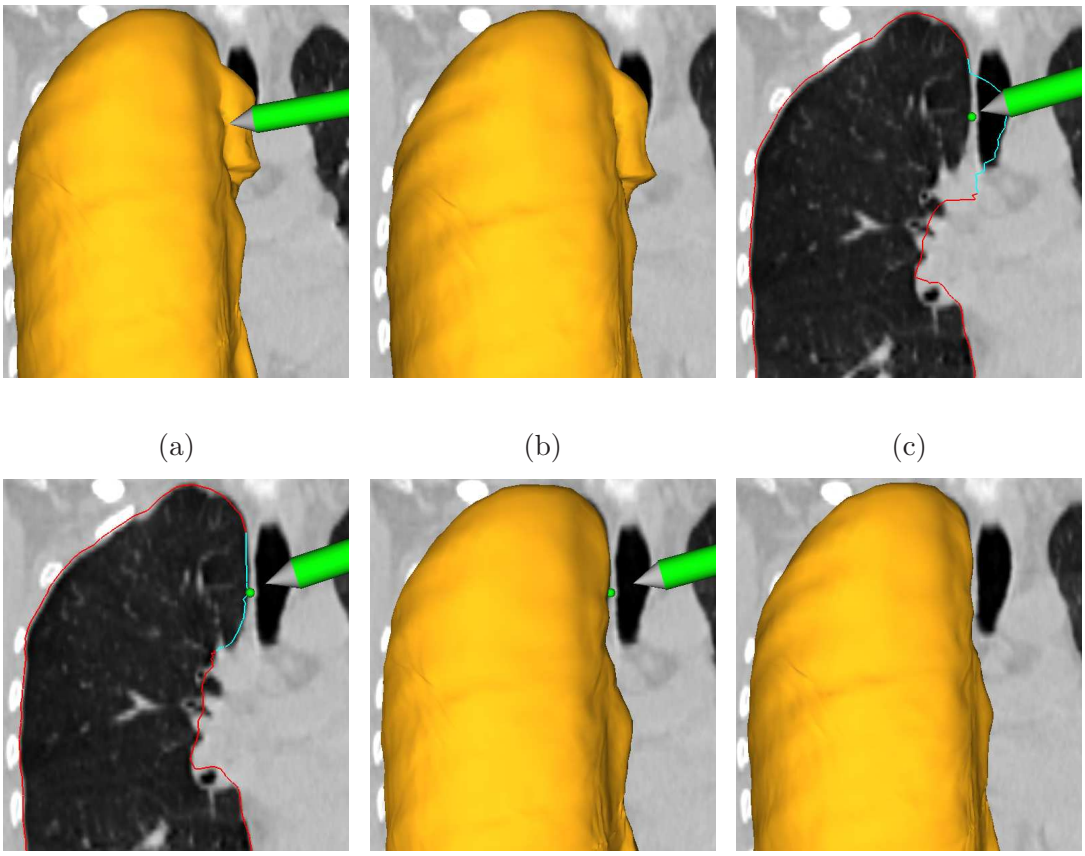
Graph node weight sets C_i with $i \in \{1, 2\}$ (objective functions) are derived from corresponding volumetric image data I_i to reflect local image characteristics. C_1 and C_2 are calculated using Eq. 3.3. In addition, linear soft smoothness constraints are utilized, as proposed by Song et al. in [105]. For this purpose, constant weight α arcs are introduced to penalize shifts between adjacent vertices on surface S_i. A gradient vector flow based approach to build column profiles [7] was utilized.

The number of mesh vertices (for M1 and M2) used for the OSF-based segmentation was $10,242$. For the soft and hard smoothness constraints, $\alpha = 0.001$ and $\Delta = 12$ were used, respectively. For the inter-phase constraints, the following parameters were utilized: $\delta^l = \delta^u = 10$. The search profile length was $l_p = 117$ nodes. Points on the search profile were obtained at discrete sampling positions with a distance of 0.35 mm between them. A Gaussian gradient filter kernel with variance $\sigma = 2.0$ mm was utilized to calculate the cost function.

6.2.2 User-Guided Segmentation Refinement

The OSF-based interactive refinement methods utilized in the 4D lung segmentation are similar to the 3D version described in Chapter 5, but work on the 4D graph structure. The individual processing steps of the developed segmentation refinement algorithm are summarized below.

1. The user selects a volume of the 4D scan (Fig. 6.4(a) and 6.4(b)). The user in-



(a) (b) (c)

(d) (e) (f)

Figure 6.4: Example of interactive 4D OSF-based segmentation refinement. The segmentation leaks to the trachea in inspiration and expiration volumes. (a) The user inspects the lung segmentation and locates a segmentation error (inspiration scan). (b) Corresponding expiration scan. (c) In a cross-section, the user selects a point on the correct boundary location of the inspiration scan. Note that the incorrect portion of the contour is highlighted in light blue, which was automatically generated based on the selected point. (d), (e) and (f) Refinement results after calculating maximum-flow. (d) and (e) Corrected inspiration scan. (f) Corrected expiration scan.

spects the segmentation result and detects an error on the surface by comparing CT data visualized on the cutting plane to the boundary of the segmentation result (Fig. 6.4(a)).

2. The incorrect part of the surface is labeled. For this purpose, the user identifies a point on the true surface location near the error region (Fig. 6.4(c)). During this process, the algorithm displays the estimated incorrect (labeled) portion of the surface interactively, which allows the user to pick a good location for the input point.
3. Two tools (generic tool and specific tool correcting leakage to trachea and main bronchus) described in Chapter 5 can be utilized for step 2 and 3. During refinement, cost function of selected scan is manipulated but due to the inter-phase motion constraints of graph structure, both surfaces might be changed (Fig. 6.4).
4. The maximum-flow is recalculated for the graph $G(N, A)$.
5. The new solutions (surfaces) are displayed (Fig. 6.4(d), 6.4(e) and 6.4(f)).

6.3 Evaluation Methodology

6.3.1 Image Data and Experimental Setup

For this pilot study, four 4D data sets are utilized for performance assessment, which show similar segmentation errors at inspiration and expiration. The image sizes ranged from $512 \times 512 \times 141 \times 2$ to $512 \times 512 \times 264 \times 2$ voxels. The slice thickness of images was 2.0 mm and the in-plane resolution 0.98×0.98 mm. For all data sets, an

initial 4D segmentation was produced with the method described in Section 6.2.1. The user was asked to inspect corresponding inspiration and expiration lung segmentation and to utilize the 4D segmentation refinement framework to correct errors, if needed.

6.3.2 Independent Reference Standard and Quantitative Error Index

For segmentation error assessment, 5 axial, 5 coronal and 5 sagittal planes were randomly selected per left and right lung that needed refinement. Thus, for 210 planes (15 planes in seven left/right lungs at two respiratory states) and independent reference standard were generated. The randomly selected planes were produced using the following procedure. First, the areas of automated ($A1$) and refined ($A2$) segmentation are calculated for each axial, coronal or sagittal plane, and difference $A_e = A1 - A2$ was calculated. Second, if either $A1 > 900\ mm^2$ or $A2 > 900\ mm^2$, the place is considered as a candidate for next steps, and the largest value A_{max} of A_e per lung slice orientation is found. Third, all candidate planes of given orientation are sorted into 3 sets. Set 1 contains all planes with $A_e \leq 50\ mm^2$, set 2 contains all planes with $50\ mm^2 < A_e \leq 0.5 \times (50 + A_{max})\ mm^2$, and set 3 contains all planes with $A_e > 0.5 \times (50 + A_{max})\ mm^2$. Fourth, 1, 2 and 2 planes are randomly selected out of set 1, 2, and 3, respectively. For each 3D lung of utilized 4D data sets, a independent reference standard was generated for selected planes by utilizing a commercial lung image analysis software package Apollo (VIDA Diagnostics Inc., Coralville, IA). An expert inspected and manually corrected Apollo lung segmentation results, if need,

for all 210 slices using the editing tool of 3D Slicer[2]. This process took 2.58 h. The Dice coefficient D [107] is utilized as quantitative error index.

6.4 Results

A comparison of performance between proposed 4D OSF-based automated and refined approach is shown in Table 6.1, and corresponding boxplots of the Dice coefficient are depicted in Fig. 6.5. For the 7 test cases, the proposed refinement process took 3.3 min of user interaction on average. The actual computing time per refinement iteration required by the algorithm was 337 ± 875 ms (median: 154 ms) with a minimum and maximum computing time of 105 and 6696 ms, respectively.

Table 6.1: Comparison of segmentation performance between automated and refined results for inspiration, expiration and 4D data.[†]

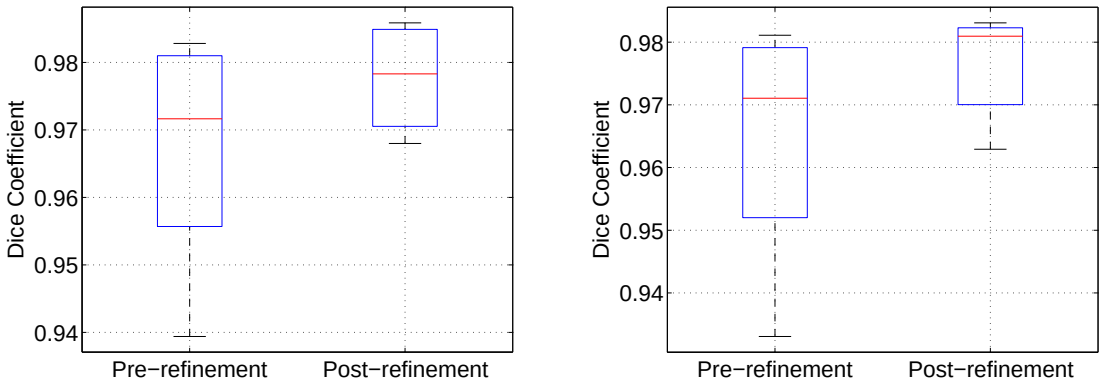
	Inspiration	Expiration	4D
Pre-refinement	0.967 ± 0.016	0.964 ± 0.018	0.966 ± 0.017
Post-refinement	0.978 ± 0.008	0.976 ± 0.008	0.977 ± 0.008

[†] The mean ± standard deviation is given for the Dice coefficient.

6.5 Discussion

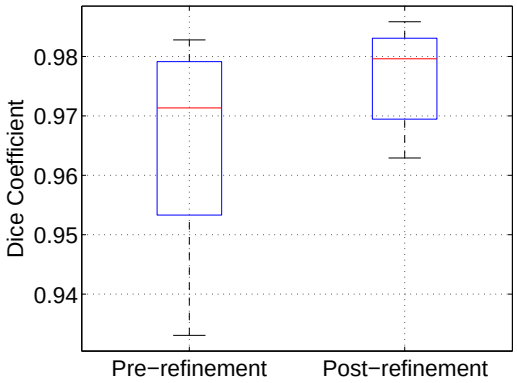
In this chapter, a feasibility study of 4D OSF-based lung segmentation with refinement was performed on four 4D CT scans. Quantitative assessment of seg-

[2]http://www.slicer.org



(a)

(b)



(c)

Figure 6.5: Boxplots of Dice coefficient for automated segmentation and segmentation refinement results. (a) Lungs at inspiration. (b) Lungs at expiration. (c) Combined inspiration and expiration results.

mentation refinement demonstrated that segmentation accuracy was increased with refinement and 4D OSF-based refinement is feasible. The average actual computing time per refinement iteration of 337 ms suggests that the proposed approach is suited for real-time interactive use. In comparison to the 3D refinement approach described in Chapter 5, the average computing time increased by 125%, mainly due to 4D OSF. However, more research is needed to fully unleash the potential of such an approach. For example, the maximum computing time of 7 s is quite long for interactive method. Potentially this could be addressed by utilizing a parallel maximum-flow algorithm. Also, only two respiratory states were used in this pilot study. Adding more time points will also increase computing time. The time needed for lung registration was approximately 24 min which is quite long comparing to matching a 3D RASM. Thus, it would be interesting to investigate a 4D RASM approach to speed up lung segmentation.

CHAPTER 7
INTERACTIVE SEGMENTATION REFINEMENT FOR 3D DUAL-SURFACE BASED IVUS SEGMENTATION

7.1 Introduction

In this chapter, the proposed interactive refinement framework is utilized and validated in the context of IVUS image segmentation. For this purpose, a dual-surface OSF-based IVUS segmentation framework is presented and application specific segmentation refinement methods are introduced. Note that due to the nature of this task (segmentation of two nested surfaces) only the 2D user interface component of the hybrid user interface is utilized, which is sufficient for this specific task.

7.2 Background and Motivation

Intravascular ultrasound (IVUS) provides two-dimensional cross-sectional images of vessel wall architecture and plaque morphology. When augmented with motorized pullback, three-dimensional images are formed. IVUS imaging is common in coronary catheterization laboratories, augmenting traditional X-ray angiography imaging and providing information about the coronary wall morphology. While traditional coronary angiography provides information about the coronary lumen, IVUS adds information about the coronary wall, its remodeling in response to the atherosclerotic processes, and – when virtual histology is included – about atherosclerotic plaque composition. IVUS imaging is routinely performed during percutaneous coronary interventions like balloon angioplasty with or without stent placement to determine ves-

sel geometry, plaque status, presence of ulcerations, to correctly size the angioplasty balloons, determine proper diameter and length of coronary stents, assess resulting stent apposition, etc.

To obtain IVUS images, an imaging catheter is extended distally to the desired location of the coronary artery under fluoroscopic guidance. Three-dimensional IVUS imaging is performed by mechanical pullback of the imaging catheter from its initial distal (downstream) position. Typically, the acquired image frames are cardiac cycle (usually R-wave) gated to provide phase-specific 3-D IVUS image sequences. Such R-wave gated sequences can be visualized and quantitatively analyzed as a straight pipe or in its correct geometry when fused with two-plane angiographic image data providing 3-D vessel geometry [120].

Atherosclerotic plaque is located between two borders (in 2-D) or surfaces (in 3-D) that can be identified in IVUS images – luminal surface (interface between blood and intima) and the surface formed by the external elastic lamina (EEL, media–adventitia interface). IVUS segmentation of the lumen and EEL borders/surfaces is of substantial clinical interest and contributes to clinical decision making. Yet, no truly reliable and consistently accurate IVUS segmentation methods exist that would guarantee segmentation success in clinical setting. This is especially true considering that close-to-real-time performance is required.

IVUS segmentation methods have been reported for almost 20 years. Despite a considerable effort devoted to this task and a number of partial successes, no perfect solution emerged that would allow reliable automated analysis of IVUS

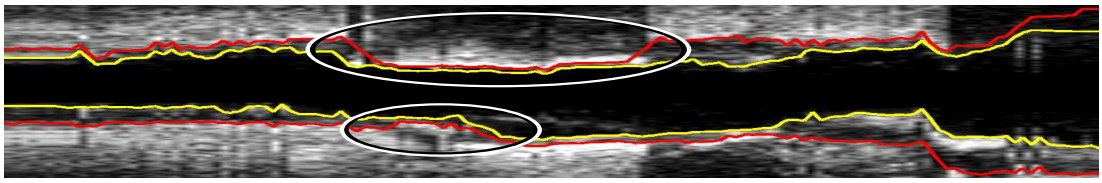


Figure 7.1: Example of regional segmentation errors caused by plaque calcification and image artifacts (locations marked with ellipses). Note that the image was axially reformatted from a sequence of R-wave gated IVUS pullback images.

data. There is a number of reasons making IVUS segmentation very difficult due to the presence of a variety of artifacts including blood speckle, near-field artifacts, strut or guidewire artifacts, reverberations, non-uniform rotational or axial distortion, missing information due to calcified plaque shadowing, etc. The early methods were based on 2-D dynamic programming detecting the lumen and EEL borders independently [108, 132, 37]. Three-dimensional approaches followed and included active surfaces [66, 65], level sets [20, 21], shape models [114], inverse scattering algorithms [79], or combination of transversal and longitudinal model- and knowledge-guided contour detection techniques [68], to name a few. Recently, LOGISMOS-based approach [75, 128] was applied to 3-D IVUS segmentation [39]. Despite the variety of approaches developed for IVUS segmentation, one observation remains omnipresent in all these methods – while each of the cited automated methods successfully segments IVUS image data in many IVUS image frames, they all fail in a considerable number of frames making automated IVUS segmentation virtually impossible to use in a clinical setting. In all these locations – many of which are of utmost clinical

relevance – the interventional cardiologist must resort to tedious and time-consuming manual tracing of a large number of IVUS frames to obtain acceptable boundaries and desirable quantitative indices of morphology or plaque virtual histology. The fact that obtaining virtual histology information about the plaque tissue is directly dependent on correct segmentation of the luminal and EEL surfaces further increases importance of this task.

Motivated by the pressing clinical need of achieving successful segmentation of all IVUS frames of interest and realizing that the most important coronary locations are the most diseased ones, which may suffer from the most severe imaging artifacts like calcium shadowing (Fig. 7.1), a two-stage approach consisting of automated and semi-automated steps based on optimal dual-surface graph based segmentation (LOGISMOS) is introduced in this chapter. Note that the work described in [39] also uses LOGISMOS for segmentation, but investigation showed that this approach is not suitable for segmentation refinement due to the utilized cost function design and issues with hard smoothness constraint. In the first stage of the proposed algorithm, the graph is built and initial optimal segmentation of the lumen and EEL are determined automatically. The second stage is optional and can be seen as a "dialog" between the user and the previously utilized algorithm, where the user provides rough clues for the desired locations of at-this-stage incorrectly positioned boundaries by augmenting the graph's objective function. Utilizing such locally targeted expert interactions that act directly on the optimized graph, sub-second interaction responses yield updated pairs of segmentation surfaces in an interactive user-driven semi-automated fashion.

A small number of interactions is typically sufficient to achieve fully satisfactory 3-D segmentations of IVUS image sequences consiting of hundreds of R-wave gated frames. As a result, an accurate, automated, and performance-efficient method has been developed facilitating routine segmentation of complete IVUS pullback sequences almost immediately after completing the IVUS image acquisition.

7.3 Method

The proposed approach to IVUS segmentation consists of two main stages: a) initial automated segmentation and b) interactive graph-optimization driven segmentation refinement, if needed. In stage a), lumen (inner) surface and EEL (outer) surface are segmented simultaneously in 3-D. For this purpose, the lumen is first roughly pre-segmented and luminal centerline determined, facilitating construction of graph for LOGISMOS-based dual surface segmentation [75, 128]. Both the pre-segmentation and the simultaneous dual-surface segmentation are fully automated and yield optimal surfaces with respect to the employed objective function. Any local or regional segmentation errors can be identified by the expert operator and efficiently corrected in the second stage of our approach. The basic idea behind our refinement stage is that the user is allowed to interact directly with the LOGISMOS-based segmentation algorithm by providing rough clues regarding the desired boundary location.

7.3.1 Graph Construction for IVUS Segmentation

Both the lumen pre-segmentation and dual-surface segmentation in stage a) of our approach utilize the LOGISMOS-based approach.

For dual-surface LOGISMOS, a weighted graph G containing two sub-graphs $G = \{G_i = (V_i, A_i) : i = 1, 2\}$ is constructed, where V_i represents a graph node set, A_i a graph arc set, G_1 inner surface (S_1) sub-graph and G_2 outer surface (S_2) sub-graph. $v \in V_i$ is a graph node on a column, the length of which is l_p. Each node v corresponds to a point of intensity $I(x, y, z)$ in the volumetric image stack I.

G_i consists of sub-graphs $G_{ik} \in G_i$ with $k = 1, 2, ..., Z$ where Z represents the number of image frames in the stack. For each sub-graph G_{ik}, n_p graph columns are generated from a center point μ_k in radial directions at $\theta_p = 2\pi/n_p$ angle increments (Fig. 7.2(a)). The distance between nodes along the column is d_n, which is usually set to match image resolution.

Intra-column arcs are built to connect nodes $n(v, m)$ to $n(v, m-1)$ on a column $col_k(v)$ with infinity weights, where m is the column node index. Inter-column arcs are built to connect the node $n(v_i, m)$ to the node $n(v_j, m - \Delta_a)$ with infinity weights (Fig. 7.2(b)), where Δ_a is intra-frame hard smoothness constraint. To obtain a 3-D graph G_i, sub-graphs $\{G_{ik} : k = 1, 2, ..., Z\}$ are connected by inter-column arcs, which pairwise connect nodes $n(v_i, m)$ to nodes $n(v_i, m - \Delta_b)$ on the column $col_k(v_i)$ and $col_{k\pm1}(v_i)$ with infinity weights (Fig. 7.2(c)), where Δ_b is inter-frame hard smoothness constraint.

Graphs G_1 and G_2 are interconnected by inter-surface arcs which connect

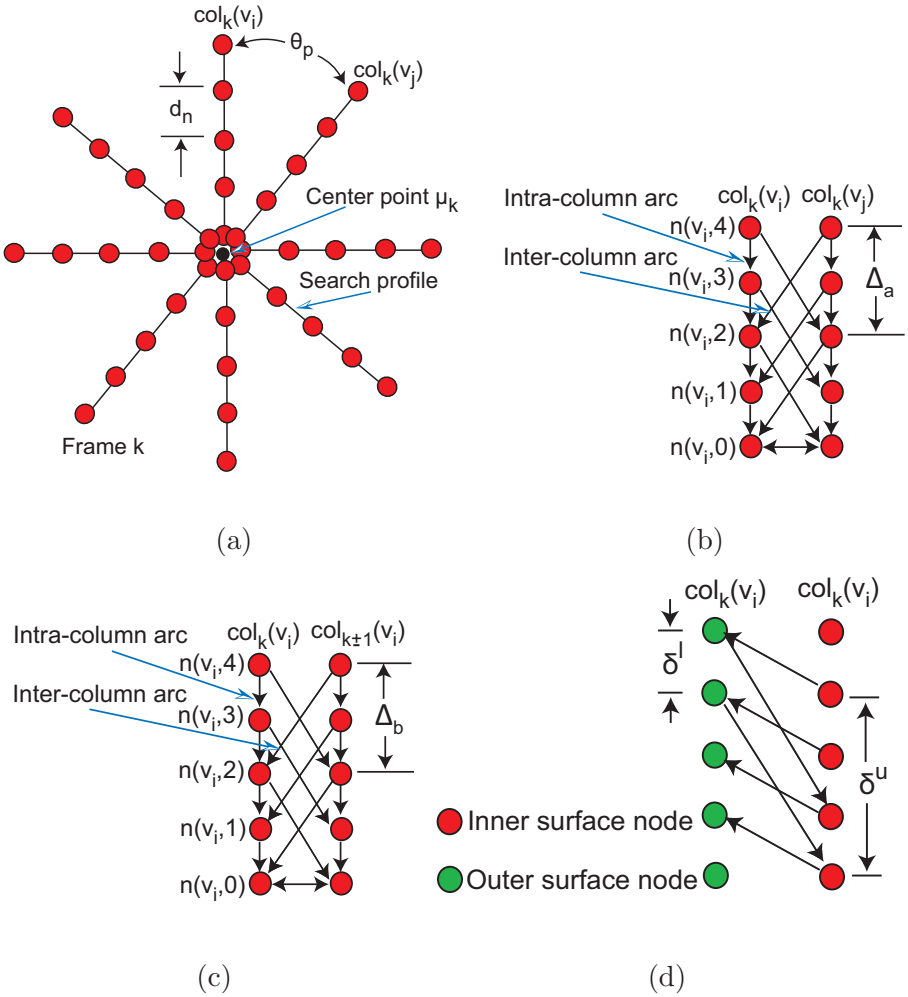


Figure 7.2: Graph construction for dual-surface LOGISMOS. (a) Search profiles of a single surface are constructed starting from vessel centerline point μ_k determined for every IVUS frame. (b) LOGISMOS graph structure of a single surface with arcs enforcing the surface smoothness constraints between adjacent columns on the same frame. (c) LOGISMOS graph structure of a single surface with arcs enforcing the surface smoothness constraints between the corresponding columns on adjacent image frames. (d) LOGISMOS graph structure with arcs enforcing the inter-surface constraints. See [75, 128] for additional details.

nodes $n(v_i, m)$ of G_2 and $n(v_i, m-\delta^u)$ of G_1 with infinity weights, and nodes $n(v_i, m)$ of G_1 and $n(v_i, m+\delta^l)$ of G_2 with infinity weights (Fig. 7.2(d)), where δ^l and δ^u are lower-limit and upper-limit interaction constraints representing the minimum and maximum allowed distances between surfaces S_1 and S_2.

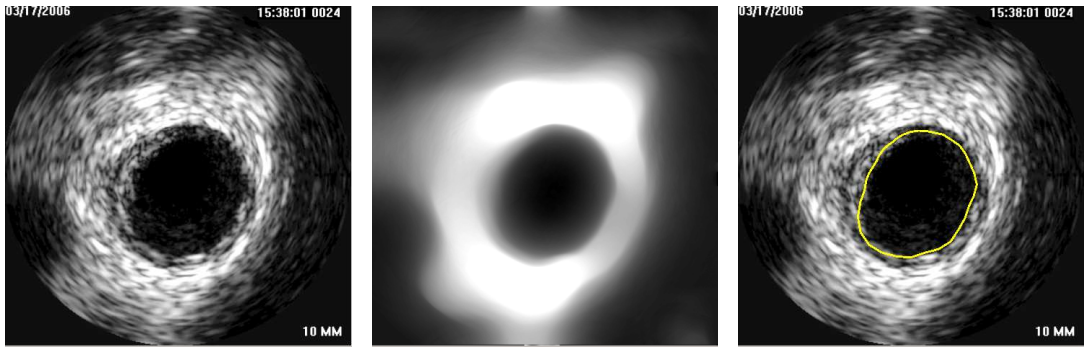
Graph node weight sets C_i with $i \in \{1, 2\}$ (objective functions) are derived from volumetric image data to reflect local image characteristics. In addition, linear soft smoothness constraints are utilized, as proposed by Song et al. in [105]. For this purpose, constant weight α_{ia} (intra-frame) and α_{ib} (inter-frame) arcs are introduced to penalize shifts between adjacent vertices on surfaces S_i.

The same LOGISMOS-based segmentation approach is utilized for lumen pre-segmentation and dual-surface lumen–EEL segmentation. Both approaches use the same graph construction, with the exceptions of hard smoothness constraints, cost function design and utilized center point locations $\{\mu_k : k = 1, 2, ..., Z\}$. In this application, the graph-construction parameters common for the pre-segmentation and dual-surface segmentation were: $n_p = 36$, $l_p = 96$ nodes, $d_n = 2$ pixels, $\delta^l = 4$ nodes, $\delta^u = l_p$, $\alpha_{1a} = 0.01$, $\alpha_{1b} = 0.01$, $\alpha_{2a} = 0.005$, and $\alpha_{2b} = 0.005$. Segmentation-step specific parameters are provided in the relevant paragraphs below.

7.3.2 Automated Segmentation

a) Lumen Pre-segmentation: The goal of this step is to roughly pre-segment the lumen (surface S_1) of the IVUS volume to estimate the lumen centerline, which will be used for the subsequent dual-surface segmentation (Section 7.3.2 b)).

For the lumen pre-segmentation, dual-surface LOGISMOS framework is utilized and G_2 presence is solely used to constrain the search for G_1; there is no goal of determining an accurate location of the S_2 surface at this step. Note that only rough boundary locations are required for this step. First, a total variation (TV) regularized L^1 model-based decomposition [126] with regularization parameter $\lambda = 0.01$ is used to remove high-frequency details like speckle noise (Figs. 7.3(a) and 7.3(b)).



(a) (b) (c)

Figure 7.3: Lumen pre-segmentation. (a) Original IVUS image. (b) Image after TV decomposition. (c) Lumen pre-segmentation result shown in yellow.

The LOGISMOS-based lumen pre-segmentation uses the center of the imaging catheter (image center) as μ_k for $k = 1, 2, \ldots, Z$. The cost functions (C_1 and C_2) for surfaces S_i assign the following costs to column nodes $n(v, j)$:

$$c_i(v,j) = \begin{cases} g_{max} & if\ \mathbf{n}_v \cdot \mathbf{g}_{dir}(v,j) < 0 \\ g_{max} - g_{mag}(v,j) & otherwise \end{cases} . \tag{7.1}$$

Normalized gradient magnitude (range of $[0, 1]$), gradient direction, and surface nor-

mal vectors (pointing away from μ_k) are denoted $g_{mag}(v,j)$, $\mathbf{g}_{dir}(v,j)$, and $\mathbf{n}_v$, respectively. The gradient calculation is based on Gaussian derivatives with standard deviation of σ. Linear interpolation is utilized to obtain costs $c_i(v,j)$. The following parameters were used for pre-segmentation: $\sigma = 0.1$ mm, $\Delta_{1a} = 15$, $\Delta_{1b} = 15$, $\Delta_{2a} = 6$, and $\Delta_{2b} = 9$. Fig. 7.3(c) shows an example of lumen pre-segmentation.

b) Dual-Surface Segmentation: The dual-surface segmentation is based on the same LOGISMOS framework (Section 7.3.1). The center position μ_k for each image frame is however derived from the lumen pre-segmentation (Section 7.3.2 a)) and original unfiltered image data are utilized in the cost functions.

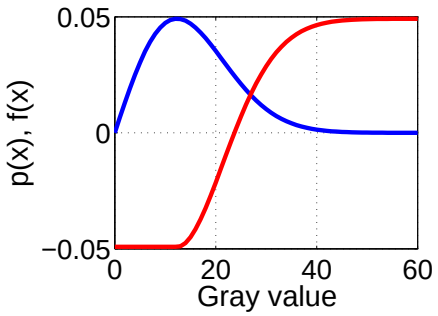


Figure 7.4: Relation between estimated Rayleigh probability density function (blue curve) and utilized gray-value weighting function $f(x)$ (red curve). $a = 12.3$ is utilized.

Outer boundary (EEL) costs $c_2(v,j)$ are calculated as given in Eq. 7.1. Designing a suitable cost function for the inner boundary is more challenging, because

this border is less well depicted in the images. In addition, calcifications, vessel bifurcations, etc. need to be considered. To address these issues, a combination of edge-based and in-region-based costs are utilized

$$c_1(v,j) = c_e(v,j) + 0.4 * c_r(v,j) \ , \tag{7.2}$$

where $c_e(v,j)$ is the edge based cost function equivalent to Eq. 7.1 and

$$c_r(v,j) = \sum_{m=1}^{j} f(x(v,m)) \ , \tag{7.3}$$

represents an in-region cost term with a gray-value weighting function

$$f(x) = \begin{cases} -\frac{e^{-0.5}}{a} & if \quad x < a \\ \frac{e^{-0.5}}{a} - 2\frac{x}{a^2}e^{-\frac{x^2}{2a^2}} & else \end{cases} \ . \tag{7.4}$$

In this context, the gray-value at node $n(v,m)$ is denoted by $x(v,m)$. The design of the weighting function is inspired by the Rayleigh probability density function (PDF), which is given by

$$p(x) = \frac{x}{a^2}e^{-\frac{x^2}{2a^2}} \ , \tag{7.5}$$

where x is the gray-value of a pixel and $a > 0$. Specifically, we model gray-values of the lumen region by means of a Rayleigh PDF, which is well suited to describe the typical speckle noise pattern found in ultrasound images [119]. A plot showing the relationship between Eqs. 7.4 and 7.5 is given in Fig. 7.4. Function $f(x)$ returns low costs if IVUS gray-values are likely belonging to the lumen. Note that $f(x)$ was designed such that the catheter, which is blackened out with a gray-value of zero also receives low costs. Parameter a of the Rayleigh PDF is estimated from the

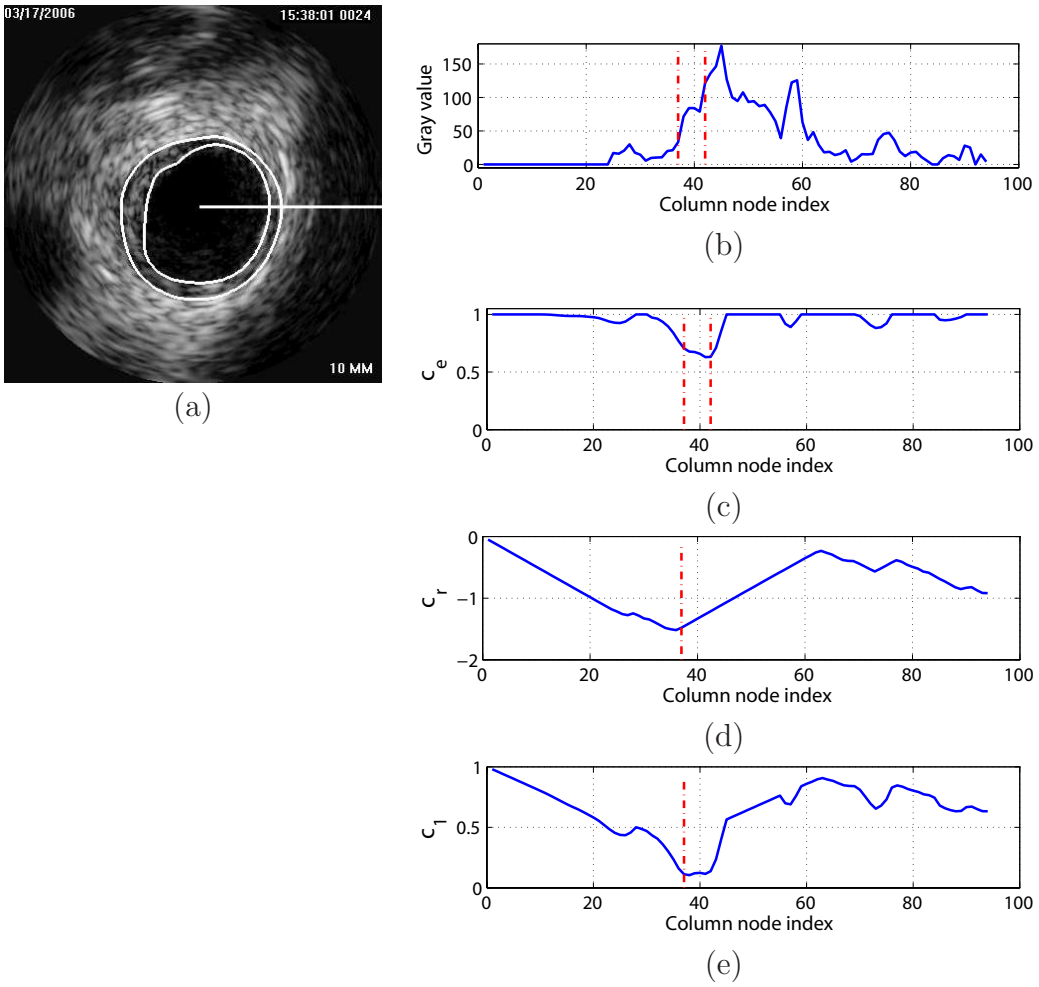


Figure 7.5: Cost function calculation for dual-surface automated segmentation. (a) Original IVUS image with expert-defined segmentation (independent standard). The horizontal white line indicates the location of a graph column (starting at the pre-segmentation determined lumen center μ_k) that is utilized for illustration of cost calculation. (b) Column-corresponding gray-value profile. Zero on the horizontal axis represents center μ_k. (c) Edge cost function c_e and (d) in-region cost function c_r derived from (b). (e) Final cost function c_1 for the inner surface. Vertical lines shown in red indicate the locations of the inner and outer contours of the independent standard (a) on this column.

pre-segmented lumen (Section 7.3.2 a)) with

$$a = \sqrt{\frac{1}{2N}\sum_{i=1}^{N} x_i^2}\ , \tag{7.6}$$

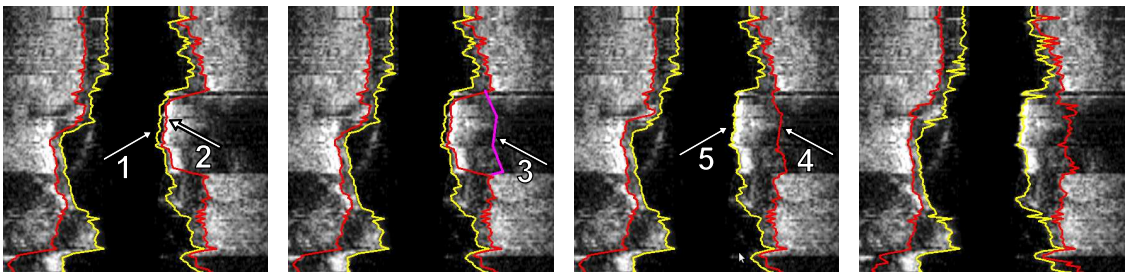
where x_i is a gray-value sampling point and N is the number of sampling points in the ROI defined as a sub-region of the volume between the catheter surface and the segmented lumen (inner) surface.

Graph parameters used for automated dual-surface IVUS segmentation were as follows: $\sigma = 0.1$ mm, $\Delta_{1a} = 5$, $\Delta_{1b} = 6$, $\Delta_{2a} = 4$, and $\Delta_{2b} = 6$. Note that parameters used in the described method were determined experimentally on five cases, which were not included in the test data set. Fig. 7.5 depicts an example of cost function calculation for dual-surface automated segmentation.

7.3.3 User-Guided Segmentation Refinement

Our segmentation refinement method is directly based on the graph structure G built in step 2 of our automated segmentation approach (Section 7.3.2 b)). The user-driven refinement method is based on the framework described in Section 4.3.2. The IVUS specific individual processing steps of the algorithm are depicted in Fig. 7.6 and are described in detail below.

1. The user inspects the segmentation result and locates a segmentation inaccuracy (Fig. 7.6(a)).
2. The user selects either the inner or outer surface for refinement and draws a polygon line and/or specifies a point roughly at the location of the desired surface boundary (Fig. 7.6(b)). This task is supported by a graphical user



(a) (b) (c) (d)

Figure 7.6: Illustration of interactive LOGISMOS-based refinement of an automatically generated IVUS segmentation. This case was previously depicted in Fig. 7.1. (a) The user inspects the IVUS segmentation produced by our automated approach and discovers a local segmentation inaccuracy of the inner (arrow 1) and outer (arrow 2) surfaces. The outer boundary segmentation got "distracted" by a high density (calcified) region inside of the vessel wall and the associated shadow. (b) The user roughly indicates the correct location of the outer wall by drawing a polygon line (arrow 3, purple line) in proximity to the desired surface location. This single polygon line is used to locally modify the cost function for the outer boundary. (c) Refinement result after recalculating the maximum-flow for the dual-surface graph. Note that outer (arrow 4) and inner boundary (arrow 5) are simultaneously corrected due to the mutually interacting dual-surface graph structure. (d) Corresponding independent standard.

interface (Fig. 7.7).

3. Utilizing the information provided by the user, the algorithm locally updates costs in the graph structure G.

4. The maximum-flow is recalculated for the updated graph G.

5. The display of the inner and outer surfaces is updated (Fig. 7.6(c)).

In the following, we provide more details describing step 3 of our algorithm.

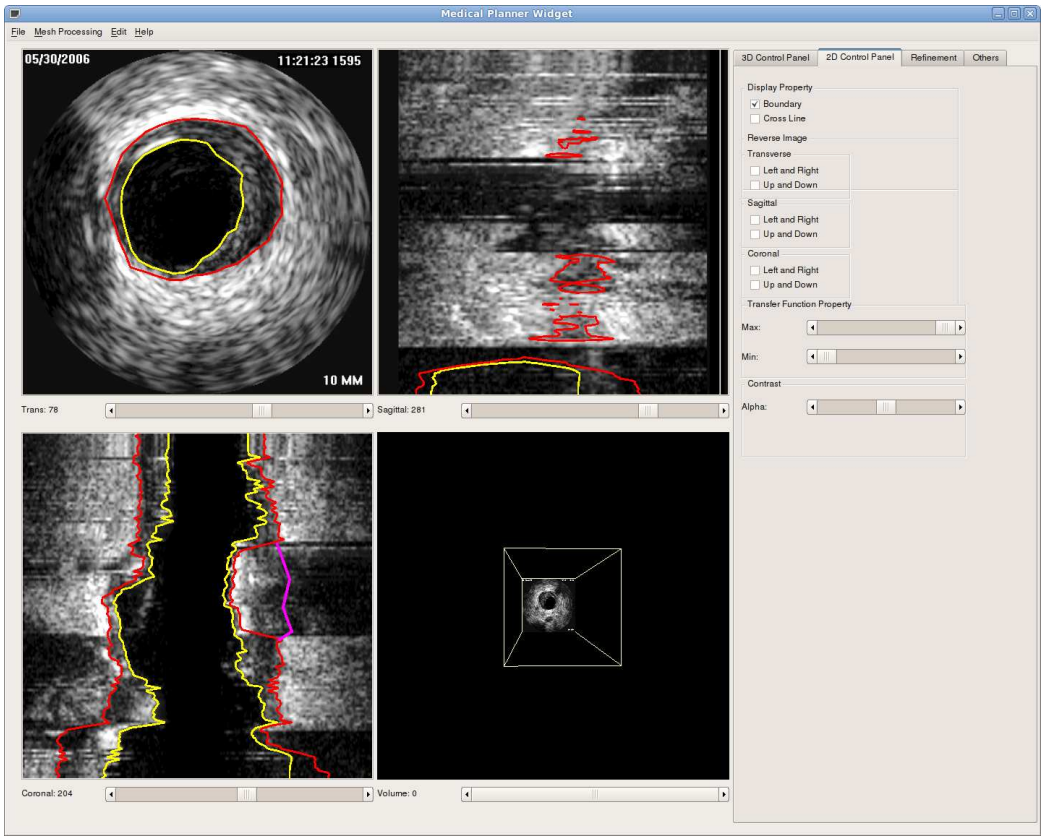


Figure 7.7: 2D graphic user interface used for IVUS segmentation refinement.

As outlined above, the user can guide the segmentation result by drawing polygonal lines in cross-sectional images along the vessel or placing single points in arbitrarily (desired) locations of the IVUS volume. Based on this interactively defined information, node costs in local neighborhoods of the entered points or polygonal lines are modified to affect the outcome of the optimal graph-search segmentation. Single points specified by the user are converted by the algorithm to a polygonal line consisting of one start and end point that are the same.

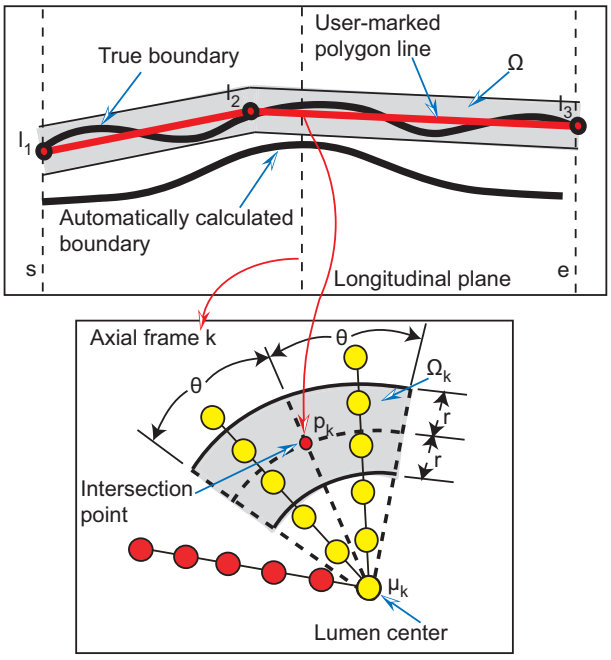


Figure 7.8: Influence region definition for segmentation refinement based on a user-specified polygon line. Affected and unaffected nodes are shown in yellow and red, respectively. See text for details.

Given a polygon line represented by the point set $L = \{l_1, l_2, ..., l_j\}$ that roughly approximates the correct boundary location (Fig. 7.8), intersection points p_k with $k = s, s+1, \ldots, e$ are calculated for each image frame, and wedge sector shaped influence regions Ω_k are defined (gray region in Fig. 7.8). For the influence regions, $r = 7$ pixel and angle $\theta = 20°$ were utilized. Affected columns are defined as those which intersect the volume Ω defined by combining all influence regions $\Omega = \{\Omega_s, \Omega_{s+1}, \ldots, \Omega_e\}$. Let $c_i^t(v,j)$ and $c_i^{t+1}(v,j)$ denote node costs before and after a refinement iteration, respectively. Nodes on the affected columns for surface S_i are updated as follows

$$c_i^{t+1}(v,j) = \begin{cases} U_R\left(c_i^t(v,j), v, j\right) & if \quad n(v,j) \in \Omega_k \\ U_B\left(c_i^t(v,j), v, j\right) & else \end{cases} , \tag{7.7}$$

and the costs of all other (unaffected) columns are left unchanged ($c_i^{t+1}(v,j) = c_i^t(v,j)$). Nodes on the affected columns inside of the region Ω are updated utilizing

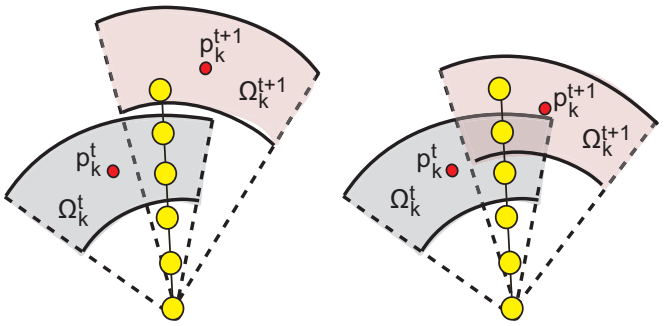
$$U_R\left(c, v, j\right) = \left(1.0 - 0.5e^{\frac{-d(p_k, n(v,j))^2}{2{\sigma_r}^2}}\right) c \, , \tag{7.8}$$

where $d()$ denotes the Euclidean distance function. Parameter $\sigma_r = 5$ pixels adjusts the locality of the cost modification. Nodes on affected columns outside of Ω are updated using

$$U_B\left(c, v, i\right) = min\left(10, c + 10(1 - e^{\frac{-d(p_k, n(v,j)^2}{2{\sigma_r}^2}})\right), \tag{7.9}$$

to penalize nodes (locations) that are very unlikely a part of correct surfaces.

Any single graph column may be impacted by multiple refinement iterations. Thus, the current $(t+1)$ and the previous (t) iteration must be considered. Let p_k^{t+1}



(a) (b)

Figure 7.9: Example of a graph column affected by a previous and current refinement operation. (a) A column is affected by two none-overlapping impact regions. (b) A column is affected by two overlapping impact regions.

and p_k^t be intersection points in frame k that correspond to impact regions Ω_k^{t+1} and Ω_k^t, respectively. For each frame, the following rules are used in conjunction with Eqs. (7.7), (7.8), and (7.9) for updating the costs of a column v that is affected by Ω_k^{t+1} and/or Ω_k^t:

1. If there is no overlap between Ω_k^{t+1} and Ω_k^t (Fig. 7.9(a)) then the impact region Ω_k^{t+1} and the corresponding intersection point p_k^{t+1} are used in Eqs. (7.8) and (7.9).
2. If Ω_k^{t+1} and Ω_k^t are overlapping (Fig. 7.9(b)) then both Ω_k^{t+1} and Ω_k^t are employed based on the rules given in Table 7.1.

Table 7.1: Rules for updating the cost function during segmentation refinement in dependence of location of node $n(v,j)$.

	$n(v,j) \in \Omega_k^{t+1}$	$n(v,j) \notin \Omega_k^{t+1}$
$n(v,j) \in \Omega_k^t$	Eq. 7.8: $d(p_k, n(v,j))$ is replaced with $min\{d(p_k^{t+1}, n(v,j)), d(p_k^t, n(v,j))\}$	Eq. 7.8: p_k is replaced with p_k^t
$n(v,j) \notin \Omega_k^t$	Eq. 7.8: p_k is replaced with p_k^{t+1}	Eq. 7.9: $d(p_k, n(v,j))$ is replaced with $min(d(p_k^{t+1}, n(v,j)), d(p_k^t, n(v,j))$

7.4 Experimental Methods

7.4.1 Image Data and Experiment Setup

For our study, 41 data sets were available originating from a Volcano IVG3 imaging system with 20 MHz solid-state catheters. Combined with a mechanical pullback device, the Volcano system provides EKG R-wave gated image sequences. Considering the usual heart rate of 60-90 beats per minute, the catheter pullback speed of 0.5 mm/sec with EKG gating provides an IVUS image frame every 0.3–0.5 mm axially. Each image frame is 384×384 pixels in size, with in-frame resolution of 0.026×0.026 mm. Temporal pullback sequences of 70 to 259 frames were included in our data set with frame-to-frame distances ranging from 0.25 to 0.69 mm.

Automated simultaneous segmentation of the inner and outer surfaces was performed on all 41 test data sets, for a total of 6467 frames. All computations were performed on a Linux workstation with a 2.93 GHz Xeon CPU with program's mem-

ory requirements never exceeding 2 GB on the tested IVUS data set. An expert was asked to inspect the segmentation results generated by the fully automated approach and refine the segmentation using the second stage of our approach.

For performance comparison, an earlier-reported automated IVUS segmentation method [39] was used. Results using approach published in [39] are labeled as "PA" (previous automated), the results of proposed new automated approach are labeled as "NA", and the results of proposed new refinement approach applied to NA are labeled "NR".

7.4.2 Independent Standard

Manual tracing of the luminal and EEL borders was performed by an expert interventional cardiologist. The independent standard resulted from frame-by-frame editing or retracing borders resulting from [39]. The manual tracing environment allowed to trace surfaces either in individual frames or in one of 6 axially reformatted planes (30 degree increments). The expert observer was allowed to select the individual planes or frames in any sequence and modify the borders until full satisfaction. In the process, all frames of each image sequence were reviewed, and most if not all manually traced and repeatedly edited. This way of defining an independent standard was very tedious and time consuming, typically requiring 2-3 hours of manual tracing and editing per image sequence. A high-quality independent standard resulted that was used for performance assessment by the methods under comparison.

7.4.3 Quantitative Indices

The following quantitative error indices are utilized: mean signed border positioning error (d_s), mean unsigned border positioning error (d_u), root-mean-square (RMS) border positioning error (d_{rms}), mean signed area error (A_s), mean unsigned area error (A_u) and RMS area error (A_{rms}). All these quantitative indices were utilized in [39]. In the case of d_s and A_s, a negative value indicates that the segmentation border is inside and a positive value indicates that the border is outside of the expert-defined boundary/surface. To compute quantitative indices, borders on each frame are considered as points in polar coordinates at 360 one-degree angles. Border positioning errors are calculated for each boundary point as a distance between the independent standard point and the point on the segmented boundary. Indices d_s, d_u, d_{rms} are calculated per sequence as averaged distances over all boundary points of the 3-D pullback sequence. Similarly, the area errors are calculated for each frame and A_s, A_u and A_{rms} are the averaged results over the entire pullback sequence.

7.5 Results

The mean and standard deviation of quantitative indices for lumen and EEL surfaces for PA, NA and NR approaches are summarized in Table 7.2. A comparison between boxplots of all quantitative indices and methods are shown in Fig. 7.10 and 7.11. In addition, paired Student's t-tests at a significance level of 0.05 were performed to determine whether the average error indices were significantly different when comparing segmentation approaches (Table 7.3).

The mean and standard deviation of the computing time needed for automated segmentation (approach NA) per data set was 42.6±13.8 s, and the median was 40.0 s. The computing time ranged between 9.0 and 70.5 s. The portion of time required for obtaining the solution of the maximum-flow calculation in luminal pre-segmentation and dual-surface segmentations steps combined was 5.1 ± 3.3 s (median: 4.2 s).

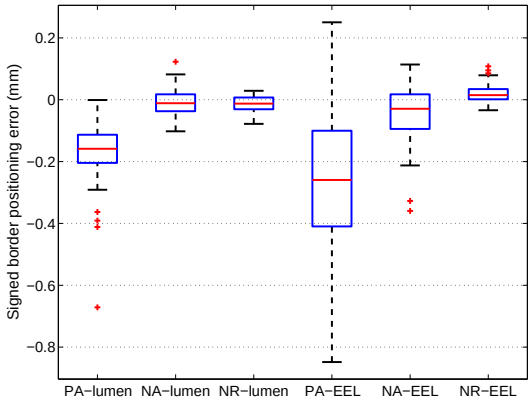
The mean and standard deviation of user interaction time needed by the expert for segmentation refinement (approach NR) per case was 5.7 ± 1.8 min. The median user interaction time was 5.8 min, and the required times ranged between 0.9 and 9.3 min for the set of all 41 tested IVUS pullbacks. The measured user interaction time includes locating local segmentation inaccuracies, identifying and marking correct border locations with polygon lines or points, and (iteratively) applying the surface refinement algorithm. The processing time required for computing the refinement results was 86 ± 57 ms (median: 79 ms) per iteration. Overall, the time needed for both stages of our method consisting of the automated segmentation and user-guided refinement was 6.5 ± 1.8 min with a median of 6.6 min. The maximum and minimum were 10.0 and 1.9 min, respectively.

Examples of segmentation results generated with the proposed automated approach (NA) (without refinement) are shown in Fig. 7.12. Comparisons with the independent standard as well as with surfaces generated using the PA, NA, and NR approaches are depicted in Figs. 7.13 and 7.14.

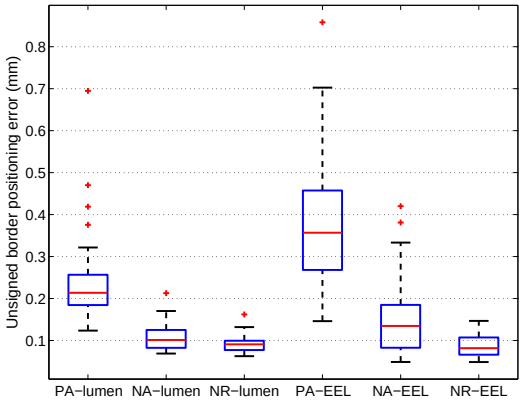
Table 7.2: Quantitative segmentation performance indices of methods PA, NA, and NR for the luminal and EEL surfaces on 41 IVUS data sets.†

	PA-lumen	NA-lumen	NR-lumen	PA-EEL	NA-EEL	NR-EEL
d_s (mm)	-0.182 ± 0.119	-0.007 ± 0.044	-0.013 ± 0.026	-0.252 ± 0.240	-0.049 ± 0.107	0.024 ± 0.034
d_u (mm)	0.238 ± 0.104	0.109 ± 0.033	0.092 ± 0.020	0.384 ± 0.158	0.148 ± 0.089	0.088 ± 0.025
d_{rms} (mm)	0.378 ± 0.175	0.168 ± 0.055	0.138 ± 0.038	0.563 ± 0.190	0.264 ± 0.169	0.136 ± 0.043
A_s (mm^2)	-2.452 ± 1.995	-0.005 ± 0.624	-0.108 ± 0.340	-3.489 ± 3.067	-0.578 ± 1.495	0.367 ± 0.601
A_u (mm^2)	2.573 ± 1.960	0.749 ± 0.406	0.551 ± 0.221	4.618 ± 2.134	1.559 ± 1.118	0.793 ± 0.435
A_{rms} (mm^2)	3.624 ± 2.532	1.124 ± 0.668	0.817 ± 0.437	5.919 ± 2.478	2.348 ± 1.611	1.173 ± 0.650

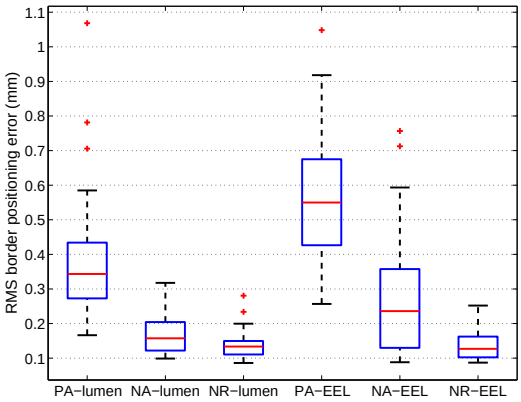
† The mean ± standard deviation are given for each index.



(a)

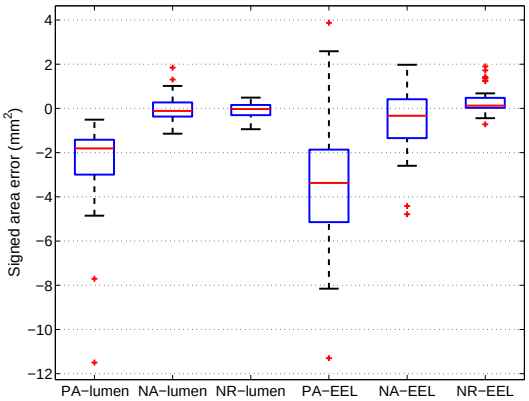


(b)

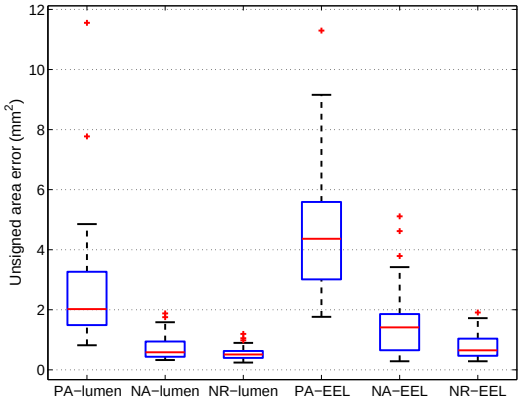


(c)

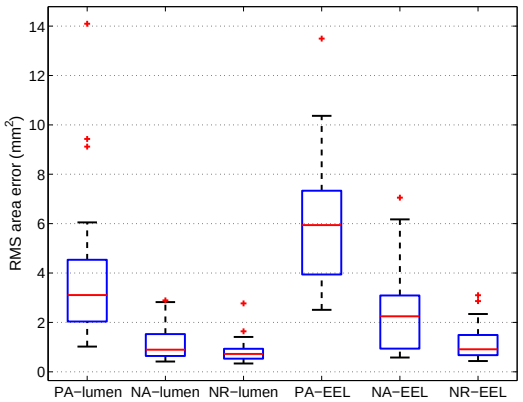
Figure 7.10: Comparison between boxplots of quantitative indices (border positioning error) for IVUS segmentation of luminal and EEL surfaces with methods PA, NA, and RA. (a) Signed border positioning error. (b) Unsigned border positioning error. (c) RMS of border positioning error.



(a)



(b)



(c)

Figure 7.11: Comparison between boxplots of quantitative indices (area error) for IVUS segmentation of luminal and EEL surfaces with methods PA, NA, and RA. (a) Signed area error. (b) Unsigned area error. (c) RMS of area error.

Table 7.3: Student's t-test statistics comparing the tested segmentation approaches - p-values provided.

	Luminal surface			EEL surface		
	PA vs. NA	NA vs. NR	PA vs. NR	PA vs. NA	NA vs. NR	PA vs. NR
d_s (mm)	$\ll 0.001$	0.25	$\ll 0.001$	$\ll 0.001$	$\ll 0.001$	$\ll 0.001$
d_u (mm)	$\ll 0.001$	$\ll 0.001$	$\ll 0.001$	$\ll 0.001$	$\ll 0.001$	$\ll 0.001$
d_{rms} (mm)	$\ll 0.001$	$\ll 0.001$	$\ll 0.001$	$\ll 0.001$	$\ll 0.001$	$\ll 0.001$
A_s (mm^2)	$\ll 0.001$	0.17	$\ll 0.001$	$\ll 0.001$	$\ll 0.001$	$\ll 0.001$
A_u (mm^2)	$\ll 0.001$	< 0.001	$\ll 0.001$	$\ll 0.001$	$\ll 0.001$	$\ll 0.001$
A_{rms} (mm^2)	$\ll 0.001$	< 0.001	$\ll 0.001$	$\ll 0.001$	$\ll 0.001$	$\ll 0.001$

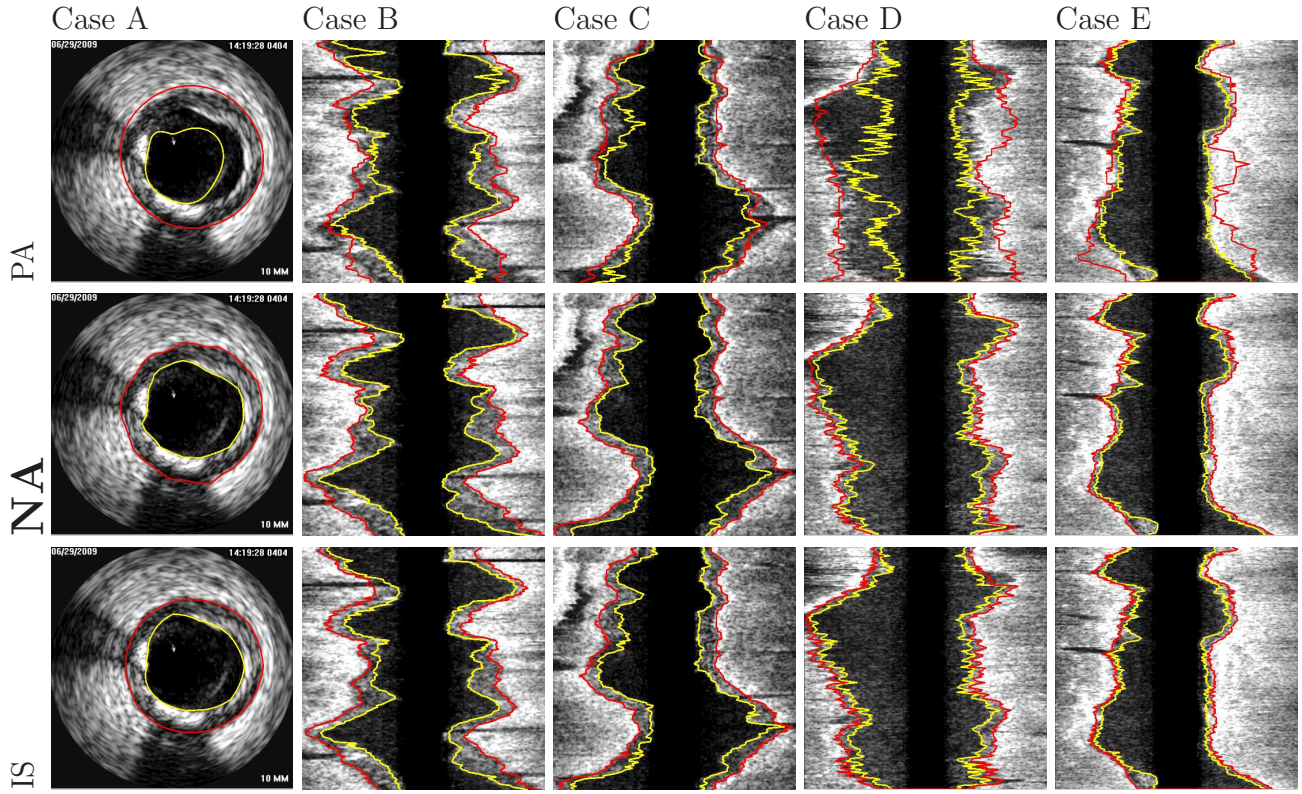


Figure 7.12: Comparison of automatically generated segmentation results on five different data sets (cases A-E). The luminal and EEL surfaces are shown in yellow and red, respectively. (PA) Method reported in [39]. (NA) Our automated segmentation approach. (IS) Independent standard.

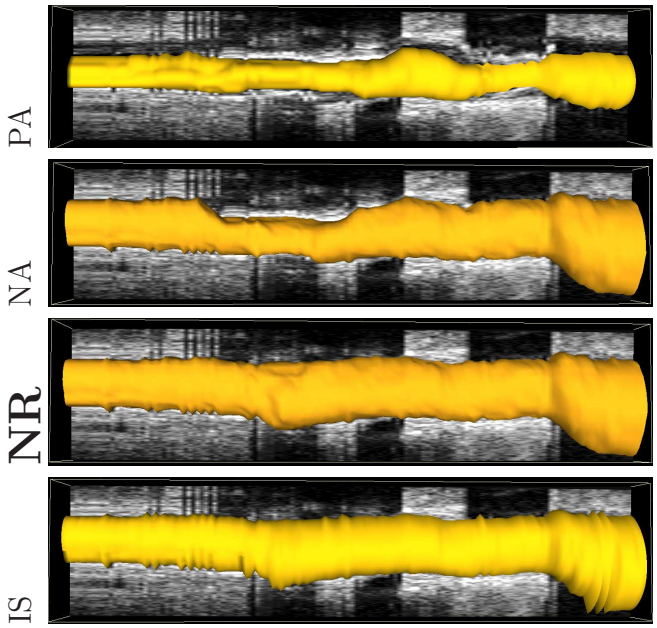


Figure 7.13: Comparison of segmentation results in mesh-based 3-D representation of the EEL surface. Note that this data set was also shown in Figs. 7.1 and 7.6. (PA) Method reported in [39]. (NA) Our automated segmentation approach. (NR) Segmentation refinement result. (IS) Independent standard.

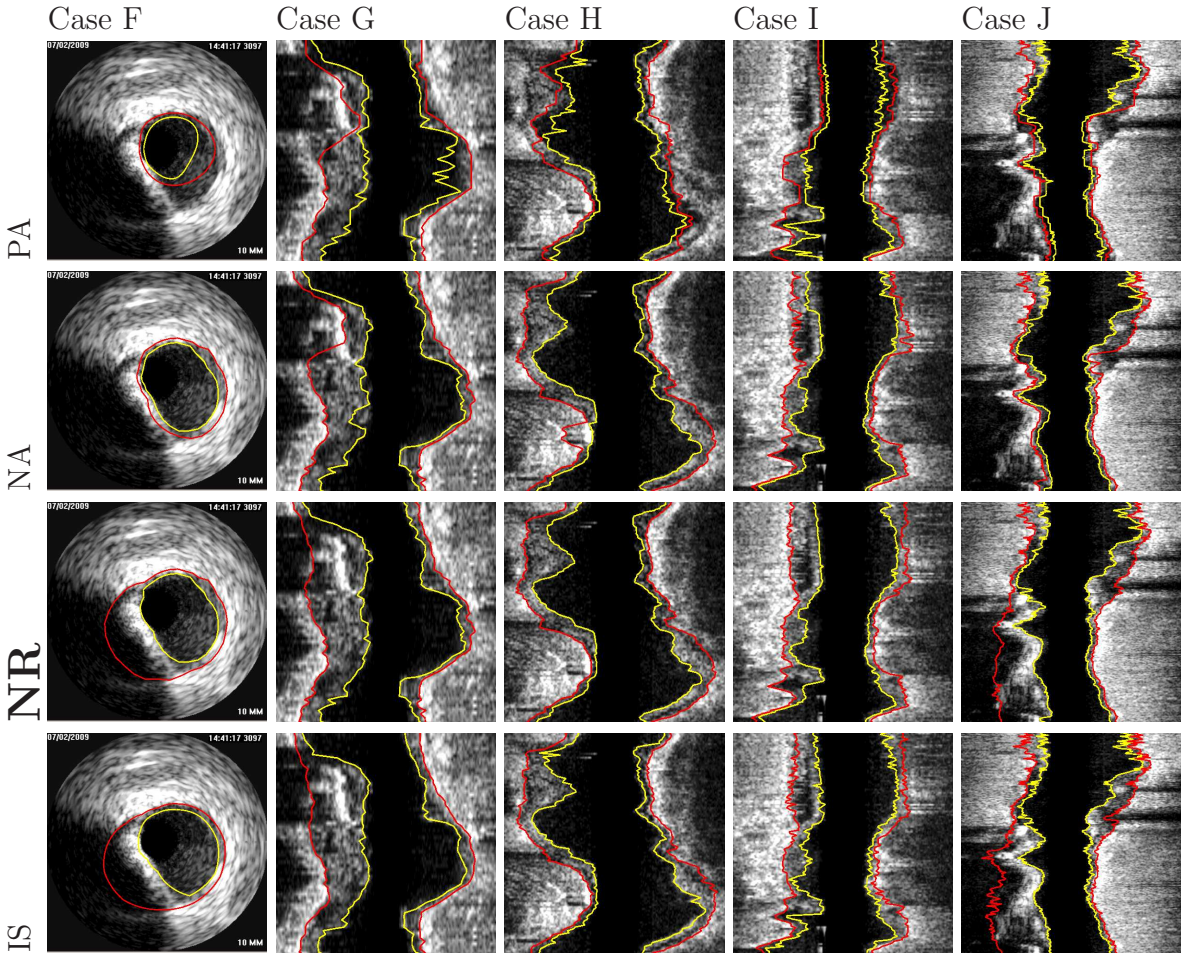


Figure 7.14: Examples of segmentation results on five different data sets (cases F-J). The luminal and EEL surfaces are shown in yellow and red, respectively. (PA) Method reported in [39]. (NA) Our automated segmentation approach. (NR) Segmentation refinement result. (IS) Independent standard.

7.6 Discussion

Proposed new automated IVUS segmentation method (the NA method) outperformed the previous published approach (the PA method) as documented by the results shown in Table 7.2, Figs. 7.10 and 7.11. The obtained improvements are practically relevant and statistically significant across all quantitative indices of border positioning and area errors (Table 7.2).

As can be seen from comparing the performance indices for all tested methods, the segmentation errors of the EEL surface (outer wall) were consistently larger than those of the luminal surface (inner wall) for all investigated segmentation methods (Table 7.2, Figs. 7.10 and 7.11). This should not be surprising, because segmenting the EEL surface is considerably more difficult than segmenting the lumen. Among others, the blood speckle dynamics, which helps resolve most ambiguities of the luminal segmentation is not relevant for resolving the EEL ambiguities, of which there are many. Out of these, calcified plaque shadows frequently cause most of the EEL segmentation inaccuracies due to a partial or complete lack of the ultrasound signal from the EEL interface. In such situations, the human experts use 3-D context as well as his/her anatomical knowledge and coronary remodeling experience to estimate the course of the EEL surface. Clearly, the automated approach such as the NA has only a limited chance to succeed in the cases of an almost complete lack of usable IVUS data depicting the EEL interface, and any human-based approach inevitably suffers from inter- and even intra-observer variability. Notably, this variability also affects the definition of the independent standard. As such, larger errors should be expected

for the EEL surface as they have been observed.

Compared to the previous dual-surface approach [39], notable improvements of the performance is attributed to the combination of the improvements to a) the underlying graph structure (both lumen-centering of the constructed graph resulting from the novel pre-segmentation step and incorporation of arc-based soft constraints allowing to model shape priors [105]) and b) the novel terms of the employed cost function dependent on edge as well as regional information from the IVUS images.

Our second-stage computer-aided approach to segmentation refinement enables the user to further improve the quality of IVUS segmentations and do it in a very time efficient manner even in difficult cases. In terms of segmentation performance, the two-stage NR approach, in which the refinement follows the automated NA stage, the segmentation improvements were again found statistically significant when compared to the tested automated approaches, both the previous automated PA approach and the new automated NA approach (Table 7.2) – with the signed error indices d_s and A_s computed from the luminal surfaces being an exception and not showing statistically significant improvement of the NR approach compared to the NA method alone. In other words, the NA approach is already providing highly accurate luminal surface as far as signed border positioning and area errors are concerned. Still, as can be seen from the boxplots in Figs. 7.10(a) and 7.11(a), the ranges of deviation around the median are smaller for NR compared to NA.

The average user interaction time required for utilizing our NR approach is more than 25-fold lower compared to fully manual editing and tracing that started

with the result of the PA method. One reason for this improvement is that the user interactions in the NR approach fully utilize the advantages of the simultaneous dual-surface segmentation approach. Thus, when correcting, e.g., the inner luminal surface, the outer EEL surface is adjusted automatically without a need to indicate the desired locations of the outer surface and vice versa (Fig. 7.6). Equally important, the proposed refinement approach is inherently three-dimensional. Thus, the resulting surfaces are smoother and less likely to show discontinuities that may be unavoidable when performing slice-by-slice manual segmentation editing. These improvements are practically important since they will contribute to making real-clinical-time IVUS segmentation a reality in the near future.

One potential disadvantage of the proposed refinement scheme might be that the shapes of any allowed refinement solutions (resulting surfaces) are limited by the hard shape constraints of the graph structure (e.g., hard smoothness constraints). These constraints cannot be changed easily on the fly without modifying the graph representation and recomputing the graph possibly from scratch. A solution of this limitation may be to perform an additional step of highly localized purely manual editing after completing the NR if/as needed. Also, in the current implementation, we assume that the vessel is formed by a single tube. Clearly, bifurcations violate this assumption. While the issue of bifurcations is not critical for catheter-based pullback images like IVUS, addressing this limitation can be a future research work.

Despite the demonstrated performance improvements in the 41 tested 3-D data sets, the study design is not free of several limitations, one of which was the way how

the independent standard was originally defined. As stated earlier, the PA method served for initial IVUS pullback segmentation and the resulting surfaces were used as a start for a manual editing process that yielded the independent standard as described in Section 7.4.B. As such, the independent standard is not fully independent from the PA method even if the manual tracing and editing required substantial changes of the surface definitions (compare panels PA and IS in Figs. 7.12, 7.13, and 7.14) and about 100 hours of expert editing. Arguing that the resulting independent standard is quite distant from the original PA segmentation would be well justified. More important and ultimately relevant to the presented study, there is no such real or perceived dependence between the NA or NR surfaces and the independent standard since the NA approach is based on a different graph construction, different interaction priors, and different cost functions. Consequently, even if there are some remaining dependencies between the independent standard and the PA segmentations, these would solely favor performance assessment of the PA method, which – however – fared worst among the three compared approaches. Clearly, any benefit that the definition of the independent standard may have provided to the PA method did not affect the ultimate assessment of the PA method as being significantly worse than the two other compared approaches. Therefore, this limitation of the study design can be regarded as insignificant with respect to the overall outcome of this work.

CHAPTER 8
CONCLUSIONS

In this thesis, two major problems in the context of OSF-based segmentation are addressed: a) generating an initial segmentation required for OSF-based segmentation and b) efficiently refining local errors in OSF segmentation results. Solutions for both issues that are provided by this thesis are discussed/summarized below.

Generating an intial segmentation can be challenging, especially for organs with atypical appearance due to disease (e.g., cancer). To address this issue, a novel robust ASM was presented in Chapter 2 (Aim 1). The robust ASM matching algorithm is specifically designed to take advantage of general purpose computation on graphics processing units (GPGPU), which reduces the execution time considerably. Our method is generally applicable to segmentation problems beyond lung segmentation. For example, the approach was utilized by Bauer *et al.* [8] for cerebella segmentation in 3D PET data.

Segmentation of lungs with large lung cancer regions is a non-trivial problem. In Chapter 3 (Aim 2), we present a new fully automated approach for segmentation of lungs with such high-density pathologies. Our method consists of two main processing steps. First, the proposed robust ASM matching method is utilized to roughly segment the outline of the lungs. The initial position of the robust ASM is found by means of a rib cage detection method. For detecting rib cage, centerlines of tubular structures in the chest image was extracted followed by rib centerline extraction using a two stage mean shift clustering algorithm. Second, an optimal surface find-

ing approach is utilized to further adapt the initial segmentation result to the lung. Left and right lungs are segmented individually. The robustness and effectiveness of our approach was demonstrated on 30 lung scans containing 20 normal lungs and 40 diseased lungs where conventional segmentation methods frequently fail to deliver usable results. Low segmentation errors were achieved in cases with and without high-density pathology compared to two clinically utilized methods, which demonstrates the robustness of our approach. Preliminary work investigating the applicability of our lung segmentation method to lungs with other kinds of diseases like idiopathic pulmonary fibrosis is promising. Also, an adapted version of our method was validated on 55 test scans provided by the LOLA11 challenge. On this diverse set of lung images, the proposed method showed comparable performance for the majority of data sets. Results on LOLA11 data also indicated that in cases where the lung shape widely deviates from the learned lung model segmentation can be challenging.

The optimal surface finding framework is a powerful approach and has demonstrated its utility in a number of medical image segmentation problems [75, 105, 133, 112, 128, 1, 92, 23]. However, when dealing with segmentation of structures/organs that are altered due to disease or other causes, designing a suitable cost function, which would work correctly for all possible situations is challenging and may be impossible since pathology augments image characteristics. As a consequence, segmentations can exhibit local errors. Such local inaccuracies or errors must be corrected prior to the subsequent quantitative analysis. For achieving full yield of medical imaging under all disease conditions, an efficient and inherently 3-dimensional approach

must be available in the workflow to facilitate efficient modification or refinement of the resulting segmentations. Clearly, the current state-of-the-art approach of slice-by-slice editing offers neither efficiency nor 3-D performance. In Chapter 4 (Aim 3), a segmentation refinement framework based on OSF and hybrid Desktop/VR user interface was presented. The basic idea behind this approach is that the user interacts directly with a segmentation algorithm to effectively correct potential errors in automatically generated OSF segmentation results. The hybrid user interface is based on stereoscopic visualization technology and advanced interaction techniques. The user interface allows natural and interactive manipulation of 3D surfaces. The user interface also supports interaction with 4D (3D + time) surfaces. The proposed visualization techniques facilitate the interactive segmentation refinement process. Our segmentation refinement method uses the same OSF-based graph structure that was utilized in preceding automated segmentation. Our method does not change the topology of the underlying graph structure. In Chapter 4, a simple point based tool was presented as an example.

In Chapter 5 (Aim 4.a), the proposed interactive refinement framework was adapted, utilized and validated in the context of lung segmentation in volumetric CT scans. Two refinement tools were presented. One is a generic lung refinement tool and the other is a specific tool to correct segmentation errors with leakage to trachea and main bronchus. The effectiveness of the approach was demonstrated on 18 lung scans with 30 volumetric ROIs which contain segmentation errors that are frequently appearing in automated OSF-based lung segmentation results. Tool response time

is one of the most important aspects leading to the acceptance or rejection of the approach by users expecting real-time interaction experience. Our approach demonstrated fast interaction responses (average 150 ms) and the average total interaction time required for reaching complete operator satisfaction was about 2 min. This time was mostly spent on human-controlled manipulation of the object to identify whether additional refinement was necessary and to approve the final segmentation result. This demonstrates the achievable benefits resulting from incorporating virtual reality in medical image analysis

In Chapter 6 (Aim 4.b), the proposed interactive OSF-based refinement approach was expanded to 4D lung segmentation refinement. The main goal of this chapter was to assess the feasibility of 4D segmentation refinement. The utilized 4D lung CT data consists of two volumetric scans imaged at inspiration and expiration. The approach includes two stages: 4D OSF-based automated segmentation, and segmentation refinement, if needed. In the automated segmentation stage, proposed RASM approach (Chapter 2) with automated model initialization (Chapter 3) was utilized to segment inspiration scans. Registration was applied to find a transformation from inspiration scan to expiration scan, which results in a pre-segmentation of the expiration scan based on the initially segmented inspiration scan. Both inspiration and expiration segmentations were then utilized to initialize 4D OSF-based segmentation. In the refinement stage, the hybrid user interface and two tools proposed in Chapter 5 were adapted to the 4D framework and utilized. The proof-of-concept study performed on four 4D lung scans demonstrated that 4D OSF-based refinement

is feasible and promising as a future research direction.

Segmentation of lumen and EEL surfaces in IVUS volumes is a difficult task. In Chapter 7 (Aim 4.c), a combination of automated segmentation and computer-aided segmentation refinement to facilitate this process was presented. The proposed new automated segmentation method (NA) delivered significantly better results compared to the work reported in [39]. The presented approach to segmentation refinement (NR) was found to be efficient, effective, and allowed the user to produce high quality segmentation results in cases of clinical quality images with a barrage of typical imaging artifacts. Overall, the average time required for producing IVUS segmentations suitable for further quantitative analysis was reduced from several hours to 6.5 min on average while demonstrating excellent segmentation accuracy. As such, the approach enables close-to-real-time IVUS segmentation, which is an important factor for enabling quantitative analysis of IVUS image data in routine clinical setting.

In general, a potential limitation of an OSF-based segmentation refinement approach is that the correct solution must be representable by the utilized graph structure, because only weights (costs) of the graph are modified during refinement. Consequently, no topology modification is possible. This issue can be addressed in two ways. One could develop segmentation refinement methods that allow the user to locally modify the graph structure. This would be demanding in terms of the required computational effort and may not be achievable in real-time. Another approach would be to revert to refinement tools described in [15, 11], which are based on an interactively user-modified deformable contour. While such an approach would

offer a larger degree of flexibility, it would likely require more user interaction steps and thus increase the overall refinement times.

Currently, the virtual reality hardware that was utilized in some of the experiments is not widely available in clinical practice. However, resulting from the advances driven by computer gaming and home entertainment, the accessibility of the VR environment components is rapidly increasing with a rapid decrease in the associated cost. Therefore, it is likely that such VR equipment will become commonplace in health care and will be utilized for a range of medical applications, similar to the VR method presented in this thesis.

REFERENCES

[1] M. D. Abràmoff, M. K. Garvin, and M. Sonka. Retinal imaging and image analysis. *IEEE Reviews in Biomedical Engineering*, 3:169–208, December 2010.

[2] A. M. Ali, A. El-Baz, and A. A. Farag. A novel framework for accurate lung segmentation using graph cuts. In *ISBI*, pages 908–911. IEEE, 2007.

[3] A. M. Ali and A. A. Farag. Automatic lung segmentation of volumetric low-dose CT scans using graph cuts. In *LNCS*, volume 5358, pages 258–267. ISVC, 2008.

[4] S. G. Armato and W. F. Sensakovic. Automated lung segmentation for thoracic CT. *Acad. Radiol.*, 11(9):1011–1021, 2004.

[5] W. A. Barret and E. N. Mortensen. Interactive live-wire boundary extraction. *Medical Image Analysis*, 1(4):331–341, 1997.

[6] C. Bauer, T. Pock, E. Sorantin, H. Bischof, and R. Beichel. Segmentation of interwoven 3D tubular tree structures utilizing shape priors and graph cuts. *Medical Image Analysis*, 14:172–184, 2010.

[7] C. Bauer, S. Sun, and R. Beichel. Avoiding mesh folding in 3D optimal surface segmentation. In *Proc. of 7th International Symposium on Visual Computing*, volume 6938, pages 214–223. LNCS, 2011.

[8] C. Bauer, S. Sun, W. Sun, J. Otis, A. Wallace, J. M. Buatti, B. J. Smith, J. J. Sunderland, M. M. Graham, M. Sonka, and R. R. Beichel. Automated measurement of uptake in cerebellum, liver, and aortic arch in full-body FDG PET/CT scans. *Medical Physics*, 39(6):3112–3123, 2012.

[9] R. Beichel, C. Bauer, A. Bornik, E. Sorantin, and H. Bischof. Liver segmentation in CT data: A segmentation refinement approach. In *Proc. of 3D Segmentation in The Clinic: A Grand Challenge*, pages 235–245. MICCAI Workshop, 2007.

[10] R. Beichel, H. Bischof, F. Leberl, and M. Sonka. Robust active appearance models and their application to medical image analysis. *IEEE Transactions on Medical Imaging*, 24(9):1151–1169, 2005.

[11] R. Beichel, A. Bornik, C. Bauer, and E. Sorantin. Liver segmentation in contrast enhanced CT data using graph cuts and interactive 3D segmentation refinement methods. *Medical Physics*, 39(3):1361–1373, February 2012.

[12] R. Beichel, S. Mitchell, E. Sorantin, F. Leberl, A. Goshtasby, and M. Sonka. Shape- and appearance-based segmentation of volumetric medical images. In *Proc. of International Conference on Image Processing*, volume 2, pages 589–592, 2001.

[13] F. L. Bookstein. Principal warps: Thin-plate splines and the decomposition of deformations. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 11(6):567–585, 1989.

[14] A. Bornik, R. Beichel, B. Reitinger, G. Gotschuli, E. Sorantin, F. W. Leberl, and M. Sonka. Computer aided liver surgery planning: An augmented reality approach. In *Proc. of SPIE (2003)*, volume 5029, 2003.

[15] A. Bornik, R. Beichel, and D. Schmalstieg. Interactive editing of segmented volumetric datasets in a hybrid 2D/3D virtual environment. In *Proceedings of the ACM symposium on Virtual reality software and technology*, pages 197–206. ACM, 2006.

[16] Y. Boykov and M.-P. Jolly. Interactive graph cuts for optimal boundary & region segmentation of objects in N-D images. In *Proc. of International Conference on Computer Vision*, volume 1, pages 105–112. IEEE, 2001.

[17] Y. Boykov and V. Kolmogorov. An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(9):1124–1137, September 2004.

[18] Y. Boykov and O. Veksler. Graph cuts in vision and graphics: Theories and applications. *Handbook of Mathematical Models in Computer Vision*, pages 79–96, 2006.

[19] G. C. Burdea and P. Coiffet, editors. *Virtual Reality Technology,Second Edition*. John Wesley & Sons, Hoboken, New Jersey, 2003.

[20] M. H. R. Cardinal, J. Meunier, G. Soulez, R. L. Maurice, E. Therasse, and G. Cloutier. Intravascular ultrasound image segmentation: a three-dimensional fast-marching method based on gray level distributions. *IEEE Transactions on Medical Imaging*, 25(5):590–601, 2006.

[21] M. H. R. Cardinal, G. Soulez, J. C. Tardif, J. Meunier, and G. Cloutier. Fast-marching segmentation of three-dimensional intravascular ultrasound images: A pre- and post-intervention study. *Med Phys*, 37:3633–3647, 2010.

[22] V. Caselles, F. Catté, T. Coll, and F. Dibos. A geometric model for active contours in image processing. *Numerische Mathematik*, 66(1):1–31, 1993.

[23] X. Chen, M. Niemeijer, L. Zhang, K. Lee, M. D. Abràmoff, and M. Sonka. 3D segmentation of fluid-associated abnormalities in retinal OCT: Probability constrained graph-search-graph-cut. *IEEE Transactions on Medical Imaging*, 31(8):1521–1531, August 2012.

[24] D. Comaniciu and P. Meer. Mean shift: A robust approach toward feature space analysis. *IEEE Trans. Pattern Anal. Machine Intell.*, 24(5):603–619, 2002.

[25] T. F. Cootes, C. Beeston, G. J. Edwards, and C. J. Taylor. A unified framework for atlas matching using active appearance models. In *Proc. of International Conference on Information Processing in Medical Imaging (IPMI)*, pages 322–333, 1999.

[26] T. F. Cootes, D. Cooper, C. J. Taylor, and J. Graham. Active shape models - their training and application. *Computer Vision and Image Understanding*, 61(1):38–59, 1995.

[27] T. F. Cootes, G. J. Edwards, and C. J. Taylor. Active appearance models. In H. Burkhardt and B. Neumann, editors, *Proc. of European Conference on Computer Vision*, volume 2, pages 484–498, 1998.

[28] T. F. Cootes, G. J. Edwards, and C. J. Taylor. Active appearance models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23(6):681–685, 2001.

[29] T. F. Cootes, A. Hill, C. J. Taylor, and J. Haslam. The use of active shape models for locating structures in medical images. *Image and Vision Computing*, 12(6):355–366, 1994.

[30] T. F. Cootes and C. J. Taylor. Active shape models - smart snakes. In *Proc. of the British Machine Vision Conference*, pages 266–275, 1992.

[31] T. F. Cootes and C. J. Taylor. Statistical models of appearance for medical image analysis and computer vision. In *Proc. SPIE*, volume 4322, pages 236–248, 2001.

[32] T. F. Cootes and C. J. Taylor. Statistical models of appearance for computer vision. Technical report, University of Manchester, Imaging Science and Biomedical Engineering, 2004.

[33] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein. *Introduction to algorithms*. MIT Press, 2009.

[34] D. Maleike D, M. Nolden, H. P. Meinzer, and I. Wolf. Interactive segmentation framework of the medical imaging interaction toolkit. *Computer Methods and Programs in Biomedicine*, 96(1):72–83, October 2009.

[35] P. Dalal, B. C. Munsell, S. Wang, J. Tang, K. Oliver, H. Ninomiya, X. Zhou, and H. Fujita. A fast 3D correspondence method for statistical shape modeling. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR'07)*, pages 1–8, 2007.

[36] H. Delingette. General object reconstruction based on simplex meshes. *International Journal of Computer Vision*, 32(2):111–142, 1999.

[37] J. Dijkstra, G. Koning, and J.H.C. Reiber. Quantitative measurements in ivus images. *Int J of Cardiac Imaging*, 15:513–522, 1999.

[38] G. Donato and S. Belongie. Approximate thin plate spline mappings. In *Proc. of the 7th European Conference on Computer Vision (ECCV2002)*, volume 2352, pages 21–31. LNCS, 2002.

[39] R. W. Downe, A. Wahle, T. Kovárník, H. Skalická, J. J. Lopez, J. Horák, and M. Sonka. Segmentation of intravascular ultrasound images using graph search and a novel cost function. In *2nd MICCAI Workshop on Computer Vision for Intravascular and Intracardiac Imaging*, 2008.

[40] I. L. Dryden and K. V. Mardia, editors. *Statistical Shape Analysis*. Wiley, New York, 1998.

[41] G. J. Edwards, C. J. Taylor, and T. F. Cootes. Interpreting face images using active appearance models. In *Proc. of IEEE International Conference on Automatic Face and Gesture Recognition*, pages 300–305, 1998.

[42] A. Frangi, D. Rueckert, J. Schnabel, and W. Niessen. Automatic construction of multiple-object three-dimensional statistical shape models: application to cardiac modeling. *IEEE Transactions on Medical Imaging*, 21(9):1151–1166, September 2002.

[43] A. F. Frangi, W. J. Niessen, K. L. Vincken, and M. A. Viergever. Multiscale vessel enhancement filtering. In *LNCS*, volume 1496, pages 130–137. MICCAI, 1998.

[44] M. K. Garvin, M. D. Abràmoff, R. Kardon, X. Wu, and M. Sonka. Intraretinal layer segmentation of macular optical coherence tomography images using optimal 3-D graph search. *IEEE Transactions on Medical Imaging*, 27(10):1495–1505, October 2008.

[45] G. Gerig, M. Jomier, and M. Chakos. Valmet: A new validation tool for assessing and improving 3D object segmentation. In *LNCS*, volume 2208, pages 516–523. MICCAI, 2001.

[46] G. Gerig, M. Styner, D. Jones, D. Weinberger, and J. Lieberman. Shape analysis of brain ventricles using SPHARM. In *Proc.of the IEEE Workshop on Mathematical Methods in Biomedical Image Analysis (MMBIA'01)*, pages 171–178, 2001.

[47] M. Gibson, D. Han, M. Sonka, and X. Wu. Maximum weight digital regions decomposable into digital star-shaped regions. In *Proc. of The 22nd International Symposium on Algorithms and Computation (ISAAC 2011)*, 2011.

[48] M. Haeker, X. Wu, M. D. Abràmoff, R. Kardon, and M. Sonka. Incorporation of regional information in optimal 3-D graph search with application for intraretinal layer segmentation of optical coherence tomography images. In *Proc. of Information Processing in Medical Imaging (IPMI)*, volume 4584, pages 607–618. LNCS, 2007.

[49] M. Harders, S. Wildermuth, and G. Szekély. New paradigms for interactive 3D volume segmentation. *Visualization and Computer Animation*, 13(5):85–95, 2002.

[50] M. Harders, S. Wildermuth, D. Weishaupt, and G. Szekély. Improving virtual endoscopy for the intestinal tract. In *LNCS*, volume 2489, pages 20–27. MICCAI, 2002.

[51] T. Heimann and H.P. Meinzer. Statistical shape models for 3D medical image segmentation: A review. *Medical Image Analysis*, 13:543–563, 2009.

[52] T. Heimann, S. Münzing, H.-P. Meinzer, and I. Wolf. A shape-guided deformable model with evolutionary algorithm initialization for 3D soft tissue segmentation. In *LNCS*, volume 4584, pages 1–12. IPMI (2007), 2007.

[53] T. Heimann, I. Wolf, and H. P. Mein. Active shape models for a fully automated 3D segmentation of the liver-an evaluation on clinical data. In *MICCAI*, volume 4191, pages 41–48, 2006.

[54] T. Heimann, I. Wolf, T.G. Williams, and H.-P. Meinzer. 3D active shape models using gradient descent optimization of description length. In *In Proc. IPMI*, volume 3565, pages 566–577. Springer, Heidelberg, 2005.

[55] L. F. Hodges. Tutorial: time-multiplexed stereoscopic computer graphics. *Computer Graphics and Applications, IEEE*, 12(2):20–30, March 1992.

[56] E. A. Hoffman and E. L. Ritman. Effect of body orientation on regional lung expansion in dog and sloth. *J. Appl. Physiol.*, 59(2):481–491, 1985.

[57] S. Hu, E. A. Hoffman, and J. M. Reinhardt. Automatic lung segmentation for accurate quantitation of volumetric X-ray CT image. *IEEE Transactions On Medical Imaging*, 20(6):490–498, 2001.

[58] P. Hua. Segmentation of lung tissue in CT images with disease and pathology. Master's thesis, The University of Iowa, 2010.

[59] P. Hua, Q. Song, M. Sonka, E. A. Hoffman, and J. M. Reinhardt. Segmentation of pathological and diseased lung tissue in CT images using a graph-search algorithm. In *ISBI*, pages 2072–2075. IEEE, 2011.

[60] R. M. Taylor II, T. C. Hudson, A. Seeger, H. Weber, J. Juliano, and A. T. Helser. VRPN: A device-independent, network-transparent VR peripheral system. In *Proc. of the ACM Symposium on Virtual Reality Software and Technology 2001*, pages 55–61, 2001.

[61] A. K. Jain, Y. Zhong, and S. Lakshmanan. Object matching using deformable templates. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 18(3):267–278, 1996.

[62] M. Kass, A. Witkin, and D. Terzopoulos. Snakes: Active contour models. *International Journal of Computer Vision*, 1:321–331, 1987.

[63] T. Kitasaka, K. Mori, J. Hasegawa, and J. Toriwaki. Lung area extraction from 3D chest X-ray CT images using a shape model generated by a variable Bézier surface. *Systems and Computers in Japan*, 34(4):60–71, 2003.

[64] S. Klein, M. Staring, K. Murphy, M. Viergever, and J. Pluim. elastix: a toolbox for intensity-based medical image registration. *IEEE Transactions on Medical Imaging*, 29(1):196–205, January 2010.

[65] J.D. Klingensmith, A. Nair, B.D. Kuban, and D.G. Vince. Segmentation of three-dimensional intravascular ultrasound images using spectral analysis and a dual active surface model. In *Ultrasonics Symposium, 2004 IEEE*, volume 3, pages 1765 – 1768 Vol.3, 2004.

[66] J.D. Klingensmith, R. Shekhar, and D.G. Vince. Evaluation of three-dimensional segmentation algorithms for the identification of luminal and medial-adventitial borders in intravascular ultrasound images. *Medical Imaging, IEEE Transactions on*, 19:996 –1011, 2000.

[67] Jan J. Koenderink and Andrea J. van Doorn. Surface shape and curvature scales. *Image Vision Comput.*, 10:557–565, October 1992.

[68] G. Koning, J. Dijkstra, C. Birgelen, J.C. Tuinenburg, J. Brunette, J.C. Tardiff, P.V. Oemrawsingh, C. Sieling, S. Melsa, and Reiber J.H.C. Advanced contour detection for three-dimensional intracoronary ultrasound: A validation – in-vitro and in-vivo. *Int J of Cardiac Imaging*, pages 235–248, 2002.

[69] P. Korfiatis, A. Karahaliou, S. Skiadopoulos, and L. Costaridou. Texture classification-based segmentation of lung affected by interstitial pneumonia in high-resolution CT. *Medical Physics*, 35(12):5290–5302, November 2008.

[70] R. Korn, J. Kim, G. Schmidt, and G. Binnig. Description of a fully automatic lung segmentation algorithm based on the cognition network technology. In *Proc. of the Fourth International Workshop on Pulmonary Image Analysis*, pages 233–240, 2011.

[71] B. Lassen, J.-M. Kuhnigk, M. Schmidt, S. Krass, and H.-O. Peitgen. Lung and lung lobe segmentation methods at Fraunhofer MEVIS. In *Proc. of the Fourth International Workshop on Pulmonary Image Analysis*, pages 185–199, 2011.

[72] J. K. Leader, B. Zheng, R. M. Rogers, F. C. Sciurba, A. Perez, B. E. Chapman, S. Patel, C. R. Fuhrman, and D. Gur. Automated lung segmentation in X-ray computed tomography. *Acad. Radiol.*, 10(11):1224–1236, 2003.

[73] Karim Lekadir, Robert Merrifield, and Guang zhong Yang. Outlier detection and handling for robust 3-D active shape models search. *IEEE Transactions on Medical Imaging*, 26:212–222, 2007.

[74] B. Li and J. M. Reinhardt. Automatic generation of object shape models and their application to tomographic image segmentation. In *Proc. SPIE (Medical Imaging)*, volume 4322, pages 311–322, 2001.

[75] K. Li, X. Wu, D.Z. Chen, and M. Sonka. Optimal surface segmentation in volumetric images - a graph-theoretic approach. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28(1):119–134, 2006.

[76] P. Lo. *Segmentation of Lung Structures in CT*. PhD thesis, University of Copenhagen, 2010.

[77] P. Lo, J. Goldin, D. Oria, A. Banola, and M. Brown. Historic automated lung segmentation method: Performance on LOLA11 data set. In *Proc. of the Fourth International Workshop on Pulmonary Image Analysis*, pages 257–260, 2011.

[78] W. E. Lorensen and H. E. Cline. Marching cubes: A high resolution 3D surface construction algorithm. *ACM SIGGRAPH Computer Graphics*, 21(4):163–169, July 1987.

[79] E.G. Mendizabal-Ruiz, G. Biros, and I.A. Kakadiaris. An inverse scattering algorithm for the segmentation of the luminal border on intravascular ultrasound data. *Med Image Comput Comput Assist Interv*, 12:885–892, 2009.

[80] S. C. Mitchell, H. G. Bosch, B. P. F. Lelieveldt, R. J. van der Geest, J. H. C. Reiber, and M. Sonka. 3-D active appearance models: Segmentation of cardiac MR and ultrasound images. *IEEE Transactions on Medical Imaging*, 21(9):1167–1178, 2002.

[81] S. C. Mitchell, B. P. F. Lelieveldt, R. J. van der Geest, H. G. Bosch, J. H. C. Reiber, and M. Sonka. Multistage hybrid active appearance models matching: Segmentation of left and right ventricles in cardiac MR images. *IEEE Transactions on Medical Imaging*, 20(5):415–423, 2001.

[82] S. C. Mitchell, B. P. F. Lelieveldt, R. J. van der Geest, H. G. Bosch, J. H. C. Reiber, and M. Sonka. Time-continuous segmentation of cardiac MR image sequences using active appearance motion models. In *Proc. of SPIE (Medical Imaging - Image Processing)*, volume 4322, pages 249–256, 2001.

[83] S. C. Mitchell, B. P. F. Lelieveldt, R. J. van der Geest, J. Schaap, J. H. C. Reiber, and M. Sonka. Segmentation of cardiac MR images: an active appearance model approach. In *Proc. of SPIE (Medical Imaging - Image Processing)*, volume 3979, pages 224–234, 2000.

[84] A. Montillo. Context selective decision forests and their application to lung segmentation in CT images. In *Proc. of the Fourth International Workshop on Pulmonary Image Analysis*, pages 201–212, 2011.

[85] M. Sonka N. Duta. Segmentation and interpretation of MR brain images: An improved active shape model. *IEEE Transactions on Medical Imaging*, 17(6):1049–1062, 1998.

[86] H. Nguyen, editor. *GPU Gems 3*. Addison-Wesley, 2007.

[87] D. L. Pham, C. Xu, and J. L. Prince. A survey of current methods in medical image segmentation. *Annual Review of Biomedical Engineering*, 2:315–338, 2000.

[88] R. Pinho, V. Delmon, J. Vandemeulebroucke, S. Rit, and D. Sarrut. Keuhkot: A method for lung segmentation. In *Proc. of the Fourth International Workshop on Pulmonary Image Analysis*, pages 225–232, 2011.

[89] M. N. Prasad, M. S. Brown, S. Ahmad, F. Abtin, J. Allen, I. da Costa, H. J. Kim, M. F. McNitt-Gray, and J. G. Goldin. Automatic segmentation of lung parenchyma in the presence of diseases based on curvature of ribs. *Academic Radiology*, 15(9):1173 – 1180, 2008.

[90] J. Pu, D. S. Paik, X. Meng, J. E. Roos, and G. D. Rubin. Shape "break-and-repair" strategy and its application to automated medical image segmentation. *IEEE Transactions on Visualization and Computer Graphics*, 17:115–124, 2011.

[91] J. Pu, J. Roos, C. A. Yi, S. Napel, G. D. Rubin, and D. S. Paik. Adaptive border marching algorithm: Automatic lung segmentation on chest CT images. *Computerized Medical Imaging and Graphics*, 32(6):452 – 462, 2008.

[92] G. Quellec, K. Lee, M. Dolejsi, M. K. Garvin, M. D. Abràmoff, and M. Sonka. Three-dimensional analysis of retinal layer texture: Identification of fluid-filled regions in SD-OCT of the macula. *IEEE Transactions on Medical Imaging*, 29(6):1321–1330, June 2010.

[93] G. Reitmayr and D. Schmalstieg. An open software architecture for virtual reality interaction. In *Proc. of the ACM symposium on Virtual reality software and technology*, pages 47–54. ACM, 2001.

[94] M. Rogers and J. Graham. Robust active shape model search. In *In Proceedings of the European Conference on Computer Vision*, pages 517–530. Springer, 2002.

[95] J. P. Rolland, L. Davis, and Y. Baillot. A survey of tracking technologies for virtual environments. In W. Barfield and T. Caudell, editors, *Fundamentals of Wearable Computers and Augmented Reality*, pages 67–112. Eds.Mahwah, NJ Lawrence Erlbaum, 2001.

[96] J. C. Ross, R. S. Estépar, A. Díaz, C. F. Westin, R. Kikinis, E. K. Silverman, and G. R. Washko. Lung extraction, lobe segmentation and hierarchical region assessment for quantitative analysis on high resolution computed tomography images. In *Proc. of MICCAI (1)*, volume 12, pages 690–698, 2009.

[97] Z. Salah, J. Orman, and D. Bartz. Live-wire revisited. In *Proc. of Workshop Bildverarbeitung für die Medizin*, pages 93–97, 2005.

[98] A. Schenk, G. Prause, and H.-O. Peitgen. Efficient semiautomatic segmentation of 3D objects in medical images. In *Proc. of MICCAI (2000)*, pages 186–195, 2000.

[99] D. Schmalstieg, A. Fuhrmann, G. Hesina, Z. Szalavári, L. M. Encarnaçäo, M. Gervautz, and W. Purgathofer. The studierstube augmented reality project. *Presence: Teleoperators and Virtual Environments*, 11(1):33–54, February 2002.

[100] T. Schwarz, T. Heimann, R. Tetzlaff, A. M. Rau, I. Wolf, and H. Meinzer. Interactive surface correction for 3D shape-based segmentation. In *Proc. of SPIE Medical Imaging 2008: Image Processing*, volume 6914, 2008.

[101] S. Senger. Visualizing and segmenting large volumetric data sets. *IEEE Computer Graphics and Applications*, pages 32–37, 1999.

[102] R. Shojaii, J. Alirezaie, and P. Babyn. Automatic lung segmentation in CT images using watershed transform. In *LNCS*, pages 1270–1273. IEEE International Conference On Image Processing, 2005.

[103] A. Silva, J. S. Silva, B. S. Santos, and C. Ferreira. Fast pulmonary contour extraction in X-ray CT images: a methodology and quality assessment. In *Proc. SPIE (Medical Imaging)*, volume 4321, pages 216–224, 2001.

[104] I. Sluimer, M. Prokop, and B. van Ginneken. Toward automated segmentation of the pathological lung in CT. *IEEE Transactions On Medical Imaging*, 24(8):1025–1038, 2005.

[105] Q. Song, X. Wu, Y. Liu, M. Smith, J. Bautti, and M. Sonka. Optimal graph search segmentation using arc-weighted graph for simultaneous surface detection of bladder and prostate. In *Proc of Medical Image Computing and Computer Assisted Intervention (MICCAI)*, volume 5762, page 827835. LNCS, 2009.

[106] M. Sonka and J. M. Fitzpatrick, editors. *Handbook of Medical Imaging, Volume 2. Medical Image Processing and Analysis*. SPIE, 2000.

[107] M. Sonka, V. Hlavac, and R. Boyle. *Image Processing, Analysis and Machine Vision*. Thomson Learning, Toronto, Canada, third edition, 2007.

[108] M. Sonka, X. Zhang, M. Siebes, M. S. Bissing, S. C. DeJong, S. M. Collins, and C. R. McKay. Segmentation of intravascular ultrasound images: A knowledge-based approach. *IEEE Transactions on Medical Imaging*, 14:719–732, 1995.

[109] M. Staring, S. Klein, J. H. C. Reiber, W. J. Niessen, and B. C. Stoel. Pulmonary image registration with elastix using a standard intensity-based algorithm. In *Proc. of Grand Challenges in Medical Image Analysiss*, pages 73–79, 2010.

[110] M. Storer, P.M. Roth, M. Urschler, and H. Bischof. Fast-robust PCA. In *16th Scandinavian Conference on Image Analysis*, pages 430–439, 2009.

[111] S. Sun, C. Bauer, and R. Beichel. Robust active shape model based lung segmentation in CT scans. In *Proc. of the Fourth International Workshop on Pulmonary Image Analysis*, pages 213–223, 2011.

[112] S. Sun, C. Bauer, and R. Beichel. Automated 3D segmentation of lungs with lung cancer in CT data using a novel robust active shape model approach. *IEEE Transactions on Medical Imaging*, 31(2):449–460, February 2012.

[113] P. Therasse, S. Arbuck, E. Eisenhauer, J. Wanders, R. S. Kaplan, L. Rubinstein, J. Verweij, M. Van Glabbeke, A. T. van Oosterom, M. C. Christian, and S. G. Gwyther. New guidelines to evaluate the response to treatment in solid tumors. *Journal of the National Cancer Institute*, 92(3):205–216, 2000.

[114] G. Unal, S. Bucher, S. Carlier, G. Slabaugh, T. Fang, and K. Tanaka. Shape-driven segmentation of the arterial wall in intravascular ultrasound images. *IEEE Trans Inf Technol Biomed*, 12:335–347, 2008.

[115] B. van Ginneken, A. F. Frangi, J. J. Staal, B. M. ter Haar Romeny, and M. A. Viergever. Active shape model segmentation with optimal features. *IEEE Transactions on Medical Imaging*, 21(8):924–933, 2002.

[116] E. M. van Rikxoort, B. de Hoop, M. A. Viergever, M. Prokop, and B. van Ginneken. Automatic lung segmentation from thoracic computed tomography scans using a hybrid approach with error detection. *Medical Physics*, 36:2934–2947, 2009.

[117] E. M. van Rikxoort and B. van Ginneken. Automatic segmentation of the lungs and lobes from thoracic CT scans. In *Proc. of the Fourth International Workshop on Pulmonary Image Analysis*, pages 261–268, 2011.

[118] V. Vezhnevets and V. Konouchine. "Grow-cut"- interactive multi-label N-D image. In *Proc. of Graphicon*, pages 150–156, 2005.

[119] R. F. Wagner, S.W. Smith, J. M. Sandrik, and H. Lopez. Statistics of speckle in ultrasound B-scans. *IEEE Transactions on Sonics and Ultrasonics*, 30(3):156–163, 1983.

[120] A. Wahle, J. J. Lopez, M. E. Olszewski, S. C. Vigmostad, K. B. Chandran, J. D. Rossen, and M. Sonka. Plaue development, vessel curvature, and wall shear stress in coronary arteries assessed by X-ray angiography and intravascular ultrasound. *Medical Image Analysis*, 10(4):615–631, August 2006.

[121] J. Wang, Q. Li, and F. Li. Automated segmentation of lungs with severe interstitial lung disease in CT. *Medical Physics*, 36(10):4592 – 4599, 2009.

[122] Y. Wang and R. Beichel. Graph-based segmentation of lymph nodes in CT data. In *Proc. of ISVC*, volume 6454, pages 312–321. LNCS, 2010.

[123] O. Weinheimer, T. Achenbach, C. P. Heussel, and C. Düber. Automatic lung segmentation in MDCT images. In *Proc. of the Fourth International Workshop on Pulmonary Image Analysis*, pages 241–255, 2011.

[124] X. Wu and D. Z. Chen. Optimal net surface problems with applications. In *Proc. of the 29th International Colloquium on Automata, Languages and Programming (ICALP)*, volume 2380, page 10291042. LNCS, 2002.

[125] C. Xu and J. L. Prince. Snakes, shapes, and gradient vector flow. *Image Processing, IEEE Transactions on*, 7(3):359 – 369, March 1998.

[126] W. Yin, D. Goldfarb, and S. Osher. The total variation regularized L1 model for multiscale decomposition. *Multiscale Modeling and Simulation*, 6(1):190–211, 2007.

[127] Y. Yin, Q. Song, and M. Sonka. Electric field theory motivated graph construction for optimal medical image segmentation. In *Proc. of Graph-Based Representations in Pattern Recognition, 7th IAPR-TC-15 International Workshop (GbRPR)*, volume 5534, pages 334–342. LNCS, 2009.

[128] Y. Yin, X. Zhang, R. Williams, X. Wu, D. D. Anderson, and M. Sonka. LOGISMOS-layered optimal graph image segmentation of multiple objects and surfaces: Cartilage segmentation in the knee joint. *IEEE Transactions on Medical Imaging*, 29(12):2023–2037, December 2010.

[129] H. Zhang, A. K. Abioseb, D. N. Campbellb, M. Sonka, J. B. Martinsb, and A. Wahle. Left-ventricle segmentation in real-time 3D echocardiography using a hybrid active shape model and optimal graph search approach. In *Proc. of SPIE Medical Imaging, Image Processing*, pages 7626–7647, 2010.

[130] H. Zhang, A. Wahle, R. K. Johnson, T. D. Scholz, and M. Sonka. 4-D cardiac MR image analysis: Left and right ventricular morphology and function. *IEEE Transactions on Medical Imaging*, 29(2):350–364, 2010.

[131] L. Zhang, E.A. Hoffman, and J.M. Reinhardt. Atlas-driven lung lobe segmentation in volumetric X-Ray CT images. *IEEE Transactions on Medical Imaging*, 25(1):1–16, 2006.

[132] X. Zhang, C. R McKay, and M. Sonka. Image segmentation and tissue characterization in intravascular ultrasound. *IEEE Transactions on Medical Imaging*, 17:889–899, 1998.

[133] X. Zhang, J. Tian, K. Deng, Y. Wu, and X. Li. Automatic liver segmentation using a statistical shape model with optimal surface detection. *IEEE Transactions on Biomedical Engineering*, 57(10):2622–2626, September 2010.

[134] F. Zhao, H. Zhang, A. Wahle, T. D. Scholz, and M. Sonka. Automated 4D segmentation of aortic magnetic resonance images. In *Proc. of British Machine Vision Conference (BMVA)*, volume 1, pages 247–256, 2006.

[135] F. Zhou, H. B.-L. Duh, and M. Billinghurst. Trends in augmented reality tracking, interaction and display: A review of ten years of ISMAR. In *Mixed and Augmented Reality, 2008. ISMAR 2008. 7th IEEE/ACM International Symposium on*, pages 193 – 202, 2008.